Mathematical and Computational Applications

MDPI

Article

Mathematical Attack of RSA by Extending the Sum of Squares of Primes to Factorize a Semi-Prime

Anthony Overmars and Sitalakshmi Venkatraman *

School of Engineering, Construction & Design, Melbourne Polytechnic, Locked Bag 5, Preston, Victoria 3072, Australia; AnthonyOvermars@melbournepolytechnic.edu.au

* Correspondence: SitaVenkat@melbournepolytechnic.edu.au; Tel.: +61-3-9269-1171

Received: 8 September 2020; Accepted: 25 September 2020; Published: 28 September 2020

Abstract: The security of RSA relies on the computationally challenging factorization of RSA modulus $N = p_1\, p_2$ with N being a large semi-prime consisting of two primes $p_1$and p_2, for the generation of RSA keys in commonly adopted cryptosystems. The property of p_1 and p_2, both congruent to 1 mod 4, is used in Euler's factorization method to theoretically factorize them. While this caters to only a quarter of the possible combinations of primes, the rest of the combinations congruent to 3 mod 4 can be found by extending the method using Gaussian primes. However, based on Pythagorean primes that are applied in RSA, the semi-prime has only two sums of two squares in the range of possible squares $\sqrt{N-1}$, $\sqrt{N/2}$. As N becomes large, the probability of finding the two sums of two squares becomes computationally intractable in the practical world. In this paper, we apply Pythagorean primes to explore how the number of sums of two squares in the search field can be increased thereby increasing the likelihood that a sum of two squares can be found. Once two such sums of squares are found, even though many may exist, we show that it is sufficient to only find two solutions to factorize the original semi-prime. We present the algorithm showing the simplicity of steps that use rudimentary arithmetic operations requiring minimal memory, with search cycle time being a factor for very large semi-primes, which can be contained. We demonstrate the correctness of our approach with practical illustrations for breaking RSA keys. Our enhanced factorization method is an improvement on our previous work with results compared to other factorization algorithms and continues to be an ongoing area of our research.

Keywords: cryptography; RSA; semi-primes; prime factorization; Euler's factorization; Pythagorean primes; sum of squares of primes

1. Introduction

The RSA (Rivest–Shamir–Adleman) public-key primitive named after its inventors is the most widely used in cryptography since its introduction in 1977 [1]. The generation of RSA keys in public key cryptosystems is based on the modulus of a positive integer $N = p_1 p_2$, where N is a semi-prime, which is a product of two large primes p_1 and p_2 [2,3]. The computationally intractable factorization property of semi-primes when they are large has been the fundamental premise in using RSA keys for computer security in several applications [4–6]. Modern cryptosystems make wide use of RSA encryption and digital signatures for secure message exchange and communication over different types of networks within government as well as various industry sectors [7–9]. These include popular transactions in various applications including Internet of Things (IoT) such as mobile banking, online shopping, smart card payments, e-health and e-mail communications that are available to the common man over the Internet [10,11]. Despite difficulty in breaking RSA keys (i.e., semi-prime factorization), cybercrimes and RSA key attacks are still on the rise [12–14].

A cryptanalytic attack of a short RSA key by M. J. Wiener was established as the first of its kind in 1990 [15,16]. Hence, the difficulty of factoring RSA modulus N by choosing strong prime factors p_1 and p_2 was considered as a solution to address these attacks. Since then, it has become a common practice to employ 512-bit RSA to provide the required strong primes for many cryptographic security protocols [17]. Even though 512-bit RSA modulus was first factored in 1999, with the high computational power of today, there is still difficulty with 512-bit factorization in real-time applications [18]. Therefore, 512-bit RSA keys are actively used by popular protocols for email authentication such as Domain Keys Identified Mail (DKIM), retrieving messages from email servers by the client such as Post Office Protocol 3 (POP3), Simple Mail Transfer Protocol (SMTP) and Internet Message Access Protocol (IMAP), data exchange on the web such as Hypertext Transfer Protocol Secure (HTTPS), connecting two computers such as Secure Shell (SSH), and providing privacy such as Pretty Good Privacy (PGP) [19–21].

Enthusiastic amateurs have tried to factor 512-bit RSA keys and in 2012, Zachary Harris factored the 512-bit DKIM RSA keys used by Google and several other major companies in 72 h per key, by using distributed cloud services [22–24]. The Factoring as a Service (FaaS) project demonstrates that a 512-bit RSA key can be factored reliably within four hours in a public cloud environment [19]. Factorization of the 768-bit RSA key has also been demonstrated using various methods such as the number field sieve factoring method and is several thousand times more difficult than 512-bit RSA keys. The higher the key size, the more difficult it is to factor the RSA key. Hence, with new methods and progress in computing power over time, there are risks and implications for the future of RSA. This forms the motivation for studying the mathematical principles of semi-prime factorization in proposing a novel method to increase the likelihood of breaking an RSA key.

Euler's factorization of a semi-prime $N = p_1 p_2$ is based on the property of p_1, and p_2, both congruent to 1 mod 4 [25]. These constructions are based on Pythagorean primes that are applied in RSA [26–28]. These contribute to only a quarter of the possible combinations of primes and the rest of the combinations congruent to 3 mod 4 are found based on the Gaussian prime extension method [29]. The semi-prime has only two sums of two squares in the range of possible squares from $\sqrt{N-1}$ to $\sqrt{N/2}$, and therefore, we extend previously established methods to increase the likelihood that a sum of two squares can be found [30,31]. Even though many sums of squares exist, once any two of them are found, we show that it is sufficient to only find two solutions to factorize the original semi-prime. Our enhanced factorization method is an improvement of our previous work. We apply our method for case scenarios and provide the necessary conjectures. Our algorithm is practically simple and is implemented using rudimentary arithmetic operations that require minimal computational memory with search cycle time being a factor to be considered for very large semi-primes. Further, we demonstrate the successful breaking of RSA keys such as 768-bit RSA verified through the implementation of our algorithm in Java. We provide the results highlighting the complexity of our enhanced factorization algorithm and comparing the performance with other factorization algorithms laying the scope for future research as well.

The rest of the paper is organized as follows. Section 2 discusses related work and the uniqueness of our work. Section 3 provides our enhanced semi-prime factorization by extending the sum of squares method and specifies our algorithm with implementation steps. In addition, the performance of the algorithm, its complexity and comparison with other factoring methods are described. Section 4 demonstrates the correctness of our algorithm when applied to break the 768-bit RSA key. Section 5 discusses the results and finally conclusions along with future research directions are given in Section 6.

