2021/08/27, 2021 보안 세미나

암호학과 네트워크 보안

- DES, AES-

이 하 늘(haneul@pel.sejong.ac.kr)

세종대학교 프로토콜공학연구실

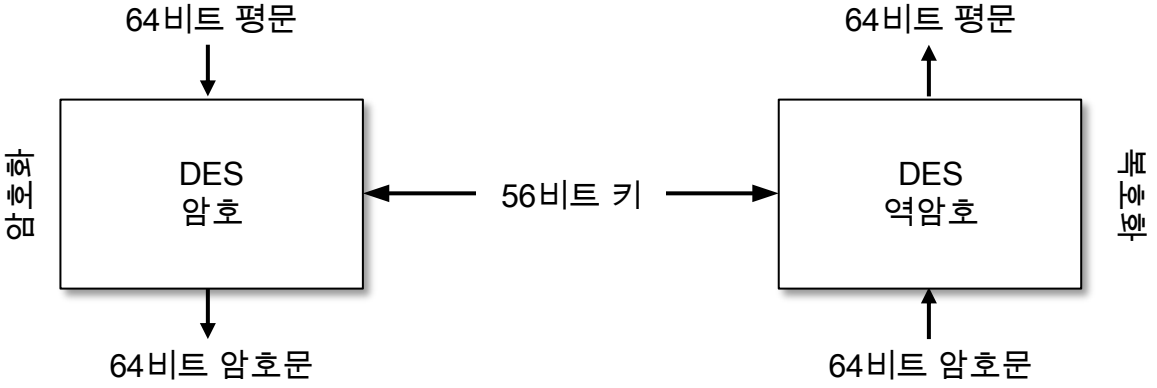
목 차

DES의 개요

- 역사
  - 1973년 미국국립기술표준원이 국가적으로 사용할 대칭키 암호시스템의 제안요청서를 발표
  - IBM의 제안이 DES로 채택됨
  - 1975년 3월에 연방관보에서 연방정보처리기준의 초안으로 공표

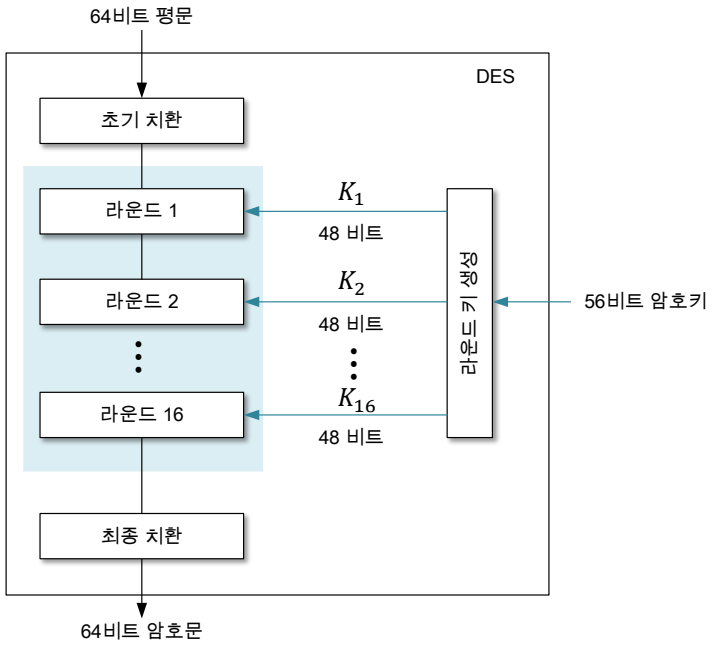
DES의 개요

- 구조 개요
  - 암호화 과정
    - 64비트 평문을 가지고 64비트 암호문 생성
  - 복호화 과정
    - 64비트 암호문을 가지고 64비트 평문 생성
  - 암호 키
    - 56비트 암호 키가 암호화와 복호화에 사용



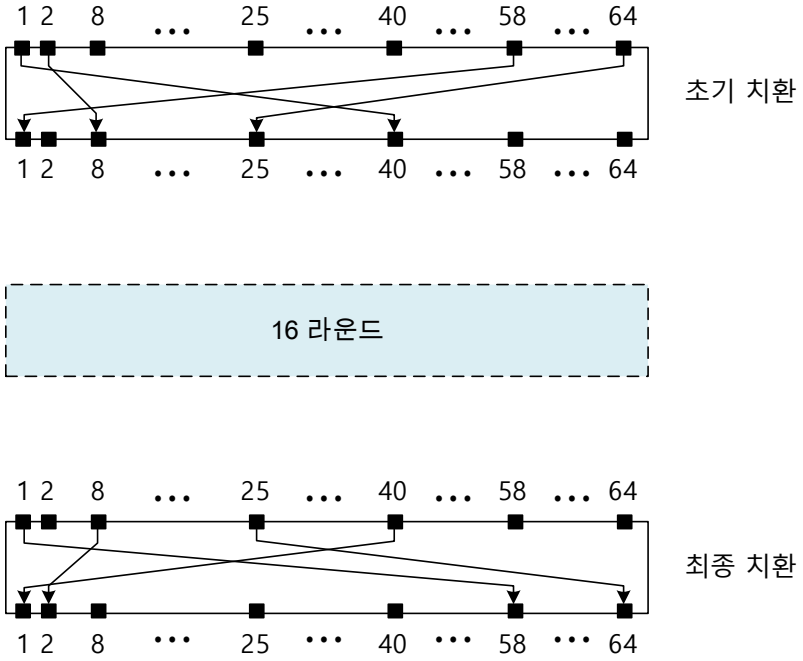
DES의 구조

- 구조
  - 두 개의 치환 박스(P-박스)와 Feistel 라운드 함수로 구성
  - 두 개의 P-박스는 초기 치환과 최종 치환이 있음
  - 각 라운드는 라운드 키 생성기에 생성된 48비트 라운드 키 사용



DES의 구조

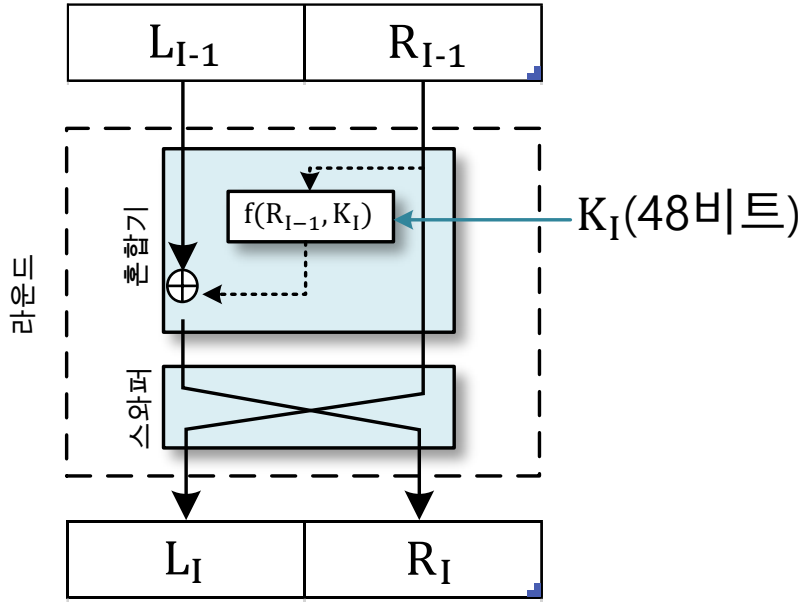
- 초기 치환과 최종 치환



- 입력 포트와 그에 대응하는 출력 포트 보유
- 초기 치환과 최종 치환은 서로 역 관계이며, 키가 없는 단순 치환

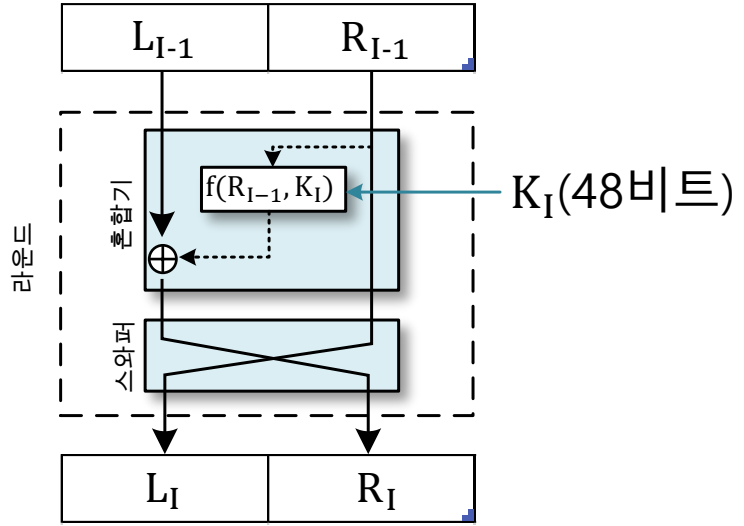
DES의 구조

- 라운드
  - 각 라운드는 Feistel 암호
  - 이전 라운드의 출력 값 L_{I-1}과 R_{I-1}을 입력으로 받음
  - 다음 라운드에 입력으로 적용될 L_1과 R_1 생성



