DES ŞİFRELEME ALGORİTMASI

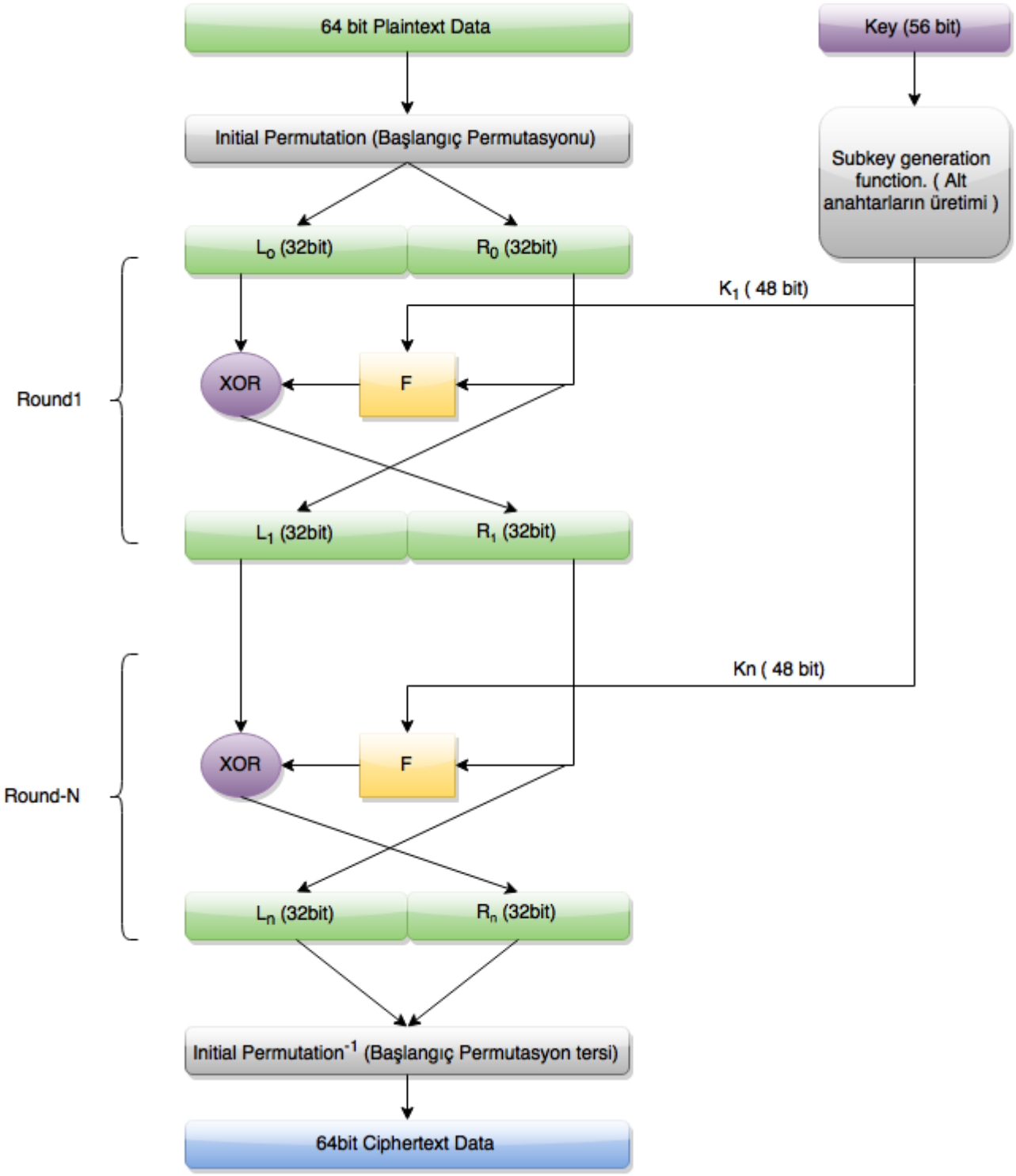
Data Encryption Standard (DES)

DES, ilk simetrik şifreleme algoritmasıdır. IBM tarafından 1970 lerde geliştirilmiş ve daha sonra NSA tarafından 1977'de bir standart haline gelmiştir.

DES'in, önceki substitution (yerine koyma) algoritmalarından en büyük farkı, işlemlerini bitler (0 ve 1) üzerinden yapıyor olmasıdır.

