A new side-channel attack on RSA prime numbers generation.

Isac Iulian-George[1], **Emil Simion** [2]

[1] Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi,
Email:iuliangeorge2017@gmail.com
[2]Politehnica University of Bucharest, Email: emil.simion@upb.ro

Abstract

The purpose of this article is to present,illustrate and to put in evidence a new side-channel attack on RSA cryptosystem based on the generation of prime numbers. The vulnerability of the cryptosystem is spotted during the execution of the key generation step.The probability of success of the attack is around 10-15% in the case of realistic parameters.

Keywords: *Side-Channel Attack,RSA,Prime Generation,prime candidate*

1. Introduction

Since the first practical use of RSA cryptosystem, many researchers and dedicated people tried to find vulnerabilities of cryptosystem based on properties of the elements mathematical defined. At the time of the its formulation RSA proved to be a safe and secure cryptosystem,its security being based on one of the hardest problem in computer science,the prime factorization of very large numbers. Over time many attacks which exploits the vulnerabilities of RSA were implemented. The author of [1] put in evidence a series of attacks of RSA of different types like integer factorization attacks, discrete logarithm attacks, exponentiation attacks or side-channel aspects. Our paper focus on a side-channel attack during the execution of key generation step. Also other side-channel attacks on RSA are described in [2, 3, 4, 5] which proves the fact that side-channel attacks have a long tradition.

Those attacks have as an objective the discovery of private key d through the RSA exponentiation procedure. Another important aspect is that the idea to develop a side-channel attack which concentrates on the generation of the prime numbers p,q, the private key d and the public key e (e,n=pq) can be found also in other papers [6, 7, 8].

The main difference from the side-channel attacks on RSA exponentiation with the secret key d consist in the situation of a potential attacker which seems to be much harder because the key pair is generated just once without using any known or chosen external input. [8]

The power attack proposed in this paper exploits a straight-forward implementation for generating prime numbers where each number is incremented by 2. In order to generate a prime candidate c, a set of trial divisions must be passed entirely and also a number of Miller-Rabin primality tests . An assumption is made regarding the number of trial divisions for each prime candidate, which yields information on p and q,namely $p(\mathrm{mod}\ s)$ and $q(\mathrm{mod}\ s)$ where s is a product of small primes. The attack finish with success if s is big enough.The probability of success of the attack is around 10-15% in the case of realistic parameters [8].

Our goal is to present a side-channel attack on key generation step which use prime generation methods. On the other hand we want to notify that the community that the key generation step of RSA plays an important role in terms of security and side-channel analysis especially in case of insecure environments for execution.

The paper is organized as follows: In Section 2 a closer look is took at the RSA prime generation step. In Section 3 the attack alongside its theoretical background is explained. In 4 we present the mathematical background needed for understanding the $\mathbb{NP}$-hard problem on lattices,SVP. Results from simulations and conclusions from the power analysis of an exemplary implementation on a standard microcontroller are presented in 5. The ending of the paper is marked by countermeasures and conclusions.

2. Methods of generation prime candidates.

The step of key generation for RSA is constituted by the generation of prime candidates. In this section our focus is concentrated on some generation methods of prime candidates also making some remarks regarding the side-channel leakage that can appear.

Definition 1 *For any $k \in \mathbb{N}$ a k-bit integer is an integer that is contained in $[2^{k-1}, 2^k)$ and its binary representation has the length equal with k. Let be $m \geq 2$ we define $Z_m := \{0, 1..m-1\}$ and also $Z_m^* := \{a \in Z_m | gcd(a, m) = 1\}$.*

Algorithm 1 *Prime generation algorithm.*

Input: $k \in \mathbb{N}$
Output: *v - k-bit prime integer*
$v \leftarrow RandomOddIntegerFromInterval([2^{k-1}, 2^k))$;
if $isPrime(v)$ **then**
  return v
else
  rerun algorithm
end

Algorithm 1 can be used in order to generate the p and q from the key generation step . Making trial divisions by small primes from a set $C := \{c_2, ...c_N\}$ and to those candidates that are not multiple of any element from the set C , the Miller-Rabin primality test [9] is applied several times. Taking in consideration the fact that this algorithm is running until it founds a prime number being making numerous calls to RNG (random number generator) ,which may be time and resource consuming for many applications the authors of [8] propose a new algorithm below which resolve this problem requiring just one k-bit random number per generated prime. [10]

Assumptions 1
1. *The implementation of the algorithm will be made on a target device.*
2. *It can be identified by a potential attacker which trial division failed and for which possible prime candidate and also the number of Miller-Rabin tests made. [8]*

Observations .

