Modern Cryptography

Table of contents

Abstract

This paper dives into the techniques used in modern day cryptography to aid in data secrecy. It outlines different approaches used in modern day cryptography techniques. It majorly informs us with the different approaches to cryptography, their types and algorithms. We have explored multiple cryptography techniques such as symmetric, asymmetric and touched hashing as well. To complement the covered topics we have dived into their working mechanisms and algorithms.

Introduction

Cryptography is a branch of cryptology, the science that deals with secure communications. It is the practice and study of tools and techniques which aid in concealing important information from adversaries.

Cryptography mainly deals with securing the data that might be easily accessed by actors with malicious intent. Cryptography utilizes multiple algorithms and tools to obfuscate the input data to the point where it cannot be easily recognized.

Objectives of Cryptography

In addition to confidentiality, cryptography must also achieve the following things:

Authenticity: The source of the message must be clear to the receiver. It should not be altered during the overall communication cycle.

Integrity: It should be possible to verify that the message received has not been altered in any way during the communication process. If it is altered it must be reflected accordingly in the receiving end.

Non repudiation: The source should not backoff from taking accountability for the deliverance of the message. The source must take full responsibility for the message it produces with its integrity.

History of Cryptography

Here is a table of important events in history that is presented as a timeline. In the table below we can see the time during which time period the algorithms came into existence or equivalent. [Seetha. R, Mythili. N.]

Year	**Techniques Used**
1900 BC	Evidence shows cipher text (Jumble letters) was carved on a stone in Egypt
1500 BC	In Mesopotamia information to be secured are written on clay slabs
500 BC	Spartans used Scytale device for sending and receiving secret messages. Scytale used transposition cipher.
800AD	Frequency analysis technique was used for breaking mono alphabetic ciphers
1467AD	Poly alphabetic ciphers was used
1400s - 1600s	Cryptography used for political and religious issues.
1853-1856	Charles Babbage used Vigenere cipher
1854	Playfair cipher was invented by Charles Wheatstone. It was used upto World War II.
1917	Gilbert Vernam devised a teleprinter cipher
1920s-1930	Enigma rotor machine (German) was introduced and later its performance was improved using Typex machine (British) and SIGABA machine (US).
1942	JN-25 Japanese Navy Cryptosystem was broken by US navy
1950	VIC cipher was discovered
1975	DES was established for secure electronic communication in financial organizations
1976	Diffe-Hellman key exchange was introduced
1991	PGP (Pretty Good Privacy) was released
2001	AES came into use

Classical Cryptography vs Modern Cryptography

The approach to classical cryptography is a bit different from current modern practices. As there was no computational power during those times. They had to rely on manual obfuscation techniques.

Some of the techniques used during the classical period were substitution and transposition of the letters used to deliver the message. The actual working mechanism was kept a secret as it would make it easier to reverse the encrypted message during the delivery.

In contrast to the modern cryptographic techniques where complex algorithms are produced. And millions of computations are done per second making it practically impossible to manually perform such calculations.

Let us look at the differences between classical and modern cryptography.

Classical Cryptography	**Modern Cryptography**
Operation is performed to the digits and letters directly.	The operation is performed at the unit level of data, such as bits and bytes.

Operating algorithms were kept secret to ensure data safety.	Cryptographic algorithms might be made public to everyone. As the computational complexity will make things harder to reverse the cipher text.
It requires all the tools and techniques used in a cryptosystem to be confidential.	Only the encryption / decryption key needs to be kept secret with the sender and receiver.
Examples of Algorithms: Substitution, Transposition, etc.	Examples of cryptography types are: Hash function, Symmetric Encryption, Asymmetric Encryption, etc.

Where is Cryptography used?

Cryptography is used almost everywhere in this day and age. There is almost no place where cryptography is not used. From *IoT* to the *servers* that provide 24/7 communications, cryptography is used in some form or another.

Even the algorithms to secure the bits are built in the chips themselves, for example *C.P.U*, *R.A.M.* It contains some mechanism to prevent the malicious actor from gaining unauthorized access to its internal processing data. Even though it is built into the OS itself, the point still stands.

