SRH HOCHSCHULE HEIDELBERG

PROJECTREPORT

COMPARISON OF AES&DES

SUBMITTED TO:Dr.KARLIZSAK

ABHIJITH RADHAKRISHNAN MENON 11013632

NIKHILGOWDA SHIVASWAMY 11013382

RAKSHIT ADDERI ROHIT 11013631

AHMED ABDULMANEA 11013620

Date: 05/02/2020

Table of Contents

ABSTRACT

In this report we will be presenting a detail report on Advance Encryption System (AES) & Data Encryption System (DES). AES & DES both are symmetric key block cipher. The motive of this report is to compare AES & DES in means of there functions and security. At the end of report, we also mention advantages and disadvantages of both AES & DES systems.

INTRODUCTION

In this report we discuss two powerful Symmetric-key block ciphers, i.e Data Encryption Standard and Advanced Encryption Standard.

The *Data Encryption Standard (DES)* has been by far the most popular block cipher for most of the last 30 years. Even though it is nowadays not considered secure against a determined attacker because the DES key space is too small, it is still used in legacy applications. Furthermore, encrypting data three times in a row with DES — a process referred to as *3DES* or *triple DES* — yields a very secure cipher which is still widely used today. Perhaps what is more important, since DES is by far the best-studied symmetric algorithm, its design principles have inspired many current ciphers. Hence, studying DES helps us to understand many other symmetric algorithms.

In 1999 the US National Institute of Standards and Technology (NIST) indicated that DES should only be used for legacy systems and instead triple DES (3DES) should be used. Even though 3DES resists brute-force attacks with today's technology, there are several problems with it. First, it is not very efficient with regard to software implementations. DES is already not particularly well suited for software and 3DES is three times slower than DES. Another disadvantage is the relatively short block size of 64 bits, which is a drawback in certain applications, e.g., if one wants to built a hash function from a block cipher. Finally, if one is worried about attacks with quantum computers, which might become reality in a few decades, key lengths on the order of 256 bits are desirable. All these consideration led NIST to the conclusion that an entirely new block cipher was needed as a replacement for DES.

In 1997 NIST called for proposals for a new *Advanced Encryption Standard (AES)*. Unlike the DES development, the selection of the algorithm for AES was an open process administered by NIST. In three subsequent AES evaluation rounds, NIST and the international scientific community discussed the advantages and disadvantages of the submitted ciphers and narrowed down the number of potential candidates. In 2001, NIST declared the block cipher *Rijndael* as the new AES and published it as a final standard (FIPS PUB 197). Rijndael was designed by two young Belgian cryptographers.

Overview of AES:

AES is by now the most important symmetric algorithm in world. The AES cipher is almost identical to the block cipher Rijndael. In Rijndael block and key size vary between 128, 192 and 256 bits. However, In AES standard block size is standardized to 128 bits. Hence, only Rijndael with a block length of 128 bits is known as the AES algorithm.

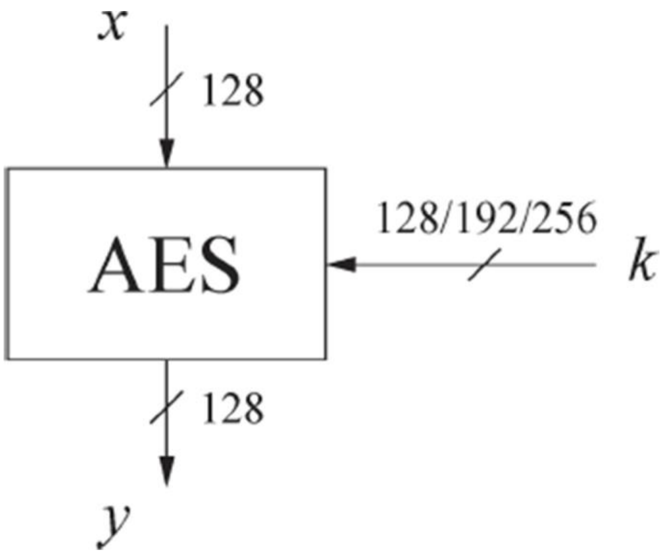


Fig 1.0: AES Input output parameters.

- In figure 1.0 the plane text is denoted as "*x*".
- The cipher text is denoted as "*y*"
- Key is denoted by "*k*"
- In contrast to DES, AES does not have Feistel structure.
- AES encrypts all 128 bits in one iteration.
- The AES block cypher has data path of 128-bit block size ("*x*")
- AES supports key size which varies between 128, 192 and 256 bit.

Key lengths	**# rounds = n_r**
128 bit	10
192 bit	12
256 bit	14

Table 1.0: Key lengths and number of rounds for AES

AES consists of layers, there are only 3 types of different layers. Each layer manipulates all 128 bits of data-path. Rather than DES, AES does not have a Feistel structure. Feistel systems don't scramble a whole piece for each iteration. AES, on the other hand, encrypts all 128 bits in one iteration. This is one reason why it has a comparably small number of rounds.

