CONTRIBUTION TO THE THEORY AND PRACTICE OF GENERATING RSA ALGORITHM KEYS

Tomislav Unkašević[1, 2, *],
Zoran Banjac[2],
Milan Milosavljević[1],
Predrag Milosav[1, 2],
Medhat Abdelrahman Mohamed Mostafa[3]

[1]Singidunum University, Belgrade, Serbia
[2]Vlatacom Institute, Belgrade, Serbia
[3]European University, Belgrade, Serbia

Abstract:
The RSA cryptographic algorithm is widely used in data protection and electronic business. One of the most important parameter of the achieved level of security is the quality of the generated cryptographic keys with which the algorithm is applied. In this paper a few cases in which poorly generated RSA cryptographic keys caused dis-accreditation of the system are described. The recommendations and checks that must be fulfilled during generation of RSA algorithm cryptographic keys are formulated in order that keys were considered safe for implement on.

Keywords:
RSA algorithm, RSA keys, random number generation.

1. INTRODUCTION

The level of data protection achieved by the application of cryptographic algorithm in data protection mostly depends, on the quality of the cryptographic algorithm and the quality of the applied cryptographic key. The significance and role of the applied cryptographic algorithm is indisputable and clear. The role and significance of the applied cryptographic key is often overlooked. In the case of using the RSA cryptographic algorithm, we will show the role and significance of the quality of the applied cryptographic key to the security of data that are protected.

The cryptographic algorithm RSA, [1], is one of the cryptographic cornerstones of protection in information systems and e-commerce. It belongs to the class of computationally secure cryptographic algorithms based on the complexity of the problem of factorization of natural numbers.

The RSA algorithm is defined by natural numbers n, p, q, e and d which meet the following requirements:

- $n = p \cdot q$ where p, q are prime numbers
- e and d meet the following requirements

$$gcd\left(e, \varphi(n)\right) = 1, \quad ed = 1 \bmod \left(\varphi(n)\right),$$
$$\varphi(n) = \left(p-1\right)\left(q-1\right),$$

Correspondence:
Tomislav Unkašević

e-mail:
tomislav.unkasevic@vlatacom.com

DOI: 10.15308/Sinteza-2019-589-594

Wherein $\varphi(n)$ is Euler's function, $gcd(a,b)$ is the greatest common divisor of *a, b,* and $a \bmod n$ is remainder of the division of the number *a* by *n.* Then $k_e = (n,e)$ is the encryption key and it is called the public key. Decryption key is $k_d = (n,d)$ and it is called a secret key.

Messages m are represented by numbers from the interval $0 \le m < n$ and their enciphered form *c* is obtained as

$$c = E_{k_e}(m) = m^e \bmod n \tag{1}$$

Decryption takes place as follows

$$D_{k_d}(c) = c^d \bmod n = m \tag{2}$$

The RSA algorithm belongs to the class of computationally secure cryptographic algorithms, which means that its parameters must have appropriate characteristics so that messages protected by this algorithm have the desired level of security. The RSA algorithm's security rests on ignorance of efficient ways to factorize large natural numbers, and in order to achieve security in practical terms its parameters must have appropriate characteristics. Bearing in mind that in order to achieve a minimum security level, when applying the RSA algorithm, it is required that the length of parameters *p* and *q*, marked with $|p|$ and $|q|$, respectively, $|p| = |q| = 1024$ bits and $|n| = 2048$ bits.

The cryptographic algorithm RSA is most often used to create an electronic signature and an electronic envelope.

From the point of view of the safety of the generic RSA algorithm, the primary role is played by attacks that reconstruct the secret key (n,d). In this context; attacks can be grouped according to the methods used to achieve this goal:

- Algorithms for factorization of the RSA algorithm module
- Characteristics of the applied public and secret key
- Properties induced by the method of implementing the RSA algorithm

Details can be founded in [2], [3], [4], [5], [6].

