An Overview on Embedded Systems

A embedded system, is a computer system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints. It is designed to perform a specific task and it is embedded as a part of a complete device often including hardware and mechanical parts. Some examples of portable devices can be digital watches, MP3 player, factory controllers and larger ones such as stationary installations like traffic lights, factory controllers or even hybrid vehicles.

Embedded systems have been widely developed in comparison to years ago. These devices interact with the real world and are connected to the internet and so they are open to attacks, hackers and multiple threats. These attacks may result in physicial side effect as potential damages or personal injury. This is why security in these systems becomes a must. (link to wiki https://en.wikipedia.org/wiki/Embedded_system)

One of the considerations in security for embedded systems, due to their rapid growth are authentication for users, intrusion detection and so, security requirements have become critical. Main issues in security are encountered when the data is routed through Ethernet, Wi-fi, WiMAX or Bluetooth. So, one possibility of adding security to them is by implementing cryptography algorithms in software. However, they are many challenges and requirements to it. First, performance of these algorithms is key, because they must run at the same rate as the data transmission so it can remain invisible to the user and can't be too fast either, because it might means higher production costs. Also, guranteeing security is a challenge. Encryption algorithm has limited physicial security as the secure storage of keys is difficult in most operative systems. Hardware encryption devices can be securely encapsulated to prevent attackers from tampering with the system, that means custom hardware is the way to go. However, hardware solution come with a reduced flexibility and again higher costs. Finally, many of the security protocols decouple the choice of cryptographic algorithm from the design of the protocol. Users negotiate the choice of algorithm to use for a particular secure session. This means, that new devices, to support these applications, should support multiple cryptographic algorithms and protocols or *algorithm agile*, that is be able to select from a variety of algorithms. (link to http://web.itu.edu.tr/~orssi/projeler/ESD_crypto.html)

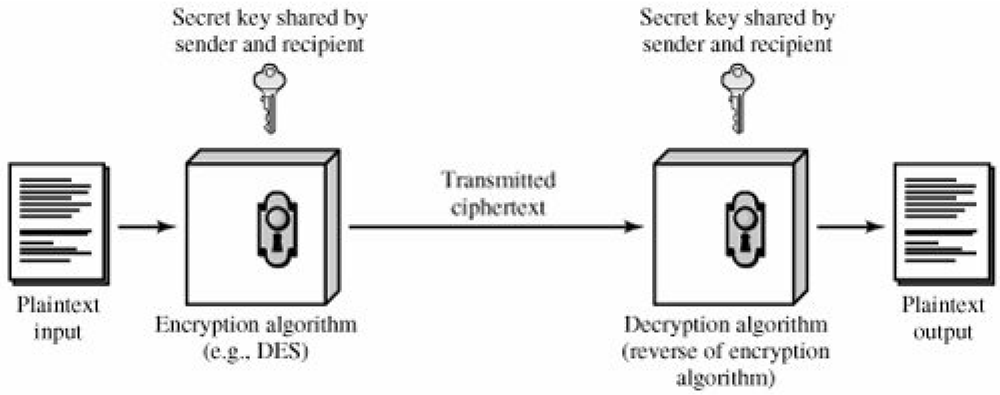
Criptography

Crypthography is the study and practice of techniques for secure communication, is about constructing and analyzing protocols to prevent third parties to read private messages. As the technology advances, security issues became a mayor concern, thus increasing the use of cryptography as a method for maintaining data confidentiality, integrity and authentication.

Simmetric Cryptography

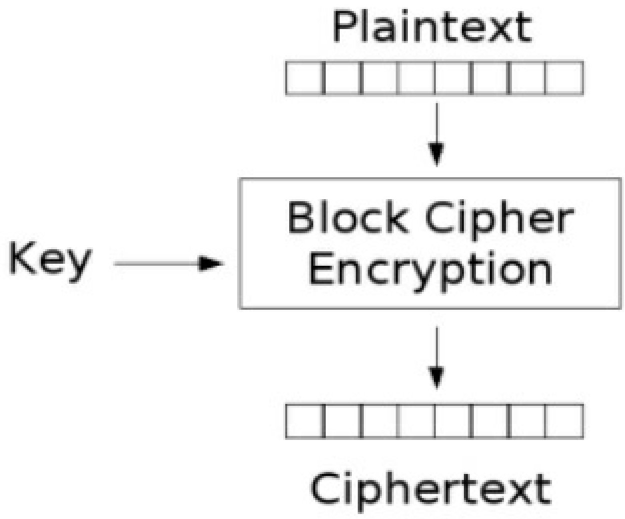
Also known as single key encryption, in where encryption and decryption processes are performed by using the same key. It contains 5 elements:

