CryptoLab

컴퓨터공학부 **박재현**

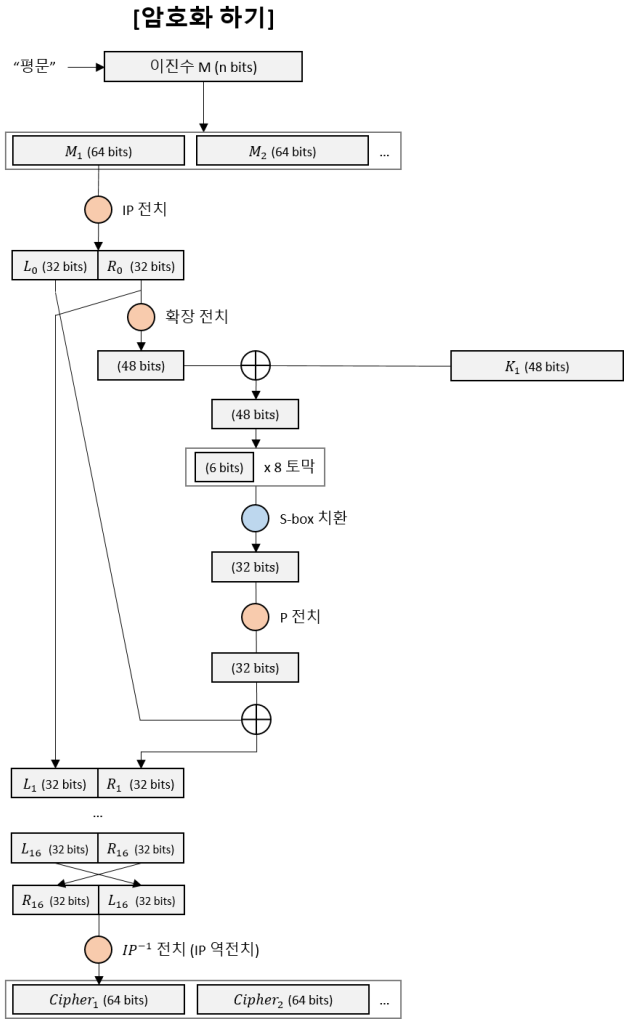
목차

DES

- Data Encryption Standard의 약자로 블록암호의 일종.

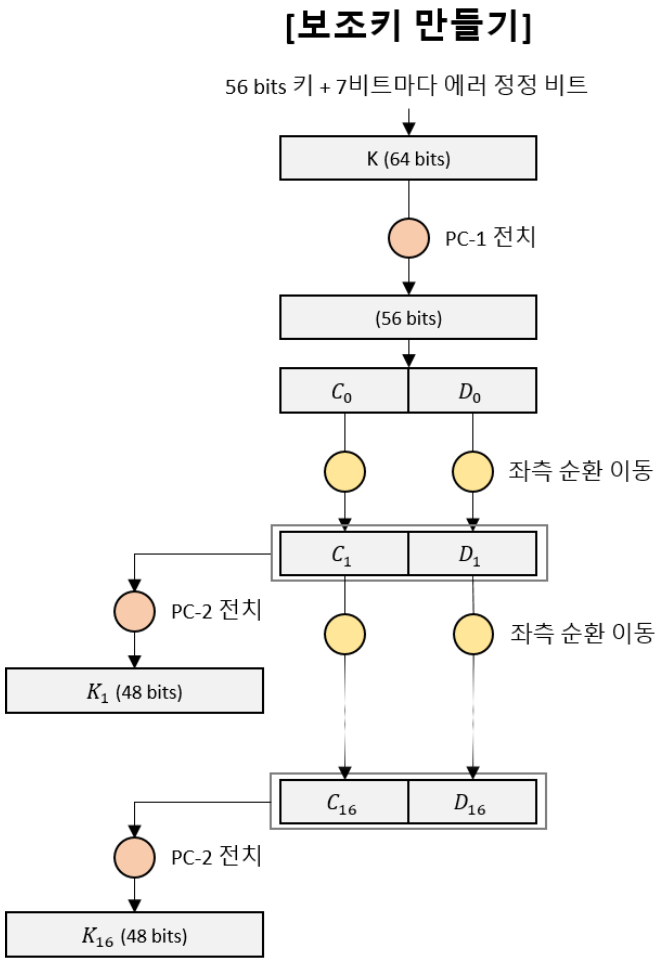
- 미국 NBS(현 NIST)에서 국가 표준으로 정한 암호.

