Safe primes are not so safe

Balazs Benics
balazs.benics@inf.elte.hu
Eötvös Loránd University
Budapest, Hungary

Gergely Nagy
nagygeri97@caesar.elte.hu
Eötvös Loránd University
Budapest, Hungary

Gabor Bencze
gaborbencze@caesar.elte.hu
Eötvös Loránd University
Budapest, Hungary

Gabor Kruppai
gabor.kruppai@inf.elte.hu
Eötvös Loránd University
Budapest, Hungary

Áron Attila Meszaros
ameszaros@inf.elte.hu
Eötvös Loránd University
Budapest, Hungary

ABSTRACT

In the last few years privacy and security have been becoming more important as more of our private information is moving online. When transferring these sensitive information between devices the message is usually encrypted using RSA. The secureness of this method depends on not having an effective way of factorizing integers. In this paper we present a new mathematical model for factorizing RSA keys which were generated using at least one safe prime. We also propose an effective implementation, and show that we are able to efficiently break twice as much keys in the same amount of time as state-of-the-art algorithms.

CCS CONCEPTS

• **Security and privacy → Cryptanalysis and other attacks**; *Public key encryption*;

KEYWORDS

RSA cryptosystem, security strength, safe primes, prime classes, Chinese remainder theorem

ACM Reference format:
Balazs Benics, Gergely Nagy, Gabor Bencze, Gabor Kruppai, and Áron Attila Meszaros. 2020. Safe primes are not so safe. In *Proceedings of Chapcha '22: ACM SIGSAC Conference on Computer and Communications Security, Chapcha, Bhutan, Nov 11–14, 22 (Chapcha '22),* 5 pages.
https://doi.org/10.1145/1122446.1122458

Chapcha '22, Nov 11–14, 22, Chapcha, Bhutan

ACM ISBN 978-1-4242-4242-14/22/08…$15.00
https://doi.org/10.1145/1122446.1122458

1 INTRODUCTION

RSA algorithm is an asymmetric encryption method which was published by Ron **R**ivest, Adi **S**hamir és Len **A**dleman in 1978 [14]. In this algorithm both participants (*sender* and *receiver*) has it's own public and private key. These keys are *large enough* prime numbers. For sending a message, we need to calculate the product of our *private* key and the recipient's *public* key. Let call this product $n = pq$. Here p and q will be the secret and n will be shared publicly besides the participant's public key and it will be used as a modulus during the encryption and decryption. Then the encryption function of the RSA algorithm will be the following polynomial *(with $e \geq 3$ integer)*:

$$f(x) \equiv x^e \pmod{n} \tag{1}$$

This encryption function is a so called, one-way trapdoor function, since finding the inverse of this function is so computation heavy that it is considered unfeasible on current hardware.

For transmitting a message M from Alice to Bob, we need to do the following to get the encrypted *chipertext*:

$$C \equiv M^e \pmod{n} \tag{2}$$

This chipertext C could be transmitted and Bob could recover the original message M using this method:

$$C^d \equiv (M^e)^d \pmod{n} \equiv M \tag{3}$$

Clearly, RSA based encryption methods rely on the fact, that factoring numbers is difficult even for modern hardware, still there were already several attempts to break RSA keys [3, 4, 8, 16].

State-of-the-Art

In this subsection we summarize various methods that have been used to factorize RSA keys, and the general approach to attack the RSA cryptosystem.

RSA Factoring Challenge.
The RSA Factoring Challenge requires participants to factorize large RSA keys. The list of keys was published in 1991 by RSA Laboratories [7]. The published keys were randomly generated using a hardware random number generator.

The largest successfully factorized keys on the list is RSA-768 [5, 9], which was successfully factorized in 2009. The factorization took approximately 2000 years of CPU time. There are claims that RSA-240, a 795-bit RSA key, was also successfully factorized [17], but at the time of writing this paper the results have not yet been published.

Lenstra-Lenstra-Lovász algorithm.
The Lenstra-Lenstra-Lovász (LLL) lattice basis reduction algorithm [10] can, in general, be used to find integer solutions to many problems, which include the cryptanalysis of public-key encryption schemes. In particular, Renault et al. have used the LLL algorithm to factorize square-free integers in polynomial time [12]. May has used LLL-reduction for solving RSA and factorization problems [11], and has shown that it can be used to solve a relaxed version of the RSA factorization problem in polynomial time.

General attacks on RSA type cryptosystems.
Wiener's attack [19], which uses the continued fraction method, can be used to calculate the private key when it is suitably small. Coppersmith [6] also introduced two algorithms to find small roots of bivariate polynomial equations and find small solutions of univariate modular polynomial equations. Combining these findings, Wiener's method was further imporved by an extension by Bunder et al. [4] to some RSA type schemes.

