Enhancing the Security of Real-World Applications through the Implementation of the RSA Algorithm

Toufik Ghrib [1,2], Ahmed Ghali*[1], Fouzi Benbrahim[2], Yusef Awad Abusal[3]
[1]University of Kasdi Merbah, Ouargla, Algeria
[2]École Normale Supérieure de Ouargla, Algeria
[3]Ufa State Petroleum Technological University, Russia
E-mail : ghrib.toufik@ens-ouargla.dz, ghali.ahmed@univ-ouargla.dz, benbrahim.fouzi@ens-ouargla.dz, yousef-abusal@mail.ru
*Corresponding author

Keywords: RSA algorithm, Diffie-Hellman, data security, cryptography, K-NN classification

Received: August 18, 2024

In the recent times, during COVID-19 statics tell, cyber-attacks have increased by up to 37%. While the lockdown all the companies are leveraging technologies accessible via internet to conduct their day to today work. The amount of transfer of data and information be it personal or otherwise has also escalated. Today, cyberattack is a major threat to every company. On the other hand, companies and hackers have been trying to take the advantage of it to attack people. Thus, security of data and keeping individuals or enterprises information safe has become a crucial area of research. Cryptography provides major breakthroughs by providing techniques to encrypt the data before sharing it over internet. But with increasing computational power of technology and AI; there's a need to make advancements to these algorithms. This paper provides a novel approach to RSA algorithm to increase the security and performance. The proposed algorithm is compared with the standard RSA algorithm by varying the number of primes. A Machine Learning (ML) model is also proposed which suggests the number of primes required to encrypt the message to optimize the processing time based on the length of the message and the execution time.

Povzetek: Prispevek obravnava izboljšanje varnosti RSA algoritma za šifriranje podatkov s pomočjo modela strojnega učenja, ki optimizira uporabo praštevil glede na dolžino sporočila. Rezultati kažejo izboljšano varnost in zmogljivost v primerjavi s klasičnim RSA algoritmom.

1 Introduction

The trend of transfer of information over internet and online transactions have surged due to COVID-19. Every other enterprise or business has turned on an online platform. With the fourth industrial revolution, technology has reached at every nook and corner and have made technology as readily available as water. With every third person having a smart phone, the vices have outweighed the virtues of technology owing it to irresponsible usage as well as the abuse of technology by criminals. There will be billions of dollars spent on security all around the world, according to Gartner's projections [3]. In the wake of the COVID-19, cybersecurity is the need of the hour. Maintenance of the security of the data is crucial. Thus, advanced algorithms need to be developed to increase the security and performance of the systems. Due to digitalization, every task related to e-commerce, banking, ATM's, credit cards, aadhaar card, insurance, purchase have moved online.

Therefore, security has to be provided to the data being transmitted online and for the same several encryption algorithms are available but RSA is the most widely used. The cryptography is the application and examination of methods for safeguarding communication and guaranteeing the secrecy, integrity, and validity of data. It is a pivotal element of information security, essential for safeguarding data against unwanted access or manipulation. Cryptography guarantees the secure transmission and storage of sensitive information, including personal, financial, and organizational data. The encryption and decryption of data depends on the various algorithmic approaches established. The security of the systems varies based on the encryption and decryption technique applied to the network structure. In this paper, we propose a modified RSA algorithm application technique driven through ML model to increase the security and performance of encryption. At the heart of cryptography are two main types of encryption: symmetric encryption and asymmetric encryption. In symmetric encryption, the same key is used for both encryption and decryption. This means that both the sender and receiver must possess the secret key. Asymmetric encryption, also known as public-key cryptography, uses two keys: a

public key and a private key. The public key is used to encrypt data, while the private key is used to decrypt it. Based on the ML model prediction, we present to use n primes numbers to execute existing RSA (Rivest–Shamir–Adleman) algorithm which optimizes the usage of length of key and execution time of encrypting the data. RSA is an asymmetric cryptography algorithm which takes two types of keys i.e., public and private key. The public key is shared with everyone while the private key is kept secret. The client requests the data from the server sending the public key along, the server encrypts the data with the users' public key and sends the encrypted data back to client. Being asymmetric, only browser can decrypt the data even if some other party has the public key. RSA works a basic concept where it is difficult to factorize large prime number and it's nearly impossible for computers to do so as well. In the paper, we also propose a modified RSA algorithm to encrypt data with 'n' prime numbers. The central problem addressed by this paper is the need to enhance the RSA algorithm to meet current security demands while improving its performance.