Not only in the highly secure facilities but it also affects our everyday lives. From our everyday **social media accounts** to **bank accounts**, everything relies on cryptography for its security.

All of the cryptographic tools such as **hash**, **encryption**, **M.A.C (Message Authentication Code)** work together to provide security.

Terminologies

Here is a list of terminologies that you might come across in this paper.

Plain text / Clear text: It is any input that is feeded to the algorithm to produce the cipher text.

Cipher text: The obfuscated output that is produced after processing the plain text by the algorithm. It is not readable without reversing with its respective key.

Keys: The secure bytes used to transform plaintext to ciphertext and vice versa. They are sometimes referred to separately as encryption keys and decryption keys.

Encryption: The process of transforming the plaintext into the cipher text.

Decryption: The process of reversing the encryption. Or simply it is the process of converting the ciphertext back to plaintext.

Types of cryptography

There are multiple types of cryptography techniques and tools. Here are a list of most common concepts:

Hash functions

Hash functions are a set of algorithms that produce a fixed length output for any given input bytes. The produced output is only a *fraction* of the size of the input data. Produced output bytes are not reversible to the original input data. And the output produced should never repeat for unique input. But it should always produce the same output for the given input.

Properties of hash functions

Here is a list of properties of hash functions that make them what they are.

Preimage resistance: This property of hash function should make it computationally infeasible to reverse the hash output back to its input. In other words if someone tries to convert the output back to the input, it should be close to impossible to reverse.

Second Pre-Image Resistance: This property of any hash function states that for any given input and its hash value. It should be extremely hard to find the same hash with different input values.

Collision Resistance: This property of hash function ensures that it should be hard to have two different inputs of varying lengths that produce the same output hash value. In other words this property ensures that the hash function is collision free.

How do hash functions work?

The hash functions are all mathematical logics that operate on two fixed sized blocks of input. After the application of the algorithm it produces a fixed length output.

The size of the blocks is highly dependent on the algorithm used. It can vary anywhere between *128 bits* to *512 bits*.

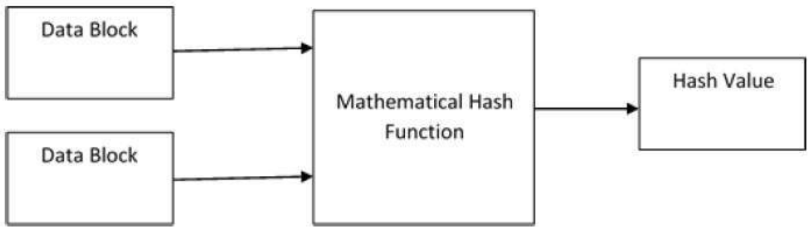


Image taken from https://www.tutorialspoint.com/cryptography/cryptography_hash_functions.htm

The algorithm first takes the seed value and the initial input to produce output bytes. And then repeatedly uses the output bytes from each preceding function iteration to calculate new bytes until no input bytes are left.

This is just a vague generalization of hash functions but it is necessary for us to understand the working algorithm.

For the above mentioned reason even a single bit input difference will yield completely different hash output.

Popular hash functions

Now that we've learned about hash functions let us understand the modern day hash functions that are widely accepted and implemented. Here is a list of hash functions that are considered cryptographically secure and modern to current standards.

Message Digest (MD)

It is a family suite of multiple hash functions such as: **MD2**, **MD4**, **MD5** and **MD6**. The implementation standard is also mentioned in the Internet Standard **RFC 1321**.

The **MD5** hash function was very popular in the recent past. The output of the hash functions in this family include a fixed length **128 bit** value. It was generally used to generate the integrity verification hash for software distributions.

However, in 2004 cases of hash collisions were detected by the security experts. Hence, it is no longer considered secure for scaled enterprise use.

Secure Hash Functions (SHA)

SHA is also a family of four hash functions. The four hash functions in SHA are : SHA-0, SHA-1, SHA-2, and SHA-3. The SHA family has varying bits output in each of them.

