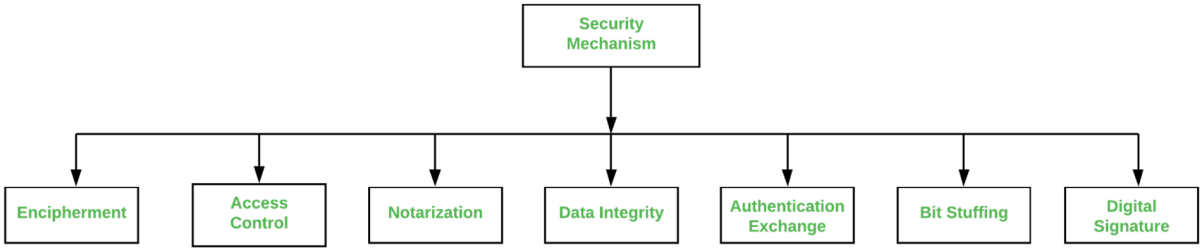
Computer Network Security Mechanisms

Network Security is a field in computer technology that deals with ensuring security of computer network infrastructure. As the network is very necessary for sharing of information whether it is at hardware level such as printer, scanner, or at software level. Therefore security mechanisms can also be termed as a set of processes that deal with recovery from security attacks. Various mechanisms are designed to recover from these specific attacks at various protocol layers.



Types of Security Mechanism are :

->Encipherment :

This security mechanism deals with hiding and covering of data which helps data to become confidential. It is achieved by applying mathematical calculations or algorithms which reconstruct information into non-readable form. It is achieved by two famous techniques named Cryptography and Encipherment. Level of data encryption is dependent on the algorithm used for encipherment.

->Access Control :

This mechanism is used to stop unattended access to data which you are sending. It can be achieved by various techniques such as applying passwords, using a firewall, or just by adding PIN to data.

->Notarization :

This security mechanism involves use of trusted third parties in communication. It acts as a mediator between sender and receiver so that any chance of conflict is reduced. This mediator keeps record of requests made by sender to receiver for later denied.

->Data Integrity :

This security mechanism is used by appending value to data to which is created by data itself. It is similar to sending a packet of information known to both sending and receiving parties and checked before and after data is received. When this packet or data which is appended is checked and is the same while sending and receiving data integrity is maintained.

->Authentication exchange :
This security mechanism deals with identity to be known in communication. This is achieved at the TCP/IP layer where two-way handshaking mechanism is used to ensure data is sent or not

->Bit stuffing :

This security mechanism is used to add some extra bits into data which is being transmitted. It helps data to be checked at the receiving end and is achieved by Even parity or Odd Parity.

->Digital Signature :

This security mechanism is achieved by adding digital data that is not visible to eyes. It is a form of electronic signature which is added by sender which is checked by receiver electronically. This mechanism is used to preserve data which is not more confidential but the sender's identity is to be notified.

Encryption

Encryption is the method by which information is converted into secret code that hides the information's true meaning. The science of encrypting and decrypting information is called cryptography.

- Cryptography, a word with Greek origins, means "secret writing."
- Security in networking is based on cryptography, the science and art of transforming messages to make them secure and immune to attack.
- Cryptography can provide several aspects of security related to the interchange of messages through networks.
- These aspects are confidentiality, integrity, authentication, and nonrepudiation.