The paper is outlined as follows, Section II of the paper discusses about the related work in the field of cryptography and RSA algorithm. Section III describes the proposed algorithm, where we input the length of the message and the estimated time of encryption and the ML model based on K nearest neighbor predicts the number of primes to be used to encrypt the message. Also, when the length if message is short, we present a novel approach to encrypt the message using 4 prime numbers increasing it security and maintaining its execution performance. In the section IV and V, we discuss about the hardware and the software used to implement the proposed algorithm along with the results obtained respectively. The last Section VI talks about the conclusion and future scope of the project.

2 Literature review

Shekhar et. al proposed a dynamic algorithm which is based on RSA algorithm to increase the efficiency and security of data over a network [4]. They proposed an algorithm which is a combination of the two most popular in asymmetric encryption-based algorithms, that are RSA algorithm and Diffie–Hellman algorithm. The Diffie–Hellman algorithm proposed by Whitfield Diffie and Martin Hellman is based on elliptic curve to generate points and get the secret key that can be used for communication of data over public network. They proposed a password protection mechanism which is based on RSA algorithm and Diffie–Hellman algorithm was applied to increase the security. First, the Diffie–Hellman algorithm is applied to the text entered which generates two keys (say for example K1 and K2), then use the standard RSA to encrypt and decrypt the message which are computed as $c = m^e \ mod \ n$ and $m = c^d \ mod \ n$ respectively.

An option to manually implement the initial values of full algorithm is kept in case the flow breaks at any stage, the input value can be changing quickly and thus the algorithm will produce completely different values increasing the security. A limitation of the paper is that the large key size and the time for computation is quite high. Sudhansu et. al proposed a novel approach to the asymmetric RSA algorithm which reduces complex calculation of key (public key and private key), cipher text and original message generation [5]. The RSA algorithm uses two different keys to encrypt and decrypt the messages. There are major drawbacks to it like the slowness of algorithm in encrypting and decrypting long data which this paper presents a solution for. They present new approach for calculating the public key k such that $kl \bmod \varphi(z) = 1$ and the private key(l) such that l is a number which is relative to $\varphi(z)$, such that $gcd(l,\varphi(z))=1$, $1 <= l <= \varphi(z)$. They compared their proposed RSA algorithm with the standard RSA algorithm in terms of Mk and $M * k$. The experiments were performed on a dual processor with 8 GB RAM and 1:80 GHz processor speed. The output depicts that their modified RSA algorithm performed better in terms of complexity of calculation which reduces the key size, but a major drawback is that the time taken to compute the keys is large. Ritu et.al presented a modified RSA cryptosystem based on offline storage [6]. They propose a novel approach to RSA algorithm by using three prime numbers instead to two used by standard RSA algorithm. To transfer information securely over a network, prime numbers are used since they are tough to factorize and hence, they increase the security.

The security of RSA majorly depends on factorization, therefore three large prime numbers making their modified RSA more secure. For increasing security, the developed an offline storage system to store the keys and values from the algorithm in two different table making it harder to for hacker to access the database tables. With enhanced security comes the drawback of large key size and large time for decryption comes in. Although they increased the security, speed of calculation still remains and issue and the optimality of prime numbers usage that decreases the key size is a major drawback. Naga Hemanth et. al propose an approach of applying mixed algorithms to reap the benefits of all the algorithms and increase security of data transmission [7]. Their implementation works in three parts - Playfair algorithm of 9 × 6 matrix, RSA algorithm, and an XOR operation. They have modified the traditional 5*5 matrix of playfair algorithm to 9*6 matrix by introducing new characters. The focus is to overcome the drawbacks of 5*5 playfair algorithm of key size and message strength. They have provided simulation results on comparing the proposed algorithm and original playfair cipher, RSA algorithms with different data sizes. Their algorithm performs better in terms of encoding more characters (making it more secure, hence increasing key strength and message strength) than in traditional playfair algorithm but the encryption and decryption time of text is comparatively higher. Lin Zou et. al propose a hybrid encryption algorithm using both AES (symmetric

encryption algorithm) and RSA (asymmetric encryption algorithm) to increase efficiency and file encryption [8].

