Earthline Journal of Mathematical Sciences
E-ISSN: 2581-8147
Volume 15, Number 5, 2025, Pages 779-802
https://doi.org/10.34198/ejms.15525.779802

A Novel Cryptographic Approach for Enhanced Data Security in Cloud Computing Environments Using Residue Number System and Advanced Encryption Standard

Edward Yellakuor Baagyere[1,*], Light Quashigah[2], Peter Awonnatemi Agbedemnab[3], Regina Esi Turkson[4], Gideon Evans Wenya[5] and Iven Aabaah[6]

[1] Department of Computer Science, C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana
e-mail: ebaagyere@cktutas.edu.gh

[2] Department of Computer Science, C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana

[3] Department of Information Systems and Technology, C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana

[4] Department of Computer Science and Information Technology, University of Cape Coast, Cape Coast, Ghana

[5] School of Advanced Technologies, Engineering Science (SATES), Accra Institute of Technology, Accra-North, Ghana

[6] Department of Information Systems and Technology, C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana

Abstract

The rapid growth of cloud computing has made it an important area of study in computer science. While cloud computing offers many benefits, it also faces many challenges, including information security. To address this challenge, this paper proposes and tests a new double-layered encryption method to improve data security in the cloud. The method combines the residue number system (RNS) and the advanced encryption standard (AES) to reduce hardware complexity and improve encryption efficiency. This help address the changing landscape of security threats and computational needs that are part of cloud computing. The results show that the proposed method has the features needed to make an effective encryption algorithm for cloud computing. It combines security and speed and works better than other hybrid AES cryptographic techniques. The proposed technique was comprehensively evaluated and tested using Python Programming Language version 3.9.7, executed on a 1.4 GHz Quad-Core Intel Core ™i5 macOS processor laptop.

Received: April 7, 2025; Accepted: June 5, 2025; Published: June 27, 2025
2020 Mathematics Subject Classification: Primary: 94A60; Secondary: 11A07, 68P25, 68M14.
Keywords and phrases: cloud computing, data security, residue number system (RNS), advanced encryption standard (AES), cryptography.

[*]Corresponding author

1 Introduction

In the ever-changing field of cloud computing, organizations from various backgrounds and industries are taking advantage of the cloud's capabilities for a wide range of uses. rapidly evolving realm of cloud computing technology, organizations of diverse profiles, sizes, and industries are increasingly harnessing the cloud's capabilities for an array of purposes. These applications include a wide range of uses, such as customer-centric online services, big data analytics, email infrastructure, virtual desktops, disaster recovery systems, and robust data backup solutions. According to the National Institute of Standards and Technology (NIST), cloud computing represents a fundamental shift in how computing resources are provisioned and managed. This shift enables the rapid provisioning and release of customizable and reliable computing resources [1, 2]. These resources encompass storage, servers, networks, applications, and services, all delivered over a shared network infrastructure. This transformation necessitates minimal administrative burdens on both service providers and users [1–3]. The advent of cloud computing has revolutionized the technological landscape by eliminating the need for extensive investments in costly hardware and software, offering instead a pay-as-you-go model for renting computing resources. The cloud services umbrella encompasses Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [3].

Before cloud computing, security standards were not as rigorous as they are today. Security features were built into software and installed on customer premises, behind corporate firewalls [2], [4]. With the advent of cloud computing, vendors now play a bigger role in protecting software on behalf of customers. As a result, customers have higher expectations for security and expect vendors to meet stringent security requirements [1, 4]. While some argue that cloud security can never be 100% secure [3], most agree that security must be an integral part of the system architecture, regardless [2].

The strength of data security algorithms depends on several factors, including the type and strength of encryption keys, the level of encryption used, and the complexity of the algorithms themselves. The speed of computing power doubles every 18 months, which makes conventional cryptosystems vulnerable to parallel computing systems [5, 6]. This makes it crucial to develop multi-layered cryptosystems that are resilient to such security breaches, especially with the threat of quantum computing looming on the horizon.

Encryption emerges as a pivotal tool for safeguarding data within the cloud, ensuring that data remains incomprehensible and unusable without the requisite decryption key [7]. Cloud computing encryption encompasses a diverse range of cryptographic techniques designed to protect data at rest and during transit between the cloud and users [8]. Among these techniques, the Residue Number System (RNS) and the Advanced Encryption Standard (AES) algorithm have gained prominence in securing cloud data. RNS is a numerical representation system that allows for rapid modular arithmetic operations, which are essential for many encryption algorithms [9, 10]. In contrast, the AES algorithm is a widely adopted encryption standard known for its high level of security and efficiency. The AES 256-bit encryption standard is esteemed as one of the most robust and secure encryption standards available today [11]. By synergising these two techniques, cloud data storage encryption can achieve a blend of speed, denoted by reduced time complexity, and security, surpassing other multi-layered AES cryptographic methods. This approach leverages the inherent properties of RNS combined with the robustness of the AES encryption algorithm, offering an assurance of secure cloud data storage [12].

1.1 Related Works

Previous research has utilised either RNS or AES along with various cryptographic techniques in cloud computing to enhance data security. The related works emphasize on the importance of encryption processes in addressing modern data security challenges.

The work of Ahmed et al. [13] has highlighted the security shortcomings of current cloud service implementations and called for improvements. Their work identified common data disclosure issues in cloud computing, and suggested improvements in security, privacy, authenticity, integration, speed, scalability, and reliability. They also suggested further research, such as automated service level agreements and third-party trustees, to address these security concerns. Other researchers, [14] have categorized attacks and highlighted the importance of stakeholder collaboration, proposing a cyber resilience model.

In a novel study, Thabit et al. [15] proposed a homomorphic cryptographic algorithm designed specifically for cloud computing, and demonstrated its improved security and user control over private keys.

In another study, Korda et al. [16] introduced Semantic Discovery Caching (SDC), a method for performing computations on encrypted data without decrypting it, while preserving data privacy and enabling secure computations.

In a recent study, Wen et al. [17] introduced a layered encryption model that enhances data security on commercial platforms. Vidya and Deepa [18] focused on securing data transmission in cloud. Hodowu, Korda, and Ansong [1] combined the Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) to enhance data security.

In a study conducted by Kartit and El Marraki [19], the effectiveness of applying encryption algorithms for enhancing data security in cloud storage was investigated. In 2020, Kuang YueJuan, Li Yong, and Li Ping proposed a searchable ciphertext retrieval method based on counting Bloom filter for cloud encrypted data [20].

Moreover, El Balmany et al. [21] proposed the VMITLP protocol to ensure a trusted launch process of user generic virtual machine images in public cloud Infrastructure as a Service (IaaS) platforms. Hu et al. [22] proposed a Blockchain as a Service (BaaS) system based on an intelligent scheduling and operation cloud-edge platform, which was designed and developed to provide scalable and secure blockchain-based services.

In a comprehensive study, Schinianakis and Stouraitis [23] discussed the historical origins and potential applications of RNS in digital signatures, homomorphic encryption, and emerging areas like the Internet of Things (IoT) and cloud computing. In [24], the authors combined steganography and cryptography to embed encrypted text within images in a covert manner whereas the authors in [25] developed novel symmetric cryptographic algorithms based on the Chinese Remainder Theorem. Our work is significantly different from the above reviewed works.

The primary contributions of the paper are:

(i) Proposed a hybrid RNS-AES algorithm for securing data in cloud architecture.

(ii) Simulated the proposed scheme and comprehensively analysed the computational time complexity of the proposed scheme.

(iii) Conducted a comprehensive performance analysis of the proposed scheme benchmarking against existing data security algorithms in cloud storage environments.

Table 1: Notations utilized in the paper.

