3장 대칭 암호

공통키 암호

박 종 혁

Tel: 970-6702
Email: jhpark1@snut.ac.kr

3.0 주요 내용

- 비트열의 조작과 XOR 연산
- 블록 암호의 개념
- 스트림 암호의 개념
- 일회용 패드(one time pad)
- 대칭암호
  - DES
  - 트리플 DES
  - AES

3.1 문자 암호에서 비트열 암호로

3.1.1 부호화

- 컴퓨터의 조작 대상은 문자가 아니라 0과 1의 연속인 비트열이다.
- 문자도 화상도 비디오도 프로그램도 컴퓨터 안에서는 모두 비트열로 표현되고 있다.
- 암호화를 행하는 프로그램은 비트열로 되어 있는 평문을 암호화하여 비트열로 되어 있는 암호문을 만들어내는 것

부호화와 암호화

부호화(encoding) : 현재 사용하고 있는 문자들을 비트열에 대응시키는 것

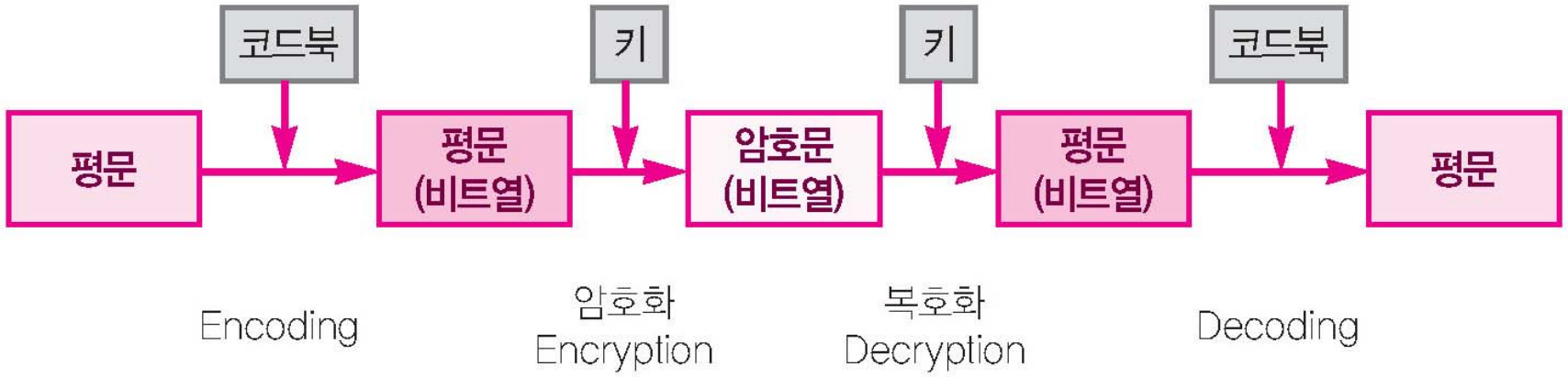


그림 3-1 부호화와 암호화

ASCII(아스키) 코드 대응규칙

- m → 01101101
- i → 01101001
- d → 01100100
- n → 01101110
- i → 01101001
- g → 01100111
- h → 01101000
- t → 01110100

3.1.2 XOR

- 한 비트의 XOR 연산은 다음과 같은 계산이다.

0 XOR 0 = 0 (0과 0의 XOR은 0이 된다)
0 XOR 1 = 1 (0과 1의 XOR은 1이 된다)
1 XOR 0 = 1 (1과 0의 XOR은 1이 된다)
1 XOR 1 = 0 (1과 1의 XOR은 0이 된다)

비트열 XOR

0 1 0 0 1 1 0 0 ... A

⊕ 1 0 1 0 1 0 1 0 ... B

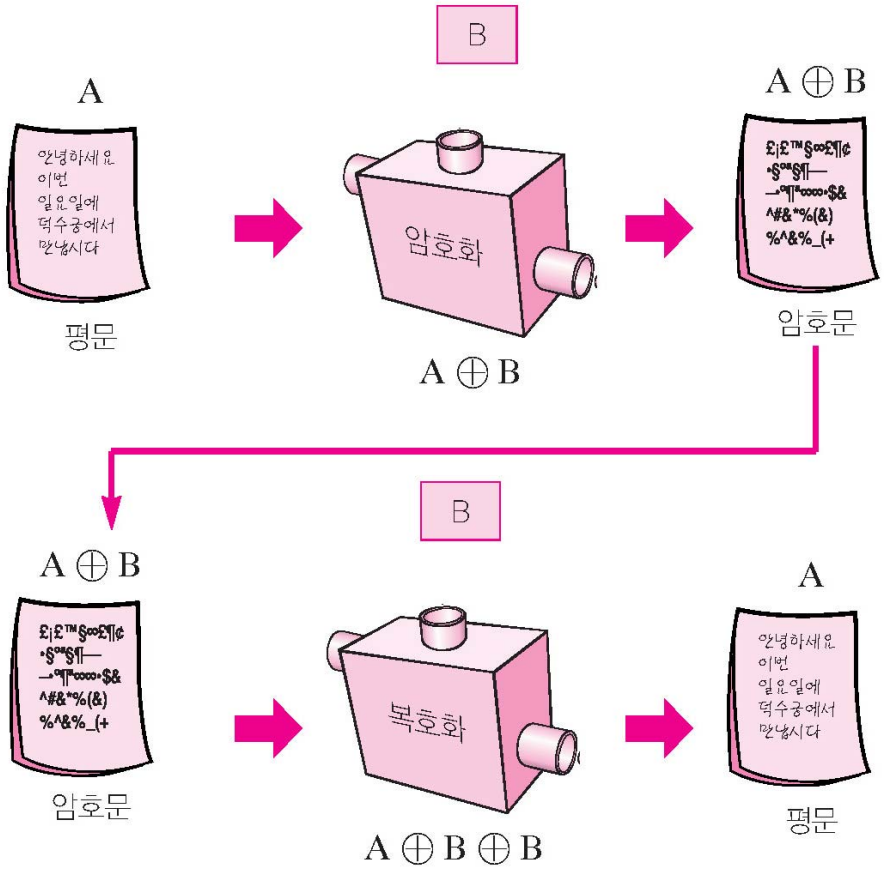
---

1 1 1 0 0 1 1 0 ... A ⊕ B

$A \oplus B \oplus B = A$

$$
\begin{array}{ll}
 & 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0 \quad A \oplus B \\
\oplus & 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0 \quad B \\
\hline
 & 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0 \quad A \oplus B \oplus B = A
\end{array}
$$

XOR을 이용한 암호화와 복호화



- 평문 A를 키 B로 암호화해서 암호문 A⊕B를 얻는다
- 암호문 A⊕B를 키 B로 복호화해서 평문 A를 얻는다

그림 3-2 XOR을 이용한 암호화와 복호화

한 비트열에 XOR을 2회 되풀이

- XOR은 그림을 마스크한다

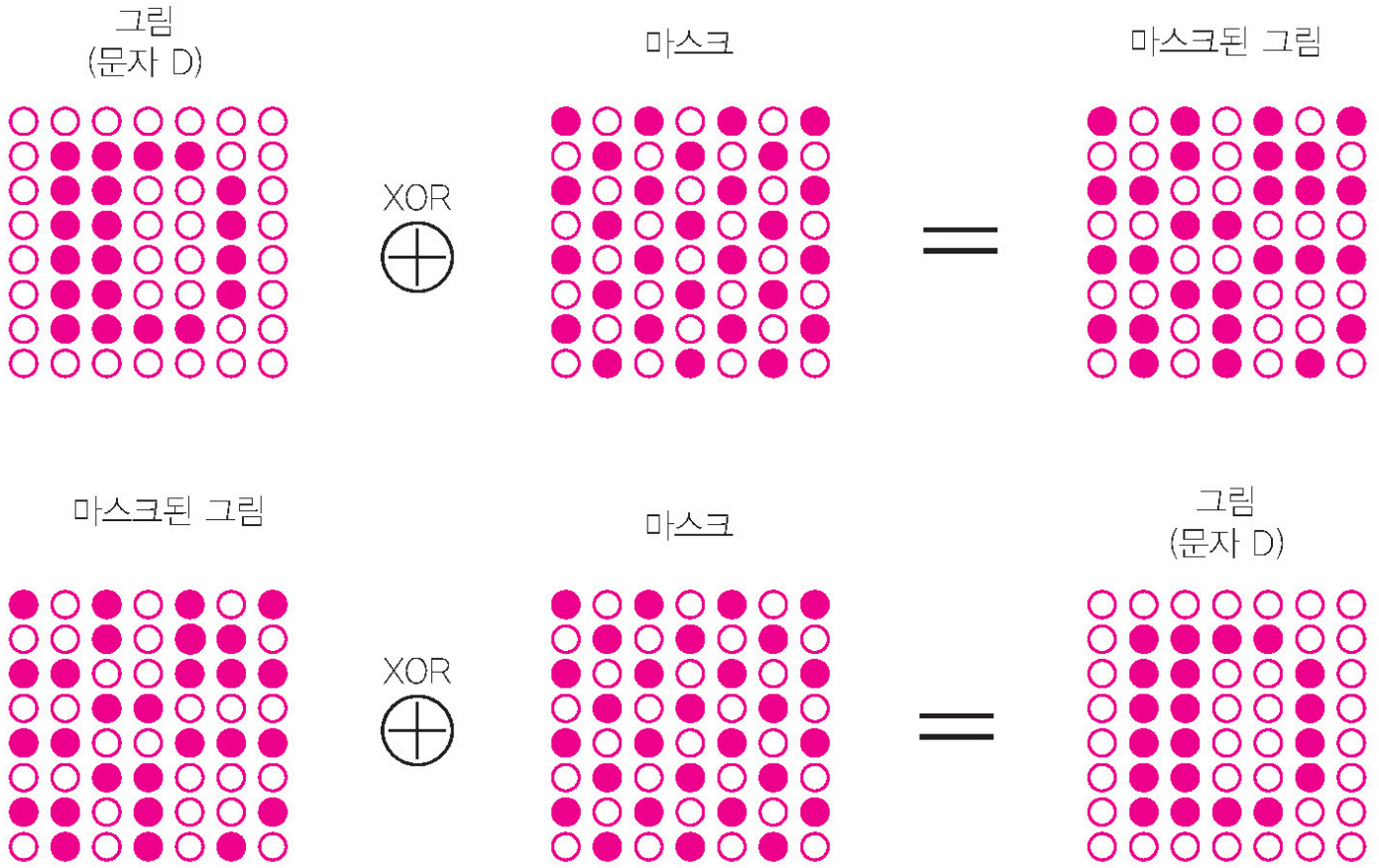


그림 3-3 XOR은 그림을 마스크한다

「예측 가능 여부」

- 암호기술에서 「예측 가능 여부」는 매우 중요한 요소 중의 하나이다.
- 예측 불가능한 비트열을 만들 수 있다면 그것을 암호기술에 매우 유용하게 사용할 수 있다.
- 이러한 예측 불가능한 비트열을 **난수(random number)**라고 부른다.

3.2 스트림 암호화 블록암호 도청자

- 블록 암호는 입력되는 하나의 블록을 한 번에 한 블록씩 처리하여 각 입력블록에 대응되는 출력 블록을 만든다.
- 스트림 암호에서는 입력되는 요소를 연속적으로 처리하여 지속적으로 한 번에 한 요소씩 배출한다.

