ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

Implementation of Hybrid Encryption Algorithm

Mohit Chotani[1]; Prerna Singh[2]

Abstract:- This paper presents an analysis of a message encryption scheme that harnesses the strengths of both the RSA and AES algorithms. Through a detailed comparison of RSA and AES in terms of encryption and decryption time, security, key management, and key length, we developed a hybrid scheme that leverages the speed advantage of AES for encryption operations and the stability and key management strengths of RSA. Our analysis demonstrates how the integration of these two algorithms can enhance overall security and performance, effectively combining their respective advantages.

➢ ***Motivation***

There are various applications using client- server mechanism but the communication channel is not secure that much and existing methods are not so efficient thus there is need to come up with an efficient and better approach.

➢ ***Objective***

Our objective through this project is to constructa hybrid algorithm which is capable of overcoming the disadvantages of the existing methods and provide us with muchbetter approach in terms of various parameters.

Keywords:- *AES, RSA, Hybrid Encryption, Hybrid Decryption, Digital Signature, Hash Algorithm.*

I. INTRODUCTION

The cloud is mostly used by individuals and businesses to store their data. You must register with the cloud in order to use this feature. Cloud only makes their services available toregistered clients after successfully authenticating them. Theprocess of authentication is necessary to ensure privacy. Single level authentication is more common, but multifactor authentication is advised for financial services or critical data security.

When symmetric encryption techniques are employed, the process of encrypting and decrypting files using asymmetric encryption techniques takes far less time than whensymmetric techniques are utilized.

However, because the key is shared by the sender and the receiver in a symmetric manner, trust in the key security is a significant problem. In order to protect the data from outsideparties, the hybrid model, which combines symmetric and asymmetric encryption algorithms, ultimately raises the bar for security. AES is commonly used for symmetric encryption to encrypt modest to big amounts of data. For the purpose of encrypting only minimal data, such as keys, passwords, and metadata, RSA is a well-known and widely used asymmetric encryption algorithm.

To reduce time complexity and increase security, combine AES and RSA in a hybrid architecture after first analyzing AES and RSA separately to assess their performance.

II. RELATED WORKS

A. Modified Advanced Encryption Standard Algorithm for information security

The paper proposes an enhanced AES algorithm through innovative changes to the SubBytes and ShiftRows transformation. In this improved version, the SubBytes transformation is made dependent on round key, while the ShiftRows transformation is randomized. These modifications are designed to increase the algorithm security by ensuring that even a single bit variation in the key leads to a significant and unpredictable change in the resulting ciphertext. By making both transformations dependent on round key, the algorithm not only enhances resistance to cryptographic attacks but also improves diffusion and confusion properties, thereby strengthening the overall encryption process.

B. Encryption Techniques for Smart Systems Data Security Offloaded to the Cloud

This paper conducts a comprehensive analysis of various data protection and encryption methods, evaluating them based on common criteria and visually illustrating their workflows. The objective of this survey is to compile and examine the prevalent data encryption techniques discussed in existing literature, specifically focusing on securing smart system data that is offloaded to cloud computing environments. By bringing these diverse encryption strategies together, the paper aims to provide a unified resource for enhancing data security in cloud-based smart systems. Additionally, this analysis highlights the strengths and weaknesses of each method, offering insights for future research and development in the field of cloud security.

C. Ensuring Security of Encrypted Information by Hybrid AES and RSA Algorithm with Third-Party Confirmation

This article presents a novel approach to securing encrypted data through a hybrid AES and RSA algorithm, supplemented by third-party verification. The proposed system ensures secure authentication, preserving both message confidentiality and the legitimacy of the communicating entities. The primary objective behind designing this encryption algorithm is to safeguard against unauthorized access and attacks. Additionally, this paper outlines a robust encryption strategy aimed at enhancing data security and privacy, effectively protecting sensitive information from unauthorized individuals.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

D. A Study of AES and RSA Algorithms based on GPU

Examining the parallel execution of AES and RSA algorithms on GPUs, this research paper significantly boosts performance and energy efficiency compared to traditional CPU-based implementations. By harnessing the parallel processing power of GPUs, this study aims to optimize cryptographic operations, resulting in notable improvements in both computational speed and energy consumption.

E. Performance Evaluation of RSA, ElGamal, and Paillier Partial Homomorphic Encryption Algorithms

The research provides fundamental insights into the implementation of homomorphic and public key techniques, focusing on the analysis of unpadded RSA, Paillier, and ElGamal algorithms. The evaluation encompasses various criteria such as confidentiality, homomorphism, encryption/decryption speed, memory utilization, and encryption throughput. Furthermore, it investigates the impact of different file and key sizes on these criteria, offering a comprehensive understanding of their interplay in real-world applications.

F. A RSA-Based Efficient Dynamic Secure Algorithm for Ensuring Data Security

In contemporary times, ensuring data security poses a significant challenge for service providers. To bolster both performance and security in data communication, diverse techniques are being deployed. This paper introduces a method employing the RSA algorithm by Ron Rivest, Shamir, and Adleman, along with the Diffie-Hellman algorithm, to address this challenge.

G. A RSA algorithm simulation method using C language

In response to the prevalent issue of data leakage in the Internet era, this study employs the RSA algorithm for data encryption. It begins by introducing fundamental concepts of cryptography, followed by an exposition of the core principles underpinning the RSA algorithm. Utilizing the C language, the paper proceeds to simulate the RSA algorithm and implements a client-server (C/S) structure for transmitting ciphertext. Through experimentation, the findings demonstrate the efficacy of the RSA algorithm in encrypting data.

H. Classification of Security Attacks in VANETs

Attacks were classified into various categories- Attack on authentication, Privacy etc and solutions were provided suchas Encryption model. It also focuses on various network challenges that arise in VANETs such as D-MAC design, Wireless access technology and its various security aspects. The paper lacked solution to various specific attacks such as Sybil, replay attack and was more generalized into categories- Privacy, Authentication.