There were two different approaches that they combine to benefit from both. They talk about Si and Tang who demonstrated proposed that small files can be encrypted by RSA algorithm but encryption for large files still remain a problem. On the other hand, Zhang et al. used AES to group large data that needs to be processed but doesn't ensure security [9]. The paper discusses about using encryption speed of the AES and high security RSA to depict a combination of AES encryption algorithm and RSA encryption algorithm applied to file encryption to increase efficiency and security. The double-layer encryption has several significant drawbacks. It fails to prevent replay attacks during the file encryption process and leaves the data vulnerable to tampering and forgery once the double key is cracked.

Moreover, employing double-layer encryption increases the complexity of decryption. Preetha et. al [10] studied various security applications and stated that users need simpler inherent security without all the complications. They compared public key RSA and enhanced RSA on the basis of execution time. They included encryption to a well-known practical RSA-OAEP algorithm. They stated that this scheme has many advantages. They also concluded that This scheme can be used to encrypt long messages without any hybridization to the algorithm. Finally, they proposed that RSA can be used for business security applications. Israt et. al [11] have proposed an improved algorithm using two public key pairs and a mathematical process. They implemented this technique to increase the end-to-end security of the algorithm instead of just sending one public key directly. Hernán Mauricio Aules Centeno et. al [12] have optimized the RSA algorithm. They have used a client and server method to implement the algorithm which led to lesser time, memory, processor and network performance when performing encryption. Their results show the efficiency of the algorithm and functionality of the algorithm in terms of information security. Mahajan et. al [13] have used GPU as a co-processor of the CPU to leverage parallel computing. They presented a novel algorithm to calculate modulus of large power numbers and they designed parallelized RSA algorithm on CUDA framework. Kamara and Lauter [14] examined Elliptic Curve Cryptography (ECC) in privacy-preserving protocols, including safe multi-party computation and homomorphic encryption. Their research shows that ECC may be used in modern cryptographic systems, especially when combined with other methods to improve security and efficiency. Another notable study by Gupta et al. [15] explored hybrid cryptography in the context of IoT, combining ECC and Lightweight AES for resource-constrained environments.

In the study of Ivy et al. [16], the authors introduce a modified RSA cryptosystem that employs multiple prime numbers ('n' primes) rather than the conventional two, thereby substantially augmenting the security of the encryption process. The updated technique enhances factorization difficulty by utilizing a bigger quantity of prime factors, hence increasing its resistance to attacks. The paper further evaluates the performance of the modified RSA system, illustrating its potential efficacy in diverse cryptographic applications while preserving computational economy. Gupta et al. [17] investigated hybrid cryptography models using ECC and Lightweight AES. Li and Wu [18] suggested merging blockchain with RSA to improve digital transaction security, whereas Xiong et al. [19] proposed a multi-prime RSA algorithm for data transmission security. Bishop and Lutz [20] also discussed modern cryptographic algorithms and their applications and future goals, highlighting cryptography's constant evolution to address security threats.

The following table summarizes the literature review, highlighting key contributions and advancements in cryptographic techniques:

Table 1: Summary of related works.