2. Background

The difficulty of the semi-prime factorization problem forms the essential aspect to the security of an RSA cryptosystem. Revisiting RSA cryptography [1,3,17], assuming that p_1 and p_2 are two primes used to generate a semi-prime $N = p_1 p_2$, the Euler's totient function φ_n is given by

$$\varphi_n = (p_1 - 1)\ (p_2 - 1)$$

In RSA based public key cryptography, two different keys known as public and private keys are used to perform the encryption and decryption of data or messages [32,33]. Any sensitive information is encrypted with a public key and it requires a matching private key to decrypt it. The public key P_U is chosen arbitrarily as a pair $P_U = (N,\ e)$ where e is an integer and not a factor of $mod\ N$ and $1 < e < \varphi_n$. The private key P_R is based on tuples $(d,\ p_1,\ p_2,\ \varphi_n)$, such that $P_R = (N,\ d)$ and d is determined using the extended Euclidean algorithm $e\ d\ mod\ \varphi_n = 1$. A public key P_U is used to encrypt a message m, into a cipher text C, such that

$$C = m^e\ mod\ N$$

To retrieve the original message, the corresponding private key P_R is used to decrypt the encrypted message such that

$$m = C^d\ mod\ N$$

RSA cryptosystems make use of public key and private key generation techniques for security of data and end-to-end security of information transmission that cannot be understood by anyone except the intended recipients. These techniques are employed to authenticate the sender and the receiver of a message and to ensure that the integrity of the data or message received without being tampered with. However, the problem of determining d from $P_U = (N,\ e)$ is equivalent to finding factors of RSA modulus N. Hence, choosing strong primes for p_1 and p_2 is very important such that the factorization of the semi-prime N, becomes computationally infeasible and nontrivial for an adversary. In practical applications, a smaller private key may be used for a faster decryption algorithm to improve the computational speed of online transactions. Once the private key is found, it can result in a total breaking of RSA posing a great security risk. Hence, an enhanced prime factorization method to attack the small decryption exponent could pose serious security challenges to RSA cryptosystems that are widely adopted even today.

Several studies have attempted to perform a general survey of attacks on the RSA cryptosystem since its introduction [16,19,34]. In 1990, Wiener introduced the first RSA cryptanalytic attack showing that if the decryption exponent is small with an upper bound given by $d < N^{0.25}$ using the continued fractions method [35]. Subsequently, Boneh and Durfee [36] proposed an attack on short decryption exponents with an improved upper bound while the RSA attack by Blomer and May [37] had demonstrated an upper bound of $N^{0.29}$ with lattices of smaller dimension. Coppersmith's technique used lattice reduction approach for finding small solutions of modular bivariate integer polynomial equations [38].

Most of the recent research has also been focusing on extending the number range upper bound of d and e in the RSA private and public keys by working on the limitation of the Wiener and Coppersmith methods by approaching the problem differently. One recent work [39] considers the prime sum $p_1 + p_2$ using sublattice reduction techniques and Coppersmith's methods for finding small roots of modular polynomial equations, achieving slight improvement in the upper bound with reduced lattice dimension. Another work [17] uses a small prime difference $p_1 - p_2$ method which is then developed into a continuous fraction as per Wiener's original method. While these extend the range of Boneh's original limit, the Lenstra–Lenstra–Lovász (LLL) method is still seen as the state of the art which was first proposed by Lenstra and Lenstra in 1991 [40].

A survey on the history of number theory reveals that it has been explored widely by mathematicians to establish different representations of integers, in particular prime numbers, with a view to arrive at more efficient methods in deriving them [41–47]. Up until about a decade ago, there had been a strong mathematical interest in polynomials which generated sums of squares. However, recently there is a reviving interest in their application to practical experimentations for establishing their rudimentary computations and implications to cryptography [29,48–52]. In line with these approaches, it has been proved in our previous work that the semi-primes can be represented as the sum of four squares [30]. A new factorization method as a faster alternative to Euler's method [25] was proposed by establishing the relationship among the four squares. Our interest is to apply this method in a novel way to the semi-prime factorization problem and is part of our ongoing research. This paper aims to explore further, on new findings of the method that once one sum of the squares is known, this can be used to find the other.

3. Proposed Method

We consider the basis of an earlier work [25,42] that showed a semi-prime N, constructed from two primes, p_1 and p_2, is also congruent to $1(mod\ 4) \equiv p_1 p_2$. Further, in [30], it was established that a sum of four squares, could be reduced to two sums of two squares using the Brahmagupta–Fibonacci identity given in [49]. We note that Gaussian primes are of the form $P \equiv 3\ mod\ 4$, and cannot be represented as the sum of two squares [29]. On the other hand, Pythagorean primes are of the form $P \equiv 1\ mod\ 4$ [26–28,53]. In accordance with Fermat's Christmas theorem, an odd prime p can be represented as the sum of two squares of integers x_1 and x_2, if and only if $1\ (mod\ 4) \equiv p = x_1^2 + x_2^2$ [42,54]. This property was useful to determine which numbers could be represented as the sum of two squares, which was later proved by Euler [25].

In an earlier work [30], it was proved that if a semi-prime is constructed using two Pythagorean primes of the form $P \equiv 1\ mod\ 4$ then Euler's factorization can be used to find two representations as the sum of two squares. Finding these two representations is non-trivial and computationally intensive for large numbers even with high performance computers. We make use of a previously established property that all Pythagorean triples can be represented as $N = n^2 + (n + 2m - 1)^2$ [28]. This equation provides a computationally simpler search using increments of n and fine convergence using m. In this paper, we extend our related works that were reported previously to show that once one sum of the squares is known, it can be used to find the other. We provide our proposed method taking a step by step approach. First, we apply our method to semi-prime factorization of two simple case examples and arrive at a conjecture. Following this, we apply our proposed method to two more large semi-primes as case examples. Finally, we demonstrate in the next section, the breaking of 768-bit RSA using our proposed method as the final result achieved.

Case Example 1: Consider the semi-prime $N = 65 = 1^2 + 8^2 = 4^2 + 7^2$

The semi-prime 65 consists of two primes $(5, 13) \equiv 1\ mod\ 4$

$$5 = 1^2 + 2^2, 13 = 2^2 + 3^2 \Rightarrow \left(1^2 + 2^2\right)\left(2^2 + 3^2\right) = 2^2 + 3^2 + 4^2 + 6^2$$

By applying Brahmagupta identity [49], $65 = (6 \pm 2)^2 + (4 \mp 3)^2 = 8^2 + 1^2 = 4^2 + 7^2$

Case Example 2: Consider the semi-prime $N = 2501 = 1^2 + 50^2 = 10^2 + 49^2$

The semi-prime 2501 consists of two primes $(41, 61) \equiv 1\ mod\ 4$

$$41 = 4^2 + 5^2, 61 = 5^2 + 6^2 \Rightarrow \left(4^2 + 5^2\right)\left(5^2 + 6^2\right) = 20^2 + 24^2 + 25^2 + 30^2$$

By applying Brahmagupta identity [49], $2501 = (30 \pm 20)^2 + (25 \mp 24)^2 = 50^2 + 1^2 = 10^2 + 49^2$

From Case Example 1 and Case Example 2, a general equation can be derived as follows:

$$N = (n^2 + (n+1)^2)((n+1)^2 + (n+2)^2) \tag{1}$$

$$\begin{aligned} N &= \left(2n^2 + 2n + 1\right)\left(2n^2 + 6n + 5\right) \\ &= (n(n+1))^2 + (n(n+2))^2 + (n+1)^4 + ((n+1)(n+2))^2 \\ &= \left(n^2 + n\right)^2 + \left(n^2 + 2n\right)^2 + \left(n^2 + 2n + 1\right)^2 + \left(n^2 + 3n + 2\right)^2 \end{aligned}$$

