DOI: 10.5281/zenodo.15290374
Link: https://zenodo.org/records/15290374

THE ROLE OF ENCRYPTION ALGORITHMS IN ENSURING INFORMATION SECURITY

Yusupova Zamira
Tashkent University of Information Technology named after Muhammad Al-Khwarazmi,
yusupovaz677@gmail.com
+998909337461
Yusupov Baxtiyor
Tashkent State Technical University named after Islam Karimov

Absract. Encryption algorithms play a very important role in ensuring information security. Their main function is to protect data by changing it into an unreadable form (cipher). Encryption algorithms encrypt data in such a way that a person who does not have a key cannot read the information. This is especially necessary to protect important information, including personal, financial or commercial secrets. Encryption algorithms can also be used to ensure the integrity of the data. Cryptographic methods, such as Hash functions, allow the data to be checked for non-tampering. Encryption algorithms play an important role in ensuring the authentication of users or systems. Examples include password encryption, secure connection setup (SSL/TLS), and authentication via digital signatures.

Keywords: AES, chacha20, RSA, diffil-hellman, elliptic curve, MD5, SHA 3, Electronic Digital Signature, algorithmic.

I. INTRODUCTION

Encryption algorithms are an important means of information security and play a decisive role in protecting confidential information from unauthorized access, capture and theft.

Encryption is the process of converting plain text or data into an unreadable format using a mathematical algorithm called cipher. Encrypted data can only be read by authorized users who have the correct password solution key to convert it to its original form.

Below is the principle of operation, application areas and other analysis of algorithms currently in widespread use.

AES (Advanced Encryption Standard) is a symmetric encryption algorithm widely used to protect confidential data. It was selected by the U.S. National Institute of standards and technology (NIST) in 2001 to replace the previous data encryption standard (DES). The key size can be 128/192/256 bits. Each encrypts data in 128-bit blocks.

This means that it takes 128 bits as input and outputs 128 bits of encrypted text as output. AES relies on the swap-Change Network principle, which means it is implemented using a series of linked operations that involve switching and mixing input data. AES performs operations not in bits, but in data bytes. Since the block size is 128 bits, The Cipher processes 128 bits (or 16 bytes) of input data at the same time [1].

ChaCha20 is a symmetric KeyFlow encryption algorithm that was introduced in 2008 by Daniel J. Designed by Bernstein. It is one of the most widely used stream ciphers and is a very safe and efficient algorithm. ChaCha20 works by creating a pseudo-random stream of bits based on a secret key and nonce (a disposable number).

This stream of bits is hacked with Open Text to create encrypted text. The ChaCha20 algorithm uses a 256-bit key and a 96-bit nonce. The ChaCha20 was designed to be very fast and safe

with a high degree of resistance to attacks such as differential and linear cryptanalysis. It is also designed to be immune to timing attacks and side-channel attacks.

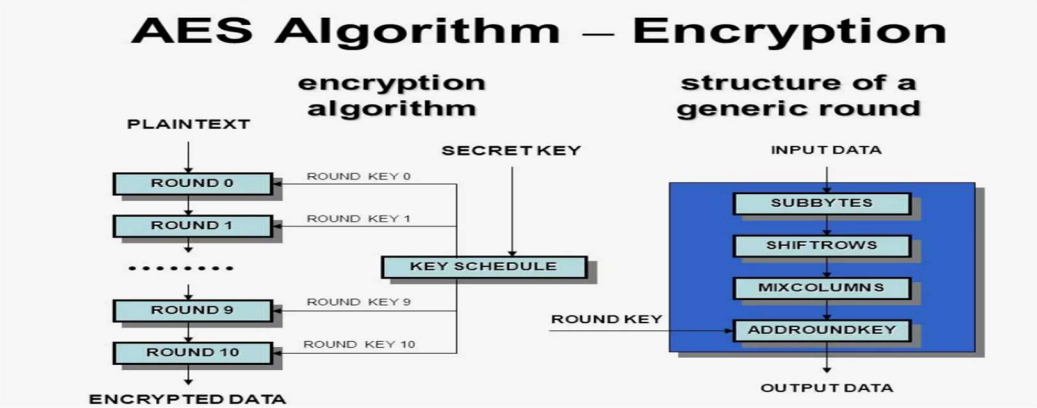

Figure 1. AES algorithm working principle

It is also the standard encryption algorithm for the latest version of the Transport Layer Security (TLS) protocol used to protect web traffic [2].