Study	Proposed Solution	Key Features	Advantages	Drawbacks
Shekhar et al.	Combined RSA and Diffie-Hellman algorithms	Used elliptic curve-based Diffie-Hellman to generate keys; applied RSA for password protection	Enhanced security through a combination of asymmetric algorithms	High computation time due to large key sizes
Sudhansu et al.	Modified RSA key generation process	Optimized the key (public and private) generation process	Reduced complexity in key calculation and encryption	Time taken to compute keys is still high
Ritu et al.	Modified RSA with three primes	Introduced three prime numbers for RSA key generation	Increased security through harder factorization	Larger key sizes, slow decryption process
Naga Hemanth et al.	Hybrid approach using Playfair, RSA, and XOR operations	Modified Playfair algorithm (9x6 matrix), combined with RSA and XOR for data encryption	Improved message strength, key strength	Higher encryption and decryption times
Lin Zou et al.	Hybrid AES and RSA encryption	Combined AES for fast encryption with RSA for security	Improved file encryption efficiency and security	Vulnerable to replay attacks, complex decryption process
Preetha et al.	RSA-OAEP algorithm for long messages	Enhanced RSA using Optimal Asymmetric	Secure encryption of long messages without hybridization	No significant drawbacks noted

		Encryption Padding (OAEP)		
Israt et al.	Two public key pairs for improved RSA	Added a mathematical process to use two public key pairs for better security	Increased end-to-end security	Added complexity to key management
Hernán Mauricio Aules Centeno et al.	Optimized RSA for client-server architecture	Reduced memory, processing time, and network performance	Improved efficiency in encryption processes	No major drawbacks noted
Mahajan et al.	Parallel RSA on GPU	Used GPU alongside CPU to compute RSA modulus in parallel	Faster encryption through parallelization on CUDA framework	Requires specialized hardware (GPU)
Gupta et al.	Hybrid Cryptography in IoT (ECC and Lightweight AES)	Combined ECC and AES for IoT encryption, balancing lightweight encryption and security.	Increased encryption efficiency in constrained settings (IoT), better security for low-power devices.	Implementation complexity, potential performance bottlenecks in highly constrained systems.
Ivy, B. P. U., Mandiwa, P., & Kumar, M	Modified RSA Cryptosystem using 'n' prime numbers	RSA modified to use more than two prime numbers for harder factorization	Increased security by making factorization more difficult, stronger than standard RSA	Requires greater computational power, leading to slower decryption times and higher resource consumption.
Li, Z., & Wu, Y.	Blockchain-enhanced RSA	Combines blockchain with RSA	Improved security and integrity in transactions	Increased complexity and resource requirements
Xiong, C., Liu, Y., & Zhang, Y.	Multi-prime RSA Algorithm	Focus on data transmission security	Enhanced security in data transmission	Complexity in key management
Bishop, D. C., & Lutz, T.	Review of Cryptographic Techniques	Overview of contemporary algorithms	Highlights future directions in cryptography	Generalization may overlook specific solutions

3 Proposed algorithm

In the approach, the plain text is encrypted through the algorithm varying the number of primes to observe the execution time for differing length of plain text. Next, the observed values obtained from the algorithm — the length of text, execution time, public key, private key and the cipher text form the data frame. The obtained data frame is then put through an ML model to predict the use of number of primes based on the length of plain text and the execution time.

The algorithm consists of two main parts: the RSA encryption process and the machine learning model for predicting the optimal number of primes to use.

Creating dataset:

1. Create a file containing number of primes, example primes.txt
2. Iterate over a list of no of primes to be used to encrypt the message, let say for example, Iteration 1, number of primes = x
3. Enter the message to encrypt m
4. Read x number of primes randomly from primes.txt with t iterations and append those to an array, let say array = [p1, p2, …p_x]
5. Note the start and end time before and after the encryption to calculate the execution time as end time – start time
6. Write a function to generate keypairs i.e., the public key (e) and the private key (d) along with n and $\emptyset$ (where n = multiplication of x prime numbers; p1*p2*…..*p_x and phi = (p1-1)*(p2-1)*…….(p_x-1)) using p1 prime numbers
7. In the above function calculate e as $gcd\,(e, \emptyset(n)) = 1$ and $1 < e < \emptyset(n)$ and d as $e - 1(mod\ \emptyset(n))$
8. Then create a function to encrypt the message m with public key e such that $Encrypted\_message = m^e\ mod\ n$ and where m<n
9. Now, create a function to decrypt the message such that $m = Encrypted\_message^d\ mod\ n$

Lastly, created a data frame to populated the calculated values of execution time, public key, private key, encrypted message, number of primes used, length of the message for each of the primes used separately and write it to a .csv file.