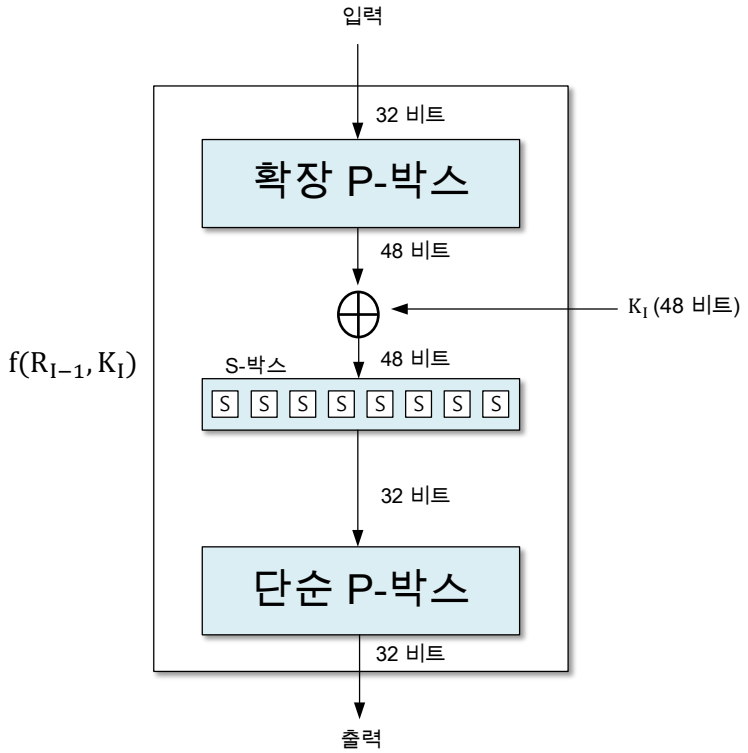
DES의 구조

- 라운드
  - 암호 요소 혼합기와 스와퍼
    - 암호 요소 혼합기는 XOR연산
    - 스와퍼는 텍스트의 오른쪽 절반을 왼쪽 절반과 교환



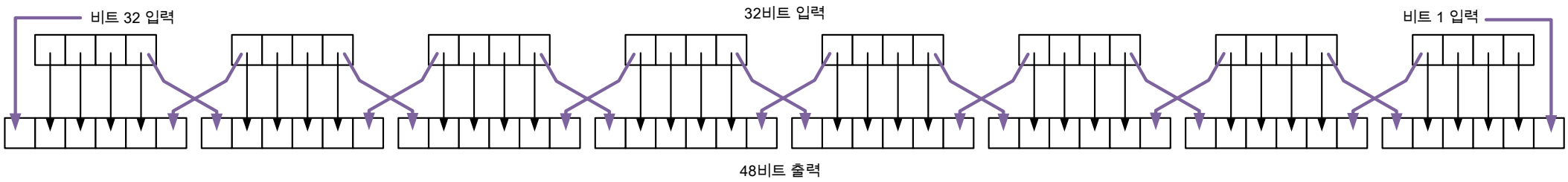
DES의 구조

- 라운드
  - DES 함수
    - 라운드 함수에 사용된 $f(R_{I-1}, K_I)$를 가리킴
    - 확장 P-박스, 키 XOR, 8개의 S-박스, 단순 P-박스로 구성



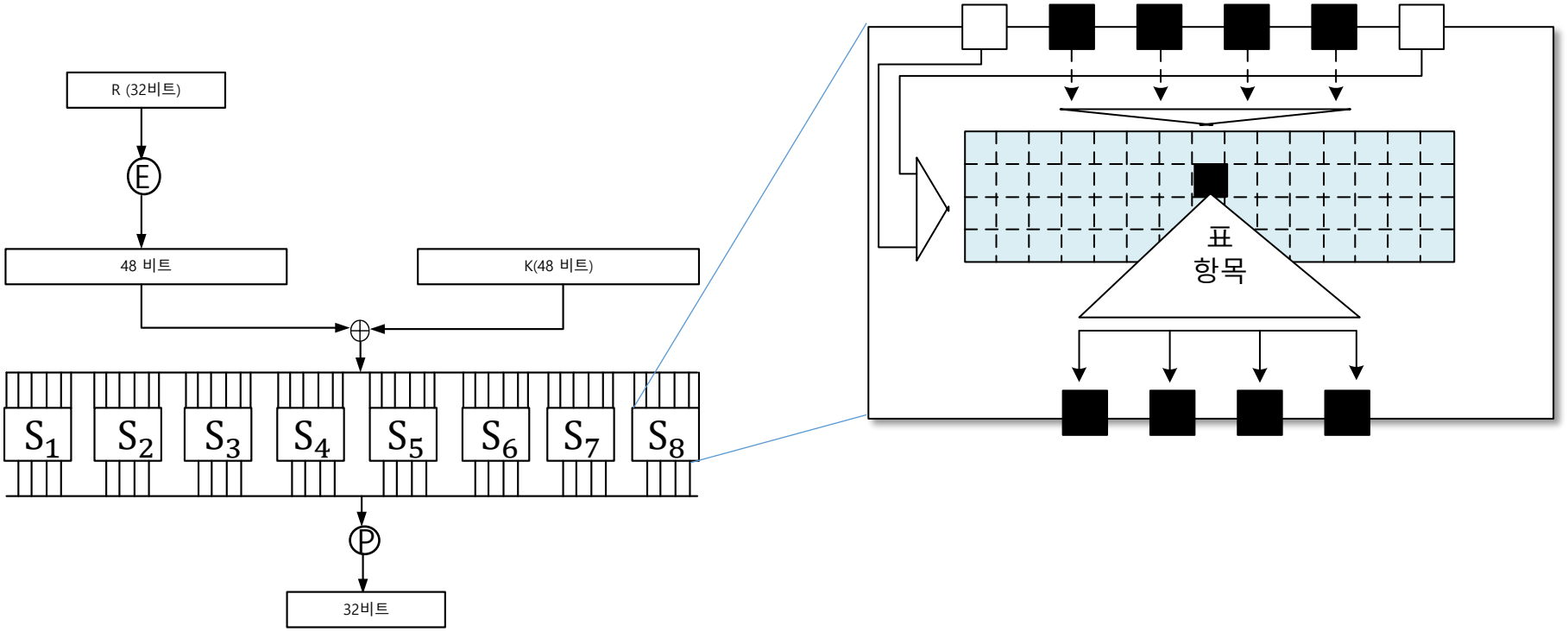
DES의 구조

- 라운드
  - DES 함수
    - 확장 P-박스
      - 32비트 R_{I-1}를 48비트로 확장하는 과정
        1. 32비트를 8개의 4비트로 나누어 6비트로 확장
        2. 입력 비트 1,2,3,4는 출력 비트 2,3,4,5로 복사
        3. 출력 비트 1은 이전 4비트의 입력 값의 4비트로부터 나옴
        4. 출력 비트 6은 다음 4비트 입력 값의 비트 1로부터 나옴



