International Journal on Optimization and Applications

IJOA. Vol. 4, Issue No. 3, Year 2024, http://www.ijoa/ma

IDEA Cipher : Study of methods to strengthen the algorithm – Hybrid Encryption

1st Saad ETTOUAHRI
ENSA – Kenitra
saad.ettouahri@uit.ac.ma

2nd Ayoub ELBAHI
ENSA – Kenitra
ayoub.elbahi3@uit.ac.ma

3rd Raliya OMAR
ENSA – Kenitra
ralia.omarismail@uit.ac.ma

4th Achraf RAHIME
ENSA – Kenitra
rahimeachraf0@gmail.com

5th Abderrahmane BOUHIR
ENSA – Kenitra
abderrahmane.bouhir@uit.ac.ma

***Abstract*— IDEA Cipher Hybridization with AES as a strengthening method.**

The field of cryptography plays a crucial role in ensuring the security and confidentiality of sensitive information transmitted over networks. In recent years, there has been a growing interest in developing more robust and efficient encryption algorithms. This article proposes a novel approach by hybridizing the IDEA (International Data Encryption Algorithm) and AES (Advanced Encryption Standard) algorithms to create a unified ciphering algorithm.

The IDEA algorithm is known for its strong security and efficient performance, while AES is widely recognized as a highly secure and widely adopted encryption standard. By combining the strengths of both algorithms, the proposed hybrid algorithm aims to provide enhanced security and improved performance.

The hybridization process involves integrating the key generation, substitution, permutation, and diffusion techniques of IDEA and AES. This fusion allows for the creation of a unified ciphering algorithm that leverages the best features of both algorithms, resulting in a more robust and secure encryption method.

The article presents a detailed analysis of the hybrid algorithm, including its structure, key generation process, and encryption/decryption procedures. Additionally, the performance of the hybrid algorithm is evaluated through various metrics such as encryption speed, key sensitivity, and resistance to known attacks.

The results of the study demonstrate that the hybrid algorithm achieves a higher level of security compared to individual IDEA and AES algorithms. Furthermore, it exhibits improved performance in terms of encryption/decryption speed and resistance to known attacks.

In conclusion, the hybridization of IDEA and AES as a unified ciphering algorithm offers a promising approach to enhance the security and efficiency of encryption techniques. The proposed algorithm provides a robust solution for protecting sensitive information in various applications, including data transmission over networks and secure storage.

Keywords—IDEA, AES, Encryption, Strengthening.

I. Introduction

The IDEA cipher, also known as the International Data Encryption Algorithm, is a symmetric-key block cipher that was first introduced in 1991. It was designed to provide secure encryption for digital data and has been widely used in various applications such as secure communications, financial transactions, and electronic voting systems.

IDEA uses a block size of 64 bits and a key size of 128 bits. It employs a series of mathematical operations, including modular arithmetic, bit shifting, and exclusive OR (XOR) operations, to transform the plaintext into ciphertext. The cipher is designed to be highly secure and resistant to various types of attacks, including differential and linear cryptanalysis.

One of the strengths of IDEA is its efficient implementation in software and hardware. The algorithm is relatively fast and requires only a small amount of memory and processing power, making it suitable for use in embedded systems and other resource-limited applications.

While IDEA is considered to be a highly secure and effective encryption algorithm, there are ways to reinforce the security further.

The article titled "IDEA Cipher: Study of Methods to Strengthen the Algorithm" focuses on exploring different approaches to enhance the security and strength of the IDEA Cipher. It delves into the analysis of the existing algorithm and proposes methods to reinforce its resistance against potential attacks and vulnerabilities.

The main objective of the article is to evaluate the current state of the IDEA Cipher and identify potential weaknesses that could be exploited by attackers. By studying the algorithm's design and implementation, the article aims to propose practical and effective techniques to strengthen its security.

The article may cover various topics related to strengthening the IDEA Cipher, including:

1. Analysis of the IDEA cipher's mathematical operations and their impact on security.
2. Exploration of potential vulnerabilities and weaknesses in the algorithm.
3. Examination of existing attacks and their effectiveness against the IDEA Cipher.
4. Proposal of new cryptographic techniques or modifications to the algorithm to enhance its security.
5. Evaluation of the proposed enhancements through theoretical analysis and practical experiments.
By studying the methods to strengthen the IDEA cipher, the article aims to contribute to the field of cryptography and provide valuable insights for researchers, practitioners, and developers working with encryption algorithms.