DES


[암호화 하기]
"평문"
이진수 M (n bits)
M_1 (64 bits)
M_2 (64 bits)
...
IP 전치
L_0 (32 bits)
R_0 (32 bits)
확장 전치
(48 bits)
K_1 (48 bits)
(48 bits)
(6 bits)
x 8 토막
S-box 치환
(32 bits)
P 전치
(32 bits)
L_1 (32 bits)
R_1 (32 bits)
...
L_{16} (32 bits)
R_{16} (32 bits)
R_{16} (32 bits)
L_{16} (32 bits)
IP^{-1} 전치 (IP 역전치)
$Cipher_1$ (64 bits)
$Cipher_2$ (64 bits)
...

DES 보조키 - SubKey

- 56 bit 의 비밀키 사용
- 7 bit 마다 에러 정정 비트 삽입
- 보조키를 생성시 총 64 bit의 키 사용
  56 + 8 = 64



보조키 만드는 순서

1. Key값(56bit)을 받고 7비트마다 에러 정정 비트를 삽입해 64 bit 생성

2. 64 bit의 키에서 에러 정정비트를 제외한 후 비트의 배열을
   PC1**(Permutation Choice)**을 적용해서 재배열
   (Left 28 bit, Right 28 bit)

3. Left, Right를 각각 총 16번의 좌측 순환 이동시킨뒤 둘을 합치고
   PC2를 적용해 재배열시켜 16개의 보조키 생성
   (48 bit의 보조(서브)키 생성)

보조키 만드는 순서

Ex) Key 값을 Secrets일때

- 16진수로 표현하면

  0x53 0x65 0x63 0x72 0x65 0x74 0x73

- 2진수로 표현하면

  01010011 01100101 01100011 01110010 01100101 01110100 01110011 (56 bit)

- 여기에 7bit마다 에러 정정 비트를 삽입

  0101001? 1011001? 0101100? 0110111? 0010011? 0010101? 1101000? 1110011? (64 bit)

보조키 만드는 순서

앞서 8bit의 에러 정정 비트를 삽입해

64bit를 만든 뒤 PC-1을 적용하여

전치를 시킵니다.

PC-1

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

보조키 만드는 순서

Left

1100 0010 1100 1101 1011 1010 0100

Right

1011 1011 1001 1000 0010 1100 0111

이 둘을 이용해 좌측 표와 같이 순환이동을 시킨뒤

둘을 합쳐 PC-2를 적용합니다.

Rotations

Round number	Number of left rotations
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

보조키 만드는 순서

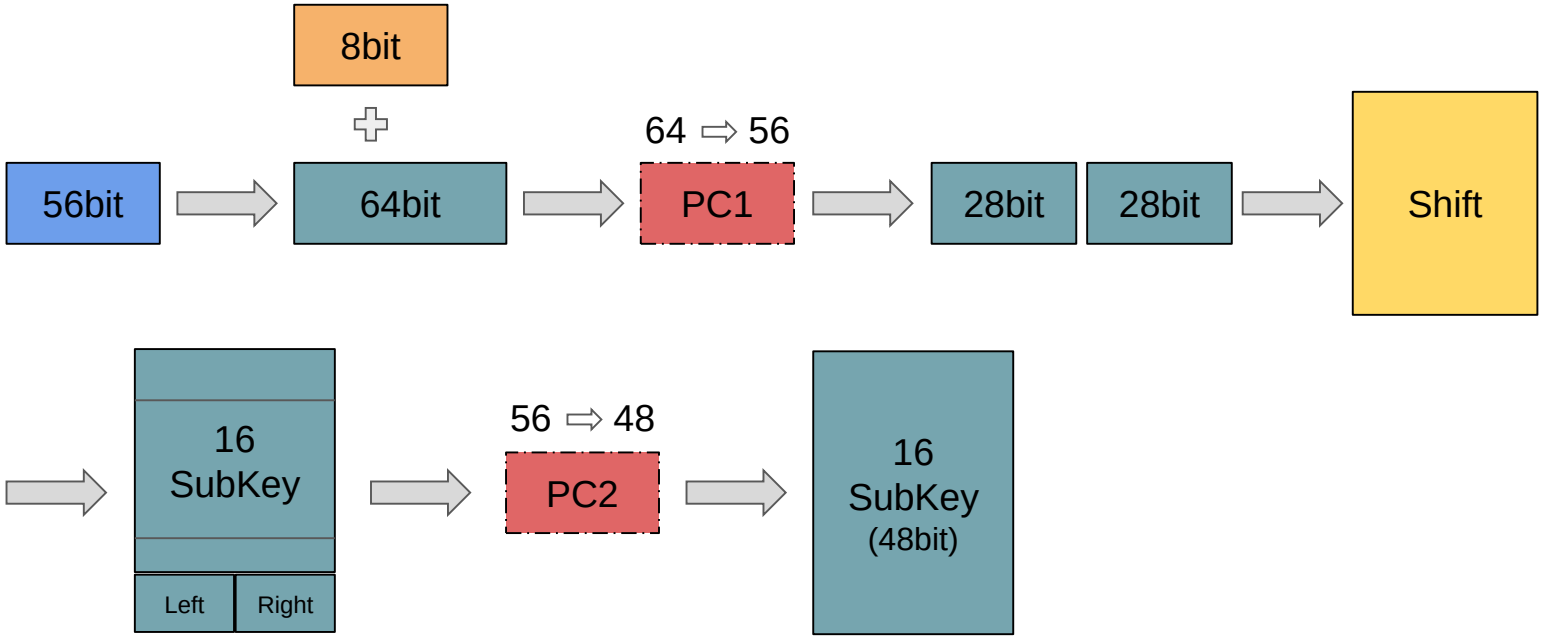
- PC-2과정을 거치면 8bit가 줄어 48bit의 보조키 하나가 생기게 됩니다. 이 과정을 16번 반복하여 16개의 보조키를 만들게 됩니다.
- Key 1 ~ Key 16 생성