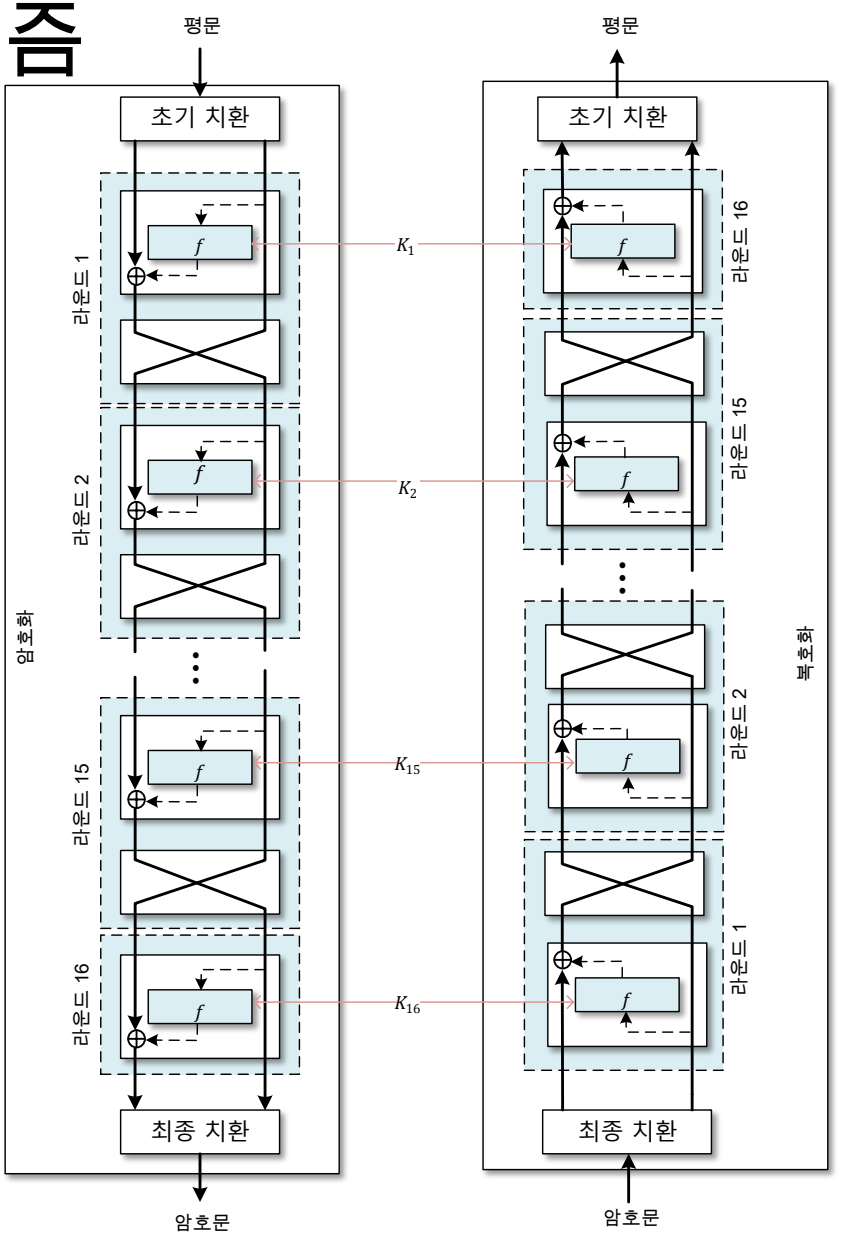
DES의 구조

- 라운드
  - DES 함수
    - S-박스
      - 48비트 값을 32비트로 S-박스에서 압축
      - 8개의 6비트 값을 4비트 값으로 출력



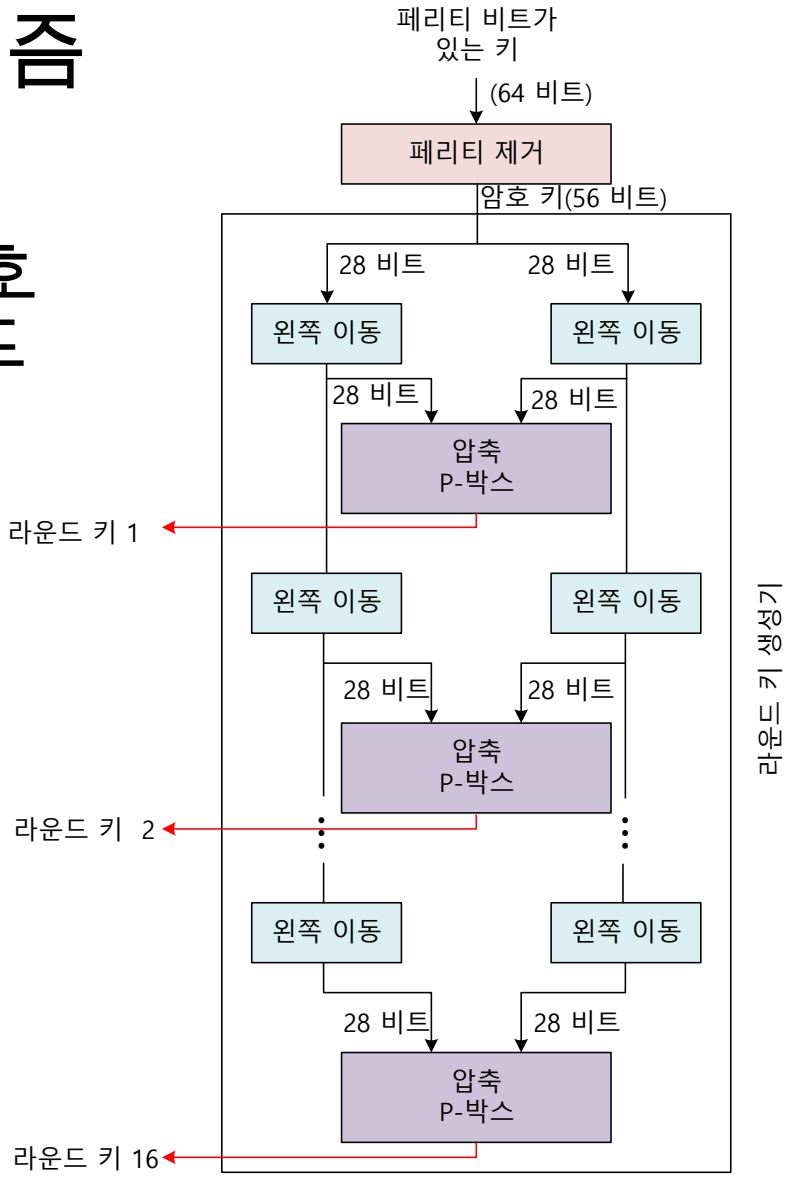
DES의 구조

- 암호 알고리즘과 복호 알고리즘
  - 마지막 라운드 함수에는 스와퍼가 없고 혼합기만 존재
  - 암호 알고리즘에 들어가는 라운드 키 입력 순서를 거꾸로 하면 복호 알고리즘이 됨
    - 라운드 키들이 역순으로 적용
    - 1라운드는 K_1을 사용하고 16 라운드는 K_{16}사용



DES의 구조

- 암호 알고리즘과 복호 알고리즘
  - 키 생성
    - 라운드 키 생성기는 56비트 암호 키로부터 16개의 48비트 라운드 키를 만듦
    - 암호화 키는 64비트 키로 주어지며, 이 중 8비트는 패리티 비트로 실제 키 생성 과정 전에 제거



DES의 구조

- 암호 알고리즘과 복호 알고리즘
  - 키 생성
    - 패리티 제거
      - 64비트 키에서 패리티 비트들을 버리고, 남은 비트들을 치환
      - 남아있는 56비트 값은 라운드 키를 생성하기 위하여 사용되는 실제 암호 키
    - 좌측 순환 이동
      - 56비트 키를 두 개의 28비트로 나눔
      - 왼쪽으로 1 또는 2비트를 순환 이동
        - 1,2,9,16번째 라운드에서 1비트, 나머지는 2비트

라운드	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
이동	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

DES의 구조

- 암호 알고리즘과 복호 알고리즘
  - 키 생성
    - 축소 치환
      - 축소 치환(P-박스)은 56비트를 48비트로 바꾸는 데 사용
      - 48비트 출력 값은 한 라운드의 라운드 키로 사용

DES 분석

- 특성
  - 큰 쇄도 효과
    - 키의 작은 변화가 결과값에 큰 영향을 미치는 효과가 큼
    - 암호문으로부터 키를 유추하기 힘듦
  - 높은 완전성
    - 암호문 각각의 비트가 원래의 내용의 비트에 의존적
    - P-박스와 S-박스에 의해 높은 완전성 효과를 보여줌
    - 암호문으로부터 평문을 유추하기 힘듦

DES 분석

- 취약성
  - 암호 키 취약점
    - 키 크기
      - 공격자는 2^{56}비트를 조사 해야함
        - 전수조사에 취약
        - 1977년 한 연구팀은 인터넷에 연결된 3500개의 컴퓨터를 이용하여 120일 만에 키를 찾아냄
        - 3500대의 컴퓨터가 120일 만에 키를 찾을 수 있다면 42000명의 회원을 가진 비밀 집단은 10일 안에 키를 찾을 수 있게 됨
    - 취약 키

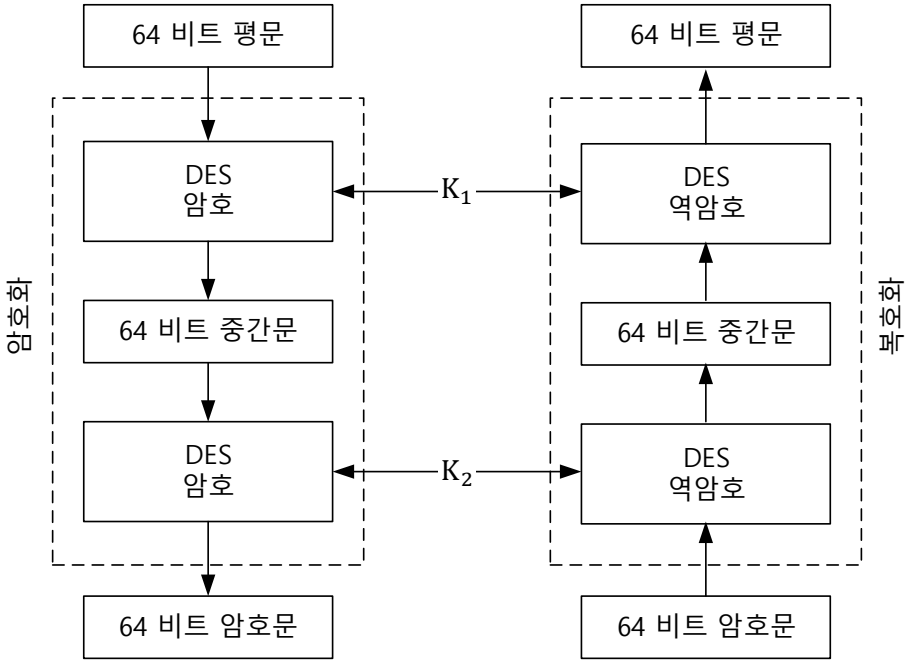
페리티 제거 전 키(64 비트)	실제 키(56 비트)
0101 0101 0101 0101	0000000 0000000
1F1F 1F1F 1F1F 1F1F	0000000 FFFFFFF
E0E0 E0E0 F1F1 F1F1	FFFFFFF 0000000
FEFE FEFE FEFE FEFE	FFFFFFF FFFFFFF

