ISSN No. 0976-5697

Volume 8, No. 5, May-June 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Implementation & Security Analysis for RSA based Algorithms on Variable Key Length

Anas Irfan
Department of Computer Science & Engineering
Jamia Hamdard
New Delhi, India

Aqeel khalique
Department of Computer Science & Engineering
Jamia Hamdard
New Delhi, India

Mohd. Abdul Ahad
Department of Computer Science & Engineering
Jamia Hamdard
New Delhi, India

Abstract: Cryptography provides confidentiality, integrity and availability. It is used to provide security and to communicate securely while the data is in transmission or at rest. Public key cryptography is used to implement these security mechanisms using secret keys for encryption and decryption. One of the most widely used public key cryptosystem is RSA. In this paper, implementation of the variants of RSA algorithm are performed and comparative analysis of these variants based on the time consumption for key generation, encryption and decryption process for different variants have been done. Result has been obtained and analysis is performed showing that security increases with increase in the key length. But it comes with a trade-off that it is more time consuming because of computational complexity of these algorithms. We implemented the variants of RSA algorithms namely ESRKGS and GRSA-AA algorithm. The comparative result is presented based on different parameters after which analysis is performed.

Keywords: Cryptography; RSA algorithm; GRSA-AA algorithm; public key; private key; key generation; encryption; decryption.

I. INTRODUCTION

Information Security basic concept is to secure the data and information against security threats, for securing information from unauthorized access and to prevent unauthorized modification while in transmission or at rest. It includes maintaining the confidentiality to ensure that the data is kept secret among legitimate user, maintaining the integrity of the data to ensure the completeness and correctness of data such that no changes is been made by any illegitimate user and maintaining the availability of the data whenever it is required by a legitimate user [1].

There are several ways to implement necessary security mechanisms; one of them is cryptosystem or encipherment which is used to implement confidentiality, integrity & availability. Cryptographic system mainly relies on the following three dimensions [1]:

- Methods involved in the processing of the plain text.
- Key length and the number of keys used in the process.
- Algorithms involved in the process of encryption and decryption i.e. from plain text to cipher text and vice versa.

There are two types of cryptographic systems used widely, based on the number of keys used by them for encryption and decryption of data namely symmetric key cryptosystem & public key cryptosystem (asymmetric key cryptosystem).

Symmetric key cryptosystem uses the same secret key for both encryption-decryption and secret key is shared between the communicating parties. As the communicating parties will be using and accessing the same shared secret key, it became one main drawback of symmetric key cryptosystem in comparison with asymmetric key cryptosystem. In asymmetric key cryptosystem, different set of keys has been used for both encryption and decryption known as the public key and the private key. Both these keys are large numbers, mathematically linked together but they are different (asymmetric). Out of these, public key is shared with everyone and the other key is kept private for the user and the message can be encrypted using either of the two keys and the other key will be used to decrypt the message. Strength of the algorithm or method used for encryption and decryption is directly dependent on the key length used and increasing key length will deliver an exponential increase in security strength. Whitfield Diffie, Martin Hellman and Ralph Merkle give the concept of asymmetric key cryptosystem [2].

In asymmetric key cryptosystem, we use public key to encrypt plain text or to verify a digital signature created by the private key, and the other key i.e. private key is used to decrypt the cipher text encrypted earlier by other key from its pair i.e. (public key) or to create the digital signature to ensure the integrity and the confidentiality. The basic requirements for asymmetric key cryptosystem that must satisfy the public key cryptographic schemes or the algorithm are listed below [1]:

- It is easy for the communicating parties (users) to generate the pair of keys which is being used in communication i.e. (public key and private key).
- It is easy for the user (sender) to generate the cipher text (encrypt) by using the public key along with an encryption algorithm on the message (plain text) and

send the resulting cipher text.

- It is easy for the user (receiver) to retrieve the original message (plain text) by using the private key from the same pair of keys along with the decryption algorithm on the received cipher text from the user (sender).
- It is computationally impossible for an opponent (attacker) to determine or guess the private key from the shared public key.
- It is computationally impossible for an opponent (attacker) to retrieve or guess the original message (plain text) from the public key and the cipher text (encrypted message).

Asymmetric key cryptosystem has basically the following components - a) Users (Communicating parties or sender & receiver), b) Plain text (Message to transfer), c) Encryption Decryption Algorithm, d) Secret keys (Public & Private keys) and e) Cipher Text (encrypted message or the secret message) [1]. One of the most popular Asymmetric cryptosystems is the RSA public key cryptosystem [3], named after their inventors R. Rivest, A. Shamir and L. Adleman. RSA algorithm was developed in 1977. RSA algorithm uses two large prime numbers to mathematically compute and generate a pair of keys (public & private keys). One of the key (generally public key) is used to encrypt plain text (message) to generate the cipher text (encrypted message) and other key (private key) is used to decrypt the cipher text (encrypted message) to obtain the original plain text (message) [3] using the expression.

