A REVIEW OF MECHANISM BEHIND THE RSA ALGORITHM IN CRYPTOGRAPHY: AN APPLICATION OF NUMBER THEORY

Bhoomi Gupta, Ishika, Shreya Mittal

Sri Guru Gobind Singh College of Commerce, Delhi University

Bachelor of Science (Computer Science) Hons.

Abstract

Cryptography is the process of transferring information securely, in a way that no unwanted third party will be able to understand the message. It has been used for thousands of years. Number theory and Cryptography are inextricably linked. RSA (Rivest–Shamir–Adleman) is a public-key cryptosystem, one of the oldest widely used for secure data transmission. In a public-key cryptosystem, the encryption key is public and distinct from the decryption key, which is kept secret (private). An RSA user creates and publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers are kept secret. Messages can be encrypted by anyone, via the public key, but can only be decoded by someone who knows the private key. The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers, the "factoring problem". Breaking RSA encryption is known as the RSA problem. Assume that the organization uses a weak algorithm that generates easily predictable or easy to factor random numbers. In such scenarios, attackers can easily determine the factors of the RSA modulus, which enables them to break the encryption and gain access to the private key. Therefore, the keys must be managed properly to avoid such vulnerabilities. In this research the mechanism behind such an algorithm is explained so as to avoid such vulnerabilities.

Keywords**:** *Cryptosystem, Encryption key, Decryption key, Public Key, RSA Algorithm, Factoring Problem*

1.Introduction

Cryptography is a technique of securing information and communications through use of codes so that only those people for whom the information is intended can understand it and process it. In Cryptography the techniques which are used to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet and to protect confidential transactions such as credit card and debit card transactions. In today's age of computers cryptography is often associated with the process where an ordinary plain text is converted to cipher text which is the text made such that the intended receiver of the text can only decode it and hence this process is known as encryption. The process of conversion of cipher text to plain text, this is known as decryption[1].

Types of Cryptography

In general there are three types Of cryptography.

- **Symmetric Key Cryptography**: It is an encryption system where the sender and receiver of a message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange key in a secure manner. The most popular symmetric key cryptography systems are Data

Encryption System(DES) and Advanced Encryption System(AES).

- **Hash Functions**: There is no usage of any key in this algorithm. A hash value with fixed length is calculated as per the plain text which makes it impossible for contents of plain text to be recovered. Many operating systems use hash functions to encrypt passwords.
- **Asymmetric Key Cryptography**: Under this system a pair of keys is used to encrypt and decrypt information. A receiver's public key is used for encryption and a receiver's private key is used for decryption. Public key and Private key are different. Even if the public key is known by everyone, the intended receiver can only decode it because he alone know the private key. The most popular asymmetric key cryptography algorithm is RSA algorithm.

What is Network Security ?

Network security refers to the practice of protecting Computer Networks from unauthorized access, misuse, modification, or destruction. It involves using various technologies, processes, and policies to safeguard network resources' confidentiality, integrity, and availability.

The main goal of network security is to prevent unauthorized access to network resources, such as data, applications, and devices. This is achieved by implementing various security measures, including firewalls, intrusion detection and prevention systems, access controls, encryption, and virtual private networks (VPNs)[2].

Types of Cryptography and Network Security

The importance of Cryptography and Network Security are described below.

- Protects confidential information
  Cryptography secures sensitive data such as financial transactions, personal information, and business secrets from unauthorized access, theft, and fraud.

- Prevents unauthorized access
  Network security measures such as firewalls, intrusion detection systems, and access controls ensure that only authorized users can access a network or system.

- Maintains privacy
  Cryptography techniques such as encryption and decryption help maintain privacy by keeping messages and data hidden from third parties.

- Ensures data integrity
  Cryptography provides methods for detecting whether data has been tampered with or altered.

- Enables secure communication
  Cryptography and network security enable individuals and organizations to communicate securely and exchange sensitive information without the risk of interception or eavesdropping.

It is a fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is a multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size

and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible tasRSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes, the Public Key is given to everyone and the Private key is kept private. The idea of RSA is based on the k[3].

