Symmetric Key Cryptography

Symmetric key cryptography (SKC) is the oldest and most intuitive form of cryptography. With SKC, confidential information is secured through symmetric key encryption (SKE), that is, by using a *single secret key* for both encryption and decryption.

SKC involves:

- An encryption function that converts a given plain text instance to ciphertext while utilizing a secret key
- A decryption function that inverts the operation by converting the ciphertext back to plain text using the same secret key

In cryptography, a symmetric key is one that is used both to encrypt and decrypt information. This means that to decrypt information, one must have the same key that was used to encrypt it. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption.

Plain text can mean any kind of unencrypted data such as natural language text or binary code whose information content is in principle directly accessible, while ciphertext refers to encrypted data whose information content is intended to be inaccessible before decryption.

An algorithm that describes the encryption and decryption operations using a shared secret key is also called a symmetric cipher.

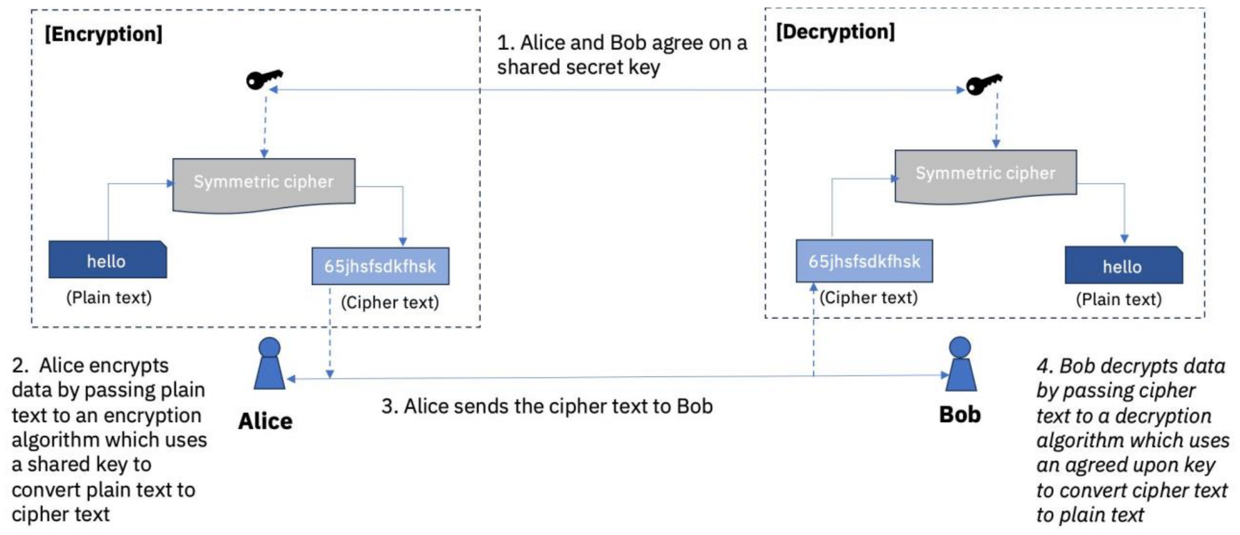
[Encryption]
1. Alice and Bob agree on a shared secret key
[Decryption]
Symmetric cipher
hello
(Plain text)
65jhsfsdkfhsk
(Cipher text)
Symmetric cipher
65jhsfsdkfhsk
(Cipher text)
hello
(Plain text)
2. Alice encrypts data by passing plain text to an encryption algorithm which uses a shared key to convert plain text to cipher text
Alice
3. Alice sends the cipher text to Bob
Bob
4. Bob decrypts data by passing cipher text to a decryption algorithm which uses an agreed upon key to convert cipher text to plain text

Figure 1: Symmetric key encryption of a given plain text to ciphertext and decryption back to plain text using the same key.

Properties of symmetric key cryptosystems

A symmetric key cryptosystem should ensure the following properties to secure messages statically stored data and/or communications over some transmission channel:

- **Confidentiality:** Refers to the property that the information content of encrypted messages is protected from unauthorized access.
- **Integrity:** Refers to the property that any tampering of encrypted messages during storage or transmission can be detected.
- **Authenticity:** Refers to the property that the receiver of a message can verify the identity of the sender and detect impersonation by an unauthorized party.

Furthermore, these properties should be realized in a setting where the algorithms or ciphers used for encryption and decryption may be public and where access to the information content of encrypted messages is controlled exclusively through access to the secret key.