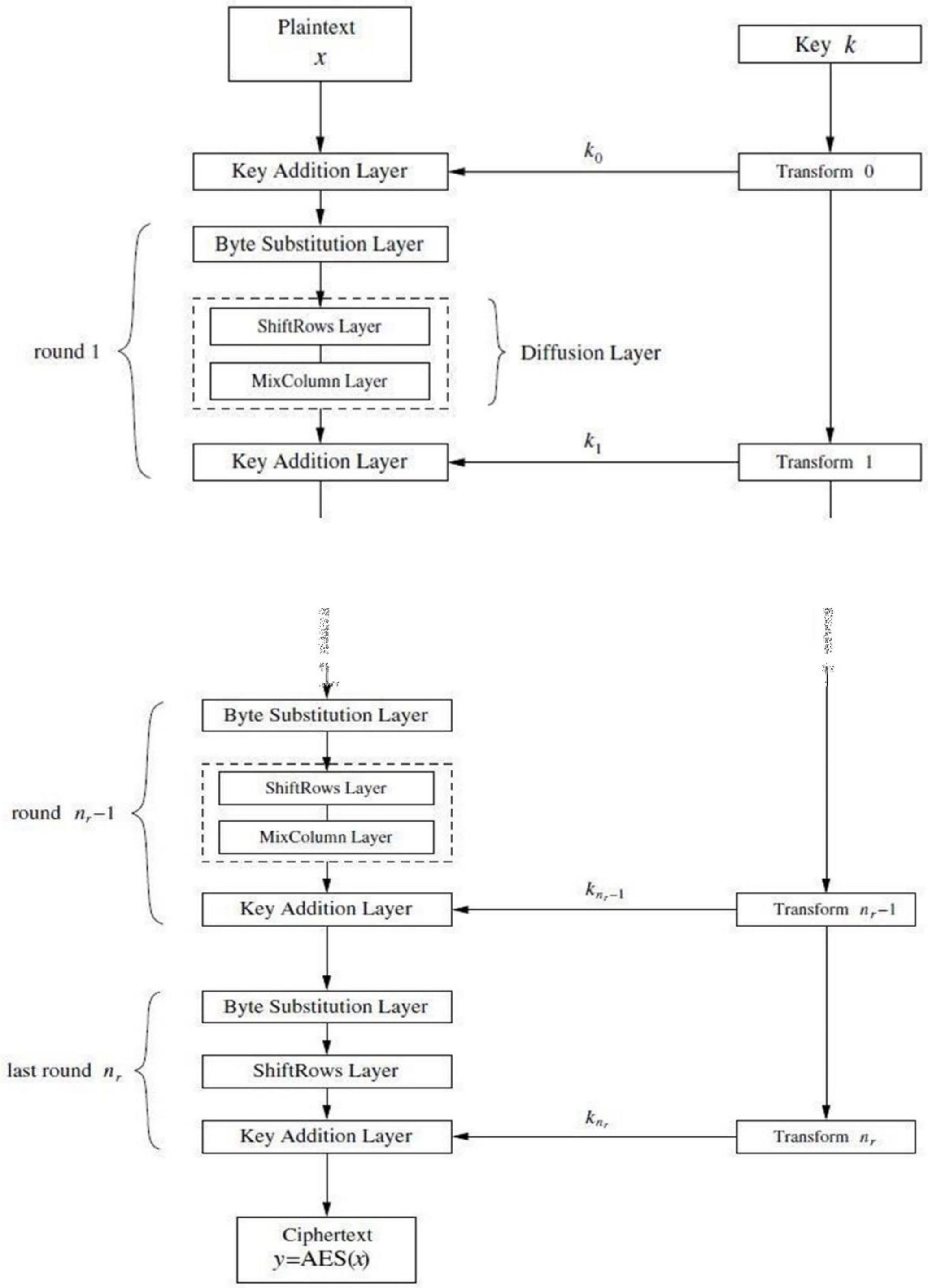


Fig 1.1: AES encryption block diagram.

As described in fig 1.1 AES has mainly three layers namely Byte substitution, Diffusion layer and Key addition layer for each round. There are n_r such rounds with exception of last round not containing mix column layer.

Since AES is Byte oriented Cipher, 128 bit input data-path is arranged in 16-Byts. In last round, Mix Columns transformation is omitted. Example: for 128 bit key length will have 9 such rounds and 10th round will be without Mix Columns transformation.