3.2.1 블록암호

- 고정된 크기의 블록 평문을 입력으로 하여 동일한 크기의 블록 암호문을 생성해내는 암호시스템
- 암호 응용에 있어서 가장 보편적으로 사용되는 대칭 암호 알고리즘이 대표적인 블록 암호이다.
- 가장 중요한 블록 암호의 예
  - DES,
  - 트리플 DES
  - AES

블록 암호 흐름도

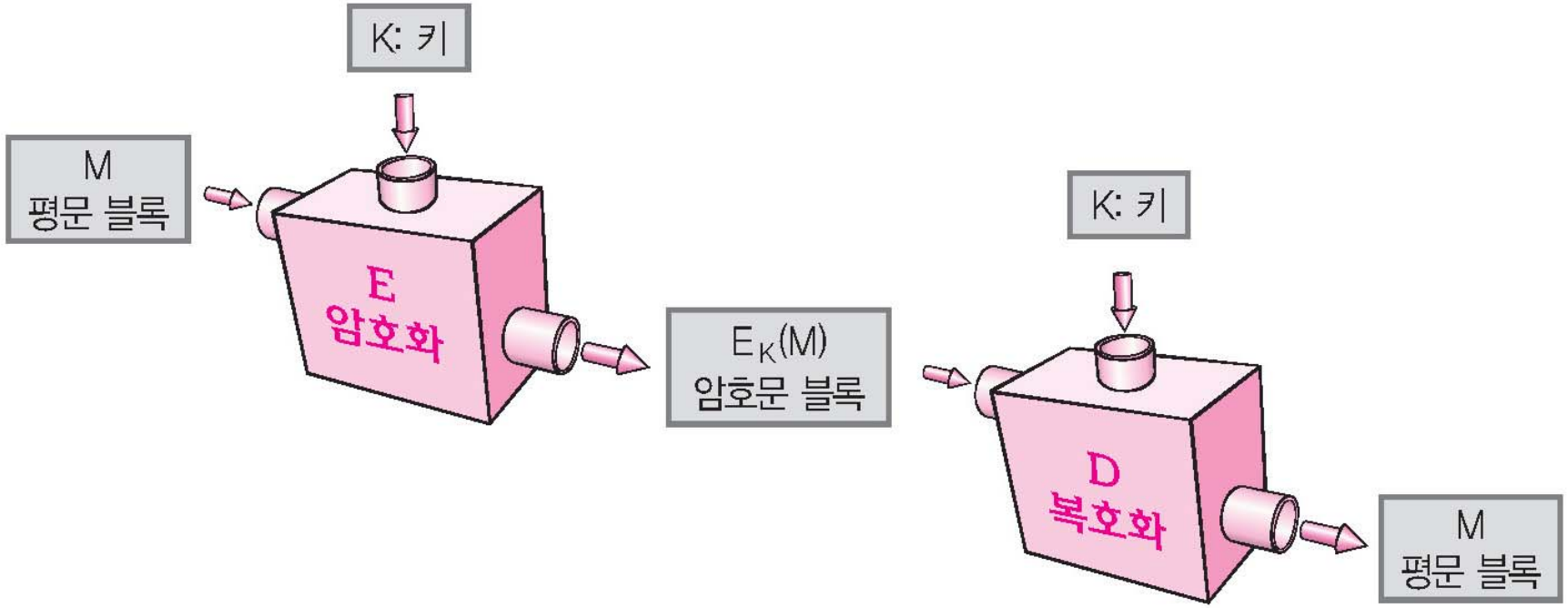


그림 3-4 블록 암호 흐름도

3.2.2 스트림암호

- 한 번에 한 바이트씩 암호화
- 초기값을 필요로 한다
- 암호화의 예

```
   10001010     평문
⊕  01001101     키 스트림
   ---------
   11000111     암호문
```

동일한 의사랜덤 키 스트림

- 복호화의 예

  11000111　　암호문

⊕ 01001101　　키 스트림

  10001010　　평문

스트림 암호 흐름도

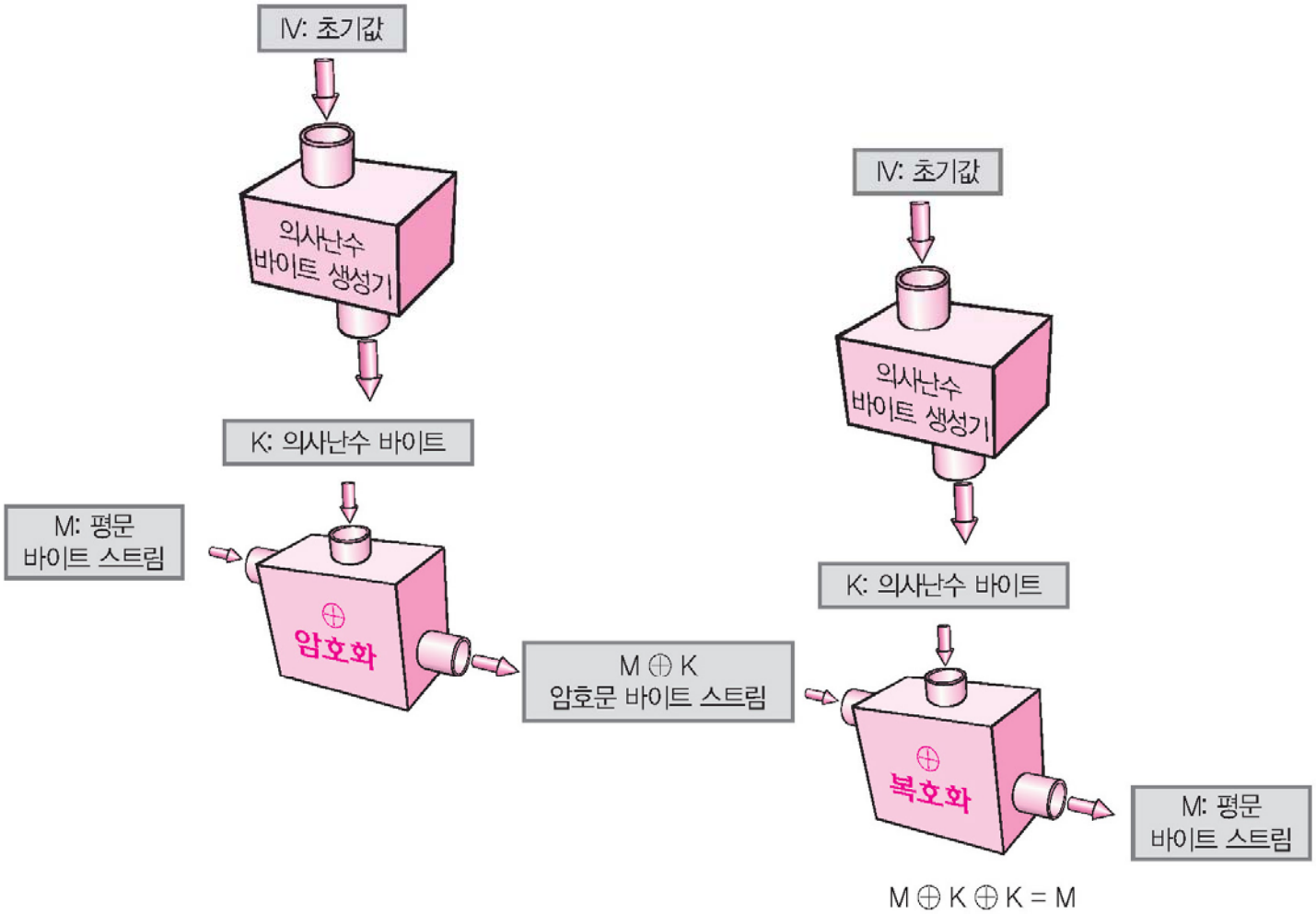


그림 3-5 스트림 암호 흐름도

3.3 일회용 패드

- 절대로 해독할 수 없는 암호인 일회용 패드라는 암호를 XOR의 연습을 겸해서 소개하도록 한다

3.3.1 일회용 패드란

- 키 공간에 속한 모든 키들을 전부 시도하는 전사 공격을 수행하면 어떤 암호문이라도 반드시 언젠가는 해독할 수 있다.
- **일회용 패드**(one-time pad)라는 암호는 예외이다.
- 일회용 패드는 전사공격으로 키 공간을 전부 탐색하더라도 절대로 해독할 수 없는 암호

3.3.2 일회용 패드의 암호화

- 일회용 패드는 「평문과 랜덤한 비트열과의 XOR만을」 취하는 단순한 암호

문자열 “midnight” 을 ASCII로 부호화

문자	m	i	d	n	i	g	h	t
ASCII 코드	01101101	01101001	01100100	01101110	01101001	01100111	01101000	01110100
단어	midnight							

평문의 길이와 같은 64비트의 랜덤한 비트열을 준비

키	01101011	11111010	01001000	11011000	01100101	11010101	10101111	00011100

비트열 생성

평문과 키 비트열의 XOR을 계산해서 새로운 비트열을 만든다

	01101101	01101001	01100100	01101110	01101001	01100111	01101000	01110100	midnight
⊕	01101011	11111010	01001000	11011000	01100101	11010101	10101111	00011100	키
	00000110	10010011	00101100	10110110	00001100	10110010	11000111	01101000	암호문

3.3.3 일회용 패드의 복호화

복호화는 암호화의 역계산이다.

	00000110	10010011	00101100	10110110	00001100	10110010	11000111	01101000	암호문
⊕	01101011	11111010	01001000	11011000	01100101	11010101	10101111	00011100	키
	01101101	01101001	01100100	01101110	01101001	01100111	01101000	01110100	midnight

3.3.4 일회용 패드의 해독

- 위와 같이 일회용 패드는 매우 단순하다.
- 이처럼 단순한 암호가 해독 불가능하다는 것은 놀랄 만한 일이다.
- 여기서 말하는 해독 불가능이라는 것은 단순히 현실적인 시간으로 해독이 곤란하다는 의미는 아니다.
- 아무리 임의 크기의 키 공간 전체를 순식간에 계산할 수 있는 무한대의 계산력을 갖는 컴퓨터가 발명되었다고 하더라도 일회용 패드는 해독할 수 없다.