Implementing a secure symmetric key cryptosystem therefore involves two main tasks:

1. Employing a robust symmetric key encryption algorithm resistant to cryptographic attacks.

2. Ensuring confidentiality in the distribution and management of secret keys.

Illustration of symmetric key encryption using python

We show a simple example of encrypt and decrypt operations using the classical Caesar shift cipher and the modern Advanced Encryption System (AES), which has been the standard for symmetric key encryption since 2001. First, we set up some Python libraries that provide the needed symmetric key encryption ciphers, and then define the plain text we wish to encrypt.

Caesar shift cipher:

If you have a message you want to transmit securely, you can encrypt it (translate it into a secret code). One of the simplest ways to do this is with a shift cipher. Famously, Julius Caesar used this type of cipher when sending messages to his military commanders. A shift cipher involves replacing each letter in the message by a letter that is some fixed number of positions further along in the alphabet. We'll call this number the encryption key. It is just the length of the shift we are using.

The Caesar cipher shifts all the letters in a piece of text by a certain number of places. The key for this cipher is a letter which represents the number of place for the shift. So, for example, a key D means "shift 3 places" and a key M means "shift 12 places". Note that a key A means "do not shift" and a key Z can either mean "shift 25 places" or "shift one place backwards". For example, the word "CAESAR" with a shift P becomes "RPTHPG"

Caesar shift encryption involves defining

- An alphabet of possible characters to encode
- A *shift value* which can be between 0 (unencrypted) and the length of the alphabet. We consider this the *key*.

It is known as a *monoalphabetic substitution cipher* since each letter of the plain text is substituted with another in the ciphertext. In this example we will use lowercase letters of the alphabet.

Lets use our jupyter notebooks

Applications of symmetric key cryptography

Application	Description	Examples
Data Protection	Protects sensitive data at rest or in transit.	Full-disk encryption, SSL/TLS for secure web browsing.
Secure Communication	Ensures confidentiality and integrity of messages.	Wi-Fi networks (WPA2), Virtual Private Networks (VPNs).
Financial Transactions	Secures financial data during transactions.	Banking apps, card payments using AES encryption.
Messaging Apps	Provides end-to-end encryption for secure messaging.	iMessage, WhatsApp (combines with asymmetric methods).
File Storage and Compression Tools	Adds security to compressed files or stored data.	ZIP file encryption, encrypted cloud storage services like Dropbox when using symmetric keys internally for efficiency

symmetric cryptosystems have the following problems:

— ***Keys must be distributed in secret***. They are as valuable as all the messages they encrypt, since knowledge of the key gives knowledge of all the messages. For encryption systems that span the world, this can be a daunting task. Often couriers' hand-carry keys to their destinations.

— ***If a key is compromised (stolen, guessed, extorted, bribed, etc.),*** then Eve can decrypt all message traffic encrypted with that key. She can also pretend to be one of the parties and produce false messages to fool the other party.

— ***Assuming a separate key is used for each pair of users in a network***, the total number of keys increases rapidly as the number of users increases. A network of *n* users requires $n(n - 1)/2$ keys. For example, 10 users require 45 different keys to talk with one another and 100 users require 4950 keys. This problem can be minimized by keeping the number of users small, but that is not always possible.

The Characteristics of symmetric key cryptography include:

- Use the same key for encryption and decryption
- Generally faster and more efficient than asymmetric encryption
- Suitable for encrypting large amounts of data
- Require secure key exchange mechanisms
- Typically use keys of 128, 192, or 256 bits

There are four types of symmetric cryptography: classical. Substitution, block and stream ciphers.

Classical ciphers

Classical ciphers are early encryption techniques that were used before the advent of modern computer-based cryptography. Some types of classical ciphers include:

- Substitution ciphers: These replace each letter in the plaintext with another letter or symbol. E.g. Simple substitution cipher where Each letter is replaced by another letter consistently and Caesar cipher: A type of substitution cipher where each letter is shifted a fixed number of positions in the alphabet.
- Transposition ciphers rearrange the order of letters in the plaintext without changing the actual letters used. E.g. Columnar transposition: The plaintext is written out in rows of a fixed length, and then read out column by column and Rail fence cipher where the plaintext is written downwards and diagonally on successive "rails" of an imaginary fence, then read off row by row.

rearranges the position of the plain text’s characters. In transposition Cipher Technique, The position of the character is changed but character’s identity is not changed.