1. The Miller-Rabin test and trial division algorithm prove to be protected against a side-channel attack meaning that there are no leakage information on the dividend of the trial division.
2. The assumption that is made about the attack is fulfilled if the beginning or the end of each trial division.
3. There are more efficient algorithms in terms of time and space complexity that solve our problem of prime generation candidates. [6, 11]

Algorithm 2 *Prime generation algorithm*

Input: $k, t \in \mathbb{N}$
Output: v - *k-bit prime integer*
$v \leftarrow RandomOddIntegerFromInterval([2^{k-1}, 2^k))$;
$i \leftarrow 2$;
while $i \leq N$ **do**
  if $v\%c_i == 0$ **then**
    $v \leftarrow v + 2$;
    GOTO step 2;
  end
  $i \leftarrow i + 1$;
end
$b \leftarrow 1$;
while $m \leq t$ **do**
  if *not MillerRabbinTest(v)* **then**
    $v \leftarrow v + 2$;
    GOTO step 2;
  else
    $b \leftarrow b + 1$;
  end
end

3. The attack

3.1. Description attack

Let's consider $v_m := v_0 + 2m$ as the prime number generated by Algorithm 2. Also, let's be $v_j = v_0 + 2j$,$v_j \equiv 0\ (mod\ c_i)$ the value for which the Algorithm 2 returned to step 2 after the trial division by c_i. [8] Taking in consideration it can be concluded that :

$$p = v_m = v_j + 2(m - j) \equiv 2(m - j)\ (\ mod\ c_i\ ) \tag{1}$$

$$A_p := \{2\} \cup \{c \in C |\ \text{division by c caused a return to step 2 for at least one } v_j\ \} \tag{2}$$

The expression 'caused a return to step 2' doesn't mean the fact that at least one v_j is divided by c because it may happen that c can happen to be in $C \setminus A_p$ when divides v_j but the loop finish earlier due to other divisor of v_j. [8] By combining all the formulas of type (1) or by applying Chinese Remainder Theorem (CRT) it can obtain :

$$w_p \equiv p\ (mod\ s_p) \text{ where } a_p \text{ is known value and } s_p = \prod_{b \in A_p} b \tag{3}$$

Knowing that $n = pq$ we have :

$$w_q :\equiv q \equiv w_p^{-1} n (mod\ s_p) \tag{4}$$

By examining the method of generating q, it can derive :

$$z_q \equiv q (mod\ s_q)\ and\ z_p \equiv p \equiv z_q^{-1} n (mod\ s_q) \tag{5}$$

Note : The sets A_p and A_q are defined in a analogously way and also the values s_q and s_p. From the equations (3),(4),(5) we get the CRT integers h_p,h_q and s :

$$s := lcm(s_p, s_q)\ , h_p \equiv p (mod\ s)\ , h_q \equiv q (mod\ s) \text{ with } 0 \leq h_p, h_q \leq s \tag{6}$$

And using (6) => :

$$p = sx_p + h_p \text{ and } q = sy_q + h_q \text{ the pair } (x_p, y_q) \in \mathbb{N}^2 \text{ [8]} \tag{7}$$

Knowing that h_p, h_q, s can be determined the problem of finding the pair (p, q) it comes down to the problem of finding the roots of a bivariate polynomial over $\mathbb{Z}$.[8]

Lemma 1 *1. The pair (x_p, y_q) is the solution of the polynomial*

$$f : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}, \quad f(x,y) = h_p y + h_q x - t + sxy \ \ where \ \ t = (n - h_p h_q)/s \tag{8}$$