By using the Brahmagupta identity [49], we get

$$N = \left(\left(n^2+3n+2\right) \pm \left(n^2+n\right)\right)^2 + \left(\left(n^2+2n+1\right) \mp \left(n^2+2n\right)\right)^2$$
$$N = \left(2n^2+4n+2\right)^2 + 1 = (2n+2)^2 + \left(2n^2+4n+1\right)^2$$

$$N = \left(2(n+1)^2\right)^2 + 1 = \left(2(n+1)^2 - 1\right)^2 + \left(2(n+1)\right)^2 \tag{2}$$

For Case Example 1, $n_1 = 1,\ N = 65,\ 65 = 8^2 + 1^2 = 7^2 + 4^2$

For Case Example 2, $n_1 = 4,\ N = 2501,\ 2501 = 50^2 + 1^2 = 49^2 + 10^2$

Let $n_2 = n_1 - 3$.

Considering Case Example 2, we have, $n_1 = 4,\ n_2 = 1,$

$$\Rightarrow N = \left(2(n+4)^2\right)^2 + 1 = \left(2(n+4)^2 - 1\right)^2 + \left(2(n+4)\right)^2 \tag{3}$$

Conjecture: A composite consisting of p unique primes ≡ 1 *mod* 4, has 2^{p-1} sums of squares

Let $p_1 = (n^2 + (n+1)^2)((n+1)^2 + (n+2)^2)$
$= \left(2(n+1)^2\right)^2 + 1 = \left(2(n+1)^2 - 1\right)^2 + \left(2(n+1)\right)^2$
$p_2 = ((n+3)^2 + (n+4)^2)((n+4)^2 + (n+5)^2) = \left(2(n+4)^2\right)^2 + 1 = \left(2(n+4)^2 - 1\right)^2 + \left(2(n+4)\right)^2$
$N = p_1 p_2 = (n^2 + (n+1)^2)((n+1)^2 + (n+2)^2)((n+3)^2 + (n+4)^2)((n+4)^2 + (n+5)^2)$
$N = \left(2\,n^2 + 2\,n + 1\right)\left(2\,n^2 + 6\,n + 5\right)\left(2\,n^2 + 14\,n + 25\right)\left(2\,n^2 + 18\,n + 41\right)$

The four possible combinations are expanded to produce the 4 sums of 4 squares. Using the Brahmagupta identity [49] and Equations (1)–(3), we obtain 8 sums of 2 squares.

$$\begin{aligned} N &= \left(\left(2(n+1)^2\right)^2 + 1\,\right)\left(\left(2(n+4)^2\right)^2 + 1\,\right) \\ &= \left(\left(2(n+1)(n+4)\right)^2\right)^2 + \left(2(n+1)^2\right)^2 + \left(2(n+4)^2\right)^2 + (1)^2 \end{aligned} \tag{4}$$

$$N = \left(\left(2(n+1)(n+4)\right)^2 \pm 1\right)^2 + \left(2\left((n+4)^2 \mp \left(n+1\right)^2\,\right)\right)^2 \tag{5}$$

$$\begin{aligned} N &= \left(\left(2(n+1)^2\right)^2 + 1\,\right)\left(\left(2(n+4)^2 - 1\right)^2 + \left(2(n+4)\right)^2\right) \\ &= \left(\left(2\,(n+1)^2\right)\left(2\,(n+4)^2\ -\ 1\right)\right)^2 + \left(4\,(n+1)^2\,(n+4)\right)^2 \\ &+ \left(2\,(n+4)^2 - 1\right)^2 + \left(2\,(n+4)\right)^2 \end{aligned}$$

$$N = \left(\left(2\,(n+1)^2\right)\left(2\,(n+4)^2\ -\ 1\right) \pm \left(2\,(n+4)\right)\right)^2 + \left(4\,(n+1)^2\,(n+4)\right) \mp \left(2\,(n+4)^2 - 1\right)\right)^2 \tag{6}$$

$$\begin{aligned} N &= \left(\left(2(n+1)^2 - 1\right)^2 + \left(2(n+1)\right)^2\,\right)\left(\left(2(n+4)^2\right)^2 + 1\,\right) \\ &=\ \left(\left(2\,(n+4)^2\right)\left(2\,(n+1)^2\ -\ 1\right)\right)^2 + \left(4\,(n+4)^2\,(n+1)\right)^2 \\ &+\ \left(2\,(n+1)^2 - 1\right)^2 + \left(2\,(n+1)\right)^2 \end{aligned}$$

$$\begin{aligned} N =&\ \left(\left(2\,(n+4)^2\right)\left(2\,(n+1)^2\ -\ 1\right) \pm 2\,(n+1)\right)^2 \\ +&\ \left(4\,(n+4)^2\,(n+1) \mp \left(2\,(n+1)^2 - 1\right)\right)^2 \end{aligned} \tag{7}$$

$$\begin{aligned} N &= \left(\left(2(n+1)^2 - 1\right)^2 + \left(2(n+1)\right)^2\right)\left(\left(2(n+4)^2 - 1\right)^2 + \left(2(n+4)\right)^2\right) \\ &=\ \left(\left(2\,(n+4)^2 - 1\right)\left(2\,(n+1)^2\ -\ 1\right)\right)^2 \\ &+\ \left(\left(2\,(n+1)\right)\left(2\,(n+4)^2\ -\ 1\right)\right)^2 + \left(2(n+4)\left(2\,(n+1)^2 - 1\right)\right)^2 \\ &+\ \left(4(n+1)(n+4)\right)^2 \end{aligned}$$

$$\begin{aligned} N =&\ \left(\left(2\,(n+4)^2 - 1\right)\left(2\,(n+1)^2\ -\ 1\right) \pm 4(n+1)(n+4)\right)^2 \\ +&\ \left(\left(2\,(n+1)\right)\left(2\,(n+4)^2\ -\ 1\right) \mp 2(n+4)\left(2\,(n+1)^2 - 1\right)\right)^2 \end{aligned} \tag{8}$$

In accordance with the conjecture, we can arrive at the following:

By employing Equations (4)–(8), when $p = 4$, the 8 sums of 2 squares can be derived as follows:

$$n = 1,\ N = \left((2(n+1)^2)^2 + 1\right)\left((2(n+4)^2)^2 + 1\right) = (65)(2501) = 162565$$
$$n = 1,\ (400 \pm 1),\ (50 \mp 8) \quad \Rightarrow N = 401^2 + 42^2 = 399^2 + 58^2$$
$$n = 1,\ (392 \pm 10),\ (80 \mp 49) \quad \Rightarrow N = 402^2 + 31^2 = 382^2 + 129^2$$
$$n = 1,\ (350 \pm 4),\ (200 \mp 7) \quad \Rightarrow N = 354^2 + 193^2 = 346^2 + 207^2$$
$$n = 1,\ (343 \pm 40),\ (196 \mp 70) \Rightarrow N = 383^2 + 126^2 = 303^2 + 266^2$$

Let us consider $\sqrt{162565}$ which approximates to 403.