To use a substitution cipher we replace (substitute) each letter of the plaintext with a different letter in the cipher text. To use this cipher we need a table of letter replacements. For example, look at the following table

Plain	C	D	E	H	I	N	P	R	S	T	Y
Cipher	X	J	L	A	Z	E	V	K	H	O	M

Using this substitution,
the plaintext THIS SENTENCE IS ENCRYPTED is changed to
this ciphertext OAZH HLEOLEXL ZH LEXKMVOLJ

Both Substitution cipher technique and Transposition cipher technique are the types of Traditional cipher which are used to convert the plain text into cipher text.

- Polyalphabetic substitution ciphers use multiple substitution alphabets. The most famous example is the Vigenère cipher, which uses a keyword to select different Caesar cipher shifts for each letter.
- Nomenclators use a mix of substitution techniques, including letter substitutions, bigram/trigram substitutions, and code words for common phrases.

Classical ciphers were used for centuries but are now considered insecure for serious applications, as they can be broken relatively easily using frequency analysis and other cryptanalytic techniques. However, they form an important part of cryptography's history and help illustrate basic encryption concepts.

Substitution ciphers

Substitution ciphers are a type of encryption technique where each letter in the plaintext is replaced by another letter or symbol in the ciphertext. Simple substitution ciphers replace each letter with another letter consistently. For example, every 'A' might be replaced with 'X', every 'B' with 'Y', etc. They are vulnerable to frequency analysis attacks, since the frequency distribution of letters in the ciphertext matches that of the plaintext language. Polyalphabetic substitution ciphers use multiple substitution alphabets to make frequency analysis more difficult. The Vigenère cipher is a well-known example. Homophonic substitution ciphers replace each letter with multiple possible ciphertext symbols to flatten the frequency distribution. While simple for humans to use manually, substitution ciphers are considered weak by modern cryptographic standards. They formed the basis for more advanced ciphers and encryption techniques developed later. Famous historical examples include the Caesar cipher (shifting the alphabet) and nomenclators used by early governments and militaries. Modern block ciphers like DES and AES use substitution (along with other techniques) as part of their encryption process, but in a much more complex way.

Block ciphers

Block ciphers encrypt **fixed-size** (typically 64 or 128 bits) blocks of data, Block ciphers are more common for general-purpose encryption. Further reading here https://mrajacse.wordpress.com/wp-content/uploads/2012/01/applied-cryptography-2nd-ed-b-schneier.pdf .

Examples include:

- ✓ **Advanced Encryption Standard (AES).** It's the gold standard for symmetric encryption since its adoption by the U.S. government in 2001. This block cipher operates on 128-bit blocks and supports key sizes of 128, 192, or 256 bits, offering a balance of security and efficiency across various platforms. AES's widespread adoption is due to its security and versatility, making it the go-to choose for many applications today from Wi-Fi networks (WPA2/WPA3) to cloud storage services like Google Drive and Dropbox. It's also widely used in VPNs, secure messaging apps like WhatsApp, and government communications. (check the security settings of your phones or when you try to put wifi passwords or the famous whatsapp notification" this message is encrypted"

- ✓ **Data Encryption Standard (DES) and Triple DES (3DES)** developed in the 1970s, was once the primary encryption standard but has since been phased out due to its vulnerability to modern attacks. Its 64-bit block size and 56-bit key are no longer considered secure. To address these shortcomings, Triple DES (3DES) was introduced, applying the DES algorithm three times with different keys. While 3DES offers improved security over its predecessor, it is slower than modern alternatives. They can still be found in legacy systems in the financial sector. 3DES is used in electronic payment systems and as a fallback in some TLS implementations, though it's being phased out.
- ✓ **Blowfish** created by Bruce Schneier in 1993, was designed as a fast and secure alternative to DES. With its 64-bit block size and variable key length ranging from 32 to 448 bits, Blowfish offers flexibility and security for various applications. Its efficiency and open-source nature have contributed to its continued use in many systems.
- ✓ **Twofish** the successor to Blowfish, was a finalist in the AES selection process. It operates on 128-bit blocks and supports key sizes up to 256 bits. While Twofish is considered highly secure, it has not gained as much traction as AES in widespread adoption.
- ✓ **Serpent** another AES finalist, was designed with a focus on security over speed. It uses a 128-bit block size and supports key sizes of 128, 192, or 256 bits. Although Serpent is regarded as very secure, its slower performance in software implementations compared to AES has limited its adoption.