Furthermore, there can be a viable strengthening method using a combination of two already relatively strong algorithms such as IDEA & AES.
Being the most recent and considered being the strongest, AES alone can provide a worldwide acknowledged level of security. Using it in the hybridization of the IDEA algorithm might inquire the use of the same 128 bits key for both algorithms, but will for sure result in an enhanced security measure compared to the implementation of IDEA cipher by itself.

II. Symetric-key algorithm

An algorithm for cryptography that uses the same key for both encryption and decryption is known as a symmetric key algorithm.
The many parties who wish to maintain some confidential information share this key as a shared secret.

Definition: "Take into consideration an encryption scheme that consists of the sets of transformations for encryption and decryption, respectively, $\{E_e: e \in K\}$ and $\{D_d: d \in K\}$, where K is the key space. When it is computationally "easy" to find d from e and to determine e from d for any related encryption/decryption key pair (e, d), the encryption technique is said to be symmetric-key.

The name "symmetric-key" becomes suitable as most realistic symmetric-key encryption methods have e = d. Additional terminology found in the literature include conventional encryption, one-key, private-key, and single-key encryption."

Symmetric key algorithms come in two varieties:
• Stream ciphers: encrypt a message's digits, or generally its bytes, one at a time.
• Block ciphers: these encrypt multiple bits as a single unit, padding the plaintext to be greater than the block size.

III. Grammar and Acronyms

Block ciphers are encryption schemes that encrypt one block at a time by segmenting the plaintext messages to be delivered into strings, or blocks, of a given length **t** over an alphabet **A.**
Definition: An encryption function that specifies a block cipher receives a bit string **P** of length **n**, known as the block size, and a key **K** of bit length k, or the key size, as inputs, and outputs a string **C** with **n** bits. **C** is referred to as the ciphertext, whereas **P** is known as the plaintext. The function $\mathbf{E_k(P)}$ must be an invertible mapping on $\{0, 1\}^n$ for all K.

$$\mathbf{E_k(P) := E(K, P) : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n}$$

The inverse for E is defined as a function

$$\mathbf{E_k^{-1}(C) := D_k(C) = D(K, C) : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n}$$

Taking a key K and a ciphertext C to return a plaintext value P, such that

$$\mathbf{\forall K : D_k(E_k(P)) = P}$$

$\mathbf{E_k}$ is a permutation (a bijective mapping) over the set of input blocks for each key K. From the possible set of $(2^n)!$ permutations, each key chooses one.

Statistical investigation suggests that a block cipher that has an excessively small block size **n** could be subject to attacks. A basic frequency analysis of the ciphertext block is an example of such attack. But, using a blocksize **n** that is too high could cause problems because many ciphers' implementation complexity increases quickly with block size.

IV. Operation Modes

An algorithm known as a mode of operation encrypts messages of any length with a block cipher to ensure confidentiality or authenticity. Only one fixed-length group of bits can be securely transformed (encrypted or decrypted) using a block cipher on its own.
Known as a block. A mode of operation explains how to safely convert amounts of data bigger than a block by continually using a cipher's single-block operation.

- **ECB mode: Electronic Codebook**

ECB is the most simple encryption mode, it divides the text into two blocks and encrypts each one of them separately.
Input: *k-bit* key *K*;

n-bit plaintext blocks $x_1, \ldots, x_t$.

Produce ciphertext blocks $c_1, \ldots, c_t$; decrypt to recover plaintext.

1. Encryption: *for* $1 \le i \le t$, $c_j \leftarrow E_k(x_j)$.
2. Decryption: *for* $1 \le i \le t$, $x_j \leftarrow E_k^{-1}(c_j)$.

The problem with this algorithm is that the block of identical text are encrypted into identical ciphertext blocks, which creates noticeable patterns.