일회용 패드 암호화

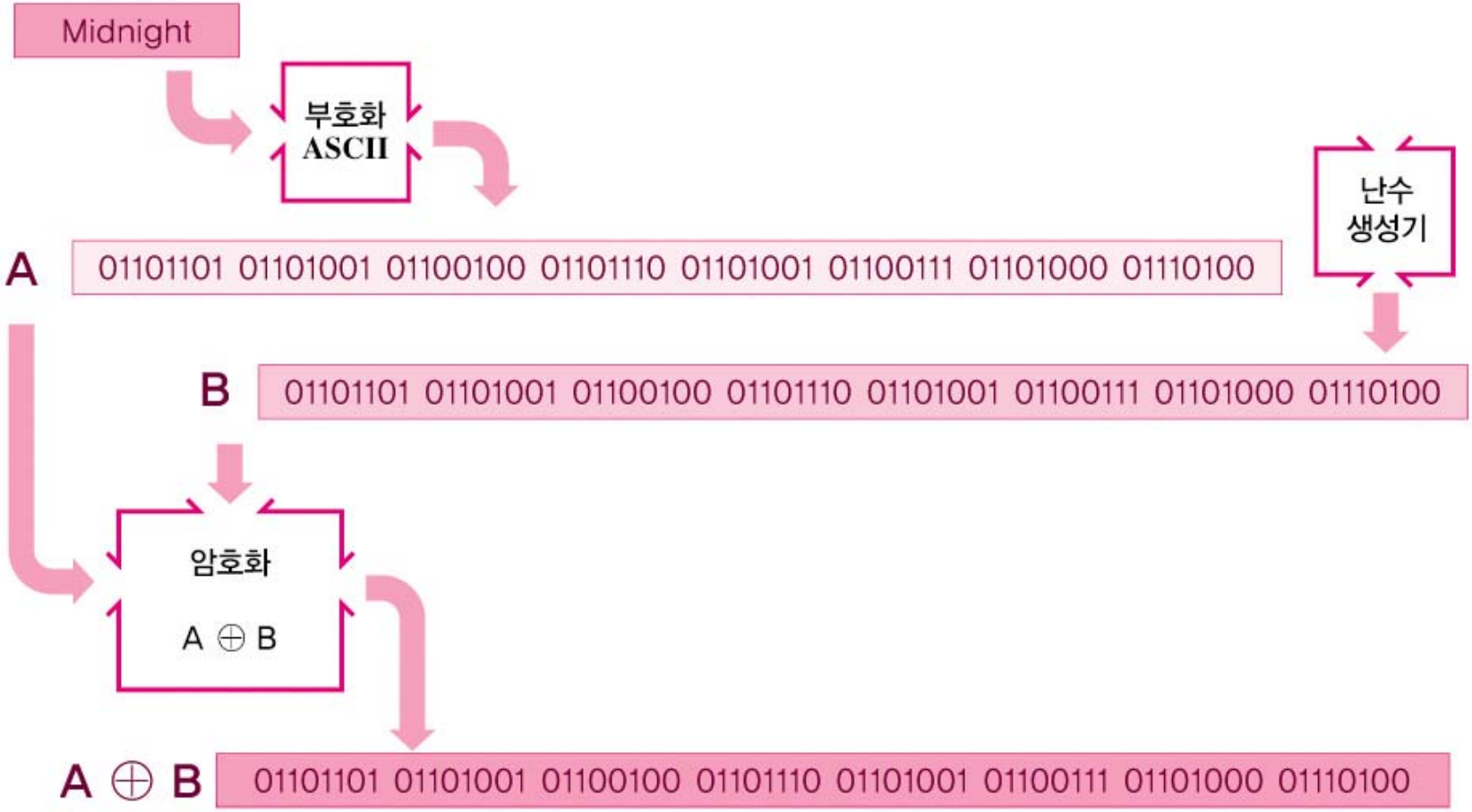


그림 3-6 일회용 패드 암호화

일회용 패드 복호화

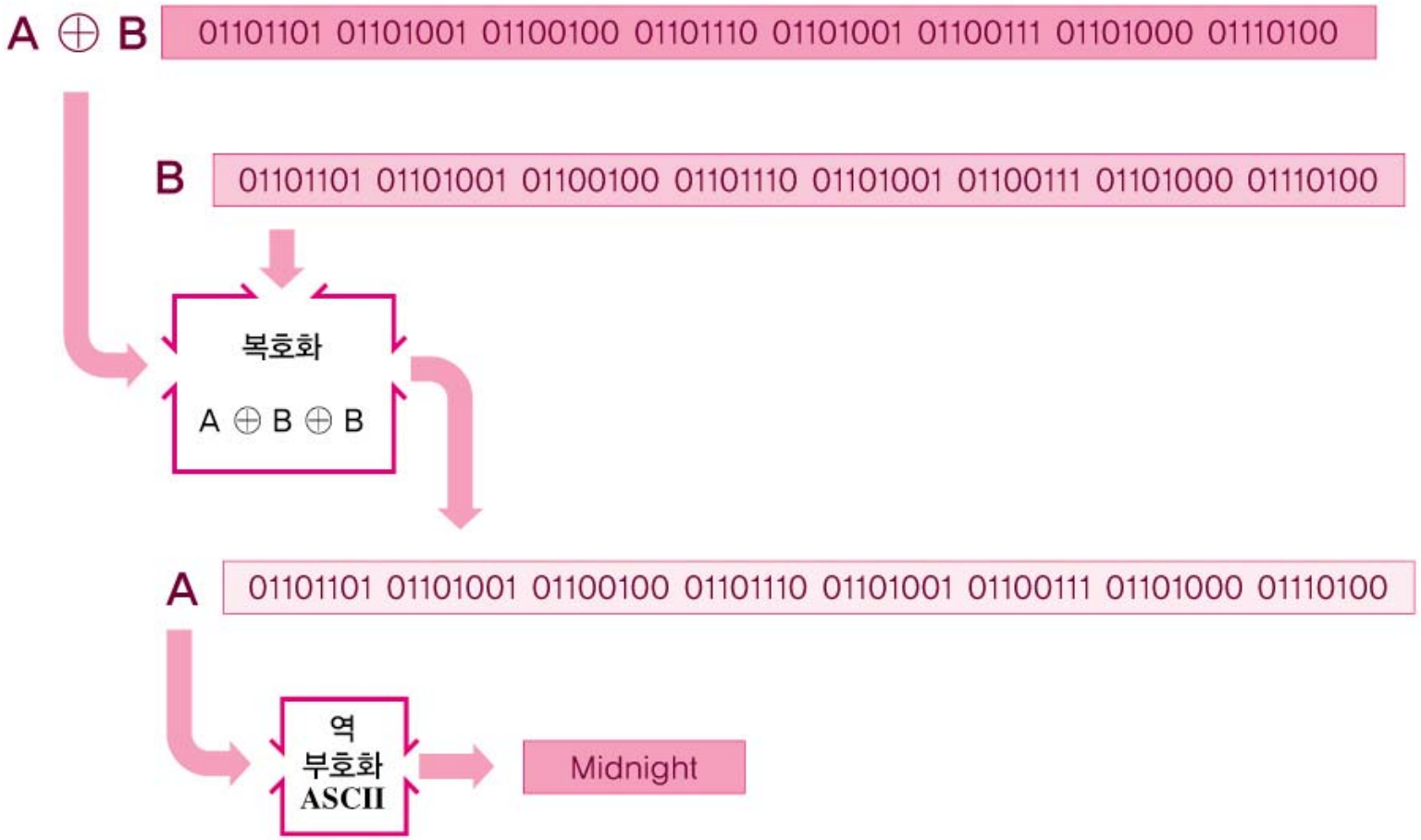


그림 3-7 일회용 패드 복호화

어째서 절대로 해독할 수 없는 것일까?

- 일회용 패드의 암호문에 대해 전사공격을 시도했다고 가정해 보자. 그러면 언젠가는 암호화에 사용한 것과 똑같은 키를 시도하게 될 것이고, 그 때 midnight라는 문자열이 복호화 될 것이다.
- 이것은 사실이다. 그러나 – 이것이 중요한 점이다 – 설령 midnight라는 문자열이 복호화 되었다 하더라도, **이것이 바른 평문인지 어떤지를 판정할 수가 없는** 것이다.

어째서 절대로 해독할 수 없는 것일까?

- 왜냐 하면 일회용 패드의 복호화를 시도하고 있는 도중에는 모든 64비트의 패턴이 등장하게 되기 때문이다.
- 그 중에는 aaaaaaaa, abcdefgh, zzzzzzzz 등과 같은 규칙적인 문자열도 있고, midnight, onenight, mistress 등과 같은 영어단어도 포함된다.
- 또한 %Ta_AjvX, HY(&JY!z, $@~*₩^^), Er#f6)(% 와 같이 읽을 수 없는 문자열도 등장하게 된다.
- 평문으로서 가능한 모든 패턴이 등장하기 때문에 어느 것이 바른 평문인지(즉 어느 키를 사용하면 바르게 복호화할 수 있는지)를 판단할 수가 없는 것이다.

어째서 절대로 해독할 수 없는 것일까?

- 일회용 패드에서는 키들을 적용하여 얻어진 것이 바른 평문인지 아닌지를 판정하는 것이 불가능하다.
- 그러므로 일회용 패드를 해독할 수 없는 것이다.

3.3.5 일회용 패드의 미사용

- 일회용 패드는 실제로는 거의 사용되지 않고 있다.
- 왜냐 하면 일회용 패드는 매우 사용하기 어려운 암호이기 때문이다.
- 이제, 그 이유를 살펴보자.

키의 배송

- 수신자가 복호화를 행하기 위해서는 송신자가 암호화했을 때 사용했던 키와 같은 키가 필요하다.
- 송신자는 수신자에게 이 키를 보내지 않으면 안 된다.
- 여기서 우리가 주의해야 할 것은 그 키의 길이가 통신문의 길이와 같다는 사실이다.
- 따라서 키를 안전하게 보낼 수 있는 방법이 있다면 평문 그 자체를 같은 방법으로 안전하게 보낼 수 있을 것이다.

키의 보존

- 일회용 패드에서는 평문과 같은 비트 길이의 키가 필요하다.
- 게다가 그 키는 평문의 비밀을 지키고 있는 것이므로 도청자에게 들키지 않도록 보존되어야 한다.
- 하지만 평문과 같은 비트 길이의 키를 안전하게 보존할 수 있다면 평문 그 자체를 안전하게 보존해 둘 수 있을 것이다.
- 즉, 암호는 처음부터 필요 없었다는 것이다.

키의 재이용

- 일회용 패드에서는 과거에 사용한 랜덤한 비트열을 절대로 재이용해서는 안 된다.
- 일회용 패드의「일회용」이라는 이름은 이런 이유 때문에 붙여진 것이다.
- 키로 사용한 비트열이 누설되어 버리면 과거에 행한 비밀 통신이 (도청자 이브가 과거의 통신내용을 전부 보존하고 있다면) 복호화되어 버리게 된다.

키의 동기화

- 송신자와 수신자 사이에 전송되는 키가 되는 비트열이 1비트라도 어긋나서는 안 된다.
- 1비트라도 어긋나면 그 이후의 복호화는 불가능해진다.
- 따라서 비트열의 전송시 동기화 문제는 중요하게 된다.

키의 생성

- 일회용 패드에서는 난수를 대량으로 생성할 필요가 있다.
- 비용에 무관하게 기밀성을 최우선으로 고려해야 할 사항인 경우에만 사용한다.
  - 강대국끼리의 핫라인
- 일회용 패드는 그다지 실용적인 암호는 아니다.

3.4 DES

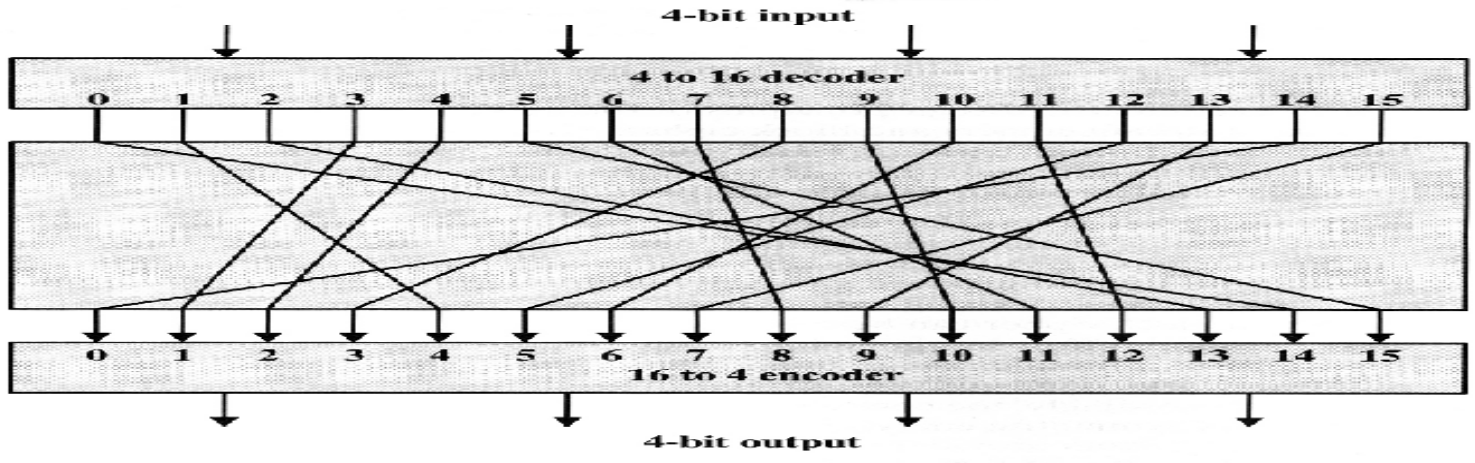
- DES는 64비트의 키를 적용하여 64비트의 평문을 64비트의 암호문으로 암호화 시키는 대칭형 블록 암호이다.
- DES 알고리즘에서는 사용하는 함수
  - 대체(substitution)
  - 치환(permutation)
- 대체와 치환은 1949년도에 Claude Shanon이 제시한 혼돈(confusion)과 확산(diffusion)이라는 두 가지 개념에 기반을 두고 있다

블록 암호 기법의 원리

- 스트림 암호
  - 한 번에 1비트 혹은 1바이트의 디지털 데이터 스트림을 암호화 하는 방식
- 블록 암호 기법
  - 평문 블록 전체를 가지고 같은 크기의 암호문 블록을 생성
  - 모드를 이용하여 스트림 암호 기법과 동일한 효과
  - Feistel 암호 방식 (혼돈과 확산: Confusion & Diffusion)
    - Claude Shannon 소개(SHAN49): "매우 이상적인 암호는 암호문에 대한 모든 통계적 정보가 사용된 키와 독립적이어야 한다."
    - 통계적 분석에 기초한 암호 해독 방지
    - **혼돈** : 키를 발견하기 어렵게 하기 위해 암호문에 대한 통계 값과 암호 키 값 사이에 관계를 가능한 복잡하게 하는 것
    - **확산** : 평문의 통계적 구조가 암호문의 광범위한 통계값에 분산
      - 키를 추론하기 어렵게 하기 위해 평문과 암호문 사이에 통계적인 관계를 가능한 한 복잡하게 만드는 것