I. An Analytical Study of Routing Attacks in VANETs

Here, new method named Attacked Packet Detection Algorithm (APDA) was proposed which is used to detect invalid requests & attacked packets and the information is stored in Road Side units (RSU). The proposed method may be useful for various attacks such as Wormhole or Blackhole that have attacked packets but it won't be useful for Replay attack in which the data packet may not modified but sent again and again which may lead to network delay.

J. REPLACE: A Reliable Trust Based Platoon Service Recommendation Scheme in VANET

This approach presents a recommended security solution for protecting vehicle communications in ad hoc vehicle networks. Dubbed hybrid key cryptography, this security framework is uniquely designed to shield VANETs. Although both techniques involved are not new and may add overhead in dynamic networks, they effectively ensure secure communication through key exchange without experiencing any noticeable performance degradation.

III. GAPS IDENTIFIED IN EXISTING METHODOLOGIES

A. AES Algorithm

- Key management poses challenges, making the distribution and security management of keys somewhat complex.
- AES necessitates both parties to agree on a pre-shared key for encryption and decryption, with the crucial task of ensuring that the key remains inaccessible to third parties to prevent potential breaches.
- Its algebraic structure is overly simplistic.
- Implementation of AES in counter mode is particularly intricate when considering both performance and security aspects.
- Each block is consistently encrypted in the same manner, potentially compromising security.
- Implementing AES with software proves challenging.

B. RSA Algorithm

- Suited for encrypting small data volumes.
- The algorithm's complexity, characterized by large key sizes and extended calculation times, poses challenges.
- The involvement of large numbers results in a slow data transfer rate.
- Decryption requires substantial processing power at the receiver's end.
- Losing the private key renders all received messages undecipherable.

IV. PROPOSED WORK / METHODOLOGY

➢ *Module Description*

We have 2 modules in our project. One is Client Module, inour connected car use case Client is the car and the other module is Server Module and Cloud to which the car makesrequest is our server.

➢ *Client Module*

In the client side, we have the hash algorithm which converts the message into message digest which is further passed on to the RSA algorithm along with Client Private Key in order to generate Client Digital Signature token which can be verified against theServer Digital token in order to verify

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

the same. This is the first part of the Client-Side implementation.

Further, we have special secret key in our algorithm which along with server public key is passed on to RSA algorithm to generate encrypted secret key whichwill be used on server side.

After this in final stage, we pass the secret key and message to AES algorithm to generate cipher text.

The contents of cipher text generated are as follows:
+Encrypted Secret Key
+Client Signature
+Client Public key

Now, we send this cipher text generated to Server Side for server-side implementation.

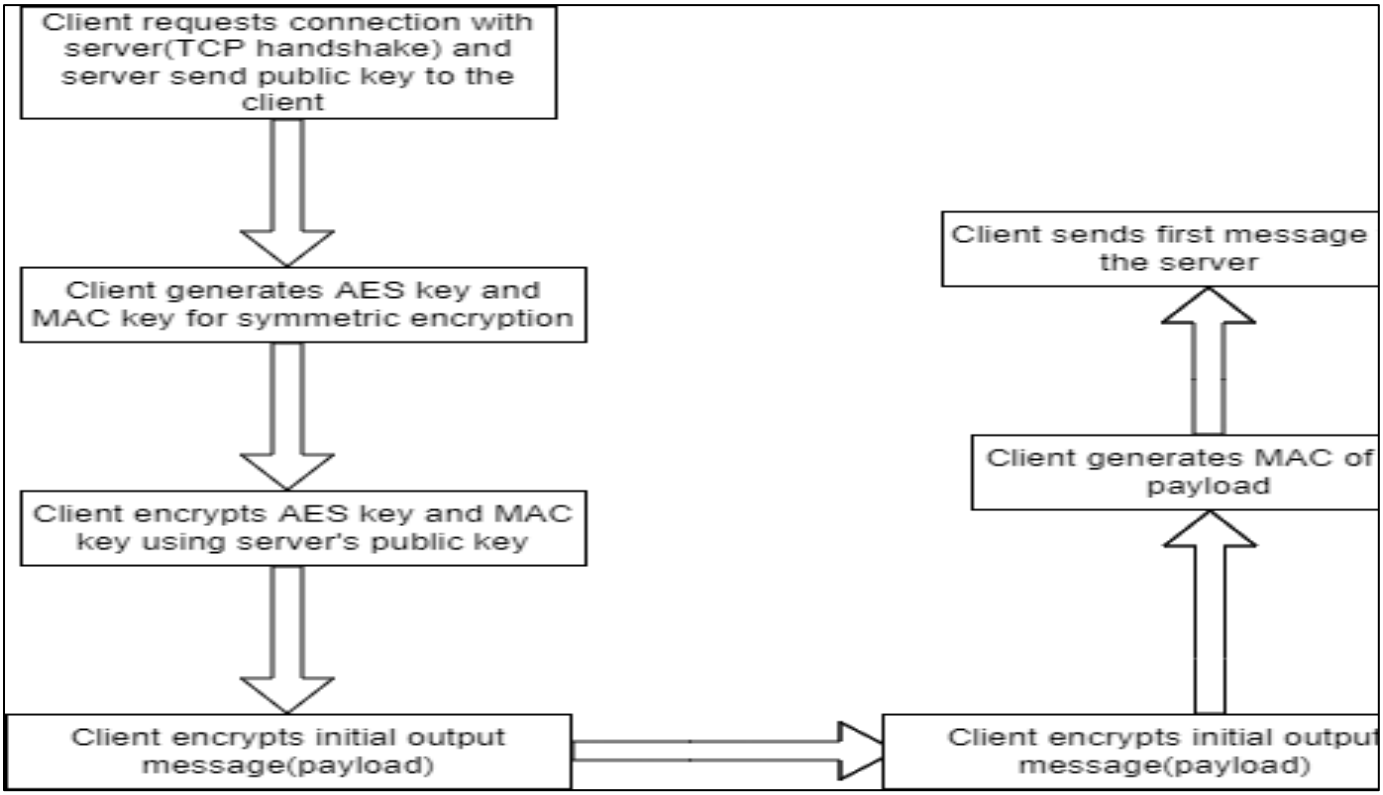


Fig 1: Client Side

➢ *Server Module*

In the server side, the received encrypted secret key along withthe server private key is used to pass on to RSA algorithm inorder to decrypt and obtain the secret key.