Machine Learning model:

1. Read the above created datasets and append that to a single data frame to draw patterns from the available data
2. Perform data cleaning and pre-processing if necessary, such as replacing the null values with the mean or removing the outliers to improve the data quality.
3. Then, the feature matrix is extracted from the available date to segregate the prediction

(dependent variable) and predictors (independent variables)

4. Next important step is to visualize the data, for which create a 3D scatter plot with the execution time, length of the message and number of primes used
5. Since our data has discrete values at output, use K-NN model to classify the data with k as x
6. Before performing the classification, the data is split into test and train data in a ratio of 80:20
7. Later, validation was done with the test data along with calculating the accuracy of the model

Here's the part of Python code of the model:

```
python
import pandas as pd
import numpy as np
import random
import time
from math import gcd
from sklearn.model_selection import
train_test_split
from sklearn.neighbors import
KNeighborsClassifier
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Function to create a dataset of prime numbers
def create_dataset(file_name='primes.txt',
num_primes=100):
    primes = []
    for num in range(2, 1000):  # Adjust range
as needed for more primes
        for i in range(2, int(num**0.5) + 1):
            if num % i == 0:
                break
        else:
            primes.append(num)
    with open(file_name, 'w') as f:
        for prime in primes[:num_primes]:
            f.write(f"{prime}\n")

# RSA Encryption function
def rsa_encryption(message, x):
    # Read x random primes from the file
    with open('primes.txt', 'r') as f:
        primes = [int(line.strip()) for line in
f.readlines()]
    selected_primes = random.sample(primes, x)

    # Calculate n and φ(n)
    n = np.prod(selected_primes)
    phi = np.prod([p - 1 for p in
selected_primes])

    # Generate public key (e) and private key
(d)
    e = random.randint(2, phi-1)
    while gcd(e, phi) != 1:
        e = random.randint(2, phi-1)

    # Calculate d
    d = pow(e, -1, phi)

    # Encrypt the message
    encrypted_message = pow(message, e, n)
```

4 Hardware and software used

The configuration used during the experiment was i3 Processor with 2.00 GHz and 8 GB RAM. The algorithm was implemented on the following software Python 3.7, Jupyter Notebook.

An Open Hardware Monitor software was used to observe the CPU usage during the execution of the algorithm.

5 Result

The first five rows of the dataset developed is as follows:

Table 2: First five rows of dataset.

	Execution Time	public key	private key	Len	Message	Encrypted
0	0.031248	(2197, 5917)	(3133, 5917)	3	cat	[4612, 1940, 4094]
1	0.015606	(2237, 2587)	(1829, 2587)	3	cat	[164, 197, 1923]
2	0.015624	(19, 274)	(179, 274)	3	cat	[63, 245, 134]
3	0.015610	(1561, 2329)	(3241, 2329)	3	cat	[1448, 124, 1091]
4	0.031248	(8481, 9379)	(10817, 9379)	3	cat	[4467, 8705, 2926]

The finding that it is not possible to use more than four prime integers for RSA encryption is mostly a result of the computational limits that are imposed by the capabilities of the hardware that is now available. When we utilize a modified version of the RSA method that makes use of four prime numbers, we are able to strike a balance between increasing the level of security and preserving the level of efficiency. The inclusion of each prime integer in the RSA algorithm markedly enhances the intricacy of the encryption and decryption procedures. The required computations get exponentially more complex, resulting in longer execution times and the need for more powerful hardware. Many traditional systems may find it challenging to manage this increased complexity effectively, leading to prolonged processing durations and elevated resource utilization.