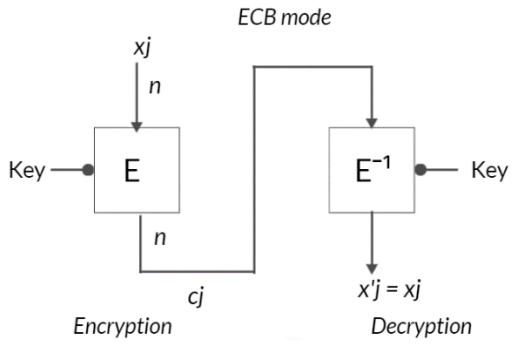

Figure 4.1

- **CBC mode: Cipher Block Chaining**

In this operation mode, each block of plaintext along with the previous Ciphertext that has been encrypted are subject to an XOR operation, creating a dependency of each ciphertext block on all plaintext blocks previously processed.

Input: *k-bit* key *K*;

n-bit IV;

n-bit plaintext blocks $x_1, \ldots, x_t$.

Produce ciphertext blocks $c_1, \ldots, c_t$; decrypt to recover plaintext.

1. Encryption $c_0 \leftarrow IV$. *For* $1 \le j \le t$, $c_j \leftarrow E_k(c_{j-1} \oplus x_j)$.
2. Decryption $c_0 \leftarrow IV$. *For* $1 \le j \le t$, $x_j \leftarrow c_{j-1} \oplus E_k^{-1}(c_j)$.

The drawback can be the fact that the encryption method is sequential creating a padding of a message that is a multiple of the cipher block size.

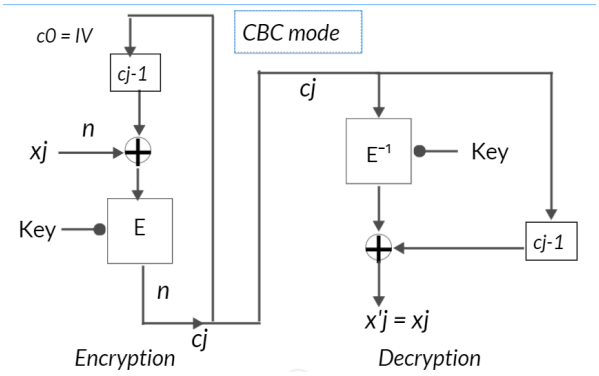

Figure 4.2

- **CFB mode: Cipher Feedback**

CFB mode is required for some applications that are delay sensitive and which require that *r-bit* plaintext units be encrypted and transmitted without delay for a fixed $r < n$ ($r = 1$ or $r = 8$).

Input: *k-bit* key *K;*

n-bit IV;

r-bit plaintext blocks $x_1, \ldots, x_u$ $(1 \le r \le n)$.

Produce ciphertext blocks $c_1, \ldots, c_t$; decrypt to recover plaintext.

1. Encryption: $I_1 \leftarrow IV$. (I_j is the input value in a shift register). *For* $1 \le j \le u$:
   (a) $O_j \leftarrow E_k(x_j)$. (Processes the output block cipher).
   (b) $t_j \leftarrow$ the r leftmost bits of O_j. (Assume the leftmost is identified as bit 1).
   (c) $c_j \leftarrow x_j \oplus t_j$. (Transmit the *r-bit* ciphertext block c_j).
   (d) $I_{j+1} \leftarrow 2^r . I_j + c_j \bmod 2^n$. (Shift c_j into right end of shift register).
2. Decryption: $I_1 \leftarrow IV$. *For* $1 \le j \le u$, upon receiving cj:
   $xj \leftarrow cj \oplus tj$, where tj , Oj and Ij are processed as above.

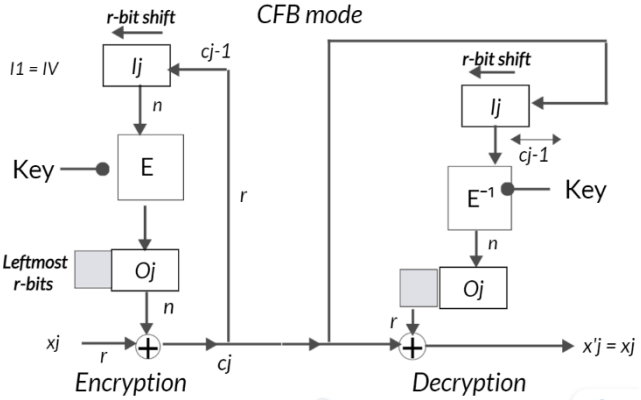

Figure 4.3

- **OFB mode: Output Feedback**

An asynchronous stream cipher can be created from a block cipher using the Output Feedback (OFB) mode. The ciphertext is obtained by XORing the keystream blocks that are produced with the plaintext blocks. Similar to other types of stream ciphers,

When a bit in the ciphertext is flipped, the corresponding bit in the plaintext is also flipped. Many error-correcting codes can operate normally even when applied before encryption thanks to this trait.