      - 모두 0이거나 1이거나 절반은 0이고 절반은 1인 키

다중 DES

- 개요
  - 짧은 키 길이에 대한 전수조사 공격 예방
  - 여러 개의 키를 가지고 DES를 여러 번 암호화하기 위함

- 이중 DES
  - DES과정을 두 번씩 진행

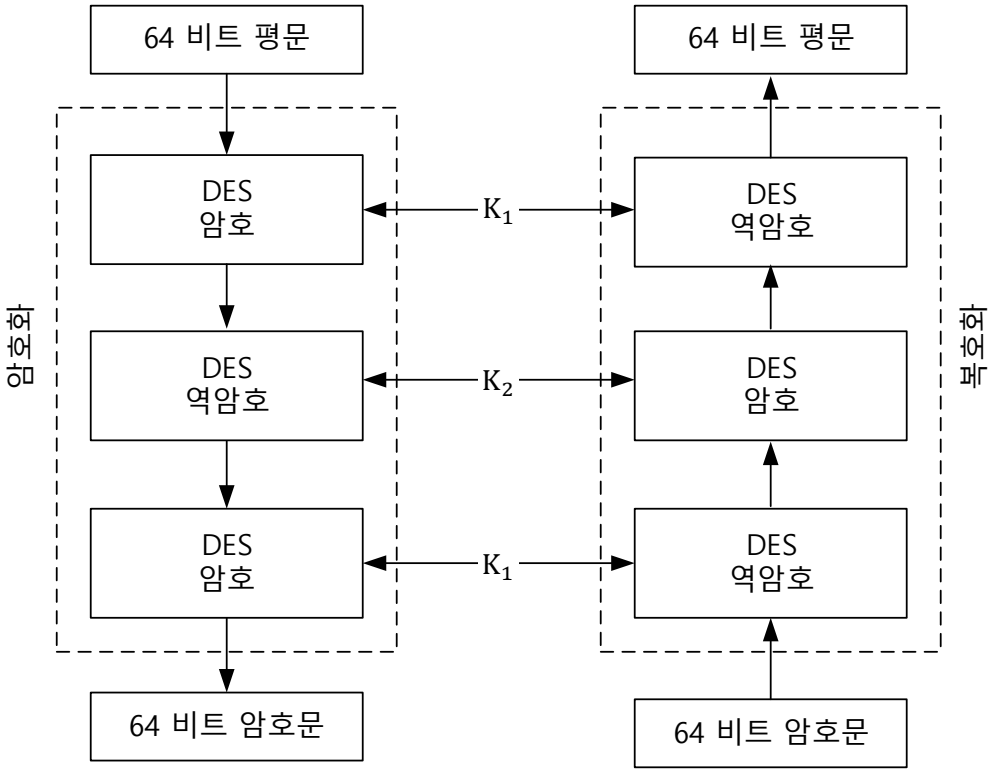


다중 DES

- 이중 DES
  - 중간일치 공격에 취약
    - 키 사이즈가 두 배인 112비트가 되는 것이 아닌 57비트로 약간의 향상만 있을 뿐임
    - 중간일치 공격을 수행하여 이전의 평문과 암호문을 가로채어 전수조사 공격 실행

다중 DES

- 삼중 DES
  - 두 개의 키를 갖는 삼중 DES(암호화-복호화-암호화)
    - 두 개의 키 K_1과 K_2만 사용
      - 첫 번째와 세 번째 단계는 K_1사용
      - 두 번째 단계는 K_2사용
      - 기지평문 공격에 취약



다중 DES

- 삼중 DES
  - 세 개의 키를 갖는 삼중 DES
    - 두 개의 키를 갖는 삼중 DES의 기지평문 공격의 가능성 때문에 어떤 응용프로그램에선 세 개의 키를 갖는 삼중 DES 사용

      e.g.,PGP와 같은 응용프로그램에서 사용
    - 복호화는 암호화의 역이 됨
      - K_3 ,K_2, K_1 순으로 복호화-암호화-복호화

DES의 안전성

- 전수조사 공격
  - 짧은 길이의 암호 키 사용
  - 최근 DES
    - 두 개의 키를 갖는 삼중 DES 또는 세 개의 키를 갖는 삼중 DES 사용으로 전수조사 공격 예방
- 차분 공격
  - 라운드 수를 16으로 설정하여 차분 공격에 대비
  - 차분 공격에 대비한 S-박스 설계

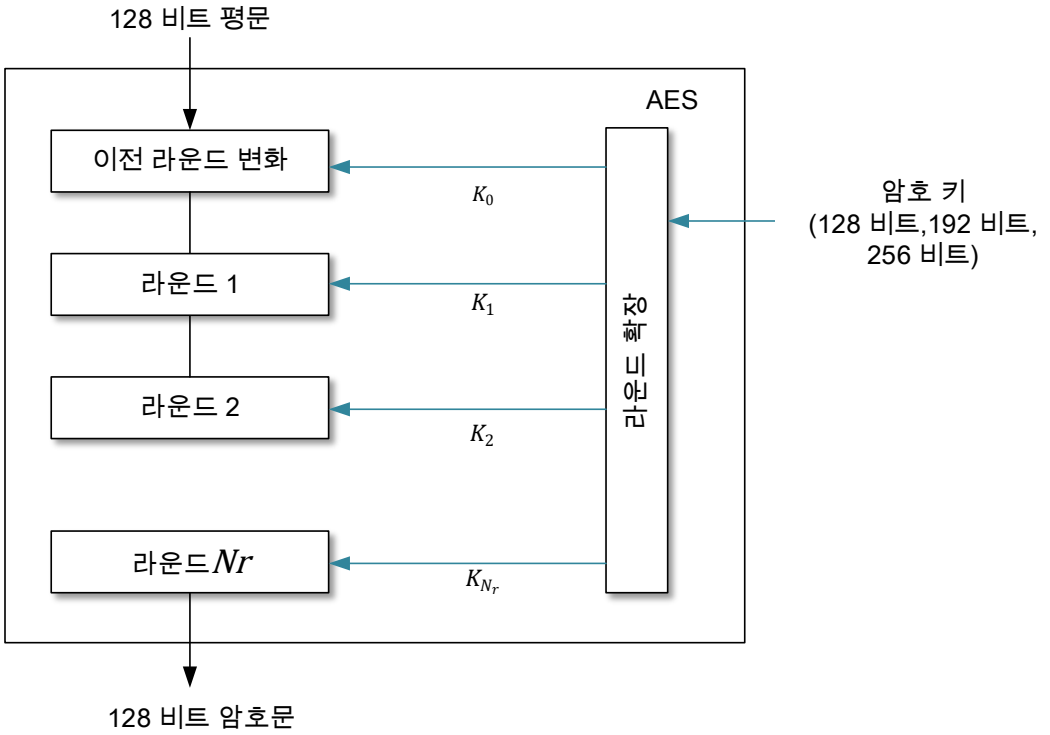
목 차

AES의 개요

- 역사
  - 1997년 미국 국립기술표준원은 AES로 불리는, DES를 대체할 암호 알고리즘 공모
  - 2001년 2월 연방정보처리기준(FIPS, Federal Information Processing Standard)에 초안을 발표
  - 2001년 10월 연방관보에 FIPS 197로 공표

AES의 개요

- 라운드
  - AES는 128비트 평문을 128비트 암호문으로 출력하는 알고리즘
  - 키 길이에 따라 3가지 버전이 존재
    - 128비트 키는 10 라운드
    - 192비트 키는 12 라운드
    - 256비트 키는 14 라운드



AES의 개요

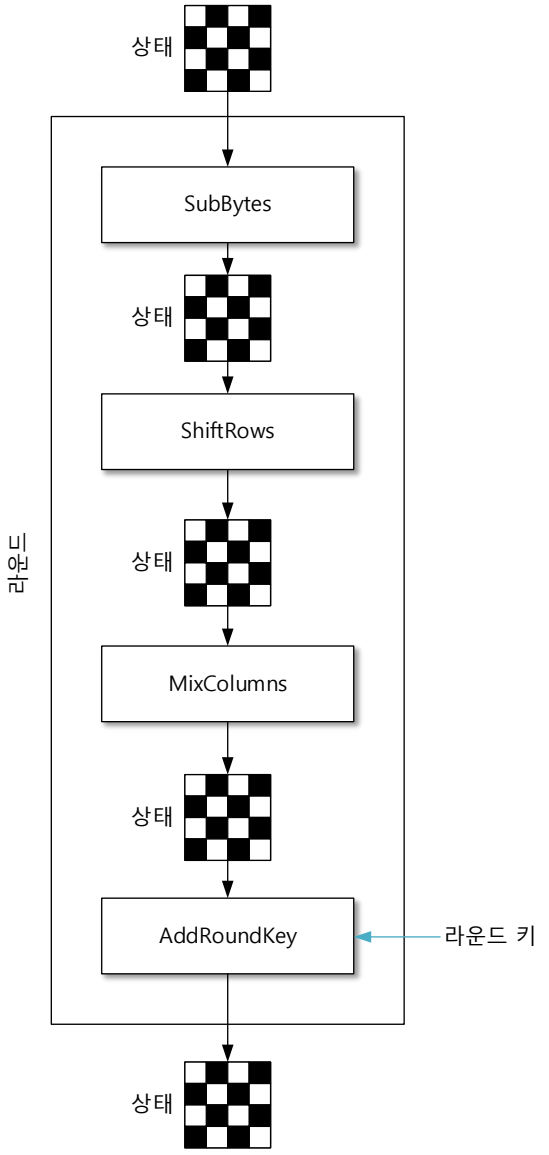
- 데이터 단위
  - 비트(bit)
    - 0 또는 1의 2진 데이터 값으로 다른 데이터 단위들을 이루는 가장 작은 요소
  - 바이트(byte)
    - 8개의 비트로 이루어져 각 비트들을 원소로 하는 행 행렬 또는 열 행렬로 표현
  - 워드(word)
    - 32비트로 이루어지고 각 바이트를 원소로 하는 행 행렬 또는 열 행렬로 표현
  - 블록(block)
    - 128비트로 구성되어 16바이트를 원소로 하는 행렬로 표현