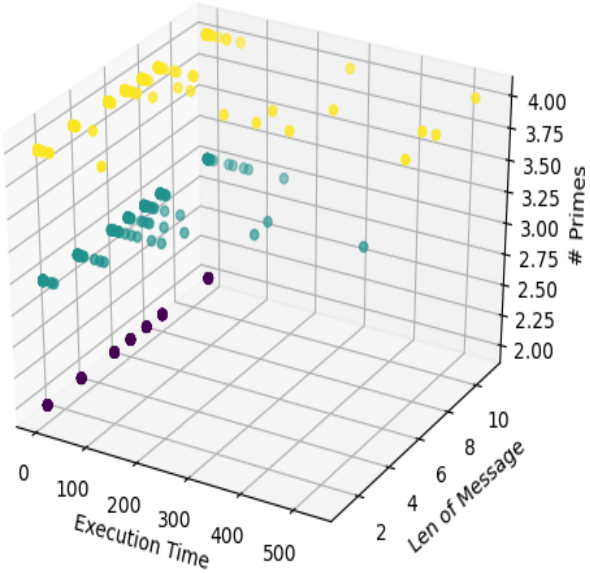

Figure 1: 3D Scatter plot of execution time, length of message and number of primes used.

The above graph shows the relationship between the execution time, number of primes used to generate the cipher text and the input length of plain text. The yellow, green and purple color depicts the plain text encrypted using 4 prime numbers, 3 prime numbers and 2 prime numbers respectively. An important outcome and observation from data visualization is that we can use 4 prime numbers to encrypt data of small lengths with short execution time increasing the security.

The accuracy of the obtained model was 66.66% due to lack of data, but it was observed that increasing the dataset the accuracy of the model increased to 68.94%.

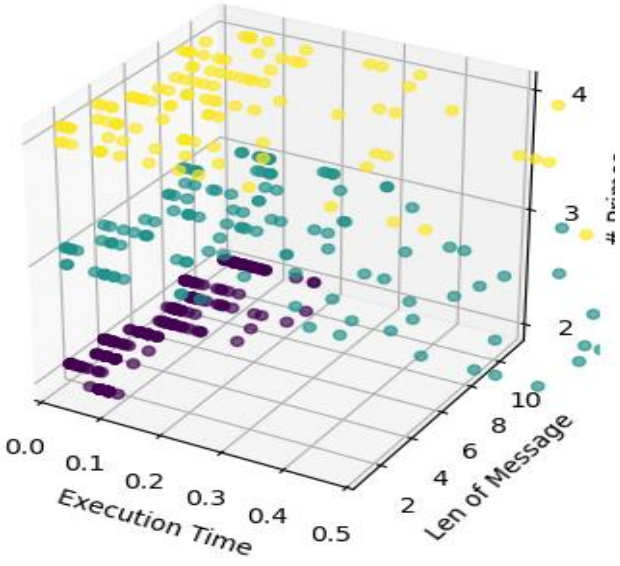

Figure 2: 3D Scatter plot of execution time, length of message and number of primes used with a larger dataset.

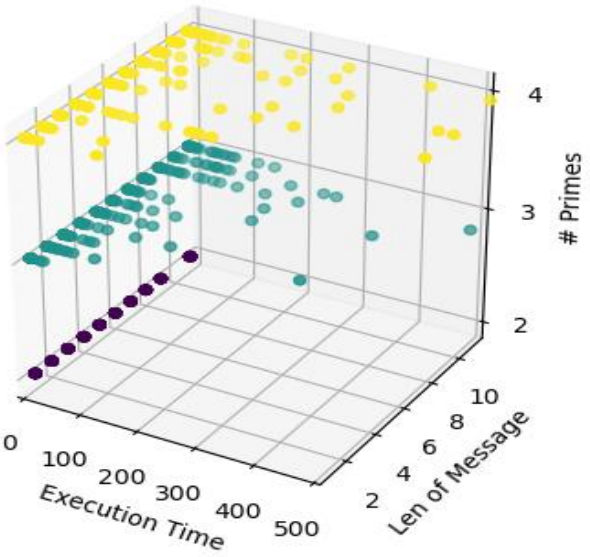

Figure 3: 3D Scatter plot categorizing the data into three different sets of prime numbers used.

Also, the algorithm decides the best and optimized number of primes to be used to encrypt the data based on experiences enabling effective use of CPU and memory. The following graph shows the CPU usage in percentage (%) with respect to the time consumed. We observe that for 3 prime and 4 prime numbers in between the graph overlaps indicating encryption 4 prime numbers will also take same time and CPU usage as 3 prime numbers and hence can be used for encrypting the plain text with increasing security.