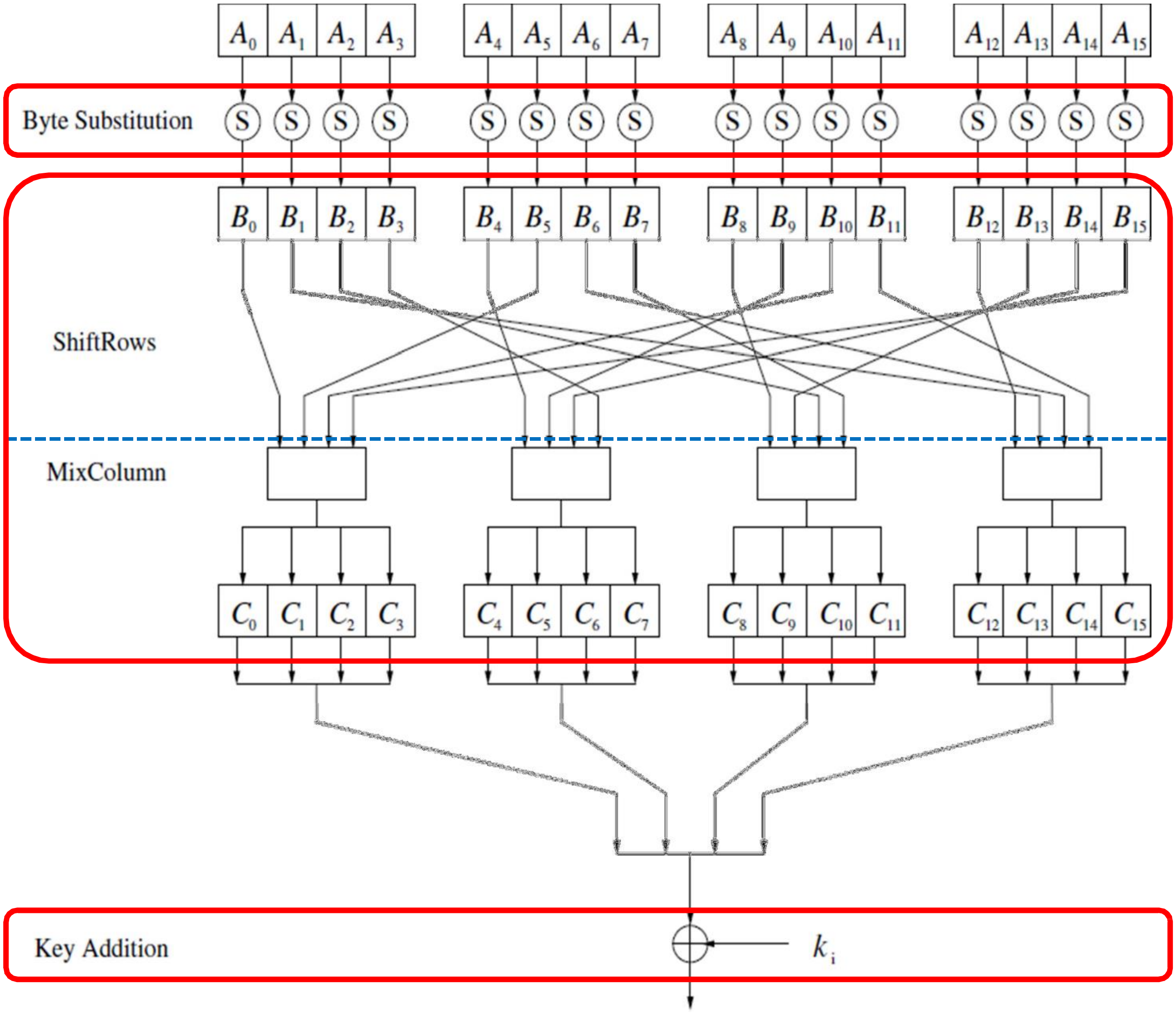

A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9 A_{10} A_{11} A_{12} A_{13} A_{14} A_{15}
Byte Substitution
S
B_0 B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8 B_9 B_{10} B_{11} B_{12} B_{13} B_{14} B_{15}
ShiftRows
MixColumn
C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8 C_9 C_{10} C_{11} C_{12} C_{13} C_{14} C_{15}
Key Addition
k_i

Fig 1.2: AES round function for rounds 1,2, . . . ,n-1.

We will discuss each layer in detail below.

1. Byte Substitution Layer:

Each element of the state is nonlinearly transformed using lookup tables with special mathematical properties. This introduces *confusion to the* plaintext, i.e., it assures that changes in individual state bits propagate quickly across the data path. The Byte Substitution layer consists of 16 S-Boxes. Unlike in DES, All S-Box are identical.

The S-Box is the only *nonlinear* element of AES, i.e., it holds that *ByteSub (A) +ByteSub (B) 6≠ ByteSub (A+B)* for two states A and B. S-Box substitution is a bijective mapping, i.e., each of the 28 = 256 possible input elements is one-to-one mapped to one output element. Bijective property is important for decipher as this allows us to uniquely reverse the S-Box.

