Journal link: http://kmf-publishers.com/tatscr/

Theoretical and Applied Technological Science Review

Volume: 3 Issue: 1 Year: 2025
ISSN-2958-7824

KMF PUBLISHERS

www.kmf-publishers.com/tatscr/

Received: 10 January 2025 | Revised: 10 February 2025 | Accepted: 25 February 2025

DOI: https://doi.org/10.5281/zenodo.14965619

Article

Hybrid Cryptographic Algorithm Based on Aes, Rsa and Walsh Transform

Sergo A. Episkoposian[1,2]; Svetlana A. Grigoryan[1]

[1]National Polytechnic University of Armenia
Armenia
[2]Yerevan State University
Armenia

Correspondence
Svetlana Grigoryan
PhD Student
Institute of Information and Telecommunication Technologies and Electronics,
National Polytechnic University of Armenia,
Armenia.
Email: svetlanagrigoryan65@gmail.com , svetlanagrigoryanmt840@polytechnic.am

ABSTRACT

This paper proposes a hybrid cryptographic algorithm combining the RSA algorithm and the Walsh transform to enhance data security and performance. The hybrid approach leverages the strengths of asymmetric encryption (RSA) and the computational efficiency of the Walsh Transform. RSA ensures robust key exchange security, and the Walsh Transform introduces lightweight linear operations for preprocessing data. Experimental results demonstrate the hybrid algorithm's superior performance, achieving robust encryption with a notable trade-off between speed and security. Comparative analyses against standalone RSA and Walsh methods reveal the hybrid algorithm's competitive edge, particularly in high-security scenarios. Key challenges, including computational overhead and implementation complexity, are also discussed, along with future improvements. Our scientific group has begun exploring the applications of the Walsh transform in both textual and audio information processing. Specifically, this includes the development of new hybrid algorithms, such as AES and RSA. This work will present the hybrid algorithm combining the RSA algorithm and the Walsh transform.

Keywords: Hybrid Cryptography, RSA, Walsh Transform, Data Security, Encryption Performance.

CRYPTOGRAPHIC ALGORITHMS

Primitive cryptographic techniques existed in ancient times, with early civilizations using cryptography to some degree. Symbol replacement, the most basic form, appears in ancient Egyptian and Mesopotamian writings. The earliest example, found in the tomb of Egyptian noble Khnumhotep II (3,900 years ago), was used to enhance linguistic appeal.

Around 3,500 years ago, a Mesopotamian scribe used cryptography to conceal a formula for pottery glaze on clay tablets. By antiquity, cryptography protected military information, as seen in Sparta's cylinder encryption and coded messages used by Indian spies in the 2nd century BC. The Romans developed the Caesar cipher, shifting letters in the Latin alphabet to encode messages.

Cryptography hides messages from unintended recipients by scrambling data via algorithms. The first proven use dates back to 1900 B.C. in Ancient Egypt, where irregular hieroglyphs were used in a document. Modern cryptography algorithms, offering greater security, are divided into secret key, public key, and hash functions. This paper focuses on public key cryptography (PKC) algorithms.

- *Hash Functions:* Uses a mathematical transformation to irreversibly "encrypt" information, providing a digital fingerprint. Primarily used for message integrity.
- *Secret Key Cryptography (SKC):* This type of encryption uses a single key for both encryption and decryption. It is also called symmetric encryption. It is primarily used for privacy and confidentiality.
- *Public Key Cryptography (PKC):* This type of encryption uses one key for encryption and another for decryption. It is also called asymmetric encryption. PKC is primarily used for authentication, non-repudiation, and key exchange.

Public key cryptography is considered the most significant development in cryptography in the last 300-400 years. Modern PKC was first publicly described in 1976 by Stanford professor Martin Hellman and graduate student Whitfield Diffie. Their paper introduced a two-key system enabling secure communication over non-secure channels without sharing a secret key.

Known as asymmetric encryption, PKC uses dual keys: the recipient's public key encrypts the message, and the private key decrypts it, eliminating the need for a shared secret key as in symmetric cryptography [5]. PKC is widely used for authentication, non-repudiation, and key exchange [10]. The most widely used PKC algorithm today is Rivest, Shamir, and Adleman (RSA) [5], [9], [10]. Launched in 1977 and named after inventors Ron Rivest, Adi Shamir, and Leonard Adleman, RSA comprises two algorithms: key generation, producing public and private keys, and RSA function evaluation for encryption and decryption.