- Plaintext
- Encryption Algorithm
- Private Key
- Cyphertextx
- Decryption Algorithm


Secret key shared by sender and recipient
Secret key shared by sender and recipient
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Block Cipher Principles

Block Cipher is a type of symmetric encription algorithm that operates on fixed-length groups of bits of plaintext, called blocks and transform them into ciphertext block of the same length. Due that it is symmetric cryptography, it uses the same private key for encryption and decryption. The block size is usually 64 or 128 bits long.



A block cipher consists of two paired algorithms, one for encryption, E, and the other for decryption, D. Both algorithms accept two inputs: an input block of size n bits and a key of size k bits; and both yield an n-bit output block. The decryption algorithm D is defined to be the inverse function of encryption, $D=E^{-1}$.

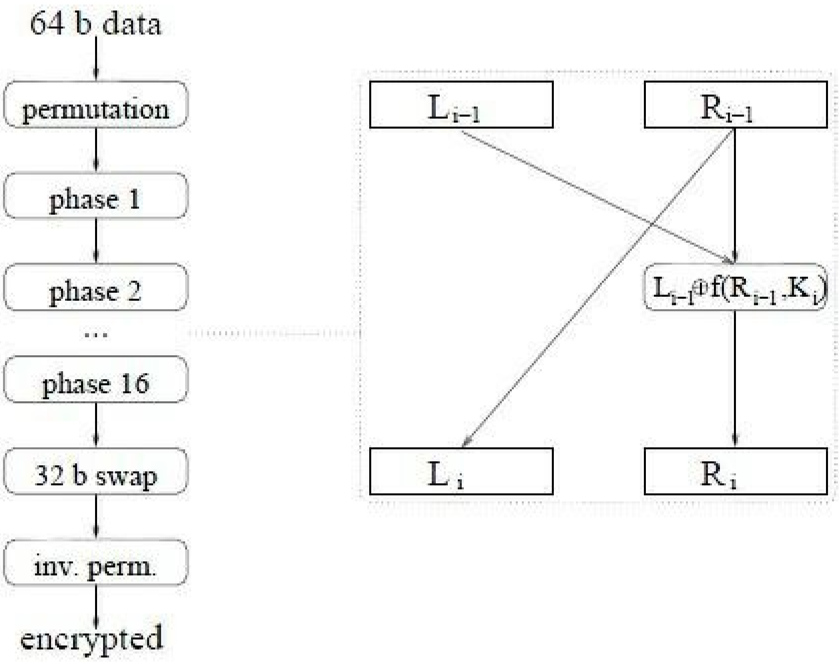
Data Encryption Standard or DES Algorithm

Developed in the early 1970s at IBM and based on an earlier design by Horst Feistel, the algorithm was submitted to the National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of sensitive, unclassified electronic government data. In 1976, after consultation with the National Security Agency (NSA), the NBS eventually selected a slightly modified version (strengthened against differential cryptanalysis, but weakened against brute force attacks), which was published as an official Federal Information Processing Standard (FIPS) for the United States in 1977. DES is now considered insecure for many applications and so it has been replaced by AES algorithm.

