cryptography

MDPI

Article

Affine Cipher Encryption Technique Using Residue Number System

Mykhailo Kasianchuk [1], Ruslan Shevchuk [2,3,*], Bogdan Adamyk [4], Vladlena Benson [4], Inna Shylinska [5] and Mykhailo Holembiovskyi [1]

[1] Department of Cyber Security, West Ukrainian National University, 46009 Ternopil, Ukraine; kmm@wunu.edu.ua (M.K.); mykhailo.2097@gmail.com (M.H.)
[2] Department of Computer Science and Automatics, University of Bielsko-Biala, 43-309 Bielsko-Biala, Poland
[3] Department of Computer Science, West Ukrainian National University, 46009 Ternopil, Ukraine
[4] Aston Business School, Aston University, Birmingham B4 7ET, UK; b.adamyk@aston.ac.uk (B.A.); v.benson@aston.ac.uk (V.B.)
[5] Department of Foreign Languages, Information and Communication Technologies, West Ukrainian National University, 46009 Ternopil, Ukraine; inna.shylinska2012@gmail.com
* Correspondence: rshevchuk@ubb.edu.pl

Abstract: This paper presents a new encryption technique, which combines affine ciphers and the residue number system. This makes it possible to eliminate the shortcomings and vulnerabilities of affine ciphers, which are sensitive to cryptanalysis, using the advantages of the residue number system, i.e., the parallelization of calculation processes, performing operations on low bit numbers, and the linear combination of encrypted residues. A mathematical apparatus and a graphic scheme of affine encryption using the residue number system is developed, and a corresponding example is given. Special cases of affine ciphers such as shift and linear ciphers are considered. The cryptographic strength of the proposed cryptosystem when the moduli are prime numbers is estimated, and an example of its estimation is given. The number of bits and the number of moduli of the residue number system, which ensure the same cryptographic strength as the longest key of the AES algorithm, are determined.

Keywords: affine ciphers; residue number system; module; Caesar cipher; cryptographic methods; encryption keys

Academic Editor: Josef Pieprzyk

Received: 11 March 2025
Revised: 17 April 2025
Accepted: 22 April 2025
Published: 24 April 2025

Citation: Kasianchuk, M.; Shevchuk, R.; Adamyk, B.; Benson, V.; Shylinska, I.; Holembiovskyi, M. Affine Cipher Encryption Technique Using Residue Number System. *Cryptography* **2025**, *9*, 26. https://doi.org/10.3390/cryptography9020026

1. Introduction

The protection of information that is constantly transmitted and electronically stored is a critical issue in modern society [1–5], and cryptographic methods [6–9] are one of the key components of data security that ensure the confidentiality, integrity, and authenticity of data, which is especially important in the age of digital technologies. Given the ever-increasing number of cyber threats [10–13], cryptographic protection plays a crucial role in ensuring privacy and security in the digital environment. Encryption methods are used to prevent data leakage and ensure reliable functioning of modern information systems [14–16]. However, with increasing key lengths, especially in asymmetric cryptosystems, certain problems arise when encrypting short messages.

Affine ciphers are among the simplest symmetric ciphers. They are easy to implement and require low computational resources, which makes them suitable for use in resource-constrained environments, such as embedded systems, mobile devices, or Internet of Things systems [17–19]. Due to their simplicity, they quickly perform encryption and decryption, providing a sufficient level of protection in many everyday applications without requiring

a lot of power or memory. However, it is relatively easy to cryptanalyze affine ciphers, which limits their applications.

The combination of affine ciphers and the residue number system (RNS) [20,21] provides an additional level of protection by dividing the numeric notation of characters into several independent components. This complicates cryptanalysis, as the attacker needs to take into account the number of moduli and their combinations. This approach increases the cipher resistance to various attacks, including brute force attacks [22,23]. Affine ciphers based on the RNS (ACRNS) can be easily adapted to different sizes of alphabets and coding systems. This makes them flexible for a variety of applications, ranging from textual data to specialized data formats [24]. Scalability is ensured by selecting appropriate moduli [25], which allows one to control the level of security [26].

1.1. Contribution

The main contribution of this paper is as follows:

(1) A new efficient technique for the encryption of information flows based on a combination of affine ciphers and the RNS is developed, and its scheme and mathematical justification are proposed;
(2) An example of affine encryption using RNS is given, in which the calculation of the basic parameters of the RNS, key generation, as well as the encryption and decryption features are considered;
(3) It is determined that the cryptographic strength of the proposed system depends on the number of moduli and their bit size;
(4) The parameters of the cryptographic system, which has the same resistance to cryptanalysis as the AES-256 symmetric encryption standard, are defined.

1.2. Organization

The rest of this paper is organized as follows: Section 2 presents an analysis of the related work. The theoretical foundations of affine ciphers and the RNS are given in Section 3. Based on the use of affine ciphers and the RNS, a new technique for encrypting information flows is developed. Section 4 presents an example of the cryptographic transformation of integer data using the developed technique. Section 5 provides an estimation of the cryptographic strength of the proposed technique and a comparison with the AES-256 symmetric encryption standard. Section 6 summarizes the contents of the paper and outlines the prospects for further research.

2. Related Work

ACRNSs can provide a significant level of information protection due to their simplicity, efficiency, and security, which the RNS ensures. Although they are not the most resistant to all possible attacks, their use can be justified in practice, especially when speed and simplicity of implementation are required [27]. They also serve as a good basis for the further study and development of more complex cryptographic systems.

For example, in [28], to increase the security level of databases, it was suggested to use an affine cipher after the Caesar cipher. The encrypted text is stored as raw data in a text file that is separated between fields. In [29], using the State Transition Diagram technique, a cryptographic password prototype was built and implemented using modified affine transformations.

One of the ways to increase the cryptographic strength of affine ciphers is to use additional keys. In particular, the algorithm proposed in [30] combines the advantages of affine and stream ciphers, which makes it possible to reduce the number of operations that are required for modern modifications of affine ciphers. The new cipher also increases the

degree randomization for the affine cipher by adding an additional key and can be used in online data transmission. In [31], a method was developed in which the first random number is used to generate the key stream of unique values of an affine cipher. This makes it possible to dynamically change the encryption process. The simulation results showed that, on the basis of the proposed method, a more randomized encrypted text is created, in contrast to the traditional affine cipher. In [32], an affine cipher with the Diffie–Hellman key exchange was successfully implemented on an Arduino ATmega2560 device. It can be applied on IoT devices, given their limited computing resources. In the avalanche test, the algorithm showed a high security level.