RSA Security Inc. held the RSA algorithm patent from 1983 until its expiry in 2000 but released its claim two weeks before the expiry, making RSA public domain. This prompted competitor Baltimore Technologies to offer a free version of its Keytools developer toolkit, which previously cost $10,000 to $20,000 [4]. During the patent years, RSA had over 800 licensed customers and was used in more than 1,000 applications, including Microsoft Windows, Netscape Navigator, Lotus Notes, PGP, Cisco routers, credit cards, iPhone text messaging, and web connections, securing trillions of transactions [4], [7], [11].

WALSH FUNCTIONS AND HADAMARD MATRIX

Walsh Functions

To define the Walsh function, it is necessary to define the Rademacher function.

Definition 1. The Rademacher system is defined as

$$r_0(x) = \begin{cases} 1, x \in \left[0, \frac{1}{2}\right), \\ -1, x \in \left(\frac{1}{2}, 1\right], \end{cases}$$

$$r_0(x+1) = r_0(x),\ r_k(x) = r_0(2^k x), \qquad k = 1,2,\ldots,$$

i.e. to find the r_k Rademacher function, the interval $[0;1)$ is divided into 2^{k+1} equal subintervals, where the function $r_k(x)$ alternates between +1 and -1 successively.

The Walsh system is defined as all possible finite products of Rademacher functions. More precisely, we define the Walsh system as follows:

Definition 2. $W_0(x) \equiv 1$. Let n be any natural number, represented as $n = \sum_{s=1}^{k} 2^{m_s}$, $m_1 > m_2 > \cdots > m_s$. The n-th Walsh function will be defined as follows:

$$W_n(x) = \prod_{s=1}^{n} r_{m_s}(x)$$

Properties of Walsh functions

Orthogonality: The Walsh functions $W_n(x)$ are orthogonal on the interval [-1, 1], meaning that for all $m, n \in \mathbb{N}$, the following holds:

$$\int_0^1 w_m(x) w_n(x) dx = \begin{cases} 1, & when\ m = n \\ 0, & when\ m \neq n \end{cases}$$

Unit Energy: Each function $w_k(x)$ has the norm L_2 equal to 1:

$$\|w_k(x)\|_2 = \sqrt{\int_0^1 w_k^2(x) dx} = 1$$

Completeness: Any function $f(x) \in L_2[0,1)$ can be expanded in a Walsh series:

$$f(x) = \sum_{k=0}^{\infty} c_k w_k(x)$$

where the coefficients c_k are defined as:

$$c_k = \int_0^1 f(x) w_k(x) dx$$

Hadamard Matrix

We can get Walsh functions using the Hadamard Matrix.

A Hadamard matrix is a square matrix whose elements are either +1 or -1 and whose rows are mutually orthogonal. This means that the scalar product of any two distinct rows is zero. A Hadamard matrix of order n (where n equals powers of 2) is formed as:

$$H_n = \begin{bmatrix} H_{n/2} & H_{n/2} \\ H_{n/2} & -H_{n/2} \end{bmatrix},$$

where $H_1 = [1]$.

For example:

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

Walsh functions are essentially the rows of the Hadamard matrix rearranged in order. The Hadamard matrix provides a simple, recursive structure for deriving Walsh functions[1].

WALSH-HADAMARD TRANSFORM

Consider a text represented as a data vector $x \in \mathbb{R}^N$, where $N = 2^n$. The Walsh-Hadamard transform for encryption is performed as follows:

1. Direct transformation:
$$y = H_n x.$$
2. Inverse transformation: to decipher the data, the transposed matrix is used:
$$x = \frac{1}{N} H_n^T y.$$

Let us denote e as the vector of noise superimposed on the encrypted data:
$$\tilde{y} = y + e$$
When reversed, the recovered data will be:
$$\tilde{x} = \frac{1}{N} H_n^T \tilde{y} = x + \frac{1}{N} H_n^T e$$
If the noise vector e is small in the L_2 norm, the reconstruction error will also be small:
$$\|\tilde{x} - x\|_2 = \frac{1}{N} \|H_n^T e\|_2 \leq \frac{1}{N} \|H_n\|_2 \|e\|_2 = \|e\|_2$$

WALSH FUNCTIONS AND RSA ALGORITHM IN CRYPTOGRAPHY

Walsh Transform

The Walsh transform does not require asymmetric keys; instead, it uses a shared key on both the sender and receiver sides for data transformation and its reversal. This makes it simpler and less computationally intensive than asymmetric methods.