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

보조키 만드는 순서

정리


8bit
56bit
64bit
64 ⇨ 56
PC1
28bit
28bit
Shift
16
SubKey
Left
Right
56 ⇨ 48
PC2
16
SubKey
(48bit)

암호화하기

- 먼저, 암호화 하려는 평문을 이진수로 바꾸고,
  해당 이진수를 **64 bits 의 블록** 단위로 나누는 작업 진행

- 암호화 알고리즘은 쪼개진 64 bits 블록에 대해 각각 적용

- 64 bit 의 블록은 **IP(Initial Permutation)** 라는 초기 순열을 이용한
  전치를 거쳐 새로운 64 bits 를 생성

암호화하기

암호화하고 싶은 평문을 2진수로 64bit씩 끊어 줍니

여기에 IP를 적용하여 전치시켜 32bit짜리 L0와 R0

를 만들어 줍니다.

IP

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

암호화하기

- 아래의 규칙을 통해 L16과 R16의 값을 구합니다.

  Ln = Rn-1
  Rn = Ln-1 XOR F(Rn-1,Kn)
  *n은 순서
  *Kn은 보조 키값 - K1, K2...

- F(Rn-1,Kn) 함수

  Rn-1(32bit)의 값과 Kn(48bit)의 값을 입력값으로 받음

  Rn-1을 E 테이블을 통해 48bit로 확장(중복 연산)

  Rn-1(48bit)와 Kn(48bit)를 XOR 연산을 취해
  48bit의 결과가 도출

E

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

암호화하기

F(Rn-1,Kn) 함수

XOR을 통해 나온 48bit를 다시 32bit로 변환 - **S-box(Substitution boxes)** 사용

48bit를 6bit짜리 8개로 나눔
ex) 011101 110011 ...

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

6bit중 첫번째와 여섯번째 비트가 합쳐져 S-box의 행
나머지 가운데 비트가 열을 결정
ex) 011101 - 01(1) 행, 1110(14) 열 - S-box에서 03
즉, 011101은 S-box를 통해 03(0011)로 변환

P Box

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

마지막으로 이를 P-Box를 통해 전치시킵니다 - F(Rn-1, Kn)의 결과 도출(32bit)

F(Rn-1, Kn)의 결과 값을 Ln-1과 XOR하면 Rn의 값을 구할 수 있습니다.

암호화하기

S-box

Ex) 6bit 가 100100 이라고 할때

행 : 첫번째와 여섯번째 1,0

열 : 나머지 0,0,1,0

정수 값 : 14

이진수로 변환 : 1110

S-boxes

S_1	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	
0yyyy1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	
1yyyy0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	
1yyyy1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	

S_2	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	
0yyyy1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	
1yyyy0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	
1yyyy1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	

S_3	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	
0yyyy1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	
1yyyy0	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	
1yyyy1	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	

S_4	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	
0yyyy1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	
1yyyy0	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	
1yyyy1	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	

S_5	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	
0yyyy1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	
1yyyy0	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	
1yyyy1	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	

S_6	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	
0yyyy1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	
1yyyy0	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	
1yyyy1	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	

S_7	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	
0yyyy1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	
1yyyy0	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	
1yyyy1	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	

S_8	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x111
0yyyy0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	
0yyyy1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	
1yyyy0	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	
1yyyy1	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	

암호화하기

S-box를 통해 32bit를 알아내면 P로 전치를 시켜줍니디

위 과정을 반복해 L16과 R16을 알아내면 둘의 위치를 바꿔줍니다. (L16+R16 -> R16+L16)