$C = M^e \bmod n$ --- (Eq. 1)

$M = C^d \bmod n$ --- (Eq. 2)

Where (e, n) is public key, (d, n) is private key, (C) is the cipher text and (M) is the plain text.

In this paper, we implement & analyze the variants of RSA algorithm based on variable Key Length and different number of large prime numbers which is used to generate keys for encryption and decryption.

This paper is organized in sections and subsections. In Section 2, we present the algorithms in subsections for RSA algorithm with 2 large prime numbers[3], ESRKGS algorithm which uses 4 large prime numbers[4], & GRSA-AA algorithm which uses 2k large prime numbers (where k>=2)[6]. Section 3, we present the implementation of all the algorithms. Section 4, we present result & analysis for algorithms RSA, ESRKGS & GRSA-AA with 2k prime numbers with different key length of size 128, 256, 512, 1024, 2048 & 4096 bits for time consumption in milliseconds (ms), for the different phases in algorithm such as key generation, encryption & decryption. Result is obtained by analyzing key generation, time consumed in encryption & decryption for studied algorithms using variable key length. In Section 5, we present the future scope. In Section 6, we present the conclusion for the research paper.

II. PUBLIC KEY CRYPTOSYSTEM

Public key cryptosystem uses asymmetric pair of keys (public & private keys) for encryption and decryption the message, these keys mathematically connected to one another one is used for encryption then another key from the pair is used for decryption of the encrypted message and to obtain the secret message.

A. RSA Algorithm

Public key cryptosystem RSA is one of the first and widely used cryptosystem, for secure data transmission. RSA is made of the initial letters of the surnames of its inventors - Ron Rivest, Adi Shamir and Leonard Adleman. They proposed the RSA public key cryptosystem algorithm in 1977 [3]. Strength of RSA depends on its key length, more the key length more is the strength and it is more difficult to perform cryptanalysis on RSA. Observed difficulty with the RSA cryptosystem is both public and private keys are generated using the mathematical function on ‘n’ which is the product of 2 very large prime numbers, this ‘n’ number can be factorize using integer factorization [5] and various other cryptanalysis on RSA like Low Private or Public Exponent, Common Modulus, Blinding [7][8] to find out the keys thus breaking the security of RSA. RSA cryptosystem algorithm has the following four steps: key generation, key distribution, encryption and decryption [1]. Algorithm for RSA cryptosystem uses two large prime numbers [3]:

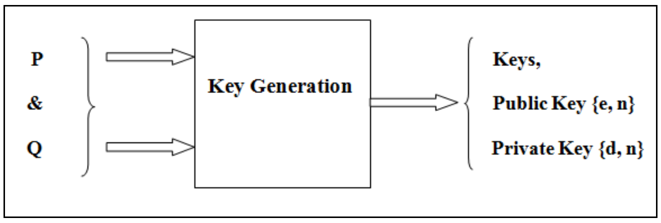

Figure 1: a) RSA cryptosystem Key Generation

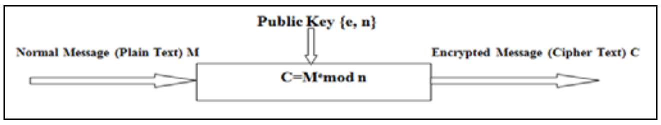
Public Key {e, n}
Normal Message (Plain Text) M
Encrypted Message (Cipher Text) C
C=M^e mod n

Figure 1: b) RSA cryptosystem Encryption

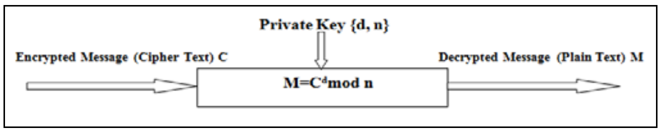
Private Key {d, n}
Encrypted Message (Cipher Text) C
Decrypted Message (Plain Text) M
M=C^d mod n

Figure 1: c) RSA cryptosystem Decryption

Figure 1: a) shows block diagram for the RSA algorithm for key generation process and uses 2 large prime numbers to generate public and private keys which is further used for encryption process. Figure 1: b) takes plain text M as input and gives out cipher text C as output, & for decryption process as in figure 1: c), which takes cipher text C as input to the decryption function and yields plain text message M as output.

In Key distribution for RSA algorithm [3]; consider that user A wants to send a secret message to user B. If they decide to use RSA, user A must know user B’s public key to encrypt the message and, user B must use private key to decrypt that message. To enable user A to send encrypted messages, user B transmits only public key (e, n) to user A through a reliable, but not necessarily secret route. User B's private key (d, n), is never distributed.

Encryption algorithm for RSA public key cryptosystem [3], the sender (user A) will encrypt the message M (plain text) using the RSA encryption algorithm and user B's public key (e, n) and send the encrypted message (cipher text) C to the user B; using the expression for encryption.

Decryption algorithm for RSA public key cryptosystem [3], the sender (user B) will decrypt the received encrypted message (cipher text) C using the decryption algorithm and own private key (d, n) to obtain the original message M (plain text); using the expression for decryption.