Encryption involves representing the original data as a vector, applying the Walsh transform by multiplying the Walsh matrix with the data vector, and obtaining Walsh coefficients as encrypted data ([2], [3]).

Pre-processing, such as formatting, normalization, or adding padding, is performed before the Walsh transform. The transform decomposes the original data into coefficients representing each Walsh basis function's contribution to the data. This process is mathematically represented as follows:

Suppose we have original data in the form of a vector $x = [x_1, x_2, \ldots, x_n]$, where n is the dimension of the data. The Walsh transform is applied to the vector x using a transformation matrix H of dimension, and the result is a vector of coefficients $c = [c_1, c_2, \ldots, c_n]$:
$$c = H * x.$$
After the Walsh transform, we obtain a vector of coefficients.

To decrypt the data, it is necessary to reconstruct the original data based on the selected significant coefficients. This step can be represented as follows: Suppose we have a vector of significant coefficients $c' = [c_1', c_2', \ldots, c_k']$, where k is the number of significant coefficients. Data recovery is performed using the inverse Walsh transform, using only the selected significant coefficients:
$$x' = H^T * c',$$
where H^T is the transposed matrix of the Walsh transform.

The description of the steps can be found in *Figure 1*.

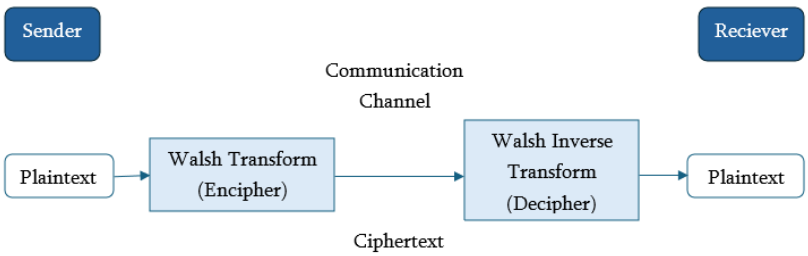

Figure 1: *Walsh Transform*

RSA Algorithm

RSA (Rivest Shamir Adleman) is the most widely used public key encryption algorithm. It operates with integers modulo $n = p \cdot q$, where p and q are prime numbers. RSA requires keys of at least 1024 bits for adequate security, with 2048-bit keys providing optimal security. RSA is primarily used for secure communication channels and authentication with identity service providers. While too slow for encrypting large data volumes, it is commonly used for key distribution. The following steps are used to generate public and private keys in RSA. [8]

Generate a public key (e) and a private key (d) by choosing two very large prime numbers p and q, each around 256 bits (75 digits).

Multiply p and q, and let the result be n. $n = p \cdot q$. Note that the factors p and q remain secret and n is public. Even if n is known, it is not practically feasible to get back p and q since factorising a very large number is computationally infeasible.

Generate a public key by choosing a number e, which is relatively prime to the "totient" function $\varphi(n) = (p-1)(q-1)$.

Generate a private key by choosing a number d, which is the multiplicative inverse of e mod $\varphi(n)$. Note that the public key is $< e, n >$, which is known to everyone, and the private key is $< d, n >$ which is known only to the person who has to decrypt or sign the message.

Encrypt a message m $(< n)$, raise m to the power e under modulo n. The result is the cipher text (c).

$$c = m^e \ mod \ n$$

Decrypt the cipher text, raise the cipher to the power d under modulo n.

$$m = c^d \ mod \ n.$$

Sign the message by encrypting it with the private key $< d, n >$ and decrypting it with the public key $< e, n >$. [6]

The description of the RSA algorithm can be found in *Figure 2.*

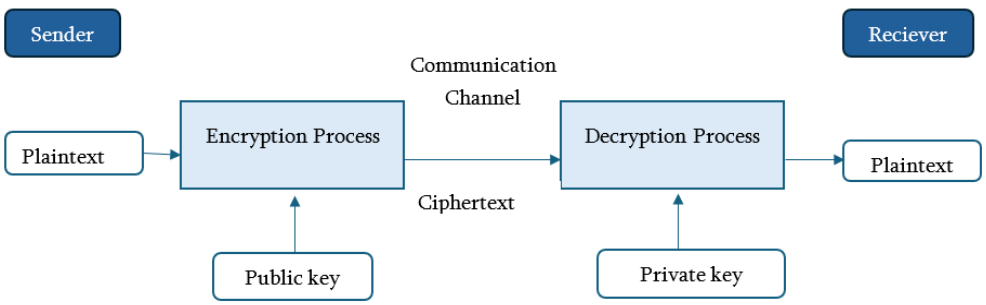

Figure 2: RSA Encryption

The strengths and weaknesses of the RSA algorithm are detailed in *Table 1*.