2. IMPACT OF CRYPTOGRAPHIC KEYS

The importance of cryptographic keys for safe use of cryptographic algorithms was detected at the end of the nineteenth century and formulated as the Kerckhoff's principle. According to Kerckhoff's principle, [7] the security of each cryptographic system rests with a good cryptographic algorithm on the assumption that its keys are generated in a random way and are unknown to the opponent. The following examples illustrate what can happen in data protection systems when this principle is not respected.

Non random values from certified true random number generator

The first example concerns the system for access to electronic state and commercial services in Taiwan, the analysis presented in [8].

Each user has a smart card called the "Citizen Digital Certificate card" with his electronic certificate issued by the appropriate accredited certification body and which is used for identification, authentication and authorization when using electronic services. As an algorithm for creating an electronic signature, the RSA is used and its keys are generated on a smart card with a built-in random number generator that has a FIPS 140-2 certificate level 2. In [8], the authors analyzed the quality of the generated modules, number n in the description of the RSA algorithm, on a sample of two million issued electronic certificates. Module size in the analyzed cases is 1024 bits. The authors said that in a given sample they successfully factored 184 modules. A number of modules, 103 of them, are factorized due to the fact that some of them had a common prime factor, and the remaining detection of repetition forms that occur when generating prime numbers in this system.

The two most common factors detected in the analysis are given in the following equations.

$$\begin{aligned} p = c&0000000000000000000000000000000\\ &0000000000000000000000000000000\\ &0000000000000000000000000000000\\ &000000000000000000000000000002f9 \end{aligned} \tag{3}$$

$$\begin{aligned} q = c&92424922492924992494924492424922\\ &49292499249492449242492249299249\\ &92494924492424922492924992494924\\ &49242492249292499249492442 4e5 \end{aligned} \tag{4}$$

The factor (3) indicated by *p* appears forty-six times, and the factor (4) indicated by *q* appears seven times in the one hundred and three cases.

The presented results of the conducted analysis indicate the possibility of identity theft and phishing in electronic commerce. A rough estimate is that in the analyzed system, one identity can be stolen on every ten thousand issued electronic identities, which is not at all harmless and incomparably greater than the probability that the theory recognizes as a danger in the case of good random generators. Similar analyzes for other environments and their results are given in papers [9], [10].

These results indicate that the probable cause of the problem is the generator of random numbers on the smart cards as the theoretical analysis shows that the probability of the event is negligible. Namely, probability that two randomly generated modules n_1, n_2 of the RSA algorithm have a common factor, $gcd(n_1, n_2) \neq 1$ provided that it is $n = pq, |p| = |q| = k$, is approximately equal

$$P\left(gcd(n_1, n_2) \neq 1\right) \approx \frac{k(k-1)}{(k-2)2^k} \to 0 \quad when \quad k \to \infty \quad (5)$$

On the other hand, the appearance of the bit pattern in the generated prime numbers, the numbers *p* and *q*, points to the possible multiple use of the random sequence generated for prime number candidates.

Non random prime number generation from pseudoraandom number generator

In the paper [11], the authors analyzed the statistical properties of the RSA algorithm key generated by various software libraries for this purpose. The result of their analysis showed that it is possible to detect with high probability the library by which the sample of RSA public keys is generated. That analysis showed that the keys generated by the Infineon RSA Library version v1.02.013 have significant deviation from the uniform distribution for events $(p \mod x)$ and $(n \mod x)$ when *x* is a small prime number. This result was intrigued by another group of researchers who came up with the idea of trying to factor RSA keys obtained in this way.

In the first step they found that all the prime numbers generated by this library have a form

$$p = k \cdot M + \left(65537^a \bmod M\right) \quad (6)$$

Wherein *M* is a product of the first *N* prime numbers, i.e.

$$M = \prod_{i=1}^{N} P_i = 2 \cdot 3 \cdot \ldots \cdot P_N \quad (7)$$

and *k, a* are unknown random numbers. So,

$$n = \left(k \cdot M + \left(65537^a \bmod M\right)\right) \cdot \left(l \cdot M + \left(65537^b \bmod M\right)\right) \quad (8)$$

Coppersmith method, [12], is used for factorization. The method can be used for factoring RSA modules when one number of the high bits of one factor is known, [11].