SHA-0 did not get as popular as other recent variants. It is due to the fact that SHA-0 had many issues. Out of which many issues were solved in SHA-1 and it is the most used version of SHA. Even the SSL suite uses SHA-1 for its proper functionality.

However, a method to generate collisions was found in 2005 for SHA-1. Which made the experts make new versions of SHA for long term reliability. Hence, SHA-2 and SHA-3 are introduced as recent additions.

RIPEMD (RACE Integrity Primitives Evaluation Message Digest)

It is a hash function that was generated by the open research community of European Hash Function. The family includes RIPEMD, RIPEMD-128 and RIPEMD-160.

The algorithm for *RIPEMD-128* uses the same principles which were used in *MD4*. Which was produced as a result to solve the issues and vulnerabilities in the original *RIPEMD* implementation.

The latest addition to the suite is RIPEMD-160 is the most widely used version because it has improved in many different areas of its implementation.

Encryption

Encryption is the process of obtaining the ciphertext after applying an encryption algorithm with an encryption key to the plaintext to conceal the insecure information from everyone.

In other words encryption is the process of scrambling the clear text in such a way that the obtained output cannot be understood by the party other than who is authorized to read it.

How does encryption work?

Here is a general overview of how the encryption process works. Please note that this is just a simple explanation of the steps involved in the encryption. The actual implementation depends on the algorithm used and its mode of operation.

- **Step #1**: Generate a random nonce to randomize the output of algorithm
- **Step #2**: Generate a random encryption key or derive it from the user input such as password
- **Step #3**: Apply multiple rounds of mathematical calculations / algorithms such as transposition, shifting and **XOR**ing
- **Step #4**: Produce the cipher text to bemused later for decryption

Types of encryption

There are mainly two types of encryption in modern cryptography. They are explained below

Symmetric key encryption

Symmetric encryption is the type of encryption which uses the same key for the encryption and the decryption process. In this type of encryption one single key is used for both encryption and decryption.

Here is a list of common encryption algorithms in symmetric key encryption: *DES*, *3DES*, *AES*, *IDEA*, *RC4*, and *RC5*.

Encryption and decryption with a symmetric algorithm are denoted by:

$E_K(M) = C$ (the cipher text)
$D_K(C) = M$ (the plain text)

Modes of operation

In symmetric key encryption there are two modes of operation. The mode of operation is how the algorithm operates with the input data. The two modes of operation used in symmetric key encryption is described below:

Stream Cipher

In this mode of operation the input data is directly processed byte by byte. The algorithm directly takes the input character and converts it into the cipher text using the specified encryption key. Due to the nature of this algorithm the operation uses *8 bits (in AES)* to perform encryption.

The stream ciphers are generally complex but in contrast they are not as secure as block ciphers. But they are fast compared to block ciphers because they deal with the input data directly to produce cipher texts.

Some common examples of stream ciphers are **Salsa20**, **RC4**, etc.

Block Cipher

In this mode of operation the input data is broken down into the blocks of bits ranging from 64 bits to 512 bits or more. The algorithm breaks down the plaintext into the blocks of bits and applies multiple rounds of arithmetic operations such as **transposition**, **shifting** and **XORing**.

The internal working mechanism for each algorithm might vary but the general principle remains the same.

Examples of block ciphers are: **AES**, **DES**, **3DES**. Let us look at the most widely used encryption algorithm

AES (Advanced Encryption Standard)

AES is the most advanced and cryptographically secure algorithm used for symmetric encryption. It is also sometimes known as Rijndael encryption because it is based on the algorithm of the same name.

It is faster than most other algorithms such as ***Feistel***, ***3DES***, etc. And the steps involved in the encryption process of AES are outlined below.

Operation of AES
The AES is a bit different from other algorithms because it breaks down the input into blocks of bits. However it performs calculations on bytes rather than bits. Meaning the 128 bit block will only be treated as 16 byte block for all the operation.

