Available at https://jtsiskom.undip.ac.id (4 August 2019) Jurnal Teknologi dan Sistem Komputer, 7(3), 2019, 116-120
DOI: 10.14710/jtsiskom.7.3.2019.116-120

SMS Security Improvement using RSA in Complaints Application on Regional Head Election's Fraud

Dwi Yuny Sylfania*), Fransiskus Panca Juniawan, Laurentinus, Harrizki Arie Pradana

Department of Information Technology, STMIK Atma Luhur
Jl. Jend. Sudirman, Selindung Baru, Pangkalpinang, Indonesia 33117

How to Site: D. Y. Sylfania, F. P. Juniawan, L. Laurentinus, and H. A. Pradana, "SMS Security Improvement using RSA in Complaints Application on Regional Head Election's Fraud," *Jurnal Teknologi dan Sistem Komputer*, vol. 7, no. 3, pp. 116-120, 2019. doi: 10.14710/jtsiskom.7.3.2019.116-120, [Online].

Abstract - In the campaign period of regional heads election, fraud can occur, such as money politics, blaming campaign facilities, campaign time violations, and black campaign. This study implemented a secure SMS application for election fraud complaints as a tool for the society to report all forms of election fraud that have occurred to the election supervisory department safely. The RSA algorithm was applied to encrypt the messages for sender privacy protection. The application was able to perform the message randomization function properly with a 10.44% avalanche effect. Brute force attack using a 16-bit key length needs 3.7 milliseconds for each try to find 32.768 possible private keys.

Keywords *– encrypted SMS; public privacy; RSA crypto; election fraud complaint*

I. INTRODUCTION

Regional Heads Election (Pilkada) is the process of recruiting politicians who will become representatives of the people. This process is also a process of people's selecting to elect their regional head, both Governor/Deputy Governor and Regent/Deputy Regent or Mayor/Deputy Mayor [1]. The local election (Pilkada) is not only used as political competition through program ideas, vision, and mission but also as an arena for economic competition between contestants, namely by conducting money politics. Money politics exists because of mutual relations between actors (parties, politicians, or intermediaries) and victims (people) [2].

Nowadays, most people have become smart voters; they took the money but did not vote for the contestants. In this case, complaints or reports from the public are the most important thing to report on all forms of election fraud for the sake of the implementation of the LUBERJURDIL (direct, general, free, Confidential, honest, and fair) election. Therefore, the existence of the short message service (SMS) application can be used to accommodate complaints of all forms of election fraud [3]. With SMS services available on all kinds of mobile phones at all levels in society, they can do fraud reporting.

Many applications applied an encryption algorithm to randomize data so that the contents of the message are kept confidential. Mantoro et al. [4] developed message integrity and secure authentication for smart homes using a smartphone. Juniawan [5] produced an Android-based e-voting BEM security application using RSA. Neyman et al. [6] protected the copyright markers in vector maps using the FFT algorithm. Kridalaksana et al. [7] produced a security SMS using the AES 128-bit algorithm. Other research provides a secured SMS application using RSA [8]-[10]. Sugiyanto and Hapsari [11] applied Vigenere and AES to the SMS messages. Ginting et al. [12] developed an encrypted email using the RSA Algorithm. Atmojo et al. [13] applied RC6 for the secured SMS application. The impact of all secured applications is the increased message security from irresponsible people. A combination of LSB and RSA has been used for hiding messages in digital images resistant to salt and pepper attacks [14]. Another research is a public-key cryptographic implementation with bits ranging from 56 bits to 88 bits with the RSA-CRT algorithm in instant messaging applications [15].

However, [12], [14], [15] are implemented as a desktop application. Mobile applications in [7], [11]-[13] used the symmetric algorithm using the same encryption and decryption key. The key is relatively easy to guess and solve. The key exchange between parties needs a secured connection. In [5]- [9], the user must enter and send the encryption key and decrypt it. The secured SMS application in [10] only able to send a message up to 160 characters.

