ICIC Express Letters
Volume **17**, Number **7**, July **2023**

 ISSN 1881-803X
pp. **825–831**

HACKING OF RSA CRYPTOGRAPHY ALGORITHM USING PYTHAGOREAN AND QUADRATIC EQUATION WITH MICROSOFT EXCEL SOLVER

Martin Suhartana[1] and Aryan Wibowo[2]

[1]Computer Science Department, BINUS Online Learning
Bina Nusantara University
Jl. K. H. Syahdan No. 9, Kemanggisan, Palmerah, Jakarta 11480, Indonesia
martin.suhartana@binus.ac.id

[2]Computer Science Department, Computer Science Program
Bina Nusantara University
Jalan Raya Kebon Jeruk No. 27, Kebon Jeruk, Jakarta Barat, Jakarta 11530, Indonesia
aryan_wibowo@yahoo.com

Received July 2022; accepted October 2022

Abstract. *Cryptography is well recognized as one of the most adopted data security methods. Since the designers of Rivest-Shamir-Adleman (RSA) invented a reliable encryption and description technique that is claimed to be difficult to crack, the point of difficulty lied in the factoring process $n - p * q$ (from RSA). This paper attempted to demonstrate that factorization could be accomplished by combining two mathematical equations, namely Pythagorean and quadratic equations. The factorization process of the two equations has been proven using the Microsoft Excel Solver feature. This method is an inspiration for future research in developing more proven process using larger numbers as required by RSA, which could not be performed in Excel due to technical software constraint. Future research should utilize more sophisticated tools with adequate capability to crunch large numbers using server instead of desktop application. Simulation on this research could factorize RSA with limitation of maximum $n = 15$ digits.*
Keywords: Cryptography algorithm, Quadratic equation, Pythagorean, Microsoft Excel Solver

1. **Introduction.** *Cryptography*, as is well known, is one of the most extensively utilized methods of data security. With *cryptography*, you can get services like, confidentiality, data integrity, authentication, and non-repudiation [1,12]. Whitfield Diffie and Martin Hellman of Stanford University introduced public key *cryptography* in 1975, then three people, namely *Ron Rivest, Adi Shamir, and Len Adleman* of Massachusetts Institute of Technology, described it in 1977, and it was patented in 1983 by the Massachusetts Institute of Technology in the United States, known as RSA (which comes from the initiation of their names *Rivest-Shamir-Adleman*). To generate n, you will need two prime integers, p and q (from RSA) [1,2,7,10,11,14,15].

We studied the former research that suggested several methods of sum square [3], Pythagorean [5], SAT solver [13], IMFv3 [14], and then performed combination of 2 equations of quadrant and Pythagorean which were tested using MS Solver. The outcome has exhibited a possibility in cracking RSA algorithm security.

Then, Euler's Theorem to generate a key pair, public key e and private key d by $(d * e) \bmod \varphi(n) = 1$, is then used to construct a key pair, public key e and private key d. As a result, the key asymmetric key might be used to convert a plaintext communication to an encrypted message (ciphertext), and then descripted message back to plaintext [1-3,7,10,11]. The problem to be able to open a ciphertext message for a hacker is the value

DOI: 10.24507/icicel.17.07.825

of n (from RSA), by factoring it first to get the values of $p-1$ and $q-1$ so that they can disassemble the asymmetric key pair [2-6].

This case study has demonstrated how to utilize two equations to obtain the values of p and q, as well as how to disassemble using Ms Excel Solver.

2. RSA, Pythagorean, and Quadratic Equation.

It is well known that the most difficult challenge of cracking RSA is the factoring procedure. The purpose of this study was to demonstrate how factorization could be accomplished by combining two mathematical equations, which were Pythagorean and quadratic equations. The attempts of factoring n mathematically were successful.

2.1. RSA.

The RSA model is named after the algorithm of two random prime number variables that produces the multiplication of $n = p*q$, and Euler's Theorem to generate an asymmetric key pair. In general, RSA has the following variables with Table 1 [1,2,6-9,14].