Notation	Symbolism Description
AES	Advance Encryption Standard
RNS	Residue Number System
$\mathcal{P}_{txt}$	Plaintext
$\mathcal{C}_{txt}$	Ciphertext
$\mathcal{S}_{key}$	Secret Key
X_n	Ascii Values
ϕ_i	Encoded Residue
Enc.	Encryption
Dec.	Decryption
Hex.	Hexadecimal
PSN	Pseudo-random Number
CRT	Chinese Remainder Theorem
XOR	Exclusive OR
F/C	Forward Conversion
R/C	Reverse Conversion
MPF	Modified Perfect Form
ECC	Elliptic-curve cryptography
RSA	Rivest, Shamir, Adleman
DSA	Digital Signature Algorithm
NIST	National Institute of Standards and Technology
ASCII	American Standard Code for Information Interchange

Various notations and symbols used in the paper are shown in Table 1.

2 Background

Cryptography is the study of mathematical methods related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication [26]. The moduli $\{2^n - 1, 2^n, 2^n + 1\}$ form a balanced representation, where the difference between adjacent moduli is relatively small (*i.e.*, 1 or 2). The balanced property simplifies the design of modular adders and enables efficient arithmetic operations [27, 28] Hardware implementation of modular addition using moduli of the form $\{2^n - 1, 2^n, 2^n + 1\}$ can take advantage of the balanced representation and exploit properties like symmetry and carry propagation across adjacent moduli. This leads to more efficient and optimized hardware designs for modular adders [12, 27].

In performing a modulus operation on a number with respect to the modulus. For example, given the moduli set $\{m_1, m_2, ..., m_n\}$, the decimal/binary number X is represented uniquely as $x_i = |X|m_i, i = 1, 2, ..., n$. Translating from conventional notation into residue notation is called forward converting carryout by the RNS forward converter (F/C) whilst converting RNS notation back to conventional notation is done by the RNS reverse converter (R/C) [29], illustrated in Equation (1) using the CRT.

$$x = (x_{k-1}|...|x_2|x_1|x_0)_{RNS} = \left|\sum_{i=0}^{k-1} M_i \left|\alpha_i x_i\right| m_i\right|_M \tag{1}$$

where by definition,

$$M_i = \frac{M}{m_i}, \text{ and } \alpha_i = \langle M_i^{-1} \rangle$$

is the multiplicative inverse of M_i with respect to m_i. Similarly, new variants such as CRTs I-III can be used to achieve the R/C.

The AES block cipher operates on a fixed-size block of plaintext and produces corresponding blocks of ciphertext distributed by NIST in the year 2000. The main reason for this technique is to supplant the DES encryption after realizing some few security gaps. Several rounds (10 rounds - 128 bits), (12 rounds - 192 bits), and (14 rounds - 256 bits) of transformation, which include substitution, permutation, and diffusion operations on the plaintext block, utilizing a shared secret key for both the ciphering and deciphering processes [3, 30, 31].

3 Proposed Technique

We developed the proposed encryption algorithm through a combination of strategies, employing the Advanced Encryption Standard algorithm in conjunction with the Residue Number System representation's moduli set, specifically $\{2^n - 1, 2^n, 2^n + 1\}$:

(i) **RNS Encoding**: The initial step in encryption process involves encoding the data using the RNS algorithm. This encoding technique facilitated the transformation of large numerical values into smaller ones, resulting in faster processing times.

(ii) **AES Encryption**: Following the RNS encoding, the data was subjected to an additional layer of security through encryption using the AES algorithm. This ensured that the data remained confidential and protected against unauthorized access.

(iii) **Cloud Storage**: The encrypted data was subsequently stored in cloud-based repositories. This secure storage method allowed for efficient data management and remote accessibility while maintaining the data's confidentiality.

(iv) **Data Retrieval**: When data retrieval was necessary, a reverse process was employed. Initially, the AES decryption procedure was executed to recover the original RNS-encoded data.

(v) **RNS Decryption**: Subsequently, the R/C process was applied to decode the data back into its original form, ready for utilization.

Through the combination of RNS and the AES technique in this approach, we proposed a text encryption and decryption algorithm that guarantees strong security measures and faster encryption to streamline data management in the cloud environment.

3.1 Algorithm for the Proposed Technique

The algorithm for the proposed encryption technique is presented in Algorithm 1. Referred to as the RNS-AES Encryption, this encryption system takes ($\mathcal{P}_{txt}$) as input and computes the F/C using RNS and produces a cipher as the second input for the AES encryption denoted as ($\mathcal{C}_{txt}$) in 32-bits Hex. Secret keys, PSN and ($\mathcal{S}_{key}$), are symmetrically used for both encryption and decryption. For the decryption process, CRT in Equation (1) is employed to revert ($\mathcal{C}_{txt}$) back to its original form as, ($\mathcal{P}_{txt}$). A flowchart in Figure 1 and Figure 2 provide a visual representation of these processes, elaborating on the various stages of the proposed algorithm. Algorithm 2 shown in Appendix B outlines the decryption process.

Table 2: Computational Requirements

S/N	RNS	AES
1	Ascii - Decimal Conversion	Substitution
2	Forward Conversion (F/C)	Permutation
3	Exclusively-OR operation (XOR)	Linear Transformation
4	Reverse Conversion (R/C)	Transformation and Key expansion

As shown in Table 2, the proposed technique uses forward conversion and an exclusively-OR operation at the RNS level as its mathematical concepts whilst AES uses substitution, permutation, linear transformation and Key expansion (bit-wise XOR, substitution and permutation) as its mathematical concepts. The RNS and AES cryptographic properties contribute to the robustness and security of the developed techniques.

Algorithm 1: *RNS-AES Encryption*

```
// Encryption Process for RNS
Input  : plaintext ($\mathcal{P}_{txt}$)
Output: obtained $X_n$, i.e., ascii_values of corresponding $\mathcal{P}_{txt}$
Input  : $X_n, m_i$, $M$, $n \ni n \in M; M$ is the dynamic range.
Output: $x_i = X \mod m_i$: $\phi_i = [x_1, x_2, \ldots, x_N]$
GenPSN [100-999], a pseudo-Random Number generator;
Compute $\phi_i = x_i \oplus PSN$;
Cipher ($\mathcal{C}_{txt}$) = [$0x00_1$, $0x00_2$, $\ldots$, $0x00_{16}$] // the hex representation of $\phi_i$
// AES Encryption Process starts here:
Input  : Cipher ($\mathcal{C}_{txt}$), Secret_key ($\mathcal{S}_{key}$)
Output: AES_state
state = Init State ($\mathcal{C}_{txt}, \mathcal{S}_{key}$);
Input  : AddKey(state, $\mathcal{S}_{key_0}$)
for $i = 1$ to $n_r - 1$ do
    $SubBytes(state)$;
    $ShiftRows(state)$;
    $MixColumns(state)$;
    $AddKey(state, key_i)$;
$SubBytes(state)$;
$ShiftRows(state)$;
$AddKey(state, key_{n_r-1})$;
Output: Ciphertext: = ($\mathcal{C}_{txt}$)
```