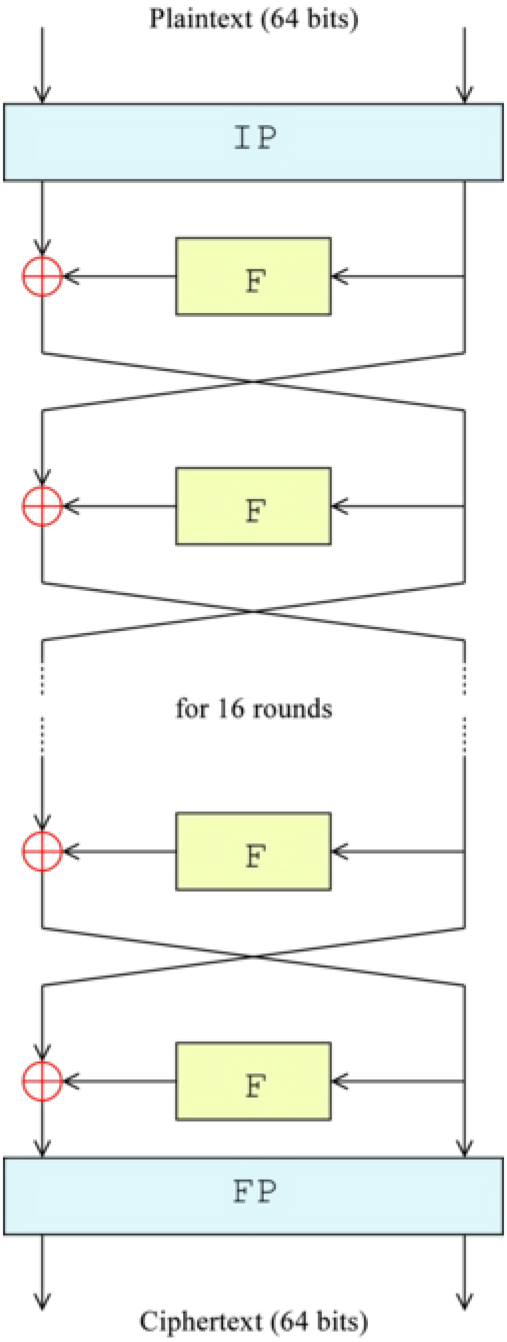
DES encryption process is to transform a 64 bit input into a 64 bit output, uses a 64 bit key, though only 56 are actually used by the algorithm and the 8 remain are used for checking parity and are discarded, so the effective key lenght is 56 bits. Des algorithm envolves a sixteen step process in order to do the transformation, called rounds, the first and last one, is permutation (*IP* and *FP* in the figure below), then, the block is broken into two blocks of 32 bit lenght blocks ($L_i, R_i)$. Both block use a individual key:

$$K_i L*i :=R*{i-1}; R*i:=L*{i-1} \oplus f(R_{i-1, K_i})$$

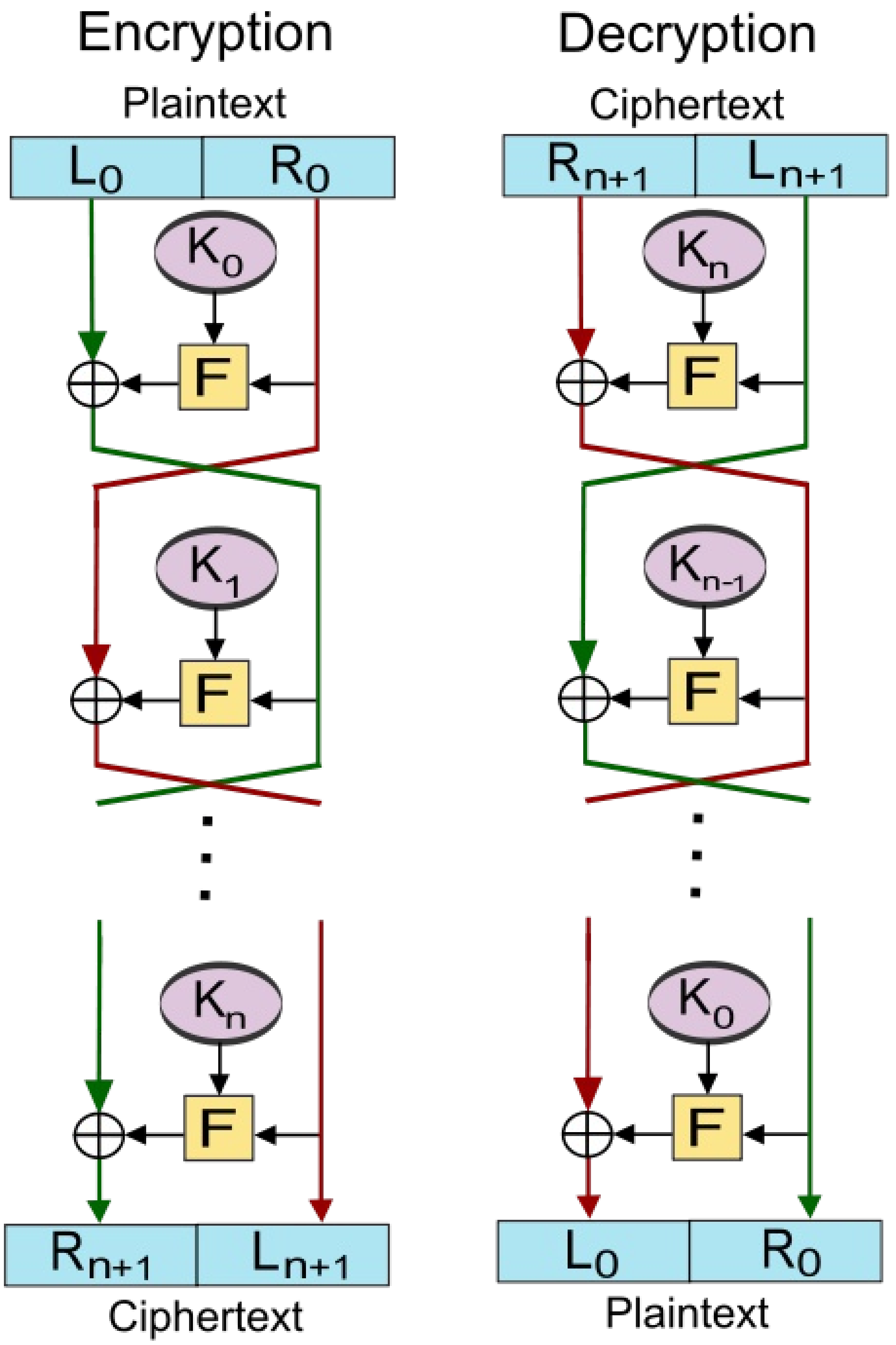
As shown in the figure below.