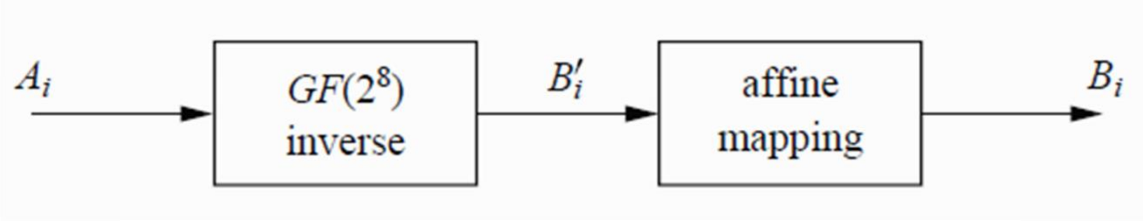


Fig 1.3: Mathematical description of S-Box.

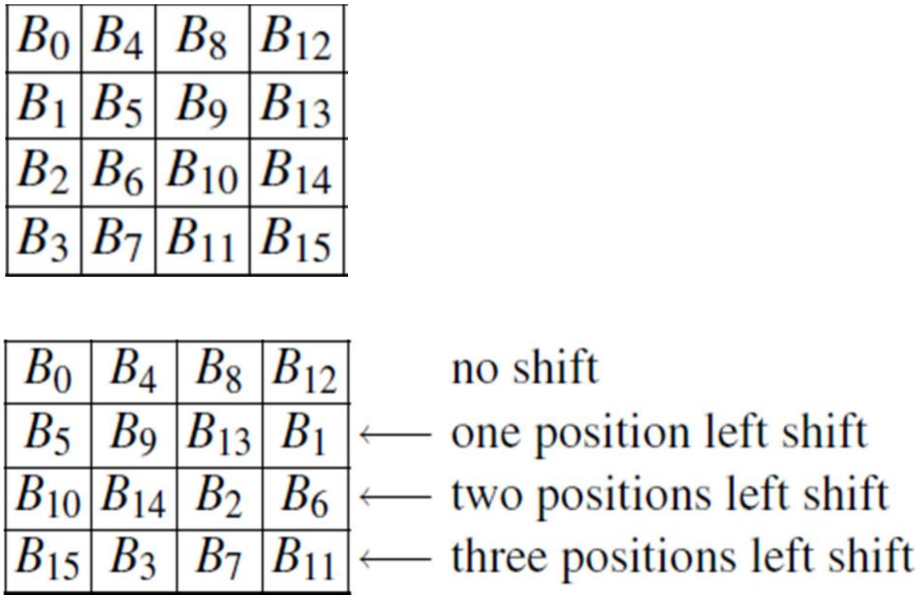
The first part of the substitution is a Galois field inversion, for each input element Ai, the inverse is computed: $B'_i = A_i^{-1}$, where both Ai and B_i are considered elements in the field GF (2^8) with the fixed irreducible polynomial.

In the second part of the substitution, each byte B'_i is multiplied by a constant bit-matrix. Followed by the addition of a constant 8-bit vector.

2. Diffusion layer:

In AES, the Diffusion layer consists of two sublayers, the ShiftRows transformation and the MixColumn transformation. Diffusion is the spreading of the influence of individual bits over the entire state. Diffusion layer performs linear operation. i.e. *ByteSub (A) +ByteSub (B) 6 = ByteSub (A+B)*.

ShiftRows Sublayer: Performs permutation of the data on a byte level. The ShiftRows transformation cyclically shifts the second row of the state matrix by one byte to left, the third row by two bytes to the left and the fourth row by three byte to the left. The first row is not changed by the ShiftRows transformation.



B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

B_0	B_4	B_8	B_{12}	no shift
B_5	B_9	B_{13}	B_1	⟵ one position left shift
B_{10}	B_{14}	B_2	B_6	⟵ two positions left shift
B_{15}	B_3	B_7	B_{11}	⟵ three positions left shift

Fig 1.4: ShiftRow operation.

MixColumn Sublayer: Matrix operation which combines (“mixes”) locks of four bytes. The combination of the ShiftRows and MixColumn layer makes it possible that after only three rounds every byte of the state matrix depends on all 16 plaintext bytes. Each 4-byte column is considered as a vector and multiplied by a fixed 4×4 matrix.