Also, in the 128 bit mode the algorithm does a 10 round calculation, 12 rounds for 192 bit keys and 14 rounds for 256 bit keys. Each round of operation will have a different key for its operation known as round key.

Let's take an example of 128 bits (16 bytes) to see how the operations are performed in AES.

Byte substitution: The 16 bytes are substituted by looking at a fixed table given in the design of the algorithm. The resulting output will be a matrix of 4x4.

Shiftrows: In this step of the operation each row of the matrix is shifted to the left. If the values fall off of the table then they are inserted to the other side while shifting. Here are the steps which is performed during the shiftrows:

- The first row is not shifted
- The second row is shifted by one byte, in other words it is shifted one row
- Third row is shifted two positions to the left
- Fourth row is shifted three position to the left

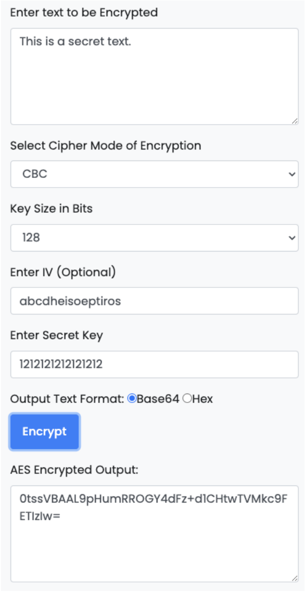
After performing the above operations the resulting matrix is a new matrix with same contents but shifted with respect to each other.

MixColumns: This is another step in the AES encryption operation. Each column is now transformed by using a mathematical function. The function takes four bytes of each column and produces a new output after applying mathematics, which replaces the original column. The result is another new matrix consisting of 16 new bytes.

Addroundkey:
This is the last operation in the AES encryption. Now the algorithm considers the 16 bytes as 128 bits. Then, an XOR operation is performed between the bytes and the round key. If this is the final round then the ciphertext is given out else the process is repeated.

The **decryption of the AES** encryption is performing the same above operations but in reverse order.

Example: The AES encryption example shown below is performed online here this link. The output is Base64 encoded string which can be converted to actual byte output if we wish.



Enter text to be Encrypted

This is a secret text.

Select Cipher Mode of Encryption: CBC

Key Size in Bits: 128

Enter IV (Optional): abcdheisoeptiros

Enter Secret Key: 1212121212121212

Output Text Format: ● Base64 ○ Hex

AES Encrypted Output:

```
0tssVBAAL9pHumRROGY4dFz+d1CHtwTVMkc9F
ETIzlw=
```

Asymmetric key encryption

Asymmetric encryption is also sometimes referred to as the Public Key encryption. It is called so because in this mode of encryption a pair of keys are used. One key is used for encrypting the data and the other is used for the decryption.

In this type of encryption the plaintext is generally encrypted using the **Public Key** and later the **Private Key** is used for the decryption process. The asymmetric encryption type also includes multiple algorithms. The usage of one type of algorithm entirely depends on what you want to use the mode of encryption for.

The first public key algorithm to be publicly disclosed was Diffie Hellman key exchange. After some time the *RSA (Rivest Shamir Addleman)* algorithm was also publicly made available.

Where is asymmetric encryption used ?

Asymmetric key encryption is generally used to encrypt and send plaintext from the public domain for example PGP. It can also be used to create digital signatures by using the private key during the encryption process. In fact the digital signatures are created exactly this way.

In practice the asymmetric encryption is used only to encrypt short plain texts, this is because of the fact that the algorithm is inherently slow. It will not be practical to use the RSA algorithm to encrypt larger plaintext for normal usage.

So, in practice a hybrid approach is used whenever asymmetric encryption is required. The faster symmetric encryption is used to encrypt the data and later the asymmetric algorithm is used to encrypt the key used for the encryption in symmetric encryption.

There are multiple types of algorithms for asymmetric key encryption: *RSA*, *DSA ECC*. Let us look at how the RSA algorithm works.