Table 1. Component RSA variable

Var	Definition	Status	Remark
P	Prime	Private	Generated n from $p*q$
Q	Prime	Private	Generated n from $p*q$
N	One parameter public as key public for generating (M = message) to (C = ciphertext) and reversing (C = ciphertext) to (M = message)	Public	n as key public to generate M to C, and C to M
$\varphi(n)$	Relative n to get d (description key)	Private	Variable $r = (p-1)*(q-1)$. Call phi [φ], and write the symbol with code 03c6 (lowercase), 03A6 (uppercase), then $alt + x$.
E	One parameter as public key (prime)	Public	Encryption key
D	One parameter as private key (prime)	Private	Decryption key
M	Message plaintext or descripted	Private	Message in plaintext
C	Message ciphertext or encrypted	Public	Message in ciphertext

The RSA algorithm generates variables with private and public statuses, which means that some variables will be published, and others will not be published as private values. The variable n is an important value in the RSA model, as it is one of the key values of RSA *cryptography*. RSA *cryptography*, on the other hand, necessitates the use of two keys to transform a message (M) to ciphertext (C) using the public key $\{n, e\}$, and vice versa. Ciphertext (C) is transformed back into message (M) using the public and private keys $\{n, d\}$ – also known as asynchronous keys of RSA [1,3-6,8-11]. Here is the formula for encrypting and describing messages (M) with the RSA values $\{n, e\}$ and $\{n, d\}$:

a) Encryption message $(C) = m^e \bmod n$

b) Description message $(M) = c^d \bmod n$

c) Generating key pair, public key e and private key d by $(d*e) \bmod \varphi(n) = 1$, where $1 < e < \varphi(n)$ with e and $\varphi(n)$ being coprime.

Figure 1, illustrating the encryption process and cryptographic algorithm decryption work asymmetrically, explains clearly how the authors carried out the research. The method must describe the research design clearly, the replicable research procedures, and describe how to summarize and analyze the data.

The notion depicted in Figure 1 is used to safeguard the exchange of two communicating entities. For example, Alice communicates with Bob. Bob chooses the key pair (e and d). Bob delivers the encryption key e (public key) to Alice over any channel but keeps the decryption key hidden, d (private key). Then Alice wishes to send a message (M)

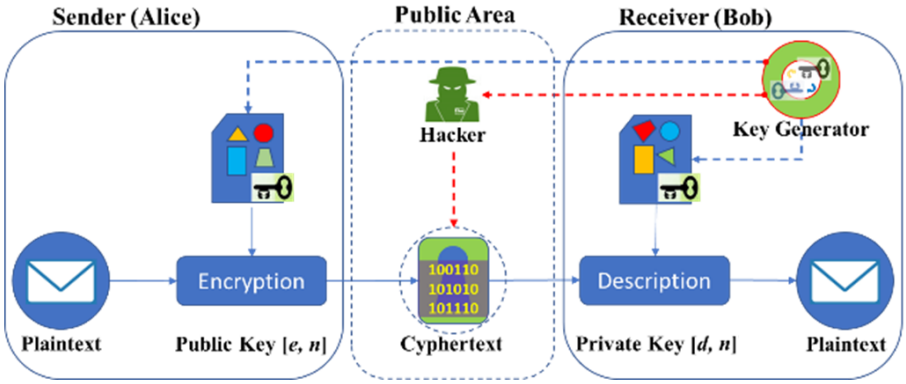


FIGURE 1. Asymmetric key cryptography

using Bob's public key to obtain $(C) = m^e \bmod n$ and then send C as a communication channel (which needs not be secure). Bob decrypts the ciphertext C with his private key to obtain $(M) = c^d \bmod n$ [1,6,8-11].

2.2. **Pythagorean.** Fermat's Christmas theorem on the sum of two squares demonstrated that there are two sums of two squares representations, and that Euler's factorization can be used [3]. Similarly, Pythagorean is defined as the triple variables a, b, and c that form an angled triangle. In statement (2) $c^2 - b^2 + a^2$ substitution $a^2 - c^2 - b^2$, right triangle is formed by the sum of two squares, in which formula is obtained by finding variable c where $c^2 = \approx \sqrt{(n+i)}$ and $i = i + 1$. It is the same as $n = p * q = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2$ equal to $n = x^2 - y^2$. In addition, shown by Figure 2, right triangle and three squares are explained in statements (1)-(5) as follows.