B. ESRKGS Algorithm

An Enhanced and Secured RSA Key Scheme (ESRKGS) proposed by M. Thangavel in 2014 [4], an extended version of RSA algorithm which uses 4 large prime numbers to mathematically compute and generate the set of keys (public and private keys) rest of the algorithm functionality stays the same as RSA algorithm. The main focus of ESKRGS is to make it computationally difficult for the attacker to factorize 'n' by integer factorization [5]. This 'n' which is the product of 2 large prime numbers can be factorized. ESKRGS uses 4 large prime numbers which is computationally impossible to use integer factorization to determine the prime number and to find the keys.

ESRKGS algorithm has four steps: key generation, key distribution, encryption and decryption. Algorithm for ESRKGS uses 4 large prime numbers [4]:

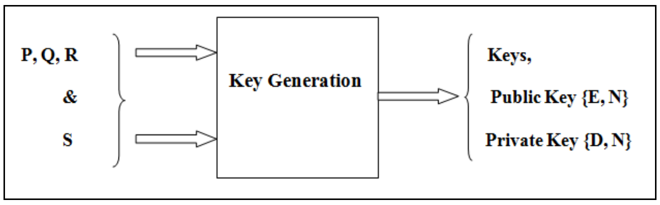

Figure 2: a) ESRKGS cryptosystem Key Generation

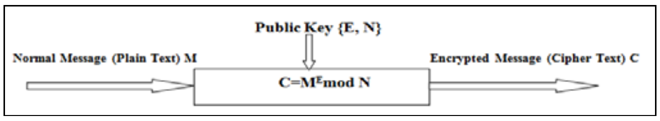
Public Key {E, N}
Normal Message (Plain Text) M
Encrypted Message (Cipher Text) C
C=M^E mod N

Figure 2: b) ESRKGS cryptosystem Encryption

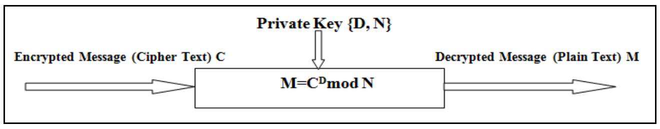
Private Key {D, N}
Encrypted Message (Cipher Text) C
Decrypted Message (Plain Text) M
M=C^D mod N

Figure 2: c) ESRKGS cryptosystem Decryption

Figure 2: a) represents block diagram for the ESRKGS algorithm for key generation process and uses 4 large prime numbers to generate public and private keys which is further used for encryption process. Figure 2: b) takes plain text M as input and gives out cipher text C as output, & for decryption process as in figure 2 c), which takes cipher text C as input to the decryption function and yields plain text message M as output.

In key distribution for ESRKGS algorithm, consider that user A wants to send a secret message to user B. If they decide to use ESRKGS, user A must know user B's public key to encrypt the message and, user B must use its own private key to decrypt that message. To enable user A to send encrypted messages, user B transmits only public key (E, N) to user A through a reliable, but not necessarily secret route. User B's private key (D, N), is never distributed.

In encryption algorithm for ESRKGS [4]; user A will encrypt the message M using the encryption algorithm along with user B's public key (E, N) and send the encrypted cipher text C to user B; using the expression for encryption.

In decryption algorithm for ESRKGS [4]; user B will decrypt the received encrypted message (cipher text) C using the decryption algorithm with own private key (D, N) to decrypt and obtain the original message M; using the expression for decryption.

With Crypt analyzing ESRKGS, it was claimed that it is highly secure and not easily breakable in comparison to the RSA algorithm because it uses 4 large prime numbers instead of 2 large prime numbers as in RSA algorithm. It was also claimed that even if someone factorizes n the product of prime numbers, in order to obtain the private key D, brute force must be used to obtain the other two primes. To cryptanalysis it uses an alternative private key D1 to break the system. It does not need to employ brute force to break the system when n is factored. Hence, ESRKGS has the same security level as the RSA algorithm [9].

C. GRSA-AA Algorithm

Generalized RSA using 2^k prime numbers with secure key generation proposed by AH. Lone & A. Khalique in 2016 [6], proposed a generalized scheme of RSA algorithm which uses 2^k large prime numbers, named it as GRSA-AA (Generalized RSA-Advance & Adaptable). It uses 2^k large prime numbers (where k >=2) with secured key generation mechanism which uses 4, 8, 16, 32... Prime numbers providing defense against the direct attacks like Integer factorization. GRSA-AA include secure key generation algorithm making it more secure thanl RSA algorithm, making it computationally difficult for the attacker to factorize 'n' by integer factorization [5] and various other cryptanalysis on RSA like Low Private or Public Exponent, Common Modulus, Blinding [7][8].. It uses 4 or more large prime numbers (in order 2^k, where k>=2) which makes it computationally impossible to use integer factorization to determine the keys used.