The first 3 sums of two squares can easily be found by decrementing from 403 such that a perfect square remains. By applying Equations (1)–(5), we arrive at the following sequence of steps:

$$162565 - 403^2, 162565 - 402^2 = 31^2, 162565 - 401^2 = 42^2, 162565 - 400^2, 162565 - 399^2 = 58^2$$

In an earlier work [30], the modified Euler factorization was given by $k_{1,2} = \frac{a \pm c}{d - b}$, $(a, c\ even; b, d\ odd)$ Consider the solutions (402,31), (401,42)

$$k_{1,2} = \frac{402 \pm 42}{401 - 31} = \frac{360}{370}, \frac{444}{432} = \frac{36}{37}, \frac{37}{36} \Rightarrow \frac{(36^2 + 37^2)}{(5)(13)(41)} = 1$$

Consider the solutions (401,42), (399,58)

$$k_{1,2} = \frac{58 \pm 42}{401 - 399} = \frac{16}{2}, \frac{100}{2} = \frac{8}{1}, \frac{50}{1} \Rightarrow 8^2 + 1^2 = 65,\ 50^2 + 1^2 = 2501 \Rightarrow (65, 2501)$$

A factorization of $N = 162565 = \left(1^2 + 8^2\right)\left(1^2 + 50^2\right)$

In this way, by using the modified Euler factorization from previous work [30], the factors of a compound number can be found. This can be extended to compound numbers where the factorization is not known.

Factorizing a Semi-Prime

With an increase in the application of number theory in information security, it has drawn the attention of researchers to explore the interesting problem of factorizing a semi-prime, which is a positive integer N that has two prime factors, p_1 and p_2 forming $N_1 = p_1 p_2$. Encryption algorithms such as RSA and public-key cryptosystems rely on special large prime factors of a semi-prime for encoding a sender's message and decoding it at the receiver end. Since only one of the two prime factors of the semi-prime is known at either end of sender and receiver, even if the semi-prime is revealed, an interceptor is required to know both prime factors and only then the message can be decoded. Hence, with the evolution of information and communication technologies, a fast and efficient method factorization of very large semi-primes forms much interest among mathematicians and information security researchers.

With the current state of knowledge, we apply our proposed method of using the sum of squares (SoS) to factorize a semi-prime and illustrate our algorithm with case examples. In broad terms, our proposed algorithm for factorizing a semi-prime consists of three parts as given below:

1. Part 1 of the algorithm, namely new sum of squares (N-SoS1) is proposed in this paper,
2. Part 2 of the algorithm, namely polynomial-based sum of squares (P-SoS2) leveraging on another work [53], and
3. Part 3 of the algorithm, namely a modified Euler-based sum of squares (E-SoS3) factorization using an earlier work [30].

Algorithm of N-SoS1 involves performing the following seven key steps:

Step 1. Generate a special number (N_2) such that $N_2(m, n) = (2n^2 + 2n + 2mn + m + 1)^2 + m^2$

Step 2. Multiply $N_1 \times N_2$, recalling $N_1 = p_1 p_2$
Step 3. Find *Sqrt* $(N_1 \times N_2)$
Step 4. Subtract Integer part of Square root
Step 5. Test for Perfect Square
Step 6. If perfect square recover N-SoS1 for N_1 GOTO P-SoS2
Step 7. If NOT perfect square, Increment (m, n) and GOTO Step 1.

In the above algorithm, we take advantage of N_2 having special attributes which when known allows for the recovery of N-SoS1 using "simple" algebra as given below:

$$N_2(m,n) = (2n^2 + 2n + 2mn + \ m + 1)\ ^2 + m^2$$

This has special properties to be discussed in a future paper. In short, it generates two numbers m, n of the form:

$$N_2(m,n) = \left(n^2 + (n+1)^2\right)\left((n+m)^2 + (n+m+1)^2\right)$$

Numbers whose squares are one apart as per n^2 and $(n+1)^2$ enable special factorizing properties to be maintained. Hence, multiplying a semi-prime (N_1) to be factored by such a number N_2 allows for the recovery of N-SOS1 for N_1. It is conjectured in this paper that by multiplying N_1 by N_2, leads to the determination of N-SoS1 more quickly. The algebra takes the form of 8 possible equations as explained below:

We posit that once an SoS for the product of $N_1 \times N_2$ is found, N-SoS1 for N_1 can be recovered using one of 8 equations.

Let $N_1 \times N_2 = a^2 + b^2$
$a\left(2n^2 + 2n + 2mn + \ m + 1\right) + b(m)$, $a\left(2n^2 + 2n + 2mn + \ m + 1\right) - b(m)$
$b\left(2n^2 + 2n + 2mn + \ m + 1\right) + a(m)$, $b\left(2n^2 + 2n + 2mn + \ m + 1\right) - a(m)$
$a\left(2n^2 + 2n + 2mn + \ m\right) + b(2n + m + 1)$, $a\left(2n^2 + 2n + 2mn + \ m\right) - b(2n + m + 1)$
$b\left(2n^2 + 2n + 2mn + \ m\right) + a(2n + m + 1)$, $b\left(2n^2 + 2n + 2mn + \ m\right) - a(2n + m + 1)$

The result of these equations is then the greatest common divisor (GCD) with $(N_1 \times N_2)$ and if the GCD result is N_2 for any one (or two) of these equations then N-SOS1: $N_1 = c^2 + d^2$ is recovered.

This is the essence of the N-SoS1 algorithm we have proposed in this work.

Once an N-SoS1 for $(N_1 \times N_2)$ becomes known a simple GCD test with a result of N_2 determines which of the 8 equations will yield N-SOS1 for N_1. Only a simple division is then required to yield the N-SoS1 solution for N_1. Once N-SoS1 is known, this is then used to find P-SoS2. Further, once N-SoS1 and P-SoS2 are known, a modified Euler's factorization using sum of squares (E-SoS3) is able to yield p_1 and p_2 and hence factorization is achieved.

A Java implementation of N-SoS1 uses simple arithmetic operations such as +, −, * and / as well as GCD and is provided as Supplementary Resources.

It is possible to avoid P-SoS2 once the first square becomes known as shown in Case Example 3 (below), by decrementing the square from (N_1, N_2). However, Case Example 4 illustrates that even for small numbers not using P-SoS2 creates unpredictability of finding the second SoS. The RSA case illustrates that as per Case Example 3 and 4 a solution exists but poor selection of N_2 leads to an intractable result for RSA768.