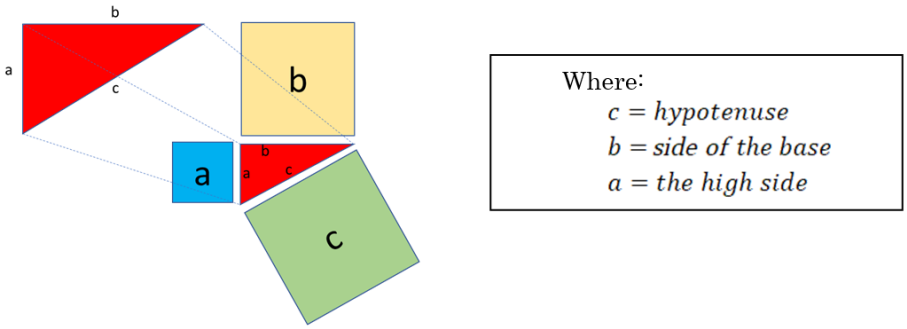


FIGURE 2. Right triangle and three squares

2.3. **Quadratic equation.** Furthermore, $(x - p)(x - q)$ is an quadratic equation on the right side of statement (1) below [3,5]. The quadratic equation is one in which the variable with the greatest power equals two variables, p and q. The following is the generic form of the quadratic equation which is written as follows:

$$a \cdot x^2 - b \cdot x + c = 0 \tag{1}$$

where a = coefficients of equation, for this case as 1; b = coefficients of equation, for this case as $(p + q)$; c = constant, for this case as $n = p * q$.

The difficulty in completing quadratic equations was determining variable b, which might be derived from the Pythagorean formula acquired by finding variable c as follows:

Since quadratic equation: $a \cdot x^2 - b \cdot x + c = 0$, where $c = n$

Statement (2) for Pythagorean:

$$a^2 = c^2 - b^2, \text{ where } b^2 = n \tag{2}$$

In statement (3), get variable c from Pythagorean, where a and c are positive integers:

$$c^2 =\approx \sqrt{(n+i)}, \text{ where } i = i+1 \tag{3}$$

Quadratic equation will then be derived from Pythagorean variable c

$$b = 2 * c \tag{4}$$

Remark 2.1. *The variable c has been confirmed by Pythagorean testing, and the result of variable c may be multiplied by 2, which is equivalent to variable b in the quadratic equation as stated in statement (4). The probability experiment is presented in statement (3) with integer data type limitation for a and c, which is used to determine the c variable.*

Therefore, since the quadratic equation has been finished, it could be determined that the factorization results of the p and q of the quadratic equation were possible to be discovered. In order to get the p and q values, the *Quadratic Formula* could be used to find the values of x_1 and x_2, following the formula in statement (5):

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{5}$$

3. **Results.** Both equations have been shown in the scenario above by various tests and simulations that have managed to acquire p and q, with Fermat's statement $x^2 - y^2 = (x - p)(x - q)$ [3]. It was recognized that calculating the factorization of two prime numbers has become a mathematical difficulty that necessitates a large amount of computing resources. In statement (2), $a^2 = c^2 - b^2$ was obtained after factorizing RSA of n – to extract p and q values using the following formulas: $p = c + a$ and $q = c - a$. Another method for factorization is to take the quadratic equation in statement (1) with the limitation of statement (4) where variable b is twice of variable c of Pythagorean, the factorization n of RSA to obtain p and q values with the following statement (5). Figure 3 below depicts what has been tested and simulated.

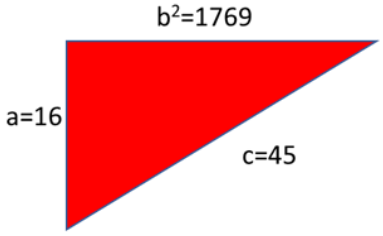


FIGURE 3. Right triangle simulation result

Please see the following Table 2 and Table 3 for additional experiment outcome as demonstrated by any number of variable n optioning p and q with both equations.