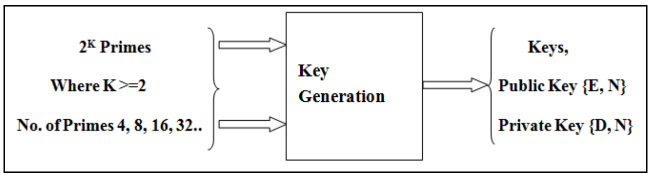

Figure 3 a) GRSA-AA Cryptosystem Key Generation

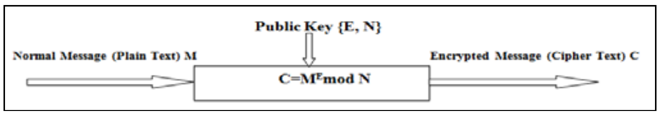
Public Key {E, N}
Normal Message (Plain Text) M
Encrypted Message (Cipher Text) C
C=M^E mod N

Figure 3 b) GRSA-AA Cryptosystem Encryption

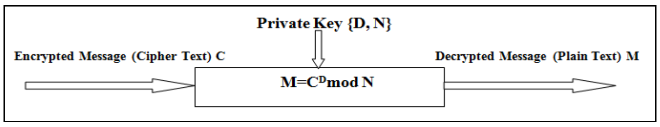
Private Key {D, N}
Encrypted Message (Cipher Text) C
Decrypted Message (Plain Text) M
M=C^D mod N

Figure 3 b) GRSA-AA Cryptosystem Decryption

Figure 3: a) shows block diagram for the GRSA-AA algorithm for key generation process and uses 2k large prime numbers to generate public and private keys which is further used for encryption process. Figure 3: b) takes plain text M as input and gives out cipher text C as output, & for decryption process as in figure 3 c), which takes cipher text C as input to the decryption function and yields plain text message M as output.

GRSA-AA algorithm basically has following steps: key generation, key distribution, encryption and decryption.

Key distribution for GRSA-AA, consider that user A wants to send a secret message to user B. If they decide to use GRSA-AA, user A must know user B's public key to encrypt the message and, user B must use private key to decrypt that message. To enable user A to send encrypted messages, user B transmits only public key (E, N) to user A through a reliable, but not necessarily secret route. User B's private key (D, N), is never shared.

Encryption algorithm for GRSA-AA, User A will encrypt the message M using the encryption algorithm along with user B's public key (E, N) and send the encrypted cipher text C to the user B; using the expression for encryption.

Decryption algorithm for GRSA-AA, User B will decrypt the received encrypted message (cipher text) C using the decryption algorithm along with own private key (D, N) to obtain the decrypted original message M; using the expression for decryption.

Similarly, GRSA-AA algorithm operates with 2^4, 2^5 & higher powers of 2 large prime numbers, for larger values for key length in bits (4096bits) ,will increase the computational complexity exponentially, as the result time consumed will rise exponentially for the key generation, encryption and decryption process.

III. IMPLEMENTATION

In this paper, variants of RSA is implemented which uses variable key length to study and analyze its security strength with different key size. Variants of RSA use different number of primes for providing more security and to make it computationally infeasible for attacker to perform integer factorization to find the primes used for the generation of keys.

For simulation purpose, RSA algorithm using two prime numbers, ESRKGS using four prime numbers, GRSA-AA using 2^3, 2^5 & with higher powers number of primes all are implemented using JAVA Big Integer library functions. Big Integer provide with numerous operations like predefined modular methods, GCD methods, random number generation and many other methods, First user is asked to specify the key length in bits for randomly generation of prime number and thus generating public & private keys along with encryption and decryption operations performed and time is calculated for each of these process which is used for performing analysis and generating results. All these algorithms are implemented using JAVA JDK through Net-beans IDE running on i5 Intel Core Processor 2330M CPU @ 2.45 GHz and 8GB of RAM.

IV. RESULT AND ANALYSIS

In RSA security strength is based on the key length used for encryption and decryption, where key length is the size of public and private keys (in bits) used in the algorithm. In this paper variable key length is used with all the variants for RSA algorithm. Increasing key size gives more strength to the cryptosystem with a tradeoff computational complexity resulting in more time consumption for the process of key generation.

A. RSA Algorithm using 2 Primes

Table 1 shows the time (ms) taken by RSA algorithm for different phases like key generation, encryption and decryption using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) with 2 prime numbers. With increasing key size, time consumed also increases with higher computational complexity. Key generation time increases gradually with increase in key length where as in encryption it consumes less time and comparatively it consumes more time for decryption operation to obtain the original message from the encrypted message received.

Figure 4 shows the graphical representation for the above data from table 1 which shows time consumed for encryption, decryption and key generation for different key lengths for RSA algorithm which uses 2 prime numbers. We can clearly see that decryption consumes more time than encryption and key generation.