Input: *k-bit key K;*

n-bit IV ;

r-bit plaintext blocks $x_1, \ldots, x_u$ $(1 \le r \le n)$.

Produce ciphertext blocks $c_1, \ldots, c_u$; decrypt to recover plaintext.

1. Encryption: $I_1 \leftarrow IV$. *For* $1 \le j \le u$, given plaintext block xj :
   (a) $Oj \leftarrow E_k(I_j)$. (Processes the output block cipher).
   (b) $t_j \leftarrow$ the r leftmost bits of O_j. (Assume the leftmost is identified as bit 1).
   (c) $c_j \leftarrow x_j \oplus t_j$. (Transmit the *r-bit* ciphertext block c_j).
   (d) $I_j+1 \leftarrow O_j$. (Update the block cipher input for the next block).
2. Decryption: $I_1 \leftarrow IV$. *For* $1 \le j \le u$, upon receiving $c_j : x_j \leftarrow c_j \oplus t_j$, where t_j , O_j and I_j are computed as above

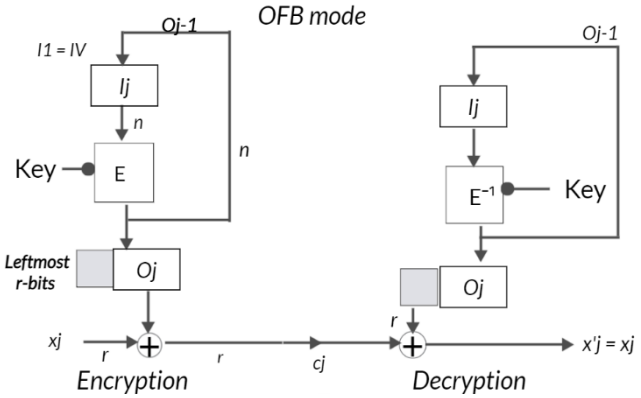

Figure 4.4

V. Methodology Overview : Hybrid Encryption-IDEA & AES

In this methodology, we will explain the proceedings of ciphering with the IDEA (International Data Encryption Algorithm) and then using the same 128 bits key to cipher with AES (Advanced Encryption Standard). Both IDEA and AES are symmetric encryption algorithms that use the same key for encryption and decryption.

1. IDEA Algorithm**:**
The IDEA algorithm is a block cipher that operates on 64-bit blocks of data.
It uses a 128-bit key for encryption and decryption. The following steps outline the ciphering process with IDEA:
a. Key Generation: Generate a 128-bit key that will be used for both IDEA and AES encryption.
b. Data Preparation: Divide the plaintext into 64-bit blocks.
c. Initial Permutation: Perform an initial permutation on each 64-bit block of plaintext.
d. Round Function: Perform a series of 8 rounds, each consisting of the following steps:
i. Substitution: Substitute bytes using a substitution table.
ii. Permutation: Permute the bytes within each 16-bit half-block.
c. Mixing: Mix the bytes using modular addition and multiplication operations.

e. Final Permutation: Perform a final permutation on each 64-bit block of ciphertext.
f. Output: The resulting ciphertext is obtained.

2. AES Algorithm**:**
The AES algorithm is a block cipher that operates on 128-bit blocks of data. It also uses a 128-bit key for encryption and decryption. The following steps outline the ciphering process with AES:
a. Key Expansion: Expand the 128-bit key into a set of round keys using a key schedule.
b. Data Preparation: Divide the plaintext into 128-bit blocks.
c. Initial Round: Perform an initial round of transformations on each 128-bit block of plaintext using the round key.
d. Rounds: Perform a series of 10, 12, or 14 rounds (depending on the key size), each consisting of the following steps:
i. SubBytes: Substitute bytes using a substitution table.
ii. ShiftRows: Shift the rows of the state matrix.
iii. MixColumns: Mix the columns of the state matrix.
iv. AddRoundKey: XOR the state matrix with the round key.

e. Final Round: Perform a final round of transformations on each 128-bit block of ciphertext using the round key.