This study applied an asymmetric algorithm, namely RSA [16], to a mobile Android application to secure SMS messages. This application can send a message with more than 160 characters other than the standard 160 characters one. This application aims to facilitate the public in reporting all forms of election fraud for the sake of the implementation of "LUBERJURDIL" elections while maintaining the confidentiality of messages sent. The implementation of this application can also be used for other public complaints services,

*)Correspondence author (Dwi Yuny Sylfania)
Email: dysylfania@atmaluhur.ac.id

Submitted: 18 January 2019; Revised: 27 July 2019; Accepted: 27 July 2019; Published: 31 July 2019

such as complaints performance services for employees and services for complaints of traffic violations.

II. METHODOLOGY

The stages carried out in this study are algorithm analysis, system design, implementation, and testing.

A. Algorithm Analysis

The initial stage of this study was to analyze the RSA algorithm. RSA is an asymmetric algorithm, which has two keys, namely public key and private key. Public keys are used in the encryption process, and private keys are used in the decryption process [16]. RSA password security lies in the difficulty of factoring large numbers into prime factors to obtain the private key. The RSA algorithm scheme consists of three processes, namely, keys generation process, encryption, and decryption.

The keys generation process performed to obtain encryption and decryption keys. The results of keys generation are public keys e and n, and the private key d and n. Algorithm 1 describes the keys generation steps.

Algorithm 1. Key generation process

1: Select two prime numbers arbitrarily (*p* and *q*), where $p \neq q$.
2: Calculate the value of *n*, where $n = pq$
3: Calculate $\phi n = (p - 1)(q - 1)$
4: Determine public key, *e*, which is random, where $1 < e < \phi n$, $FPB(\phi n, e) = 1$.
5: Calculate private key, *d*, using the formula $ed = 1 \bmod \phi n$, so that, *d*, can be calculated by following Eq. 1.

$$d = \frac{1 + k\phi(n)}{e} \quad (1)$$

Encryption performed by using a public key *(e, n)* to change the plaintext to the ciphertext that will be sent to the recipient so that the message cannot be read. Thus, only people who have a private key can decrypt the ciphertext. Algorithm 2 describes the encryption process.

Algorithm 1. Encryption process

1: Convert each character to decimal number 65 – 90, where *A* = 65, *B* = 66, …, *Z* = 90, and 97 - 122, where *a* = 97, *b* = 98, …, *z* = 122.
2: Calculate C, where, $C = M^e \bmod n$ *and* $0 \leqslant M < n$ *where C is ciphertext.*

Decryption performed by using the private key *(d, n)* to change the ciphertext to plaintext so that the message can be known. The steps are each ciphertext block decrypted with formula $M = C^d \bmod n$, where *M* is the plaintext.

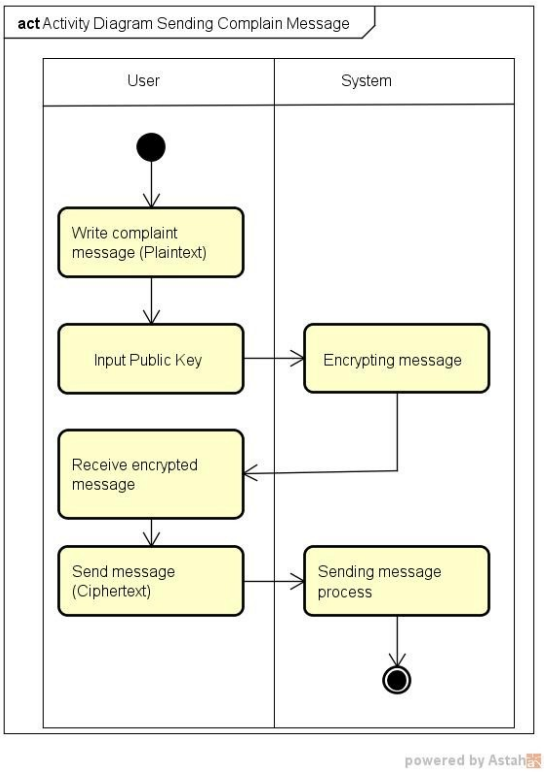

Figure 1. Sending message process

B. System Design

The purpose of this stage is to describe the system process, where the system can send encrypted messages and receive messages as the plaintext ones. Figure 1 explains the complaint message sending process, involving the user (society) in the form of plaintext. The user enters the public key for the encryption process and sends the message in the form of a ciphertext that has been encrypted by the system. Figure 2 describes the process of receiving the complaint message, involving the admin as the party receiving the ciphertext message, then entering the private key for the decryption process, and the system will send the ciphertext to the plaintext for the admin.

