2017/07/08, 2017 보안 기초 세미나

Network Security Essentials

- 2장 대칭 암호와 메시지 기밀성 (2) -

최창준 (changjun@pel.smuc.ac.kr)

상명대학교 프로토콜공학연구실

목 차

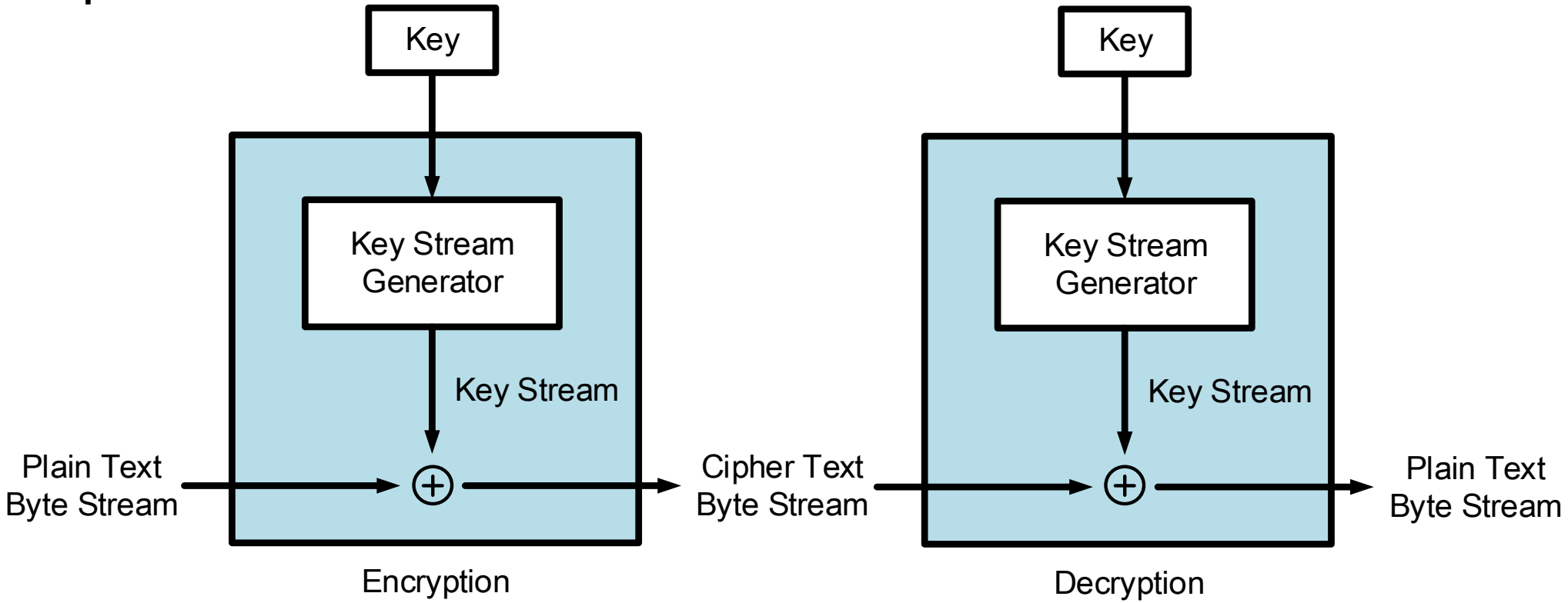
- 스트림 암호
  - RC4
- 암호 블록 운용 모드
  - ECB (Electronic CodeBook) 모드
  - CBC (Cipher Block Chaining) 모드
  - CFB (Cipher FeedBack) 모드
  - OFB (Output FeedBack) 모드
  - CTR (CounTeR) 모드

스트림 암호

- 스트림 암호(Stream cipher)
  - 비트나 바이트 단위로 입력되는 요소를 연속적으로 처리하는 대칭키 암호 알고리즘
  - 키 스트림 생성기(Key stream generator) 사용
    - 키 스트림 생성기는 의사 난수 생성기(PRNG) 이용
    - 원래의 키를 입력 받아 랜덤한 8bits 키 스트림을 생성
    - 한 번에 한 바이트씩 평문, 암호문과 XOR연산하여 암·복호화
  - 전수 조사 공격에 대응하기 위해 최소 128bits 이상의 키 길이를 가져야 함