Components of encryption

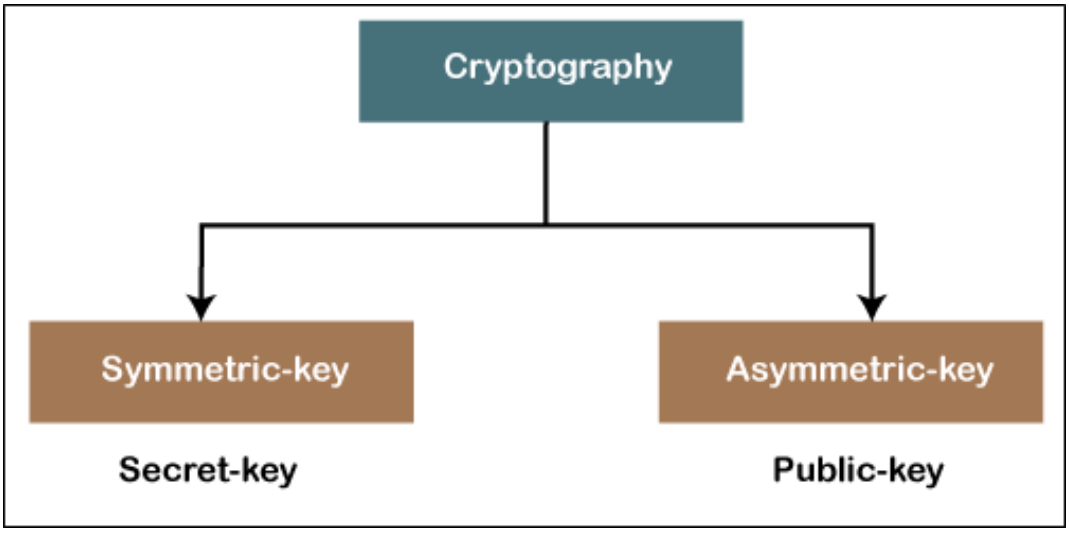
Plain Text :The original message, before being transformed, is called plaintext.An encryption algorithm transforms the plain text into ciphertext; a decryption algorithm transforms the ciphertext back into plaintext. The sender uses an encryption algorithm, and the receiver uses a decryption algorithm.

Cipher Text:After the message is transformed, it is called ciphertext.We refer to encryption and decryption algorithms as ciphers. The term cipher is also used to

refer to different categories of algorithms in cryptography. This is not to say that every sender-receiver pair needs their very own unique cipher for a secure communication. On the contrary, one cipher can serve millions of communicating pairs.

Encryption: Encryption is the process of translating plain text data (plaintext) into something that appears to be random and meaningless (ciphertext).

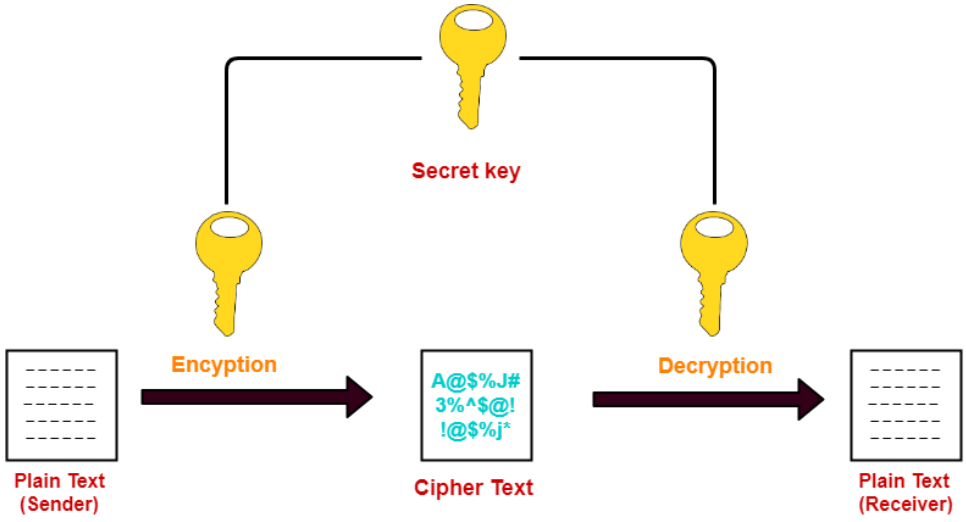
Decryption:Decryption is the process of converting ciphertext back to plaintext.



Symmetric and Asymmetric key cryptography.