In this case, the idea is as following.

Denote the order of the element 65537 in the group Z_M^* by $ord(65537)$.

Since the system is designed for factorization of the modules in the form (8), the first step is to determine are the modules of desired shape. This is achieved by the following analysis

$$n = \left(k \cdot M + \left(65537^a \bmod M\right)\right) \cdot \left(l \cdot M + \left(65537^b \bmod M\right)\right) = \\ = 65537^{a+b} \bmod M = 65537^c \bmod M \quad (9)$$

So, *n* is good candidate if there is c such that it is $n = 65537^c \bmod M$

Then for each number $a \leq ord(65537)$ using the Coppersmith's algorithm obtains number *k* and checks if the obtained number is a factor of *n*. If answer is yes, *n* is factorized, $q = n / p$ but if it is not the next *a* is chosen.

Using this algorithm, the RSA keys obtained by Infineon RSA Library version v1.02.013, it is possible to factorize efficiently modules of lengths 512 and 1024 bits, and special cases of keys 2048 and 4096 bits long.

Details and more efficient algorithms can be found in the [11].

Special properties of the prime numbers usage

In this case, we will consider one feature of RSA algorithm.

We call the message *m* fixed point relative to the RSA algorithm with parameters n, p, q, e and d if it is satisfied that

$$m^e = m \bmod n \quad (10)$$

i.e. encryption does not change message. For each selection of parameters n, p, q, e and d there are fixed

point messages but their number varies depending on the selected p, q and e. This can be a problem when implementing a digital envelope system

Digital envelope is a key distribution technique for symmetric cryptographic algorithms in which the data are encrypted by symmetric cryptographic algorithm, for example AES, and the key used is encrypted by the RSA algorithm public key of the recipient. Both cipher texts are sent to the recipient and he first decrypts the key with the secret RSA key and the RSA algorithm, then decrypts the received message with the AES algorithm.

For RSA algorithm with parameters n, p, q, e and d the number of fixed point messages, *NP*, is given by the term

$$NP = \left(gcd\left((e-1),(p-1)\right)+1\right)\cdot\left(gcd\left((e-1),(q-1)\right)+1\right) \quad (11)$$

This results in the one who implements the system and controls the procedures for generating keys for the RSA algorithm can control the number of fixed point messages and thus compromise the system, for example according to [13], by stating that the generated parameters satisfy the following relations:

- $p = 2^m p_1 + 1,\ q = 2^n q_1 + 1,\ e = 2^{\max\{m,n\}} p_1 q_1 + 1, p_1$ and are prime numbers, then all the messages are fixed point messages.
- $p = 2^m p_1 + 1,\ q = 2^n q_1 + 1,\ e = 2^{\min\{m,n\}} p_1 q_1 + 1, p_1$ and q_1 are prime numbers, then randomly selected message is fixed point message with probability approximately $1/2^{|m-n|}$

Controlling the size of the numbers m and n the attacker can adjust the security of the system with its attacking capabilities.

If one uses a digital envelope technique by combining, for example, a 256-bit AES algorithm and RSA algorithm with a 2048-bit key, its security would, at first glance, be respectable. However, if the parameters for RSA generated as in the previous examples, it would practically not be protected.

In case of the parameters p, q, e are randomly chosen, the probability that the randomly selected message is fixed point message is approximately $(\ln n)^3 / n$ which is negligible considering the size of the number n .

3. GENERATING RSA KEY

In part I the RSA algorithm definition is given. As we mentioned in introduction the RSA algorithm's security rests on the fact that there is no efficient ways to factorize large natural numbers, and in order to achieve security in practical terms its parameters must have appropriate size. So, selection of parameters must ensure that the modulus of specific RSA algorithm has a size that prevents efficient factoring.

Module size $n = p \cdot q$ satisfies $|p|+|q|-1 \le |n| \le |p|+|q|$.

Have a security parameter k given, and $k = |p| = |q|$ then the number *2k* represents the level of security of the RSA algorithm.