C. Implementation

From the results of the system design, the RSA algorithm will be discussed from the keys generation stage, the encryption process, and the message decryption process. In addition, the system will be applied in some cases.

D. Testing

Some testings performed to measure the performance of the system. The tests carried out are functionality testing using Blackbox, testing avalanche effect to test the results of encryption with the use of words that have the same middle letter, and brute force Attack testing to test private key piracy by irresponsible people.

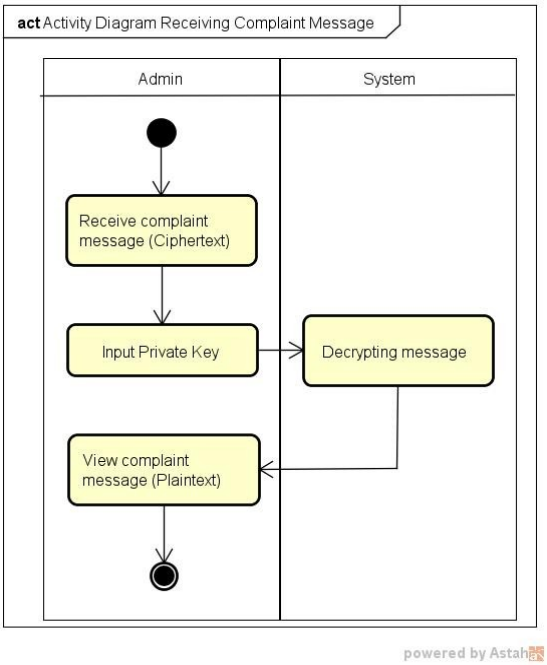

Figure 2. Receiving message process

III. Results and Discussion

This chapter explains one case usage of the proposed system in keys generation, message encryption, and ciphertext decryption.

A. Keys Generation

The keys generation process is the first step that must be done before carrying out the encryption and decryption process using Algorithm 1. The generated keys are the public key for encryption and the private key for decryption. For a client, the value of primes *p* is 17, and *q* is 11, then giving $n = 187$, $\phi n = 160$, and GCD calculation in Table 1. The value of *e* may not be a factorial value of ϕn, which is a prime number (2 or 5) so that the value of *e* is determined, namely 7.

Table 2 shows the value calculation of *d* using Eq. 1. The value of *d* is an integer, where *k* is a trial value (in the form of integers) = 1, 2, 3, ..., so that the value of *d* is 23, with $k = 1$. So, the public key is obtained (7, 187), and the private key is (23, 187).

The mechanism for public key exchange between users and system admins is not disseminated but directly implemented into the application. In this application, the public key has been implemented directly on the application, so that users only type a complaint message then do the encryption process by pressing the "encrypt button" and sending a ciphertext to the admin. As with the decryption process, the admin no longer needs to enter the private key because it has been implemented directly into the application.

B. Message Encryption

Each character of a message is converted into decimal value and encrypted by using the public key *(e, 7n)* that has been obtained from the previous keys generation process. The encryption process of the message will produce a ciphertext that will be sent to the recipient.

Table 1. GCD calculation

No.	e	GCD ($\phi n = 160$,e)
1	e = 2	GCD (160,2)
2	e = 3	GCD (160,3)
3	e = 4	GCD (160,4)
4	e = 5	GCD (160,5)
5	e = 6	GCD (160,6)
6	e = 7	GCD (160,7)
7	e = 8	GCD (160,8)
8	e = 9	GCD (160,9)