Case Example 3: Consider a semi-prime $N_1 = 34121$ along with an upscaling number $N_2 = 162565$.
Let $N = N_1 N_2 = (34121)(162565) = 5546880365$
$\sqrt{5546880365} = 74477$
We provide the results obtained after search iterations as follows:
Iteration 1: $5546880365 - 74477^2$
Iteration 16: $5546880365 - 74461^2 = 1562^2$

Iteration 19: 5546880365 – $74458^2 = 1699^2$ (18 search cycles required)

$$k_{1,2} = \frac{74458 \pm 1562}{74461 - 1699} = \frac{72896}{72762}, \frac{76020}{72762} = \frac{544}{543}, \frac{70}{67} \Rightarrow \frac{544^2 + 543^2}{(5)(13)(61)} = 149,\ \frac{70^2 + 67^2}{(41)} = 229$$

A factorization of $N = 5546880365 = \left(67^2 + 70^2\right)\left(543^2 + 544^2\right)$

From which the factorization of $N_1 = 34121 = (149)(229)$.

Case Example 4: Consider another semi-prime $N_1 = 27161$ along with an upscaling number $N_2 = 162565$.

Let $N = N_1 N_2 = (27161)(162565) = 4415427965$

$\sqrt{4415427965} = 66448$

We provide the results obtained after search iterations as follows:

Iteration 1: 4415427965 – 66448^2

Iteration 14: 4415427965 – $66434^2 = 1397^2$

Iteration 27: 4415427965 – $66421^2 = 1918^2$

Iteration 155: 4415427965 – $66293^2 = 4546^2$ (154 search cycles required)

$$k_{1,2} = \frac{66434 \pm 1918}{66421 - 1397} = \frac{64516}{65024}, \frac{68352}{65024} = \frac{127}{128}, \frac{267}{254} \Rightarrow \frac{127^2 + 128^2}{(13)(41)(61)} = 1,\ \frac{267^2 + 254^2}{(5)} = 27161$$

A factorization of $N = 4415427965 = \left(127^2 + 128^2\right)\left(267^2 + 254^2\right)$

However, this did not lead to the factorization of N_1 and an additional sum of squares is required.

$$k_{1,2} = \frac{66434 \pm 4546}{66293 - 1397} = \frac{61888}{64896}, \frac{70980}{64896} = \frac{967}{1014}, \frac{35}{32} \Rightarrow \frac{967^2 + 1014^2}{(5)(41)(61)} = 157,\ \frac{35^2 + 32^2}{(13)} = 173$$

A factorization of $N = 4415427965 = \left(32^2 + 35^2\right)\left(967^2 + 1014^2\right)$

Hence, from the above, the factorization of $N_1 = 27161 = (157)(173)$.

4. Results

We employ our proposed method to demonstrate the factorization of a large RSA key. The commonly adopted 512-bit RSA key has been attempted to be attacked by successfully completing factorization several times with different factorization methods [22–24]. To take the challenge of RSA key attack further in this work, we consider a higher key size of RSA such as 768-bit RSA key to apply our proposed semi-prime factorization method with the enhanced sum of squares of primes.

4.1. The Case of 768-bit RSA Key (RSA768)

Consider the semi-prime $N_1 = RSA768$

= 1230186684530117755130494958384962720772853569595334792197322452151726400507263
6575187452021997864693899564749427740638459251925573263034537315482685079170261
2214291346167042921431160222124047927473779408066535141959745985690214341 3

$N_2 = 162565$

Let $N = N_1 N_2 = (RSA78)(162565)$

= 1999852983706385928627889129098514647024389405412656004935577244340454022984633 1
6484534813795608287396378274349072065689112828928081750520955869144269989531351 5
4616272689645332522456561509595851329774949472336285352686106163729694393434 5

$\sqrt{(RSA768)(162565)}$

= 1414161583308776610123959976834864514970891887137250625831420984796122007273734 1
77845828533280776844625313347156719273 6

Solutions exist at

$$(1412787730219941840214401773630012877277472789798745428589247690964326418143719
2853505139112349023422853441255921539388,
62320245878612852617150182000501063946977252010614514612703390566686883394487261
8809601622236362159521884255867254901)$$

$$(14112213940567228983701489093327037410260584181290607617784742567737022253008998
279452672940600926940335093953424001948,
911436265626157603651373517639466163137665016626969025415278212470167733473016311
049637630083822731344355566877562379)$$

From the modified Euler factorization of previous work [30], we have

$$k_{1,2} = \frac{a \pm c}{d-b}\ , (a, c\ even\ ;\ b, d\ odd)$$

$a - c$
$= 1412787730219941840214401773630012877277472789798745428589247690964326418143719$
$2853505139112349023422853441255921539388$
$-14112213940567228983701489093327037410260584181290607617784742567737022253008998$
$279452672940600926940335093953424001948$
$= 15663361632189418442528642973091362514143716696846668107734341906241928428194$
$574052466171748096482518347302497537440$
$d - b$
$= 911436265626157603651373517639466163137665016626969025415278212470167733473016$
$311049637630083822731344355566877562379$
$-623202458786128526171501820005010639469772520106145146127033905666868833944872$
$618809601622236362159521884255867254901$
$= 288233806840029077479871697634455523667892496520823879288244306803298899528143$
$692240036007847460571822471311010307478$

$$k_1 = \frac{a-c}{d-b} = \frac{313964076552969675277689223019594269926166116787107977840}{5777499308315937806714612090100083182548223876592695640833}$$

p_1
$= 313964076552969675277689223019594269926166116787107977840^2$
$+5777499308315937806714612090100083182548223876592695640833^2$
$= 33478071698956898786044169848212690817704794983713768568912431388982883793878002$
$287614711652531743087737814467999489$
$p_2 = \frac{RSA768}{N_1}$
$= 36746043666799590428244633799627952632279158164343087642676032283815739666511279$
$233373417143396810270092798736308917$
The factorization of $N_1 = RSA768 = p_1 p_2$

$$\left(\begin{array}{c}3347807169895689878604416984821269081770479498371376856891243138898288379387800228\\7614711652531743087737814467999489\end{array}\right)*$$

$$\left(\begin{array}{c}367460436667995904282446337996279526322791581643430876426760322838157396665112792\\33373417143396810270092798736308917\end{array}\right)$$

The RSA case illustrates that a solution exists but poor selection of N_2 leads to an intractable result for RSA768. The selection of m, n of N_2 (m, n) can be better determined. As presented in this paper, as n is incremented, a range of m values is searched through. This is continued until a perfect square is found. The attributes of m have been determined via experimentation. The authors are currently characterizing n with a view to reducing the search field for a suitable N_2. This is an area of ongoing research.

4.2. *Performance and Comparison*

In Table 1, we summarize the cost performance of our proposed method of the extended sum of squares approach to factorize semi-primes of different lengths (key sizes). For the various case examples of semi-primes including RSA768 that were computed in this work, we provide the cost factors of memory used and the non-optimized search time taken in terms of decrement loops of the square for completing the semi-prime factorization using our method. If a linear search is used from the square root of $N = N_1 N_2$, the number of iterations is given thus. However, this can be significantly reduced by using $N = n^2 + (n + 2m - 1)^2$ congruent to 1 *mod* 4.

Table 1. A performance comparison of proposed semi-prime factorization for different lengths.