After we have obtained the secret key now we use thisto pass it to AES decryption logic and obtain back themessage.

After that we have the decrypted message but now still we need to verify the digital signature of the Client who has requested. So, we again pass this message tohash algorithm in order to generate message digest andnow this message digest along with Client Public Keyis passed to RSA algorithm to verify the Client digitalsignature token.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

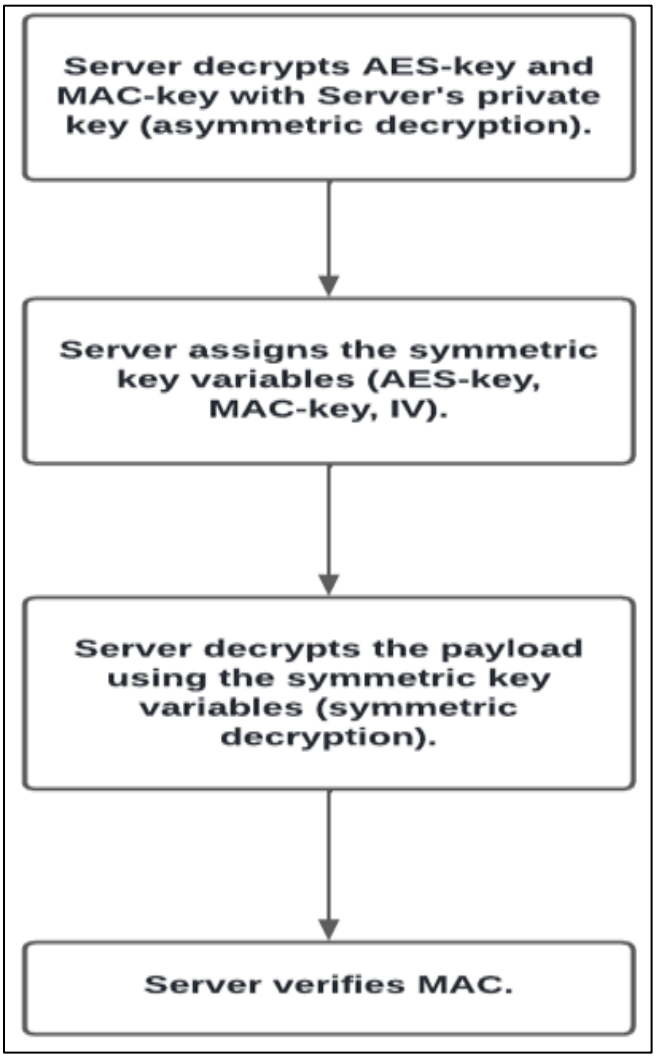


Fig 2: Server Side

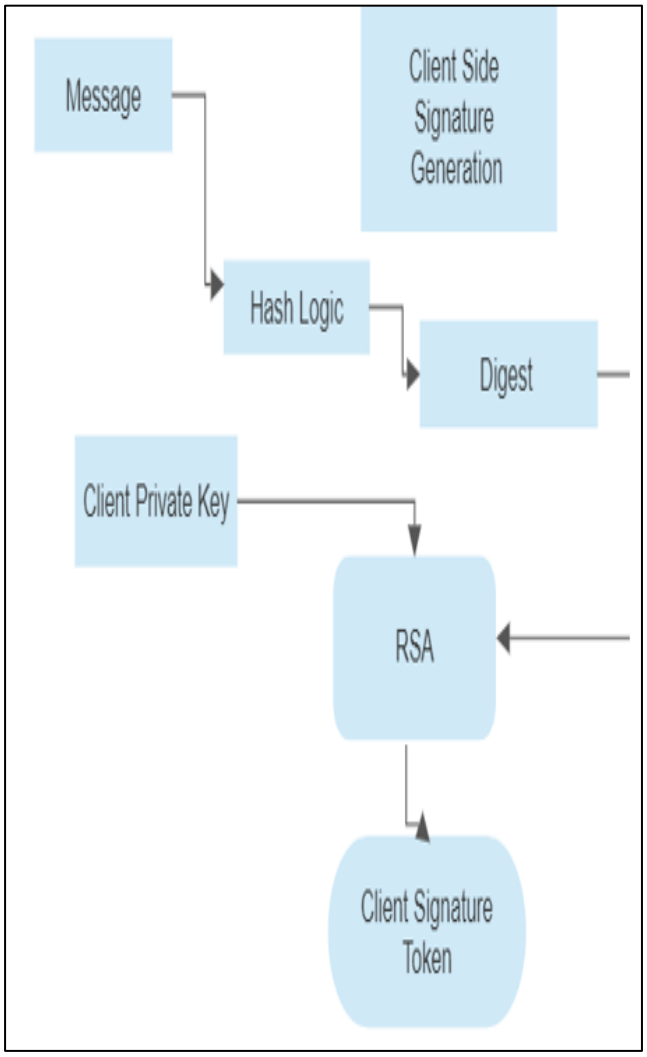


Fig 3: Client Side Signature Verification Procedure

V. ALGORITHM- PSEUDO CODE/ EQUATIONS

A. Client Side

BEGIN
Message->Input Secret Key->Input
Public Key Parameters, Private Key Client-> Input Request server for public key using JSON
If received response from server display "received" Ciphertext-> call aes_encryption(message,secret key)
Encrpyted_secret_key-> call rsa(secret key, server public key)
Generate message digest
Arrange the data and send to server using JSON and sockets (cipher_text)
END

B. Server Side

BEGIN
Using Sockets connect to Client (TCP session) Public key Paarameters, Private Key-> Input Receive the ciphertext in json format
Secret_key -> Call decrypt(encrypted_secret key, server public key)
Message-> Call aes_decryption (secret_key ,ciphertext)
Generate message digest
Verify Client Digital Signature Token
END