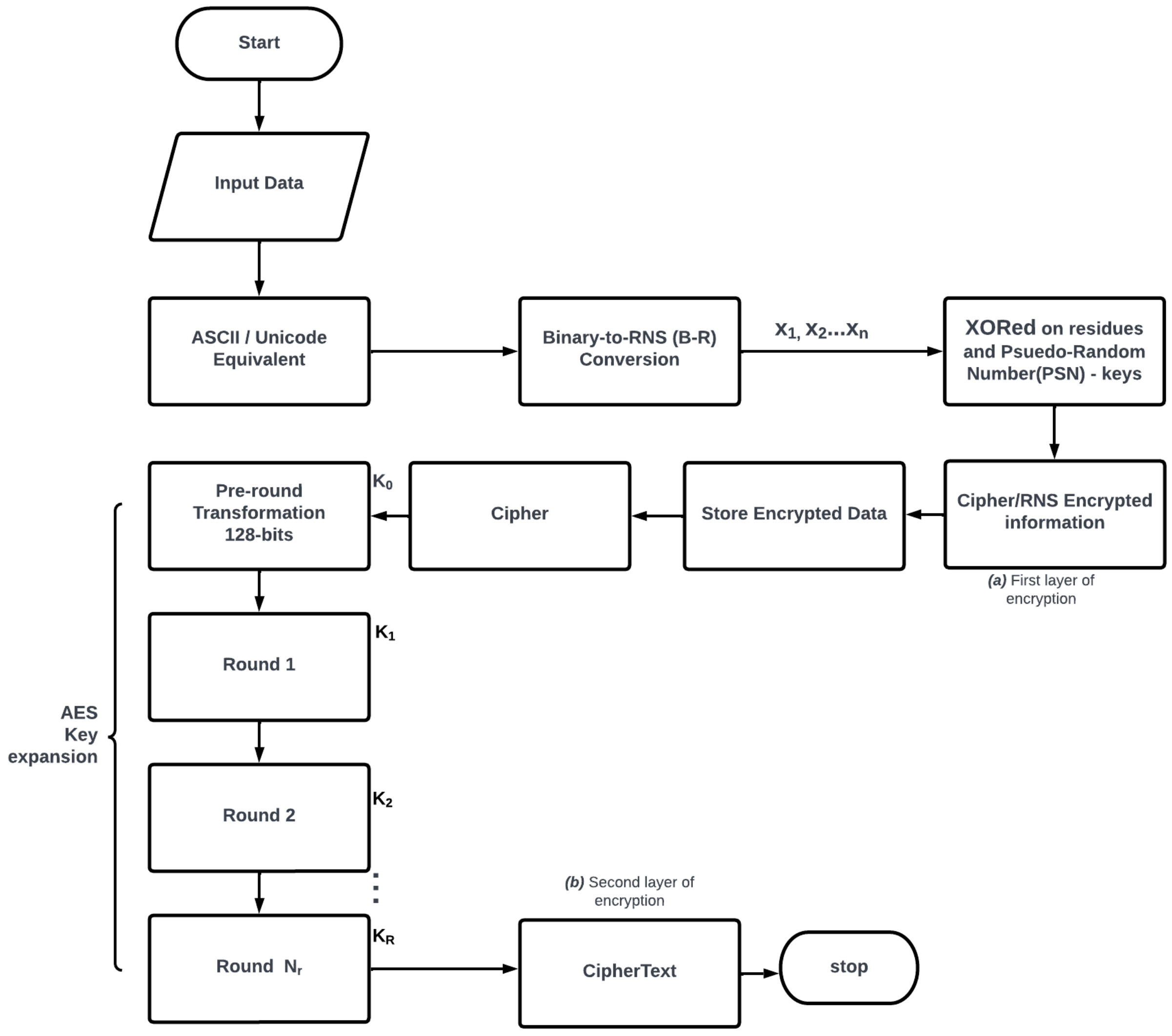

Start
Input Data
ASCII / Unicode Equivalent
Binary-to-RNS (B-R) Conversion
$x_1, x_2 \ldots x_n$
XORed on residues and Psuedo-Random Number(PSN) - keys
Cipher/RNS Encrypted information
(a) First layer of encryption
Store Encrypted Data
Cipher
Pre-round Transformation 128-bits
K_0
Round 1
K_1
Round 2
K_2
Round N_r
K_R
AES Key expansion
(b) Second layer of encryption
CipherText
stop

Figure 1: Flowchart for the Proposed Encryption Algorithm.

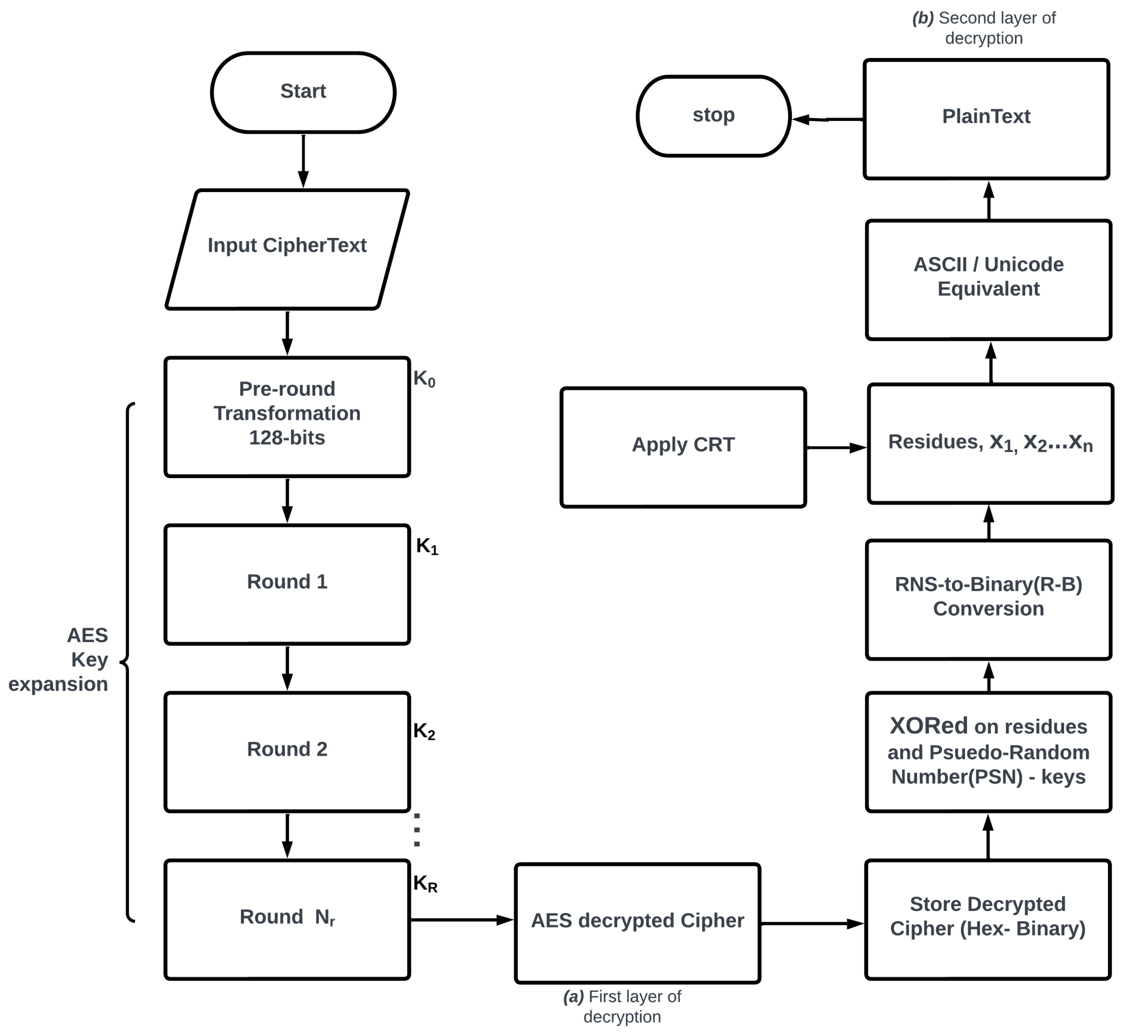

(b) Second layer of decryption
Start
stop
PlainText
Input CipherText
ASCII / Unicode Equivalent
Pre-round Transformation 128-bits
K0
Apply CRT
Residues, X1, X2...Xn
Round 1
K1
RNS-to-Binary(R-B) Conversion
AES Key expansion
Round 2
K2
XORed on residues and Psuedo-Random Number(PSN) - keys
Round Nr
KR
AES decrypted Cipher
Store Decrypted Cipher (Hex- Binary)
(a) First layer of decryption

Figure 2: Flowchart for the Proposed Decryption Algorithm.