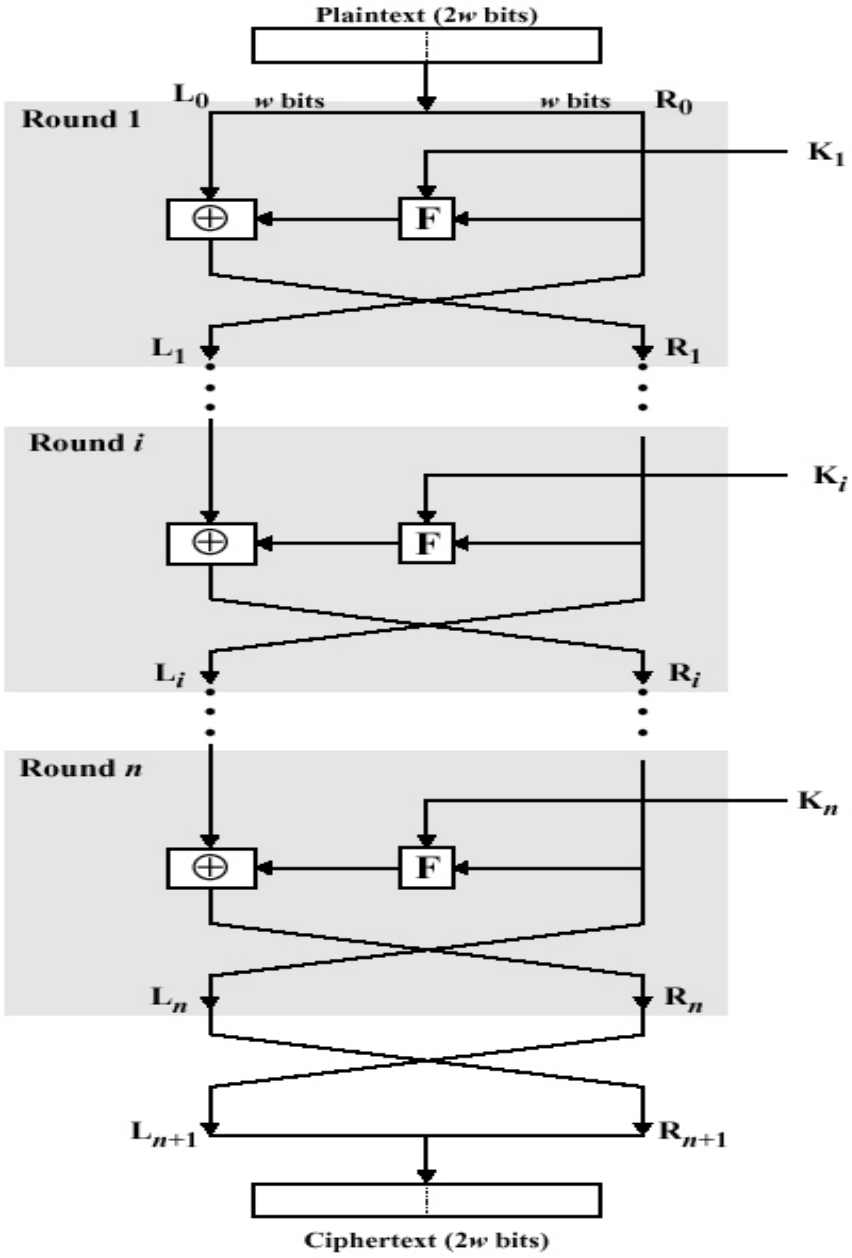
Feistel 암호 구조의 유도

- n비트 블록처리: n비트 평문을 입력으로 n비트 암호문 출력
  - 역으로 n 비트 암호문 입력에 대해 n비트 평문 출력(역의 성립: reversible, 비단수형: nonsingular)
  - 2^n 가지의 서로 다른 블록 존재 가능
- n 비트-n 비트 블록치환(n=4 인 경우)
  - 4비트입력으로 16개 값 중 하나 선택하고 , 내부 치환에 의하여 16개 출력 값 중 하나 대응하여 4비트 출력



Feistel 암호 구조

- 처리구조
  - 길이 2w 비트인 평문 블록(L_0, R_0) 분할 처리
  - K로부터 유도된 n개의 키(K_i) 사용
  - n회의 동일한 반복 구조 실행
- 그림: 고전 Feistel 구조
- 하나의 반복 구조
  - 오른 쪽 반 R0에 반복 함수 F 적용
    - 반복 서브키 K_1 적용($K \neq K_i$)
  - 왼쪽 반 L0와 XOR(치환 작용)
  - 좌우 양쪽 결과를 교환(순열 작용)



3.4.1 DES란

- DES(Data Encryption Standard)는 1977년에 미국의 연방정보처리표준규격(FIPS)으로 채택된 대칭암호
- RSA사가 주관한 DES의 DES Challenge 결과
  - 1997년의 DES Challenge Ⅰ에서는 96일,
  - 1998년의 DES Challenge Ⅱ-1에서는 41일,
  - 1998년의 DES Challenge Ⅱ-2에서는 56시간,
  - 1999년의 DES Challenge Ⅲ에서는 22시간 15분 만에 DES 키가 깨졌다.

3.4.2 암호화/복호화

- DES로 한 번에 암호화할 수 있는 것은 64비트뿐이다
- 그것보다 긴 비트 길이의 평문을 암호화하기 위해서는 평문을 블록 단위로 잘라낸 다음 DES를 이용해서 암호화를 반복할 필요가 있다.
- 이렇게 반복하는 방법을 **모드**(mode)라고 한다.

3.4.3 DES의 구조

- **페이스텔 네트워크**(Feistel network), **페이스텔 구조**(Feistel structure), 혹은 **페이스텔 암호**(Feistel cipher)
- **라운드**(round)라는 암호화의 1 단계를 여러 번 반복해서 수행

DES의 암호화 · 복호화

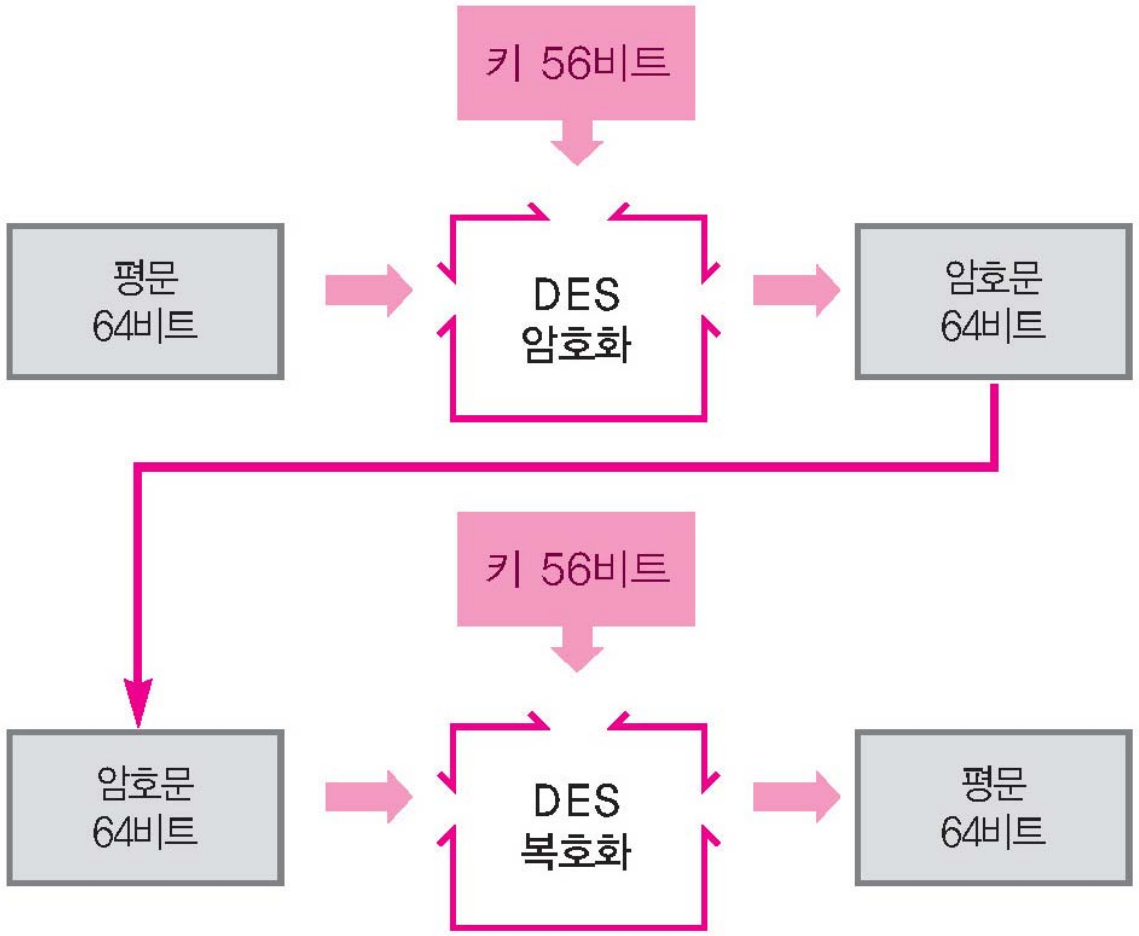


그림 3-8 DES의 암호화 · 복호화

페이스텔 네트워크의 1 라운드

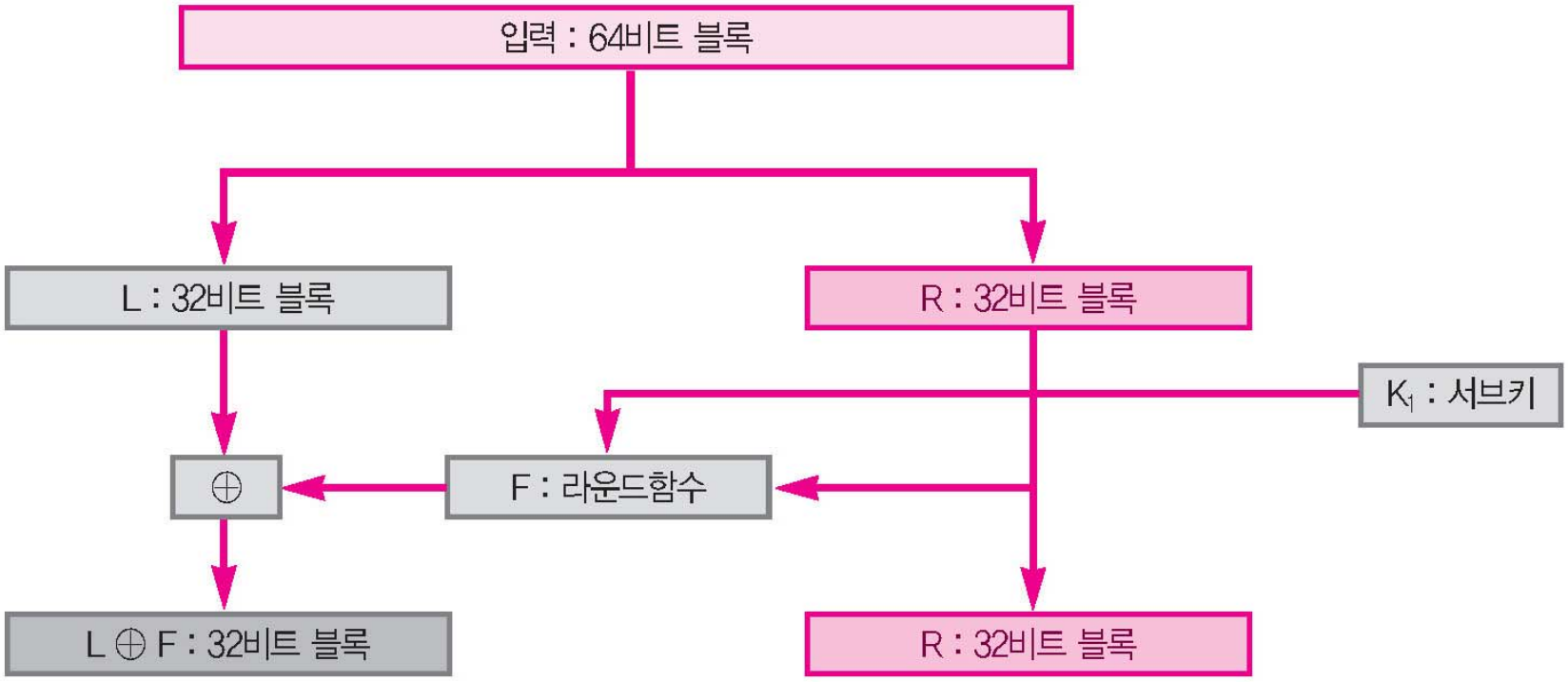


그림 3-9 페이스텔 네트워크 1 라운드

페이스텔 네트워크의 암호화(3라운드)