2. Literature Review

Number theory has turned out to be one of the most useful when it comes to computer security. For example, number theory helps to protect sensitive data such as credit card numbers when you shop online[3]. This is the result of some remarkable mathematics research from the 1970s that is now being applied worldwide. Sensitive data exchanged between a user and a Website needs to be encrypted to prevent it from being disclosed to or modified by unauthorized parties. The encryption must be done in such a way that decryption is only possible if the secret decryption key is known. The decryption key should only be known by authorized parties. In traditional cryptography, which was available prior to the 1970s, the same key is used to perform the encryption and decryption operations. Establishing a shared key between the parties is an interesting challenge. If two parties already share a secret key, they could easily distribute new keys to each other by encrypting them with prior keys. But if they don't already share a secret key, how do they establish the first one? This challenge is relevant to the protection of sensitive data on the Web and many other applications like it. Your computer doesn't initially share any secret keys with Web sites. Diffie and Hellman didn't identify a method with the full public-key encryption/decryption properties that would also enable digital signatures. However, they did introduce a specific method based on number theory for establishing secret keys by parties who don't previously share a secret. The method, called Diffie-Hellman key agreement, is still in use today. The security of the method is related to a longstanding problem in number theory, discrete logarithms. The full public-key method would come a year later as an application of another famous problem, integer factorization.

The various observations just stated form the basis for the RSA public-key cryptosystem, which was invented at MIT by Ronald Rivest[4]. The public key in this cryptosystem consists of the value n, which is called the modulus, and the value e, which is called the public exponent. The private key consists of the modulus n and the value d, which is called the private exponent. An RSA public-key / private-key pair can be generated by the following steps: 1. Generate a pair of large, random primes p and q. 2. Compute the modulus n as n = pq. 3. Select an odd public exponent e between 3 and n-1 that is relatively prime to p-1 and q-1. 4. Compute the private exponent d from e, p and q. (See below.) 5. Output (n, e) as the public key and (n, d) as the private key. The encryption operation in the RSA cryptosystem is exponentiation to the e th power modulo n: c = ENCRYPT (m) = me mod n . The input m is the message; the output c is the resulting ciphertext. In practice, the message m is typically some kind of appropriately formatted key to be shared. The actual message is encrypted with the shared key using a traditional encryption algorithm. This construction makes it possible to encrypt a message of any length with only one exponentiation[5]. The decryption operation is

exponentiation to the dth power modulo n: m = DECRYPT (c) = c d mod n .

Encryption: A process of plain text (normal text) to a cipher/coded text (random sequence of bits)
Decryption: Inverse process of encryption, conversion of to a cipher/coded text to plain or decoded text
Cipher: The mathematical function, i.e. a cryptographic algorithm which is used to convert plain text to cipher
text
Key: Information that is required to induce the output of the cryptographic algorithm.

Al-Kaabi and Belhaouari[6] used several ideas on different approaches used to strengthen the RSA algorithm and improve its security in their survey work of 2019. Few RSA variants that this paper studied include variants like combining the RSA method with the Diffie-Hellman or ElGamal algorithms, modifying RSA to include three or four prime numbers, offline storing of generated keys, a secured RSA process where the message can be encrypted using dual encryption keys, and few more such variants. They have discussed a total of 10 such RSA variants. The publication also explains the methods involved in solving the Number Field Sieve (NFS), which is used to factor very large integers. It also includes a discussion of the consequences for moduli larger than RSA-768.

It has been a long road from Diffie and Hellman's discovery of public-key cryptography in 1976 and the invention of the RSA public-key cryptosystem in 1977, to the widespread deployment of public-key cryptography we see today. Public-key cryptography finds its strongest application when parties who have no prior relationship (and therefore no opportunity to establish shared secret keys) want to exchange sensitive data with each other. Throughout the 1980s, however, most of the applications that needed to protect data had centralized control. Banking networks and pay-TV systems are typical examples where secret keys could generally be pre established by a central authority. Applications that didn't have centralized control – like e-mail – were meanwhile growing without much attention to security. Equally important was the fact that the mathematical operations in public-key cryptography required considerable computational resources relative to computer performance at the time. As a result, public-key cryptography was a slow sell through that first full decade. (This was probably a good thing, because cryptographers were still learning what it would mean for a public-key cryptosystem to be "secure," and many of the initial proposals for applying the RSA algorithm would later turn out not to be.) With the advent of the World Wide Web in the 1990s, however, the situation changed. Computer performance had by then advanced to the point that the time for the encryption and decryption operations was no longer an issue. Meanwhile, the "killer application" of online purchasing had exactly the characteristics that required public-key cryptography. The Web inherently didn't have central control for security: Any merchant could go online without any prior security relationships with anyone else. And there was clearly a need to protect sensitive data: Many consumers would not shop online if there were a risk that credit card numbers and order information might be intercepted by an eavesdropper. Public-key cryptography caught on rapidly as a result[7].