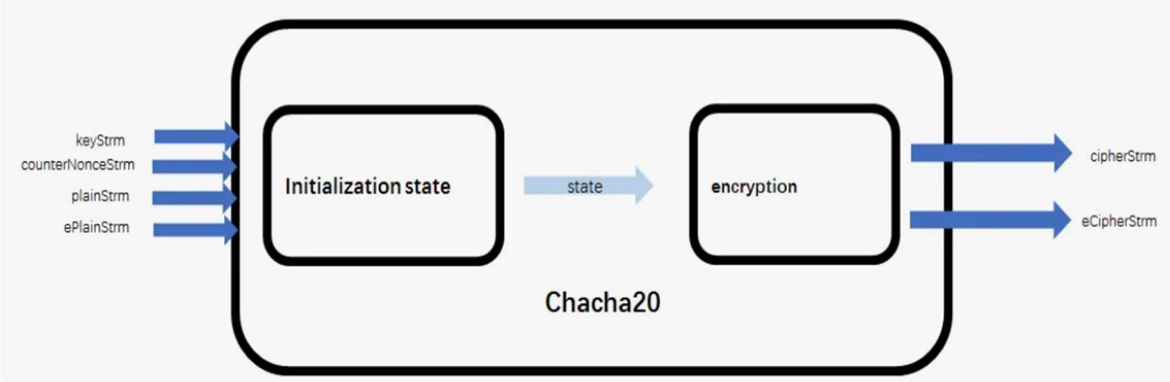

Figure 2. ChaCha20 algorithm working principle

RSA is a public key encryption algorithm developed in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman. It is widely used for secure data transmission over the Internet, as well as for digital signature and key exchange. RSA works using a pair of keys: a public key and a private key. The public key is used to encrypt data, and the private key is used to encrypt data. The public key consists of two parts: a module (the derivative of two large primes) and a pointer (usually a small number, for example, 65537). The secret key consists of an indicator calculated on the basis of the main factors of the same module and module. To encrypt data using RSA, Open Text is first converted to a numerical value using a specific encoding scheme (e.g. ASCII or Unicode). Then the Open Text is raised to the power of the public indicator and divided into modules, resulting in encrypted text. To decrypt the encrypted text, the recipient has his own personal

To decrypt the ciphertext, the recipient uses his or her private key to raise the ciphertext to the power of a private pointer and divide it into a module, resulting in the original Open Text. RSA is a very secure encryption algorithm because it is based on the difficulty of factoring large prime numbers. At the same time, it works relatively slowly compared to other modern encryption algorithms and is not recommended for encrypting large amounts of data[1].

The Diffie-Hellman algorithm is a key exchange algorithm that allows a shared secret key to be set over a non-secure communication channel without transmitting a key to either side. The algorithm was developed in 1976 by Whitfield Diffie and Martin Hellman and is one of the first practical open-key algorithms. The Diffie-Hellman algorithm works by using a mathematical function to create a common secret key between two parties. The function contains two values inside.

The function contains two values: a large prime number and the root root of that prime number. Each side selects a secret number, then uses a base number and a primitive Root to create a common key. The two sides exchange their public keys and then use their secret numbers and the other side's public keys to create a shared secret key. The security of the Diffie-Hellman algorithm is based on the difficulty of computing discrete logarithms. In other words, it is easy to calculate the value of the equation raised to the force, but it is very difficult to determine the force itself. While the Diffie-Hellman algorithm does not provide direct encryption, it is often used in conjunction with other encryption algorithms such as AES to establish a secure communication channel between the two parties. A common secret key created by the Diffie-Hellman algorithm can be used as a key for symmetric encryption.