3.2 Numerical Evaluation of the Proposed Technique

3.2.1 RNS Encryption (*first-layer of encryption*)

Encrypting **'picy'** with moduli set $\{2^n - 1, 2^n, 2^n + 1\}$.

(i) Express characters in ASCII/Unicode values (X_n) as a set of residues.
ASCII/Unicode equivalent values $\{X_1, \ldots X_n\}$.

$$\text{p - } \{X_1\} = \{112\}$$

$$\text{i - }\{X_2\} = \{105\}$$
$$\text{c - }\{X_3\} = \{099\}$$
$$\text{y - }\{X_4\} = \{121\}$$

(ii) With the moduli set, for $n = 3$ yields.

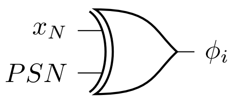
$$\{m_1, m_2, m_3\} = \{7,\ 8,\ 9\}$$

(iii) Compute, $X_n \equiv x_N(\mod m)$.

$$|112|_{7|8|9} = \langle 0, 0, 4\rangle = (00, 000, 100)$$
$$|105|_{7|8|9} = \langle 0, 1, 6\rangle = (00, 001, 110)$$
$$|099|_{7|8|9} = \langle 1, 3, 0\rangle = (01, 011, 000)$$
$$|121|_{7|8|9} = \langle 2, 1, 4\rangle = (10, 001, 100)$$

(iv) Encoding residues $\langle x_1, \ldots, x_4\rangle$ with XOR operation. $\therefore$ residues $\langle x_1, \ldots, x_4\rangle$ of the plaintext **'picy'** are: (00000100), (00001110), (01011000), (10001100).

(v) Compute,



$$00000100 \oplus 01001100 = 01001000$$
$$00001110 \oplus 01001100 = 01000010$$
$$01011000 \oplus 01001100 = 00010100$$
$$10001100 \oplus 01001100 = 11000000$$

3.2.2 AES Encryption (*second-layer of encryption*)

Input ="48 42 14 C0 00 00 00 00 00 00 00 00 00 00 00 00"
AES Cipherkey = "2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C"

(i) AES Pre-round Transformation (Start of Round)

$$\begin{bmatrix} 48 & 00 & 00 & 00 \\ 42 & 00 & 00 & 00 \\ 14 & 00 & 00 & 00 \\ C0 & 00 & 00 & 00 \end{bmatrix} \oplus \begin{bmatrix} 2B & 28 & AB & 09 \\ 7E & AE & F7 & CF \\ 15 & D2 & 15 & 4F \\ 16 & A6 & 88 & 3C \end{bmatrix} = \begin{bmatrix} 63 & 28 & AB & 09 \\ 3C & AE & F7 & CF \\ 01 & D2 & 15 & 4F \\ D6 & A6 & 88 & 3C \end{bmatrix}$$

(ii) AES SubBytes (Substitution Box), and ShiftRows

$$\begin{array}{|c|c|c|c|} \hline 63 & 28 & AB & 09 \\ \hline 3C & AE & F7 & CF \\ \hline 01 & D2 & 15 & 4F \\ \hline D6 & A6 & 88 & 3C \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|} \hline FB & 34 & 62 & 01 \\ \hline EB & E4 & 68 & 8A \\ \hline 7C & B5 & 59 & 84 \\ \hline F6 & 24 & C4 & EB \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|} \hline FB & 34 & 62 & 01 \\ \hline E4 & 68 & 8A & EB \\ \hline 59 & 84 & 7C & B5 \\ \hline EB & F6 & 24 & C4 \\ \hline \end{array}$$

(iii) MixColumns

$$\begin{array}{|c|c|c|c|} \hline \mathbf{02} & 03 & 01 & 01 \\ \hline 01 & 02 & 03 & 01 \\ \hline 01 & 01 & 02 & 03 \\ \hline 03 & 01 & 01 & 02 \\ \hline \end{array} \otimes \begin{array}{|c|c|c|c|} \hline \mathbf{FB} & 34 & 62 & 01 \\ \hline E4 & 68 & 8A & EB \\ \hline 59 & 84 & 7C & B5 \\ \hline EB & F6 & 24 & C4 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline \mathbf{68} & A2 & 19 & 55 \\ \hline 28 & 85 & CD & CC \\ \hline 8B & 4E & 7C & CC \\ \hline 66 & 47 & 18 & CE \\ \hline \end{array}$$

Utilizing the Galois Field polynomial theorem,

$$\begin{cases} 02 = 00000010 = X \\ FB = 11111011 = X^7 + X^6 + X^5 + X^4 + X^3 + X + 1 \end{cases}$$

$$(X) \cdot (X^7 + X^6 + X^5 + X^4 + X^3 + X + 1)$$
$$(X^8 + X^7 + X^6 + X^5 + X^4 + X^2 + X)$$

Using the irreducible polynomial theorem,

$$GF(X^8) = X^4 + X^3 + X + 1$$

$$\cancel{X^4} + X^3 + \cancel{X} + 1 + (X^7 + X^6 + X^5 + \cancel{X^4} + X^3 + \cancel{X} + 1)\ X^7 + X^6 + X^5 + X^3 + X + 1 = \mathbf{11101101}$$
$$11101101 \oplus 00110111 \oplus 01011001 \oplus 11101011 = \underline{\underline{\mathbf{01101000} = \mathbf{(68)}}}$$

(iv) AES Key Schedule (from $K_0 - K_1 ... K_r$)

$$\begin{array}{|c|} \hline 09 \\ \hline C7 \\ \hline 4F \\ \hline 3C \\ \hline \end{array} \rightarrow \text{RotWord} \begin{array}{|c|} \hline C7 \\ \hline 4F \\ \hline 3C \\ \hline 09 \\ \hline \end{array} \rightarrow \text{SubBytes} \begin{array}{|c|} \hline 8A \\ \hline 84 \\ \hline EB \\ \hline 01 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline 2B \\ \hline 7E \\ \hline 15 \\ \hline 16 \\ \hline \end{array} = \begin{array}{|c|} \hline A1 \\ \hline FA \\ \hline FE \\ \hline 17 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline 01 \\ \hline 00 \\ \hline 00 \\ \hline 00 \\ \hline \end{array} = \begin{array}{||c|} \hline A0 \\ \hline FA \\ \hline FE \\ \hline 17 \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline A0 \\ \hline FA \\ \hline FE \\ \hline 17 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline 28 \\ \hline AE \\ \hline D2 \\ \hline A6 \\ \hline \end{array} = \begin{array}{||c|} \hline 88 \\ \hline 54 \\ \hline 2C \\ \hline B1 \\ \hline \end{array} \qquad \begin{array}{|c|} \hline 88 \\ \hline 54 \\ \hline 2C \\ \hline B1 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline AB \\ \hline F7 \\ \hline 15 \\ \hline 88 \\ \hline \end{array} = \begin{array}{||c|} \hline 23 \\ \hline A3 \\ \hline 39 \\ \hline 39 \\ \hline \end{array} \qquad \begin{array}{|c|} \hline 23 \\ \hline A3 \\ \hline 39 \\ \hline 39 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline 09 \\ \hline CF \\ \hline 4F \\ \hline 3C \\ \hline \end{array} = \begin{array}{||c|} \hline 2A \\ \hline 6C \\ \hline 76 \\ \hline 05 \\ \hline \end{array}$$

Thus, the first round key is:

$$Round(K_1) \longrightarrow \begin{array}{||c|c|c|c|} \hline A0 & 88 & 23 & 2A \\ \hline FA & 54 & A3 & 6C \\ \hline FE & 2C & 39 & 76 \\ \hline 17 & B1 & 39 & 05 \\ \hline \end{array}$$

$\therefore$ ciphertext ($\mathcal{C}_{txt}$): ec2f893365b9ebccf8344de984bb42a4

As indicated in Table 3, the AES ($\mathcal{C}_{txt}$) is decrypted initially, yielding "48 42 14 C0 00 00 00 00 00 00 00 00 00 00 00 00". This output is subsequently split into its individual components, converted into binary, and decoded using the stored PSN. The results are then represented as residues (x_n) in decimal before applying CRT.