TABLE 2. Self-processed data by Pythagorean

No.	n	Polarization i for c	Pythagorean			
			$c^2 - n = a^2$		$p = c + a$	$q = c - a$
			Getting c	Getting a	Getting p	Getting q
1	1769	3	45	16	61	29
2	162733	4	407	54	461	353
3	4394647	848	2944	2067	5011	877
4	682690031	21932	48060	40337	88397	7723

TABLE 3. Self-processed data by quadratic equation

No.	n	Polarization i for j	Quadratic equation			
			$a \cdot x^2 - b \cdot x + c = 0$		$x_{1,2} = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$	
			Getting b	Getting c	Getting p	Getting q
1	1769	3	90	1769	61	29
2	162733	4	814	162733	461	353
3	4394647	848	5888	4394647	5011	877
4	682690031	21932	96120	682690031	88397	7723

Clearly, we could obtain the values of p and q from the two equations above; hence, those formulas might be utilized as process inputs in Ms Excel Solver. The simulation results executed in Ms Excel Solver were recorded in Table 4.

TABLE 4. Self-processed data by Ms Excel Solver

Pythagorean $c^2 = b^2 + a^2$			**Quadratic equation** $a \cdot x^2 - b \cdot x + c = 0$				**Result (Output)**		
b	c	a	a	b	c	Object	c	p	q
$(n = p * q)$	(Ms Excel Solver)	$a^2 = c^2 - b^2$	Default $a = 1$		$(n = p * q)$	$a \cdot x^2 - b \cdot x + c = 0$	(Ms Excel Solver)	(Ms Excel Solver)	(n/p)
Input n	Found	Verified	Input	Input $2 \times c$ (c from Pythagorean)	Input n	Input formulas	Found	Found	Found
1769	45	16	1	90	1769	0	45	29	61
162733	407	54	1	814	162733	0	407	353	461
4394647	2944	2067	1	5888	4394647	0	2944	877	5011
682690031	48060	40337	1	96120	682690031	0	48060	7723	88397

To execute the solver, the following configuration steps must be performed in Ms Excel Solver.