2. In a particular mode

$$t \in \mathbb{N}, \ \ f \ is \ irreducible \ over \ \mathbb{Z} \ , \ and \tag{9}$$

$$0 < x_p, y_q < max\{\frac{p}{s}, \frac{q}{s}\} < \frac{2^k}{s} \tag{10}$$

[8]

Proof. In order to proof the first point of this lemma we can use (9),in order to check if (x_p, y_q) is the solution, $0 = pq - n = (sx_p + h_p)(sy_q + h_q) - n = s^2 x_p y_q + sh_p y_q + sh_q x_p - (n - h_p h_q)$. The proof for the other point we observe that $n \equiv h_p h_q \ (mod\ s) \Rightarrow t \in \mathbb{Z}$. Seeing that $h_p \equiv p \not\equiv 0 \ (mod\ c_j)$ and $h_q \equiv q \not\equiv 0 \ (mod\ c_j) \Rightarrow \forall c_j - divisor\ prime\ of\ s$ we observe that $gcd(s, h_p) = gcd(s, h_q) = 1$ and $gcd(s, h_p, h_q, t) = 1$. We rewrite $f(x,y) = (ax + by + c)(dx + ey + f)$ where $a, b, c, d, e, f \in \mathbb{Z}$ after comparing the coefficients $\Rightarrow (a = e = 0)$ or $(b = d = 0)$. Applying the greatest common divisor properties $\Rightarrow gcd(bd, bf) = gcd(bd, cd) = 1$ and $gcd(ae, af) = gcd(ae, ce) = 1$, $b = d = a = e = 1$ which leads to contradiction.

The problem of finding solutions for polynomials with multiple variables over $\mathbb{Z}$ proves to be difficult. Despite the fact that it seems to be a difficult problem a well-known solving exists which succeeds to find solutions using the LLL-algorithm [12] which reduces this problem to SVP. [13]

Theorem 1 *(1) Let be $p(x,y)$ - irreducible polynomial with two variables over $\mathbb{Z}$ and γ the maximum degree in each variable separately. Let be X, Y the upper bounds for $|x_0|$ and $|y_0|$ where (x_0, y_0) - solution. Define $v(x,y) := p(xX, yY)$ and W the absolute value of biggest coefficient of w.*

If $XY < W^{2/(3\gamma)} \Rightarrow$ exists an algorithm which runs in polynomial time with the purpose of finding all the pairs (x_0, y_0) with $p(x_0, y_0) = 0, |x_0| < X$ and $|y_0| < Y$.

(2) Let be $p, q - k$-bit primes and $n = pq$. If $s, c_p \in \mathbb{Z}$ with $s \geq 2^{\frac{k}{2}}$ and $c_p \equiv p \ (mod\ s) \Rightarrow$ the factorization of n can be determined in polynomial time. [8]

Proof. (1) The demonstration can be found in [14], Corollary 2.
(ii) We apply the first point of theorem to the polynomial from Lemma (1). Using (10) we obtain $0 < x_p < X := 2^k/s$ and $0 < y_q < Y := 2^k/s$. We define $g(x,y) = f(xX, yY)$ and W - the absolute value of biggest coefficient of g.

Knowing $W \geq sXY = \frac{2^{2k}}{s}$, and for $s > 2^{\frac{k}{2}}$ we conclude $XY = (\frac{2^k}{s})^2 < (\frac{2^{2k}}{s})^{\frac{2}{3}} \leq W^{\frac{2}{3}}$.