Our research on the RSA algorithm brings a significant amount of mathematics to bear. For instance, we have needed to understand how best to use the algorithm to establish keys and sign messages. This requires a careful analysis of various methods for relating of the value m to the actual message or key, some of which are much better for security than others. We have

needed to understand the impact of various proposed improvements to integer factorization methods, especially in terms of recommended key sizes. This requires assessment of the effectiveness of those methods. Because of the widespread deployment of the RSA algorithm, many other researchers are looking at these same problems today. We benefit significantly from their work as we look to improve our own products and provide guidance to our customers. Our work in industry standards aims to promote the adoption of the best practices we have learned[8].

Simple concepts in mathematics – prime numbers, integer factorization, modular exponentiation – have had a dramatic impact on computer security, particularly for online commerce. The theory is working well in practice through algorithms like DiffieHellman key agreement, the RSA public-key cryptosystem and more recently elliptic curve cryptography. In cryptography, “it’s not broken” is no reason to avoid trying to fix it. Mathematicians still don’t know whether or not there are faster methods for integer factorization than the ones currently available. Research is needed to try to find faster methods, as well to try to prove that there aren’t any. A related research problem is to confirm whether modular root-extraction is or isn’t as hard as integer factorization. Interestingly, much faster methods for integer factorization already exist in theory, but they run on computers that haven’t yet been built. In particular, if one could build a full scale quantum computer, it will be possible to break a large number into its factors essentially as easily as it is to put the number together by multiplication[8]. (Such a computer would also break the Diffie-Hellman and elliptic curve algorithms.) In case one or more of the current public-key cryptosystems is broken in the future, it would be helpful to have alternatives to choose from. This is another important area for research. What other hard problems in mathematics are there from which a public-key cryptosystem and digital signature scheme might be derived? Mathematics has many more applications in computer security than just public-key cryptography, of course. The design and analysis of more traditional encryption algorithms and one-way function also has a strong mathematical component, although perhaps not one so elegant as for public-key cryptography. Intriguing mathematical constructions have also led to new types of cryptography – like identity-based encryption, a form of public-key cryptography where one’s name or e-mail address itself becomes the public key, avoiding the need for a directory. More groundbreaking applications are likely to emerge over time as knowledgeable people continue to search what’s concealed within mathematics[9].

Numerical study on RSA Algorithm

Step 1: Select primes p=11, q=3.
Step 2: n = pq = 11.3 = 33 phi = (p-1)(q-1) = 10.2 = 20
Step 3: Choose e=3 Check gcd(e, p-1) = gcd(3, 10) = 1 (i.e. 3 and 10 have no common factors except 1), and check gcd(e, q-1) = gcd(3, 2) = 1 therefore gcd(e, phi) = gcd(e, (p-1)(q-1)) = gcd(3, 20) = 1
Step 4: Compute d such that ed ≡ 1 (mod phi) i.e. compute d = e-1 mod phi = 3-1 mod 20 i.e. find a value for d such that phi divides (ed-1) i.e. find d such that 20 divides 3d-1. Simple testing (d = 1, 2, ...) gives d = 7 Check: ed-1 = 3.7 - 1 = 20, which is divisible by phi. Step 5: Public key = (n, e) = (33, 3) Private key = (n, d) = (33, 7). This is actually the smallest possible value for the modulus n for which the RSA Algorithm works. Now say we want to encrypt the message m = 7, c = me mod n = 73 mod 33 = 343 mod 33 = 13. Hence the cipher text c = 13. Step 6: To check decryption we compute m' = cd mod n = 137 mod 33 = 7. Note that we don't have to

calculate the full value of 13 to the power 7 here[10].
We can Make use of the fact that a = bc mod n = (b mod n).(c mod n) mod n so we can break down a potentially large number into its components and combine the results of easier, smaller calculations to calculate the final value. One way of calculating m' is as follows:- m' = 137 mod 33 = 13(3+3+1) mod 33 = 133.133 .13 mod 33.
=(133 mod 33).(133 mod 33).(13 mod 33) mod 33 =(2197 mod 33).(2197 mod 33).(13 mod 33) mod 33 =19.19.13 mod 33 = 4693 mod 33 =7.
Now if we calculate the ciphertext c for all the possible values of m (0 to 32), we get m 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 c 0 1 8 27 31 26 18 13 17 3 10 11 12 19 5 9 4 m 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 c 29 24 28 14 21 22 23 30 16 20 15 7 2 6 25 32

3. Methodology

The keys for the RSA algorithm are generated in the following way:
Choose two large prime numbers p and q.
To make factoring harder, p and q should be chosen at random, be both large and have a large difference. For choosing them the standard method is to choose random integers and use a primality test until two primes are found.
p and q should be kept secret.