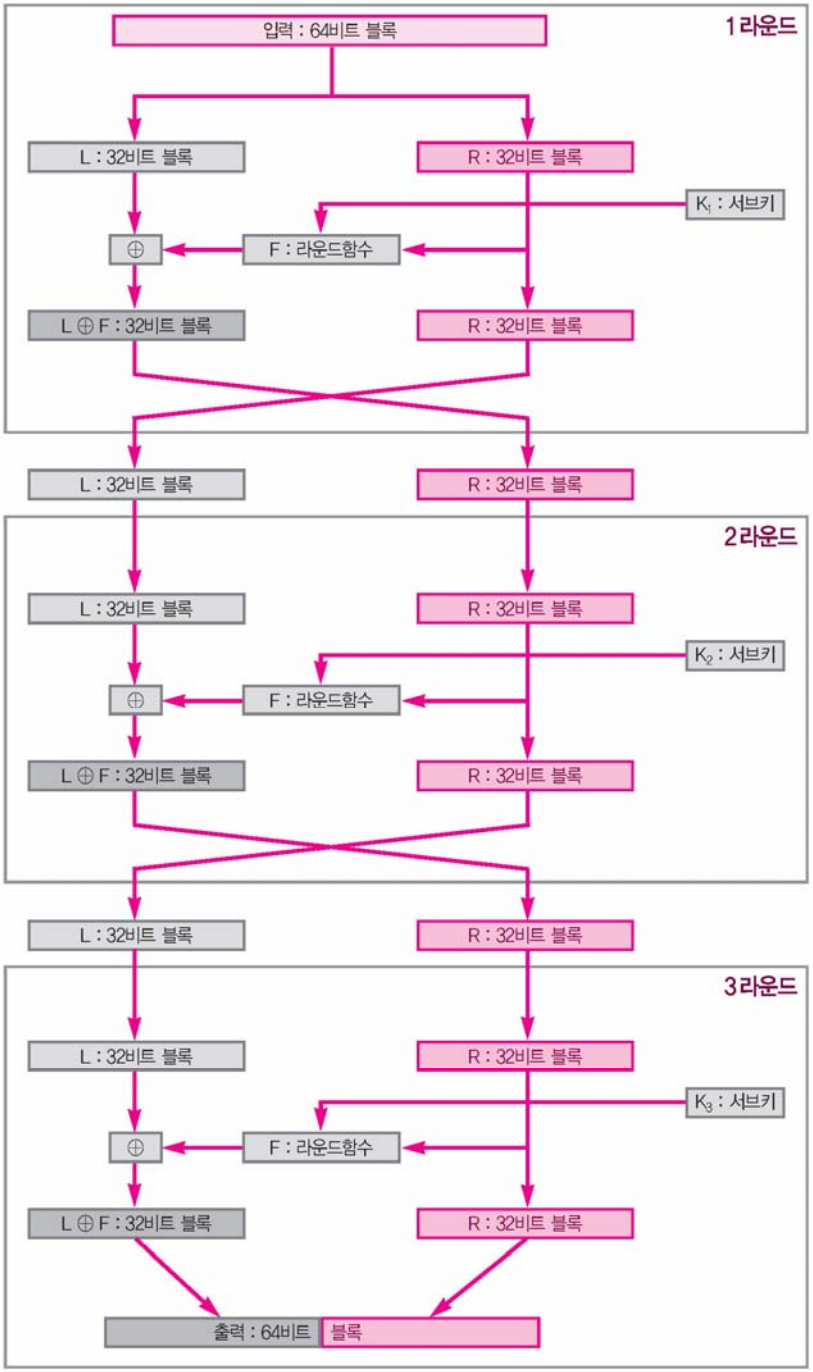


그림 3-10 페이스텔 네트워크의 암호화(3라운드)

같은 서브 키로 페이스텔 네트워크를 2회 통과시

키면 원래로 돌아간다

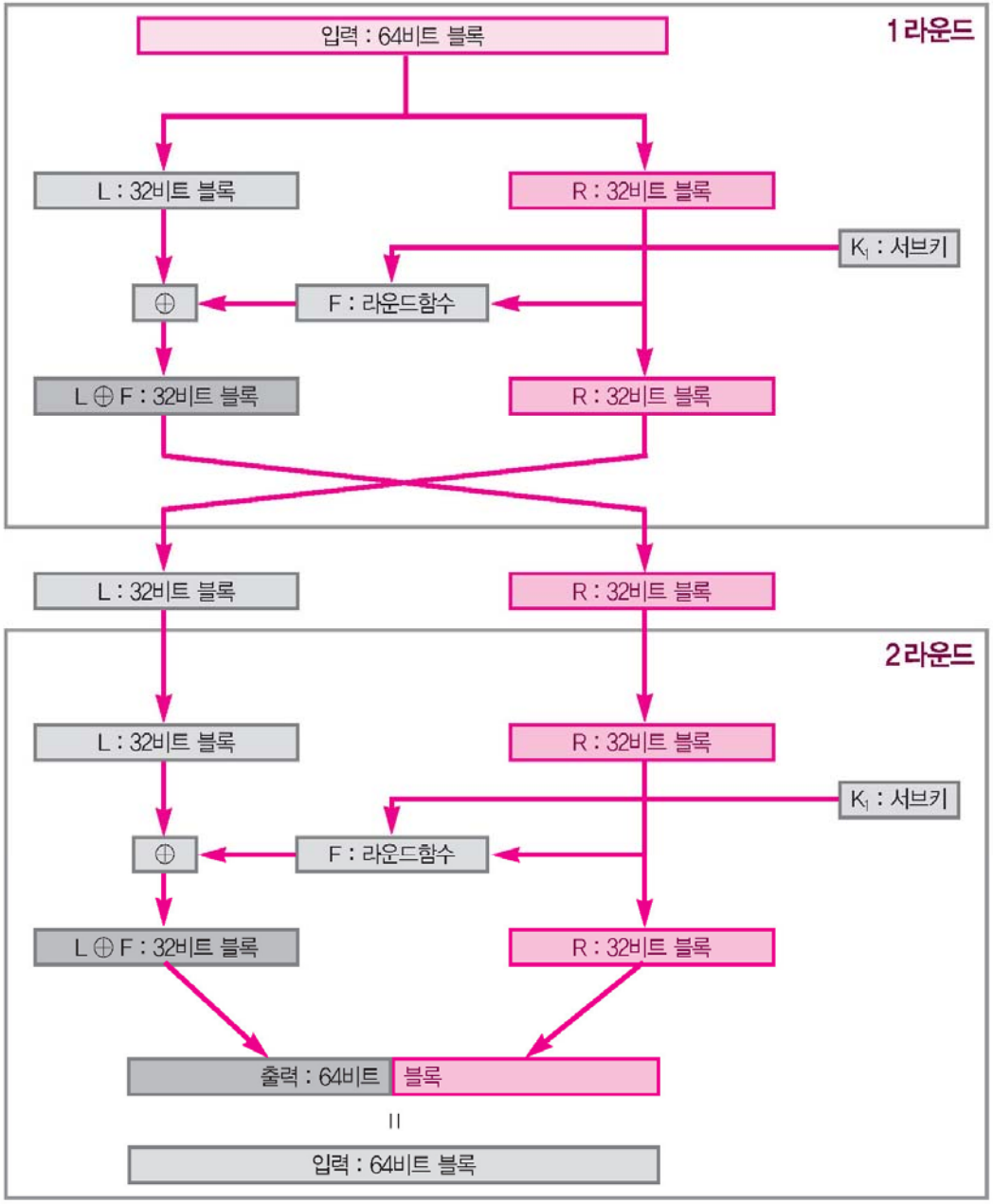


그림 3-11 같은 서브 키로 페이스텔 네트워크를 2회 통과시키면 원래로 돌아간다

페이스텔 네트워크의 복호화(3라운드)