Contributions

In this paper we present a mathematical model and proof of having a less computation-heavy method for factoring integer numbers of two primes, when at least one of them is a *safe-prime*. We also provide a proof-of-concept algorithm and implementation with real-world statistics and analysis. These results opens the discussion whether we should avoid using *safe-primes* for cryptography, especially for RSA cryptosystems.

2 METHODS

To generate safe RSA keys, we have to find a huge, randomly chosen prime number. In key generation algorithms, it is achieved by choosing a random number p and check its primality with probabilistic or deterministic primality tests, such as Miller–Rabin, Baillie-PSW or AKS [1]. If the prime test finds a number that witnesses p, another p will be generated and tested in cycle until a prime p was not found. Finding or generating random p and q prime numbers to produce the $m = pq \pmod b$ shared key, where b is the length of the key usually in bits, is a fairly easier task than its reverse operation i.e. finding the appropriate p and q from a given n which is a remainder by modulo b. To speed up this reverse prime search, we examined various prime classes in order to find class-specific prime properties that help reduce the computational cost of primality tests on these prime classes.

Prime class	FN	PRNG	HRNG
Safe primes	0.0412	0.0452	0.0448
Fibonacci primes	0.0348	0.0352	0.0350
Cullen primes	0.0302	0.0287	0.0291
Wieferich primes	0.0153	0.0142	0.0144

Table 1: The acronyms stand for the methods used to generate primes: First N prime number, Pseudo-Random Number Generator(+primality test), Hardware Random-Number Generator(+primality test). The numbers shows the coverage of different prime classes based on the statistics made by the listed methods.

Prime classes

We collected, selected and examined the largest prime classes which are statistically cover a considerably large subset of primes. The statistical coverage of a prime class C over the set of primes was determined by the following limit:

$$\mathcal{P}_C = \lim_{n\to\infty} f(x) = \frac{\gamma(n,m)}{\pi(n,m)} \tag{4}$$

where
$\pi(n,m) = |\{p : p \text{ is a prime} \wedge n \le p \le m\}|$,
$\gamma(n,m) = |\{p : p \in C \wedge n \le p \le m\}|$.

3 MODEL FOR FACTORIZING RSA KEYS

In this section we describe a mathematical model and an algorithm, that can be used to factorize RSA keys, that have been generated using at least one safe prime.

Safe primes are, as described in Section 2, prime numbers p, where there exists a q prime, such that $p = 2q + 1$ is also a prime, or in other words, $\frac{p-1}{2}$ is also a prime[18]. This is the most important property of safe primes and the following model and algorithm all originate from this fact about safe primes.

Mathematical model

In the following description, we assume that an RSA public key is given and we know that is was generated using at least one safe prime. We present a model for factorizing the modulus part of the public key.

Let's assume that the modulus $n = pq$ and p is a safe prime and q is prime. In this case the exponent part of the public key, which is defined to be co-prime to $\lambda(pq) = lcm(p-1, q-1)$ [4, 12], is also co-prime to $p-1$.

Since p is a safe prime then $p - 1 = 2r$ where r is also a prime number.

Lemma 3.1. *Given an integer $n \in \mathbb{N}$, a prime number q and a safe prime $p = 2q + 1$, the following statement holds:*

$$p|n \Leftrightarrow q|n-1$$

Proof. First, let's assume $n = p^r s$ for some $s \in \mathbb{N}, r \in \mathbb{N}^+$. In this case $q = \frac{p-1}{2}$ and using the binomial formula for exponentiation we get

$$n - 1 = (p-1)p^{r-1}s + (p-1)s \tag{5}$$
$$n - 1 = (p-1)s(p^{r-1} + 1) \tag{6}$$
$$n - 1 = \frac{p-1}{2}2s(p^{r-1} + 1) \tag{7}$$
$$\Downarrow \tag{8}$$
$$q|(n-1) \tag{9}$$

□

It is not necessary to find a prime divisor of n, because if n was generated using at least one safe prime, there exists a prime number $q < \frac{n-1}{2}$, such that $q|(n-1)$. In this case, based on Lemma 3.1, $p|n$.

In order to find such a prime q, we have to consider the quotient of the modulus and the exponent part of the public key. This yields a system of congruences, whose solution gives a suitable q prime.

$$q \equiv 1 \pmod{p_1} \tag{10}$$
$$q \equiv 2 \pmod{p_2} \tag{11}$$
$$\dots \tag{12}$$
$$q \equiv r \pmod{p_r} \tag{13}$$

where $r = \lfloor ln(\frac{n-1}{e}) \rfloor$ and $p_1, p_2 \dots p_r$ are the first $r+1$ primes in increasing order. The Chinese remainder theorem can be used to solve this system of congruences [8].