Affine ciphers are also used to protect graphic resources. For example, in [33], a new algorithm for encrypting color images was presented using a related configuration of a two-dimensional Henon map, a three-dimensional logistic map with XOR operation, and an affine Hill cipher (AHC) technique. The proposed scheme introduces a modified procedure for generating the initial key values of the three-dimensional logistic map, which significantly increases the security of the system. The obtained results and comparative analysis with the existing methods prove the safety, efficiency, and effectiveness of the proposed method. In addition, the correct location of the keys in the AHC technique at the decryption stage, the key space, and their sensitivity counteract various types of cryptanalytic attacks, increasing the overall security of the system.

An algorithm for image encryption based on an affine cipher was presented in [34]. The algorithm first encrypts the pixel positions using a Lorenz chaotic sequence; then, an affine cipher is used to scatter and scramble the pixel values. The theoretical analysis and experimental results showed that the algorithm has keys with high sensitivity, better encryption security, and high diffusion and confusion values. In [35], the concept of the theory of affine groups was applied to protect digital images using the DES algorithm and wavelet transform. Matrix multiplication and vector addition operations were used. As a result, the image that was transformed using the affine transformation was saved without changing its dimensions.

Increasing the reliability of affine ciphers can be achieved by combining them with other cryptographic algorithms. Thus, in [36], the affine cryptography technique was first used, and then, a transposition cipher was applied to the received text. This complicates the cryptanalysis of ciphertext due to the use of keys of different types. At the same time, in general, plaintext and encrypted text differ in the number of characters. In [37], the authors combined the Rabin asymmetric cryptographic algorithm and a symmetric affine cipher. As a result, a hybrid scheme was obtained, in which the affine transformation was used to encrypt messages, and the Rabin cryptosystem was used to encrypt and decrypt the key. It was shown that *.pdf files could be recovered without the loss of integrity.

In [38], an affine cipher using asymmetric keys generated from rectangular matrices was introduced. This made it possible to increase the cryptographic strength of the proposed encryption algorithm. An asymmetric cryptosystem using an affine Hill cipher was developed in [39]. The proposed method increases the security of the system, as it involves the use of two or more digital signatures when modulating a prime number. In [40], it was shown that the modification of the original message using a linear congruent generator could increase the level of security by increasing the number of messages and adding random numerical values to the plaintext. In [41], a cryptographic algorithm was proposed, combining an affine cipher with the Blum Blum Shub pseudorandom number generation algorithm. This made it possible to generate a random stream of keys, the use of which increases the unpredictability of the encrypted text and, accordingly, increases the cryptographic strength of the proposed algorithm.

Therefore, the combination of affine ciphers and the RNS allows us to use the advantages of the latter, in particular the parallelization of the calculation process, the possibility of performing arithmetic operations on lower bit operands, and the absence of inter-bit carriers. The proposed cryptosystem shows high cryptographic strength due to the linear combination of encrypted residues.

3. Materials and Methods

Sections 3.1 and 3.2, respectively, highlight the theoretical foundations of affine ciphers and the RNS. In Section 3.3, a new encryption technique based on a combination of affine ciphers and the RNS is developed, and the use of the developed technique in some special cases is discussed in Section 3.4.

3.1. Analysis of Affine Ciphers

An affine cipher is a special case of the more general monoalphabetic substitution cipher, and it has all the vulnerabilities of this type of ciphers. In particular, it is easily subjected to frequency cryptanalysis; that is, its cryptographic properties are weak. In an affine cipher, each character of the plaintext is mapped to a numeric equivalent of the letter in the English alphabet, not taking into account uppercase and lowercase characters. The correspondence between letters and numbers can be made as shown, for example, in Table 1.

Table 1. Correspondence between letters of the English alphabet and numbers.

Letter	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**
Number	00	01	02	03	04	05	06	07	08	09	10	11	12
Letter	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
Number	13	14	15	16	17	18	19	20	21	22	23	24	25

Table 1 can be expanded to include not only textual information but also any other kind of information such as video, audio, graphic, etc. In addition, standard encoding schemes such as ASCCI code or Unicode can be used.

Then, based on the properties of modular arithmetic, for each number that corresponds to a plaintext character, a new number is calculated that replaces the previous one. Thus, the ciphertext is generated. At the same time, each letter is encrypted on the basis of the linear function and can be shown as follows:

$$X = (ax + s) \bmod n, \tag{1}$$

where x and X are letter numbers of the plaintext and encrypted text, respectively; pair a and s are cipher keys, for which the following conditions must be met: $1 \leq a \leq n - 1$, and GCD $(a, n) = 1$, $0 \leq s \leq n - 1$.

The following conversion is used for decryption:

$$x = (AX + S) \bmod n, \tag{2}$$

where $A = a^{-1} \bmod n$ is the inverse of number a relatively prime modulo n; $S = (-As) \bmod n$.

The number of possible keys for an affine cipher can be written using Euler's function as follows: $\phi(n) = -1$ (in the case when $a = 1$, $s = 0$ is not taken into account).

If $a = 1$, the Caesar cipher is obtained, and the encryption and decryption functions are reduced to a simple linear shift:

$$X = (x + s) \bmod n,\ x = (X + S) \bmod n, \tag{3}$$

where $S = (-s) \bmod n = n - s$.

The number of keys in this case is $n - 1$ (the case when $s = 0$ is not taken into account).

If $a \neq 1$ and $s = 0$, then only the multiplication operation is performed for encryption and decryption:

$$X = (ax) \bmod n,\ x = (AX) \bmod n, \tag{4}$$

The number of keys will be $\phi(n) - 1$.

3.2. Theoretical Foundations of the Residue Number System

Any number N, given in the decimal number system, can be written in the RNS as a set of residues b_i from its division by certain selected numbers p_i, which are called moduli [42–44]:

$$b_i = N \bmod p_i. \tag{5}$$

At the same time, two conditions must be met:

(1) All moduli are relatively prime;

(2) The selected number N is less than the product of all moduli: $N < P = \prod_{i=1}^{k} p_i$, where k is a number of moduli.

To recover the decimal notation of a number from its residues, the Chinese Remainder Theorem (CRT) can be used [45,46]:

$$N = \left(\sum_{i=1}^{k} M_i m_i b_i \right) \mathrm{mod} P, \tag{6}$$

where $M_i = \frac{P}{p_i} = p_1 p_2 \ldots p_{i-1} p_{i+1} \ldots p_k$ is the product of all moduli, except for p_i; $m_i = M_i^{-1} \mathrm{mod} p_i = (M_i \mathrm{mod} p_i)^{-1} \mathrm{mod} p_i$ represent the corresponding modular inverses.