VI. IDEA Cipher

With a 128-bit input key K, the IDEA cipher encrypts 64-bit plaintext blocks to 64-bit ciphertext blocks. It consists of 8 computationally identical rounds, sort of like a novel generalization of the Feistel structure, followed by a transformation of the output. In round r, a 64-bit input X is converted into an output of four 16-bit blocks, which are eight inputs for the following round, using six 16-bit subkeys $K_i^{(r)}$, $1 \le i \le 6$.
After the round 8 output is entered into the output transformation, the final ciphertext, Y = (Y1, Y2, Y3, Y4), is produced by using four further subkeys, $K_i^{(9)}$, $1 \le i \le 4$. The same algorithm is used for both encryption and decryption.
K serves as the source of all subkeys.
In IDEA, combining operations from three distinct algebraic groups of 2^n elements is a prominent design idea.

The group operations that correspond to them on sub-blocks a and b with bitlength n = 16 are as follows:

- a $\oplus$ b: bitwise XOR.
- a [+] b: addition *mod* 2^n : (a + b) AND 0xFFFF.
- a [*] b: (modified) multiplication *mod* $2^n + 1$, with 0 $\in Z_{2^n}$ associated with $2^n \in Z_{2^{n+1}}$.

In each round, the sequence of events is as follows:

1. Multiply X1 and the first subkey.
2. Add X2 and the second subkey.
3. Add X3 and the third subkey.
4. Multiply X4 and the fourth subkey.
5. XOR the results of steps (1) and (3).
6. XOR the results of steps (2) and (4).
7. Multiply the results of step (5) with the fifth subkey.
8. Add the results of steps (6) and (7).
9. Multiply the results of step (8) with the sixth subkey.
10. Add the results of steps (7) and (9).
11. XOR the results of steps (1) and (9).
12. XOR the results of steps (3) and (9).
13. XOR the results of steps (2) and (10).
14. XOR the results of steps (4) and (10).

The output of the round is the four fragment blocks that are the results of steps (11), (12), (13), and (14).
Swap the two internal blocks (except for the last round) and that is the input to the next round.
After the eighth round, there is a final output transformation:

1. Multiply X1 and the first subkey.
2. Add X2 and the second subkey.
3. Add X3 and the third subkey.

4. Multiply X4 and the fourth subkey.

Finally, the four sub-blocks are reattached to produce the ciphertext.
The following chart resumes the IDEA cipher encryption process:

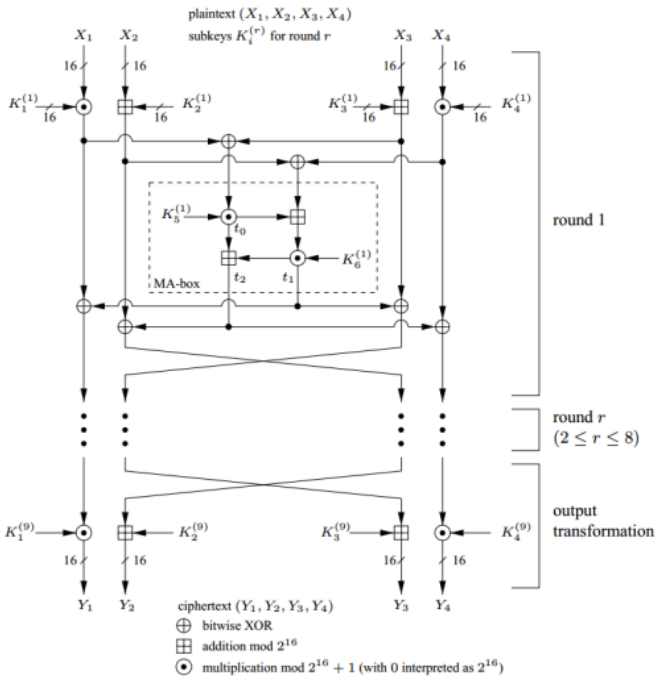

Figure 6.1

VII. AES CIPHER

AES (Advanced Encryption Standard), is an algorithm for block encryption standardized by NIST in 2001, in order to replace DES and 3DES.