-> Symmetric cryptography:
In symmetric key cryptography, an individual key is used for both encryption and decryption. The sender needs the key to encrypt the plaintext and sends the cipher document to the receiver. The receiver used a similar key (or ruleset) to decrypt the message and recover the plaintext. Because an individual key is used for both functions, symmetric key cryptography is also known as symmetric encryption.

Symmetric key cryptography schemes are usually categorised such as stream ciphers or block ciphers. Stream ciphers work on a single bit (byte or computer word) at a time and execute some form of feedback structure so that the key is constantly changing.

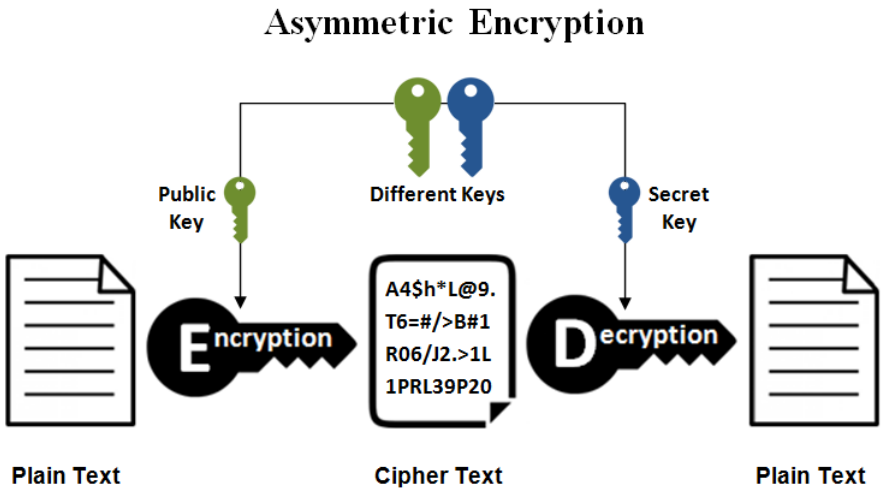


Symmetric Key Cryptography

-> Asymmetric cryptography:

Asymmetric cryptography uses two keys for encryption and decryption. It depends on the technique of public and private keys. A public key, which is interchanged between higher than one user. Data is decrypted by a private key, which is not transformed. It is slower but more secure. The public key used in this encryption technique is applicable to everyone, but the private key used in it is not revealed.

In asymmetric encryption, a message that is encrypted utilising a public key can be decrypted by a private key, while if the message is encrypted by a private key can be decrypted by utilising the public key. Asymmetric encryption is broadly used in day-to- day communication channels, particularly on the internet.