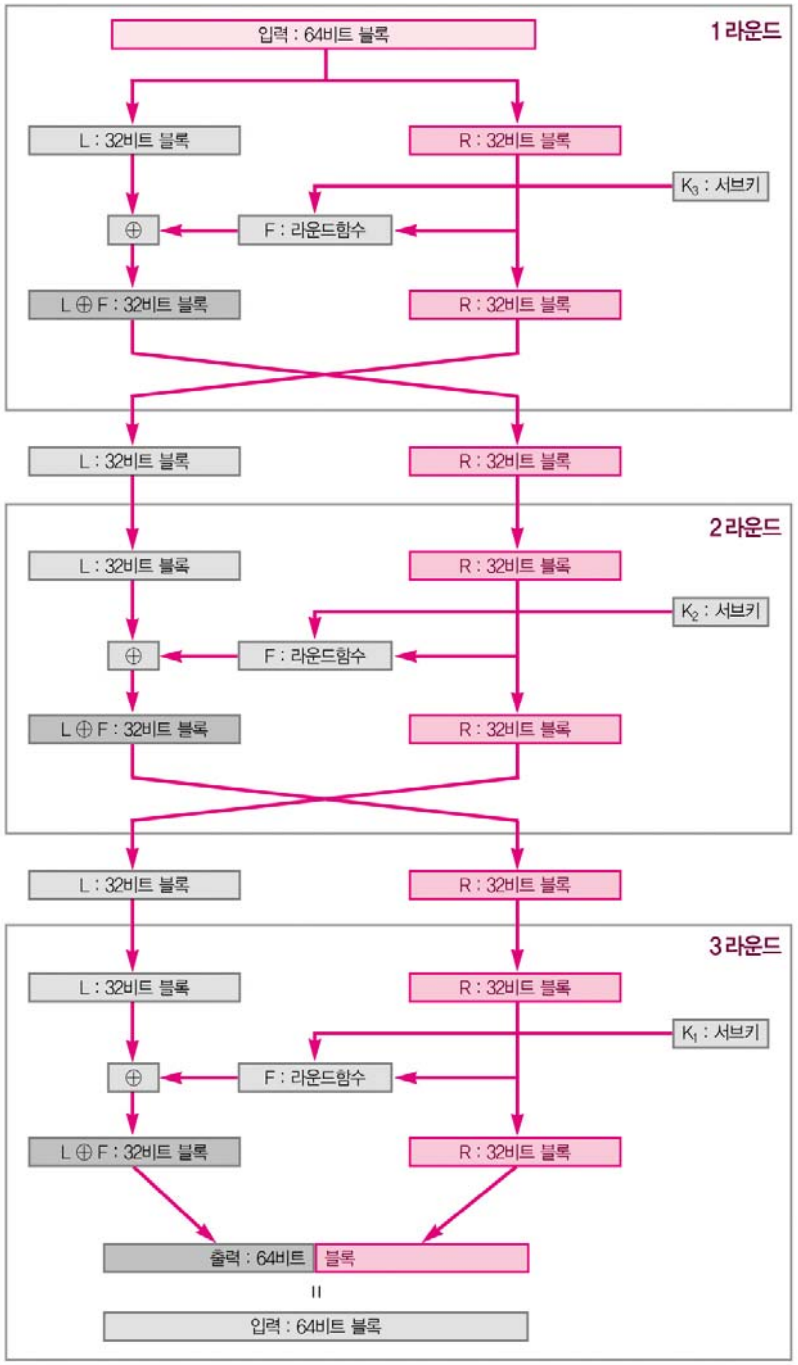


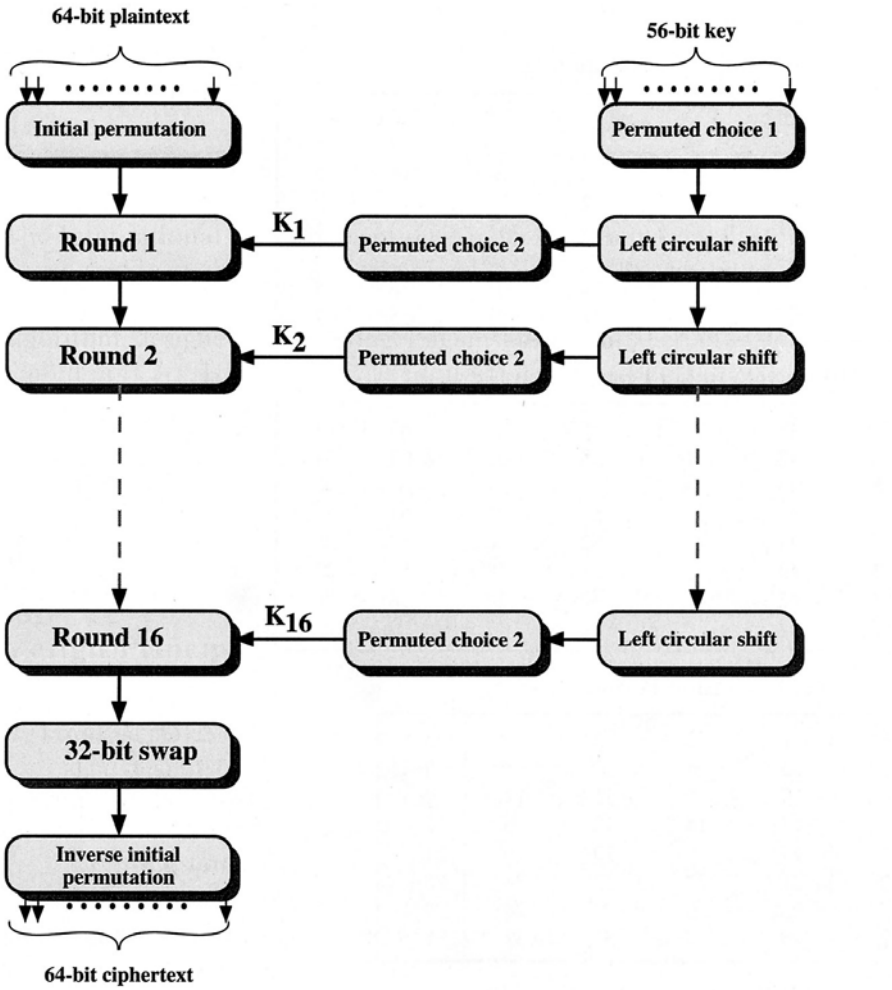
그림 3-12 페이스텔 네트워크의 복호화(3라운드)

페이스텔 네트워크의 성질

- 원하는 만큼 라운드수를 늘릴 수 있다
- 라운드 함수 F에 어떤 함수를 사용해도 복호화가 가능하다
- 암호화와 복호화를 완전히 동일한 구조로 실현할 수 있다

DES

- 암호화키 56비트를 이용하여 64비트 출력으로 변환



초기순열(IP)

58	50	42	34	26	18	10	3
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	61	43	35	27	19	11	3
61	63	45	37	29	21	13	5
63	55	47	39	31	23	15	7

역 초기 순열(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

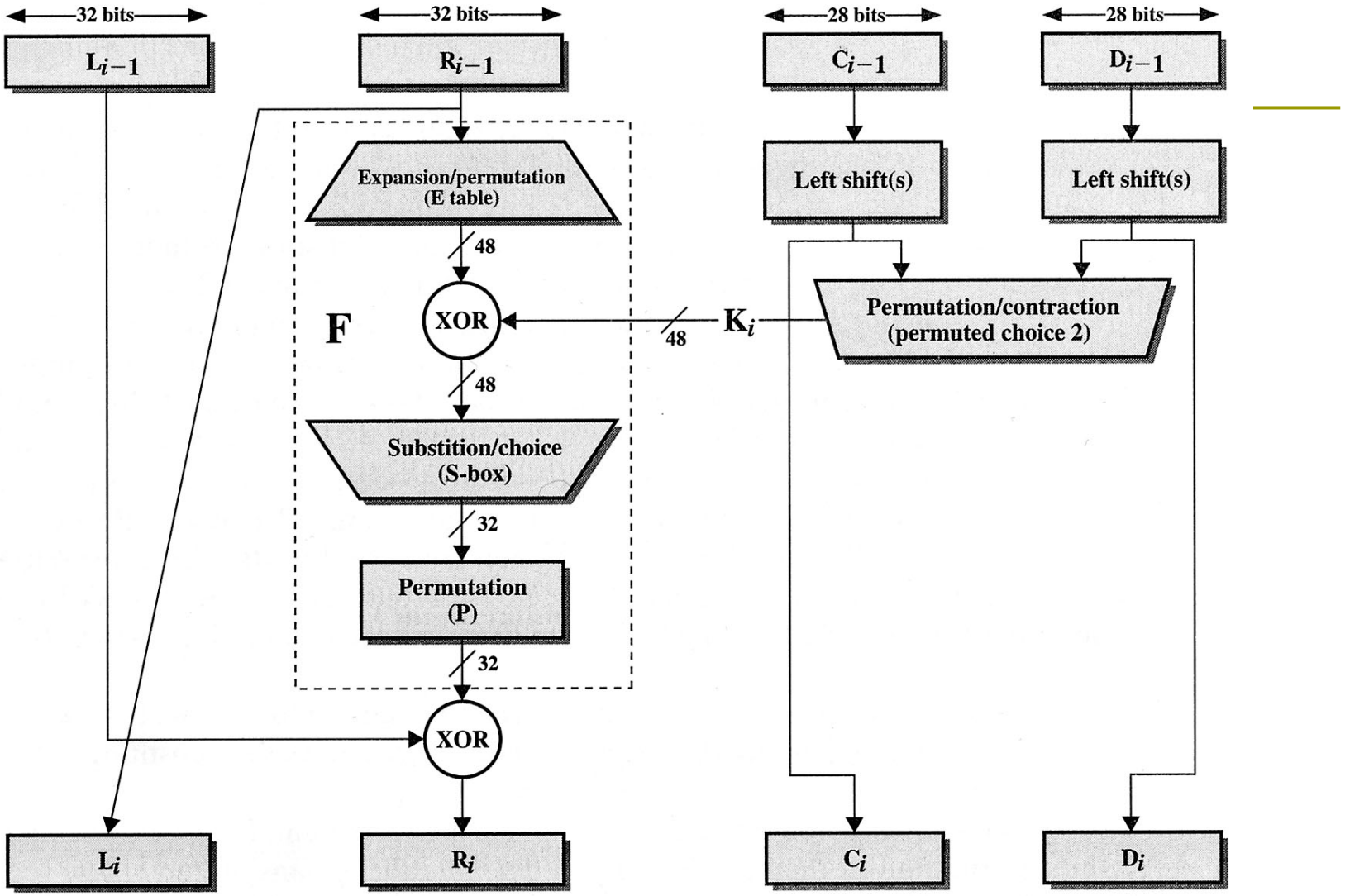
확장 순열(E)

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

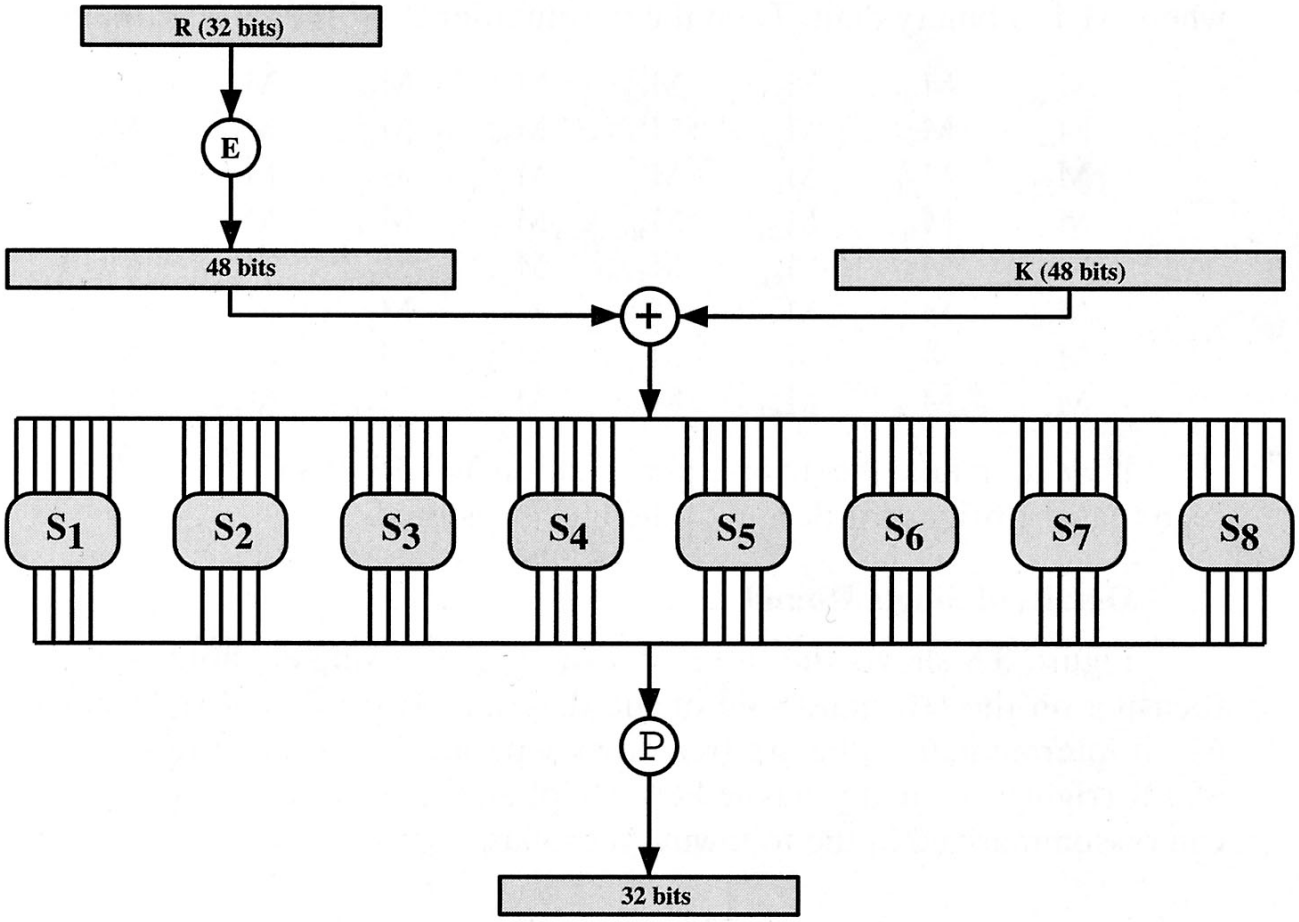
순열함수(P)

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

- DES 알고리즘의 단일 반복 과정



함수 F(R,K)의 계산



DES의 S-박스 정의

S1	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
S2	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
S3	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
S4	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
S5	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
S6	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
S7	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
S8	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

DES(Data Encryption Standard)

- 키 생성
  - DES 키 의 단계별 계산표
  - 순열선택1(PC-1)

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
63	55	47	39	31	23	15
7	63	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

- 순열선택2(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

3.4.4 DES 암호해독

- DES의 키의 길이가 56비트이므로 가능한 모든 키의 개수는 $2^{56} \approx 7.2 \times 10^{16}$ 개다.
- 한 개의 키를 선택해서 암호화하는 데 1 마이크로 초(10-6초)가 소요된다면 적용된 비밀키를 알아내는 데에는 약 1,000년이 소요된다.
- 1998년의 DES Challenge II-1 에서 22,000명의 참여자들이 협력하여 인터넷에 연결된 약 50,000대의 개인용 컴퓨터와 워크스테이션을 이용하여 72,057,594,037,927,936개의 키를 41일 만에 검색했다.

- 1998년의 DES Challenge II-2에서는 Electronic Frontier Foundation (EFF)에서 개발한 Deep Crack을 이용하여 56시간 만에 해결하였다.
- 이때 상금은 10,000달러였는데 EFF의 Deep Crack Machine을 만드는데 250,000달러가 들었다.
- 1999년의 DES Challenge III에서는 22시간 15분 만에 DES 키가 깨졌다.