The size of an AES block is 128 bits, whereas the size of the encryption key can be 128, 192 or 256. But for the sake of our research purposes we're going to focus on the 128 bits key encryption that we will previously use in the IDEA algorithm.

Block cipher algorithms should enable encryption of the plaintext with size which is different from the defined size of one block as well. We can use some algorithms for padding block when the plaintext is not enough a block, like PKCS5 or PKCS7, it also can defend against PA attack, if we use ECB or CBC mode. Or we can use the mode of AES which support a stream of plaintext, like CFB, OFB, CTR mode.

In PKCS5Padding, arbitrary data lengths are accepted; the ciphertext will be padded to a multiple of 8 bytes, as described in PKCS#5. The decryption process will remove the padding from the data so that the correct plaintext is returned. This Cipher will accept a javax.

PKCS#5 padding is identical to PKCS#7 padding, except that it has only been defined for block ciphers that use a 64-bit (8-byte) block size. In practice, the two can be used interchangeably. The maximum block size is 255, as it is the biggest number a byte can contain.

The five modes of AES.

- ECB mode: Electronic Code Book mode
- CBC mode: Cipher Block Chaining mode
- CFB mode: Cipher FeedBack mode
- OFB mode: Output FeedBack mode
- CTR mode: Counter mode

As previously detailed in the IDEA chapter, these block cipher operation modes guarantee a secure encryption.
Starting from the top of the list as we go down, their efficiency and complexity goes up for more security.

The attack modes that are considered subject to counter-measures in AES

- PA: Padding attack
- CPA: Chosen Plaintext Attack
- CCA: Chosen Ci

The algorithm go through multiple rounds of substitution and permutation for each block, then concatenate everything. There are multiple modes of operation as mentioned in the last paragraph. In this article we are going to focus on the ECB mode (the simplest one)

The ECB (Electronic Code Book) mode is the simplest of all. Due to obvious weaknesses, it is generally not recommended, it is used here for demonstration purposes only.

The length of an AES block, 128 bytes, is the division of the plaintext into blocks. In order to make the data equal to the block length, the ECB mode must pad the data. Subsequently, each block will undergo encryption using an identical key and technique. Thus, we will have the same ciphertext if we encrypt the same plaintext. Thus, this approach carries a considerable risk. There is a one-to-one correlation between the plaintext and ciphertext blocks. We can encrypt and decrypt the data simultaneously since the encryption and decryption processes are independent. Furthermore, breaking one block of plaintext or ciphertext won't impact other blocks.

An attack can be launched even if they are unable to obtain the plaintext thanks to an ECB feature.

For example, if we encrypt the data about our bank account, like this: The ciphertext: C1 (account number): 21 33 4e 5a 35 44 90 4b, and C2 (The password): 67 78 45 22 aa cb d1 e5 the data can be copied in C1 to C2. Then the system can be logged in with the account as with the password which is easier to get.

Plaintext

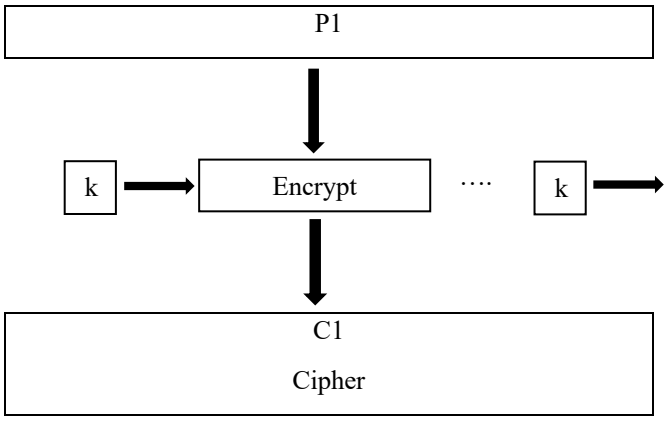

Figure 7.1

The AES consist of four basic operations that are repeated over N rounds. These four operations are ADDING, SUBSTITUTING, SHIFTING, and MIXING, being done while the key is expanded with different bitwise rotations to maximize its use and for it to be sufficient for the completion of the encryption.