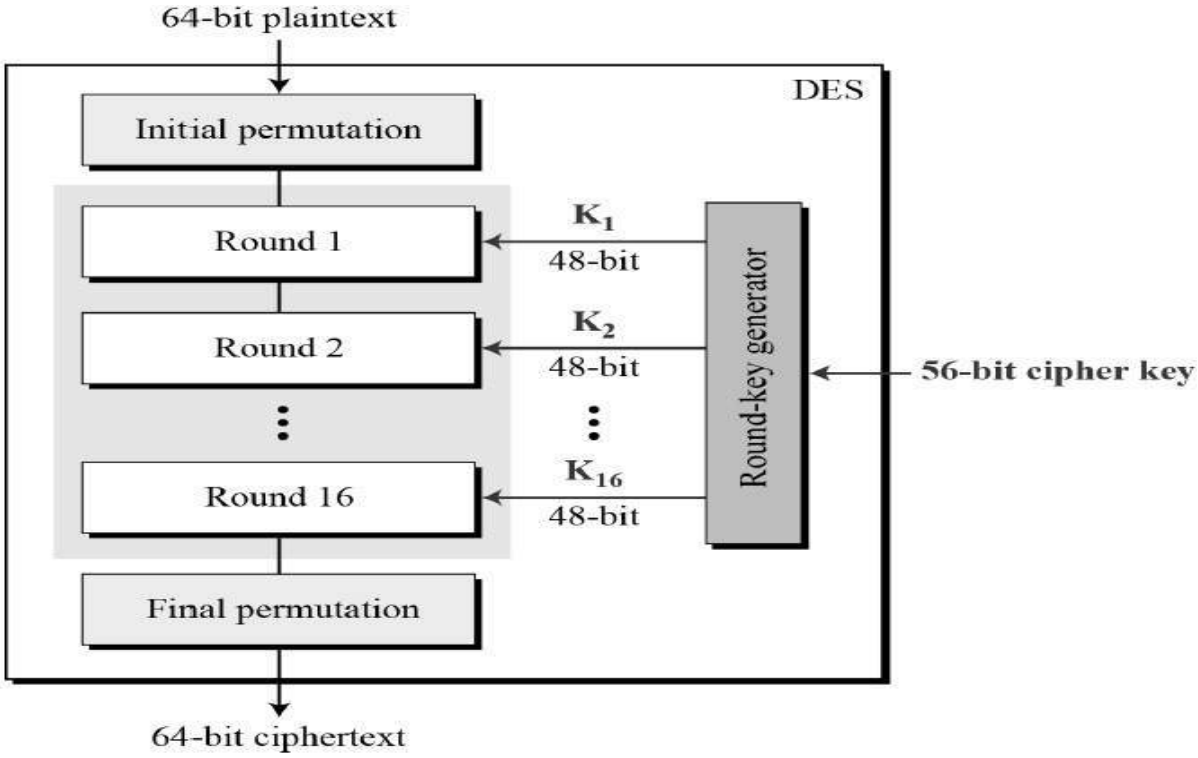
comparison between Symmetric key and Asymmetric Key cryptography

Symmetric key cryptography	Asymmetric Key cryptography
There is only one key used, and the similar key can be used to encrypt and decrypt the message.	There are two different cryptographic keys, known as the public and the private keys, that are used for encryption and decryption.
It is effective as this technique is recommended for high amounts of text.	It is inefficient as this approach is used only for short messages.
Symmetric encryption is generally used to transmit bulk information.	It is generally used in smaller transactions. It is used for making a secure connection channel before transferring the actual information.
Symmetric key cryptography is also known as secret-key cryptography or private key cryptography.	Asymmetric key cryptography is also known as public-key cryptography or a conventional cryptographic system.
Symmetric key cryptography uses fewer resources as compared to asymmetric key cryptography.	Asymmetric key cryptography uses more resources as compared to symmetric key cryptography.
The length of the keys used is frequently 128 or 256 bits, based on the security needs.	The length of the keys is much higher, such as the recommended RSA key size is 2048 bits or higher.

DES Algorithm with Example

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses a 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only)

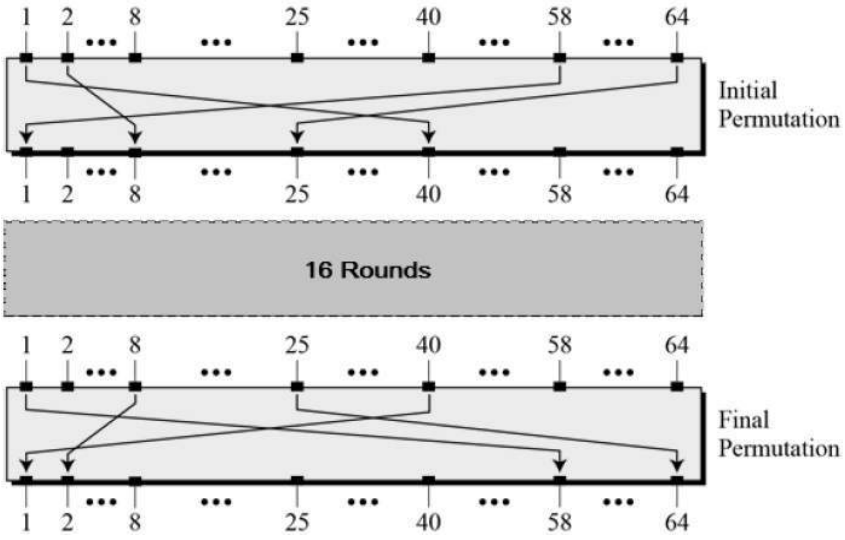


DES is based on the Feistel Cipher, all that is required to specify DES is –

- Round function
- Key schedule
- Any additional processing – Initial and final permutation