In the second image, it's possible to see the 16 rounds and the criss-crossing is known as the Feistel scheme.



The Feistel Scheme used in DES can be summarized in the following image, with K_n being the *K-th* key of the round n



DES Rounds

1. **Permutation (IP or Initial Permutation)**

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1

59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

The matrix represents the vector of the first permutation of the 64 bit block; where for instace the first bit takes the 58th bit spot, the seventh bit takes the position number 10 and so on.

1. **Permutation (FP or Final permutation):** $FP^1$

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

This is the inverse of the initial permutation and the final step in the rounds.

1. **Expansion function**

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

In this step, the block has been splited in two parts already ($L$ y $R$), and the new blocks are $32$ bits each. The expansion function duplicates some of the bits and the output size increases from $32$ to $48$ bits.

Key Generation

1. **Permuted Choice I** (PC - 1)

Left Block:

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36

Right Block:

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36

Here only $56$ bits are selected and splited into two blocks, while the remainding $8$ are used for parity checking

1. **Permuted Choice II** (PC - 2)

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Both blocks are shifted one or two bits to the left and only $48$ bit are selected ($24$ from each block) for the subkey for each round.

Finally, as it was pointed out before, the decryption process is donde by using the reverse of the operations.

Triple DES

Triple Des or Triple Data Encryption Algorithm is a symmetric-key block cypher which applies the DES algorithm three times to each data block.

It is believed to be more secure than DES only, because triple DES increases the size of the key and so protects it against brute force attacks, though there has been studies about theoretical attacks to it.

AES: Advanced Encryption Standard

AES became a standard in 2002 and since then, it is considered as the most popular and secure symmetric system used. Some of the improvements, in comparison to it's predecesor DES, are:

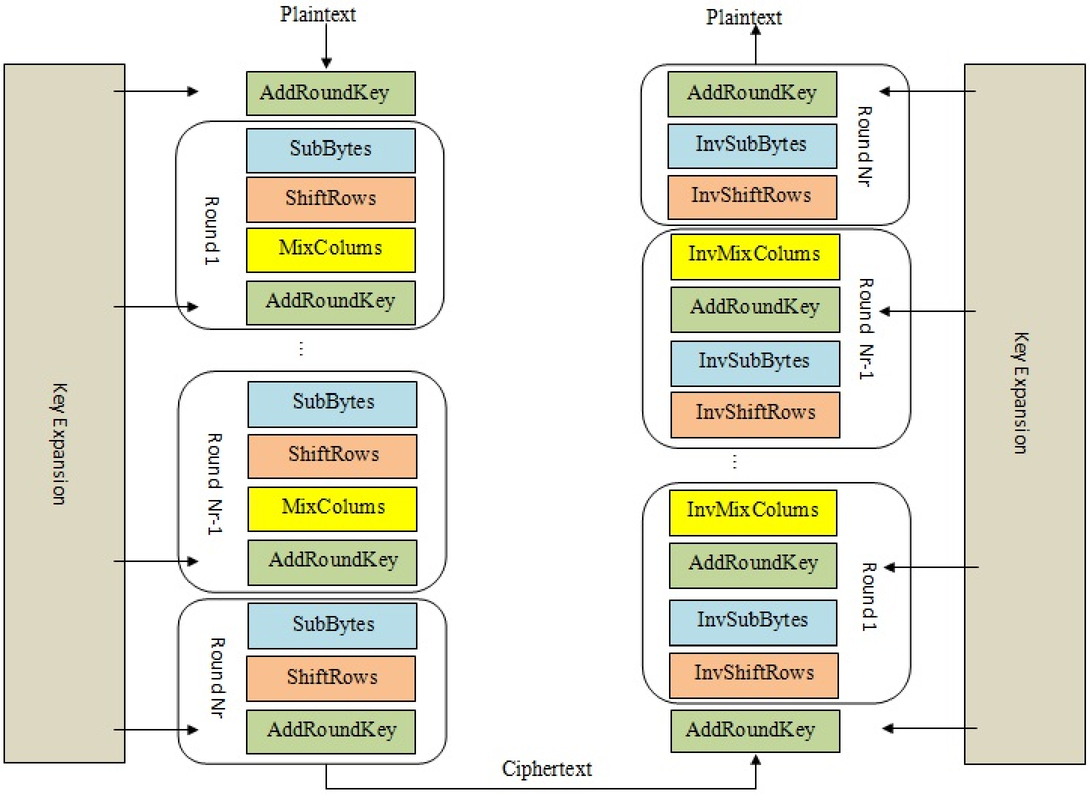
1. Faster in software adn hardware
2. Relatively easy implementation
3. Requires few memory
4. It's a Substitution-permutation network (DES is a Feistel network)
5. Larger key ($128$, $192$ or $256$ bits size)

AES Algorithm