Table 3: Results of Decrypted ciphertext using CRT

Residues	**moduli**	**M$_n$**	**Inverse**	**Dynamic range**
$x_1 = (0, 0, 4)$ $x_2 = (0, 1, 6)$ $x_3 = (1, 3, 0)$ $x_4 = (2, 1, 4)$	$m_1 = 7$ $m_2 = 8$ $m_3 = 9$	$M_1 = 72$ $M_2 = 63$ $M_3 = 56$	${M_1}^{-1} = 4$ ${M_2}^{-1} = 7$ ${M_3}^{-1} = 5$	M $=$ 504

$$\begin{gathered}
\text{for } \langle 0,0,4\rangle : X_1 = |0\cdot 72\cdot 4 + 0\cdot 63\cdot 7 + 4\cdot 56\cdot 5|_{504}\\
X_1 = |1120|_{504}\\
X_1 = \mathbf{112}
\end{gathered}$$

$$\begin{gathered}
\text{for } \langle 0,1,6\rangle : X_2 = |0\cdot 72\cdot 4 + 1\cdot 63\cdot 7 + 6\cdot 56\cdot 5|_{504}\\
X_2 = |441 + 1680|_{504}\\
X_2 = \mathbf{105}
\end{gathered}$$

$$\begin{gathered}
\text{for } \langle 1,3,0\rangle : X_3 = |1\cdot 72\cdot 4 + 3\cdot 63\cdot 7 + 0\cdot 56\cdot 5|_{504}\\
X_3 = |288 + 1323|_{504}\\
X_3 = \mathbf{99}
\end{gathered}$$

$$\begin{gathered}
\text{for } \langle 2,1,4\rangle : X_4 = |2\cdot 72\cdot 4 + 1\cdot 63\cdot 7 + 4\cdot 56\cdot 5|_{504}\\
X_4 = |576 + 441 + 1120|_{504}\\
X_4 = \mathbf{121}\\
\{X_1, X_2, X_3, X_4\} = \{112, 105, 99, 121\}
\end{gathered}$$

Convert (X_n) into ASCII/Unicode.

$\therefore \{112, 105, 99, 121\} =$ **picy** as the plaintext ($\mathcal{P}_{txt}$).

3.3 Simulation: Encryption & Decryption Implementation

The proposed technique was implemented and tested using Python Version 3.9.7 programming language on a Core i5 processor computer. The simulated result is shown in Table 4.

Table 4: Encryption and decryption results.

Char. length	**Plaintext**	**Ciphertext**	**Enc. Time(s)**	**Dec. Time(s)**
4	picy	ec2f893365b9ebcc-f8344de984bb42a4	0.0312	0.0326
6	gbcdef	430846c11a07578-c9a9645661c376476	0.0343	0.0346
8	hbsdefgh	a29b28327a46cb-5e14493f9006b273c2	0.0357	0.0364
10	wbcmefghkjklq	d174bfb5bdd55-844ac0d55b82cfaf6f4	0.0362	0.0372
15	fbceefghij12345	431cb48108fd582d-d1d2765a91d30da3	0.0372	0.0378

3.3.1 Time Complexity of the proposed Algorithm

The overall time complexity is O(n) and the worst-case time complexity is also $O(n)$. This is because the steps that have the greatest impact on the overall time complexity, such as the AddKey, SubBytes, ShiftRows, and MixColumns steps, all have a time complexity of $O(n/16)$, and there are n of these steps. Additionally, the outer loop also has a time complexity of $O(n)$. This is because the overall time complexity is dominated by the time complexity of the AddKey, SubBytes, ShiftRows, and MixColumns steps, and the worst-case time complexity. The proposed algorithm's O(n) time complexity aligns with the observed linear trend between the input size of the given characters and the RNS and AES Encryption/Decryption functions, as depicted in Figure 3.

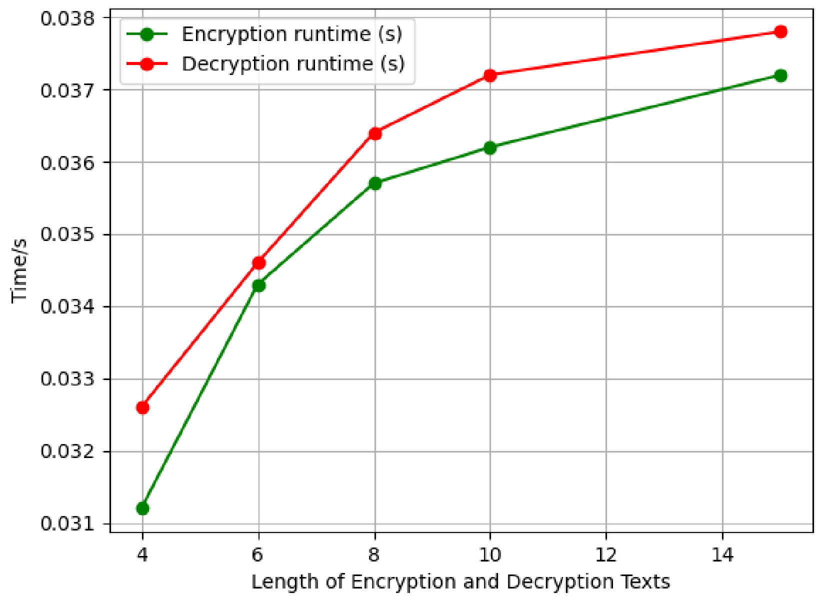


Figure 3: Input Text Size against Time.

3.3.2 Space Complexity of the Proposed Algorithm

The space complexity of the algorithm is $O(n)$, where n is the length of the input text. This is because the algorithm stores the ciphertext and the intermediate results of the AddKey, SubBytes, ShiftRows, and MixColumns steps, and each of these steps has a space complexity of $O(n/16)$. So the total space complexity is dominated by the space complexity of these steps, and is $O(n)$. Additionally, the initialization step has a space complexity of $O(1)$, but this is negligible compared to the space complexity of the AddKey, SubBytes, ShiftRows, and MixColumns. This confirms the optimal and scalable nature of the proposed algorithm in terms of it memory usage.

3.4 Performance Assessment with other Techniques

In assessing the performance of [1] and [32] techniques, an Open Secure Sockets Layer (OpenSSL) was used to conduct the assessment based on random text length shown in Table 4. AES and ECC techniques were used in the implementation of [1] whilst RSA, Digital Signature Algorithm (DSA), ElGamal, and AES algorithms were used respectively for the implementation of [32]. Figure 4 shows a bar graph of average runtime measurements of the existing techniques with the proposed technique for encryption text

Table 5: Performance assessment with other techniques.