Another method for decimal number recovery from its residues is to use Garner's algorithm, according to which N can be uniquely noted as follows:

$$N = N_0 + N_1 p_1 + N_2 p_1 p_2 + \ldots + N_{k-1} p_1 p_2 \ldots p_{k-1}, \tag{7}$$

where $0 \leq N_i < p_{i+1}$, $i = 0, 1, \ldots, k - 1$; N_i parameters can be successively calculated one by one using the recurrence relation:

$$N_i = \left(\left((p_1 p_2 \ldots p_i)^{-1} \mathrm{mod} p_{i+1} \right) \cdot (b_{i+1} - (N_0 + N_1 p_1 + \ldots + N_{i-1} p_1 p_2 \ldots p_{i-1})) \right) \mathrm{mod} p_{i+1}, \tag{8}$$

These and other methods of recovering a decimal number from its residues (such as adding the product of moduli or residues from the product of moduli) [47] are rather cumbersome and are characterized by high time requirements. It can be reduced using the modified perfect form (MPF) of the RNS [48], in which the moduli are selected in such a way that the following conditions are met for any of them:

$$m_i = M_i^{-1} \mathrm{mod} p_i = (M_i \mathrm{mod} p_i)^{-1} \mathrm{mod} p_i = \pm 1, \tag{9}$$

Calculations are carried out according to the CRT based on Formula (6), in which the sum becomes sign-changing and each term consists of two, not three, factors. In addition, it is not necessary to find the multiplicative modular inverse.

3.3. Affine Ciphers in the Residue Number System

The difference between a simple affine cipher and a combination of an affine cipher and the RNS is that when using a simple affine cipher, each letter is separately encrypted, and a combination of an affine cipher and the RNS makes it possible to convert a block of plaintext N, which must be smaller than the product of the selected moduli P.

Then, residues b_i are found by Formula (5) and subjected to cryptographic transformation:

$$B_i = (a_i b_i + s_i) \bmod p_i. \tag{10}$$

Similar conditions that must be met for affine ciphers are required for the a_i, s_i, and p_i keys: $1 \le a_i \le p_i - 1$, and GCD $(a_i, p_i) = 1$, $0 \le s_i \le p_i - 1$.

If $a_i = 1$ (shift cipher) or $s_i = 0$, the formulas for encryption are as follows:

$$B_i = (b_i + s_i) \bmod p_i; \tag{11}$$

$$B_i = (a_i b_i) \bmod p_i. \tag{12}$$

The concatenation of the changed residues B_i can be a ciphertext. However, in order to increase the resistance of the latter to cryptanalysis, it is expedient to recover the decimal number K with residues B_i:

$$K = \left(\sum_{i=1}^{k} M_i m_i B_i \right) \bmod P. \tag{13}$$

Message K is the final ciphertext.

To decipher it, it is first necessary to find the changed residues from the following expressions:

$$B_i = (a_i b_i + s_i) \bmod p_i. \tag{14}$$

The calculation of real residues is performed according to formulas that are similar to (2):

$$b_i = (A_i B_i + S_i) \bmod p_i, \tag{15}$$

where $A_i = {a_i}^{-1} \bmod p_i$ represent the inverses of a_i by relatively prime moduli p_i; correspondingly, $S_i = (-A_i s_i) \bmod p_i$.

Figure 1 shows the scheme of the proposed affine encryption technique using the RNS.

The number of possible keys in this case increases significantly compared to the classical affine cipher. Its value can be estimated as $\prod_{i=1}^{k} (\phi(p_i) \cdot p_i) - 1$ (the possibility when $a_i = 1$ and $s_i = 0$ is not considered). If we assume that all moduli are prime numbers, the number of keys can be estimated from the expression $\prod_{i=1}^{k} ((p_i - 1) \cdot p_i) - 1$.

The encryption and decryption process is described in the following (see Algorithms 1 and 2, respectively).

If the plaintext message N exceeds the product of moduli P, then it, similarly to block ciphers, is divided into numerical blocks that are smaller than P, which can be encrypted according to the corresponding modes. Another method is to increase the number of moduli or their bit size and, accordingly, the number P.