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

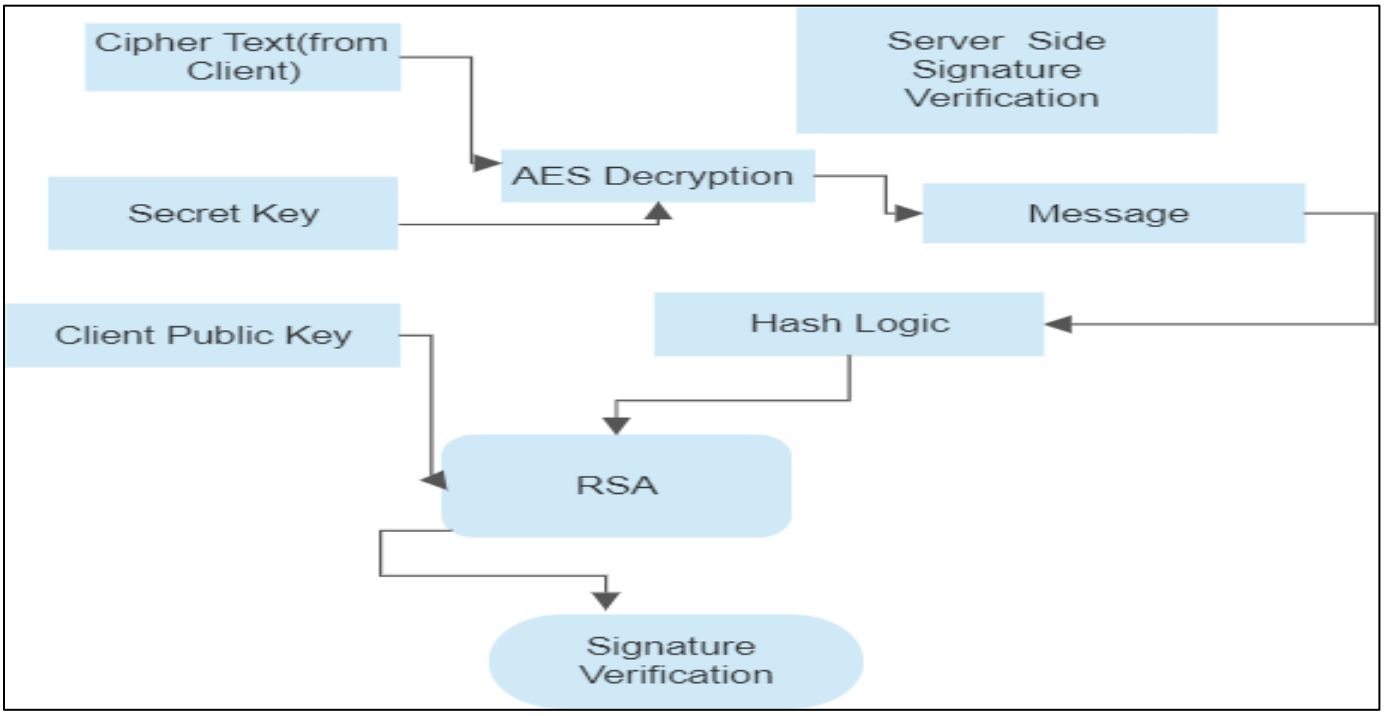


Fig 4: Server- Side Signature Verification

C. Implementation

```
C:\Users\djbud\Desktop\AES-RSA-Encryption-main>python client_main.py
Client And Server are Connected
Enter message :10010101
Enter secret key:1001
Enter the public keys of client: 2 34
Enter the private key of client: 32
Enter ' y ' to request server for it 's public key: y
Public Key Received!
<------------------OUTPUT------------------>
Message 149
Public Key Parameters 2 and 34
Private Key Parameters 32
Encrypted Secret Key 13
Digest f2217062e9a397a1dca429e7d70bc6ca
Digital Client Signature 18
```

Fig 5: Client Side

The figures 7 depicts the performance of Hybrid Algorithmshowing the elapsed time and CPU time of the algorithm. Thefigure 8 shows the graph of CPU utilization time taken by hybrid algorithm. Similarly figure 9 and 10 shows the performance of RSA algorithm in terms of execution time and CPU utilization for the same respectively.

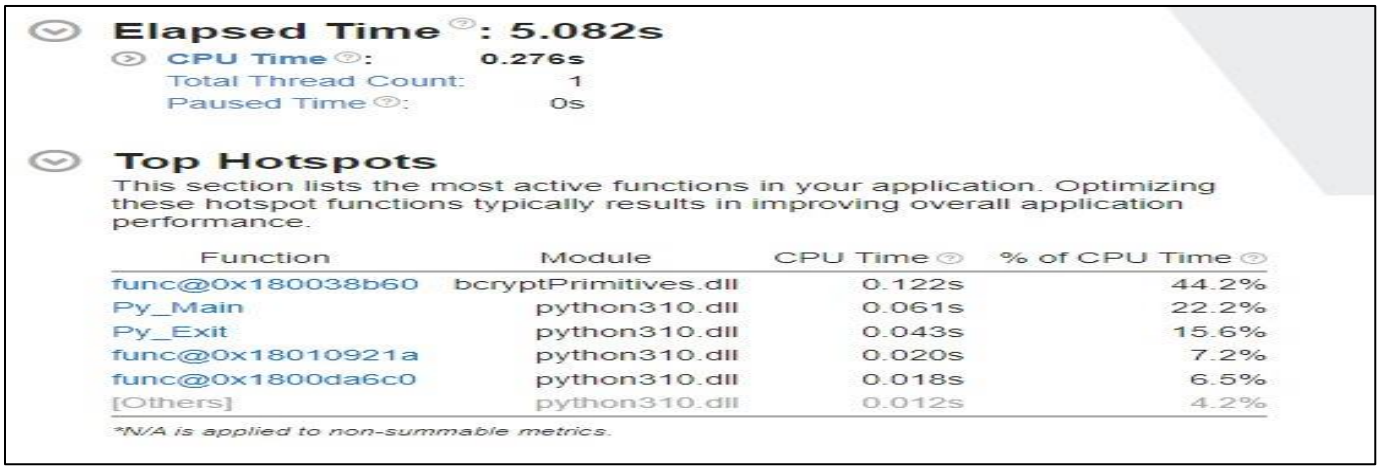
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

Fig 6: Server Side

VI. RESULTS AND DISCUSSIONS

A. *Statistical Analysis*

AES and RSA are implemented on Intel X-64 based Processor i5-8250 U with speed 1.60 to 1.80 GHz with 8 GB in environment of 64-bit operating System. Implement and analyze the time complexity of various modes of AES algorithm on 5 Notepad text file of size 500kb, 1 MB, 2 MB, 3 MB,4 MB, and 5 MB respectively, over average of 100 iteration.