AES의 개요

- 데이터 단위
  - 상태(state)
    - 암호 알고리즘의 시작과 끝의 값을 데이터 블록
    - 각 단계 전후에 있는 데이터 블록은 상태라고 정의
    - 16개의 바이트로 구성되어 4×4 행렬로 나타냄

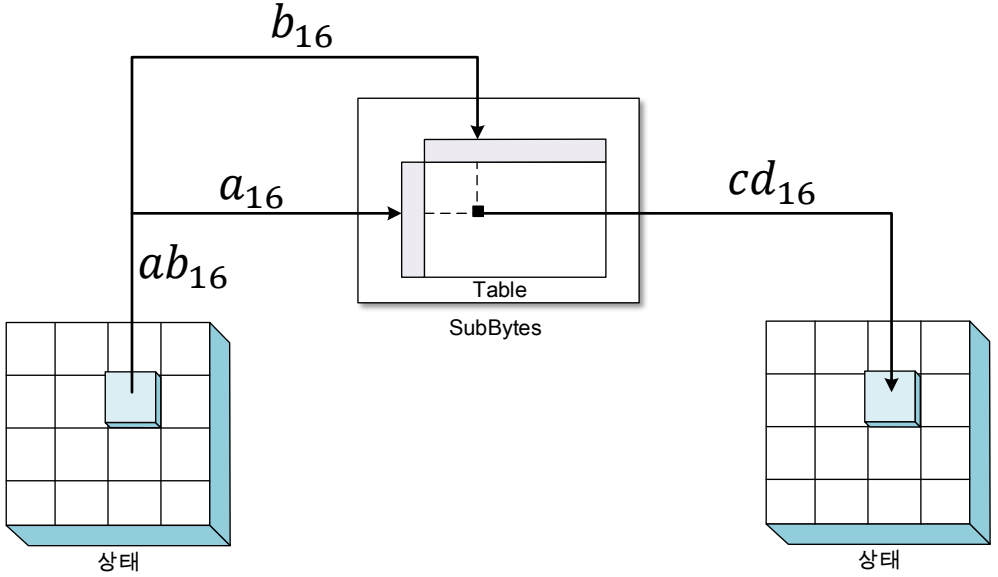
AES의 개요

- 라운드 함수의 구조
  - Feistel 구조가 아닌 SPN(Substitution-Permutation Network) 구조
    - 각 변환은 하나의 상태를 입력으로 하여 다음의 변환 혹은 다음의 라운드에서 사용될 또 다른 상태를 생성



변환

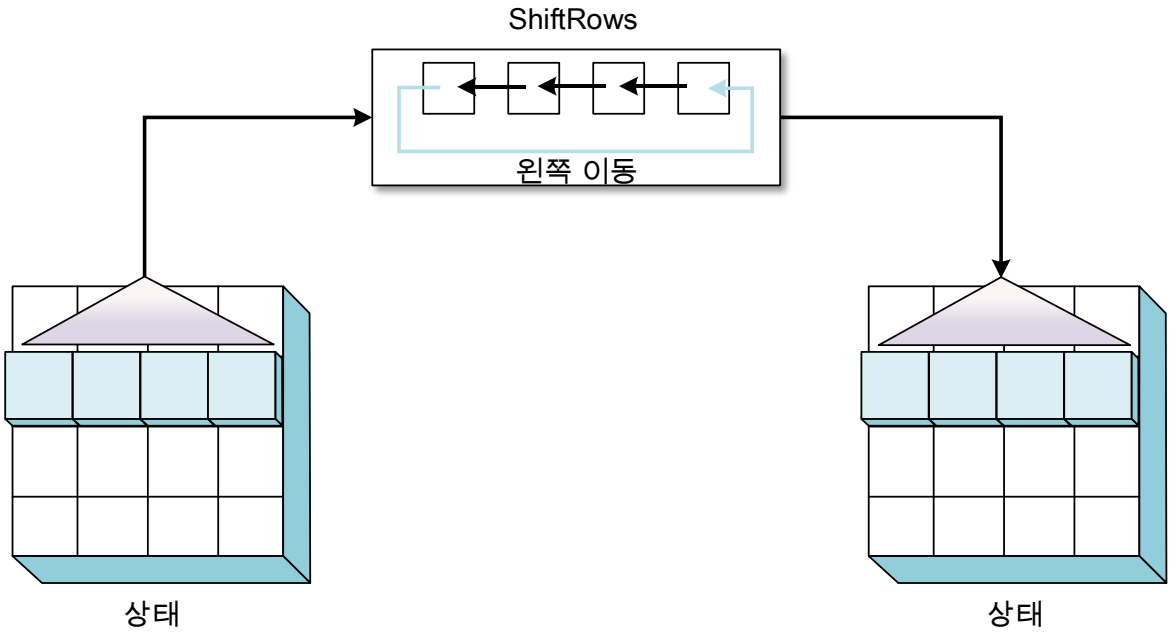
- 대치
  - 부분바이트(SubBytes)
    - AES 암호화 과정에서 사용되는 대치 함수
    - 각 바이트를 4비트씩 2개의 16진수로 계산
    - 왼쪽 4비트를 S-박스 행으로 오른쪽 4비트를 열로 표를 읽음



  - 역서브바이트(InvSubBytes)
    - 부분바이트의 역변환

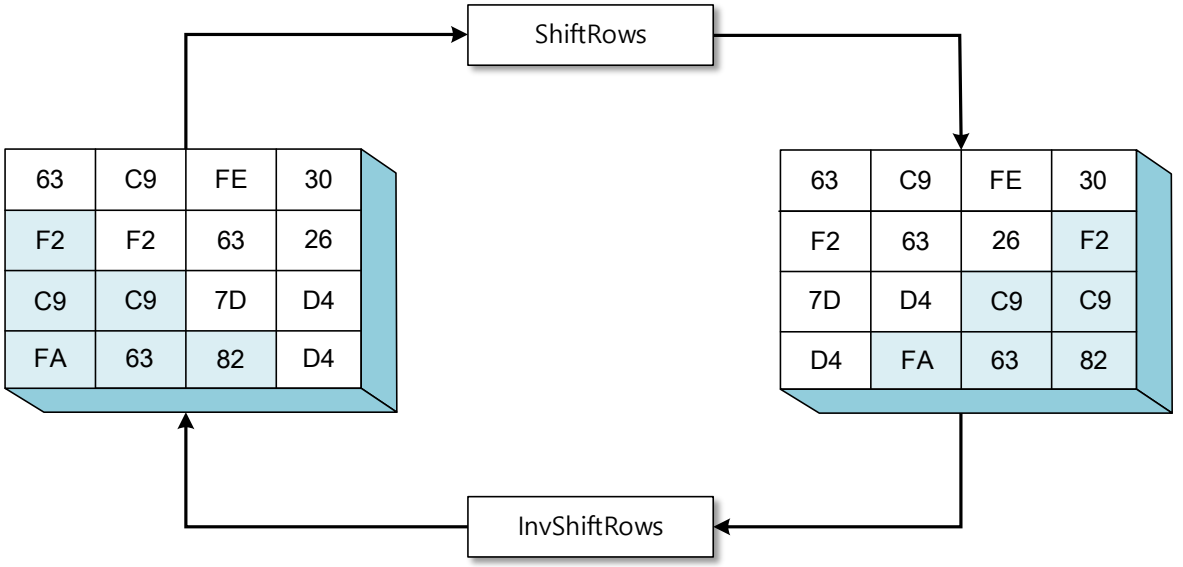
변환

- 치환
  - 이동 변환(shifting)
    - ShiftRows
      - 암호화 과정에서 사용하고 왼쪽으로 이동 수행
      - 이동하는 수는 상태 행렬의 행 번호(0~3)에 의존
      - 0번째 행에서는 이동하지 않고, 마지막 행에서는 3바이트 만큼 순환