Veriyi bloklara ayırarak şifreleme yapar. Her blok 64 bitten oluşur. Ve 56 bitlik anahtar kullanır. Aynı anahtarla hem şifreleme hem de deşifreleme yapar.

Aslında, orjinal anahtar 64 bittir. Fakat parite kontrolu için 8 biti ayırır. Dolayısıyla kullandığı 56 bittir. Bu projede, her karakterin ASCII değerini alıp onu 8 bitlik binary değerine dönüştürdük.



Not:

Burada kullandığımız Sbox ve diğer permutasyon tabloları public olarak yayınlandığından herkesin erişimine açıktır.

1. **Initial Permutation** (İlk karıştırma)

   Amaç, metni karıştırmaktır. Aşağıdaki tabloyu incelediğinizde, bloklara ayrılan metnin ilk biti 58.sıraya, ikinci biti 50. sıraya geçer. Bu karıştırma işleminden sonra metin 2'ye bölünür. 32 bit sol tarafa (LPT), 32 bit sağ tarafa (RPT) ayrılır.

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

Initial Permutation Table

1. **16 Rounds**

- **Key Transformation** (anahtar oluşturma)

  Circular Left shift Table (For Encryption)

56 bitlik anahtar 2'ye bölünür. C key(28 bit) ve D key (28 bit) İkiye ayrılan bu anahtara shifting (kaydırma) uygulanır. Shifting işlemi aşağıdaki tabloya göre yapılır. Shifting işlemi tamamlandıktan sonra C ve D keyleri birleştirilir ve böylece 56 bitlik anahtar üretilir.

1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Circular Left Shift Table

Deşifreleme için ise burdaki tablonun tersi kullanılır. Circular Right shift Table (For Decryption)

2. Compression Permutation (sıkıştırma)

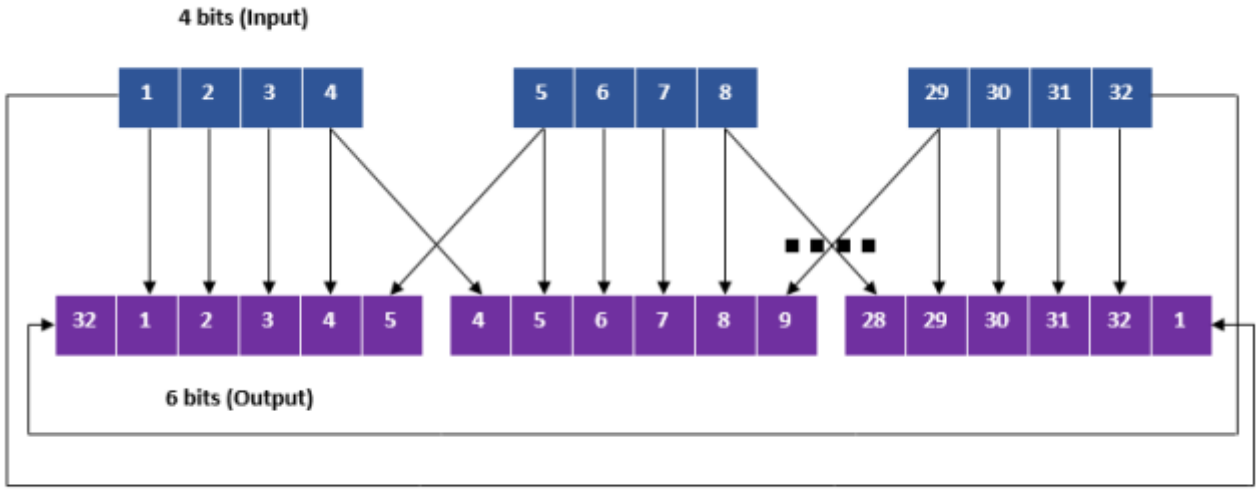
Burası da, bir üstte üretilen 56 bitlik shifted key içinden 48 bitlik anahtar üretimi içindir. Aslında burada hem sıkıştırma hem de karıştırma (permutasyon) yapılmış olur. Örneğin, key'in ilk biti 14. sırayla değiştirilir. İkinci biti 17.sırayla, gibi..