Elapsed Time: 5.082s

CPU Time: 0.276s
Total Thread Count: 1
Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time	% of CPU Time
func@0x180038b60	bcryptPrimitives.dll	0.122s	44.2%
Py_Main	python310.dll	0.061s	22.2%
Py_Exit	python310.dll	0.043s	15.6%
func@0x18010921a	python310.dll	0.020s	7.2%
func@0x1800da6c0	python310.dll	0.018s	6.5%
[Others]	python310.dll	0.012s	4.2%

*N/A is applied to non-summable metrics.

Fig 7: Performance of Hybrid Algorithm

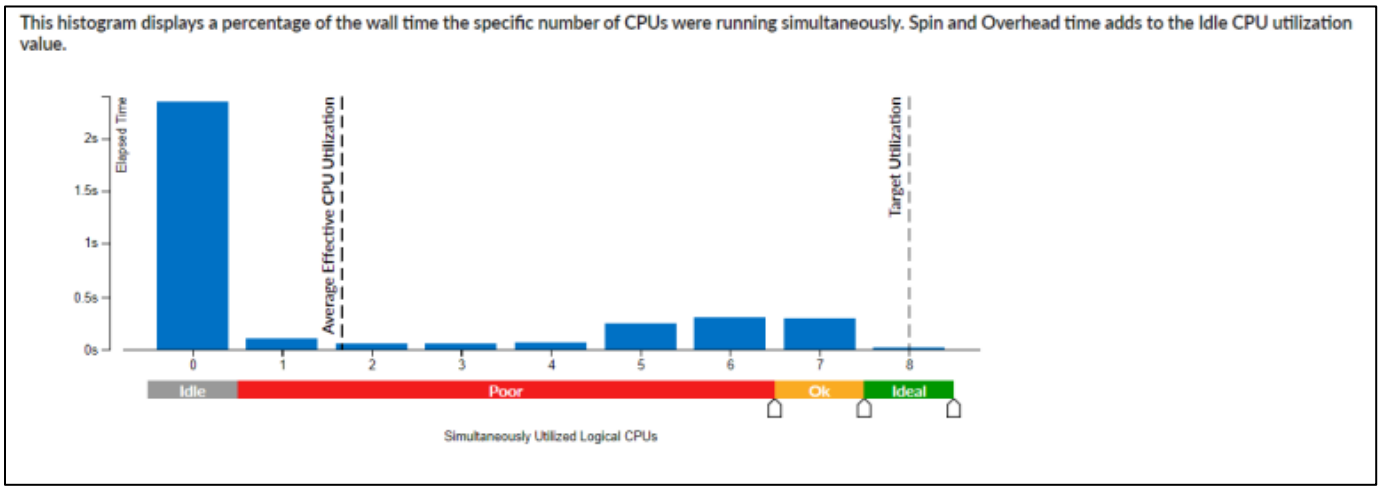


Fig 8: CPU Utilization Plot for Hybrid Algorithm

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

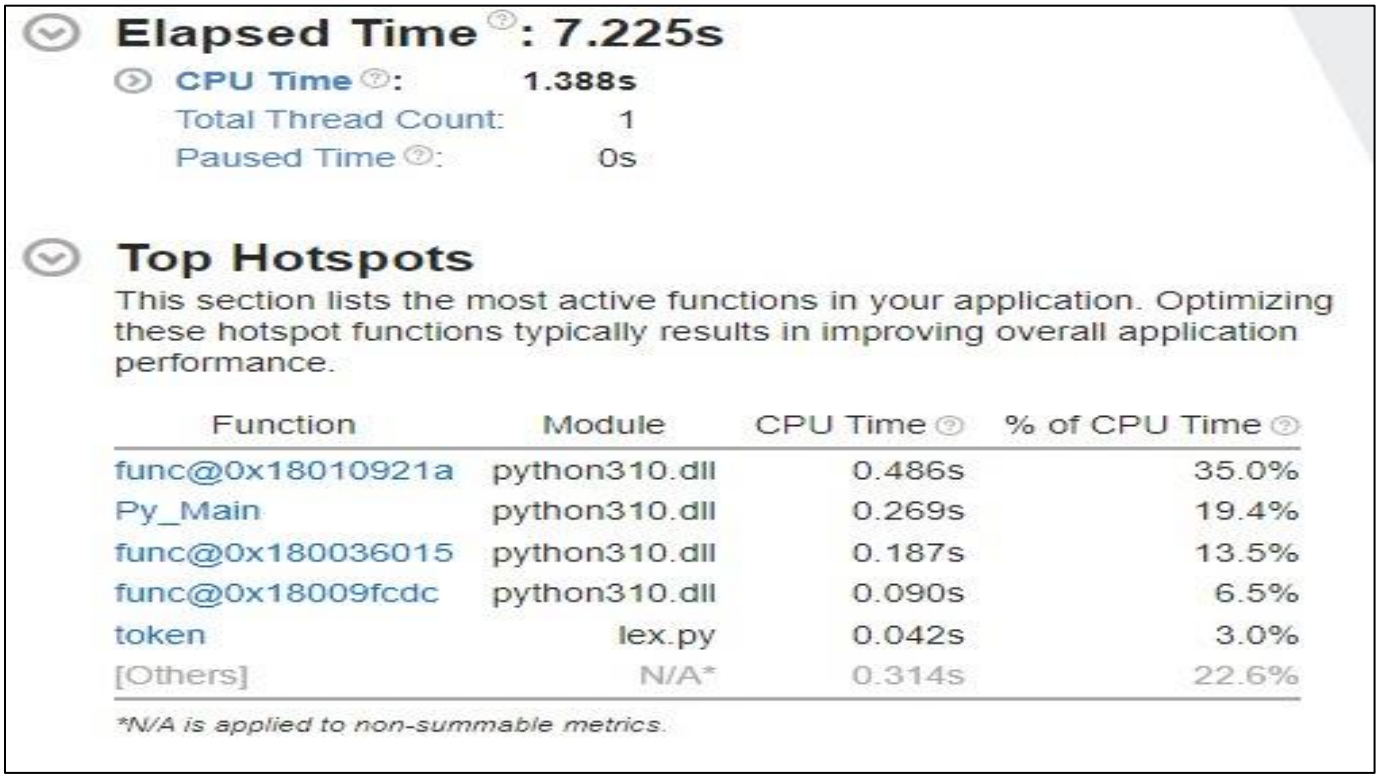


Fig 9: Performance of RSA Algorithm

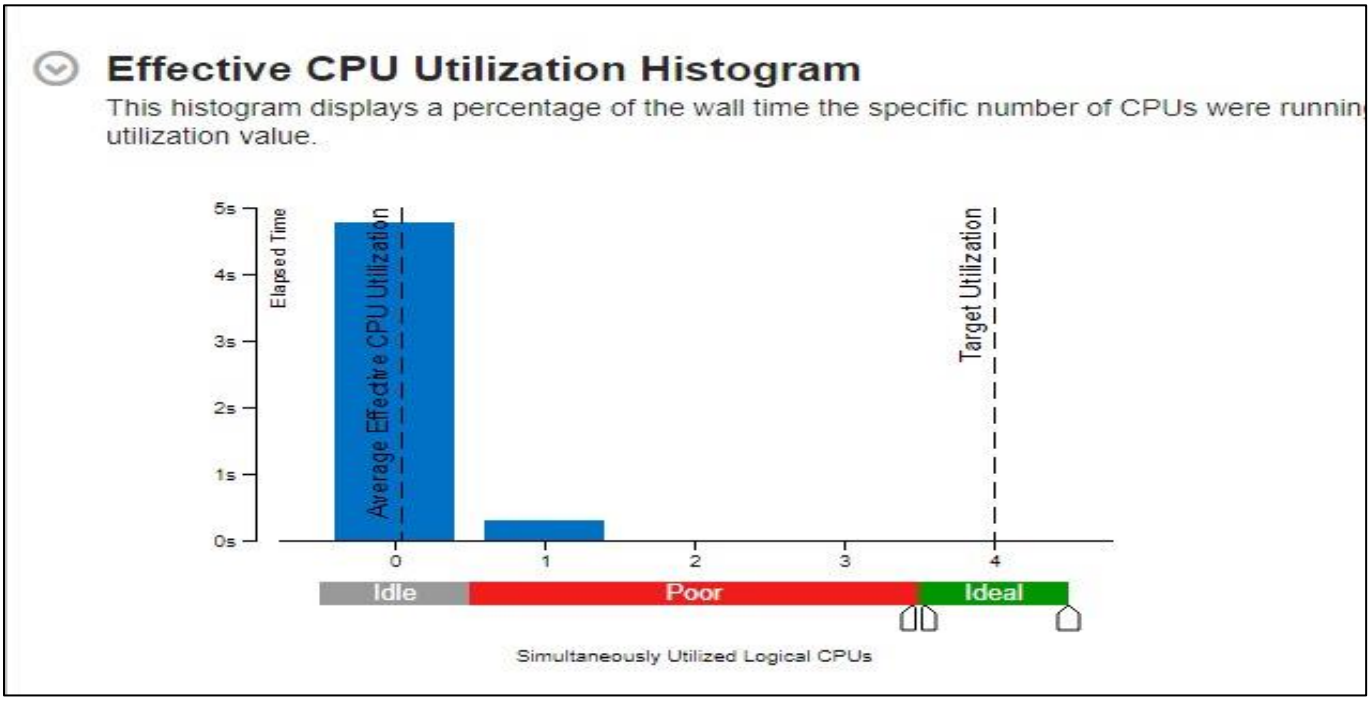


Fig 10: CPU Utilization Plot for RSA Algorithm

Table 1: Encryption Time in ms

File Size	AES	RSA	Hybrid
500 KB	63.15	3678.98	59.53
1 MB	129.25	7340.05	83.50
3 MB	196.34	29123.20	175.67
5 MB	296.24	48538.67	287.78