Techniques	**Text length**	**Encryption Time(s)**	**Decryption Time(s)**
[1]	4, 8, 15	0.051, 0.058, 0.062	0.050, 0.051, 0.079
[32]	4, 8, 15	0.101, 0.168, 0.308	0.091, 0.122, 0.247
Proposed	**4, 8, 15**	**0.031, 0.036, 0.045**	**0.037, 0.057, 0.058**

of various length. Figure 5 in Appendix C shows a similar chart for the decryption process for same text of variable length. [1] and [32] show increasing average runtimes as the text length increases. This suggests that the time required for encryption increases as the complexity of the input data (text length) increases. Between the two techniques, [1] generally has slightly lower average runtimes compared to [32] across all text lengths.

As the text length increases, the average runtime for all three techniques also tends to increase. This suggests that as the input data becomes more complex, the encryption process takes more time. The proposed technique consistently demonstrates the lowest average runtimes among the three techniques for all text lengths. This indicates that the proposed technique is potentially more efficient in terms of encryption speed compared to the existing techniques.

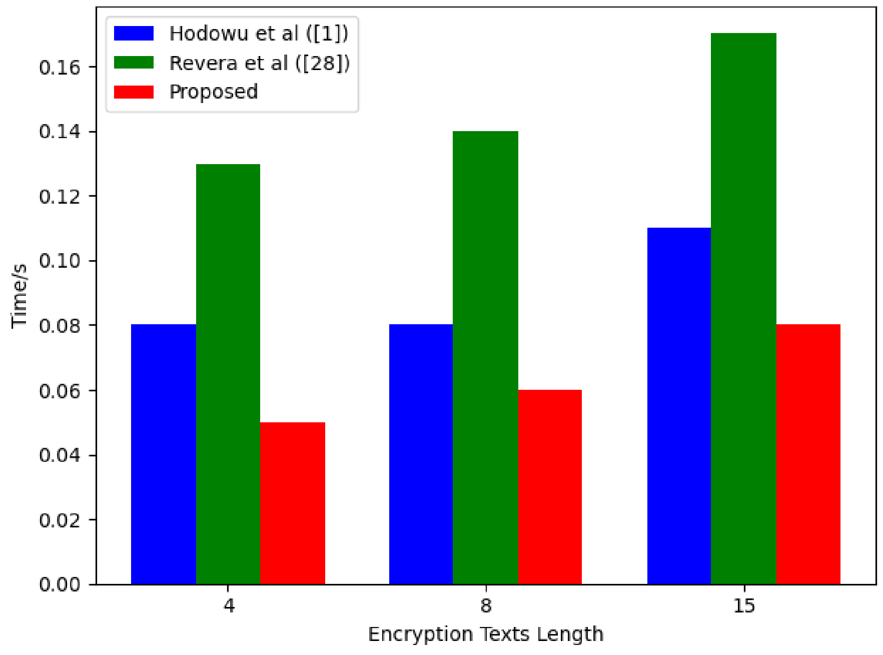


Figure 4: Encryption Runtime of Existing Techniques against Proposed Technique.

4 Conclusion

This research work developed an efficient encryption method for cloud storage by combining the benefits of both the Residue Number System (RNS) and the Advanced Encryption Standard (AES). The moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ was used to create a hybrid RNS-AES encryption that ensures secure storage of data on cloud servers without unauthorized access.

The experimental results revealed that the proposed hybrid RNS-AES technique has a number of desirable properties for developing an efficient encryption method for cloud computing. When compared to other hybrid AES techniques, the proposed method offers faster encryption and decryption times, as well as increased security, due to the combination of the RNS and AES algorithms. These properties make the hybrid RNS-AES technique a promising solution for encrypting data stored on cloud servers, as it can provide the necessary level of protection while maintaining a high level of efficiency.

For future work, the proposed hybrid RNS-AES encryption scheme has the potential to improve the security of cloud storage systems, thus should be implemented to encrypt data stored on cloud servers. The technique could also be extended to encrypt a wider range of data types such as images and other multimedia files, providing additional layers of security. Additionally, the hybrid scheme could be further tested against various cryptographic attacks to ensure its resilience and robustness. This would help to strengthen the security of cloud storage systems and make them more reliable for sensitive data storage.

APPENDICES

APPENDIX A: Encryption and Decryption Transformations of the Cryptosystem

For all $X_i \in M \in Z^+$ choosen $n \in Z^+ \ni M = \prod_{i=1}^{N} m_i$ is large enough to contain X_i with a key space $\mathcal{K} = \{m_i, PSN, \mathcal{S}_{key}\}$ as follows:

$\mathcal{E}_1(X) : Plaintext_{ASCII} \longrightarrow Decimal_{value}$
$\mathcal{E}_2(X) : X \equiv x_i (\mod m_i)$
$\mathcal{E}_3(X) : \phi_i = x_i \oplus PSN$
$\mathcal{E}_4(X) : B - Hex\ Conversion, \phi_i$
$\mathcal{E}_5(X) : \phi_i \longrightarrow AES_{\mathcal{S}_{key}}$

$\mathcal{D}_1(X) : AES_{\mathcal{S}_{key}} \longrightarrow \phi_i$
$\mathcal{D}_2(X) : Hex - B\ Conversion, \phi_i$
$\mathcal{D}_3(X) : x_i = \phi_i \oplus PSN$
$\mathcal{D}_4(X) : m_i, CRT - X$
$\mathcal{D}_5(X) : R_{Decimal} \rightarrow Plaintext_{ASCII}$

APPENDIX B: The RNS-AES Decryption Technique

Algorithm 2: *RNS-AES Decryption Technique*

```
// AES Decryption Process
Input  : Ciphertext ($\mathcal{C}_{txt}$), Secret_key ($\mathcal{S}_{key}$)
Output: AES_state
state = Init State ($\mathcal{C}_{txt}$, $\mathcal{S}_{key}$) // start off AES decryption
Input  : $AddKey(state, \mathcal{S}_{key_0})$
// AES Key Schedule
for  $i = 1$ to $n_r - 1$ do
    $SubBytes(state)$;
    $ShiftRows(state)$;
    $Inv\_MixColumns(state)$;
    $AddKey(state, key_i)$;
$Inv\_SubBytes(state)$;
$Inv\_ShiftRows(state)$;
// Skip MixColums at the lastRound
$AddKey(state, key_{n_r-1})$;
$AES\_Cipher$ ($\mathcal{C}_{txt}$) = [0x00$_1$, 0x00$_2$, ....0x00$_{16}$] // De-concatenate block and convert to
   Decimal
// Decryption Process for RNS
GenPSN [100-999]: PNS // Retrieved Pseudo-Random Number
Compute $x_N = \phi_i \oplus$ PSN ( // Decode residues
)
Input  : Compute CRT, inverse = $\left|M_n \cdot M_n^{-1}\right|_{m_n} = 1$
Output: $X_n = [x_N + M_n + M_n^{-1}]\%\ M$
Input  :  $ascii\_values$ ($X_n$)
$ascii\_values$ = [($\mathcal{P}_{txt}$)] // Convert ASCII values to plaintext
Output: plaintext($\mathcal{P}_{txt}$)
```

APPENDIX C: Descryption Runtime of Existing Techniques against the Proposed Technique