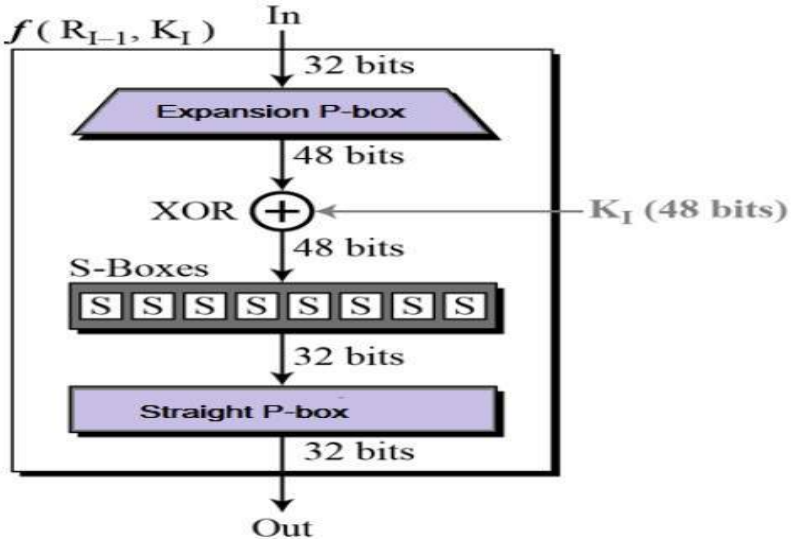
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES.