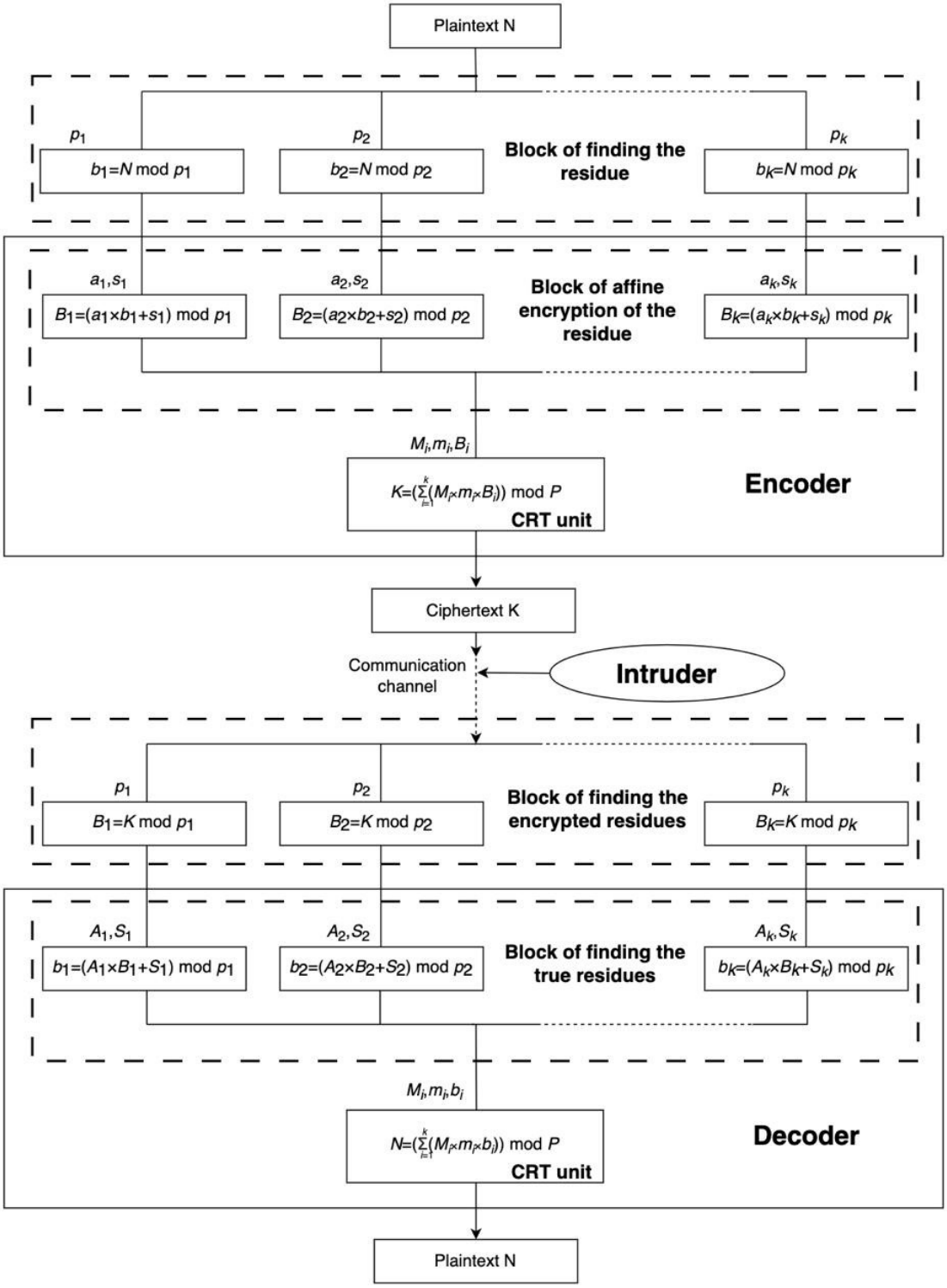

Figure 1. Affine encryption scheme using RNS.

Algorithm 1: The ACRNS encryption algorithm

Input:
N—number to be encrypted
$\mathbf{p_i}$—list of pairwise coprime moduli [p_1, p_2, . . ., p_k]
$\mathbf{a_i}$, $\mathbf{s_i}$—encryption keys, where $0 < a_i < p_i$, $0 \leq s_i < p_i$, GCD(a_i, p_i) = 1
Output:
K—encrypted number

```
function Encrypt(N, p_i, a_i, s_i):
// Compute remainders of N modulo each modulus
for x from 0 to k − 1:
    r = N mod p_i[x]
    b.append(r)
// Encrypt each remainder using the affine cipher:
//      B[x] = (a_i[x] × b[x] + s_i[x]) mod p_i[x]
for x from 0 to k − 1:
    encrypted = (a_i[x] × b[x] + s_i[x]) mod p_i[x]
    B.append(encrypted)
// Compute modular inverse using Extended Euclidean Algorithm
function modular_inverse(a, m):
    t, newt = 0, 1
    r, newr = m, a
      while newr ≠ 0:
        quotient = r // newr
        t, newt = newt, t − quotient × newt
        r, newr = newr, r − quotient × newr
    if r > 1:
        error "Inverse does not exist"
    if t < 0:
        t = t + m
    return t
// CRT to reconstruct the encrypted number
function CRT(residues, moduli):
    P = 1
    for p in moduli:
        P = P × p
    result = 0
    for i from 0 to length(moduli) − 1:
        p_i = moduli[i]
        r_i = residues[i]
        m_i = P/p_i
        mi_inv = modular_inverse(m_i, p_i)
        result = result + r_i × m_i × mi_inv
    return result mod P
    // 5. Build encrypted number K from wrong remainders
    K = CRT(B, p_i)
    return K
```

Algorithm 2: The ACRNS decryption algorithm

Input:
K—the encrypted number
$\mathbf{p_i}$—list of moduli $[p_1, p_2, \ldots, p_k]$
$\mathbf{a_i}$, $\mathbf{s_i}$—encryption keys, where $0 < a_i < p_i$, $0 \leq s_i < p_i$, $GCD(a_i, p_i) = 1$
Output:
N — decoded number

```
function Decrypt(K, p_i, a_i, s_i):
    k = length(p_i)
    B = [] // encrypted remainders extracted from K
    // Extract encrypted remainders by taking K mod each pi
    for x from 0 to k − 1:
        r = K mod p_i[x]
        B.append(r)
    // Recover original remainders using inverse affine transformation
    b = [] # decrypted (correct) remainders
    for x from 0 to k − 1:
        ai_inv = modular_inverse(a_i[x], p_i[x])
        b_x = (ai_inv × (B[x] − s_i[x])) mod p_i[x]
        b.append(b_x)
    // Modular inverse function
    function modular_inverse(a, m):
        t, newt = 0, 1
        r, newr = m, a
          while newr ≠ 0:
            quotient = r // newr
            t, newt = newt, t − quotient × newt
            r, newr = newr, r − quotient × newr
        if r > 1:
            error "Inverse does not exist"
        if t < 0:
            t = t + m
        return t
    // Use CRT to recover the original number from correct remainders
    function CRT(residues, moduli):
        P = 1
        for p in moduli:
            P = P × p
        result = 0
        for i from 0 to length(moduli) − 1:
            p_i = moduli[i]
            r_i = residues[i]
            m_i = P/p_i
            mi_inv = modular_inverse(m_i, p_i)
            result = result + r_i × m_i × mi_inv
        return result mod P
    // Recover the original number N
    N = CRT(b, p_i)
    return N
```

3.4. Special Cases of Affine Ciphers

If $a_i = 1$ (shift cipher) or $s_i = 0$, the formulas for calculating the real residues are simplified:

$$b_i = (B_i + S_i) \bmod p_i; \tag{16}$$

$$b_i = (A_i B_i) \bmod p_i, \tag{17}$$

where $A_i = {a_i}^{-1} \bmod p_i$; $S_i = (-s_i) \bmod p_i = (p_i - s_i) \bmod p_i$.

The number of possible variants of the keys will be, respectively, $\prod_{i=1}^{k} p_i - 1$ and $\prod_{i=1}^{k} (\phi(p_i)) - 1$ (or $\prod_{i=1}^{k} (p_i - 1) - 1$ if the moduli p_i are prime numbers.

For example, a plaintext message can be recovered on the basis of the RNS according to Formula (6).

4. Results

Section 4.1 shows an example of affine encryption of integers in general using the residue number system, and in Section 4.2, the applications of some special cases are given.

4.1. An Example of Affine Ciphers Using the Residue Number System

Let us consider the system of moduli $p_1 = 9$; $p_2 = 10$; $p_3 = 11$; and $p_4 = 17$. Their product (range of calculations) $P = 16{,}830$. Next, the basic parameters of this system are calculated: $M_1 = 10 \cdot 11 \cdot 17 = 1870$; $M_2 = 9 \cdot 11 \cdot 17 = 1683$; $M_3 = 9 \cdot 10 \cdot 17 = 1530$; $M_4 = 9 \cdot 10 \cdot 11 = 990$; 1870 mod 9 = 7; 1683 mod 10 = 3; 1530 mod 11 = 1; and 990 mod 17 = 4. Since the selected moduli are relatively small, it is expedient to determine the modular inverses of the found numbers in the following way: add 1 to the modulus and check whether the found sum is evenly divisible by the corresponding number. If so, then the quotient is the inverse; if not, then the modulus is added until the quotient is an integer. Therefore, 1 + 9 = 10; 10 + 9 = 19; and 19 + 9 = 28; then, $m_1 = 28{:}7 = 4$; 1 + 10 = 11; 11 + 10 = 21; $m_2 = 21{:}7 = 3$; $m_3 = 1$; 1 + 17 = 18; 18 + 17 = 35; 35 + 17 = 52; and $m_4 = 52{:}4 = 13$.