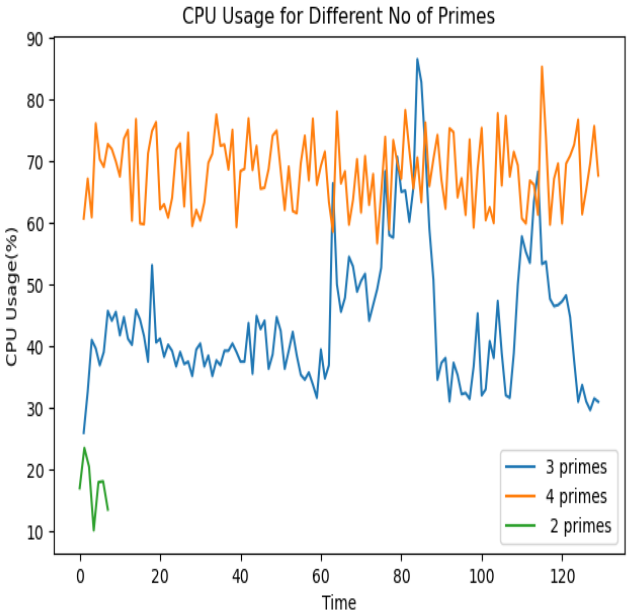

Figure 4: CPU usage based on various primes number with respect to time.

6 Conclusion and future scope

In this paper, we modified the traditional RSA algorithm to encrypt the message using 4 primes number with short length of message and short execution time. Also, created a database to implement ML model which intelligently predicts the number of primes to be used to optimize memory usage, execution time, and processing. An important observation while processing and creation of dataset was increasing the usage of number of primes increasing the security but it's processing is limited to the hardware used. But with the advent of supercomputers and processors which run at high speed, the implementation of RSA is possible by increasing the number of primes as it becomes very hard to factorize large prime numbers. However, there are still some improvements that can be made by increasing the dataset to increase the accuracy of the model, and implementing the modified RSA with more prime numbers so as to draw further conclusions to increase the performance of model. Although our enhancements have shown significant gains, chances for further development still exist. Furthermore, investigating the use of the modified RSA with additional prime numbers may provide deeper insights into speed improvement and security enhancement. This study enhances the RSA algorithm and establishes a foundation for future research and practical applications in secure communications. By integrating classical cryptography with contemporary computational methods, we enhance the endeavor to protect sensitive data in a progressively interconnected digital environment. The ongoing advancement of hardware capabilities, together with our suggested improvements, ensures a strong future for cryptographic security in multiple areas.

References

[1] Muncaster, P. (2020). Cyber-attacks up 37% over past month as #COVID-19 bites. *Info Security Magazine*. Available:https://www.infosecurity-magazine.com/news/cyberattacks-up-37-over-past-month/

[2] Talalaev, A. (2020). Website hacking statistics in 2020. *WebARX*. Available: https://www.webarxsecurity.com/website-hacking-statistics-2018-february/

[3] Moore, S., & Keen, E. (2018). Gartner forecasts worldwide information security spending to exceed $124 billion in 2019. Available: https://www.gartner.com/en/newsroom/press-releases/2018-08-15-gartner-forecasts-worldwide-information-security-spending-to-exceed-124-billion-in-2019

[4] Mondal, H. S., Hasan, M. T., Hossain, M. M., Arifin, M. M., & Saha, R. (2019). A RSA-based efficient dynamic secure algorithm for ensuring data security. In *Proceedings of International Joint Conference on Computational Intelligence, Algorithms for Intelligent Systems* (pp. 54-59). Springer Nature Singapore Pte Ltd. doi: https://doi.org/10.1007/978-981-13-7564-4_54

[5] Das, S. B., Mishra, S. K., & Sahu, A. K. (2019). A new modified version of standard RSA cryptography algorithm. In *Smart Computing Paradigms: New Progresses and Challenges* (pp. 1-10). Springer Nature Singapore Pte Ltd. doi: https://doi.org/10.1017/nlp.2024.26