$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$

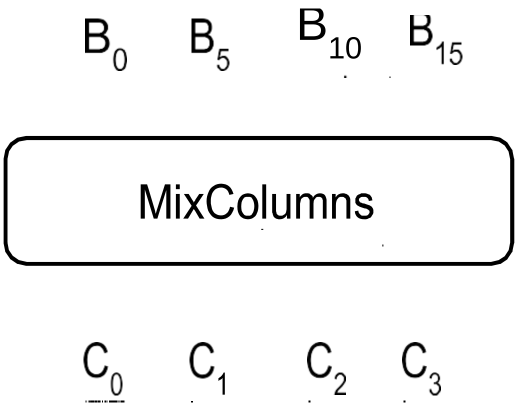


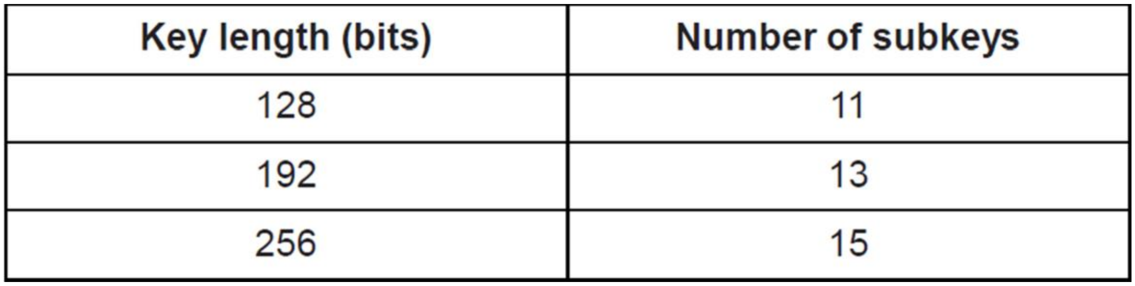
Fig 1.5 : MixColumn opertion.

All arithmetic in above matrix is done in the Galois field GF(28). where 01, 02 and 03 are given in hexadecimal notation. The second column of output bytes (C4,C5,C6,C7) is omputed by multiplying the four input bytes (B4,B9,B14,B3) by the same constant matrix.

3. Key Addition Layer:

The two inputs to the Key Addition layer are the current 16-byte state matrixand a Subkeywhich also consists of 16 bytes (128 bits). The two inputs are combined through a bitwise XOR operation. XOR operation is equal to addition in Galois field GF(2). The key schedule takes the original input key and derives the Subkeys used in AES. Subkeys are derived recursively from the original 128/192/256-bit input key. XOR addition of a subkey is used both at the input and output of AES. This process is sometimes referred to as key whitening.

Since one subkey is used at the beginning of first round, we have: Total number of subkeys= number of rounds + 1.



Key length (bits)	Number of subkeys
128	11
192	13
256	15

Fig 1.6: Table: number of subkeys generated based on key length

The AES subkeys are computed recursively, i.e. in order to derive subkey K_i, subkey K_i−1 must be known. AES key schedule is key oriented 1 word = 32 bits.

For 128 bit key, 11 subkeys are stored in a key expansion array with the elements W [0], . . . , W[43]. First subkey K_0 is the original key in AES. Leftmost word of a subkey W [4i], where i = 1, . . . , 10, is computed as:

W*4i+=W*4(i−1)++g(W*4i−1+) g() is nonlinear function with 4 byte input and output. Similarly, W*4i+j+ =W*4(i−1)++g(W*4i−1+) where j = 1,2,3 i.e. to calculate next 3 words.