Table I. **Result for RSA algorithm using 2 primes with different key length**

Key Length (bits)	Time (ms)		
	Key Generation	Encryption	Decryption
128	10	1	0.2
256	23	1	10
512	34	2	6
1024	79	11	45
2048	297	76	328
4096	577	634	2565

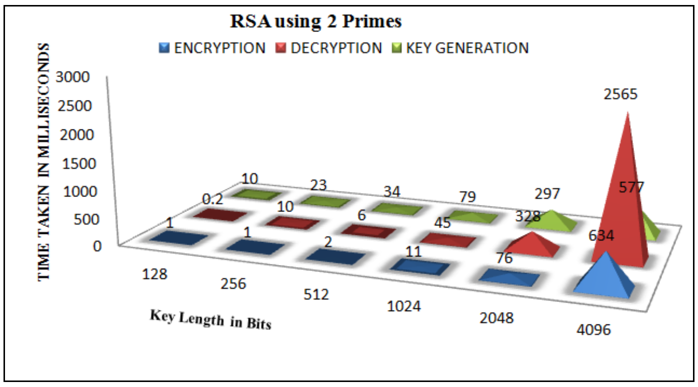

Figure 4: Graphical representation of the result for RSA algorithm using 2 primes with variable key length

B. ESRKGS using 4 Prime Numbers

Table 2 shows the time (ms) taken by ESRKGS (Enhance & Secure RSA Key Generation Scheme) algorithm for different phases like key generation, encryption and decryption using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) with 4 prime numbers, with increasing key size, time consumed also increases with higher computational complexity. Key generation time increases gradually with increase in key length where as in encryption it consumes less time and comparatively it consumes more time for decryption operation to obtain the original message from the encrypted message received.

Figure 5 shows the graphical representation for the above data from table 2, which shows time consumed for encryption, decryption and key generation for different key lengths by ESRKGS algorithm which uses 4 prime numbers. We can clearly see that key generation here consumes more time than encryption and decryption, where as decryption consumes more time than encryption operation.

Table II. **Result for ESRKGS algorithm using 4 primes with different key length**

Key Length (bits)	Time (ms)		
	Key Generation	Encryption	Decryption
128	47	1	4
256	45	6	30
512	82	17	92
1024	159	47	342
2048	1935	254	3201
4096	17745	2242	9357

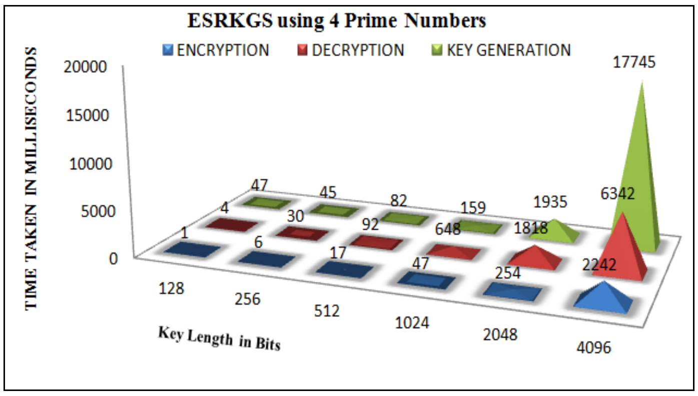

Figure 5: Graphical representation of the result for ESRKGS algorithm using 4 primes with variable key length

C. GRSA-AA using 2^3 Prime Numbers

Table 3 shows the time (ms) taken by GRSA-AA (Generalized RSA-Advanced & Adaptable) algorithm for different phases like key generation, encryption and decryption using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) which uses 2^3 prime numbers i.e. 8 prime numbers. With increasing key size, time consumed also increases with higher computational complexity. Key generation time increases gradually with increase in key length where as in encryption it consumes less time and comparatively it consumes more time for decryption operation to obtain the original message from the encrypted message received.

Figure 6 shows the graphical representation for the above data from table 3, which shows time consumed for encryption, decryption and key generation for different key lengths by GRSA-AA algorithm which uses 2^3 prime numbers i.e. 8 prime numbers. We can clearly see that key generation here consumes more time than encryption and decryption for higher key length, where as decryption consumes more time than encryption operation to complete.

Table III. **Result for GRSA-AA algorithm using 2^3 primes with different key length**

Key Length (bits)	Time (ms)		
	Key Generation	Encryption	Decryption
128	25	2	15
256	102	3	84
512	338	24	648
1024	1187	132	2576
2048	5320	912	6972
4096	31159	4133	12527

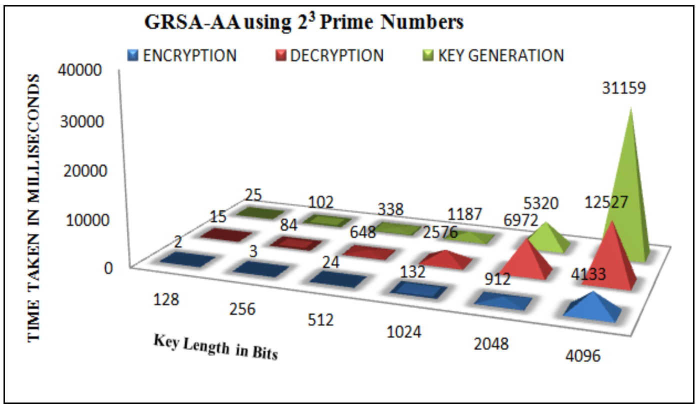

Figure 6: Graphical representation of the result for GRSA-AA algorithm using 2^3 primes with variable key length

D. GRSA-AA using 2^4 Prime Numbers

Table 4 shows the time (ms) taken by GRSA-AA (Generalized RSA-Advanced & Adaptable) algorithm for different phases like key generation, encryption and decryption using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) which uses 2^4 prime numbers i.e. 16 prime numbers, with increasing key size, time consumed also increases with higher computational complexity. Key generation time increases gradually with increase in key length where as in encryption it consumes less time and comparatively it consumes more time for decryption operation to obtain the original message from the encrypted message received.