From the fact that $2^k < s$ the first inequality is obtained and from the fact that γ is one in each variable by (1) we can find (x_p, y_q) in $O(k)$. [8]

4. Shortest Vector Problem

Definition 2 *$\mathrm{L} \subset \mathbb{R}^n$ is called lattice if it is the set of all integer linear combinations of some linearly independent basis $a_1, a_2, a_3, ...a_n \in \mathbb{R}^n$.*

$$\mathrm{L} := \{\sum_{i=1}^{n} z_i a_i \ \ :, z_i \in \mathbb{Z}\}$$

Definition 3 *SVP definition contains in the fact that given a basis $A = (a_1, ..., a_n)$ of a lattice L, find the shortest non-zero vector in L, that is, a vector $s \in \mathrm{L}$ such that $||s|| \leq \lambda_1(\mathrm{L})$,where $\lambda_1(\mathrm{L})$ is defined as the minimum of maximum of $v_1, v_1 \in \mathrm{L} -$ liniar independent vector .*

Other than the algorithm LLL ([12]), there are numerous papers that focus on finding an solution to SVP based on different aspects of it. The work papers developed by Pohst ([15]), Kannan([16]) , and Fincke–Pohst([17]) focuses on enumeration. The way an enumeration algorithm works is by taking as input a value called *enumeration radius* $R > 0$ alongside a basis $A = (a_1, ..., a_n)$ of a lattice L, and trying to outputs all non-zero vectors $s \in$ L such that $s \leq R$.

Also the authors of [18] put in evidence an algorithm capable of solving SVP which rely on the mathematical properties of the Gaussian distribution. New directions of research for solving this $\mathbb{NP}$-hard problem can be found in [19, 20, 21] which are based on lattices concepts.

5. Empirical and experimental Results

Theorem (1) states that in case of a value s sufficiently large the basic attack proves to be successful. Also by $\log_2(s)$ we define the number of bits used by the memory for storing the value, also we notice that in most cases $s \leq \prod_{r \in C}$. Another aspect shown by the results of the experiments put in the evidence that the bitsize of s is able to vary considerably for different k-bit starting candidates v_0 for Algorithm 2.One mandatory condition for Theorem (1) is that $\log_2(s) > \frac{k}{2}$ (a practical view) or even better $\log_2(s) > G$,which allows the running of LLL-algorithm with normal lattice dimension.

The authors of [8] implemented Algorithm 2 using a number of $t = 20$ Miller-Rabin tests in [22] and ran the RSA-generation process 10000 for each pair (k, N). More details about the results and statistics can be consulted in [8].

From the hardware perspective the fact that the power consumption reveals the number of trial divisions of the prime candidates $v_0 = v, v_1 = v + 2, ..$ is the main assumption. This power consumption can be defined as a voltage drop over a inserted resistor into the GND line of this chip. The implementation from [22] was done on a standard microcontrolller(Atmel ATmega).

Finding the parts of the power trace that is able to allow the identification the individual trial divisions or even the incrementation operations,which are implemented in an 8-bit arithmetic manne, proves to be the main challenge. More about the experimental results can be found [8].

6. Countermeasures

The main purpose of this chapter is to present some countermeasures in order to provide a new level of security to those systems that use RSA as method of encryption. The simplest solutions are an implementation of Algorithm 1,which generate each candidate independent from its predecessors and dividing each prime candidate by all elements of the trial base. Those solutions prove to be not so efficient in terms of number of basic operations performed.

The side-channel leakage of the trial divisions of the previous prime candidates or the maximum information leakage that is possible can be compensated by performing XOR between each prime candidate and some randoms bits to every candidate.

Other countermeasures consist in either methods of protection in case of side-channel attacks or other prime generation algorithms like the ones from [23, 24, 25, 26, 27].

7. Conclusions

Side-channel attacks proves to be the most implemented in practice in many contexts with the purpose of the exploitation of the physical characteristics of the system. The vulnerability spotted is situated during the execution of the RSA key generation step based on the prime generation used for p and q.

Under weak assumptions on the side-channel leakage the attack proved to be works and practical experiments show that the assumption may be realistic. In [27] is debated the fact that the RSA key generation process may be vulnerable to side-channel attacks.

Other important that by reduction to SVP or other well-known $\mathbb{NP}$-hard problems that can be resolved in an efficient time the RSA cryptosystem proved to be even theoretical vulnerable.

References

[1] S. Y. Yan, *Cryptanalytic Attacks on RSA*. Springer Science + Bussiness Media, 2008.

[2] P. Kocher, *P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems*. Springer Science, 1996, vol. Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109.