Elliptic Curve Cryptography (ECC) is a type of public key cryptography based on the mathematics of elliptic curves. It is widely used in modern cryptographic systems such as secure communication protocols and digital signatures [3].

The basic idea of ECC is to use elliptic curve properties to create a trapdoor function that is easy to compute in one direction but difficult to compute in the opposite direction. This trapdoor function is used to generate open and private keys that can be used for encryption and digital signatures. To create a pair of common-private keys using ECC, a random private key is first selected, which is the number between 1 and the curve. The corresponding public key is then calculated as a point on an elliptic curve by multiplying the private key by the principal point on the curve.

The resulting point is a public key, which can be freely distributed. ECC security is based on the difficulty of solving the elliptic curve discrete logarithm problem (ECDLP). While ECDLP is not considered computable, attackers can use some specialized algorithms and resources to test and hack encryption. One of the main advantages of ECC is its effectiveness. Compared to other public key cryptography systems such as RSA, ECC requires shorter key lengths for the same level of security. This means that ECC can be implemented with fewer computing resources and provide faster encryption and decryption times.

The hash function is a mathematical function that accepts input data of arbitrary size and produces a fixed-size output called a hash or message digest. Hash functions are widely used in Informatics and cryptography for various purposes such as data integrity verification, digital signature and password storage. A good hash function must be deterministic, i.e. the same input data always produces the same hash value, and it must be collision resistant, i.e. it is difficult to find two different inputs that produce the same hash value. In addition, the hash function must have an avalanche effect, that is, a small change in the input data must lead to significant changes in the hash output. One common type of hash function is the cryptographic hash function, which is designed to protect against different types of attacks, such as preimage attacks, second preimage attacks, and collision attacks. Some frequently used cryptographic hash functions include MD5, SHA-1, SHA-2, and SHA-3.

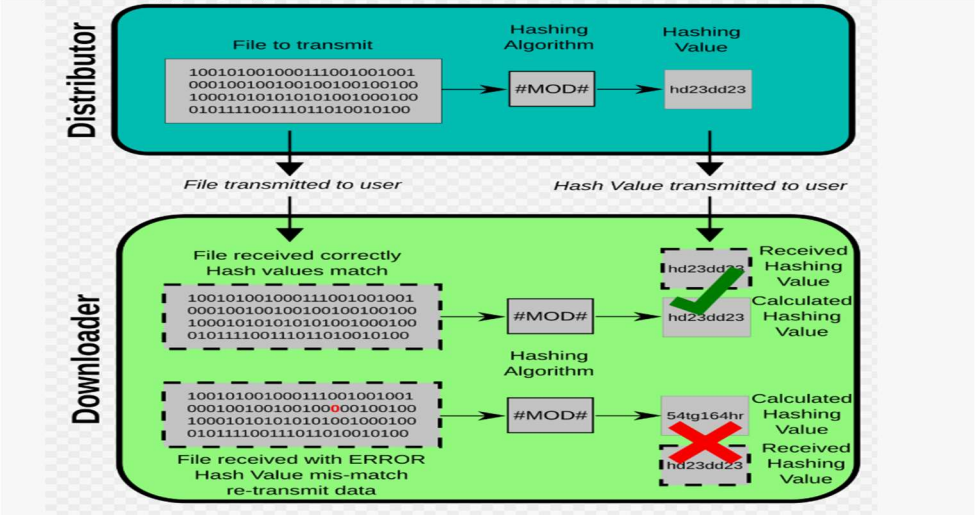

Figure 3. MD5 Hecht function working principle

One of the oldest algorithms for which MD5 is widely used, M5 is a one - way cryptographic function that modifies messages of any length and returns a solid-length line output of 32 characters. An example of MD5 hash-dijesti output looks like this: b6c7868ea605a8f951a03f284d08415e.