RSA (Rivest Shamir Addleman)

The algorithm was developed by Ron Rivest, Adi Shamir, and Leonard Addleman at MIT in 1977. And today it is the most widely used algorithm in asymmetric key cryptography. It is fast and secure in its mode of operation.

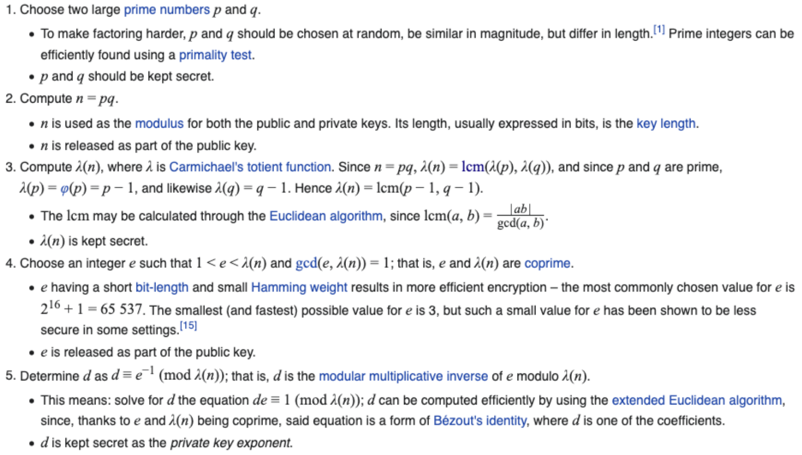
How does RSA work ?

The working mechanism of RSA is fairly simple to understand. However, the actual computation involved is rather complex as it involves prime numbers and modulo operation.

We can break down the overall operation into four simple steps to understand it better.

Key generation

The first step to the encryption with RSA is the key generation. The key generation logic is shown in the image below:



1. Choose two large prime numbers p and q.
   - To make factoring harder, p and q should be chosen at random, be similar in magnitude, but differ in length.[1] Prime integers can be efficiently found using a primality test.
   - p and q should be kept secret.
2. Compute $n = pq$.
   - n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
   - n is released as part of the public key.
3. Compute $\lambda(n)$, where λ is Carmichael's totient function. Since $n = pq$, $\lambda(n) = \operatorname{lcm}(\lambda(p), \lambda(q))$, and since p and q are prime, $\lambda(p) = \varphi(p) = p - 1$, and likewise $\lambda(q) = q - 1$. Hence $\lambda(n) = \operatorname{lcm}(p - 1, q - 1)$.
   - The lcm may be calculated through the Euclidean algorithm, since $\operatorname{lcm}(a, b) = \frac{|ab|}{\gcd(a, b)}$.
   - $\lambda(n)$ is kept secret.
4. Choose an integer e such that $1 < e < \lambda(n)$ and $\gcd(e, \lambda(n)) = 1$; that is, e and $\lambda(n)$ are coprime.
   - e having a short bit-length and small Hamming weight results in more efficient encryption – the most commonly chosen value for e is $2^{16} + 1 = 65\,537$. The smallest (and fastest) possible value for e is 3, but such a small value for e has been shown to be less secure in some settings.[15]
   - e is released as part of the public key.
5. Determine d as $d \equiv e^{-1} \pmod{\lambda(n)}$; that is, d is the modular multiplicative inverse of e modulo $\lambda(n)$.
   - This means: solve for d the equation $de \equiv 1 \pmod{\lambda(n)}$; d can be computed efficiently by using the extended Euclidean algorithm, since, thanks to e and $\lambda(n)$ being coprime, said equation is a form of Bézout's identity, where d is one of the coefficients.
   - d is kept secret as the *private key exponent*.

Image taken from: https://en.wikipedia.org/wiki/RSA_(cryptosystem)

Key distribution

This is the next step in the operation of RSA, the generated keys must be distributed for interested parties to use. If someone wants to send some message then the person must use the public key to encrypt the plaintext. The exchange of public keys must take place in a secure medium.