1. Compute n = pq.

n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
n is released as part of the public key.

2. Compute λ(n), where λ is Carmichael's totient function. Since n = pq, λ(n) = lcm(λ(p), λ(q)), and since p and q are prime, λ(p) = φ(p) = p − 1, and likewise λ(q) = q − 1. Hence λ(n) = lcm(p − 1, q − 1).

3. The lcm may be calculated through the Euclidean algorithm, since lcm(a, b) = |ab|
   gcd(a, b)
   λ(n) is kept secret.

4. Choose an integer e such that 2 < e < λ(n) and gcd(e, λ(n)) = 1; that is, e and λ(n) are coprime.

e having a short bit-length and small Hamming weight results in more efficient encryption – the most commonly chosen value for e is 216 + 1 = 65537. The smallest (and fastest) possible value for e is 3, but such a small value for e has been shown to be less secure in some settings.
e is released as part of the public key.
Determine d as d ≡ e−1 (mod λ(n)); that is, d is the modular multiplicative inverse of e modulo λ(n).

This means: solve for d the equation de ≡ 1 (mod λ(n)); d can be computed efficiently by using the extended Euclidean algorithm, since, thanks to e and λ(n) being coprime, said equation is a form of Bézout's identity, where d is one of the coefficients.

- d is kept secret as the private key exponent.

The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the private (or decryption) exponent d, which must be kept secret. p, q, and λ(n) must also be kept secret because they can be used to calculate d. In fact, they can all be discarded after d has been computed.

5. Result and Discussion

The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring large numbers and the RSA problem. Full decryption of an RSA ciphertext is thought to be infeasible on the assumption that both of these problems are hard, i.e., no efficient algorithm exists for solving them. Providing security against partial decryption may require the addition of a secure padding scheme.
The RSA problem is defined as the task of taking eth roots modulo a composite n: recovering a value m such that $c \equiv m^e \pmod{n}$, where (n, e) is an RSA public key, and c is an RSA ciphertext. Currently the most promising approach to solving the RSA problem is to factor the modulus n. With the ability to recover prime factors, an attacker can compute the secret exponent d from a public key (n, e), then decrypt c using the standard procedure. To accomplish this, an attacker factors n into p and q, and computes $\operatorname{lcm}(p-1, q-1)$ that allows the determination of d from e. No polynomial-time method for factoring large integers on a classical computer has yet been found, but it has not been proven that none exists; see integer factorization for a discussion of this problem.
Multiple polynomial quadratic sieve (MPQS) can be used to factor the public modulus n.
Finding the large primes p and q is usually done by testing random numbers of the correct size with probabilistic primality tests that quickly eliminate virtually all of the nonprimes.
The numbers p and q should not be "too close", lest the Fermat factorization for n be successful. If $p-q$ is less than $2n1/4$ ($n = p \cdot q$, which even for "small" 1024-bit values of n is 3×1077), solving for p and q is trivial. Furthermore, if either $p-1$ or $q-1$ has only small prime factors, n can be factored quickly by Pollard's $p-1$ algorithm, and hence such values of p or q should be discarded.
It is important that the private exponent be large enough.

5.1 Importance of strong random number generation

If n = pq is one public key, and n′ = p′q′ is another, then if by chance p = p′ (but q is not equal to q'), then a simple computation of gcd(n, n′) = p factors both n and n', totally compromising both keys. Lenstra et al. note that this problem can be minimized by using a strong random seed of bit length twice the intended security level, or by employing a deterministic function to choose q given p, instead of choosing p and q independently.
Strong random number generation is important throughout every phase of public-key cryptography. For instance, if a weak generator is used for the symmetric keys that are being distributed by RSA, then an eavesdropper could bypass RSA and guess the symmetric keys directly.