변환

- 치환
  - 이동 변환(shifting)
    - InvShiftRows
      - 복호화 과정에서 사용하고 오른쪽으로 이동 수행
      - 이동 수는 상태 행렬의 행 번호(0~3)에 의존
    - ShiftRows와 InvShiftRows는 역변환 관계

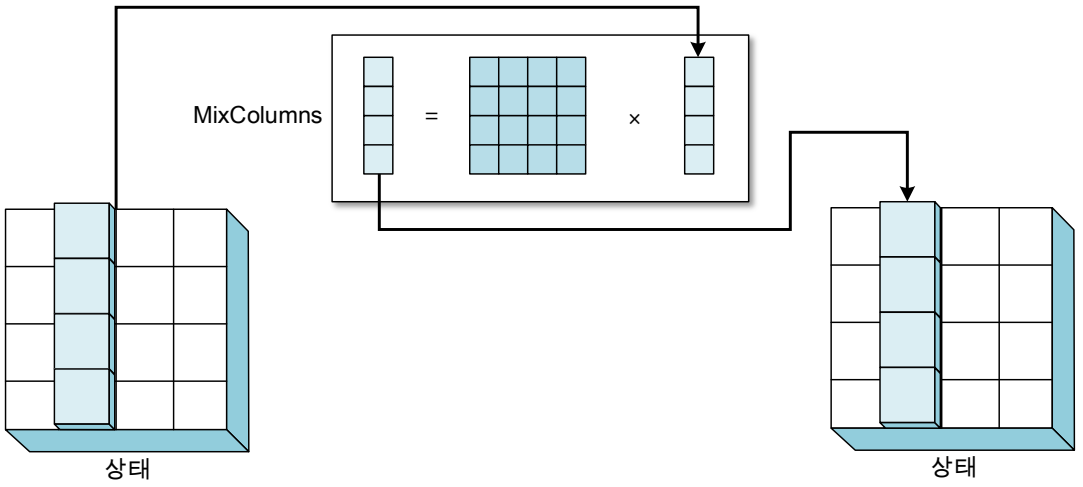


변환

- 뒤섞음
  - SubBytes는 바이트 값을 바꿈
  - ShiftRows는 바이트를 교환
  - 인접한 바이트들 안에 있는 비트들에 기반하여 비트 값을 바꾸는 바이트 내부 변환 필요
  - 과정
    - 한 번에 네 개의 바이트들을 가지고 각 바이트의 비트들을 바꾸어서 네 개의 새로운 바이트를 생성
    - 각 바이트에 서로 다른 상수 값을 곱해서 그것들을 뒤섞는 과정 수행

변환

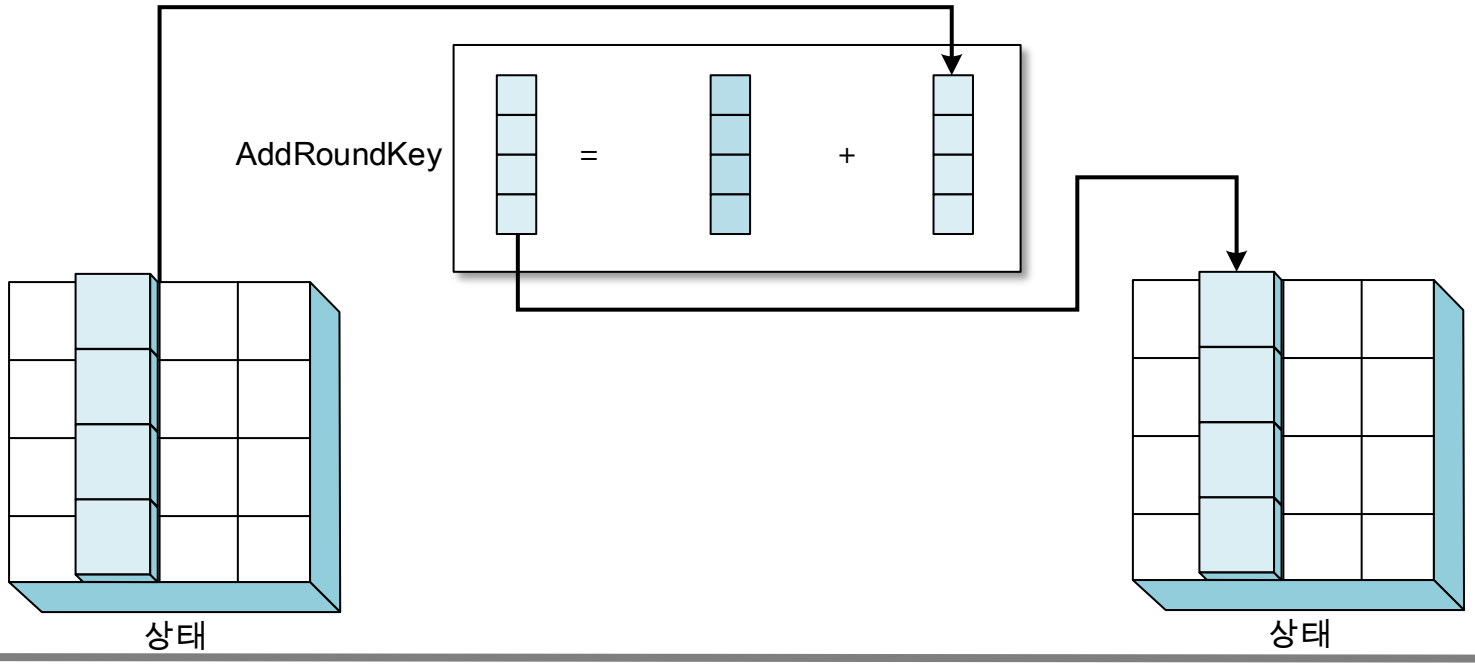
- 뒤섞음
  - MixColumns
    - 각 상태의 열을 새로운 열로 변환
    - 행렬과 각각의 열들을 곱해주는 연산



  - InvMixColumns
    - MixColumns와 역행렬 관계

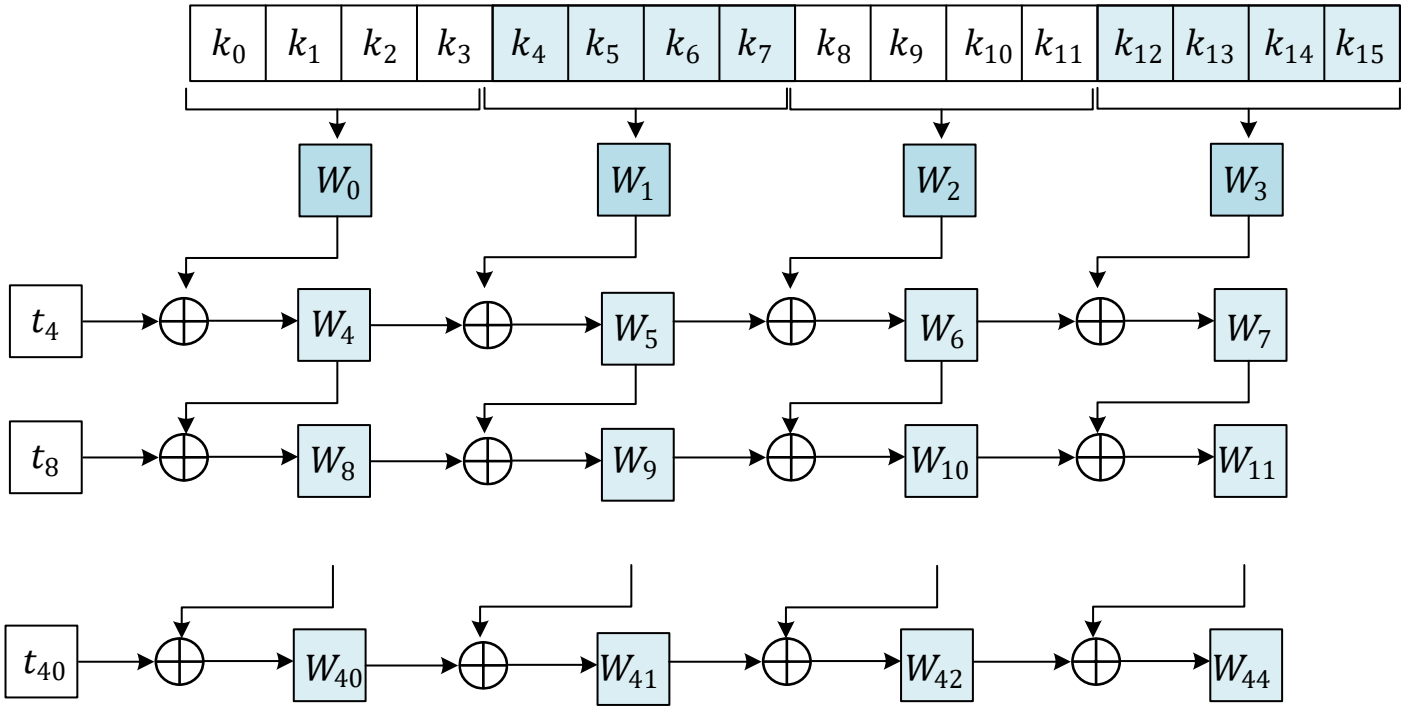
변환

- 키 덧셈
  - 상태에 키를 더하는 과정을 통해 공격자가 암호문으로부터 평문을 쉽게 찾는 것을 방지
  - AddRoundKey
    - 한 번에 한 열씩 덧셈을 이용하여 수행