Practical algorithms for generating RSA keys and their consequences

Having in mind the description given in I parameters p, q are randomly selected from a set of prime numbers. This choice is most often carried out in three steps:

1. A random number of length k is generated using a random number generator
2. Check its length (zeroes which are not significant digits are not counted) which must be k.
3. Check that the generated number is prime number.

Each of these steps takes some time.

If a good random generator is used to be a random choice of k bits get the number of length k is $1/2$

The probability of randomly selected number from the interval $(1, 2^k)$ is prime number is about $1/(k \ln 2)$. Therefore, the likelihood that the number whose representation makes randomly selected k bits has a length k and it's prime number is $1/(2k \ln 2)$ what in the case when it is $k = 1024$ roughly speaking means more than a thousand and four hundred attempts to make one successful. What further aggravates the situation is the fact that deterministic algorithms to determine if the number is prime takes long time, and, in the previous example, this type of check will occur with more than one thousand and four candidates.

Such a purely theoretical approach is inefficient for mass application, and in practice, various modifications are used to increase efficiency.

If with $b_0 b_1 \ldots b_{k-2} b_{k-1}$ denote the generated random bits, b_0, b_{k-1} to be the smallest and greatest weight respectively, the usual modifications are:

1. Set $b_0 = b_{k-1} = 1$ and bits $b_1, \ldots, b_{k-2}$ are generated randomly. This eliminates the generation of candidates who do not have length k $(b_{k-1} = 0)$ and which are certainly not prime $(b_0 = 0)$

2. It is known that the product is two k bit numbers can have a length $2k$ or $2k-1$. To reduce the chances of selecting numbers p,q which give the RSA modulo length $2k-1$ in the product an analysis is performed showing the numbers p,q should be selected from the interval $\left[\sqrt{2}\cdot 2^{k-1}, 2^{k}-1\right]$. A more efficient approach, which induces a non-significant reduction in entropy, is by putting $b_0 = b_{k-2} = b_{k-1} = 1$.

Items 1. and 2. do not distort the randomness of the selection because the selection process is such that candidates who do not meet the requirements of those items were certainly rejected.

Therefore, the basic, potential source of problems with RSA keys originates from generating random sequences on two grounds:

- The quality of the random generator
- How to use generated random sequences in the process of generating RSA keys.

In part II, we saw examples of both situations.

Instructions for generating RSA keys

In accordance with the previous one, regarding the level of security that RSA offers, the first factor in order is the quality and length of the generated keys. We have seen in that some of the properties of the generated elements can be such as to dis-accredit the system in which they are used. In order to avoid such situations in the process of generating keys for the RSA algorithm, a standard representation, it is necessary to adhere to certain rules:

1. The random number generator generating candidates for prime numbers must be thoroughly tested and reliable. Sequences representing candidates must be selected independently.
2. Every prime number p length k can be used up to a maximum of once.
3. For every prime number p length k numbers $p\pm 1$ they must have at least one large prime factor
4. For prime numbers p,q it must be valid $|p-q| \geq n^{0.25}$, [14]
5. For the secret key (n,d) it must be valid $d \geq n^{0.292}$, [15].

The above-mentioned rules are technically more detailed in international standards. Recommendations related to random and pseudo- random generators can be found in [16], [17] and [18].

Recommendations on the method of forming candidates for random numbers, checks of their properties and recommended algorithms for this purpose are given in [19], [20] and [21]. These recommendations deal with requests related to items with item numbers 3, 4, and 5. Item number 2 is not recorded as necessary due to the insignificant probability of occurrence in ideal conditions. As shown in practice, [8], [10], [9] and [12], that this property can be the cause of significant security weaknesses in some situations. The authors of the [8] root of this unexpected behavior attributed the inadequate entropy of the applied sources of randomness, although the causes could be covered in the way of implementing the defined recommendations.