Table 2. Calculation value of *d*

k	d
k=1	23
k=2	45.86
k=3	68.71

Table 3. Message encryption calculation

P	Decimal Value	$C = M^e \bmod n$
s	115	$C = 115^7 \bmod 187 = 157$
u	117	$C = 117^7 \bmod 187 = 127$
a	97	$C = 97^7 \bmod 187 = 92$
p	112	$C = 112^7 \bmod 187 = 73$
Space	32	$C = 32^7 \bmod 187 = 76$
u	117	$C = 117^7 \bmod 187 = 127$
a	97	$C = 97^7 \bmod 187 = 92$
n	110	$C = 110^7 \bmod 187 = 66$
g	103	$C = 103^7 \bmod 187 = 137$

Based on Table 3, Column P contains 'suap uang', a message (plaintext) that will send. The decimal value column is the result of converting each word to a decimal number. Furthermore, the encryption process is carried out using public keys (7, 187) and uses the mathematical equation $C = M^e \bmod n$, which produces ciphertext 157, 127, 92, 73, 76, 127, 92, 66, 137.

C. Ciphertext Decryption

The message received is in the form of ciphertext. The decryption process uses the private key *(d, n)* to find out the contents of the message from the ciphertext. Based on Table 4, column C contains the following ciphertext 157 127 92 73 76 127 92 66 137, which will be decrypted using the private key (23, 187) and using the mathematical equation $P = C^d \bmod n$. From this equation, a decryption result is a decimal number that needs to be converted first to find out the contents of the message, resulting in plaintext of 'suap uang'.

D. Implementation

This section displays the RSA implementation in the application that has been built. Figure 3 shows the77

Table 4. Message decryption calculation

C	$P = C^d \bmod n$	P
157	$P = 157^{23} \bmod 187 = 115$	s
127	$P = 127^{23} \bmod 187 = 117$	u
92	$P = 92^{23} \bmod 187 = 97$	a
73	$P = 73^{23} \bmod 187 = 112$	p
76	$P = 76^{23} \bmod 187 = 32$	Space
127	$P = 127^{23} \bmod 187 = 117$	u
92	$P = 92^{23} \bmod 187 = 97$	a
66	$P = 66^{23} \bmod 187 = 110$	n
137	$P = 137^{23} \bmod 187 = 103$	g

encryption process for the plaintext in Bahasa which states '**Pada desa mekar sari caleg DPRD Provinsi dari Partai Harum yang bernama Bunga melakukan suap berupa uang tunai dan sembako agar memilih caleg tersebut.**' The ciphertext for the message is '75; 92; 144; 92; 76; 144; 84; 157; 92; 76; 131; 84; 112; 92; 126; 76; 157; 92; 126; 96; 76; 176; 92; 48; 84; 137; 76; 51; 75; 91; 51; 76; 75; 126; 155; 101; 96; 66; 157; 96; 76; 144; 92; 126; 96; 76; 75; 92; 126; 74; 92; 96; 76; 30; 92; 126; 127; 13; 76; 77; 92; 66; 137; 76; 21; 84; 126; 66; 92; 131; 92; 76; 110; 127; 66; 137; 92; 76; 131; 84; 48; 92; 112; 127; 112; 92; 66; 76; 157; 127; 92; 73; 76; 21; 84; 126; 127; 73; 92; 76; 127; 92; 66; 137; 76; 74; 127; 66; 92; 96; 76; 144; 92; 66; 76; 157; 84; 131; 21; 92; 112; 155; 76; 92; 137; 92; 126; 76; 131; 84; 131; 96; 48; 96; 179; 76; 176; 92; 48; 84; 137; 76; 74; 84; 126; 157; 84; 21; 127; 74'.

E. Testing

The functionality test was to prove that the system built can work well and according to user needs. The Avalanche effect test was conducted, which aims to prove that with a plaintext input that is 1 character different in each experiment using the same key will produce a different ciphertext. The testing attack used a brute force attack technique to measure system resistance to attacks.

Based on Table 5, all functionalities of the system are running well. The message generation, encryption, sending process, inbox, and message decryption processes are running well. The SMS encryption and decryption application implemented in Android phone can perform well as the system in [7]-[11], [13]. Encrypting and decrypting messages using RSA can work as well as in [8]-[10]. This study has two advantages that there is no limit for the message length. Another advantage is the sender and receiver do not need to enter the key, in order to do the encryption and decryption process because it is embedded in the application. It is dedicated to facilitating the public in reporting all forms of election fraud while maintaining the confidentiality of messages sent.