Semi-Prime	**Search Cycle Time ***	**Memory (Bytes)**
N = 65	1	1
N = 2501	1	2
N = 162,565	4	2
N = 5,546,880,365	18	5
N = 4,415,427,965	154	4
N = (RSA78)(162,565)	$2.94 \ldots \times 10^{115}$	56

* decrement loops of the square (non-optimized search).

Table 1 demonstrates that the memory required for our algorithm is minimal and the search cycle time is the main factor that needs to be contained for large RSA keys. It is important to note that each search cycle refers to only six steps involving rudimentary algebra consisting of multiplication, addition, subtraction and division operations exhibiting very low processing time. Next, we elaborate on the complexity of our algorithm and its comparison to other factoring algorithms.

4.2.1. Complexity of the Algorithm

We summarize the complexity of our proposed semi-prime factorization algorithm in terms of memory and computational time. These complexity measures are followed in similar lines to existing methods reported in the literature [55].

The memory requirement of our algorithm is very minimal with most computations operating on the accumulator (using BigInteger arithmetic). The number of memory variables used for each major part and step involved in our algorithm are given below:

The manipulation of $(m,\ n)$ to generate N_2 requires the use of three variables.
The resultant SoS $(m,\ n)$ requires the use of two variables.
The recovery $N_1(m,\ n)$ requires the use of two variables.
Hence, these requirements occupy 7 BigInteger values in memory.

Studying the time complexity of our proposed algorithm, we find that each of the algorithm steps is rudimentary. Steps 3 and 5 use a *Sqrt* function, which is the most expensive function in this algorithm. Only the integer part is considered so a Newton–Rapson algorithm is used here. Step 6 uses a GCD function. The remaining parts are simply multiplication, addition, subtraction and division operations that require minimal processing time.

4.2.2. Comparison to other Factoring Algorithms

Part 1 of our proposed factorization algorithm (N-SoS1) currently requires two input variables $(m,\ n)$. n is usually small and $= \ f(n)$: $m = 2n^2 + 2n + 1$.

The current algorithm calculates m and each cycle of n creates a larger and larger range for m. $m\ (max)\ =\ 25$ when $\ n = 3$. For $n = 13$, $\ m = 365$.

Currently, a brute force search is conducted from $(m,\ n)\ =\ (1, 1)$ until an SoS is found. This stochastic search is no better than Fermat's factorization method. However, the following Case

Example 5 illustrates the gain in the search cycle time performance of our proposed factorization method when m is contained within a search field.

Case Example 5: For $N_1 = 21751495009,\ n = 3,\ m = 14$ when defined achieves the result in much shorter overall search cycle time.

A result for $N_2(14, 3)$ = 15325 generates the perfect square required.
From this, an SoS of $N_3(18257646, 4853) = 333341661012925$ is obtained.
Further, N-SoS1 = $N_1(145353, 24980)$
P-SoS2 = $N_1(132340, 65097)$
$p_1, p_2\ = \frac{(132340 \pm 24980)}{(145353 - 65097)}$ (using E-SoS3)
$p_1\ = \frac{305}{228},\ p_2\ = \frac{345}{176}$
$p_1\ =\ 228^2 + 305^2\ =\ 145009\ p_2\ =\ 176^2 + 345^2\ =\ 150001$
$N_1\ =\ p_1\, p_2\ =\ (145009)(150001)$

Continued research is underway to characterize (m, n) so that a reduced stochastic search can be undertaken in a reduced search field. Our proposed semi-prime factorization method is simple to understand and to implement. It uses rudimentary algebra of multiplication, addition, subtraction, and division operations that require minimal processing time unlike other algorithms that focus on lattice reduction (LR) techniques requiring time consuming operations. While this work is limited to an empirical study of our proposed method, formal proofs are quite extensive and are reserved for future work.

5. Discussion

By factoring the modulus of an RSA private key an attacker can compute the corresponding public key or vice versa. Various studies have surveyed RSA key size and limitations across public key infrastructure and new methods for attacking commonly used 512-bit and 768-bit RSA keys are continuing to interest researchers [19,56]. Most of the previous studies have studied attacks on RSA cryptosystems with specific focus. For instance, partial key exposure (PKE) attacks were studied in [57,58]. A PKE attack on low public-exponent RSA key of a variant of the RSA public-key cryptosystem was found to be less effective than for standard RSA. However, the large public exponent RSA key was reported to be more vulnerable to such attacks than for standard RSA. Some generalized studies have also been conducted comprehensively at a practical Internet scale to survey vulnerabilities of RSA key generation for various protocols such as SSH, HTTPS, and DKIM [19–21,59]. However, in this paper, we have considered the number theoretic properties of semi-primes that form the underlying principle behind any RSA key generation to be the fundamental cause for any RSA attack. For instance, by choosing an RSA modulus with a small difference of its prime factors, private key attacks could be improved [60]. Some early studies have considered polynomial equations to study low exponent RSA vulnerabilities and their variants [61,62]. Recent number theory-based studies in this domain have focused on lattice reduction (LR) techniques with prime number theory to study RSA key attacks [31,38,39]. Such LR based techniques come under the general category of Coppersmith attacks. However, the uniqueness of our contribution is that we propose factorization using the sum of squares that is more in line with Euler's sum of squares approach rather than Coppersmith's approach.

Whilst there is active research in LR methods, sum of squares remains an area of interest and continues to yield surprises, and it has been the motivation of this work. The subtleties in the value of N_2 are an area of interest that we explored in this study. The value of N_2 in this paper has special properties, which are described briefly. N_2 was constructed with two semi-primes. Each of the primes has special properties. They are all congruent to $1\ mod\ 4$. Their squares are one apart. In the case of both semi-primes, the highest square of the smallest prime was the lowest square in the larger prime. Both semi-primes are perfect squares when 1 was subtracted. Multiplying the two semi-primes together created 8 sums of 2 squares, three of which are very close together (402, 401, 399). When such a number is multiplied with a semi-prime to be factored, 32 sums of 2 squares are available. As was shown previously, only two sums of two squares are required to factor the semi-prime in question. This increased the likelihood of finding two such sums of squares. Equations (4)–(8) describe the general form for N_2. In this case $N_2 = (65)(2501) = 162565$.

It would appear counter-intuitive to make the semi-prime to be factored (N_1) larger, however this is offset by the likelihood of finding two sums of two squares near the square root of $N = N_1 N_2$. In the case of RSA768 the first solution was some way off from the square root. The search can be restricted by only considering congruent co-prime sums of squares that are consistent with a previous study [30]. In this way only sums of two squares which approximate N need be tested and of those, only the ones equal to N yield a valid solution. This substantially reduces the search cost. Since a linear search is conducted, the search field can easily be divided up and this lends itself very well to parallel processing. The mathematics in our approach is rudimentary, consisting of multiplications, additions and subtractions. The perfect square test does require a square root which is computationally costly, but this too can be avoided if the approach in [30] is used to find solutions equivalent to $N = n^2 + (n + 2m - 1)^2$ all of which are congruent to 1 *mod* 4.