Algorithm

To solve the system of congruences, we first need to calculate the first r prime numbers. The Sieve of Atkin [2] solves this subproblem with $O\left(\frac{p_r}{\log\log p_r}\right)$ time complexity. We use the upper bound $r \cdot (\log r + \log\log r)$ [15] instead of p_r as the input of the algorithm as p_r is still unknown. These values may be precomputed and reused for subsequent calculations as they do not depend on the exact input values.

The Chinese Remainder theorem and the Extended Euclidian Algorithm can then be used to solve the system of congruences.

The time complexity of this step is $O\left((\log_{10} p_1 \cdot p_2 \cdot \dots p_r)^2\right)$.

	LLL	RSA-CRT	Our algorithm	Newly affected
1024-bit	1.65%	3.62%	4.56%	2.65%
2048-bit	1.26%	3.59%	4.20%	2.33%
4096-bit	0.89%	3.61%	4.33%	2.32%

Table 2: The ratio of RSA keys in the sample affected by different key-breaking algorithms. The last column shows the ratio of keys that have previously been unaffected by the state of the art algorithms, but the new algorithm breaks them.

Validation

In order to test the model and algorithm we have created, we gathered a large amount of public and private RSA keys. The private keys are also needed, so that the results of the algorithm can be checked against them.

We used wide variety of public and private RSA keys to verify our algorithm. It was important to observe the ratio of the public and private keys affected out there. Amazon provides a cloud service for users, and some of them are publicly sharing their EBS volumes. We scanned those disks for RSA keys in the same way as Ben Morris presented his method doing so at Defcon conference [13].

In total, the algorithm was tested on 1543 public and private key pairs, all ranging from 1024 to 4096 bits. Our algorithm was tested against other algorithms, such as the LLL-algorithm presented by François Morain et al. [12] and the RSA-CRT algorithm presented by Kim Chong Hee [8]. The ratio of affected RSA keys in our sample is shown in Table 2.

This shows that in all categories our algorithm is able to break more than 4% of the RSA keys in our sample. This means that more than 2% of keys that could not have been previously broken are now easily breakable and are vulnerable.

Finally, we have also measured the runtime of our algorithm, compared to others in the case where all algorithms are able to break a key. We have also measured runtime in the case of newly affected keys. Figure 1 shows that in most cases our algorithm is up to 21% faster than the fastest existing algorithm, and even in the case of newly affected RSA keys, the algorithm maintains its low runtime compared to other algorithms.

To summarize, validation of our proposed new algorithm shows that more than 4% of RSA keys are affected by this exploit and of these keys more than half of them was not

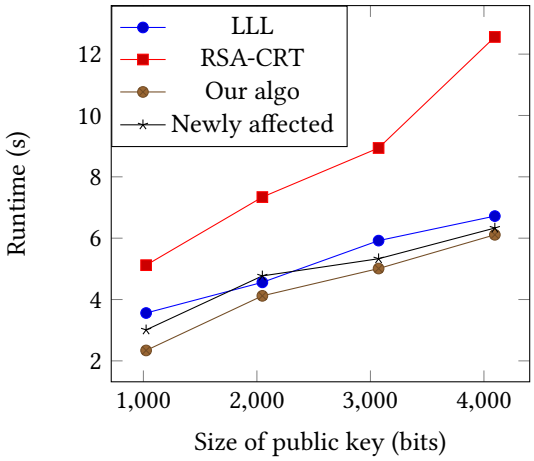

Figure 1: Runtime comparison of different RSA-key breaking algorithms that we have compared.

breakable within reasonable time using previously known methods. Our new method can efficiently factorize these prime numbers and it is as fast as the fastest state-of-the-art algorithms, while affecting much more of the publicly available RSA keys.

4 DISCUSSION

In this work, we have presented a new mathematical model for factorizing RSA keys. We have proved that this model can be used if at least one safe prime has been used for the public key generation. Based on this result, we have also created an algorithm to factorize the modulus part of the public key. We have analyzed its time complexity and the affected RSA keys using accessible public and private keys. This experiment showed that our algorithm is more time-efficient than the best previously known algorithms. Moreover, our research showed that it can factorize more than 4% of the tested RSA keys, which is more than twice as much as other solutions. This means that data encryption are not safe anymore with significant amount of keys, because we can decrypt the data within reasonable time. Hence we should not use them for encrypting. Checking our existing keys is also highly recommended to to make sure our data is safe.