The obtained results are given in Table 2.

Table 2. Basic parameters for the system of selected moduli.

i	**1**	**2**	**3**	**4**
p_i	9	10	11	17
M_i	1870	1683	1530	990
$M_i \bmod p_i$	7	3	1	4
m_i	4	7	1	13

Let us assume that the message “bot” needs to be encrypted in this system of the selected moduli: 9, 10, 11, and 17. According to Table 1, this message in a number format is as follows: 011419. The input data (plaintext N = 11,419), selected keys for encryption, and results obtained from expressions (5), (10), and (13) are shown in Table 3.

Table 3. Input data, keys for encryption, and obtained results.

i	1	2	3	4
p_i	9	10	11	17
a_i	4	3	4	8
s_i	4	6	5	10
N		11,419		
$b_i = N \bmod p_i$	7	9	1	12
$B_i = (a_i b_i + s_i) \bmod p_i$	5	3	9	4
$K = \left(\sum_{i=1}^{k} M_i m_i B_i\right) \bmod P$		3353		

It should be noted that the ciphertext can be both the concatenation of the true b_i or the changed residues of B_i (07090112 and 05030904, respectively) and the number K = (1870·4·5 + 1683·7·3 + 1530·1·9 + 990·13·4) mod 16,830 = 3353, recovered from the changed residues using the CRT (Formula (13)).

A_i's parameters, which are one part of the decryption key and are defined as modular inverses, are easy to find by adding the modulus: 1 + 9 = 10; 10 + 9 = 19; and 19 + 9 = 28; then, A_1 = 28:4 = 7; 1 + 10 = 11; 11 + 10 = 21; A_2 = 21:3 = 7; 1 + 11 = 12; A_3 = 12:4 = 3; 1 + 17 = 18; 18 + 17 = 35; 35 + 17 = 52; 52 + 17 = 69; 69 + 17 = 88; and A_4 = 88:8 = 11. The rest of the decryption keys (B_i parameters) are determined as follows: S_1 = (−7·4) mod 9 = 8; S_2 = (−7·6) mod 10 = 8; S_3 = (−3·5) mod 11 = 7; and S_4 = (−11·10) mod 17 = 9. Having obtained the true bi residues, using the CRT and the data in Table 2, the plaintext can be recovered: N = (1870·4·7 + 1683·7·9 + 1530·1·1 + 990·13·12) mod 16,830 = 11,419. Therefore, the decrypted message corresponds to the original plaintext.

The input parameters, decryption keys, and decryption results are given in Table 4.

Table 4. Input data, decryption keys, and decryption results.

i	1	2	3	4
p_i	9	10	11	17
A_i	7	7	3	15
S_i	8	8	7	3
K		3353		
$B_i = K \bmod p_i$	5	3	9	4
$b_i = (A_i B_i + S_i) \bmod p_i$	7	9	1	12
$N = \left(\sum_{i=1}^{k} M_i m_i b_i\right) \bmod P$		11,419		

4.2. Example of the Use of Special Cases of Affine Ciphers

When $s_i = 0$ ($a_i \neq 1$), using the same input parameters and having found the changed residues, the ciphertext is determined as follows: K = (1870·4·1 + 1683·7·7 + 1530·1·4 + 990·13·11) mod 16,830 = 2017. During decryption, the true bi residues are found first, from which the plaintext is recovered using the CRT and the data in Table 2. The encryption and decryption results are shown in Table 5.

When $a_i = 1$ ($s_i \neq 0$), using the same input parameters and having found the changed residues, the ciphertext is determined as follows: K = (1870·4·2 + 1683·7·5 + 1530·1·6 + 990·13·5) mod 16,830 = 12755. During decryption, the true bi residues are found first, from which the plaintext is recovered using the CRT and the data in Table 2. The encryption and decryption results are given in Table 6.

In all these cases, the decrypted text is equal to the input plaintext.

Table 5. Input data, encryption keys, and obtained results for $s_i = 0$.

i	1	2	3	4
p_i	9	10	11	17
a_i	4	3	4	8
N		11,419		
$b_i = N \bmod p_i$	7	9	1	12
$B_i = (a_i b_i) \bmod p_i$	1	7	4	11
$K = \left(\sum_{i=1}^{k} M_i m_i B_i\right) \mathrm{mod} P$		2017		
A_i	7	7	3	15
$b_i = (A_i B_i) \bmod p_i$	7	9	1	12
$N = \left(\sum_{i=1}^{k} M_i m_i b_i\right) \mathrm{mod} P$		11,419		

Table 6. Input data, encryption keys, and obtained results for $a_i = 1$.

i	1	2	3	4
p_i	9	10	11	17
s_i	4	6	5	10
N		11,419		
$b_i = N \bmod p_i$	7	9	1	12
$B_i = (b_i + s_i) \bmod p_i$	2	5	6	5
K		12,755		
S_i	5	4	6	7
$b_i = (A_i B_i) \bmod p_i$	7	9	1	12
$N = \left(\sum_{i=1}^{k} M_i m_i b_i\right) \mathrm{mod} P$		11,419		

5. Discussion of the Results

Section 5.1 is devoted to the study of the cryptographic strength of the ACRNS. Section 5.2 provides a comparison of the cryptographic strength of the proposed technique and the AES-256 symmetric encryption standard.

5.1. Cryptographic Strength of Affine Ciphers Using the Residue Number System

The cryptographic strength of ACRNSs is their ability to resist cryptanalysis, which refers to the product of the time complexity of one key variant, which is estimated using Big-Oh notation, multiplying by a number of key variants.

To approximately estimate the cryptographic strength of the ACRNS, let us assume that the moduli are prime numbers, the bit size of which is in the range from $(n - t)$ to n. According to the prime number theorem describing the asymptotic distribution of prime numbers, their number in this range can be approximated as follows: $\pi(n) = \frac{2^n}{n \cdot \ln 2} - \frac{2^{n-t}}{(n-t)\ln 2}$. Then, k moduli can be selected in the following number of ways: $C^k_{\pi(n)} = \frac{\pi(n)!}{k! \cdot (\pi(n)-k)!} = \frac{\prod_{i=0}^{k-1}(\pi(n)-i)}{k!}$. For the convenience of estimating the number of ways to select moduli, the last expression can be approximated as follows: $\left(\frac{2\pi(n)-k+1}{k}\right)^k$.