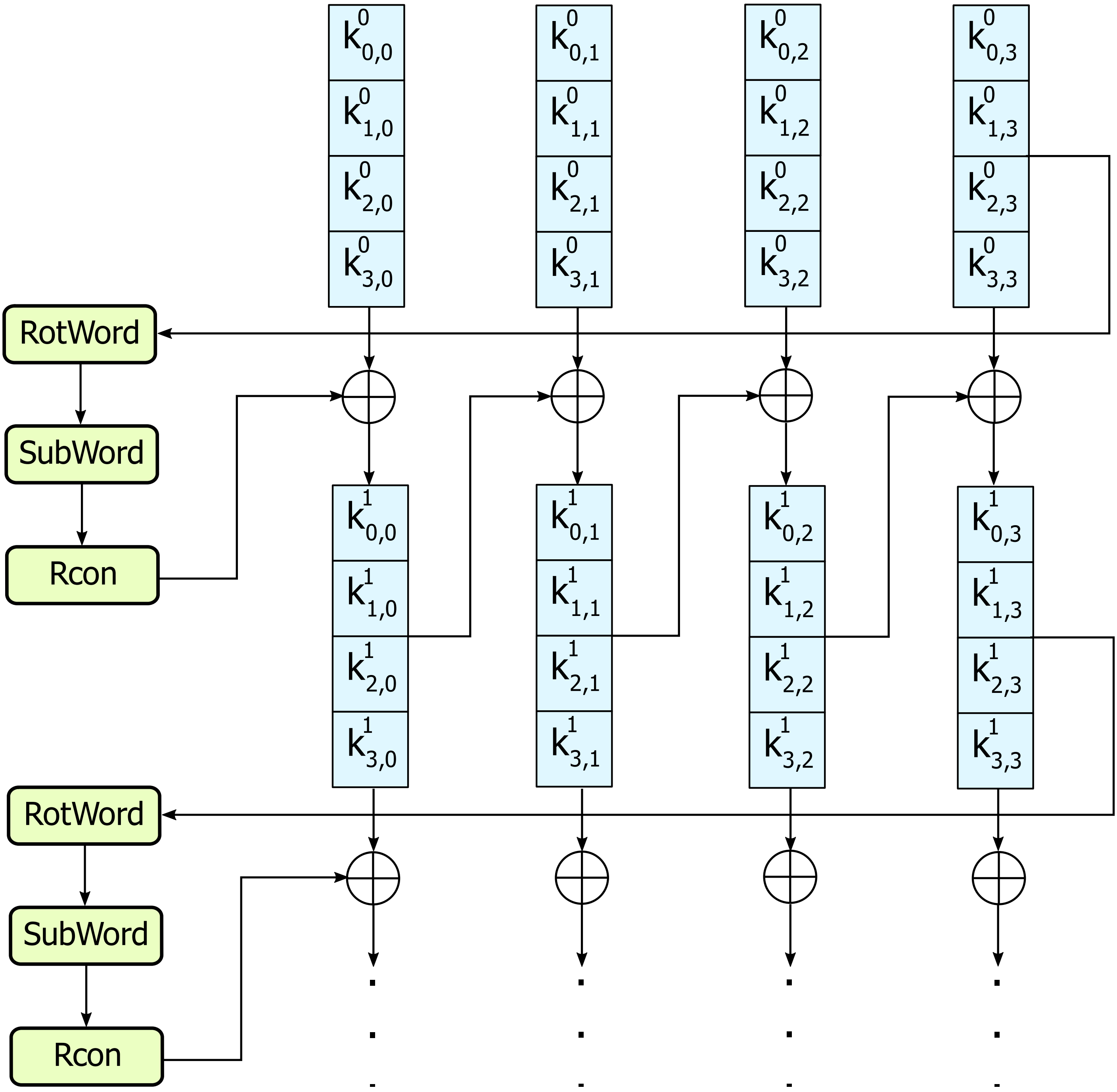
The plaintext block size used is $128$ bits (in the standard though other versions use a larger size) and operates in a $4$ x $4$ matrix of bytes known as *state*. The number of rounds depends on the key lenght, as shown in the table #