The result of the Avalanche effect testing shown in Table 6, distinguishes 1 character from each word. After the ciphertext obtained, then It appears that only those character bits are different. The average percentage of the avalanche effect is 10.44%.

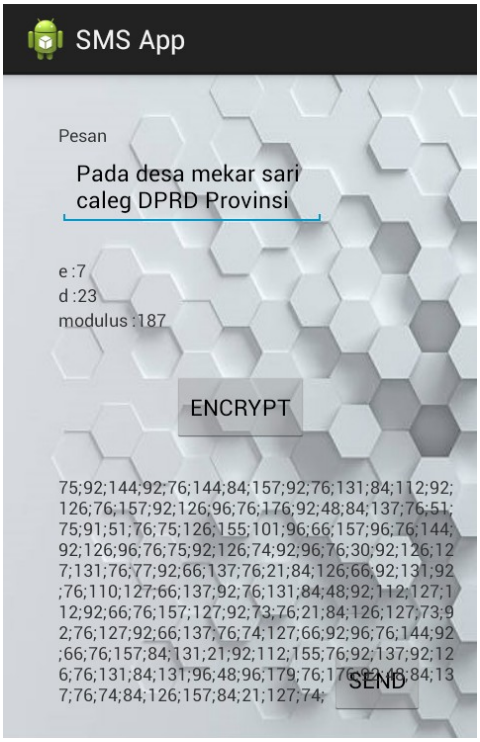

Figure 3. Encryption message in application

Table 5. Blackbox testing

Testing	Expected Results	Results
Fill the complaint message	Display message form such as destination number and message	Appropriate
Encrypt messages	The message successfully encrypted	Appropriate
Send the messages	The message (ciphertext) sent.	Appropriate
View message inbox	Display all incoming messaging (ordinary messages and encrypted messages)	Appropriate
Decrypt message	The message successfully decrypted after the recipient entered the private key	Appropriate

Table 7 shows that the private key consists of 2 digits namely 2 and 3, every 1 byte (8 bits). The bit size is obtaining from the number of digits multiplied by 8, which is 16 bits. The number of possible private keys is 65.536 keys. The experiments conducted to find the possible private key are 32.768 experiments. The private key search time is 3.7 milliseconds per test.

IV. Conclusion

The Android application can accommodate complaints using secured SMS in all forms of regional election fraud. The implementation of the RSA algorithm can keep the confidentiality and accuracy of

the contents of the message. The message encryption and decryption process produce the appropriate ciphertext and plaintext.

Table 6. Avalanche effect testing

No	Plain text	Public Key	Ciphertext	Avalanche Effect (%)
1	Beras	7, 187	110; 84; 126; 92; 157	15
	Berat		110; 84; 126; 92; 74	
2	Benar	7, 187	110; 84; 66; 92; 126	17.5
	Besar		110; 84; 157; 92; 126	
3	Muka	7, 187	121; 127; 112; 92	12.5
	Suka		8; 127; 112; 92	
4	Duka	7, 187	51; 127; 112; 92	9.4
	Duku		51; 127; 112; 127	
5	Denda	7, 187	51; 84; 66; 144; 92	7.5
	Depda		51; 84; 73; 144; 92	
6	Makan	7, 187	121; 92; 112; 92; 66	7.5
	Pakan		75; 92; 112; 92; 66	
7	Jera	7, 187	167; 84; 126; 92	12.5
	Jeri		167; 84; 126; 96	
8	Jalan	7, 187	167; 92; 48; 92; 66	10
	Jaran		167; 92; 126; 92; 66	
9	Makan	7, 187	121; 92; 112; 92; 66	10
	Makar		121; 92; 112; 92; 126	
10	Merah	7, 187	121; 84; 126; 92; 179	2.5
	Marah		121; 92; 126; 92; 179	

Table 7. Brute force attack testing

No	Parameter	Value
1.	Private key	23 (has 2 digit)
2.	Bit size (bit)	16
3.	Number of possible keys	65,536
4.	Total experiments	32,768
5.	Key search time (ms)	3.7

REFERENCES

[1] J. J. Prihatmoko, *Mendemokratiskan Pemilu Dari Sistem Sampai Elemen Teknis.* Semarang: Pustaka Pelajar, 2007.