For an approximate estimation of the number of key variants a_i and s_i, which is $\prod_{i=1}^{k}((p_i - 1) \cdot p_i) - 1 \approx \prod_{i=1}^{k} p_i^2$, it can be assumed that their average length will not be less than $\frac{n-t}{2}$. Thus, the total number of variants can be approximated as $2^{k(n-t)}$. In addition, the time complexity of affine transformations and the RNS for k moduli can be approximated as n^{2k}.

Therefore, the overall cryptographic strength of the proposed system will be equal to the product of these three estimated parameters:

$$O(n,k,t) \approx \frac{2^{k(n-t)} \cdot n^{2k}}{k^k} \cdot \left(\frac{2^n}{n \cdot \ln 2} - \frac{2^{n-t}}{(n-t) \cdot \ln 2} - k + 1 \right)^k. \quad (18)$$

For example, Table 7 shows the values of the decimal logarithm (or orders) of the cryptographic strength with different values of the t, k, and n parameters.

Table 7. The values of the decimal logarithm *lg(O(n, k, t))* of cryptographic strength with different values of the t, k, and n parameters.

t	k	n						
		16	**32**	**64**	**128**	**256**	**512**	**1024**
1	3	29	59	118	234	466	930	1855
	6	57	117	234	467	931	1858	3709
	10	93	193	388	777	1550	3095	6180
10	3	22	52	111	227	459	922	1848
	6	43	102	220	453	917	1844	3695
	10	69	169	364	753	1526	3071	6156

According to Table 7, it can be stated that with an increase in the number of moduli and their bit size, the strength of the cryptosystem increases, and with an increase in the t parameter, it decreases.

Figure 2 shows a logarithmic scale of graphical dependence of the cryptographic strength on the bit size of moduli n and their number k when $t = 3$. It can be seen that with an increase in the specified parameters, the complexity of the cryptanalysis significantly increases.

Figure 3 shows a logarithmic scale of the graphical dependence of the cryptographic strength of the ACRNS on the bit size of moduli when their number differs and $t = 10$.

The presented graphs are linear. It can be seen that the resistance to cryptanalysis increases with an increase in the bit size of the moduli.

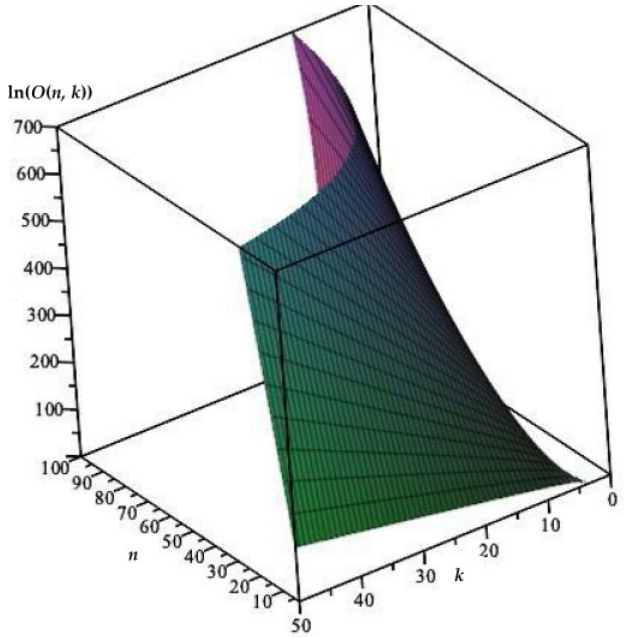

Figure 2. Graphical dependence of the cryptographic strength of the proposed technique on the bit size of moduli and their number.

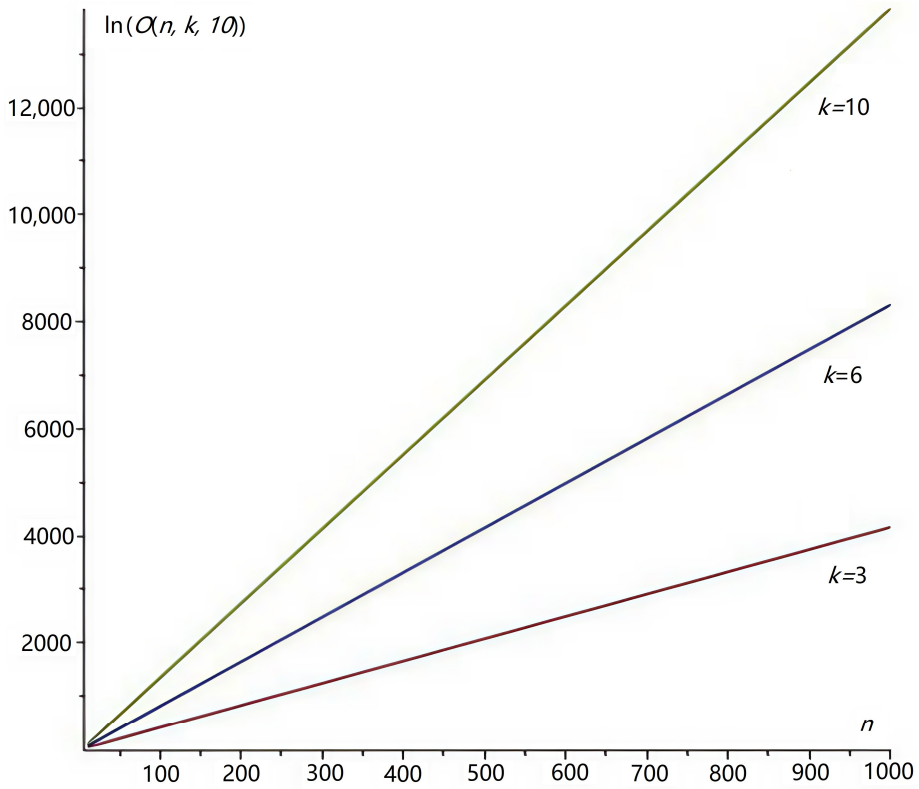

Figure 3. Graphical dependence of the cryptographic strength of the ACRNS on the bit size of moduli when their number differs and $t = 10$.

5.2. Comparison of the Cryptographic Strength of an Affine Cipher Using the Residue Number System with the AES Cryptographic Algorithm

According to [49,50], it is known that 2^{n-1} bit operations are required for cryptanalysis of the modern symmetric AES cryptographic algorithm with an n-bit key (the maximum key length of the AES algorithm is 256 bits). Then, due to the equality $O(n, k, t) = 2^{255}$, the number of RNS moduli and their bit sizes can be determined, which ensures that the cryptographic strength that is no less than that ensured by the longest key of the AES algorithm (Table 8).

The presented table shows that as the number of moduli increases, their bit size decreases, and this dependence is non-linear.

Table 8. Bit sizes and the number of RNS moduli, which ensure that the cryptographic strength is not less than that ensured by the longest key of the AES cryptographic algorithm when the parameter t has different values.