3.4.5 쇄도효과

- 쇄도효과(avalanche effect) :
  - 평문 또는 키 값을 조금만 변경시켜도 암호문에는 매우 큰 변화가 생겨기는 효과

3.5 다중 DES

- 2중 DES: DES의 안전성을 증대시키기 위한 방법 중의 하나는 두 개의 서로 다른 키로 암호화를 두 번 수행
  - 중간충돌공격(meet-in-the-middle attack)에 취약
  -
- 트리플 DES

3.5.1 트리플 DES란

- 트리플 DES(triple-DES)는 DES보다 강력하도록 DES를 3단 겹치게 한 암호 알고리즘이다.

- 트리플 DES 혹은 3중 DES라 불리기도 하고 3DES 등으로 불리기도 한다.

트리플 DES 암호화

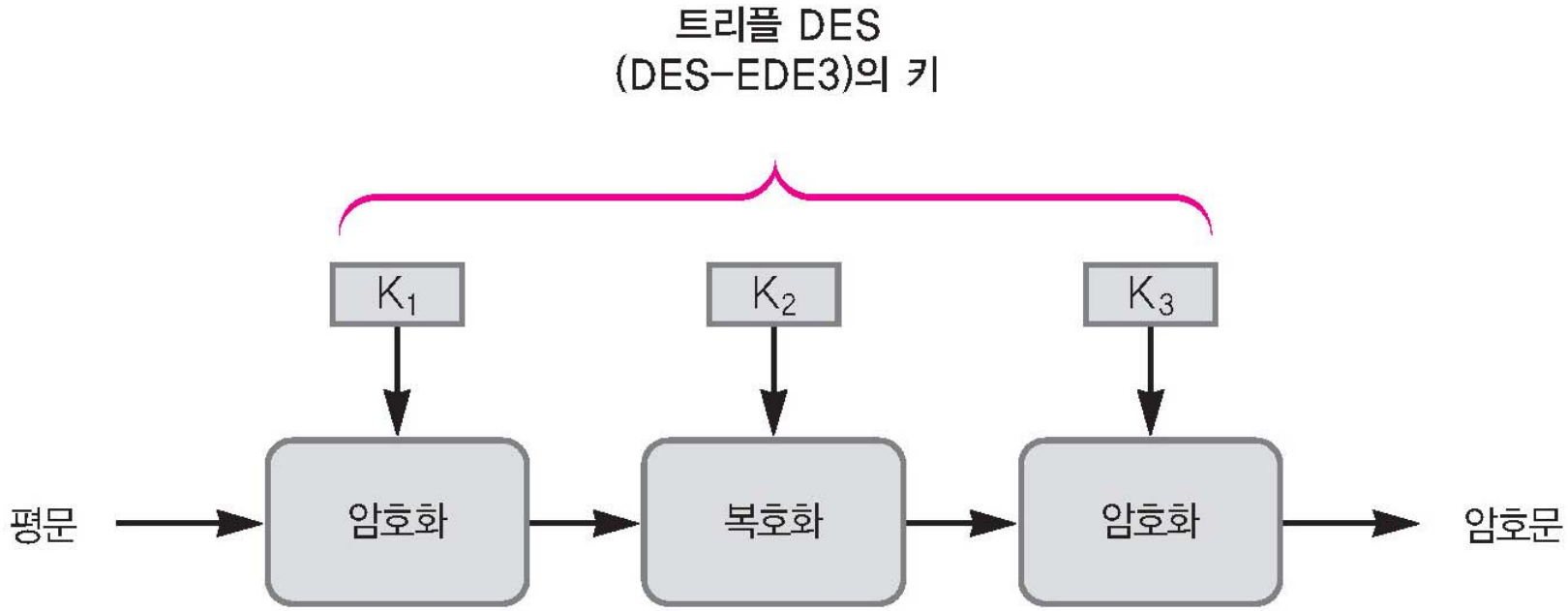


그림 3-13 트리플 DES의 암호화

트리플 DES를 DES로 사용

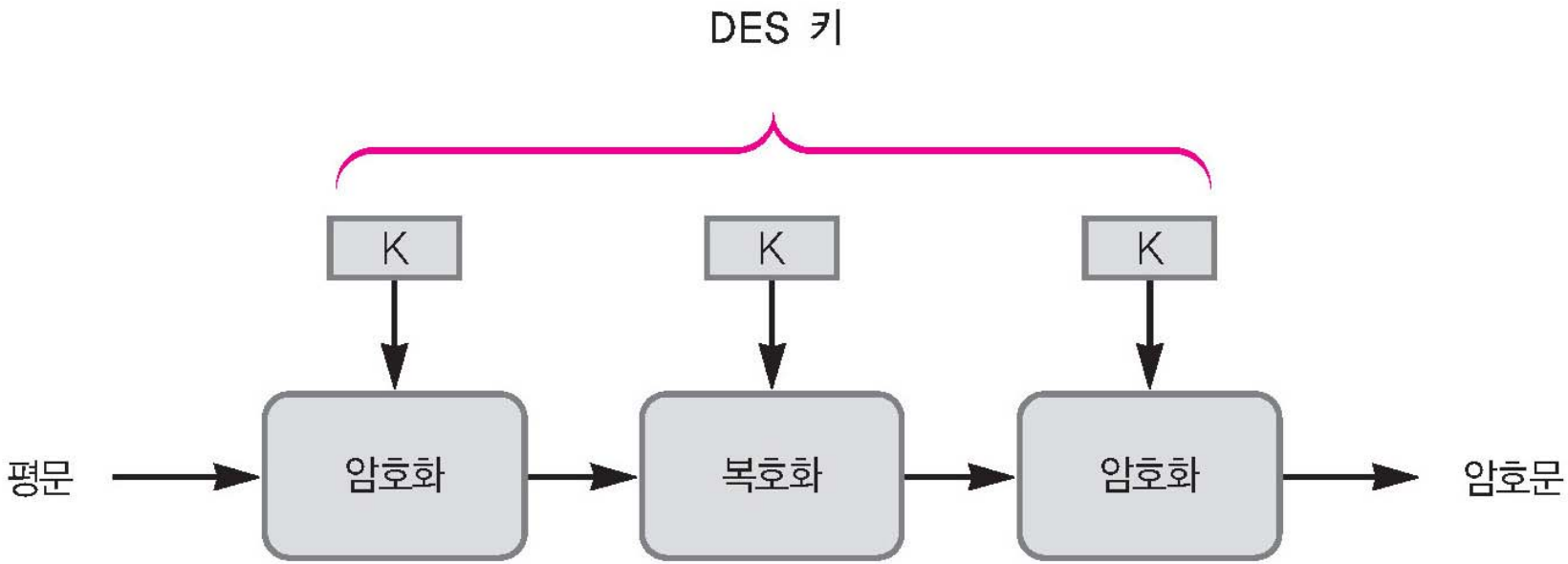


그림 3-14 트리플 DES는 DES로도 사용할 수 있다

DES-EDE2

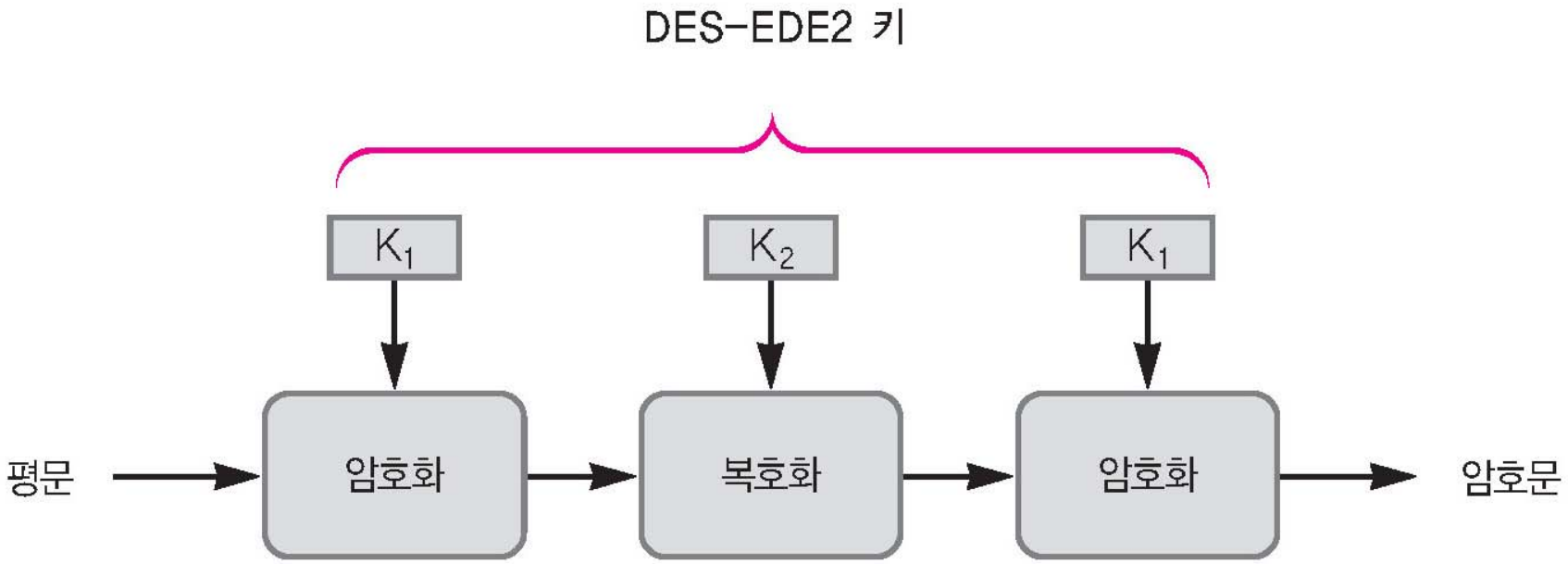


그림 3-15 DES-EDE2

트리플 DES(DES-EDE3)의 복호화

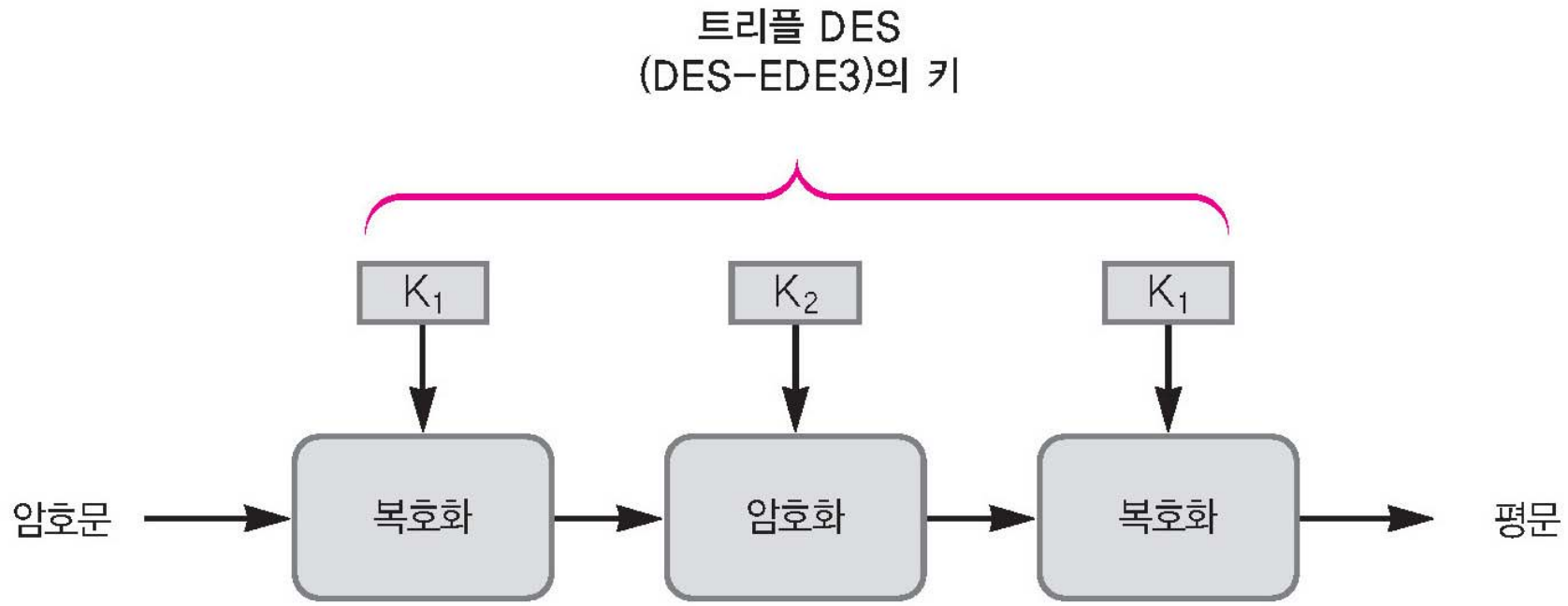


그림 3-16 트리플 DES(DES-EDE3)의 복호화

트리플 DES의 활용

- 금융권에서는 전자지불시스템 등에서 DES-EDE2는 아직 상당히 많이 사용되고 있다
- 트리플 DES의 처리 속도는 빠르지 않고 현재 암호표준으로 지정된 AES가 6배 정도가 더 빠르다

3.6 AES

- 대칭 암호의 새로운 표준인 AES에 대해 설명하도록 한다.