Table 1. Strengths and Weaknesses of the RSA Algorithm

Strengths of RSA	Weaknesses of RSA
Provides confidentiality, integrity, authenticity, and non-reputability of data.	Key generation requires two large prime numbers, making it computationally intensive.
Difficulty factorising large integers while ensuring high security.	Vulnerable to attacks if weak keys are generated or if insufficient key length is used.
Allows dual-key functionality: encrypting with one key and decrypting with the other.	Slower than other algorithms (e.g., ECC) in key generation, especially on resource-limited devices.
Supported and used widely in protocols like SSL/TLS, smart cards, and email systems applications.	Large key sizes (e.g., 2048-bit) can demand high computational resources and battery power on mobile devices.
It is compatible with many cryptographic standards and is widely adopted across industries.	Susceptible to advances in cryptanalysis and quantum computing, potentially compromising its effectiveness.
Modulus length increases security but also raises processing time and resource demands.	Some past implementations (e.g., with small key lengths) have been successfully attacked.

Walsh Transformation combined with RSA Algorithm

Here, we discuss an encryption algorithm that combines the Walsh Transform and RSA encryption algorithms. This combination enhances data security, suggesting a multi-layered approach.

In the presented algorithm, the Walsh transform acts as a preprocessing step, transforming data with Walsh functions. This step adds a layer of complexity, making it harder for attackers to decipher the data. The next step of the algorithm is using the RSA encryption algorithm, which provides a high level of security for protecting sensitive information.

Combining the two techniques (*Figure 3*) strengthens the data protection and reduces the likelihood of data breaches.

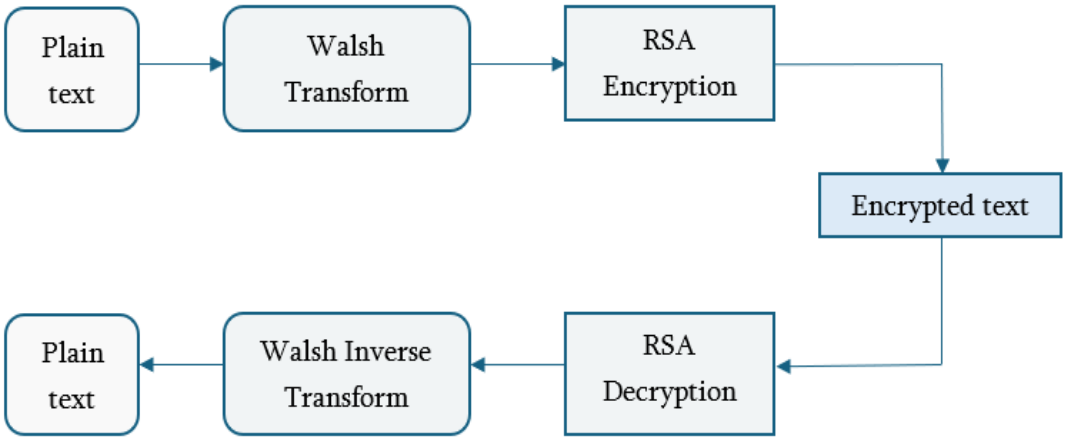

Figure 3: Walsh transform combined with RSA algorithm

With the help of AI, we developed a criteria table (*Table 2*) to compare the algorithms discussed in this article.

Table 2. Comparison of Algorithms

(Continued on next page)

Criteria	Walsh Transform	RSA Algorithm	Walsh + RSA
Key Strength	No key involved, deterministic	Strong, relies on large prime numbers and modular arithmetic	Inherits RSA's security, Walsh adds minor obfuscation.
Resistance to Attacks	Vulnerable to basic cryptanalysis	Very strong against brute-force (if keys are long enough)	Strong due to RSA's contribution. Walsh adds a layer of complexity.
Confidentiality	Minimal, as Walsh is not inherently cryptographic	High, suitable for secure communications	High, with added obfuscation from Walsh.
Encryption Time	Fast, as it's a mathematical transform	Slower, especially for large data sizes	Moderate: Walsh is fast, but RSA dominates encryption time.
Decryption Time	Fast	Slower (requires modular exponentiation)	Moderate; RSA dominates. Walsh inverse is computationally cheap.