5.2 Timing attacks

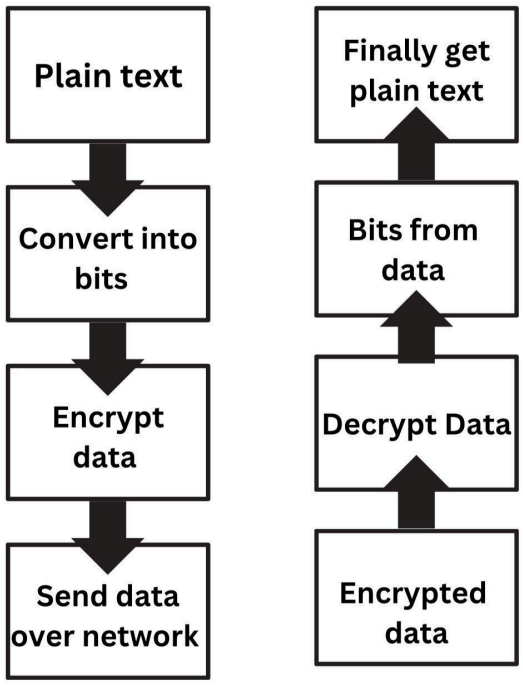
One way to thwart these attacks is to ensure that the decryption operation takes a constant amount of time for every ciphertext. However, this approach can significantly reduce performance. Instead, most RSA implementations use an alternate technique known as cryptographic blinding. RSA blinding makes use of the multiplicative property of RSA. Instead of computing $c^d \pmod{n}$, Alice first chooses a secret random value r and computes $(rec)^d \pmod{n}$. The result of this computation, after applying Euler's theorem, is $rc^d \pmod{n}$,

and so the effect of r can be removed by multiplying by its inverse. A new value of r is chosen for each ciphertext. With blinding applied, the decryption time is no longer correlated to the value of the input ciphertext, and so the timing attack fails.

5.3 Possible Defenses

There are defenses against these timing attacks. The most widely accepted method is RSA blinding. With RSA blinding, randomness is introduced into the RSA computations to make timing information unusable. Before decrypting the ciphertext C, we first compute X = reC mod N, where r is a random value and e is the public exponent. We decrypt X as usual, i.e., compute Xd mod N = red Cd mod N = r Cd mod N by Euler's theorem. We then multiply the output by r-1 to obtain Cd mod N which is the plaintext we want. Since a different r is used for each message, the original message is changed in a random way before the exponentiation operation. Thus, blinding prevents an attacker from entering a known input to the exponentiation function and using the resulting timing information to reveal the key. Blinding incurs a small performance penalty in the range of 2% to 10%.

Implementation Process



Conclusion

This research is a study of number theory and public key cryptosystems. The research paper primarily focuses on the implementation of the RSA Algorithm as how to generate the public and private key. In addition the security of the RSA Algorithm is also discussed. RSA cryptosystem produces one public key to encrypt the message. The security of the RSA key can be compromised if the two prime numbers are too close to each other in value or if one of them is too small. The keys must be managed properly to avoid such vulnerabilities. In this research the mechanism behind such an algorithm is explained so as to avoid such vulnerabilities.

References

[1] Israt Jahan, Mohammad Asif, Liton Jude Rozario, Improved RSA cryptosystem based on the study of number theory and public key cryptosystems, American Journal of Engineering Research, Volume 4, Issued 1, 2018

[2] "RSA Algorithm in Cryptography: Rivest Shamir Adleman Explained." 15 May. 2023,

[3] "Understanding the Number Theory Behind RSA Encryption." 25 May. 2020,

[4] "1-Number Theory and RSA Public-Key Encryption - Jackson State University."

[5] G. Singh, Supriya, A Study of Encryption Algorithms(RSA,DES and AES) for Information Security, 2018

[6] M. Preetha, M. Nithya, A Study and Performance Analysis Of RSA Algorithm

[7] K. Nanda Kishore, Sujan Chhetri, RSA Algorithm: A theoretical study and implementation, IRJMETS, 05 May 2020

[8] HT Sihotang, Design and Implementation of Rivest Shamir Adleman's (RSA), 2020

[9]Neha Bansal, Sukhdeep Singh, RSA Encryption and Decryption System, Computer Science 2020

[10] Taleb Samad Obaid, Study: A public key in RSA Algorithm, 1 April 2020, EJ-ENG