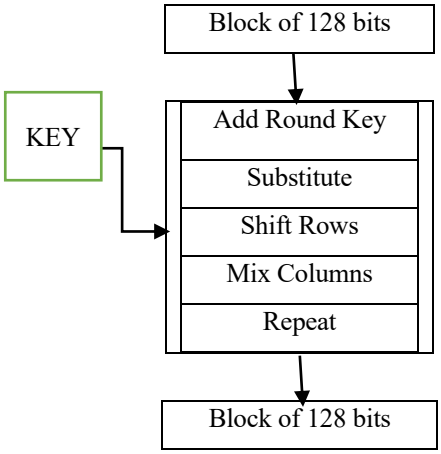

Figure 7.2

A more detailed representation of the AES algorithm rounds would give us the following diagram:

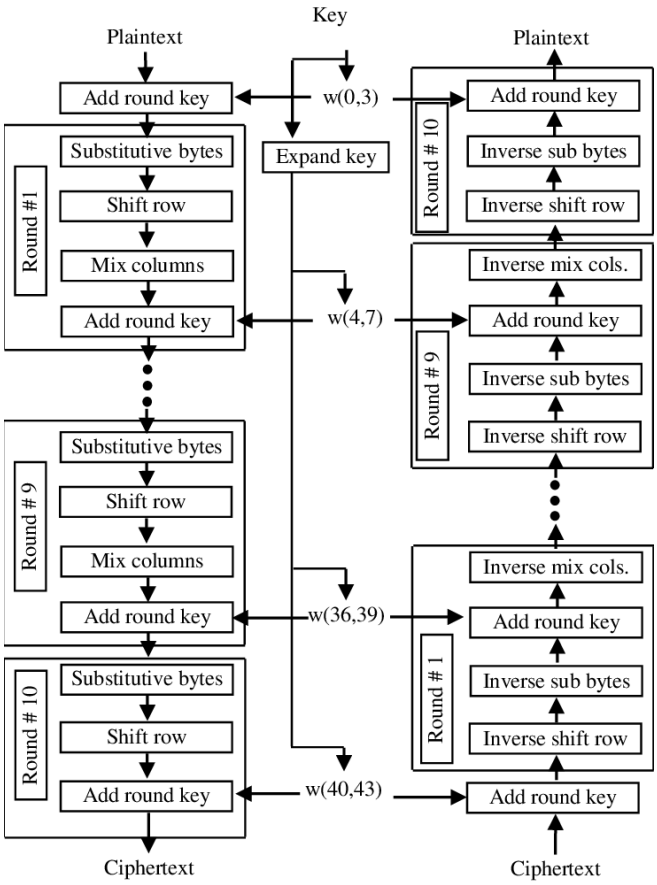

Figure 7.3

VIII. Final Output

The final output of the hybridization of IDEA and AES cipher using a 128-bit encryption key would result in a secure and robust encryption of the data.

1. **Hybrid Encryption Approach:**
   - The hybrid encryption combines the strengths of both IDEA and AES ciphers to ensure a high level of security.
   - In the hybrid approach, a secret key is generated, and the data is encrypted using IDEA then AES with the same secret key.
   - The encrypted secret key and the encrypted data are sent together to the secure recipient of choice.

2. **Final Output:**
   - The final output of the hybridization of IDEA and AES cipher using a 128-bit encryption key is a combination of the encrypted secret key and the encrypted data.
   - The encrypted data is obtained by encrypting the original data using IDEA & AES and the secret key.
   - These two components are combined and sent to the recipient as a single package.

3. **Security Considerations**:
   - The hybrid approach provides a strong level of security by combining the strengths of both IDEA and AES ciphers.

- IDEA and AES are widely recognized and trusted encryption algorithms, with AES being particularly robust and resistant to brute-force attacks
- The use of a 128-bit encryption key ensures a high level of security for the encrypted data.
- The encryption of the secret key using RSA will surely add an additional layer of security, ensuring that only the intended recipient can decrypt the data.

In summary, the final output of the hybridization of IDEA and AES cipher using a 128-bit encryption key is a combination of the encrypted secret key and the encrypted data. This approach provides a strong level of security and ensures the confidentiality of the transmitted data.

The following diagram resumes the process of the combination of both algorithms:

Figure 8.1

IX. Conclusion

The hybridization of two Cipher algorithms is one of the main solutions that can provide stronger security and produces ciphertexts that are more complex and approach a sense of immunity to the majority of external threats.