Overall, the search area to find the first sum of squares is key to finding the second. Hence, the search is focused on finding the first sum of two squares. The relationship between the two squares of the sum of two squares of the semi-prime N_1 determines the spread of, and the likelihood of, finding the first sum of two squares for N. Table 1 shows the search cycle time for different key sizes explored in this work. The search area can be limited by $\sqrt{N-1}$ and $\sqrt{N/2}$. This can be narrowed by considering the properties of N_2 and the closeness of the sums of squares of N_2 $(402, 401, 399)$. The smearing of N_1 across N by N_2 determines the search field. This field is then divided over a number of parallel processes running a very simple (but fast) $n^2 + (n + 2m - 1)^2$ algorithm. This search field to find the first sum of two squares is an area of ongoing research.

6. Conclusions and Future Work

We studied the semi-prime factorization and the impact on the security of the RSA cryptosystem. We proposed a novel extension to our previously established methods of semi-prime factorization using a sum of squares approach for cryptanalytic attacks on RSA modulus $N = p_1\, p_2$ with N being a large semi-prime, constituting two primes p_1 and p_2. We gradually developed our proposed technique by providing illustrations of semi-prime factorization for small and large numbers. We arrived at the conjecture that a composite consisting of p unique primes $\equiv 1$ *mod* 4, has 2^{p-1} sums of squares. We provided the detailed steps involved in the implementation of our enhanced semi-prime factorization algorithm. We then applied our proposed factorization algorithm to attack 768-bit RSA successfully. Finally, we discussed the strengths and limitations of our algorithm by providing the complexity of the algorithm in terms of memory and search cycle time as well as its comparison to other factorization algorithms.

In future, a more extensive comparison of various RSA keys adopted in cryptosystems will be studied for key attacks using our method as against other contemporary methods such as Shor's algorithm. Our ongoing research will also focus on characterising a reduced search field in arriving at the semi-prime factorization result and the formal theoretical proofs.

Author Contributions: Conceptualization, A.O. and S.V.; methodology, A.O.; validation, A.O.; resources, S.V.; data curation, A.O.; writing—original draft preparation, A.O. and S.V.; writing—review and editing, S.V. All authors have read and agree to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
2. Ambedkar, B.R.; Bedi, S.S. A New Factorization Method to Factorize RSA Public Key Encryption. *Int. J. Comput. Sci. Issues* **2011**, *8*, 242–247.
3. Yan, S.Y. Factoring Based Cryptography. In *Cyber Cryptography: Applicable Cryptography for Cyberspace Security*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 217–286. [CrossRef]
4. Karatsuba, A. The complexity of computations. *Proc. Steklov Inst. Math.* **1995**, *211*, 169–183.
5. Traversa, F.L.; di Ventra, M. Polynomial-time solution of prime factorization and NP-complete problems with digital memcomputing machines. *Chaos Interdiscip. J. Nonlinear Sci.* **2017**, *27*, 023107. [CrossRef]
6. Crandall, R.; Pomerance, C.B. *Prime Numbers: A Computational Perspective*; Springer Science & Business Media: New York, NY, USA, 2005.
7. Goldwasser, S.; Micali, S.; Rivest, R. A Digital Signature Scheme Secure against Adaptively Chosen Message Attacks. *SIAM J. Comput.* **1988**, *17*, 281–308. [CrossRef]
8. Kobliz, N. *A Course in Number Theory and Cryprography*; Springer: Berlin, Germany, 1994; ISBN 3-578071-8.
9. Overmars, A.; Venkatraman, S. A new method of golden ratio computation for faster cryptosystems. In Proceedings of the IEEE Cybersecurity and Cyber forensics Conference, London, UK, 21–23 November 2017.
10. Dubey, M.K.; Ratan, R.; Verma, N.; Saxena, P.K. Cryptanalytic Attacks and Countermeasures on RSA. In *Proceedings of the Third International Conference on Soft Computing for Problem Solving*; Springer: New Delhi, India, 2014; pp. 805–819.
11. Nastase, L. Security in the Internet of Things: A Survey on Application Layer Protocols. In Proceedings of the 21st International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 29–31 May 2017; pp. 659–666.
12. Aboud, S.J. An efficient method for attack RSA scheme. In Proceedings of the ICADIWT 2nd International Conference, London, UK, 4–6 August 2009; pp. 587–591.
13. Clark, J.; van Oorschot, P.C. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP), Berkeley, CA, USA, 19–22 May 2013; pp. 511–525.
14. Venkatraman, S.; Overmars, A. New method of prime factorisation-based attacks on RSA Authentication in IoT. *Cryptography* **2019**, *3*, 20. [CrossRef]
15. Wiener, M. Cryptanalysis of short RSA secret exponents. *IEEE Trans. Inf. Theory* **1990**, *160*, 553–558. [CrossRef]
16. Boneh, D. Twenty years of attacks on the RSA cryptosystem. *Not. Am. Math. Soc. (AMS)* **1999**, *46*, 203–213.
17. Kamel Ariffin, M.R.; Abubakar, S.I.; Yunos, F.; Asbullah, M.A. New Cryptanalytic Attack on RSA Modulus N = pq Using Small Prime Difference Method. *Cryptography* **2019**, *3*, 2. [CrossRef]
18. Suárez-Albela, M.; Fraga-Lamas, P.; Fernández-Caramés, T.M. A Practical Evaluation on RSA and ECC-Based Cipher Suites for IoT High-Security Energy-Efficient Fog and Mist Computing Devices. *Sensors* **2018**, *18*, 3868. [CrossRef]
19. Valenta, L.; Cohney, S.; Liao, A.; Fried, J.; Bodduluri, S.; Heninger, N. Factoring as a Service. In Proceedings of the International Conference on Financial Cryptography and Data Security, Sliema, Malta, 3–7 April 2017; pp. 321–338.
20. Messaging, Malware and Mobile Anti-Abuse Working Group. M3AAWG Best Practices for Implementing DKIM to Avoid Key Length Vulnerability. Available online: https://www.m3aawg.org/sites/default/files/m3aawg-key-implementation-bp-revised-2017-07.pdf (accessed on 26 September 2020).
21. Durumeric, Z.; Kasten, J.; Bailey, M.; Halderman, J.A. Analysis of the HTTPS certificate ecosystem. In Proceedings of the 13th Internet Measurement Conference, Barcelona, Spain, 23–25 October 2013.
22. Zetter, K. How a Google Headhunter's E-Mail Unraveled a Massive Net Security Hole. Available online: https://www.wired.com/2012/10/dkim-vulnerability-widespread/ (accessed on 26 September 2020).
23. Cavallar, S.; Dodson, B.; Lenstra, A.K.; Lioen, W.; Montgomery, P.L.; Murphy, B.; Te Riele, H.; Aardal, K.; Gilchrist, J.; Guillerm, G.; et al. Factorization of a 512-bit RSA modulus. In Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, 14–18 May 2000; pp. 1–18.