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

B. Comparison of Algorithms

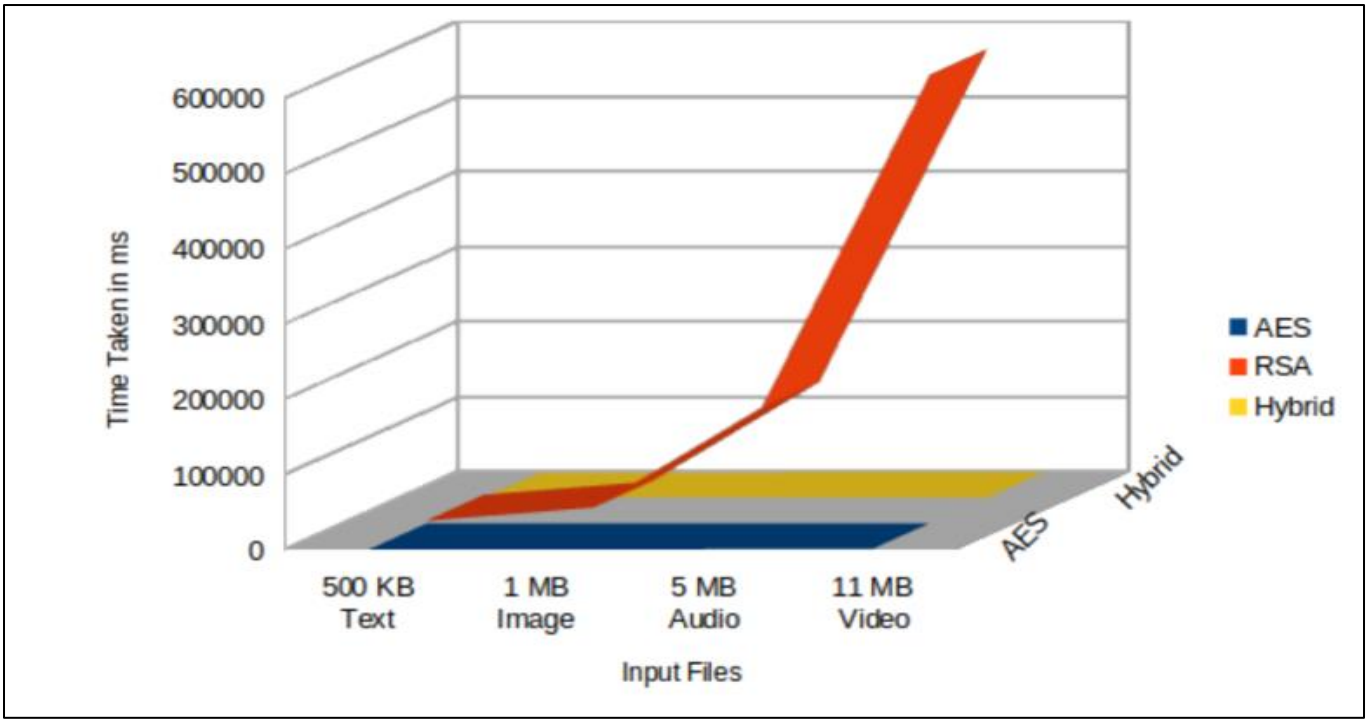


Fig 11: Comparison of Encryption Time of AES, RSA and HybridAlgorithm

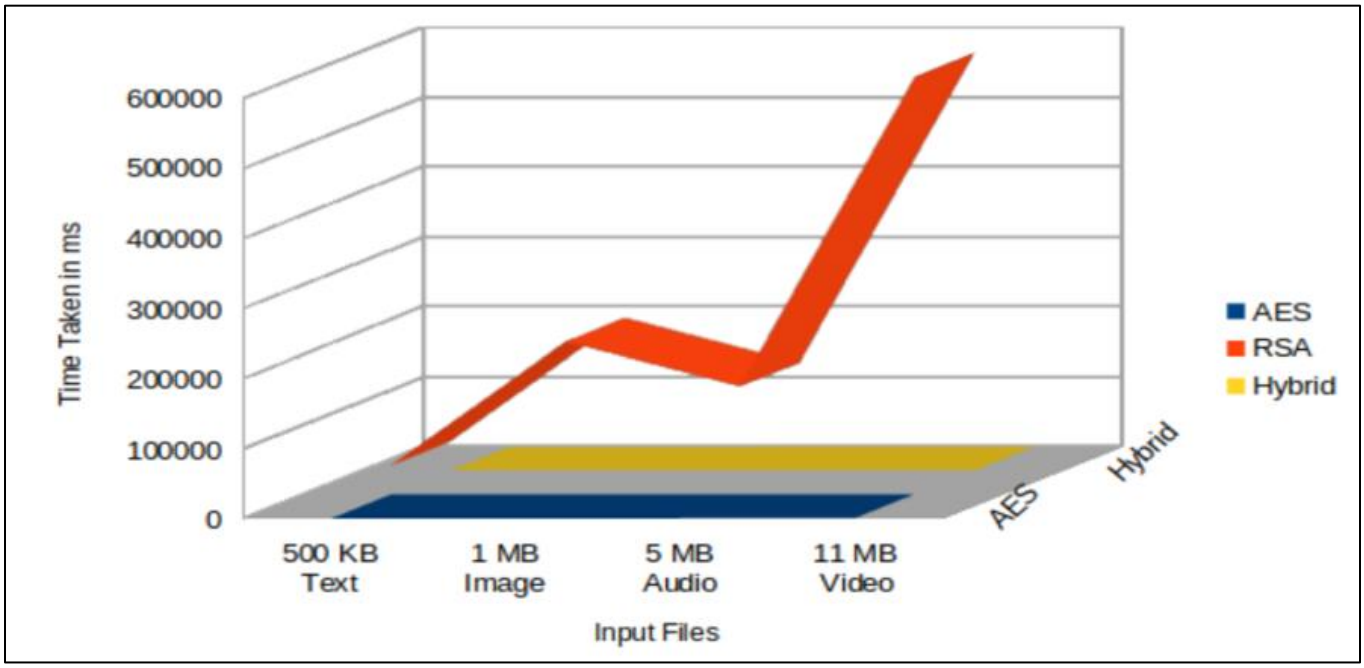


Fig 12: Comparison of Decryption Time of AES, RSA and HybridAlgorithm

Based on the observations from Fig. 11 and Fig. 12, it can be inferred that the AES-RSA hybrid algorithm combines the strengths of both encryption methods. The hybrid approach demonstrates encryption and decryption times that are nearly equivalent to those of AES, while also exhibiting an avalanche effect property comparable to that of RSA. Consequently, the hybrid algorithm not only offers superior speed but also enhances security, outperforming the individual AES and RSA algorithms in both aspects. Thus, as evident from the earlier images (7, 8) in terms of execution time our proposed hybrid algorithm is able to perform better by around 4s and when we come to the Effective CPU utilization Histogram which shows us various things in terms of Spin and Overhead time which determines the efficiency of the code and its active functions. So, in terms of efficiency our proposed Algorithm is able to perform better.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