Rounds	Key Length
10	16
12	24
14	32

In the figure # are shown the steps to both encryption and decryption for each of the rounds:



1. Key Expansion The key expansion uses a key schedule to expand a short key into a number of separate round keys. AES processes the data block on each of the rounds using substitution and permutations. The output is an array of forty four 32-bit words. The Rijndael key schedule is shown in the figure # for a 128-bit key.



1. Initial Round

    i. AddRound Key: each byte of the state is combined with a block of the round key using bitwise xor.

2. Rounds

    i. SubBytes: a non-linear substitution step where each byte is replaced with another according to a lookup table.
    ii. ShiftRows: a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.
    iii. MixColumns: a mixing operation which operates on the columns of the state, combining the four bytes in each column.
    iv. AddRoundKey.

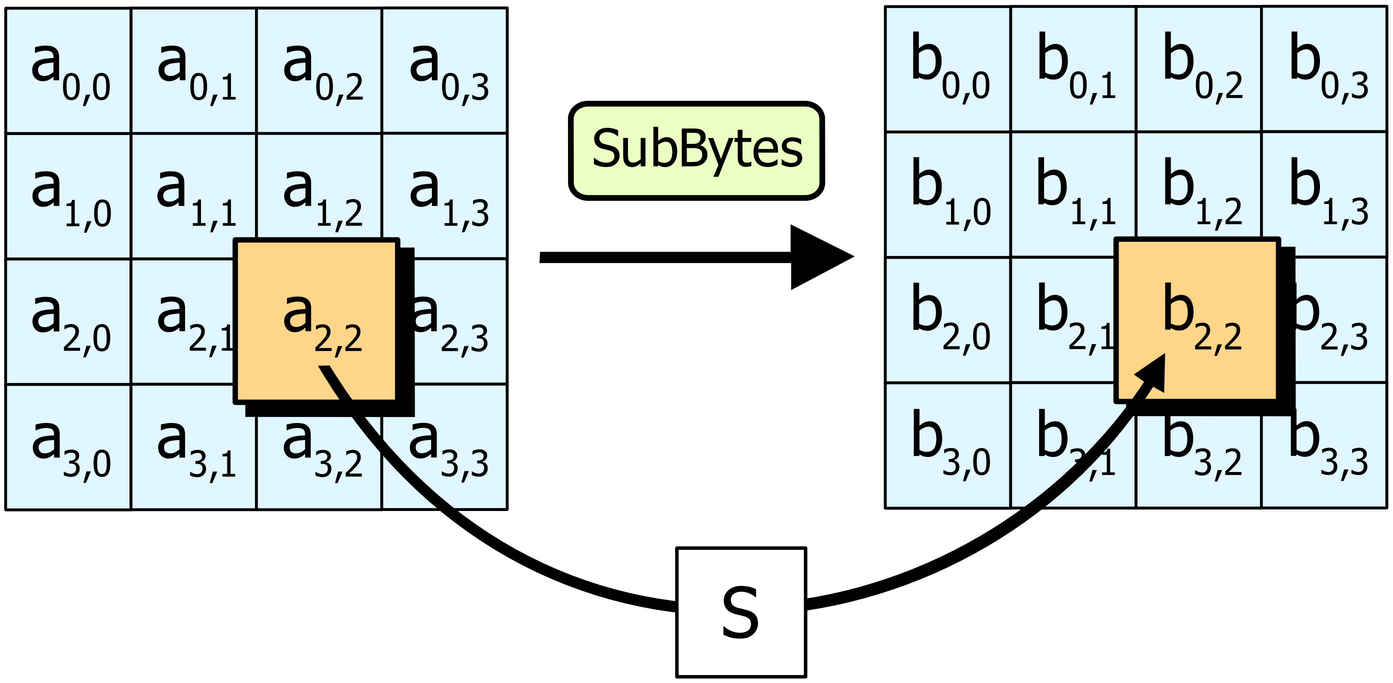
3. Final Round

    i. SubBytes
    ii. ShiftRows
    iii. AddRoundKey

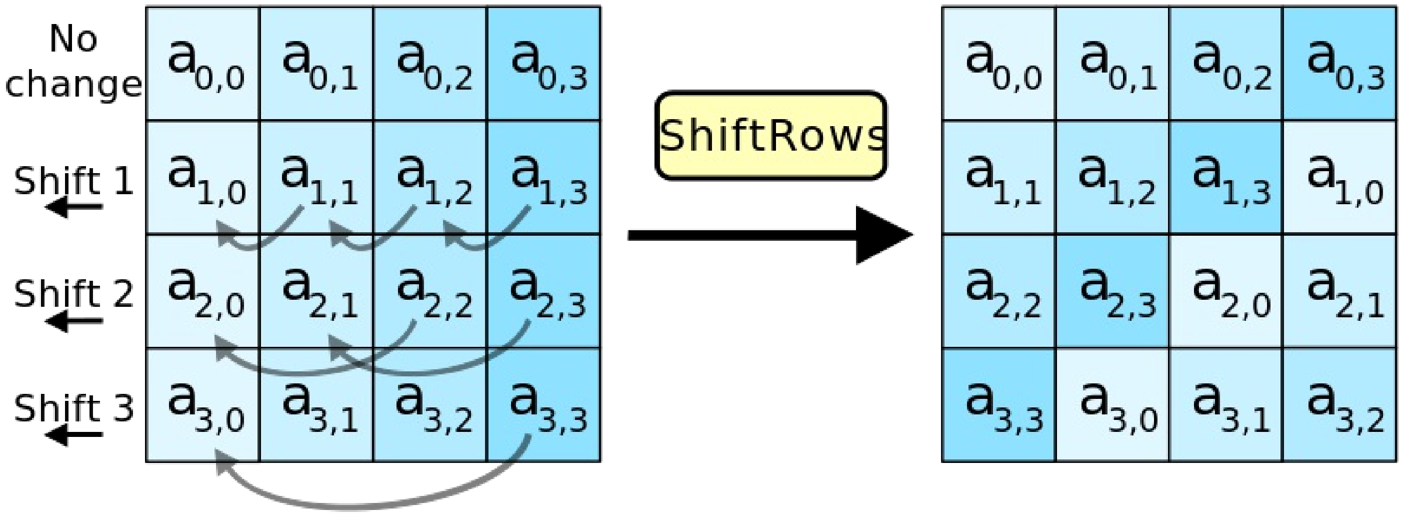
The following images are a graphic description of the four permutation and substitution steps:

alt text

AddRoundKey: The subkey is combined with its corresponding byte the state (both same size) using the **XOR** operation as shown in figure #. Being *a* the *state*, *k* the subkey matrix and *b* the result of the operation.


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

SubBytes: It replaces the elements from the state using a 8 bit *Substitution box* or *S-Box* (fixed table). This step provides the AES of the non-linearity.