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

Compression Permutation Table

3. Expansion Permutation (genişletme)

Başlangıç aşamasında yaptığımız ilk karıştırma (initial permutation) sonrasında elimizde 32 bitten oluşan 2 bölüm oluşuyor. LPT (left plain text) ve RPT (right plain text) Genişleme tablosuna (expansion table) girdiğinde 32 bitlik RPT 48 bite çıkıyor. Tabi bu esnada aynı zamanda karıştırılmış (permuted) da oluyor. 32 bitlik RPT, 4 bitlik 8 bloğa bölünür. Ve her 4 bitlik blok, 6 bite dönüştürülür. Böylece 8*6 = 48 bite ulaşır. Bu işlemi, 4 bitlik bloğun başına 1 bit ve sonuna 1 bit ekleyerek yapar. Örnekle anlatacak olursak. İlk iki bloğu düşünün 1234 ve 5678 olsun. Son 4 lü blok 32 ile biter. Bu durumda expansion yapıldığı anda, yandaki bloku ilk biti, ilk bloğun son bitine eklenir, ilk bloğun son biti de yanındaki bloğun ilk bitine eklenir. Son durumda 1234 şeklindeki ilk blok 32 1 2 3 4 5 olacaktır.



RPT Expansion Permutation Process

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

Expansion Permutation Table

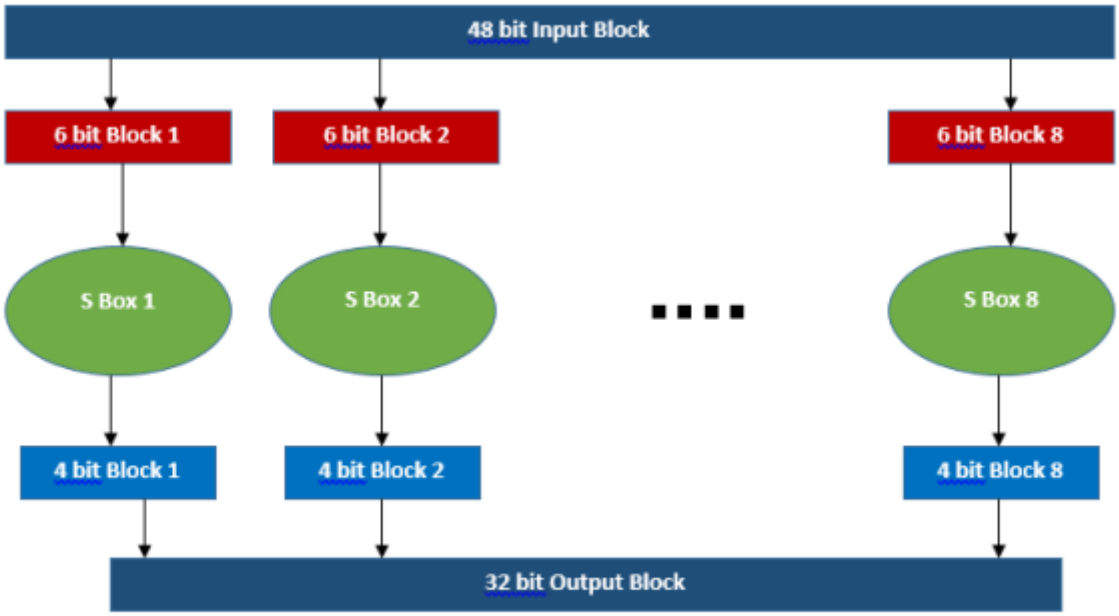
4. XOR

Burada, 48 bitlik genişletilmiş düz text (expanded RPT of 48 bit) ile 48 bitlik anahtar (compressed key of 48 bit) arasında bitwise XOR işlemi yapılıyor. Sonuç olarak yine 48 bitlik bir metin ortaya çıkacaktır.

5. **S Box Substitution**

Bir önceki işlemle XOR'lanmış metin (XORed RPT) (plain text ile keyin birleştirilmesi) S box'a verilir. Burada metin (48 bit), her biri 6 bitten oluşan 8 bloğa bölünür. Her blok için ayrı bir Sbox tablosu

bulunur. Bu sebeple de aşağıda 8 adet Sbox tablosu göreceksiniz. Sbox'lar 16 sütun, 4 satırdan oluşur. 0 ile 15 arasında değer alır. Ve her SBox 4 bitlik çıkış verir. Bütün SBox'ların görevi bittiğinde sonuç olarak (4*8) 32 bit dönecektir. (Sbox RPT) Bu 6 bitlik verinin 1. ve 6. verisi satır, 2.3.4.5 bitleri ise Sbox üzerindeki sütünları gösterir. Bunların kesiştiği nokta da dönüş değerini oluşturacaktır. Böylece her sbox'ın 4 bitlik çıkışı olacaktır.