SHA - 3 is the latest in the SHA family. Developed through a public competition promoted by NIST, it is part of the same standard and is completely different from MD5, SHA-1 and SHA-2. SHA - 3 consists of four algorithms with different hash functions, and two extensible output functions can be used to generate domain hashing, random hashing, stream encryption, and MAC addresses:

1) SHA3-224;
2) SHA3-256;
3) SHA3-384;
4) SHA3-512;
5) SHAKE-128 (output function);
6) SHAKE-256 (output function).

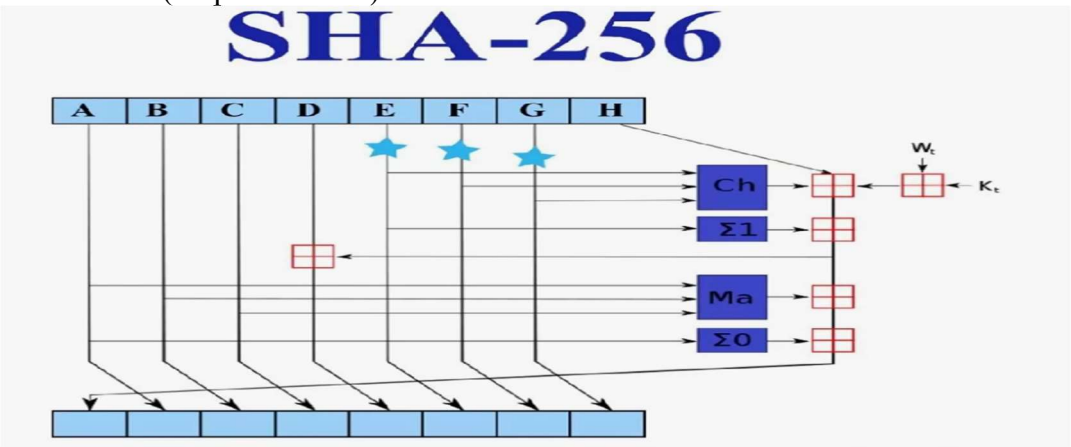

Figure 4. SHA3 Hecht function working principle

Electronic digital signature (EDS) algorithms are cryptographic protocols used to ensure the authenticity, integrity and non-rejection of digital documents or messages. They are widely used in e-commerce, online banking and other applications where secure communication and authentication are important. Several electronic digital signature algorithms exist, but one of the most commonly used algorithms is the RSA digital signature algorithm. The RSA algorithm is based on the properties of prime numbers and includes two keys, a secret key and a public key. The secret key is kept secret by the owner and used to sign messages, while the public key is used to verify the signature. To sign a message using the RSA digital signature algorithm, the sender first calculates the message hash using a cryptographic hash function such as SHA-256. The sender then encrypts the hash using his private key and creates a digital signature. After that, the recipient of the message can confirm the signature by decrypting it using the sender's public key and comparing it with the message cache.