변환

- 키 확장
  - 각 라운드에 사용하는 라운드 키를 생성하기 위해 키 확장
  - 10라운드로 이루어진 AES-128은 44워드 필요
  - 12라운드로 이루어진 AES-192는 52워드 필요
  - 14라운드로 이루어진 AES-256은 60워드 필요



변환

- 키 확장
  - AES-128의 키 확장
    1. 처음 네 개의 워드 W_0, W_1, W_2, W_3은 암호 키로부터 만듦. 암호키는 16개의 바이트로 k_0부터 k_{15}까지 구성된 배열이며, 4개씩 묶어 네 개의 바이트를 생성
    2. 나머지 워드들의 계산

       if(i mod 4 = 0) W_i=t$\oplus W_{i-4}$

       else $W_i = W_{i-1} \oplus W_{i-4}$

       t = SubWord(RotWord(W_{i-1}))$\oplus Rcon_{i/4}$

변환

- 키 확장
  - AES-128의 키 확장
    - RotWord
      - 하나의 열에만 적용되어 워드를 순환 이동
      - 4개의 바이트로 구성된 하나의 워드를 입력으로 왼쪽으로 한 바이트 씩 이동
    - SubWord
      - 4 개의 바이트에 적용되어 워드의 각 바이트를 S-박스를 이용하여 대치한 후 출력
    - RCon
      - 4바이트의 값으로, 가장 오른쪽의 3바이트는 모두 0

라운드	상수	라운드	상수
1	(01 00 00 00)	6	(20 00 00 00)
2	(02 00 00 00)	7	(40 00 00 00)
3	(04 00 00 00)	8	(80 00 00 00)
4	(08 00 00 00)	9	(1B 00 00 00)
5	(10 00 00 00)	10	(36 00 00 00)

변환

- 키 확장
  - AES-192와 AES-256의 키 확장
    - AES-192
      - 6개의 워드 단위로 계산
      1. 암호키로부터 6개의 워드 $W_0, W_1, W_2, W_3, W_4, W_5$ 생성
      2. 나머지 워드들의 계산

         if(i mod 4 = 0) W_i=t$\oplus W_{i-6}$

         else $W_i = W_{i-1} \oplus W_{i-6}$
    - AES-256
      - 8개의 워드 단위로 계산
      1. 암호키로부터 8개의 워드 $W_0, W_1, W_2, W_3, W_4, W_5, W_6, W_7$ 생성
      2. 나머지 워드들의 계산

         if(i mod 4 = 0) W_i=t$\oplus W_{i-8}$

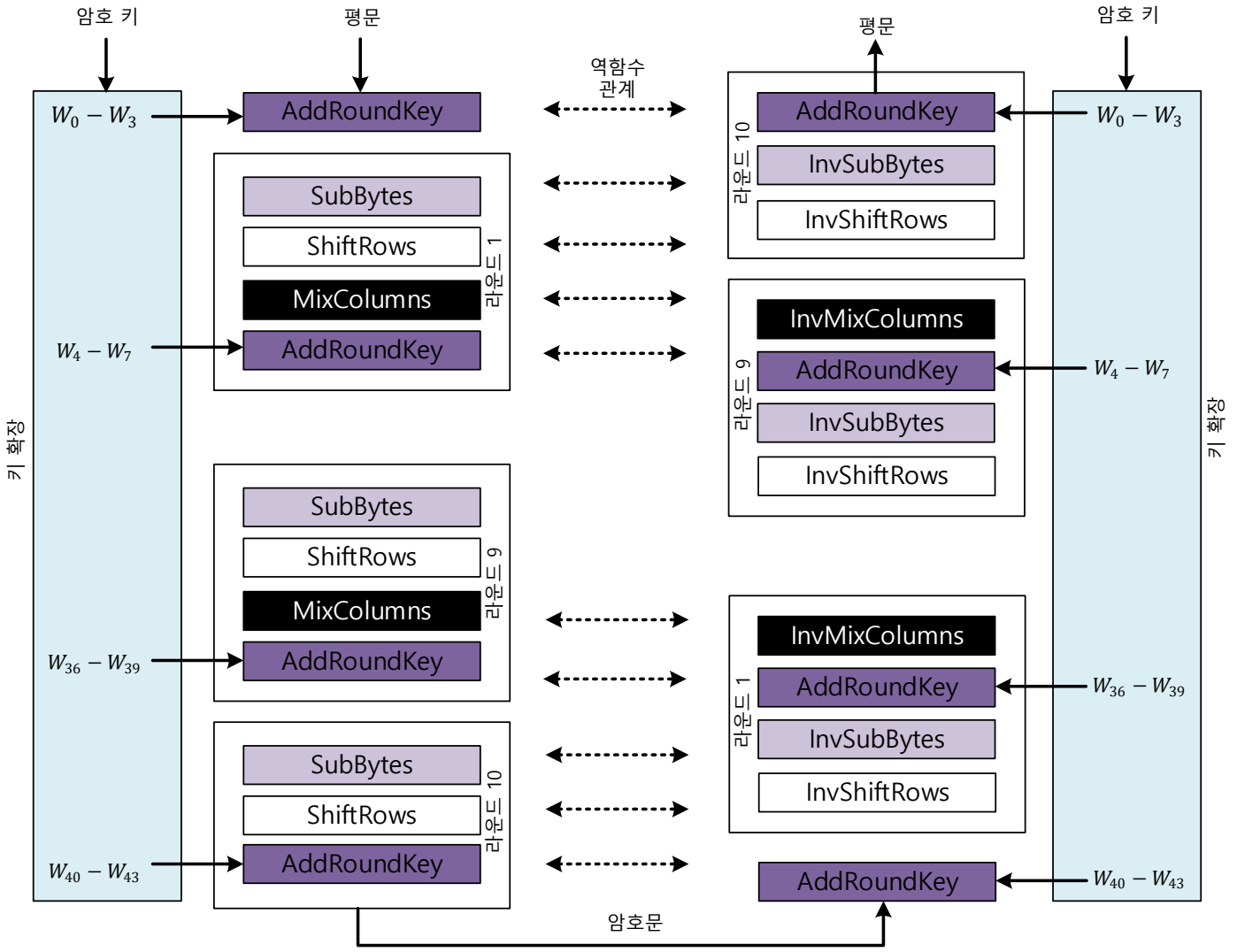
         else $W_i = W_{i-1} \oplus W_{i-8}$

변환

- 키 확장
  - AES 과정 분석
    1. 공격자가 암호 키의 일부나 라운드 키의 일부 값을 얻는다 해도 모든 라운드 키를 알기 위해선 남은 암호 키 값을 전부 다 복구해야 함
    2. 두 개의 암호 키가 오직 한 비트만 다른 값을 갖더라도 라운드 키 확장 과정을 통해 적은 수의 라운드 만에 서로 다른 라운드 키 생성
    3. 키 확장 과정에서 사용하는 상수 Rcons는 다른 변환 과정에서 발생할 수 있는 라운드 키들 사이의 대칭성 제거

암호

- 기본 설계

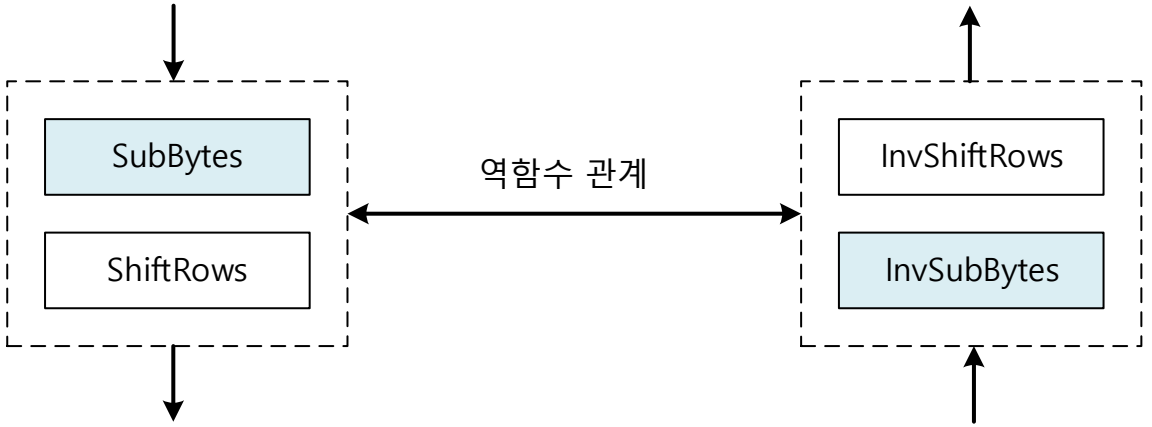

암호 키
평문
역함수 관계
평문
암호 키
$W_0 - W_3$
AddRoundKey
AddRoundKey
$W_0 - W_3$
라운드 10
InvSubBytes
InvShiftRows
SubBytes
ShiftRows
MixColumns
AddRoundKey
라운드 1
InvMixColumns
AddRoundKey
InvSubBytes
InvShiftRows
라운드 9
$W_4 - W_7$
$W_4 - W_7$
키 확장
키 확장
SubBytes
ShiftRows
MixColumns
AddRoundKey
라운드 9
InvMixColumns
AddRoundKey
InvSubBytes
InvShiftRows
라운드 1
$W_{36} - W_{39}$
$W_{36} - W_{39}$
SubBytes
ShiftRows
AddRoundKey
라운드 10
AddRoundKey
$W_{40} - W_{43}$
$W_{40} - W_{43}$
암호문