P

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

암호화하기

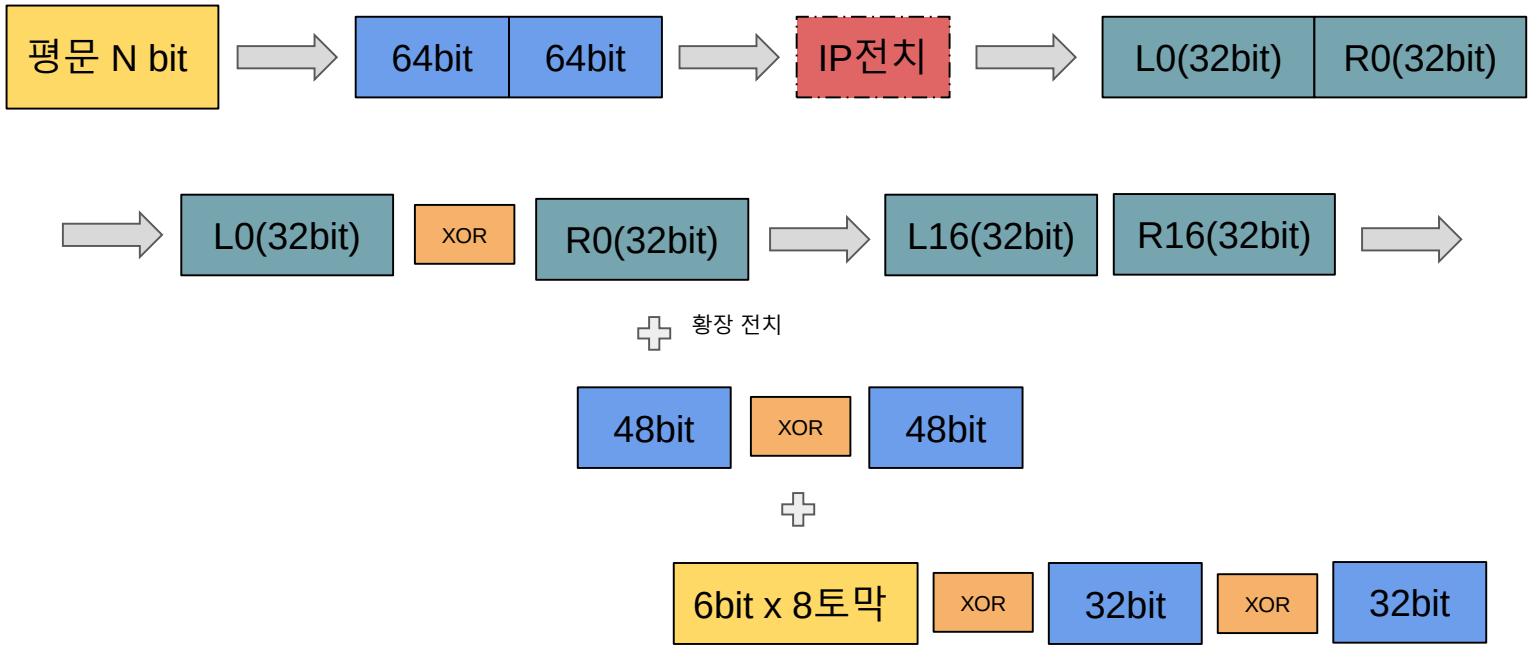
마지막으로 처음에 사용한 IP의 역 순열인 IP-1을 64bit(R16+L16)에 적용하면 DES 암호화가 끝이 납니다.

IP^{-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

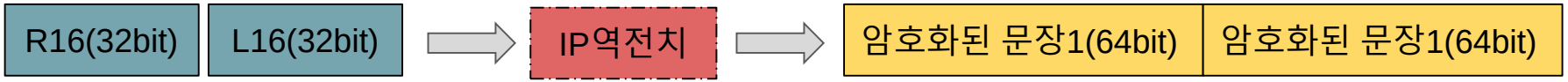
암호화하기

정리


평문 N bit
64bit
64bit
IP전치
L0(32bit)
R0(32bit)
L0(32bit)
XOR
R0(32bit)
L16(32bit)
R16(32bit)
황장 전치
48bit
XOR
48bit
6bit x 8토막
XOR
32bit
XOR
32bit

암호화하기

정리



DES의 한계

- DES는 현재 취약한 것으로 알려져 있음
- 기술의 발전에 의해 56 bits 의 키 길이는 너무 짧은 것이 되었고, Brute-Force에 의해 해독이 가능
- 현재는 DES를 세 번 반복해서 사용하는 Triple-DES나 AES(Advanced Encryption Standard)를 사용하고 있음
- 그러나 AES를 많이 씀!

감사합니다