24. Kleinjung, T.; Lenstra, A.K.; Page, D.; Smart, N.P. Using the cloud to determine key strengths. In Proceedings of the 13th International Conference on Cryptology in India, Kolkata, India, 9–12 December 2012; pp. 17–39.
25. McKee, J. Turning Euler's factoring method into a factoring algorithm. *Bull. Lond. Math. Soc.* **1996**, *28*, 351–355. [CrossRef]
26. Overmars, A.; Ntogramatzidis, L. A new parameterisation of Pythagorean triples in terms of odd and even series. *arXiv* **2015**, arXiv:1504.03163.
27. McKee, J.; Pinch, R. Old and new deterministic factoring algorithms. *Algorithm. Number Theory* **2005**, 217–224. [CrossRef]
28. Overmars, A.; Ntogramatzidis, L.; Venkatraman, S. A new approach to generate all Pythagorean triples. *AIMS Math.* **2019**, *4*, 242–253. [CrossRef]
29. Knill, O. Some experiments in number theory. *arXiv* **2016**, arXiv:1606.05971.
30. Overmars, A.; Venkatraman, S. A Fast Factorisation of Semi-Primes Using Sum of Squares. *Math. Comput. Appl.* **2019**, *24*, 62. [CrossRef]
31. Overmars, A.; Venkatraman, S. Pythagorean-Platonic lattice method for finding all co-prime right angle triangles. *Int. J. Comput. Inf. Eng.* **2017**, *11*, 1192–1195.
32. Bach, E.; Miller, G.; Shallit, J. Sums of divisors, perfect numbers and factoring. *SIAM J. Comput.* **1986**, *15*, 1143–1154. [CrossRef]
33. Overmars, A. Survey of RSA Vulnerabilities. In *Modern Cryptography—Theory, Technology, Adaptation and Integration*; IntechOpen: London, UK, 2019.
34. Durumeric, Z.; Wustrow, E.; Halderman, J.A. ZMap: Fast Internet-wide Scanning and Its Security Applications. In Proceedings of the 22nd USENIX Security Symposium, Washington, DC, USA, 14–16 August 2013.
35. Nitaj, A. Another Generalization of Wieners Attack on RSA. In Proceedings of the First International Conference on Cryptology in Africa, Casablanca, Morocco, 11–14 June 2008; Springer: Berlin, Germany, 2008; Volume 5023, pp. 174–190.
36. Boneh, D.; Durfee, G. Cryptanalysis of RSA with Private Key D Less than $N^{0.292}$. In Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, 2–6 May 1999; Springer: Berlin, Germany, 1999; Volume 1592, p. 111.
37. Blomer, J.; May, A. Low Secret Exponent RSA Revisited. In *Cryptography and Lattice*; Springer: Berlin, Germany, 2001; Volume 2146, p. 419.
38. Coppersmith, D. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comput.* **1994**, *62*, 333–350. [CrossRef]
39. Kameswari, P.A.; Jyotsna, L. An Attack Bound for Small Multiplicative Inverse of φ(N) mod e with a Composed Prime Sum p + q Using Sublattice Based Techniques. *Cryptography* **2018**, *2*, 36. [CrossRef]
40. Lenstra, A.K.; Lenstra, H.W., Jr.; Manasse, M.S.; Pollard, J.M. The Number Field Sieve. Available online: https://wstein.org/129/references/Lenstra-Lenstra-Manasse-Pollard-The%20number%20field%20sieve.pdf (accessed on 26 September 2020).
41. Grosswald, E. *Representations of Integers as Sums of Squares*; Springer: Berlin, Germany, 1985.
42. Zagier, D. A one-sentence proof that every prime p ≡ 1 (mod 4) is a sum of two squares. *Am. Math. Mon.* **1990**, *97*, 144.
43. Jackson, T. *From Polynomials to Sums of Squares*; CRC Press: London, UK, 1995.
44. Moreno, C.J.; Wagstaff, S.S. *Sums of Squares*; Chapman & Hall/CRC: London, UK, 2005.
45. Dickson, L.E. *History of The Theory of Numbers: Diophantine Analysis*, 2nd ed.; Dover Publications: Mineola, NY, USA, 2005.
46. Marshall, M. *Positive Polynomials and Sums of Squares*; American Mathematical Society: Providence, RI, USA, 2008.
47. Williams, K. *Number Theory in the Spirit of Liouville*; London Mathematical Society: London, UK, 2011.
48. Roy, T.; Soni, F.J. A direct method to generate Pythagorean triples and its generalization to Pythagorean quadruples and n-tuples. *arXiv* **2012**, arXiv:1201.2145.
49. Li, S. *The Sum of Two Squares*; Cornell University Press: Ithaca, NY, USA, 2013.
50. Kostopoulos, G.L. An Original Numerical Factorization Algorithm. *J. Inf. Assur. Cyber Secur.* **2016**, *2016*, 775081. [CrossRef]
51. Kaddoura, I.; Abdul-Nabi, S.; Al-Akhrass, K. New Formulas for Semi-Primes. Testing, Counting and Identification of the *n*th and Next Semi-Primes. *arXiv* **2016**, arXiv:1608.05405.

52. Hiary, G.A. A Deterministic Algorithm for Integer Factorization. *Math. Comput.* **2016**, *85*, 2065–2069. [CrossRef]
53. Overmars, A.; Venkatraman, S. New Semi-prime Factorization Using Pythagorean Quadruples and Triples for the Security of RSA. *Comput. Math. Methods* **2020**. submitted.
54. Bell, E.T. *The Prince of Amateurs: Fermat*; Simon and Schuster: New York, NY, USA, 1986; pp. 56–72.
55. Kloster, K. Factoring a Semiprime n by Estimating $\varphi(n)$. Available online: http://www.gregorybard.com/papers/phi_version_may_7.pdf (accessed on 26 September 2020).
56. Kleinjung, T.; Aoki, K.; Franke, J.; Lenstra, A.K.; Thomé, E.; Bos, J.W.; Gaudry, P.; Kruppa, A.; Montgomery, P.L.; Osvik, D.A.; et al. Factorization of a 768-bit RSA modulus. In Proceedings of the 30th Annual Cryptology Conference, Santa Barbara, CA, USA, 15–19 August 2010; pp. 333–350.
57. Blömer, J.; May, A. New Partial Key Exposure Attacks on RSA. In Proceedings of the 23rd Annual Crypto Conference, Santa Barbara, CA, USA, 17–21 August 2003; pp. 27–43.
58. Steinfeld, R.; Zheng, Y. On the Security of RSA with Primes Sharing Least-Significant Bits. *Appl. Algebra Eng. Commun. Comput.* **2004**, *15*, 179–200. [CrossRef]
59. Heninger, N.; Durumeric, Z.; Wustrow, E.; Halderman, J.A. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In Proceedings of the 21st USENIX Security Symposium, Bellevue, WA, USA, 8–10 August 2012.
60. De Weger, B. Cryptanalysis of RSA with Small Prime Difference. *Appl. Algebra Eng. Commun. Comput.* **2002**, *13*, 17–28. [CrossRef]
61. Coppersmith, D. Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. *J. Cryptol.* **1997**, *10*, 233–260. [CrossRef]
62. Jochemsz, E.; May, A. A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variant. In Proceedings of the 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, 3–7 December 2006; Springer: Berlin, Germany, 2006; Volume 4284, pp. 267–282.