[2] H. Hamid and E. Suandi, *Memperkokoh Otonomi Daerah: Kebijakan, Evaluasi dan Saran.* Yogyakarta: UII Press, 2004.

[3] S. Maulana, *5 Proyek Populer SMS Gateway.* Jakarta: PT Elex Media Komputindo, 2015.

[4] T. Mantoro, A. A. Media, and S. Munawwarah, "Securing the Authentication and Message Integrity for Smart Home Using Smartphone," in *2014 International Conference on Multimedia Computing and System*, Marrakech, Morocco, Apr. 2014, pp. 985-989.

[5] F. P. Juniawan, "RSA Implementation for Data Transmission Security in BEM Chairman E-voting Android based Application," in *1st International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Yogyakarta, Indonesia, Aug. 2016, pp. 93-98.

[6] S. N. Neyman, I. N. P. Pradnyana, and B. Sitohang, "A New Copyright Protection for Vector Map Using FFT-based Watermarking," *Telkomnika*, vol. 12, no. 2, pp. 367-378, 2014.

[7] A. H. Kridalaksana, E. Arriyanti, and W. Widodo, "Aplikasi Pengaman SMS dengan Metode Kriptografi Advanced Encryption Standard (AES) 128 Berbasis Android," *Sebatik*, vol. 10, no. 1, pp. 8-14, 2013.

[8] H. Bodur and R. Kara, "Secure SMS Encryption Using RSA Encryption Algorithm on Android Message Application," in *3rd International Symposium On Innovative Technologies In Engineering And Science*, Valencia, Spain, Jun. 2015.

[9] A. R. Alvianto and D. Darmaji, "Pengaman Pengiriman Pesan Via SMS dengan Algoritma RSA Berbasis Android," *Jurnal Sains dan Seni ITS*, vol. 4, no. 1, pp. A1-A6, 2015.

[10] E. R. Sardju, R. Magdalena, and R. D. Atmaja, "Implementasi Algoritma RSA untuk Enkripsi dan Dekripsi SMS (Short Message Service) Pada Ponsel Berbasis Android," *e-Proceeding of Engineering*, vol.2, no.2. pp. 2435-2442, 2015.

[11] S. Sugiyanto and R. K. Hapsari, "Pengembangan Algoritma Advanced Encryption Standard Pada Sistem Keamanan SMS Berbasis Android Menggunakan Algoritma Vigenere," *Jurnal Ultimatics*, vol. 8, no. 2, pp. 131-138, 2016.

[12] 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, pp. 253-258, 2015.

[13] W. P. Atmojo, R. R. Isnanto, and R. Kridalukmana, "Implementasi Aplikasi Kriptografi Pada Layanan Pesan Singkat (SMS) Menggunakan Algoritma RC6 Berbasis Android," *Jurnal Teknologi dan Sistem Komputer*, vol. 4, no. 3, pp. 450-453, 2016.

[14] A. E. Handoyo, D. R. I. M. Setiadi, E. H. Rachmawanto, C. A. Sari, and A. Susanto,The message encryption and decryption process produce the appropriate ciphertext and plaintext. "Teknik Penyembunyian dan Enkripsi Pesan pada Citra Digital dengan Kombinasi Metode LSB dan RSA," *Jurnal Teknologi dan Sistem Komputer*, vol. 6, no. 1, pp. 37-43, 2018.

[15] A. Arief and R. Saputra, "Implementasi Kriptografi Kunci Publik dengan Algoritma RSA-CRT pada Aplikasi Instant Messaging," *Scientific Journal of Informatics*, vol. 3, no. 1, pp. 46-54, 2016.

[16] R. L. Rivest, A. Shamir, and L. Adleman, *A Method for Obtaining Digital Signature and Public Key Cryptosystems.* Massachusets: MIT, 1977.