IDEA cipher is one of the strongest encryption algorithms available, combining it with AES, which is the most recent and also the strongest to date, insures a stronger security stance when it comes to protecting sensitive data and communications.

Using the IDEA cipher output as an input for the AES cipher can provide an additional layer of security and solidify the encryption process.

As detailed in the article, the IDEA cipher is a symmetric key block cipher algorithm that operates with a block size of 64 bits and a key length of 128 bits. It is known for its strong encryption capabilities and has been widely used in various applications. However, to further enhance the security of the encryption, it is possible to use the output of the IDEA cipher as the input for the AES cipher also known as Rijndael, is another popular symmetric key block cipher algorithm. It operates with a block size of 128 bits and supports key lengths of 128, 160, 192, 224, and 256 bits. AES is highly secure and widely used in secure communication protocols such as TLS and SSL.

By using the output of the IDEA cipher as the input for the AES cipher, we can leverage the strengths of both algorithms and create a more robust encryption scheme. This approach adds an extra layer of complexity and makes it even more difficult for an attacker to decrypt the data without the proper keys.

To implement this process, the output of the IDEA cipher can be treated as the plaintext input for the AES cipher. The AES cipher will then encrypt this input using its own encryption algorithm and produce the final ciphertext. This combined encryption process can provide a higher level of security and make it more challenging for unauthorized individuals to access the original data.

It is important to note that the security of the encryption scheme also depends on the key management and secure transmission of the keys between the sender and the receiver (which can give the introduction to another security measure in the RSA key encryption solution). Proper key generation, storage, and exchange protocols should be implemented to ensure the overall security of the system.

X. Grammar and Acronyms

A. *Abbreviations and Acronyms*

IDEA: *International Data Encryption Algorithm*

AES: *Advanced Encryption Standard*

DES /3DES: *Data Encryption Standard*

ECB: *Electronic Code Book mode*

CBC: *Cipher Block Chaining mode*

CFB: *Cipher Feedback mode*

OFB: *Output Feedback mode*

CTR: *Counter mode*

PKCS5/S7: *Public Key Cryptography Standard(Standard 5&7)*

TLS: *Transport Layer Security*

SSL: *Secure Socket Layer*

B. *Units*

Bits.
Bytes (= 8bits).

C. *Figures and Tables*

Figure 4.1 – ECB operation mode
Figure 4.2 – CBC operation mode

Figure 4.3 – CFB operation mode
Figure 4.4 – OFB operation mode
Figure 6.1 – IDEA Cipher rounds diagram
Figure 7.1 – ECB simplified diagram
Figure 7.2 – AES simplified diagram
Figure 7.3 – AES Cipher rounds diagram
Figure 8.1. – IDEA & AES implementation diagram

REFERENCES

[1] Z. Alimzhanova, M. Skublewska-Paszkowska, D. Nazarbayev Structural Periodicity of the Substitution Box in the CBC Mode of Operation: Experiment and study, in : IEEE Access, Vol 11, 2023.

[2] K. Patula, P. Gundabathina, Implementation of High Speed Modulo (2^n+1) Multiplier for IDEA Cipher, Scopus, Procedia Computer Science, Vol 171, Pages 2016- 2022, 2020.

[3] B. Schneier, The IDEA encryption algorithm, Journal USA, 18(13), Page 50, 1993.

[4] S. Basu, International Data Encryption Algorithm (IDEA) – A Typical Illustration, ResearchGate, Journal of Global Research In Computer Science, Volume 2, No.7, 2011.

[5] W. Meier, On the Security of IDEA block Cipher, Scopus, Lecture Notes in Computer Science, LNCS, volume 765, pages 371 – 385, 1994.

[6] F. Nuraeni, Y.H. Agustin, The Implementasi Caesar Cipher & Advanced Encryption Standar (AES) Papa Pengamanan Data Pajak Bumi Bangunan, ResearchGate, Jurnal llmiah Matrik 22(2), Pages 187-194, 2020.

[7] N.H.M. Ali, A.M.S. Rahma, A.M. Jaber, S. Yousef, A Byte-Oriented Multi Keys Shift Rows Encryption and Decryption Cipher Processes in Modified AES, ResearchGate, International Journal of Scientific and Engineering Research 5(4), Pages 953-955, 2014.