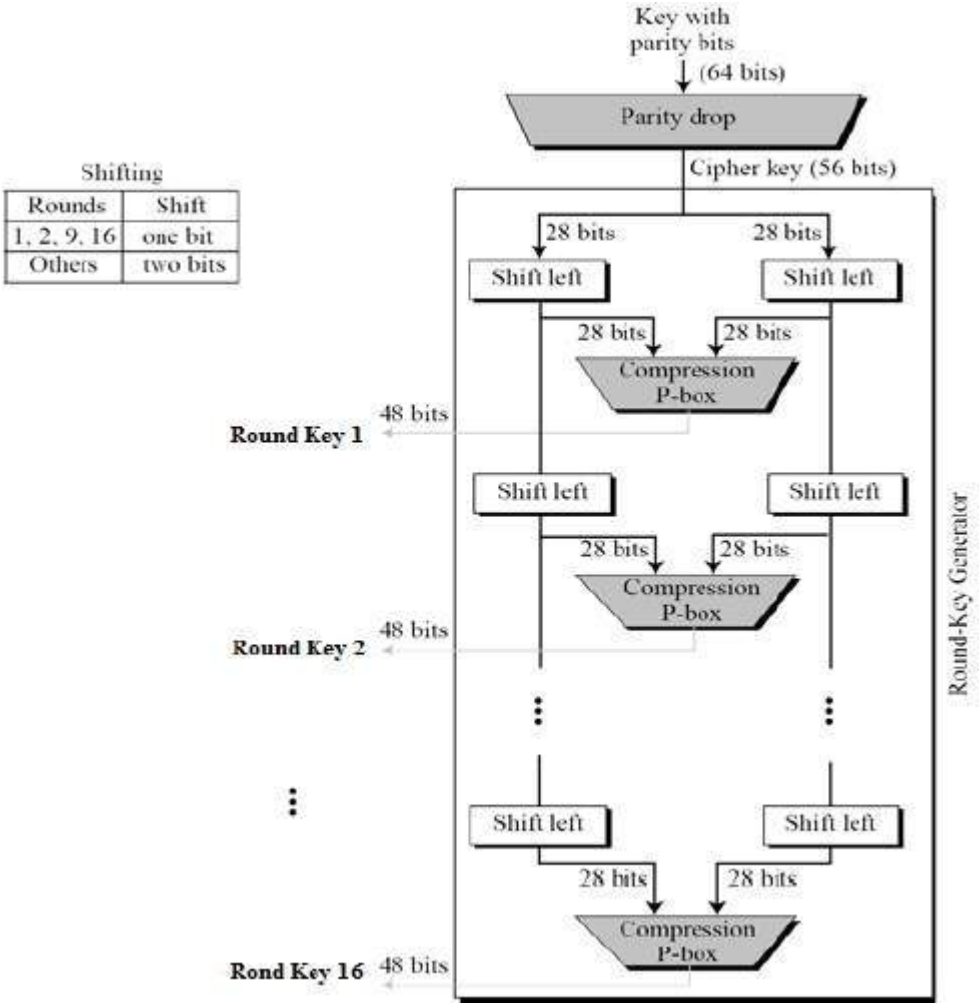
Round Function

The heart of this cipher is the DES function, f. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.



->DES Algorithm Steps

To put it in simple terms, DES takes 64-bit plain text and turns it into a 64-bit ciphertext.

The algorithm process breaks down into the following steps:

1. The process begins with the 64-bit plain text block getting handed over to an initial permutation (IP) function.
2. The initial permutation (IP) is then performed on the plain text.
3. Next, the initial permutation (IP) creates two halves of the permuted block, referred to as Left Plain Text (LPT) and Right Plain Text (RPT).
4. Each LPT and RPT goes through 16 rounds of the encryption process.
5. Finally, the LPT and RPT are rejoined, and a Final Permutation (FP) is performed on the newly combined block.
6. The result of this process produces the desired 64-bit ciphertext.

The encryption process step (step 4, above) is further broken down into five stages:

1. Key transformation
2. Expansion permutation
3. S-Box permutation

4. P-Box permutation
5. XOR and swap

For decryption, we use the same algorithm, and we reverse the order of the 16 round keys

Example :

If we take the plaintext message "8787878787878787", and encrypt it with the DES key "0E329232EA6D0D73", we end up with the ciphertext "0000000000000000". If the ciphertext is decrypted with the same secret DES key "0E329232EA6D0D73", the result is the original plaintext "8787878787878787".

This example is neat and orderly because our plaintext was exactly 64 bits long. The same would be true if the plaintext happened to be a multiple of 64 bits. But most messages will not fall into this category. They will not be an exact multiple of 64 bits (that is, an exact multiple of 16 hexadecimal numbers).

For example, take the message "Your lips are smoother than vaseline". This plaintext message is 38 bytes (76 hexadecimal digits) long. So this message must be padded with some extra bytes at the tail end for the encryption. Once the encrypted message has been decrypted, these extra bytes are thrown away. There are, of course, different padding schemes--different ways to add extra bytes. Here we will just add 0s at the end, so that the total message is a multiple of 8 bytes (or 16 hexadecimal digits, or 64 bits).

The plaintext message "Your lips are smoother than vaseline" is, in hexadecimal,

"596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A".

(Note here that the first 72 hexadecimal digits represent the English message, while "0D" is hexadecimal for Carriage Return, and "0A" is hexadecimal for Line Feed, showing that the message file has terminated.) We then pad this message with some 0s on the end, to get a total of 80 hexadecimal digits:

"596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A0000".