Figure 7 shows the graphical representation for the above data from table 4, which shows time consumed for encryption, decryption and key generation for different key lengths by GRSA-AA algorithm which uses 2^4 prime numbers i.e. 16 prime numbers. We can clearly see that key generation here consumes more time for larger key length than encryption and decryption, where as decryption consumes more time than encryption operation to complete.

Table IV. **Result for GRSA-AA algorithm using 2^4 primes with different key length**

Key Length (bits)	Time (ms)		
	Key Generation	Encryption	Decryption
128	62	31	78
256	300	31	430
512	1105	78	1818
1024	6174	484	5894
2048	36802	1650	13244
4096	334806	7294	45287

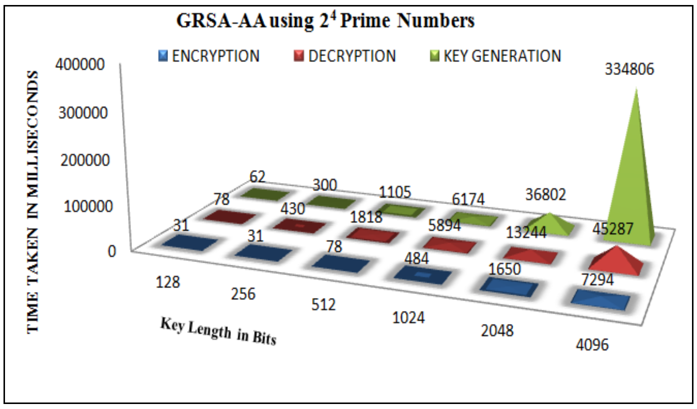

Figure 7: Graphical representation of the result for GRSA-AA algorithm using 2^4 primes with variable key length

E. GRSA-AA using 2^5 Prime Numbers

Table 5 shows the time (ms) taken by GRSA-AA (Generalized RSA-Advanced & Adaptable) algorithm for different phases like key generation, encryption and decryption using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) which uses 2^5 prime numbers i.e. 32 prime numbers, with increasing key size, time consumed also increases with higher computational complexity. Key generation time increases gradually with increase in key length where as in encryption it consumes less time and comparatively it consumes more time for decryption operation to obtain the original message from the encrypted message received.

Figure 8 shows the graphical representation for the above data from table 5, which shows time consumed for encryption, decryption and key generation for different key lengths by GRSA-AA algorithm which uses 2^5 prime numbers i.e. 32 prime numbers. We can clearly see that key generation here consumes more time for larger key length than encryption and decryption, where as decryption consumes more time than encryption operation to complete.

Table V. **Result for GRSA-AA algorithm using 2^5 primes with different key length**

Key Length (bits)	Time (ms)		
	Key Generation	Encryption	Decryption
128	327	15	215
256	1483	31	1705
512	7033	265	6342
1024	41698	1975	12366
2048	315282	15957	39151
4096	495203	22431	137112

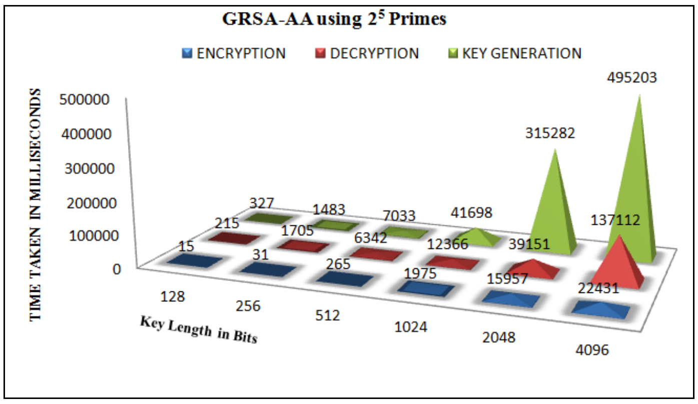

Figure 8: Graphical representation of the result for GRSA-AA algorithm using 2^5 primes with variable key length

F. Key generation time in ms for variable bit length of keys

Table 6 shows comparative analysis of key generation phase by the time (ms) taken by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes, key generation phase are using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) With all algorithms, with increasing key size, time consumption also increases with higher computational complexity. Key generation time increases gradually with increase in key length. Key generation deals with modular arithmetic and GCD methods to calculate exponential to generate & release the pair of keys as public and private key which will be used for encryption and decryption of messages.