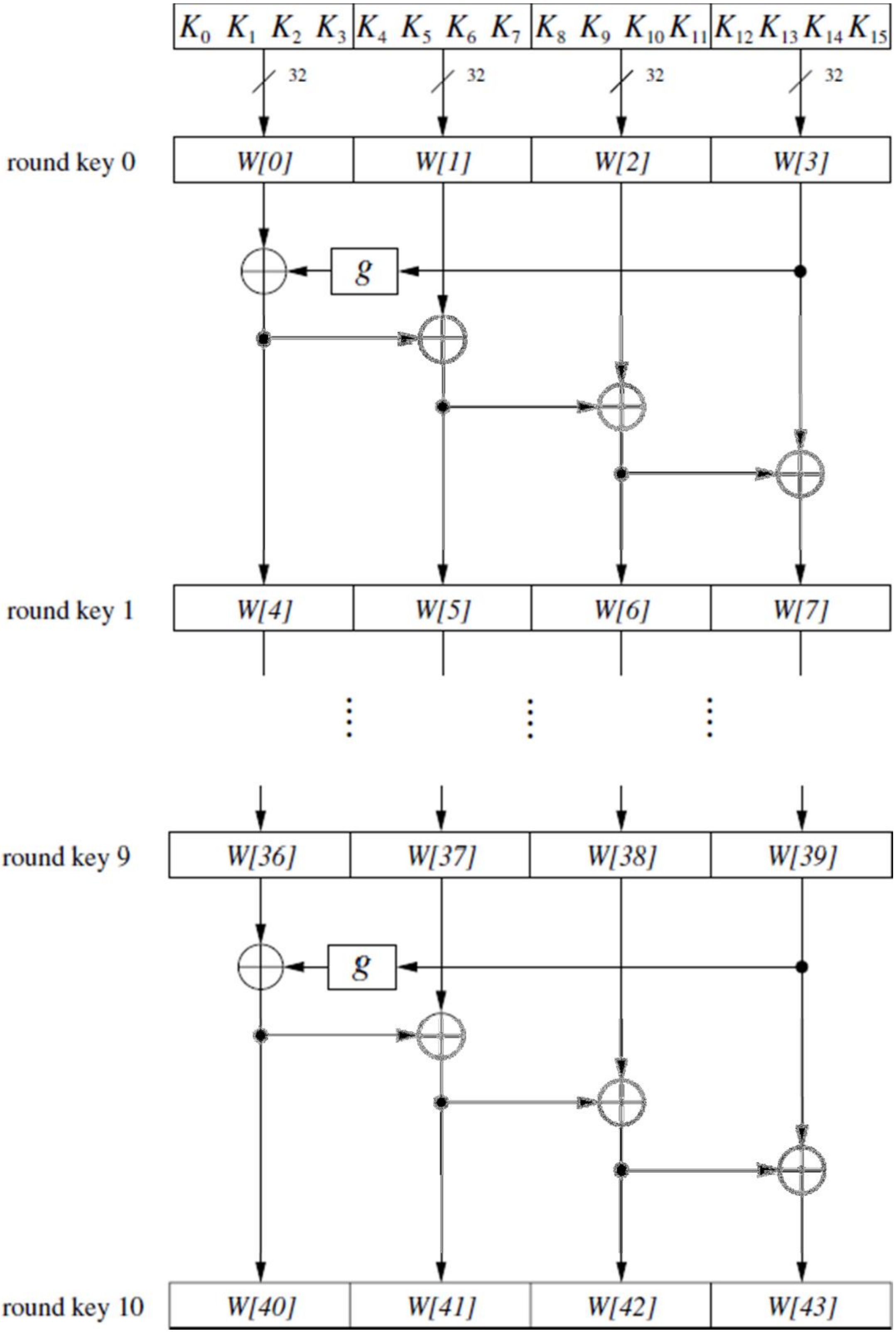


Fig 1.7: Operation of AES schedule for 128 bit key

Brute Force Attack on AES:

Brute force attack consists of an attacker which try as many number of combinations of key to decrypt the ciphertext. It is a systematic approach to verify the key by applying all possible guesses number of times.

Brute force attack works on simple principle of trying as many possible key combinations. As the key length increase the amount of time to do the search of correct key also increases exponentially.

In the world of embedded security and computation one of the most debatable topic is, weather the 128 bit symmetric key Advance Symmetric Encryption(AES) is computationally secure or not.

As the standard AES key length is 128 bit long so it means that it is a 128 bit long stream of 0 and 1. One can also say that it has probability of 50% for both. As we know brute force attack involves systematically checking all the possible key combinations until we arrive at correct one.

So in case of AES,

Key length = 128 bit

Possible combinations = (2^{128}) = 3.4 X 10^{38}

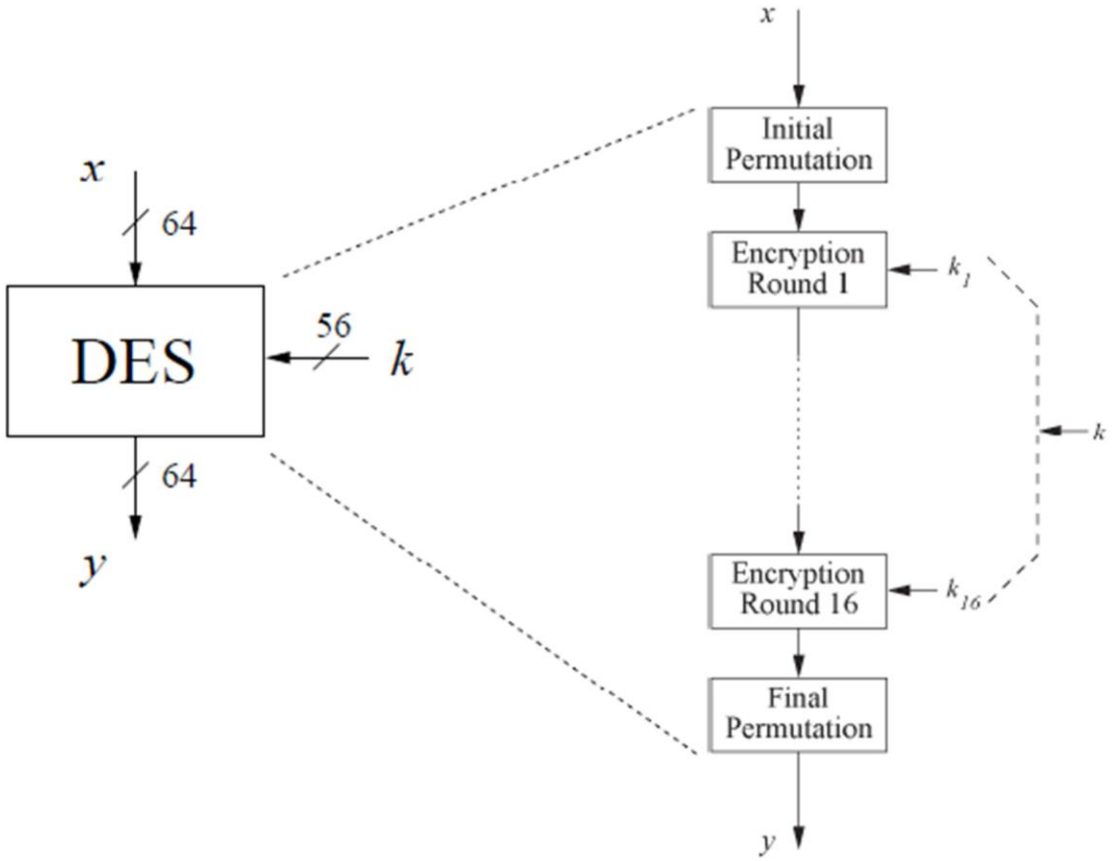
Even if you consider to use current world's fastest computer/supercomputer to break the AES encryption by brute force attack it will take almost billion years to do.