S Box Substitution

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S Box 1 Table

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S Box 2 Table

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S Box 3 Table

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S Box 4 Table

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S Box 5 Table

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S Box 6 Table

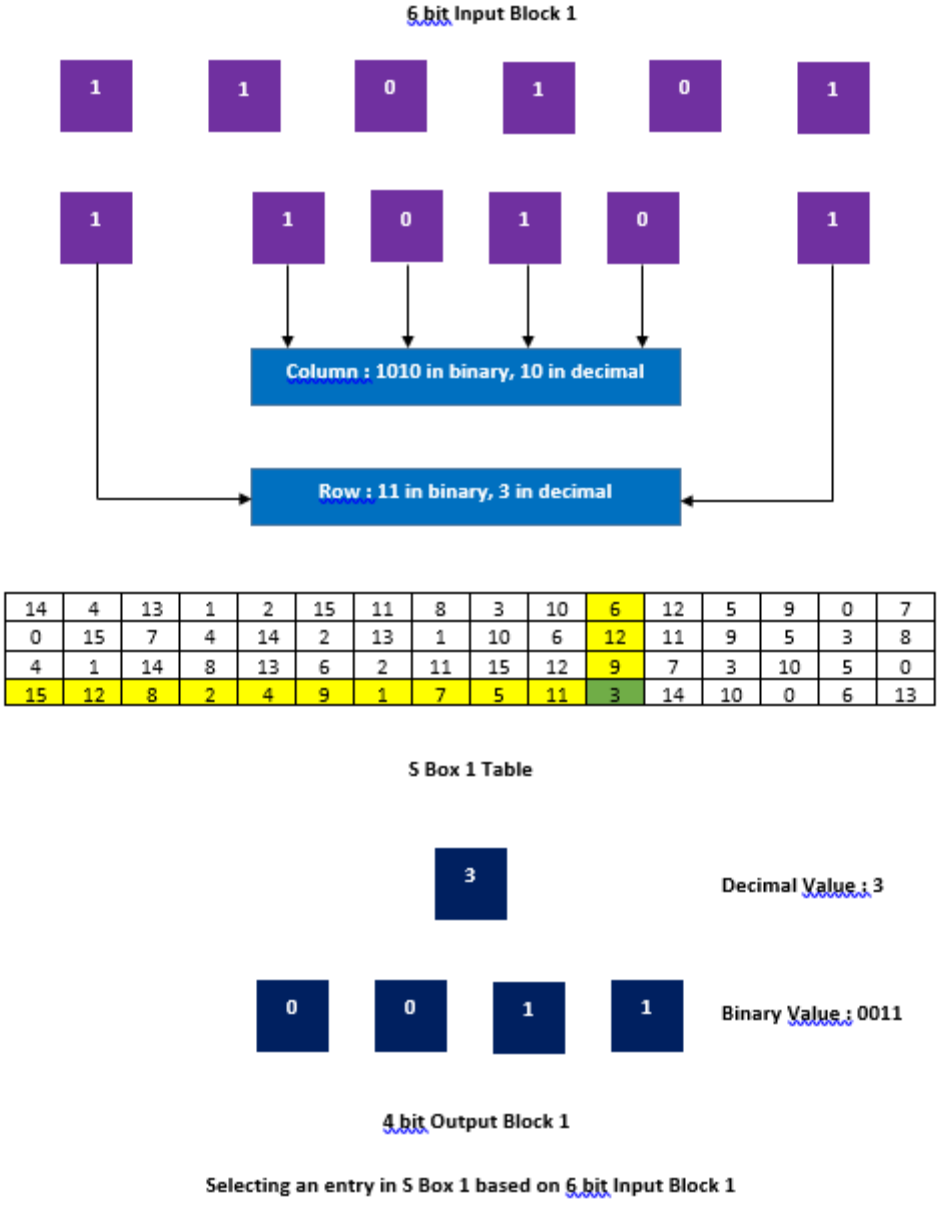
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S Box 7 Table

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S Box 8 Table

Örneğin, SBOX1 kullanarak, 110101 bloğunu ele alalım. İlk ve son biti satırı gösterecek. Bu durumda 11 olur. 11 lik binary değeri, decimal olarak 3 yapar. Yani sbox 3. indexine gidiyoruz.1010 ise sütün değerini gösterecektir. Binary 1010 ise 10 decimal yapacaktır. Bu durumda sütünlardan 10. index değerini seçiyoruz. Kesiştiği nokta bize 3' ü verecektir.