Figure 9 shows the graphical representation for the above data from table 6, which shows time consumed by key generation phase for different key lengths by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes. We can clearly see that key generation here consumes more time for larger key length (when key length>= 2048bits). When key length increases from 1024 bits time consumption increases exponentially for increase in computational complexity.

Table VI. **Shows result for key generation time (ms) for all the algorithms using different key lengths**

Key Generatio time(ms) for variants of RS	Key Length(bits)					
	128	**256**	**512**	**1024**	**2048**	**4096**
RSA	10	23	34	79	297	577
ESRKGS	47	45	82	159	1935	17745
GRSA-AA using 8 primes	25	102	338	1187	5320	31159
GRSA-AA using 16 primes	62	300	1105	6174	36802	334806
GRSA-AA using 32	327	1483	7033	41698	315282	495203

primes

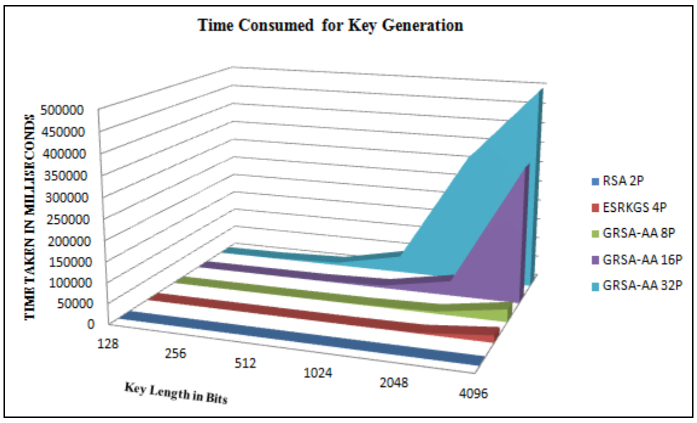

Figure 9: Graphical representation of the result for key generation algorithms using variable key lengths

G. Encryption time in ms to encrypt the same plain text with all algorithms using variable bit length of keys

Table 7 shows comparative analysis of encryption phase by the time (ms) taken by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes, encryption phase to encrypt same message using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) with all algorithms, with increasing key size, time consumption also increases with higher computational complexity. Encryption time increases gradually with increase in key length. Encryption deals with modular arithmetic to encrypt the plain text message to generate encrypted message.

Figure 10 shows the graphical representation for the above data from table 7, which shows time consumed by encryption phase of same plain text message for all algorithms with different key lengths by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes. We can clearly see that encryption here consumes more time for larger key length and with more number of primes (when key length>= 2048bits). When key length increases from 1024 bits and number of primes exceeds 16 primes, then time consumption increases exponentially for increase in computational complexity.

Table VII. **Shows result for encryption time (ms) for same plain text with all algorithms using different key length**

Key Generatio time(ms) for variants of RS	Key Length(bits)					
	128	**256**	**512**	**1024**	**2048**	**4096**
RSA	1	1	2	11	76	634
ESRKGS	1	6	17	47	254	2242
GRSA-AA using 8 primes	2	3	24	132	912	4133

GRSA-AA using 16 primes	31	31	78	484	1650	7294
GRSA-AA using 32 primes	15	31	265	1975	15957	22431

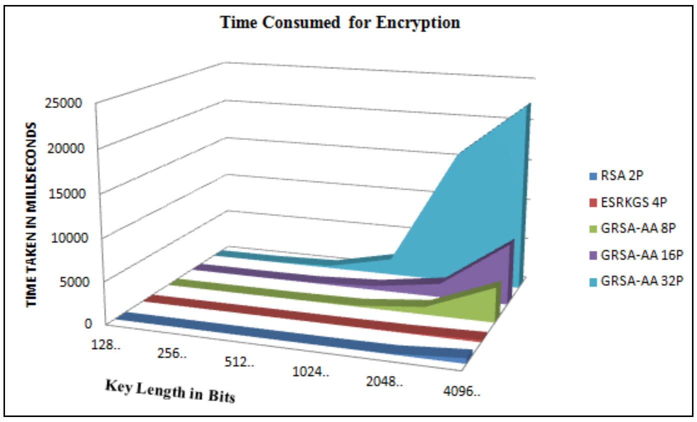

Figure 10: Graphical representation of the result for encryption operation for all algorithms

H. *Decryption time in ms to decrypt cipher text with all algorithms using variable bit length of keys*

Table 8 shows comparative analysis of decryption phase by the time (ms) taken by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes, to decrypt the received encrypted message using variable key lengths (128, 256, 512, 1024, 2048 & 4096 Bits) With all algorithms, with increasing key size, time consumption also increases with higher computational complexity. Decryption time increases gradually with increase in key length. Decryption deals with modular arithmetic to decrypt the received cipher text message to generate the original plain text message.