The bottom line is it is very hard to break AES through brute force attack as compared to *linear cryptanalysis*. *But* one cannot conclude with, brute force attack is not possible on AES. It also seen that AES is not been cracked yet by brute force attacks this might help to belief.

DES Overview:

DES is a cipher which encrypts blocks of length of 64 bits with a key of size of 56 bits. DES is a symmetric cipher, i.e., the same same key is used for encryption and decryption. DES is, like virtually all modern block ciphers, an iterative algorithm. For each block of plaintext, encryption is handled in 16 rounds which all perform the identical operation. Figure 3.4 shows the round structure of DES. In every round a different subkey is used and all subkeys *ki* are derived from the main key *k*.

The structure in the figure is called a *Feistel network*. It can lead to very strong ciphers if carefully designed. Feistel networks are used in many, but certainly not in all, modern block ciphers. In addition to its potential cryptographic strength, one advantage of Feistel networks is that encryption and decryption are almost the same operation.



DES requires only a reversed key schedule, which is an advantage in software and hardware implementations. After the initial bitwise permutation *IP* of a 64- bit plaintext *x*, the plaintext is split into two halves *L*0 and *R*0. These two 32-bit halves are the input to the Feistel network, which consists of 16 rounds. The right half *Ri* is fed into the function *f* . The output of the *f* function is XORed with the left 32-bit half *Li*. Finally, the right and left half are swapped.

It is crucial to note that the Feistel structure really only encrypts (decrypts) half of the input bits per each round, namely the left half of the input. The right half is copied to the next round unchanged. In particular, the right half is *not encrypted* with the *f* function. In order to get a better understanding of the working of Feistel cipher, the following interpretation is helpful: Think of the *f* function as a pseudorandom generator with the two input parameters *Ri*−1 and *ki*. The output of the pseudorandom generator is then used to encrypt the left half *Li*−1 with an XOR operation. As we saw in Chap. 2, if the output of the *f* function is not predictable for an attacker, this results in a strong encryption method.

The two aforementioned basic properties of ciphers, i.e., confusion and diffusion, are realized within the *f* -function. In order to thwart advanced analytical attacks, the *f* -function must be designed extremely carefully. Once the *f* -function has been designed securely, the security of a Feistel cipher increases with the number of key bits used and the number of rounds. Before we discuss all components of DES in detail, here is an algebraic description of the Feistel network for the mathematically inclined reader. The Feistel structure of each round bijectively maps a block of 64 input bits to 64 output bits (i.e., every possible input is mapped uniquely to exactly one output, and vice versa). This mapping remains bijective for some arbitrary function *f* , i.e., even if the embedded function *f* is not bijective itself. In the case of DES, the function *f* is in fact a surjective (many-to-one) mapping. It uses nonlinear building blocks and maps 32 input bits to 32 output bits using a 48-bit round key *ki*, with 1 ≤ *i* ≤ 16.

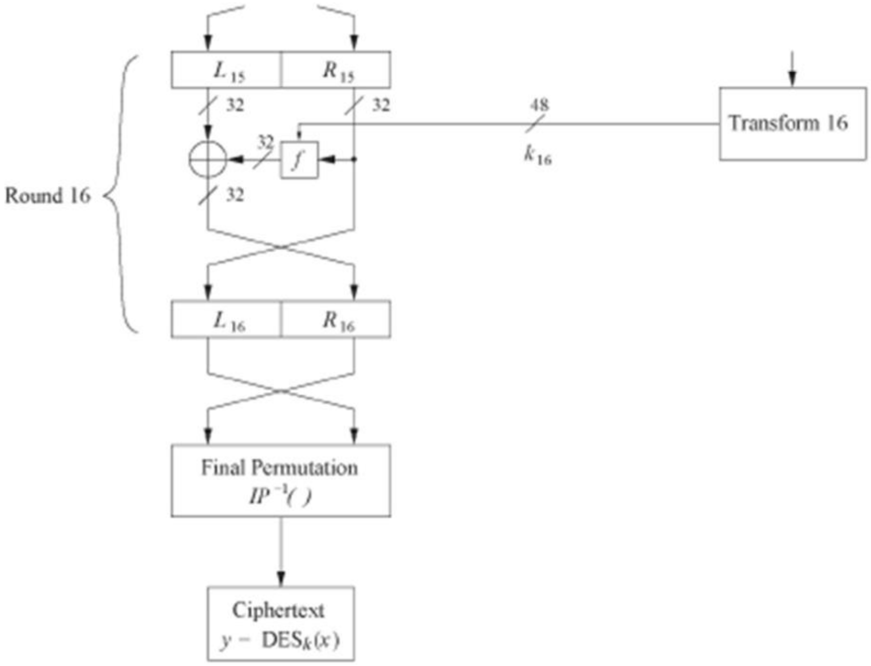
Internal Structure of DES :