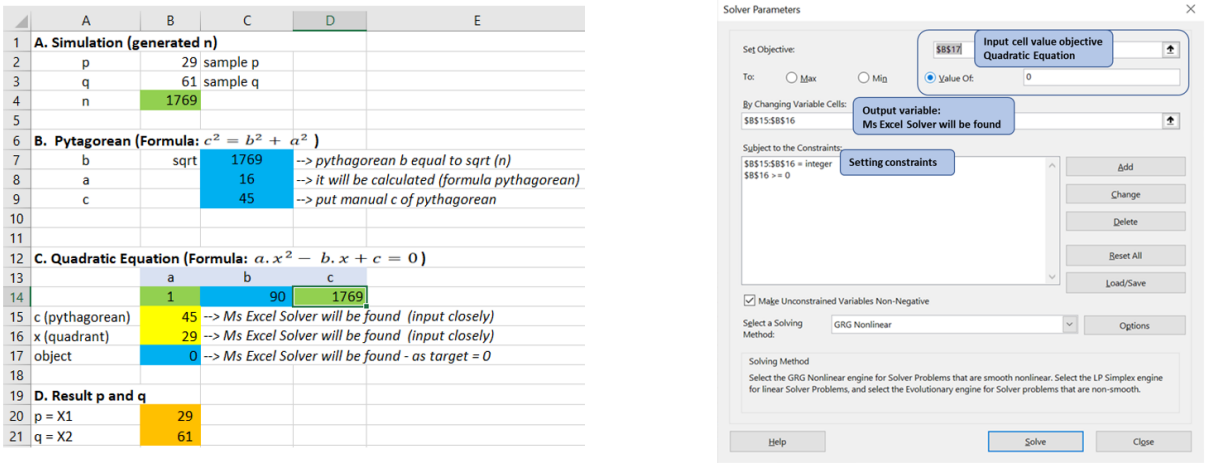
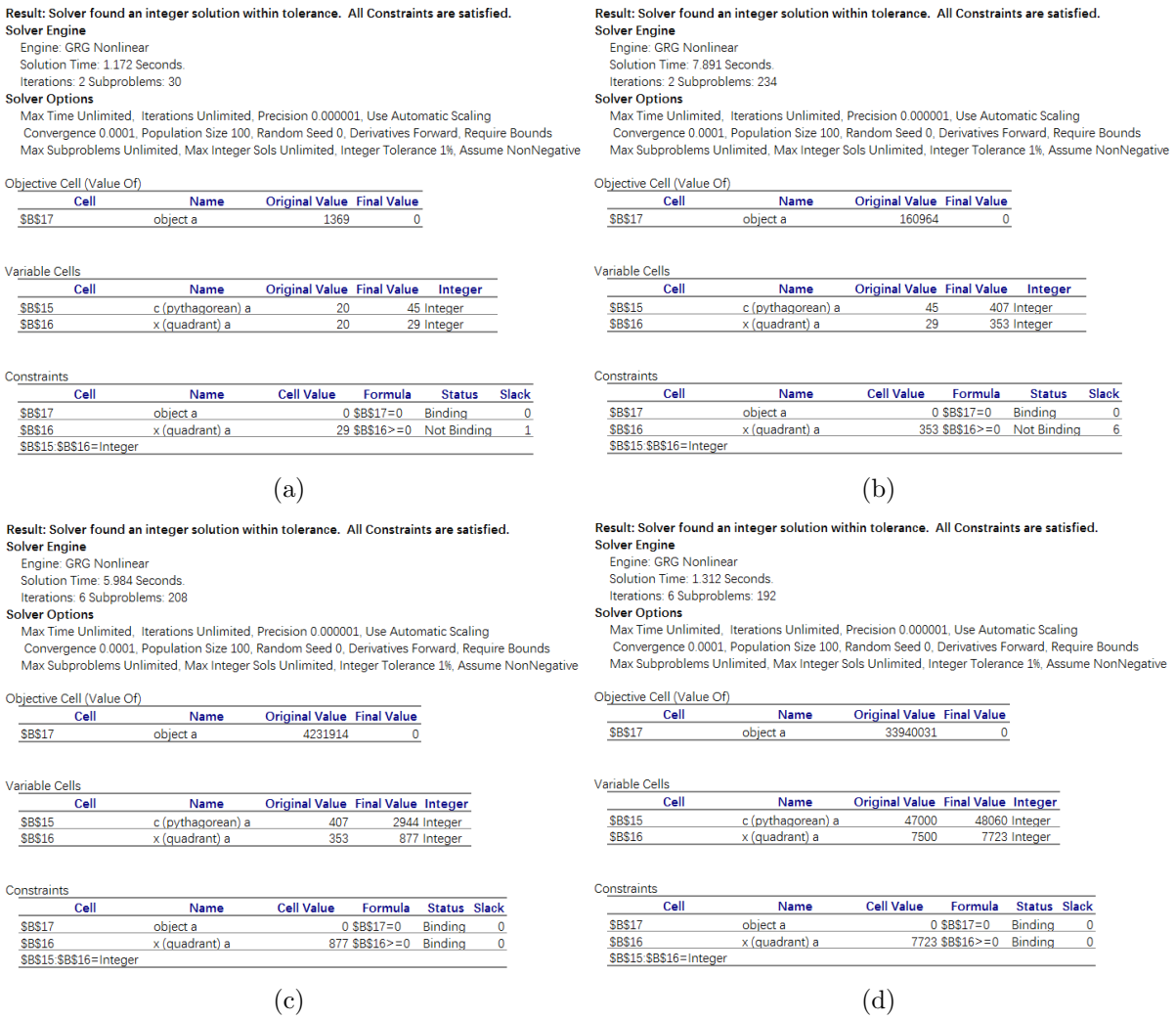


FIGURE 4. Ms Excel Solver configuration settings

4. **Design of Ms Excel Solver.** The explanation of Figure 5 came from Ms Excel Solver while running the solver.

Combination of both equations was achieved through variable c of Pythagorean (Statement (3)) that subsequently defines the value of variable b of quadratic equation (Statement (4)). Initially Ms Excel Solver iterated to find the variable c so that variable b can be defined. Having this b and c, the goal seeking of factoring quadratic equation (Statement (1)) could be performed through statement (5). Those 2 factors, once validated, were



Result: Solver found an integer solution within tolerance. All Constraints are satisfied.
Solver Engine
Engine: GRG Nonlinear
Solution Time: 1.172 Seconds.
Iterations: 2 Subproblems: 30
Solver Options
Max Time Unlimited, Iterations Unlimited, Precision 0.000001, Use Automatic Scaling
Convergence 0.0001, Population Size 100, Random Seed 0, Derivatives Forward, Require Bounds
Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%, Assume NonNegative