스트림 암호

- 구조



- e.g.,

11001100	평문	
⊕ 01101100	키 스트림	
10100000	암호문	

10100000	암호문	
⊕ 01101100	키 스트림	
11001100	평문	

스트림 암호

- 장점
  - 패딩이 필요하지 않음
    - 패딩 : 부족한 길이만큼 비트를 ‘0’으로 채워 넣는 것
  - 속도가 빠름
  - 실시간으로 사용할 수 있음
    - e.g., 실시간 오디오/비디오 스트리밍 등
- 단점
  - 서로 다른 두 개의 평문을 동일한 키 스트림으로 암호화 할 시 암호 해독이 단순해짐
    - 두 개의 평문 XOR = 두 개의 암호문 XOR

스트림 암호

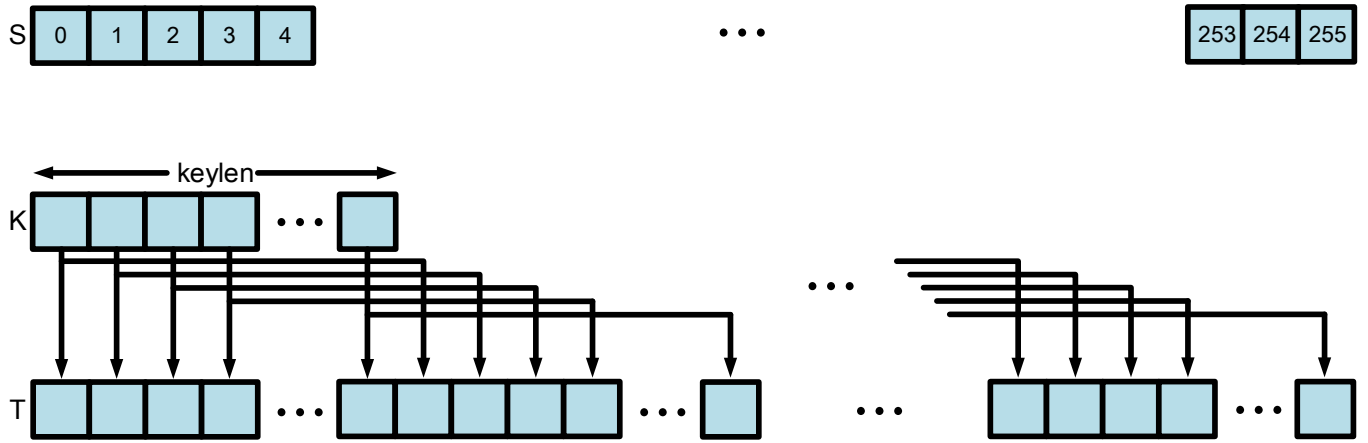
- RC4(Rivest Cipher 4)
  - 1984년 Ronald Rivest에 의해 설계된 바이트 단위의 스트림 암호 알고리즘
  - 벡터 S 사용
    - 256 bytes로 구성됨
    - S의 1byte는 암호화 키로 사용되기 위해 무작위로 선택됨
  - 랜덤 치환 기법 사용
  - 전체 동작 과정
    1. S와 T의 초기상태 초기화
    2. S의 초기 치환
    3. 키 스트림 생성

스트림 암호

1. S와 T의 초기 상태 초기화(1/2)

- 벡터 S의 초기화
  - 키를 입력으로 받아 키 배열을 생성하고 S의 값을 0에서 255까지 오름차순으로 초기화

```
for i = 0 to 255 do
  S[ i ] = i;
  T[ i ] = K[ i mod keylen ];
```