Encryption

This is the step where the actual plain text is converted to the cipher text. The process involved in this step are:

- The public key used for encryption is identified.
- The public key is used to encrypt the plaintext

Decryption

The decryption of the encrypted cipher text obtained from the above step can be done using the private key. The private key is the key pair of the public key used to encrypt the plain text.

Which cryptography type should we prefer?

We have understood multiple types of cryptography approaches in the above sections. Now let us analyze which cryptography approach we should pick.

From the above sections we learned that there are multiple approaches to cryptography such as hashing, encryption, digital signatures, etc. Let us look at the comparisons of each encryption type. [Seetha. R, Mythili. N.]

Hash Functions

The table below outlines the basic comparison points between multiple hash functions. These hash functions are already defined in the sections above.

Hash functions	**Variants**	**Rounds**	**Hash Value (bits)**	**Security**
MD	MD2 MD4 MD5 MD6	18 3 64 Varies	128 128 128 0-512	Non collision resistant
SHA	SHA-0 SHA-1 SHA-2 (256/512) SHA-3	80 80 64/80 24	160 160 256/ 512 1600	SHA-2 is secure
RIPEMD	RIPEMD RIPEMD128 RIPEMD160 RIPEMD 256 RI	48 64 80 64	128 128 160 256	RIPEMD-160 is widely used and secure

	PEMD320	80	320	

Symmetric Key Encryption Algorithms

Here is a list of symmetric key encryption algorithms’ comparison.

Symmetric key algorithms	Block / Stream	Key / Encryption process / Structure	Security
DES	Block	56 bit key / 16 rounds/ Balanced Feistel network	Weak
AES	Block	128 bit key/10 rounds/ Substitution–permutation on network	Strong and still recommended
Blowfish	Block	32-448 bit key/16 rounds/ Feistel network	Secure and used for commercial purposes
RC4	Stream	40–2048 bits key/ 1 round	Weak

Summary

We have looked into the current status of modern cryptography. Cryptography helps us secure modern day communications by obfuscating the plaintext data. It produces ciphertext which keeps the plaintext secure. There are multiple types of cryptographic approaches such as hash hash function, encryption, etc.

We outlined the currently available algorithms for hash functions and also encryption techniques. We discussed the operating mechanism of those algorithms and also compared them side by side. We were able to get an idea on how things work in modern cryptography.

References

Schneier, B. (1995). Part II—Cryptographic Techniques. In *Applied cryptography*. essay, John Wiley.

Seetha. R, Mythili. N. (2020). *Modern Cryptography - A Review*. Research journal.

Pearson Education (US). (2011). *Modern cryptography*.

Advanced encryption standard. Tutorials Point. (n.d.). Retrieved September 26, 2022, from https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm

Cryptography hash functions. Tutorials Point. (n.d.). Retrieved October 8, 2022, from https://www.tutorialspoint.com/cryptography/cryptography_hash_functions.htm

Cryptography. Wachemo University eLearning Platform. (n.d.). Retrieved October 9, 2022, from https://wachemo-elearning.net/courses/31781/lessons/chapter-seven-data-security-and-integrity/topic/7-3-cryptography/

Fruhlinger, J. (2022, May 22). *What is cryptography? how algorithms keep information secret and safe*. CSO Online. Retrieved November 1, 2022, from https://www.csoonline.com/article/3583976/what-is-cryptography-how-algorithms-keep-information-secret-and-safe.html

Khan Academy. (n.d.). *Online data security* | *computers and the internet*. Khan Academy. Retrieved November 5, 2022, from https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:online-data-security

Peter Smirnoff & Dawn M. Turner (guests). (n.d.). *Symmetric key encryption - why, where and how it's used in banking*. Cryptomathic. Retrieved November 8, 2022, from https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking

Symmetric key algorithm. Symmetric Key Algorithm - an overview | ScienceDirect Topics. (n.d.). Retrieved December 4, 2022, from https://www.sciencedirect.com/topics/computer-science/symmetric-key-algorithm

What is encryption? | *types of encryption* | *cloudflare*. (n.d.). Retrieved December 8, 2022, from https://www.cloudflare.com/learning/ssl/what-is-encryption/