[3] Kocher, P.C., Jaffe, J., Jun, B. :, *Differential Power Analysis*. Springer Science, 1999, vol. Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666.

[4] Schindler, *A Timing Attack against RSA with the Chinese Remainder Theorem*. Springer Science, 2000, vol. Paar, C., Koç, Ç.K. (eds.) CHES 2000. LNCS, vol. 1965.

[5] Schindler, *A Combined Timing and Power Attack*. Springer Science, 2002, vol. Paillier, P., Naccache, D. (eds.) PKC 2002. LNCS, vol. 2274.

[6] Boneh, D., Franklin, J., *Effcient Generation of Shared RSA keys*. Springer Science, 1997, vol. Kaliski Jr.,B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294.

[7] Clavier, C., Coron, J.-S., *On the Implementation of a Fast Prime Generation Al- gorithm*. Springer Science, 2007, vol. Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727.

[8] Emilia Käsper, Peter Schwabe (auth.), Christophe Clavier, Kris Gaj, *Cryptographic Hardware and Embedded Systems CHES 2009*. Springer Science, 2009.

[9] K. Conrard. The miller–rabin test. [Online]. Available: https://kconrad.math.uconn.edu/blurbs/ugradnumthy/millerrabin.pdf

[10] Brandt, J., Damgard, I., Landrock, P., *peeding up Prime Number Generation*. Springer Science, 1993, vol. Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739.

[11] Menezes, A., van Oorschot, P.C., Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1997.

[12] D. Micciancio. The lll algorithm. [Online]. Available: https://cseweb.ucsd.edu/classes/wi12/cse206A-a/lec3.pdf

[13] C. Peikert. Svp, gram-schmidt, lll. [Online]. Available: https://web.eecs.umich.edu/~cpeikert/lic13/lec02.pdf

[14] Coppersmith, D., *Small Solutions to Polynomial Equations, and Low Exponent Vulnerabilities*, 1997.

[15] M.Pohst, *On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications*. ACM Sigsam Bull, 1981.

[16] R.Kannan, *Improved algorithms for integer programming and related lattice problems, in Sympo- sium on Theory of Computing (STOC 1983)*. ACM, 1983.

[17] U. Fincke, M. Pohst, *Improved methods for calculating vectors of short length in a lattice, including a complexity analysis*. Math Comput, 1985.

[18] D. A. Noah Stephens-Davidowitz. Just take the average! an embarrassingly simple 2n-time algorithm for svp (and cvp). [Online]. Available: https://arxiv.org/pdf/1709.01535v1.pdf

[19] Tsuyoshi Takagi Masato Wakayama Keisuke Tanaka Noboru Kunihiro Kazufumi Kimoto Yasuhiko Ikematsu, *International Symposium on Mathematics, Quantum Theory, and Cryptography Proceedings of MQC 2019*. Springer Science, 2021.

[20] Miklos Ajtai, *Generating hard instances of lattice problem*. STOC'96, 2004.

[21] Odev Regev, *On lattices, learning with errors, random linear codes, and cryptography*. ACM, 2009.

[22] T. Finke, M. Gebhardt, and W. Schindler, *Computational Algebra Group, School of Mathematics and Statistics.* University of Sydney, 2009.

[23] Marc Joye1, Pascal Paillier1, and Serge Vaudenay. Efficient generation of prime numbers. [Online]. Available: https://link.springer.com/content/pdf/10.1007/3-540-44499-8_27.pdf

[24] Menezes, A., van Oorschot, P.C., Vanstone, S.A., *Handbook of Applied Cryptography.* CRC Press, 1997.

[25] Chabanne, H., Dottax, E., Ramsamy, L. Masked prime number generation. [Online]. Available: http://www.cosic.esat.kuleuven.be/wissec2006/papers/29.pdf

[26] Joye, M., Paillier, P., *Fast Generation of Prime Numbers on Portable Devices.* Springer Science, 2006.

[27] Clavier, C., Coron, J.-S., *On the Implementation of a Fast Prime Generation Algorithm.* Springer Science, 2007.