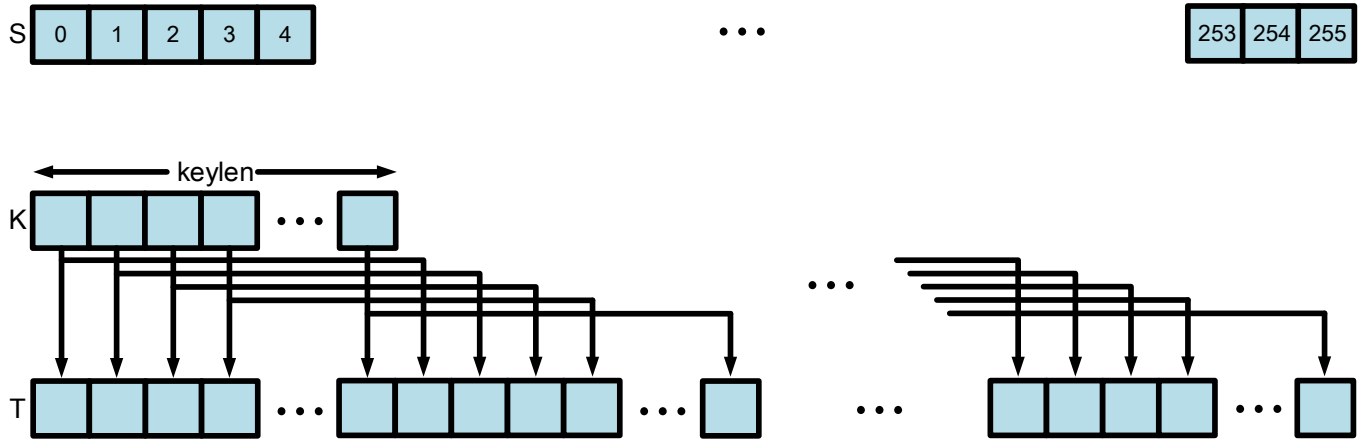


스트림 암호

1. S와 T의 초기 상태 초기화(2/2)

- 임시벡터 T 생성
  - 키 길이가 256바이트인 경우
    - 키를 그대로 임시벡터 T에 저장
  - 키 길이가 256바이트 미만인 경우
    - 키 길이만큼 T에 저장, T가 채워질 때까지 키를 반복해서 복사

```
for i = 0 to 255 do
  S[ i ] = i;
  T[ i ] = K[ i mod keylen ];
```

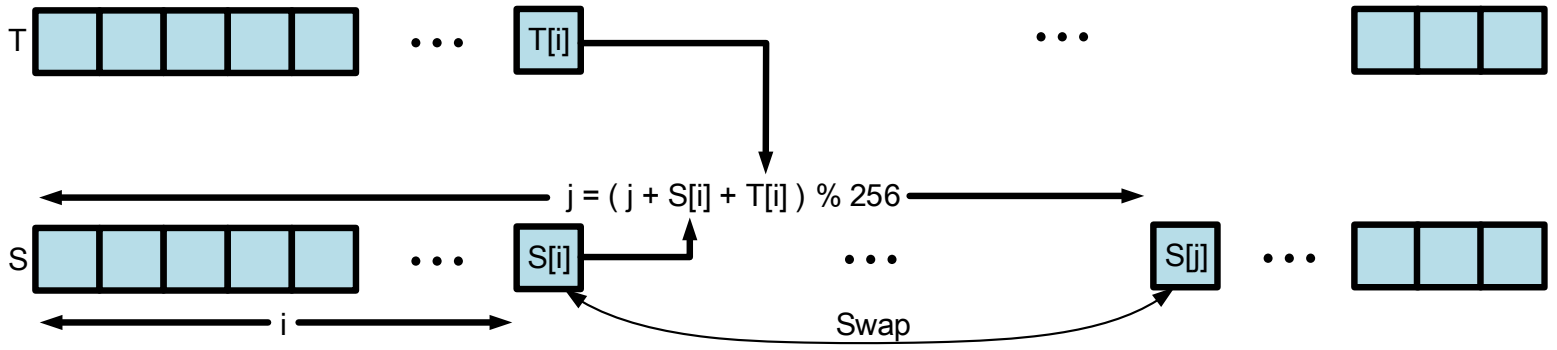


스트림 암호

2. S의 초기 치환

- 벡터 T와 S[i]를 이용하여 치환할 인덱스 j를 계산
- 벡터 S의 값을 섞어주기 위해 S[i]와 S[j]의 위치 교환

```
j = 0;
for i = 0 to 255 do
   j = ( j + S[ i ] + T[ i ] ) mod 256;
   Swap( S[ i ], S[ j ] );
```