[6] Ritu Patidar, Ms., & Bhartiya, R. (2013). Modified RSA cryptosystem based on offline storage and prime number. In *International Conference on Computational Intelligence and Computing Research* (pp. 1-6). IEEE. doi: 10.1109/ICCIC.2013.6734176

[7] Naga, P. H., Raj, N. A., & Yadav, N. (2018). Secure data transfer by implementing mixed algorithms. In *Recent Findings in Intelligent Computing Techniques* (pp. 1-10). Springer Nature Singapore Pte Ltd. doi: https://doi.org/10.1007/978-981-10-8639-7_8

[8] Lin, Z., Ni, M., Huang, Y., Shi, W., & Li, X. (2020). Hybrid encryption algorithm based on AES and RSA in file encryption. In *International Conference on Frontier Computing* (pp. 1-7). Springer Nature Singapore Pte Ltd. doi: https://doi.org/10.1007/978-981-15-3250-4_68

[9] Zhang, J., Wu, N., Zhou, F., Ge, F., & Zhang, X. (2019). Securing the AES cryptographic circuit against both power and fault attacks. *Journal of Electrical Engineering & Technology*, 14(5), 2171-2180. doi: https://doi.org/10.1007/s42835-019-00226-6

[10] Preetha, M., Madhukumar, & Nithya, M. (2013). A study and performance analysis of RSA algorithm. *International Journal of Computer Science and Mobile Computing*, 2(6), 126-139.

[11] Israt, J., Asif, M., & Rozario, L. J. (2015). Improved RSA cryptosystem based on the study of number theory and public key cryptosystems. *Online Journal of Communications and Networks*, 3(1), 1-10. doi: https://doi.org/10.1109/ojcoms.2015.3057679

[12] Hernán, C. M. A., Meneses, F., Fuertes, W., Sancho, J., Salvador, S., Flores, D., Castro, F. O., Torres, J., Jiménez Miranda, A., & Nuela, D. (2016). RSA encryption algorithm optimization to improve performance and security level of network messages. *IJCSNS International Journal of Computer Science and Network Security*, 16(8), 1-6.

[13] Mahajan, S., & Singh, M. (2014). Analysis of RSA algorithm using GPU programming. *International Journal of Network Security & Its Applications*, 6(4), 13–28. doi: https://doi.org/10.5121/ijnsa.2014.6402

[14] Kamara, S., & Lauter, K. (2021). "Cryptographic Cloud Storage." Lecture Notes in Computer Science, 6054, 136–149. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-18368-3_9.

[15] Gupta, M., Sharma, V., & Patra, M. R. (2021). Hybrid cryptography for IoT applications using ECC and

lightweight AES. International Journal of Information Security, 15(3), 287-299. https://doi.org/10.1007/s10207-021-00551-1

[16] Ivy, B. P. U., Mandiwa, P., & Kumar, M. (2012). A modified RSA cryptosystem based on ‘n’ prime numbers. International Journal of Engineering and Computer Science, 1(2), 63-66.

[17] Gupta, A., Jain, A., & Choudhary, A. (2022). *Hybrid Cryptography for Internet of Things: Integrating ECC and Lightweight AES*. IEEE Internet of Things Journal, 9(5), 3295-3306. DOI: 10.1109/JIOT.2021.3072430.

[18] Li, Z., & Wu, Y. (2023). Enhancing RSA Algorithm with Blockchain Technology for Improved Security and Integrity. Journal of Information Security and Applications, 73, 103179. DOI: 10.1016/j.jisa.2023.103179.

[19] Xiong, C., Liu, Y., & Zhang, Y. (2023). An Efficient Multi-Prime RSA Algorithm for Secure Data Transmission. Future Generation Computer Systems, 146, 210-220. DOI: 10.1016/j.future.2023.06.026.

[20] Bishop, D. C., & Lutz, T. (2023). A Review of Contemporary Cryptography: Algorithms, *Applications, and Future Directions*. Journal of Cryptology, 36(1), 123-148. DOI: 10.1007/s14531-022-00359-7.