Stream ciphers

stream ciphers encrypt **individual bytes/bits** of data. Stream ciphers are useful for encrypting continuous data streams. They are, in one sense, very simple block ciphers having block length equal to one. What makes them useful is the fact that the encryption transformation can change for each symbol of plaintext being encrypted. In situations where transmission errors are highly probable, stream ciphers are advantageous because they have no error propagation. They can also be used when the data must be processed one symbol at a time (e.g., if the equipment has no memory or buffering of data is limited). Examples include:

- ✓ ChaCha20
- ✓ RC4 (although no longer considered secure)
- ✓ Salsa20

Security principles

Symmetric ciphers adhere to two important security principles: Kerckhoff's principle and shannon's Maxim. These principles emphasize that cryptographic security should depend on the secrecy of keys rather than algorithms.

Kerckhoffs' principle

The security of a crypto-system should depend only on the secrecy of the key and not on the secrecy of the algorithm If the algorithm is also kept secret then such ciphers are known as classified ciphers.

Classified ciphers are used in defense and military organizations, the usage and practice of which are not recommended in general.

Rationale behind Kerckhoffs' principle

- It is easy to hide a short key than a long algorithm.
- Algorithm can be easily reverse engineered.
- It is easy to replace a compromised key than a compromised algorithm, if required.
- If many parties communicate with different algorithms, then the complexity of maintenance matters.

- If the algorithm is public, it is available for analysis by experts worldwide, and if it survives we gain high confidence on the algorithm.
- A flaw in the algorithm can be fixed easily, if it is in public domain.
- Keeping algorithm open helps in standardization.

How to define security

- An obvious notion from the Kerckhoffs' principle follows that, the key should not be leaked from the cipher text. But it is not sufficient, because for known plain text attack model this notion might fail.
- Plain text should not be leaked from the cipher text. This may also be insufficient. For example, if we consider a situation where 80% of the plain text is leaked and the remaining 20% is secured then the amount of leaked plain text may be containing sufficient data for a successful attack.
- No plain text character should be leaked from the cipher text. This even may be insufficient, as for example in bank transactions even if the exact amount of account balance is not revealed from the cipher text, but leakage of any extra information about the account balance would be undesirable.
- No "meaningful information" about the plain text should be revealed from the cipher text. For example, in bank transactions even if the exact balance of an account related to a transaction is not revealed, but any kind of information, for example like, "account balance is more than one lakh" might come out to be a meaningful information leading to insecurity of the account.

In summary

- The security of a cryptosystem should rely on the secrecy of the key, not on the secrecy of the algorithm.
- The algorithm should be able to fall into enemy hands without compromising security.
- This allows algorithms to be publicly analyzed and vetted by the cryptographic community.

- It enables standardization and interoperability of cryptographic systems.
- Changing keys is much easier than changing algorithms if security is compromised.

Shannon's maxim

- "The enemy knows the system."
- This is a rephrasing and generalization of Kerckhoffs' principle.
- Assume the adversary knows all details of your cryptosystem except the key.
- Security should not rely on keeping the algorithm secret.
- Cryptographic systems should be designed to be secure even if everything except the key is public knowledge.

Applications

- ❖ Secure communication protocols (e.g., TLS/SSL)
- ❖ File and disk encryption
- ❖ Virtual Private Networks (VPNs)
- ❖ Secure storage of passwords (when combined with key derivation functions)

Advantages of symmetric-key cryptography

1. SKC is fast and efficient, making it suitable for large data volumes or real-time communications. It scales well with data size and has low computational overhead.
2. SKC handles multiple users and large amounts of data effectively due to its low resource requirements.
3. SKC protocols are easier to implement compared to asymmetric methods, which makes them more accessible for developers.
4. SKC can be used as a building block for other cryptographic tools like pseudorandom number generators, hash functions, and digital signatures.
5. Keys are relatively short compared to asymmetric cryptography, which simplifies key management in some respects.

Challenges and limitations of symmetric key cryptography

1. Key Distribution and Management: SKC requires both parties to share the same secret key, which must be kept confidential. Secure distribution is a significant challenge.

2. Lack of Non-Repudiation: Since the same key is used for encryption and decryption, it's impossible to prove who sent a message.
3. Key Management in Networks: Managing multiple keys in large networks is complex and often requires an unconditionally trusted third-party authority (TTP).
4. Frequent Key Changes: Best practices dictate changing keys frequently, ideally for each session.
5. Digital Signatures Limitations: Digital signatures based on SKC typically require large keys or a TTP for verification purposes.