Objective Cell (Value Of)

Cell	Name	Original Value	Final Value
$B$17	object a	1369	0

Variable Cells

Cell	Name	Original Value	Final Value	Integer
$B$15	c (pythagorean) a	20	45	Integer
$B$16	x (quadrant) a	20	29	Integer

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$B$17	object a	0	$B$17=0	Binding	0
$B$16	x (quadrant) a	29	$B$16>=0	Not Binding	1
$B$15:$B$16=Integer					

(a)

Result: Solver found an integer solution within tolerance. All Constraints are satisfied.
Solver Engine
Engine: GRG Nonlinear
Solution Time: 7.891 Seconds.
Iterations: 2 Subproblems: 234
Solver Options
Max Time Unlimited, Iterations Unlimited, Precision 0.000001, Use Automatic Scaling
Convergence 0.0001, Population Size 100, Random Seed 0, Derivatives Forward, Require Bounds
Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%, Assume NonNegative

Objective Cell (Value Of)

Cell	Name	Original Value	Final Value
$B$17	object a	160964	0

Variable Cells

Cell	Name	Original Value	Final Value	Integer
$B$15	c (pythagorean) a	45	407	Integer
$B$16	x (quadrant) a	29	353	Integer

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$B$17	object a	0	$B$17=0	Binding	0
$B$16	x (quadrant) a	353	$B$16>=0	Not Binding	6
$B$15:$B$16=Integer					

(b)

Result: Solver found an integer solution within tolerance. All Constraints are satisfied.
Solver Engine
Engine: GRG Nonlinear
Solution Time: 5.984 Seconds.
Iterations: 6 Subproblems: 208
Solver Options
Max Time Unlimited, Iterations Unlimited, Precision 0.000001, Use Automatic Scaling
Convergence 0.0001, Population Size 100, Random Seed 0, Derivatives Forward, Require Bounds
Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%, Assume NonNegative

Objective Cell (Value Of)

Cell	Name	Original Value	Final Value
$B$17	object a	4231914	0

Variable Cells

Cell	Name	Original Value	Final Value	Integer
$B$15	c (pythagorean) a	407	2944	Integer
$B$16	x (quadrant) a	353	877	Integer

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$B$17	object a	0	$B$17=0	Binding	0
$B$16	x (quadrant) a	877	$B$16>=0	Binding	0
$B$15:$B$16=Integer					

(c)

Result: Solver found an integer solution within tolerance. All Constraints are satisfied.
Solver Engine
Engine: GRG Nonlinear
Solution Time: 1.312 Seconds.
Iterations: 6 Subproblems: 192
Solver Options
Max Time Unlimited, Iterations Unlimited, Precision 0.000001, Use Automatic Scaling
Convergence 0.0001, Population Size 100, Random Seed 0, Derivatives Forward, Require Bounds
Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%, Assume NonNegative

Objective Cell (Value Of)

Cell	Name	Original Value	Final Value
$B$17	object a	33940031	0

Variable Cells

Cell	Name	Original Value	Final Value	Integer
$B$15	c (pythagorean) a	47000	48060	Integer
$B$16	x (quadrant) a	7500	7723	Integer

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$B$17	object a	0	$B$17=0	Binding	0
$B$16	x (quadrant) a	7723	$B$16>=0	Binding	0
$B$15:$B$16=Integer					

(d)

FIGURE 5. Ms Excel Solver result: (a) Case#1; (b) Case#2; (c) Case#3; (d) Case#4

found to be the key pair value. Table 4 and Figure 5 provided both simulation inputs and the finding results executed in Ms Excel Solver.

5. **Conclusion.** According to the simulation findings, this solver could factorize, yet with limitation of maximum 15 digits of n. The simulation results were as follows:

1) If $n = 1769$, $p = 29$ and $q = 61$,
2) If $n = 162733$, $p = 353$ and $q = 461$,
3) If $n = 4394647$, $p = 877$ and $q = 5011$, then
4) If $n = 682690031$, $p = 7723$, and $q = 88397$.