암호

- 기본 설계
  - 복호 알고리즘에서 SubBytes와 ShiftRows의 순서가 바뀜
  - 복호 알고리즘에서 MixColumns와 AddRoundKey의 순서가 바뀜
  - 복호 알고리즘은 암호 알고리즘의 역함수
  - 복호 알고리즘에선 암호 알고리즘의 라운드 키를 역순으로 사용

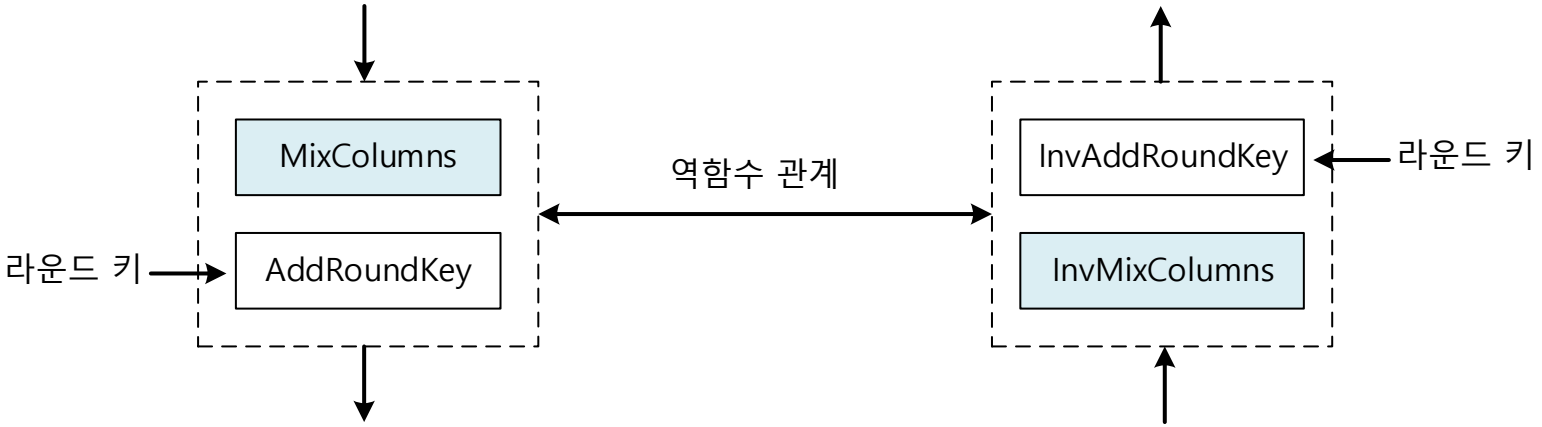
암호

- 대체 설계
  - 역암호에서 사용되는 변환이 암호에 사용되는 변환과 같은 순서로 동작하기 위해 재정렬됨
  - SubBytes/ShiftRows 쌍
    - SubBytes는 바이트의 순서를 변경하지 않고 바이트 값 변경
    - ShiftRows는 바이트 값을 변경하지 않고 순서만 변경
    - 위의 두 함수의 순서 변환 시에도 가역성 유지 가능



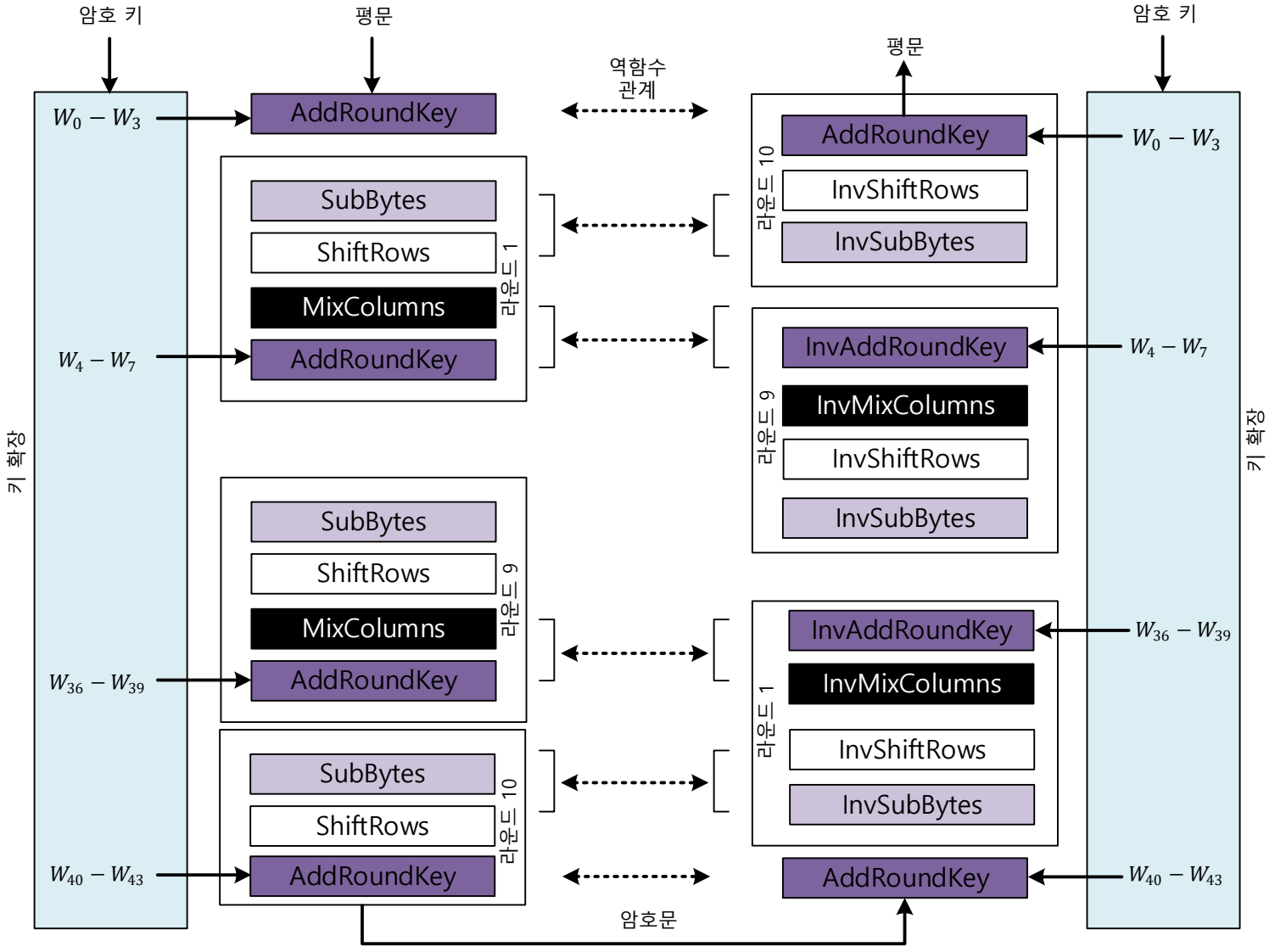
암호

- 대체 설계
  - MixColumns/AddRoundKey 쌍
    - MixColumns에서 사용된 상수 행렬의 역행렬을 키 행렬에 곱하면, MixColumns와 AddRoundKey는 서로 역함수 관계



암호

- 대체 설계


암호 키
평문
역함수 관계
평문
암호 키
$W_0 - W_3$
AddRoundKey
AddRoundKey
$W_0 - W_3$
SubBytes
ShiftRows
MixColumns
AddRoundKey
라운드 1
라운드 10
InvShiftRows
InvSubBytes
$W_4 - W_7$
InvAddRoundKey
$W_4 - W_7$
라운드 9
InvMixColumns
InvShiftRows
InvSubBytes
키 확장
키 확장
SubBytes
ShiftRows
MixColumns
AddRoundKey
라운드 9
$W_{36} - W_{39}$
InvAddRoundKey
$W_{36} - W_{39}$
라운드 1
InvMixColumns
InvShiftRows
InvSubBytes
SubBytes
ShiftRows
AddRoundKey
라운드 10
$W_{40} - W_{43}$
AddRoundKey
$W_{40} - W_{43}$
암호문

AES의 분석

- 안전성
  - 전수조사 공격
    - 긴 길이의 키를 사용하기 때문에 DES보다 더 안전
    - 2^{128}번의 테스트가 필요
    - 키를 찾는데 $2^{72} \times t$초 필요
  - 통계적인 공격
    - SubBytes, ShiftRows, MixColumns의 조합은 평문에서 나타나는 빈도수 패턴 제거

Thanks!

이 하 늘(haneul@pel.sejong.ac.kr)