Therefore, in the future we shall properly investigate the amount of the keys affected by our method. A comprehensive study on this question could help us identify precisely which keys should be avoided. In connection with that, RSA algorithms shall also be revised in order not to use safe primes for key generation. Regardless, future research could continue to explore more special classes of prime numbers, as it turned out that using particular information makes some of the RSA keys easily breakable.

Since quantum computing is a popular research target, and expecting promising future, we should consider researching the implications of our findings. Such machines have some interesting properties regards to factorisation, which might could be exploited as well.

REFERENCES

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. 2002. Primes is in P. *Annals of Mathematics* 160 (09 2002). https://doi.org/10.4007/annals.2004.160.781

[2] A. O. L. Atkin and D. J. Bernstein. 1999. Prime Sieves Using Binary Quadratic Forms. *Math. Comp.* 73 (1999), 2004.

[3] Dan Boneh. 1999. Twenty Years of Attacks on the RSA Cryptosystem. *NOTICES OF THE AMS* 46 (1999), 203–213.

[4] Martin Bunder, Abderrahmane Nitaj, Willy Susilo, and Joseph Tonien. 2017. A generalized attack on RSA type cryptosystems. *Theoretical Computer Science* 704 (09 2017), 74–81. https://doi.org/10.1016/j.tcs.2017.09.009

[5] Stefania Cavallar, Bruce Dodson, Arjen K. Lenstra, Walter Lioen, Peter L. Montgomery, Brian Murphy, Herman te Riele, Karen Aardal, Jeff Gilchrist, Gérard Guillerm, Paul Leyland, Jöel Marchand, François Morain, Alec Muffett, Chris Putnam, Craig, and Paul Zimmermann. 2000. Factorization of a 512-Bit RSA Modulus. In *Advances in Cryptology — EUROCRYPT 2000*, Bart Preneel (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–18.

[6] Don Coppersmith. 1997. Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. *Journal of Cryptology* 10, 4 (01 Sep 1997), 233–260. https://doi.org/10.1007/s001459900030

[7] Burt Kaliski. 2005. *RSA factoring challenge.* 531–532. https://doi.org/10.1007/0-387-23483-7_362

[8] Chong Hee Kim and Jean-Jacques Quisquater. 2007. Fault Attacks for CRT Based RSA: New Attacks, New Results, and New Countermeasures. In *Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems*, Damien Sauveron, Konstantinos Markantonakis, Angelos Bilas, and Jean-Jacques Quisquater (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 215–228.

[9] Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Joppe W. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, and Paul Zimmermann. 2010. Factorization of a 768-Bit RSA Modulus. In *Advances in Cryptology – CRYPTO 2010*, Tal Rabin (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 333–350.

[10] A. K. Lenstra, H. W. Lenstra, and L. Lovasz. 1982. Factoring polynomials with rational coefficients. *MATH. ANN* 261 (1982), 515–534.

[11] Alexander May. 2010. *Using LLL-Reduction for Solving RSA and Factorization Problems.* Springer Berlin Heidelberg, Berlin, Heidelberg, 315–348. https://doi.org/10.1007/978-3-642-02295-1_10

[12] François Morain, Guénaël Renault, and Benjamin Smith. 2018. Deterministic factoring with oracles. (02 2018).

[13] Ben Morris. 2019. More Keys Than A Piano: Finding Secrets In Publicly Exposed EBS Volumes. (2019). https://www.defcon.org/html/defcon-27/dc-27-speakers.html#Morris DEF CON.

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

[15] Barkley Rosser. 1941. Explicit Bounds for Some Functions of Prime Numbers. *American Journal of Mathematics* 63, 1 (1941), 211–232. http://www.jstor.org/stable/2371291

[16] Tsuyoshi Takagi and Shozo Naito. 2000. Construction of RSA cryptosystem over the algebraic field using ideal theory and investigation

of its security. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* 83, 8 (2000), 19–29. https://doi.org/10.1002/(SICI)1520-6440(200008)83:8<19::AID-ECJC3>3.0.CO;2-0

[17] Emmanuel Thomé. [n. d.]. 795-bit factoring and discrete logarithms. https://lists.gforge.inria.fr/pipermail/cado-nfs-discuss/2019-December/001139.html. ([n. d.]). Accessed: 03 December, 2019.

[18] J. von zur Gathen and I. Shparlinski. 2013. Generating safe primes. *Journal of Mathematical Cryptology* 7(4) (2013), 333–365. https://doi.org/doi:10.1515/jmc-2013-5011

[19] M. J. Wiener. 1990. Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory* 36, 3 (May 1990), 553–558. https://doi.org/10.1109/18.54902