3.6.1 AES란

- AES(Advanced Encryption Standard)는 지금까지 표준이었던 DES를 대신하는 새로운 표준
- 전 세계의 기업과 암호학자가 AES의 후보로서 다수의 대칭암호 알고리즘을 제안
- 그 중에서 Rijndael이라는 대칭암호 알고리즘이 2000년에 AES로서 선정

3.6.2 AES의 선정 과정

- AES를 공모한 것은 미국의 표준화기구인 NIST(National Institute of Standard and Technology)이다.
-
- AES의 응모 조건
  - 무료로 이용할 수 있어야 한다
  - 암호 알고리즘의 규격이 적힌 사양서, ANSI와 Java에 의한 구현, 암호해독에 대한 강도의 평가
  - 제안되는 암호 알고리즘은 설계 규격과 프로그램을 공개

3.6.3 AES 최종 후보 및 선정

- 1997년 1월 2일 NIST는 AES의 모집을 개시
- 조건
  - 속도가 빠를 것, 단순하고 구현하기 쉬울 것
  - 암호화 자체의 속도뿐만 아니라 키의 셋업 속도도 중요
  - 스마트카드나 8비트 CPU 등의 계산력이 작은 플랫폼에서부터 워크스테이션과 같은 고성능의 플랫폼에 이르기까지 효율적으로 동작
  - 블록 길이가 128 비트인 대칭 블록 암호이어야 하고 키의 길이는 128, 192, 와 256 비트를 지원

모집요구 조건

- 평가 항목
  - 보안
  - 계산적 효율성
  - 메모리 요구량
  - 하드웨어와 소프트웨어적 적합성
  - 유연성

최초 평가대상(15개)

- CAST256,
- Crypton,
- DEAL,
- DFC,
- E2,
- Frog,
- HPC,
- LOKI97,
- Magenta,
- MARS,
- RC6,
- Rijndael,
- SAFER+,
- Serpent,
- Twofish

AES의 최종후보(알파벳순 5개)

표 3-1 AES의 최종후보(알파벳순)

명칭	응모자
MARS	IBM사
RC6	RSA사
Rijndael	Daemen, Rijmen
Serpent	Anderson, Biham, Knudsen
Twofish	Counterpane사

3.6.4 Rijndael

- Rijndael의 설계자는 벨기에의 2 연구자
  - Joan Daemen
  - Vincent Rijmen
- 가 설계한 블록 암호 알고리즘
- Rijndael의 블록 길이는 128비트이고 키의 비트 길이는 128비트부터 256비트까지 32비트 단위로 선택할 수 있다

Rijndael의 암호화와 복호화

- 복수(키의 길이에 따라 10, 12, 14)의 라운드로 구성
- SPN 구조
- 기본 처리
  - SubBytes
  - ShiftRows
  - MixColumns
  - AddRoundKey

Rijndael 암호화의 1라운드

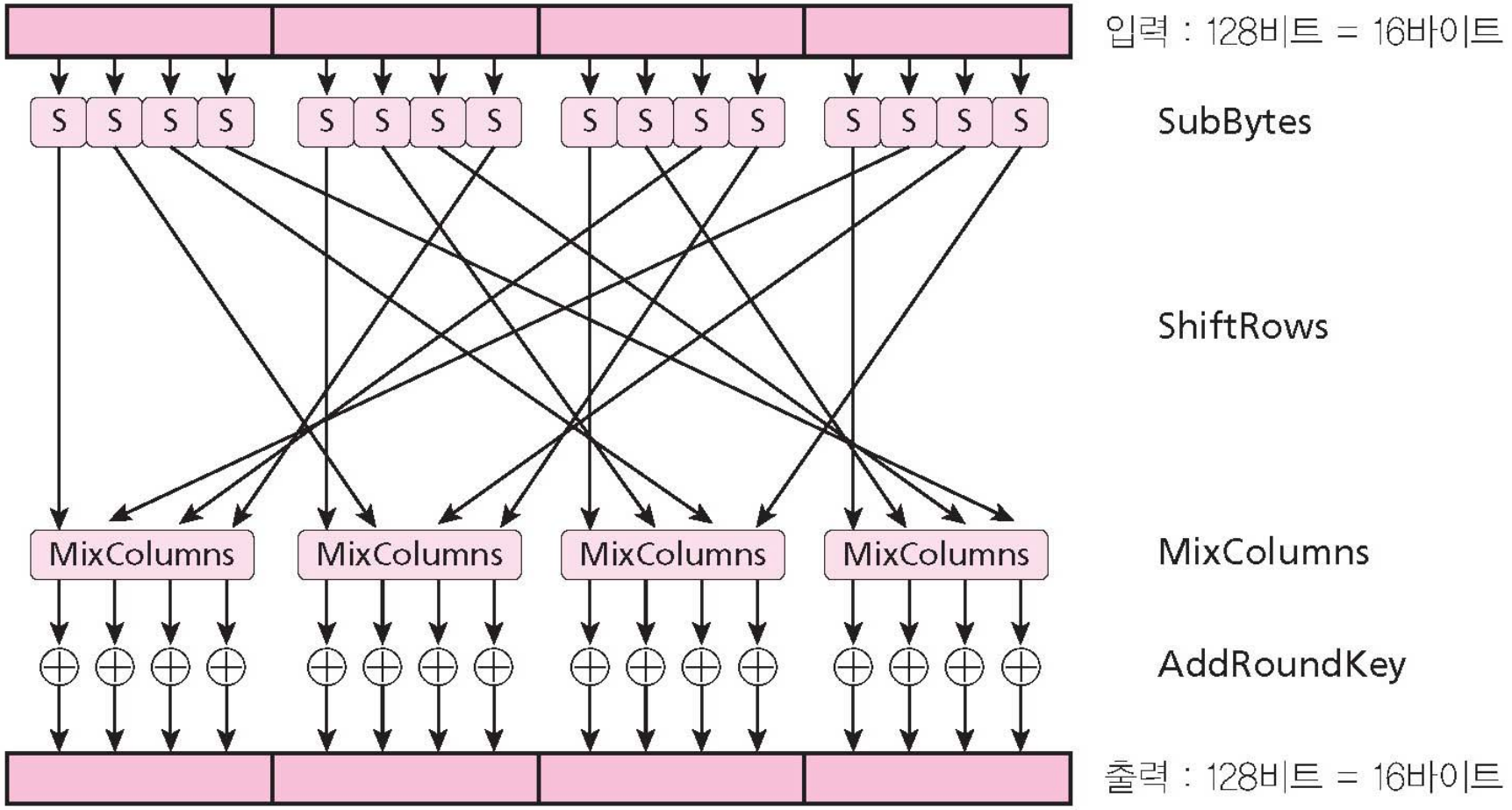


그림 3-17 Rijndael 암호화의 1라운드

Rijndael 복호화의 1라운드

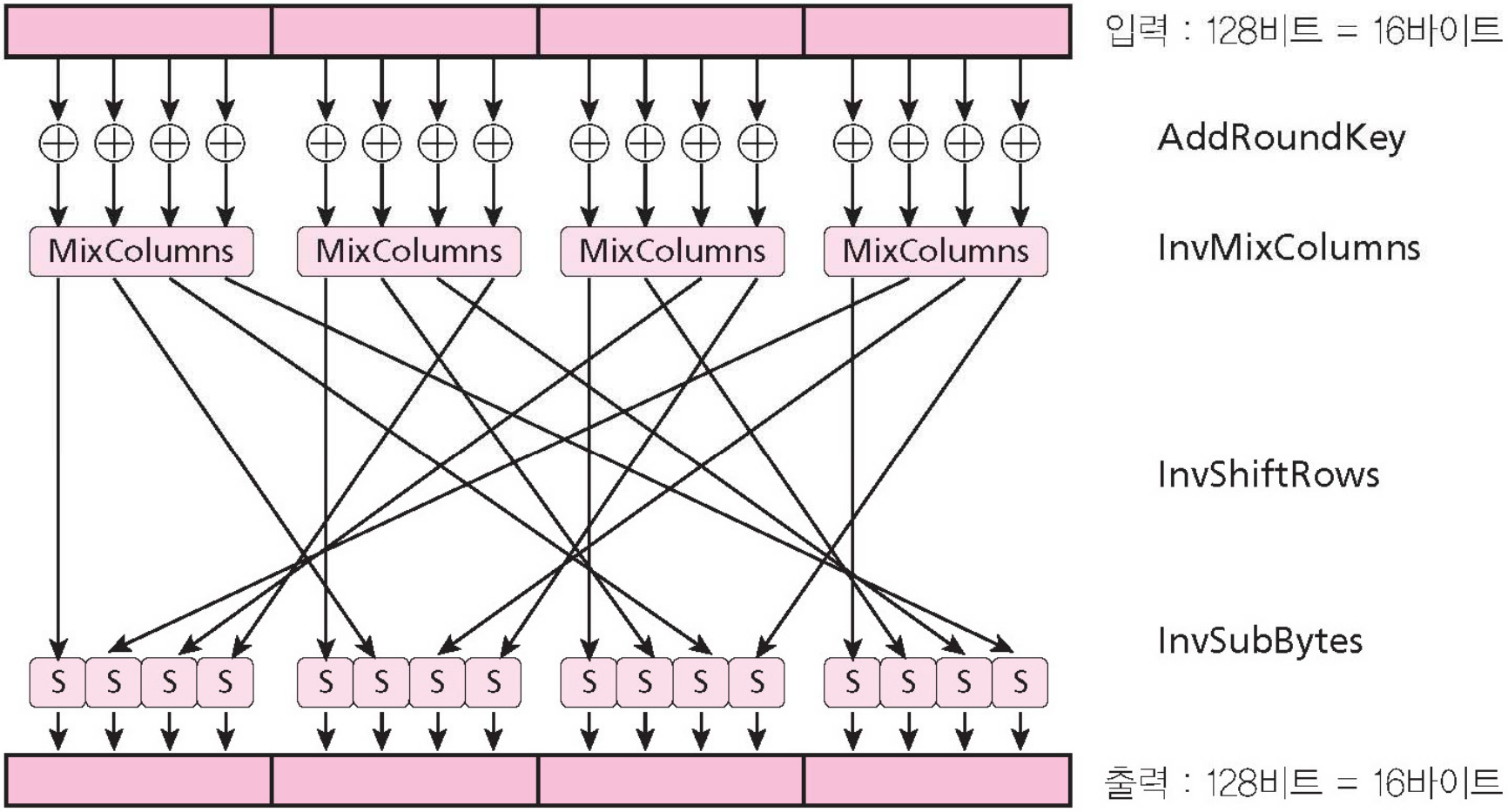


그림 3-18 Rijndael 복호화의 1라운드

Rijndael의 해독

- Rijndael에 대한 유효한 공격은 발견되지 않았다.

- 오직 알고리즘의 이론적인 약점을 공격하는 것이 아니고 알고리즘을 구현하는 하드웨어를 통한 시간정보, 전기 소모량, 자기장, 소음 등을 이용한 주변채널공격(side channel attack)만이 보고됨.

가장 좋은 대칭암호

- AES(Rijndael)가 좋을 것이다.
- 안전하고 처리 속도가 고속이며 게다가 폭넓은 플랫폼에서 이용할 수 있기 때문이다.
- 또한 AES는 전 세계의 연구자가 검증하고 있으므로, 만일 결함이 발견되더라도 전 세계에 널리 알려져 해결될 가능성이 높다.

질의 및 응답

– 끝–