Number of Moduli	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit size when $t = 1$	63	42	32	26	22	19	17	16	14	13	12	11	11	10
Bit size when $t = 10$	66	46	36	29	25	23	21	19	18	17	16	15	14	14

6. Prospects and Directions for Further Research

Due to their structural simplicity, affine ciphers provide a high speed when performing encryption/decryption operations. Therefore, despite their limited cryptographic strength, their use appears promising for solving a wide range of applied problems that require low computing resources, in particular, in mobile devices, various embedded systems, and Internet of Things technologies. The combination of affine ciphers with a non-positional RNS will allow us to increase the cryptographic strength of encryption without a significant loss of performance. This can be achieved due to the properties of the RNS, that is,

parallelization of the computation process and execution of arithmetic operations on relatively small operands. Therefore, the use of the ACRNS is especially promising in systems where time is a critical parameter.

The software and hardware–software implementation of the proposed ACRNS is considered a promising direction for further research. This will allow for comparing experimental results, in particular the strength and performance of this cipher and known standards of symmetric and asymmetric encryption. In addition, research can be carried out using a different number of moduli and their bit size, due to which it is possible to achieve an acceptable level of resistance to cryptanalysis and the speed of the algorithm.

An extremely important and promising direction for further research in this field is the development of a matrix ACRNS, as well as the use of a perfect and modified form of the RNS, which significantly simplifies the process of converting a number into a decimal notation from its residues.

7. Conclusions

In this paper, a new encryption technique is developed, which consists of combining affine ciphers and a non-positional RNS. This approach makes it possible to eliminate the shortcomings of affine ciphers, which are sensitive to cryptanalysis, due to the advantages of the residue number system, in particular the parallelization of calculation processes, the performance of operations on low bit numbers, and linear combinations of real and encrypted residues. Mathematical support is developed, and a graphical scheme for affine encryption using the RNS and an implementation example are given. Special cases of affine ciphers, including a shift cipher and a linear cipher, are considered. The cryptographic strength of the proposed encryption algorithm is estimated. When prime numbers of a given bit size are selected as moduli, its graphical dependence and a corresponding example are shown. The bit sizes and the number of RNS moduli that ensure the same cryptographic strength as the longest key of the AES algorithm are determined.

Author Contributions: Conceptualization, M.K. and R.S.; methodology, M.K. and M.H.; validation, I.S., R.S., B.A., V.B. and M.K.; formal analysis, M.H., V.B. and I.S.; investigation, R.S., B.A., M.H. and M.K.; resources, I.S.; data duration, R.S.; writing—original draft preparation, M.K., M.H. and I.S.; writing—review and editing, M.H. and R.S.; supervision, M.K.; funding acquisition, R.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nieles, M.; Dempsey, K.; Pillitteri, V.Y. *An Introduction to Information Security*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2017.
2. Shirazi, S.-R.; Shah, S.A.; Anwar, A. Information Security. In Proceedings of the 27th International Conference, Arlington, VA, USA, 23–25 October 2024.
3. Andrzejewski, K. Security Information Management Systems. *Nauki Zarz.* **2020**, *24*, 1–9. [CrossRef]
4. Andrijchuk, V.A.; Kuritnyk, I.P.; Kasyanchuk, M.M.; Karpinski, M.P. Modern Algorithms and Methods of the Person Biometric Identification. In Proceedings of the 2005 IEEE Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Sofia, Bulgaria, 5–7 September 2005; pp. 403–406.
5. Laybats, C.; Tredinnick, L. Information Security. *Bus. Inf. Rev.* **2016**, *33*, 76–80. [CrossRef]