The time required for the process was determined by the initial assumption value supplied (c Pythagorean, and p assumptions). The closer it was to the genuine value, the faster it was, and the farther it was from the true value, the slower it was, even when feeding the false value. This insight should be carried when conducting future research that leverages similar method in cracking RSA.

Based on the existing approach of formulation process and simulations, factorization could be done by applying the quadratic equation, namely $a \cdot x^2 - b \cdot x + c = 0$, with quadratic formula $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$. The difficulty in creating an quadratic equation with variable dependence of b might be derived from the statement (4) where variable b of quadratic equation is twice of variable c in Pythagorean as defined in statement (4).

Variable c was determined by doing probability test, $c^2 =\approx \sqrt{(n+i)}$, where $i = i+1$, as presented in statement (3) with the limitation that a and c were integers.

Furthermore, by performing additional proof separately using the Pythagorean in Table 1 and quadratic equations in Table 2, it demonstrated that those two equations were capable of factoring p and q. This proven concept might encourage further research to aim for faster factorization on large RSA numbers by approaching the last method from the two equations.

REFERENCES

[1] A. Ginting, R. R. Isnanto and I. P. Windasari, Implementasi algoritma kriptografi rsa untuk enkripsi dan dekripsi email, *Jurnal Teknologi dan Sistem Komputer*, vol.3, no.2, 2015 (in Indonesian).

[2] A. P. U. Siahaan, Factorization hack of RSA secret numbers, *International Journal of Engineering Trends and Technology (IJETT)*, vol.37, no.1, 2016.

[3] A. Overmars and S. Venkatraman, A fast factorisation of semi-primes using sum of squares, *Math. Comput. Appl.*, vol.24, no.2, 62, https://doi.org/10.3390/mca24020062, 2019.

[4] A. Overmars, L. Ntogramatzidis and S. Venkatraman, A new approach to generate all Pythagorean triples, *AIMS Mathematics*, vol.4, no.2, pp.242-253, 2019.

[5] A. Overmars and S. Venkatraman, New semi-prime factorization and application in large RSA key attacks, *Journal of Cybersecurity and Privacy*, vol.1, no.1, pp.660-674, 2021.

[6] D. Denning, *Cryptography and Data Security*, Addison-Wesley Publishing Company, Inc., 1982.

[7] D. Kurnia, H. Dafitri, Sugianto, M. Mardiana and A. P. U. Siahaan, RSA 32-bit implementation technique, *International Journal of Recent Trends in Engineering & Research (IJRTER)*, DOI: 10.23883/IJRTER.2017.3359.UXAIW, 2017.

[8] E. H. A. Mendrofa, E. Y. Purba and M. Zarlis, Implementasi algoritma RSA dengan kunci EM2B dalam mengenkripsi pesan, *Seminar Nasional Teknologi Informatika*, The Future of Computer Vision, 2017 (in Indonesian).

[9] I. Jahan, M. Asif and L. J. Rozario, Improved RSA cryptosystem based on the study of number theory and public key cryptosystems, *American Journal of Engineering Research*, vol.4, no.1, pp.143-149, 2015.

[10] J.-P. Aumasson, *Serious Cryptography: A Practical Introduction to Modern Encryption*, William Pollock, 2008.

[11] K. H. Rosen, *An Introduction to Cryptography*, 2nd Edition, Discrete Mathematics ITS Applications, Taylor & Francis Group, LLC, 2007.

[12] M. Suhartana, B. Pardamean and B. Soewito, Modeling of risk factors in determining network security level, *International Journal of Security and Its Applications*, vol.8, no.3, pp.193-208, 2014.

[13] M. Mosca and S. R. Verschoor, Factoring semi-primes with (quantum) SAT-solvers, *Sci. Rep.*, vol.12, 7982, DOI: 10.1038/s41598-022-11687-7, 2022.

[14] M. Usman and A. Ahmad, Factoring the RSA key with new factoring algorithm: IMFFV3 in 2015, *International Conference on Engineering & Emerging Technologies (ICEET-2015)*, 2015.

[15] R. B. N. Achmad, Comparison of cryptographic algorithms GOST and RSA in 2019, *IOP Conf. Series: Materials Science and Engineering*, vol.662, no.2, 022086, DOI: 10.1088/1757-899X/662/2/022086, 2019.