4. CONCLUSION

Although by its description and mathematical basics, the RSA algorithm does not require deep mathematical knowledge, it represents one of the cornerstones of security in the computer and networking world. Despite its simplicity, it is not easy to implement it reliably. The previous presentation shows the potential weaknesses of the RSA algorithm keys that lead to system discreditation. The theory shows that in the case of randomly generating parameters the probability of occurrence of some of these characteristics is negligible, but practice, [8], [12] shows that even in seemingly verified situations this does not have to be the case.

In addition to the previous choice of parameters, the security of the systems in which the RSA algorithm is applied is significantly influenced by the way in which it is implemented and interaction with the rest of the system. This can be seen more in [2], [4] and [6].

REFERENCES

[1] R. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, Feb. 1978.

[2] D. Boneh, "Twenty years of attacks on the RSA cryptosystem", *Notices of the American Mathematical Society (AMS)*, vol. 46, no. 2, pp. 203-213, Feb. 1999.

[3] D. Aggarwal and U. Maurer. "Breaking RSA Generically Is Equivalent to Factoring." IACR.org. https://eprint.iacr.org/2008/260.pdf (accessed Feb. 21, 2017).

[4] J. M. Heinek, *Cryptanalysis of RSA and its variants.* Chapman & Hall/CRC, 2010.

[5] R. Lehman, "Factoring Large Integers," *Mathematics of Computatio*n, vol. 28, no. 126, pp. 637-636, Apr. 1974.

[6] S. Y. Yan, *Cryptanalytic Attacks on RSA*. Springer New York, 2008.

[7] A. Kerckhoffs, "La cryptographie militaire," *Journal des sciences militaires*, vol. IX, pp. 5-83, Jan. 1883.

[8] D. J. Bernstein at al. "Factoring RSA Keys from Certified Smart Cards: Coppersmith in the Wild," in ASIACRYPT 2013, Bengaluru, India, December 1-5, 2013, pp. 341-360.

[9] A. K. Lenstra at al. "Ron was wrong, Whit is right." IACR.org. https://eprint.iacr.org/2012/064.pdf (accesed Feb. 21, 2017).

[10] N. Heninger, Z. Durumeric, E. Wustrow and J. Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices," in Proceedings of the 21st USENIX conference, Bellevue, WA — August 08 - 10, 2012, pp.205-220.

[11] P. Švenda at al., "The Million-Key Question – Investigating the Origins of RSA Public Keys," Faculty of Informatic Masaryk University, Brno, CZ, Rep. FIMU-RS-2016-03, Jul. 2016.

[12] M. Nemec, M. Sys, P. Svenda, D. Klinec and V. Matyas, "The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli," in ACM SIGSAC, Dallas, Texas, Oct. 30 – Nov. 03 2017, pp. 1631-1648.

[13] T. Unkašević, "About an weakness of the RSA algorithm and posibility for its misuse," (in Serbian) in SYMOPIS 97, Bečići, 1997, pp. 768-773.

[14] B. de Weger, "Cryptanalysis of RSA with Small Prime Difference," *Applicable Algebra in Engineering, Communication and Computing*, vol. 13, no. 1, pp. 17-28, 2002.

[15] D. Boneh and G. Durfee, "Cryptanalysis of RSA with Private Key *d* less than N^0.292," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1339-1349, 2000.

[16] E. Barker and A. Roginsky, "Cryptographic Key Generation," National Institute of Standards and Technology USA, Rep. NIST SP800-133, 2012.

[17] E. Barker and J. Kelsey, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators," National Institute of Standards and Technology USA, Rep. NIST SP800-90A 2015.

[18] E. Barker and J. Kelsey, "Recommendation for Random Number Generation (RBG) Constructions," National Institute of Standards and Technology USA, Rep. NIST SP800-90C 2016.

[19] *RFC 5859 Asymmetric Key Package*, IETF/ISOC, 2010.

[20] *Digital Signature Standard (DSS),* FIPS 186-4, National Institute of Standards and Technology USA, 2013.

[21] E. Barker, L. Chen and D. Moody, "Recommendation for Pair-Wise Key-Establishment Schemes Using Integer Factorization Cryptography," National Institute of Standards and Technology USA, SP800-56B, 2014.