If we then encrypt this plaintext message 64 bits (16 hexadecimal digits) at a time, using the same DES key "0E329232EA6D0D73" as before, we get the ciphertext:

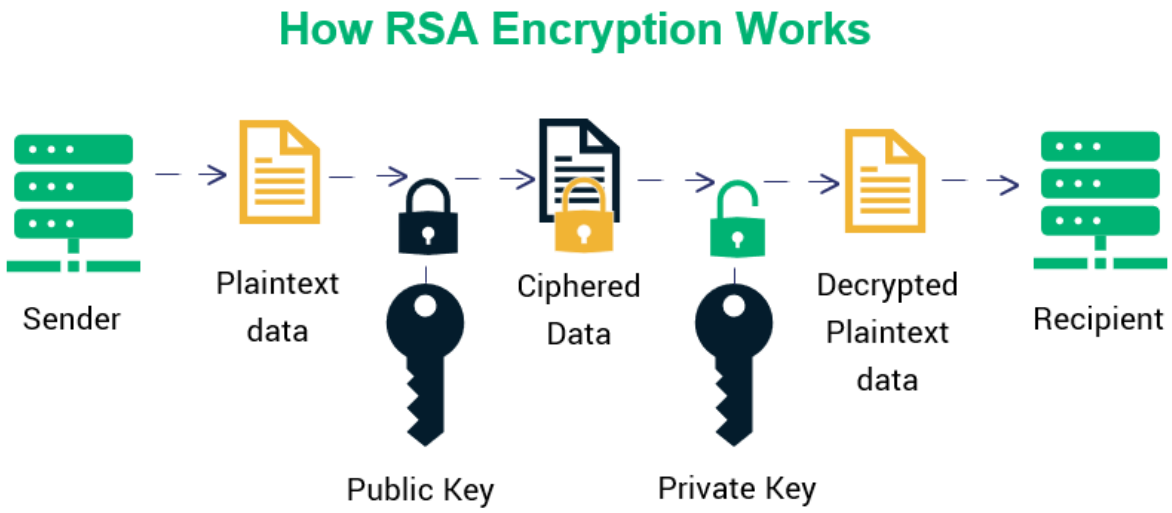
"C0999FDDE378D7ED 727DA00BCA5A84EE 47F269A4D6438190 9DD52F78F5358499 828AC9B453E0E653".

This is the secret code that can be transmitted or stored. Decrypting the ciphertext restores the original message "Your lips are smoother than vaseline".

RSA (Rivest-Shamir-Adleman)

RSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes, the Public Key is given to everyone and the Private key is kept private.

The idea! The idea of RSA is based on the fact that it is difficult to factorise a large integer. The public key consists of two numbers where one number is a multiplication of two large prime numbers. And private keys are also derived from the same two prime numbers. So if somebody can factorise the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible task.



Public key encryption algorithm:

The Public Key encryption algorithm is also called the Asymmetric algorithm. Asymmetric algorithms are those algorithms in which sender and receiver use different keys for encryption and decryption. Each sender is assigned a pair of keys:

- Public key
- Private key

The Public key is used for encryption, and the Private Key is used for decryption. Decryption cannot be done using a public key. The two keys are linked, but the private key cannot be derived from the public key. The public key is well known, but

the private key is secret and it is known only to the user who owns the key. It means that everybody can send a message to the user using the user's public key. But only the user can decrypt the message using his private key.

RSA algorithm uses the following procedure to generate public and private keys:

- Select two large prime numbers, **p** and **q**.
- Multiply these numbers to find **n = p x q**, where **n** is called the modulus for encryption and decryption.
- Choose a number **e** less than n, such that n is relatively prime to **(p - 1) x (q -1)**. It means that **e** and **(p - 1) x (q - 1)** have no common factor except 1.
- Choose "e" such that 1<e < φ (n), e is prime to φ (n),
  gcd (e,d(n)) =1.
- If **n = p x q**, then the public key is <e, n>. A plaintext message **m** is encrypted using public key <e, n>. To find ciphertext from the plain text following formula is used to get ciphertext C.
  C = m^e mod n
  Here, **m** must be less than **n**. A larger message (>n) is treated as a concatenation of messages, each of which is encrypted separately.
- To determine the private key, we use the following formula to calculate the d such that:
  De mod {(p - 1) x (q - 1)} = 1
  Or
  De mod φ (n) = 1
- The private key is <d, n>. A ciphertext message **c** is decrypted using the private key <d, n>. To calculate plain text **m** from the ciphertext c the following formula is used to get plain text m.
  m = cd mod n

Example

This example shows how we can encrypt plaintext 9 using the RSA public-key encryption algorithm. This example uses prime numbers 7 and 11 to generate the public and private keys.