The structure of DES as depicted in Fig. 3.5 shows the internal functions which we will discuss in this section. The building blocks are the initial and final permutation, the actual DES rounds with its core, the *f* -function, and the key schedule.

Initial and Final Permutation

The *initial permutation IP* and the *final permutation IP*−1 are bitwise permutations. A bitwise permutation can be viewed as simple crosswiring. Interestingly, permutations can be very easily implemented in hardware but are not particularly fast in software. Note that both permutations do not increase the security of DES at all. The exact rationale for the existence of these two permutations is not known, but it seems likely that their original purpose was to arrange the plaintext, ciphertext and bits in a bytewise manner to make data fetches easier for 8-bit data busses, which were the state-of-the-art register size in the early 1970s.

The details of the transformation *IP* are given in Fig. 3.8. This table, like all other tables in this chapter, should be read from left to right, top to bottom. The table indicates that input bit 58 is mapped to output position 1, input bit 50 is mapped to the second output position, and so forth. The final permutation *IP*−1 performs the inverse operation of *IP*



Round 16
L_{15}
R_{15}
32
32
32
f
32
48
k_{16}
Transform 16
L_{16}
R_{16}
Final Permutation
$IP^{-1}(\ )$
Ciphertext
$y = \mathrm{DES}_k(x)$

DES – Advantages

- Avalanche Effect – a small modification in the plaintext will play a big difference in the cipher text, either in the char or bit.

- Completeness – individual bit or char in cipher text depends on the multiple bits or char of plaintext

- Same algorithm is used for Encryption and decryption – Function has to be reversed and key is implemented in opposite direction.

- DES is the base for 2DES & 3DES – Implement many times to a plaintext.

- 32 GB of data can be transferred with single encryption

- Time required to check all the possible keys at 50 billion keys per second – For a 56 bit keys: 400 days

- Easy to implement in hardware and software

- Easy to analyze.

DES – Disadvantage

- Insecure, Slow and no longer in use

- Less number of Keys - 56 bit – 2^56 combinations.

- Alice and Bob must know and use the same private key.

- 64 bits block size.(out of 64bits , 8 bits is used for some reserved purposes)

- DES was not designed for software and hence runs relatively slow.

- It is vulnerable to brute force attacks and linear crypt-analysis

- We have to switch encryption keys after every 32 GB of data transfer to reduce the possibility of leaks.

- There can be same output from the S-Boxes on different inputs on permutation.

AES – Advantages

- High Efficiency and Secure
- Not complex in structure
- Low Memory
- Main strength - various key lengths - 128-bit, 192-bit or 256-bit key.
- Combination of encryption 2^128, 2^192, 2^256.
- 128bit - block size
- 256 billion GB data can be transferred with single encryption.
- Still unbreakable (but theoretical discussion about breaking AES)
- Designed for both software and hardware.
- Time required to check all the possible keys at 50 billion keys per second – for 128 bit keys: 5x1021 years (which makes it difficult for the hackers to decrypt the data.

AES – Disadvantages

- Implementing software is complex taking both performance and security into considerations.

- Need more processing. it require more rounds of communication as compare to DES.

- Uses too simple algebraic structure.

Conclusion

- When it comes to security, the winner is undoubtedly AES as it is considered unbreakable in practical use.
- It seems like DES is insecure and no longer of any use, In 1997 attack required a great deal of cooperation and the 1998 machine is too expensive to implement, and so the DES and 3DES algorithms are still beyond the capability of most attacks.
- However, the power of computers is increasing and stronger algorithms are required to face hacker attacks.
- AES is safe for now.

References:

- Understanding Cryptography –Christof Paar & Jan Pelzl;
- Research Paper-Linear and Differential Cryptanalysis-Howard M. Heys
- Matsui, Mitsuru (1994). "Linear Cryptanalysis Method for DES Cipher". *Advances in Cryptology—EUROCRYPT '93*. Springer.
- http://www.eetimes.com/document.asp?doc_id=1279619