6. **P Box Permutation**

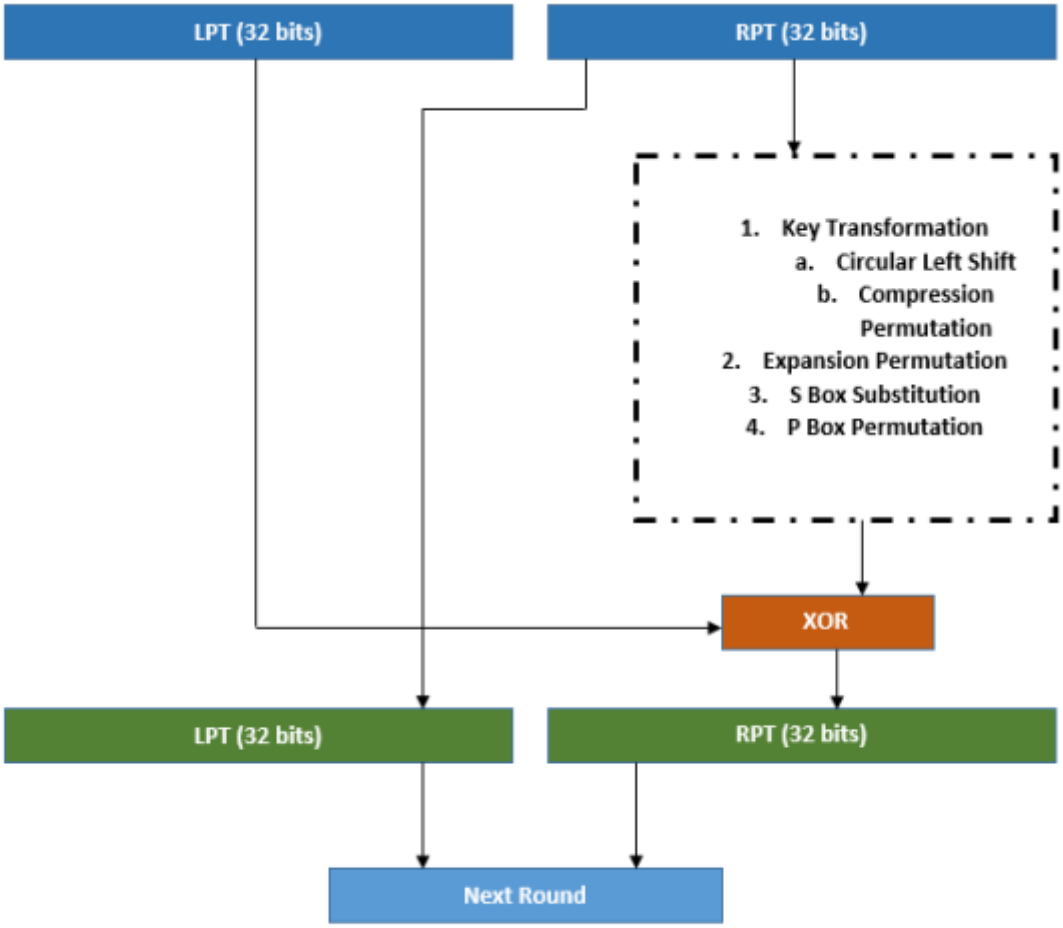
Burası da bir önceki aşamada, SBOX tan geçen verinin karıştırılma işlemi yapılır. Tüm SBOX'lardan geçtikten sonra elimize gelen 32 bit veri, bu tablo aracılığı ile karıştırılır (P Box table)

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

P Box Table

7. **XOR and Swap**

Bu aşamaya kadar LPT 'ye hiç dokunmadık. (Initial permutation sonrası LPT ve RPT olarak ikiye bölmüştük) Buraya dikkat, orjinal RPT yi(XOR'lanmamış) sola alıyoruz dolayısıyla yeni LPT oluyor. Sağ taraf ile (P Box RPT of 32 bit) ilk bölümde ayırdığımız sol taraf (LPT of 32 bit) XOR'lanıyor. Bu da bizim yeni RPT'miz oluyor. Ve bu işlem 16 kez tekrarlanıyor. 16 roundun sonunda, 32 bitlit LPT ve 32 bitlik RPT bize 64 bitlik bir çıktı veriyor.



XOR and Swap

Ve son olarak elde ettiğimiz bu veriyi final permutasyon tablosundan geçirip karıştırıyoruz.

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

Final Permutation Table