6. Hoffstein, J.; Pipher, J.; Silverman, J. *An Introduction to Mathematical Cryptography*; Springer New York: New York, NY, USA, 2008; ISBN 9780387779935.
7. Jeffrey, H.; Jill, P.; Joseph, H. *An Introduction to Cryptography*; Springer: New York, NY, USA, 2008.
8. Adki, V.; Hatkar, S. A Survey on Cryptography Techniques. *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* **2015**, *6*, 469–475.
9. Washington, L.C. *Elliptic Curves*; Chapman and Hall/CRC: London, UK, 2008; ISBN 9780429140808.
10. Fadziso, T.; Thaduri, U.R.; Dekkati, S.; Ballamudi, V.-R.; Desamsetti, H. Evolution of the Cyber Security Threat: An Overview of the Scale of Cyber Threat. *Digit. Sustain. Rev.* **2023**, *3*, 1–12.
11. Asaad, R.R.; Saeed, V.A. A Cyber Security Threats, Vulnerability, Challenges and Proposed Solution. *Appl. Comput. J.* **2022**, *2*, 227–244. [CrossRef]
12. Wang, Z.; Adeyemo, D.; Akinsoto, A. Summary of Cyber Threat Intelligence. *Int. J. Innov. Res. Multidiscip. Field* **2022**, *8*, 32–42.
13. Humayun, M.; Niazi, M.; Jhanjhi, N.Z.; Alshayeb, M.; Mahmood, S. Cyber Security Threats and Vulnerabilities: A Systematic Mapping Study. *Arab. J. Sci. Eng.* **2020**, *45*, 3171–3189. [CrossRef]
14. Tarawneh, M. *Perspective Chapter: Cryptography—Recent Advances and Research Perspectives. Biometrics and Cryptography*; IntechOpen: London, UK, 2024; ISBN 9781837682621.
15. Kumar, D.V.; Raheja, E.G.; Sareen, M.S. CRYPTOGRAPHY. *Int. J. Comput. Technol.* **2013**, *4*, 29–32. [CrossRef]
16. Onwutalobi, A.-C. Overview of Cryptography. *SSRN Electron. J.* **2011**, *1*, 1–10. [CrossRef]
17. Om, H.; Patwa, R. Affine Transformation in Cryptography. *J. Discret. Math. Sci. Cryptogr.* **2008**, *11*, 59–65. [CrossRef]
18. Dhaief, Z.S. Encryption of Data Based on Triple Encryption and Affine Algorithm. *Int. J. Adv. Sci. Res. Eng.* **2021**, *7*, 25–34. [CrossRef]
19. Rachmawati, D.; Budiman, M.A. New Approach toward Data Hiding by Using Affine Cipher and Least Significant Bit Algorithm. In Proceedings of the 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), Kuta Bali, Indonesia, 8–10 August 2017; pp. 1–6.
20. Omondi, A.R.; Benjamin Premkumar, A. *Residue Number Systems: Theory and Implementation*; World Scientific: Singapore, 2007; ISBN 9781908979117.
21. Mohan, P.-A. *Residue Number Systems*; Springer International Publishing: Cham, Switzerland, 2016; ISBN 9783319413839.
22. Mezaal, Y.S.; Abdulkareem, S.F. Affine Cipher Cryptanalysis Using Genetic Algorithms. *JP J. Algebra Number Theory Appl.* **2017**, *39*, 785–802. [CrossRef]
23. Mathews, M.M.; Panchami, V.; Ajith, V. Quantum Cryptanalysis of Affine Cipher. *Res. Sq.* **2022**, *14*, 507–519. [CrossRef]
24. Lalitha, K.V.; Sailaja, V. High Performance Adder Using Residue Number System. *J. Mater. Chem. A Mater. Energy Sustain.* **2014**, *5*, 1323–1332.
25. Nykolaychuk, Y.M.; Kasianchuk, M.M.; Yakymenko, I.Z. Theoretical Foundations for the Analytical Computation of Coefficients of Basic Numbers of Krestenson's Transformation. *Cybern. Syst. Anal.* **2014**, *50*, 649–654. [CrossRef]
26. Kasianchuk, M.M.; Yakymenko, I.Z.; Nykolaychuk, Y.M. Symmetric Cryptoalgorithms in the Residue Number System. *Cybern. Syst. Anal.* **2021**, *57*, 329–336. [CrossRef]
27. Kazemi, M.; Naraghi, H.; Golshan, H.M. *On the Affine Ciphers in Cryptography. Communications in Computer and Information Science*; Springer Berlin Heidelberg: Berlin, Germany, 2011; pp. 185–199, ISBN 9783642253263.
28. Hammood, D.A.; Maitham, A. Implementation and Enhancement Affine Cipher of Database. *J. Eng. Sustain. Dev.* **2016**, *20*, 264–276.
29. Babu, S.A. Modification Affine Ciphers Algorithm for Cryptography Password. *Int. J. Res. Sci. Eng.* **2017**, *3*, 346–351.
30. Al-Nuaimy, L. Internal Affine Stream Cipher. *J. Appl. Eng. Technol. Sci. (JAETS)* **2014**, *1*, 1–5.
31. Carlo, J. A Keystream-Based Affine Cipher for Dynamic Encryption. *Int. J. Emerg. Trends Eng. Res.* **2020**, *8*, 2919–2922. [CrossRef]
32. Putra, P.; Sari, C.A.; Isinkaye, F.O. Secure Text Encryption for IoT Communication Using Affine Cipher and Diffie-Hellman Key Distribution on Arduino Atmega2560 IoT Devices. *J. Tek. Inform. (JUTIF)* **2023**, *4*, 849–855. [CrossRef]
33. Lone, M.A.; Qureshi, S. Encryption Scheme for RGB Images Using Chaos and Affine Hill Cipher Technique. *Nonlinear Dyn.* **2023**, *111*, 5919–5939. [CrossRef]
34. Ke, Q.; Liao, Q.-N.; Li, A.-Q.; Gao, R. Digital Image Encryption Algorithm Based on Affine Cipher. In *Advances in Intelligent Systems and Computing*; Springer International Publishing: Cham, Switzerland, 2019; pp. 578–585, ISBN 9783319987750.
35. Soekarta, R.; Sigit, M. Implementation of Affine Group Algebra on Digital Image Security. *Mob. Forensics* **2023**, *4*, 137–146. [CrossRef]
36. Alhassan, M.J.; Hassan, A.; Sani, S.; Alhassan, Y. A Combine Technique of an Affine Cipher and Transposition Cipher. *J. Res. Appl. Math.* **2021**, *7*, 8–12.
37. Budiman, M.A.; Handrizal; Azzahra, S. An Implementation of Rabin-p Cryptosystem and Affine Cipher in a Hybrid Scheme to Secure Text. *J. Phys. Conf. Ser.* **2021**, *1898*, 012042. [CrossRef]
38. Maxrizal, M.; Aniska Prayanti, B.D. Application of Rectangular Matrices: Affine Cipher Using Asymmetric Keys. *CAUCHY* **2019**, *5*, 181–185. [CrossRef]

39. Sundarayya, P.; Vara Prasad, G. A Public Key Cryptosystem Using Affine Hill Cipher under Modulation of Prime Number. *J. Inf. Optim. Sci.* **2019**, *40*, 919–930. [CrossRef]
40. Arroyo, J.-T. An Improved Affine Cipher Using Blum Blum Shub Algorithm. *Int. J. Adv. Trends Comput. Sci. Eng.* **2020**, *9*, 3295–3298. [CrossRef]
41. Shoup, V. *A Computational Introduction to Number Theory and Algebra*; Cambridge University Press: Cambridge, UK, 2009; ISBN 9780521516440.
42. Laia, O.; Zamzami, E.M.; Sutarman; Larosa, F.-N.; Gea, A. Application of Linear Congruent Generator in Affine Cipher Algorithm to Produce Dynamic Encryption. *J. Phys. Conf. Ser.* **2019**, *1361*, 012001. [CrossRef]
43. Stillwell, J. *Elements of Number Theory*; Springer New York: New York, NY, USA, 2003; ISBN 9781441930668.
44. Hardy, G.H.; Wright, E.M.; Silverman, J. *An Introduction to the Theory of Numbers*, 6th ed.; Oxford University Press: London, UK, 2008; ISBN 9780199219865.
45. Srivastava, A.; Mathur, A. The Rabin Cryptosystem & Analysis in Measure of Chinese Reminder Theorem. *Int. J. Sci. Res. Publ.* **2013**, *3*, 1–4.
46. Venturi, D. *Lecture Notes on Algorithmic Number Theory*; Springer: New-York, NY, USA, 2009; 217p, ISSN 1433-8092.
47. Karpinski, M.; Rajba, S.; Zawislak, S.; Warwas, K.; Kasianchuk, M.; Ivasiev, S.; Yakymenko, I. A Method for Decimal Number Recovery from Its Residues Based on the Addition of the Product Modules. In Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Metz, France, 18–21 September 2019; Volume 5, pp. 13–17.
48. Nykolaychuk, Y.M.; Kasianchuk, M.M.; Yakymenko, I.Z. Theoretical Foundations of the Modified Perfect Form of Residue Number System. *Cybern. Syst. Anal.* **2016**, *52*, 219–223. [CrossRef]
49. Bogdanov, A.; Khovratovich, D.; Rechberger, C. Biclique Cryptanalysis of the Full AES. In *Lecture Notes in Computer Science*; Springer Berlin Heidelberg: Berlin, Germany, 2011; pp. 344–371, ISBN 9783642253843.
50. Tiessen, T. *Polytopic Cryptanalysis. Lecture Notes in Computer Science*; Springer Berlin Heidelberg: Berlin, Germany, 2016; pp. 214–239, ISBN 9783662498897.