C. Performance in Terms of Security

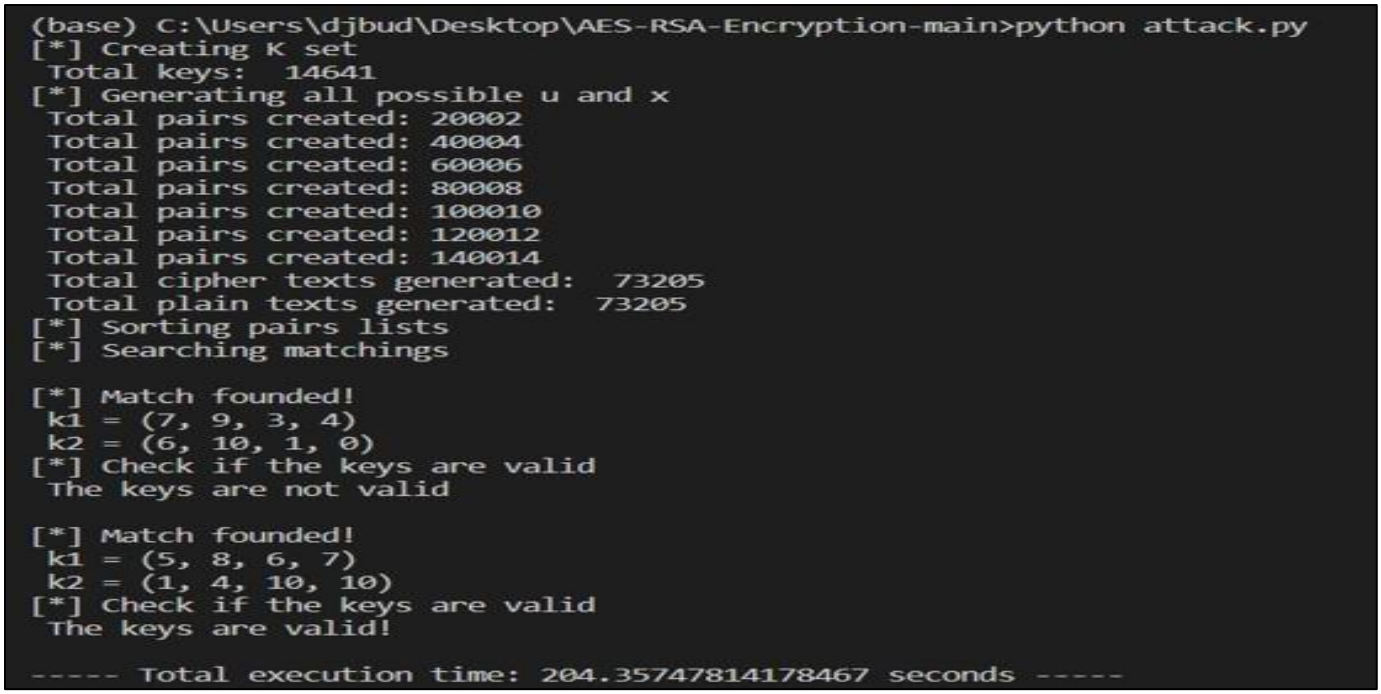


Fig 13: Performance in Terms of Security

Table 2: Total Time for Each Algorithm in Seconds

Algorithm	Time Taken to Crack the Cipher Text
Proposed	226.234s
RSA	204.357s
AES	186.672s

Basically, we are trying to perform an attack to check the security level of our proposed algorithm as against the existing algorithms. So, we have the dataset consisting of Plain text and Cipher text. We give the same plain text to each of the algorithm and then also store their cipher text. Now, using our approach we pass on the plain text to the randomized algorithm which is generating and check various combinations to find out the cipher text and trying to find a match. So, we are calculating the time taken to find the potential match and thus if it takes a longer amount of time to crack the cipher text that means that particular algorithm is better in terms of security as compared to other algorithms and if it takes a small amount of time to crack the cipher text then it is less secure as compared to others.

As evident from the above table the time taken to crack the cipher text is more thus indicating the given algorithm is moresecure as compared to other existing methods.

D. Tools Used

We have used different tools and software's for code implementation and analyzing the performance of the proposed algorithm and the normal existing algorithm.

- Code implementation: Visual Studio Code
- Performance Assessment: Intel Vtune Profiler

VII. CONCLUSION AND FUTURE SCOPE

Thus, we can conclude that the algorithm proposed here is better in terms of various aspects such as execution time, security and many more statistical metrics. So, we are able to provide a better solution to secure the client-server communication channel and focused our metrics towards the application of connected car ecosystem as well. So, we act better in terms of security by using our proposed Hybrid algorithm and we are also able to overcome the disadvantages of the existing RSA and AES algorithm. So, we are able to provide a client-server application with confidential message exchange to provide authentication, integrity and key-sharing among both the client and the server.

Future research on actual quantum computers will be necessary to enhance this concept. Testing and creating new approaches are always a good idea since encryption algorithmsare always evolving. There are several suggestions that mightbe implemented to help this thesis, such as creating a metricsmodule to evaluate various encryption techniques and basingfindings on various factors.

REFERENCES

[1]. Abikoye, O.C., Haruna, A.D., Abubakar, A., Akande, N.O. and Asani, E.O., 2019. Modified advanced encryption standard algorithm for information security. *Symmetry*, *11*(12), p.1484.

[2]. Qureshi, Muhammad Bilal, et al. "Encryption Techniques for Smart Systems Data Security Offloaded to the Cloud."*Symmetry* 14.4 (2022): 695.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1548

[3]. S. Al Mamun, M. A. Mahmood and M. A. Amin, "Ensuring Security of Encrypted Information by Hybrid AES and RSA Algorithm with Third-Party Confirmation," *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2021, pp. 337-343, doi: 10.1109/ICICCS51141.2021.9432174.

[4]. G. Yudheksha, P. Kumar and S. Keerthana, "A study ofAES and RSA algorithms based on GPUs," 2022 International Conference on Electronics and Renewable Systems (ICEARS), 2022, pp. 879-885, doi: 10.1109/ICEARS53579.2022.9752356.

[5]. S. J. Mohammed and D. B. Taha, "Performance Evaluation of RSA, ElGamal, and Paillier Partial Homomorphic Encryption Algorithms," 2022 International Conference on Computer Science and Software Engineering (CSASE), 2022, pp. 89-94, doi: 10.1109/CSASE51777.2022.9759825.

[6]. Kanika Sharma, Alka Agrawal, Dhirendra Pandey, R.A. Khan, Shail Kumar Dinkar, RSA based encryption approachfor preserving confidentiality of big data, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 5,

[7]. Z. Wang, S. Wu and Z. Gu, "A RSA algorithm simulation method using C language," 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), 2019, pp. 461-464, doi: 10.1109/EITCE47263.2019.9094826.