Table 2. Comparison of Algorithms (Continued)

Scalability	Scales well with data size	Struggles with very large data due to block size	Similar to RSA, scalability is limited by RSA.
Resource Usage	Low (simple operations)	High (modular arithmetic and key management)	Moderate: Walsh has minimal overhead compared to RSA.
Energy Consumption	Low	High	Moderate: Walsh's energy impact is negligible.
Implementation	Easy to implement	Complex (key generation, padding schemes)	Moderate: integrates both algorithms with some coordination.
Key Management	N/A	Key pair management essential	Relies on RSA's key infrastructure.
Ciphertext Size	Same as input, may pad to power of 2	Larger due to encryption overhead	Larger than input size (RSA overhead + Walsh padding).
Support for Compression	None	None	None; Walsh padding may hinder compression.
Error Propagation	Localized; single error affects part of the result	Localized; single error affects decrypted block	Localized; same as RSA.
Use Suitability	Signal processing, simple obfuscation	Secure key exchange, digital signatures	Secure communications with added obfuscation.
Hybrid Use	Limited	Often paired with symmetric cryptography	Complements RSA with Walsh for pre-encryption obfuscation.
Encryption Time	Minimal (< 1 ms)	Longer for large keys (10s of ms)	Slightly longer than RSA alone.
Decryption Time	Minimal (< 1 ms)	Similar to encryption time	Comparable to RSA alone.

For the input “Hello Armenia!” (Table 3), we have the following results: (The time measurements may vary due to system performance and runtime factors).

Table 3. Algorithm Comparison by Speed

Algorithm	Encryption time	Decryption time
RSA	0.001416 seconds	0.005079 seconds
Walsh transform	0.000182 seconds	0.001003 seconds
Walsh+RSA	0.002846 seconds	0.006097 seconds

CONCLUSION

In conclusion, we have addressed the Walsh transform, the RSA algorithm, and the Hybrid Walsh-RSA algorithm. As indicated in *Table 3*, the major problem is the algorithm's low speed. However, our hybrid algorithm provides enhanced security. In the future, we plan to present the hybrid Walsh-AES algorithm and overcome the speed limitations, resulting in a faster and more secure algorithm than the ones discussed.

REFERENCES

[1].Episkoposian, S., & Grigoryan, S. Step-by-step greedy algorithms with respect to Walsh system. Slovak International Scientific Journal, №84, p. 11-14.

[2].Episkoposian, S. A. (2023, September 12–13). Securing information with the Walsh transform. In Proceedings of the III International Scientific and Practical Conference: Questions, Hypotheses, Answers – Science XXI Century (pp. 10–14). Toronto, Canada.

[3].Episkoposian, S. A. (2023). Application of Walsh system in data encryption. Tuijin Jishu/Journal of Propulsion Technology, 44(3), 494–502.

[4].Harrison, A. (2000, September 18). Articles: Computerworld.com. Computerworld. Retrieved December 5, 2024, from http://www.computerworld.com/article/2588444/security0/rsa-encryption-patent-released.html

[5].Kessler, G. C. (2017). An overview of cryptography. Boca Raton: Auerbach Publications.

[6].Kota, C. M., & Aissi, C. (n.d.). Implementation of the RSA algorithm and its cryptanalysis. University of Louisiana at Lafayette, College of Engineering, LA, USA.

[7].Nisha, S., & Farik, M. (2017, July). RSA public key cryptography algorithm – A review. International Journal of Scientific & Technology Research, 6(7).

[8].Padmavathi, B., & Ranjitha Kumari, S. (2013, April). A survey on performance analysis of DES, AES, and RSA algorithm along with LSB substitution technique. International Journal of Science and Research (IJSR), 2(4). Online ISSN: 2319-7064.

[9].Pethe, H. B., & Pande, S. R. (2017, January 1). Comparative study and analysis of cryptographic algorithms. International Journal of Advance Research in Computer Science and Management Studies, 5(1), 48–56.

[10].Priya, N., & Kannan, M. (2017, January). Comparative study of RSA and probabilistic encryption. International Journal of Engineering and Computer Science, 6(1), 19867–19871.

[11].Simpson, S. (1997, March 24). www.laits.utexas.edu. University of Texas. Retrieved December 10, 2024, from http://www.laits.utexas.edu/~anorman/BUS.FOR/course.mat/SSim/algorithms.html