Figure 11 shows the graphical representation for the above data from table 8, which shows time consumed by decryption phase of the received cipher text message, for all algorithms with different key lengths by RSA, ESRKGS & GRSA-AA algorithm for different number of prime numbers based on algorithm like 2 primes for RSA, 4 primes for ESRKGS and GRSA-AA algorithm with 8 or 16 or 32 number of primes. We can clearly see that decryption here consumes more time for larger key length and with more number of primes (when key length>= 2048bits). When key length increases from 1024 bits and number of primes exceeds 16 primes, then time consumption increases exponentially for increase in computational complexity.

Table VIII. **Shows result for encryption time (ms) for same plain text with all algorithms using different key length**

Key Generation time(ms) for variants of RSA	**Key Length(bits)**					
	128	**256**	**512**	**1024**	**2048**	**4096**
RSA	0.2	10	6	45	328	2565
ESRKGS	4	30	92	342	3201	9357
GRSA-AA using 8 prime	15	84	648	2576	6972	12527
GRSA-AA using 16 primes	78	430	1818	5894	13244	45287
GRSA-AA using 32 primes	215	1705	6342	12366	39151	137112

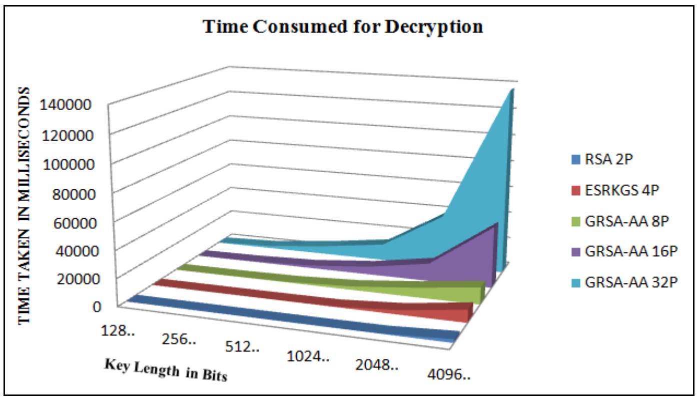

Figure 11: Graphical representation of the result for decryption operation time for all algorithms using variable key length

V. FUTURE SCOPE

The future scope of the paper is not limited as it is open for several research method implementation and new ideologies to use these findings and to use these algorithms efficiently in the future. We can perform optimization check for the algorithms to minimize the time consumed for the key generation and to enhance the performance of the algorithms by using even more number of primes and more key length for the purpose of key generation but with less time consumption.

VI. CONCLUSION

In this paper, we presented a comparative analysis approach towards RSA and its recent variants which use 2 or more than 2 prime numbers such as ESRKGS which uses 4 prime numbers and GRSA-AA (generalized RSA) scheme which use 2^k prime numbers. There is possibility for various attacks with the RSA algorithm, with increased number of primes such as 8 or 16 or 32 or even higher. It increases the computational time thus increasing the time required to guess or obtain the private key. Earlier, attackers could use integer factorization to factorize ‘n’ to determine primes used, as ‘n’ is the product of two prime numbers.

In GRSA-AA, public & private keys are not directly computed from product of primes. Public & private keys are computed through multiple steps of iterative modular and Euler functions. Hence, it is computationally infeasible for attacker to determine the private key through known factorization of n. Key generation algorithm consumes more time in GRSA-AA for more primes (i.e. 16 or more) which is comparatively much higher than earlier schemes of RSA. This makes GRSA-AA comparatively more secure and stronger than earlier schemes of RSA. Processing for encryption and decryption is more complex and decryption time depends on the key size and thus it remains same as in earlier schemes of RSA with slight increase in its time consumption. It is computationally infeasible to determine all the prime numbers used for the key generation through the product of all the primes. Based on our result, GRSA-AA with 2^k prime numbers where k>=2; is best suited for deploying more security in the distributed environment as required or needed.

VII.REFERENCES

[1] W. Stallings, “Cryptography and Network Security: Principles and Practice” (5th edition). Pearson Education, .2011.

[2] W. Diffie, M. Hellman. “New directions in cryptography: IEEE Transactions on Information Theory”; 22:644–654, 1976.

[3] Rivest, Shamir, Adleman, “Method for obtaining digital signatures and public-key cryptosystems”, Communincation ACM; 21(2):120–126, 1978.

[4] M. Thangavel, P. Varalakshmi, M. Murrali, K. Nithya, “An Enhanced and Secured RSA Key Generation Scheme (ESRKGS)”, 2014.

[5] A.K. Lenstra, “Integer Factoring Designs, Codes and Cryptography”. 19, 101–128, 2000.

[6] A. Hamid, A. Khalique, “Generalized RSA using 2k prime numbers with secure key generation”, Security Comm. Networks, 2016.

[7] D. Boneh, “Twenty Years of Attacks on the RSA Cryptosystem”, 1999.

[8] M.J. Hinek, “Cryptnalysis of RSA and its Variants”, Taylor and Francis Group, LLC, ISBN: 978-1-4200-7518-2, 2010.

[9] E. Lüya, Z.Y. Karatasb, H. Erginc, Comment on “An Enhanced and Secured RSA Key Generation Scheme (ESRKGS)”, 2016.