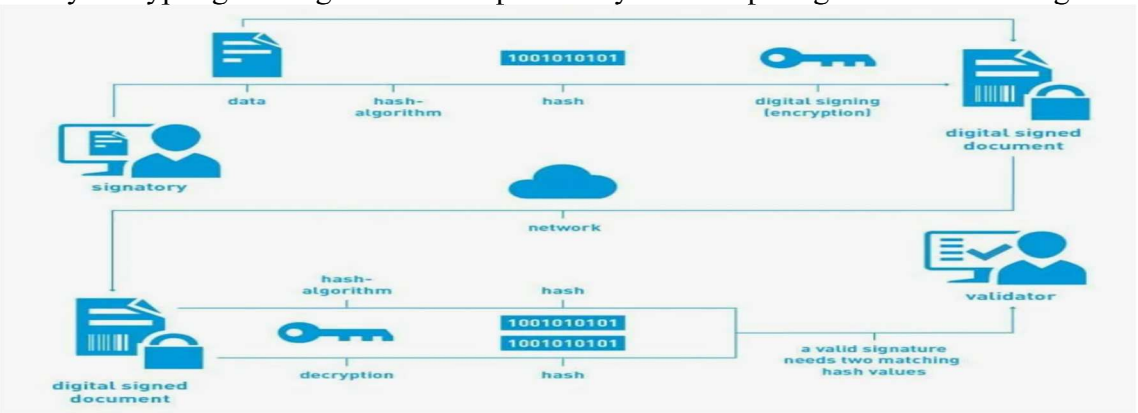

Figure 5. EDS algorithm processing circuit

Above, the main tasks of AES, chacha20, RSA, diffil-hellman, elliptic curve, MD5, SHA 3, electronic digital signature algorithms, the principles of processing, were cited, while in the qquyi table, their general analysis is keltiirl, which makes it possible to learn even more about encryption algorithms [4].

II. THEORETICAL ANALYSIS AND RESULT

Table 1. AES, chacha20, RSA, diffil-hellman, elliptic curve, MD5, SHA 3, general analysis of electronic digital signature algorithms

Algorithm name	Information security protection categories	Rate of exchange (bit per second)	Advantage rating	Potential vulnerabilities	Underdog	Threat tolerance	Programming languages	Application areas	Resistance to phishing attacks	The man in the middle is resistant to attack
AES	Privacy	1,1 mlrd	1	Basic size constraints	Vulnerable to side canal attacks	High	C, C++, Java, Python	Cryptographic	Resistant	High
ChaCha20	Privacy	3,3 mlrd	2	Not clear	Vulnerable to side canal attacks	Middle	C, C++, Python	Encryption, secure communication, VPN	Resistant	High
RSA	Authentication and data privacy	16 mil	3	Basic size constraints	Faktorizatsiya hujumlariga nisbatan zaif	Middle	C, C++, Java, Python	Encryption, digital signature, secure communication	Weak	Middle
Diffie-Hellman	Key exchange	70 mil	4	Generating weak random numbers	Vulnerable to man-in-the-middle attacks	Middle	C, C++, Java, Python	Generating weak random numbers	Weak	Low
Elliptic Curve Cryptography	Public key encryption and access control	130 mil	5	Generating weak random numbers	Elliptic curve vulnerable to discrete logarithm problem attacks	High	C, C	Cryptographic	Resistant	High
MD5	Totality	500	2	Cryptographic	Unprotected against riot attacks	Low	C, C++, Java, Python	Heshing	Weak	Low
SHA3	Totality	700	2	Not clear	Not clear	High	C, C++, Java, Python	Hashing, digital signatures, blockchain	Resistant	High
EDS	Authentication and integrity	16 mil	3	Cryptographic	Withstanding unknown attacks	Middle	C, C++, Java, Python	Autentification	Resistant	High

Conclusion

Encryption algorithms are an important component of information security, ensuring confidentiality, integrity, authentication, and non-denial of information. They are necessary to protect data, establish safe communication and resist various security threats. The correct selection and proper use of encryption algorithms is decisive in ensuring information security.

Bibliography

1) Akbarov D.E. Axborot xavfsizligini ta'minlashning kriptografik usullari va ularning qo'llanilishi // Toshkent, 2008, -B. - 394.Bruce S. Applied cryptography: protocols, algorithms, and source code in C //New York: Wiley. – 1996, - P. - 1027.
2) "Symmetric Encryption Algorithms: Live Long & Encrypt", 2020.
3) "Symmetric Encryption Algorithms", 2023.
4) Leighton Johnson, "Security component fundamentals for assessment", 2020.
Jay Thakkar, "Types of Encryption: 5 Encryption Algorithms & How to Choose the Right One", 2022.