Step 1: Select two large prime numbers, p, and **q**.

p = 7

q = 11

Step 2: Multiply these numbers to find **n = p x q,** where **n** is called the modulus for encryption and decryption.

First, we calculate

n = p x q

n = 7 x 11

n = 77

Step 3: Choose a number **e** less than **n**, such that n is relatively prime to **(p - 1) x (q -1).** It means that **e** and **(p - 1) x (q - 1)** have no common factor except 1. Choose "e" such that 1<e < φ (n), e is prime to φ (n), gcd (e, d (n)) =1.

Second, we calculate

φ (n) = (p - 1) x (q-1)

φ (n) = (7 - 1) x (11 - 1)

φ (n) = 6 x 10

φ (n) = 60

Let us now choose relative prime e of 60 as 7.

Thus the public key is <e, n> = (7, 77)

Step 4: A plaintext message **m** is encrypted using public key <e, n>. To find ciphertext from the plain text following formula is used to get ciphertext C.

To find ciphertext from the plain text following formula is used to get ciphertext C.

$C = m^e \bmod n$

$C = 9^7 \bmod 77$

C = 37

Step 5: The private key is <d, n>. To determine the private key, we use the following formula d such that:

$D_e \bmod \{(p - 1) \times (q - 1)\} = 1$

7d mod 60 = 1, which gives d = 43

The private key is <d, n> = (43, 77)

Step 6: A ciphertext message **c** is decrypted using the private key <d, n>. To calculate plain text **m** from the ciphertext c the following formula is used to get plain text m.

$m = c^d \bmod n$

$m = 37^{43} \bmod 77$

$m = 9$

In this example, Plain text = 9 and the ciphertext = 37

Diffie-Hellman Algorithm

The Diffie-Hellman algorithm is one of the most important algorithms used for establishing a shared secret. At the time of exchanging data over a public network, we can use the shared secret for secret communication. We use an elliptic curve for generating points and getting a secret key using the parameters.

1. We will take four variables, i.e., P (prime), G (the primitive root of P), and a and b (private values).
2. The variables P and G both are publicly available. The sender selects a private value, either a or b, for generating a key to exchange publicly. The receiver receives the key, and that generates a secret key, after which the sender and receiver both have the same secret key to encrypt.

Step by step for user1 (sender) and user2 (receiver)

steps	User1	User2
1	P, G => available public keys.	P, G => available public keys.
2	a is selected as a private key.	b is selected as a private key.
3	Eq. to generate key: x=Ga modP	Eq. to generate key: y=Gb modP
4	After exchanging keys, user1 receives key y.	After exchanging keys, user2 receives key x.
5	User1 generates a secret key by using the received key y:	User2 generates a secret key by using the received key x:

	ka=ya modP	kb=xb modP

Algebraically, 5th step can be shown as follows:
$k_a=k_b$
It means that both the users have the symmetric secret key to encrypt.

Example:

1. User1 and User2 get public keys P = 33 and G = 8.
2. User1 selects a as a private key, i.e., 3, and User2 selects b as a private key, i.e., 2.
3. User1 calculate the public value:
   x=(83 mod 33)=512mod33=17
4. User2 calculate the public value:
   y=(82 mod33)=64mod33= 31
5. User1 and User2 exchange public keys, i.e., 17 and 31.
6. User1 receives public key y = 31 and User2 receives public key x = 17.
7. User1 and User2 calculate symmetric keys:
   User1: ka=ya modP=313 mod33=29791 mod33=25
   User2:kb=xb modP=172 mod33=289mod33=25
8. 25 is the shared secret.

References:

1. DES algorithm
2. Diffie-Hellman algorithm
3. Network security mechanism
4. Symmetric and Asymmetric key cryptography