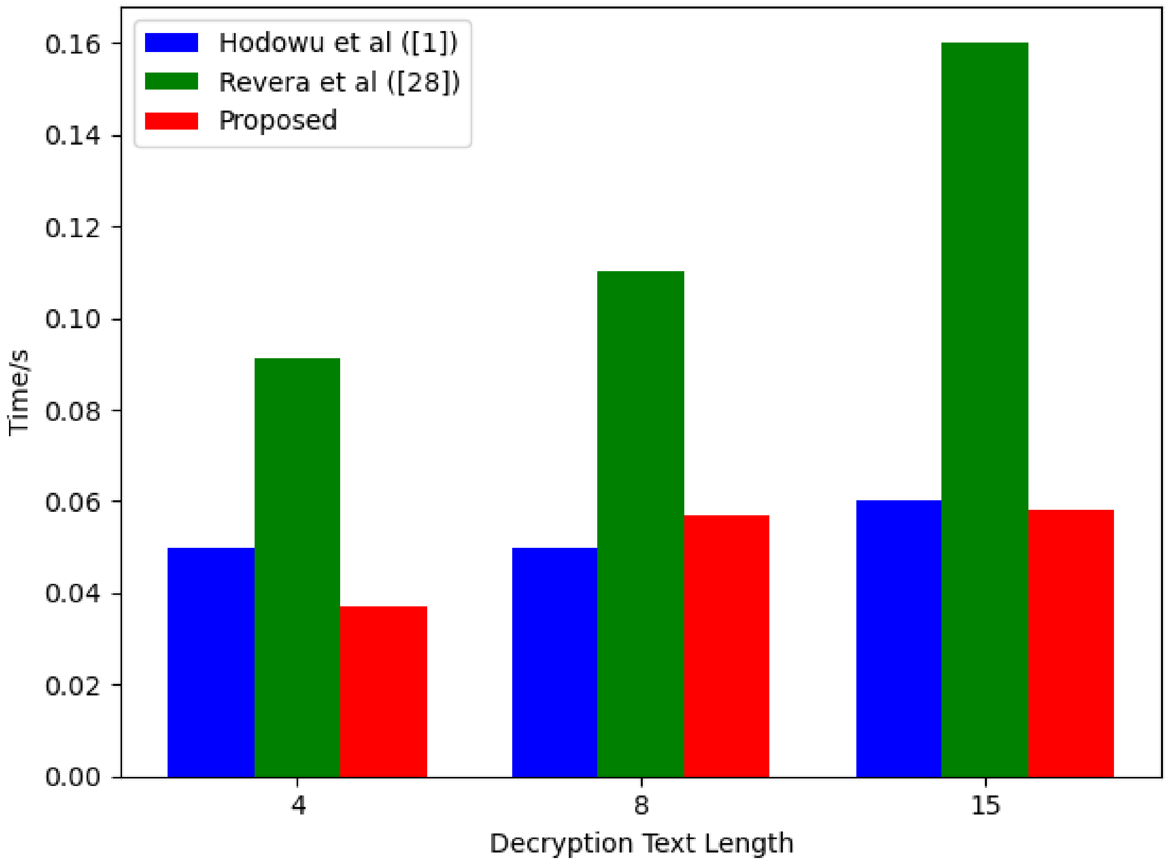
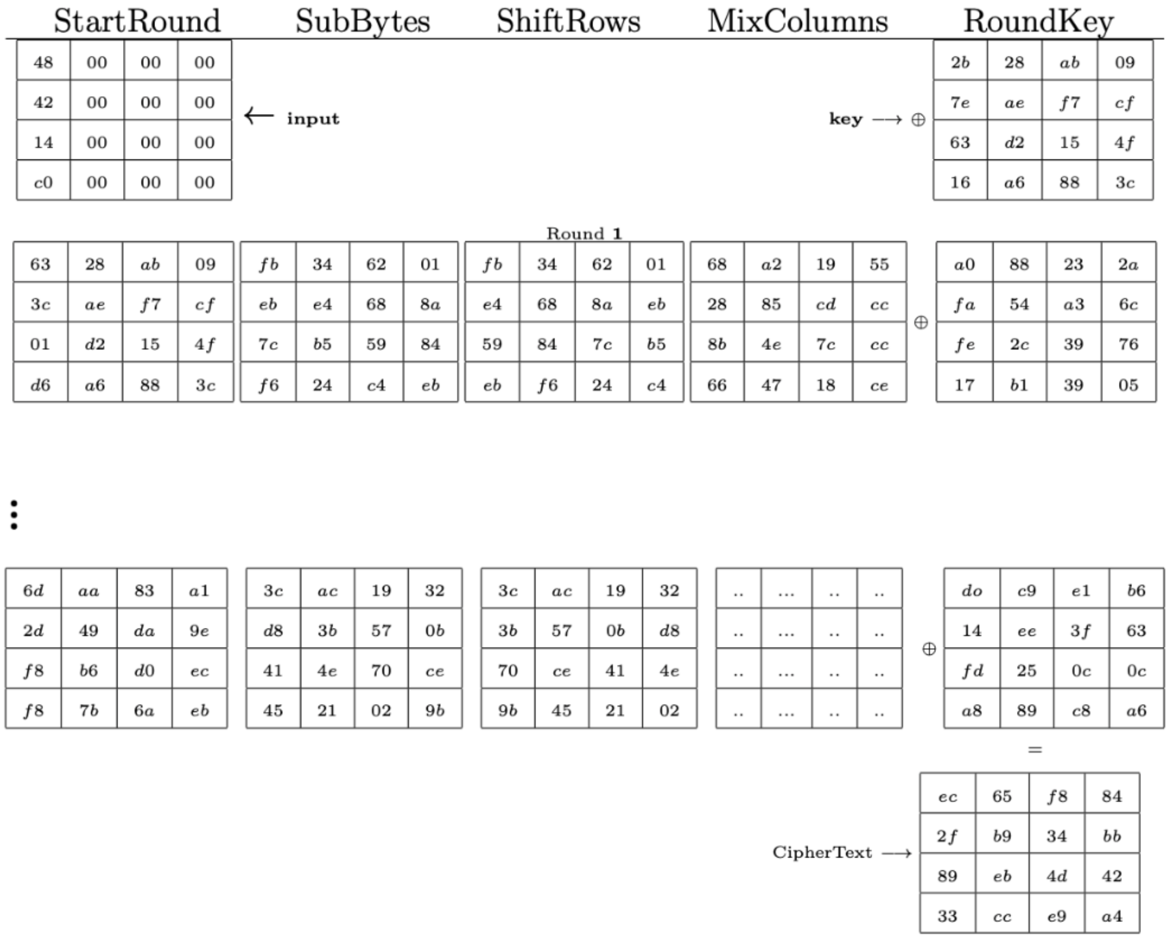


Figure 5: Decryption Runtime of Existing Techniques against Proposed Technique.

APPENDIX D: Summary of AES Encryption



StartRound (input)

48	00	00	00
42	00	00	00
14	00	00	00
c0	00	00	00

RoundKey (key)

2b	28	ab	09
7e	ae	f7	cf
63	d2	15	4f
16	a6	88	3c

Round 1

63	28	ab	09
3c	ae	f7	cf
01	d2	15	4f
d6	a6	88	3c

fb	34	62	01
eb	e4	68	8a
7c	b5	59	84
f6	24	c4	eb

fb	34	62	01
e4	68	8a	eb
59	84	7c	b5
eb	f6	24	c4

68	a2	19	55
28	85	cd	cc
8b	4e	7c	cc
66	47	18	ce

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

⋮

6d	aa	83	a1
2d	49	da	9e
f8	b6	d0	ec
f8	7b	6a	eb

3c	ac	19	32
d8	3b	57	0b
41	4e	70	ce
45	21	02	9b

3c	ac	19	32
3b	57	0b	d8
70	ce	41	4e
9b	45	21	02

..	...	..	..
..	...	..	..
..	...	..	..
..	...	..	..

do	c9	e1	b6
14	ee	3f	63
fd	25	0c	0c
a8	89	c8	a6

CipherText

ec	65	f8	84
2f	b9	34	bb
89	eb	4d	42
33	cc	e9	a4

References

[1] Hodowu, D. K. M., Korda, D. R., & Ansong, E. D. (2020). An enhancement of data security in cloud computing with an implementation of a two-level cryptographic technique, using AES and ECC algorithm. *International Journal of Engineering Research and Technology*, *9*, 639–650.