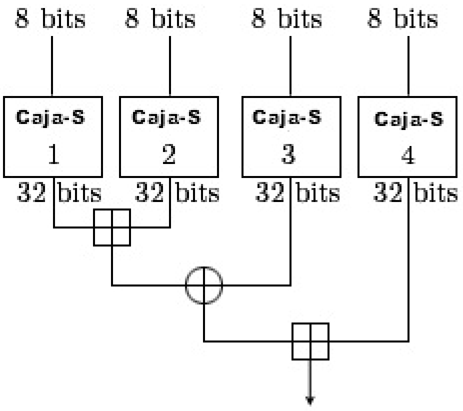

No change
Shift 1
Shift 2
Shift 3
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
ShiftRows
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,1}$ $a_{1,2}$ $a_{1,3}$ $a_{1,0}$
$a_{2,2}$ $a_{2,3}$ $a_{2,0}$ $a_{2,1}$
$a_{3,3}$ $a_{3,0}$ $a_{3,1}$ $a_{3,2}$

ShiftRows: It shiftes the bytes into a specific direction a certain number of steps as it shows the figure #

alt text **MixColumns**: The four bytes of each column of the state are combined using an invertible linear transformation. It multiplies the four bytes from the state to a fixed polynomial $c(y)$.

Finally, for decryption it uses the expanded key in reverse order. AddRoundKey step is the same as encryption, though the whole process of decryption does differe from the encryption.

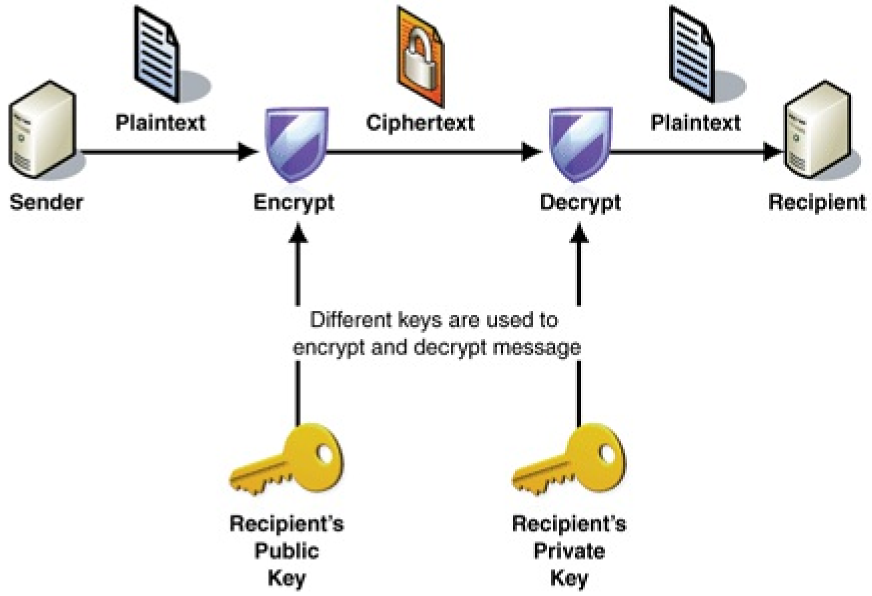
#Blowfish Blowfishis a symmetrical block cipher designed in 1993, came up as an alternative to DES, however, AES is still more popular in present day. The key sizes for the algorithm are 32 to 448 bits and the block sizes are 64 bits, unlike AES where the blocks are larger. This algorithm is not a registered patent, it can be used free. Blowfish is a 16-round Feistel cipher and uses large (key-dependent) S-Boxes.



In the figure #, the Blowfish diagram is described:

1. Blowfish is a fast block cipher, except when changing keys, due to each new key requires pre-processing and makes it slower than other block ciphers. In some applications this is a benefit, because it provides an extra protection against *dictionary attacks*.
2. It needs only 5 KB of memory so its compact and easy to implement (not so much for embedded systems as early smartcards).
3. The encryption process consist in 16+1 phases, which contain an **XOR**, a $+$ and s S-Box operation.

#Asymmetric Cryptography Asymmetric Cryptography or public-key cryptography is a crypto system that uses pairs of keys, one for encryption and another for decryption; one is public, which is widely distributed while the other is private and only known by its owner.



Example of Asymmetric Cryptography

In this scheme, the sender encrypts the plaintext using the recipient public key (available to everybody) and for decryption, the recipient utilices his private key, but only the recipient can decrypt the message because he is the only one who knows his private key.

The public key crypto systems rely on mathematical problems that currently admit no efficient solution and they do not require a secure channel for the public key distribution.

Two of the best know uses of this type of crypto are:

1. Public Key encryption as shown in figure #
2. Digital Signatures, to verify the identity of the sender and the integrity of the message.

##RSA