[2] Krutz, R. L., & Vines, R. D. (2010). *Cloud security: A comprehensive guide to secure cloud computing*. Wiley Publishing.

[3] Kavis, M. K. (2014). *Architecting the cloud: Design decisions for cloud computing service models*. Wiley Online Library.

[4] Bello, S. A., Oyedele, L. O., Akinade, O. O., Bilal, M., Delgado, J. M. D., Akanbi, L. A., Ajayi, A. O., & Owolabi, H. A. (2021). Cloud computing in construction industry: Use cases, benefits and challenges. *Automation in Construction*, *122*, 103441. https://doi.org/10.1016/j.autcon.2020.103441

[5] Lenstra, A. K., & Verheul, E. R. (2001). Selecting cryptographic key sizes. *Journal of Cryptology*, *14*, 255–293. https://doi.org/10.1007/s00145-001-0009-4

[6] Joseph, D., Misoczki, R., Manzano, M., Tricot, J., Pinuaga, F. D., Lacombe, O., Leichenauer, S., Hidary, J., Venables, P., & Hansen, R. (2022). Transitioning organizations to post-quantum cryptography. *Nature*, *605*(7909), 237–243. https://doi.org/10.1038/s41586-022-04623-2

[7] Singha, S., & Singha, R. (2023). Protecting data and privacy: Cloud-based solutions for intelligent transportation applications. *Scalable Computing: Practice and Experience*, *24*(3), 257–276. https://doi.org/10.12694/scpe.v24i3.2381

[8] Rezaeian, M., & Wynn, M. G. (2022). The impact of cloud computing on the IT support function: A case study from higher education. In *Handbook of Research on Digital Transformation, Industry Use Cases, and the Impact of Disruptive Technologies* (pp. 1–17). IGI Global. https://doi.org/10.4018/978-1-7998-7712-7.ch001

[9] Kadri, A. F. (2023). *Enhancement of advanced encryption standard performance on hidden data using residue number system* (Doctoral dissertation, Kwara State University, Nigeria).

[10] Eseyin, J. B., & Gbolagade, K. A. (2019). An overview of public key cryptosystems and application of residue number system. *KIU Journal of Humanities*, *4*(2), 37–44.

[11] Nissar, G., Garg, D. K., & Khan, B. U. I. (2019). Implementation of security enhancement in AES by inducting dynamicity in AES S-box. *International Journal of Innovative Technology and Exploring Engineering*, *8*(10), 1–9. https://doi.org/10.35940/ijitee.J9311.0881019

[12] Omondi, A. R., & Premkumar, A. B. (2007). *Residue number systems: Theory and implementation* (Vol. 2). World Scientific. https://doi.org/10.1142/9781860948671

[13] Ahmed, A., Kumar, S., Shah, A. A., & Bhutto, A. (2023). Cloud computing security issues and challenges. *Tropical Scientific Journal*, *2*(1), 1–8.

[14] Akbar, H., Zubair, M., & Malik, M. S. (2023). The security issues and challenges in cloud computing. *International Journal for Electronic Crime Investigation*, *7*(1), 13–32. https://doi.org/10.54692/ijeci.2023.0701125

[15] Thabit, F., Can, O., Alhomdy, S., Al-Gaphari, G. H., & Jagtap, S. (2022). A novel effective lightweight homomorphic cryptographic algorithm for data security in cloud computing. *International Journal of Intelligent Networks*, *3*, 16–30. https://doi.org/10.1016/j.ijin.2022.04.001

[16] Korda, D. R., Ansong, E. D., & Hodowu, D. K. M. (2021). Securing data in the cloud using the SDC algorithm. *International Journal of Computer Applications*, *183*, 24–29. https://doi.org/10.5120/ijca2021921631

[17] Wen, J. (2023). A layered encryption model PABB based on user privacy in e-commerce platforms. *Frontiers in Business, Economics and Management*, *9*(3), 10–14. https://doi.org/10.54097/fbem.v9i3.9428

[18] Vidya, S., & Deepa, T. (2022). Security enhancement using AES algorithm for emergency situation detection system. *IJISET – International Journal of Innovative Science, Engineering and Technology*, *09*.

[19] Kartit, Z., & El Marraki, M. (2015). Applying encryption algorithm to enhance data security in cloud storage. *Engineering Letters*, *23*(4).

[20] YueJuan, K., Yong, L., & Ping, L. (2020). A searchable ciphertext retrieval method based on counting bloom filter over cloud encrypted data. *IAENG International Journal of Computer Science*, *47*(2).

[21] El Balmany, C., Asimi, A., & Tbatou, Z. (2022). VMITLP: A security protocol towards a trusted launch process of a user generic virtual machine image on a public cloud IaaS platform. *IAENG International Journal of Computer Science*, *49*(1).

[22] Hu, Y., Lin, Y., Nie, Y., Peng, C., He, Y., Liu, Y., Ma, G., & Seng, D. (2024). Design and development of a BaaS system based on intelligent scheduling and operation cloud-edge platform. *IAENG International Journal of Computer Science*, *51*(3).

[23] Schinianakis, D., & Stouraitis, T. (2016). Residue number systems in cryptography: Design, challenges, robustness. In *Secure system design and trustable computing* (pp. 115–161). https://doi.org/10.1007/978-3-319-14971-4_4

[24] Baagyere, E. Y., Agbedemnab, P. A.-N., Qin, Z., Daabo, M. I., & Qin, Z. (2020). A multi-layered data encryption and decryption scheme based on genetic algorithm and residual numbers. *IEEE Access*, *8*, 100438–100447. https://doi.org/10.1109/ACCESS.2020.2997838

[25] Kasianchuk, M., Karpinski, M., Kochan, R., Karpinskyi, V., Litawa, G., Shylinska, I., & Yakymenko, I. (2020). Developing symmetric encryption methods based on residue number system and investigating their cryptosecurity. *Cryptology ePrint Archive.*

[26] Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (2018). *Handbook of applied cryptography.* CRC Press. https://doi.org/10.1201/9781439821916

[27] Parhami, B. (2010). *Computer arithmetic: Algorithms and hardware designs.* Oxford University Press.

[28] Mohan, P. V. A. (2007). RNS–to–binary converter for a new three–moduli set $\{2^n–1, 2^n, 2^n+1\}$. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *54*(9), 775–779. https://doi.org/10.1109/TCSII.2007.900844

[29] Gbolagade, K. A., Chaves, R., Sousa, L., & Cotofana, S. D. (2010). An improved RNS reverse converter for the $\{2^{2n+1}$-1, 2^n, 2^n-1$\}$ moduli set. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems* (pp. 2103–2106). IEEE. https://doi.org/10.1109/ISCAS.2010.5537062

[30] Daemen, J., & Rijmen, V. (2002). *The design of Rijndael* (Vol. 2). Springer. https://doi.org/10.1007/978-3-662-04722-4

[31] Abdullah, A. M., et al. (2017). Advanced encryption standard (AES) algorithm to encrypt and decrypt data. *Cryptography and Network Security*, *16*(1), 11.

[32] Rivera, L. B., Bay, J. A., Arboleda, E. R., Pereña, M. R., & Dellosa, R. M. (2019). Hybrid cryptosystem using RSA, DSA, ElGamal, and AES. *International Journal of Scientific & Technology Research*, *8*(10), 1777–1781.