스트림 암호

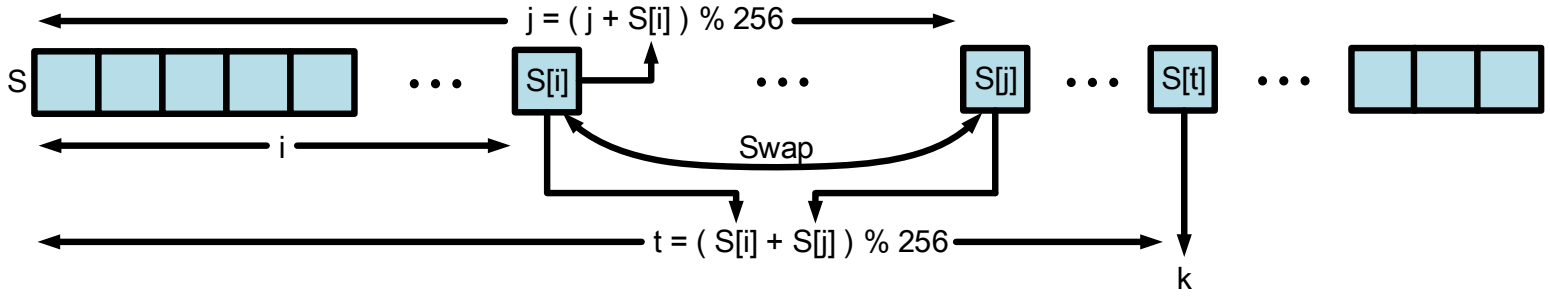
3. 키 스트림 생성

- 각 S[i]는 현재 상태 값에 따라 값이 교환됨
- S[i]와 S[j]를 이용하여 인덱스 t를 계산
- S[t]에 있는 값을 키 스트림으로 생성
- S[255]까지 도달하면 S[0]에서 처음부터 과정 반복

```
i, j = 0;
while (true)
   i = ( i + 1) mod 256;
   j = ( j + S[ i ] ) mod 256;

   Swap( S[ i ], S[ j ] );
   t = ( S[ i ] + S[ j ] ) mod 256;
   k = S[ t ];

   return plaintext ^ k;
```

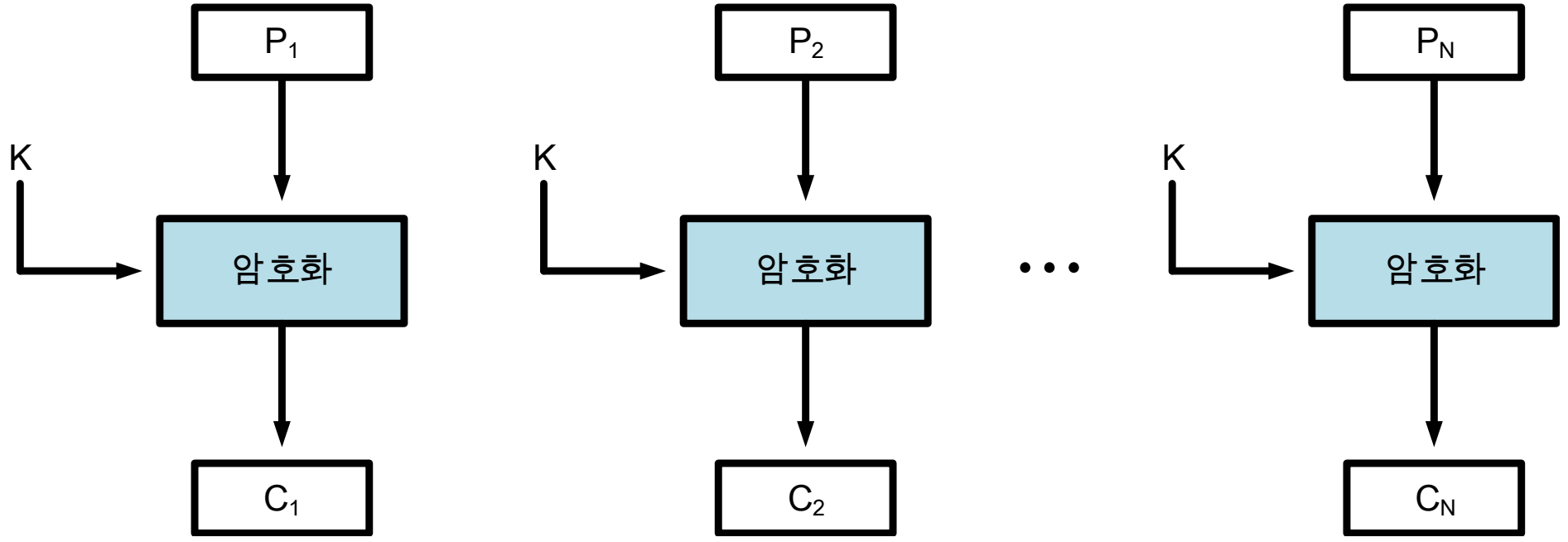


블록 암호 운용 모드

- 블록 암호 운용 모드
  - 하나의 키를 사용하여 블록 암호를 반복적으로 이용하는 암호화 방식
  - 길이가 가변적인 데이터를 암호화하는 방법
  - NIST에서 5가지 운용 모드를 정의
    - 전자 코드북(ECB, Electronic Codebook) 모드
    - 암호 블록 체인(CBC, Cipher Block Chaining) 모드
    - 암호 피드백(CFB, Cipher Feedback) 모드
    - 출력 피드백(OFB, Output Feedback) 모드
    - 카운터(CTR, Counter) 모드

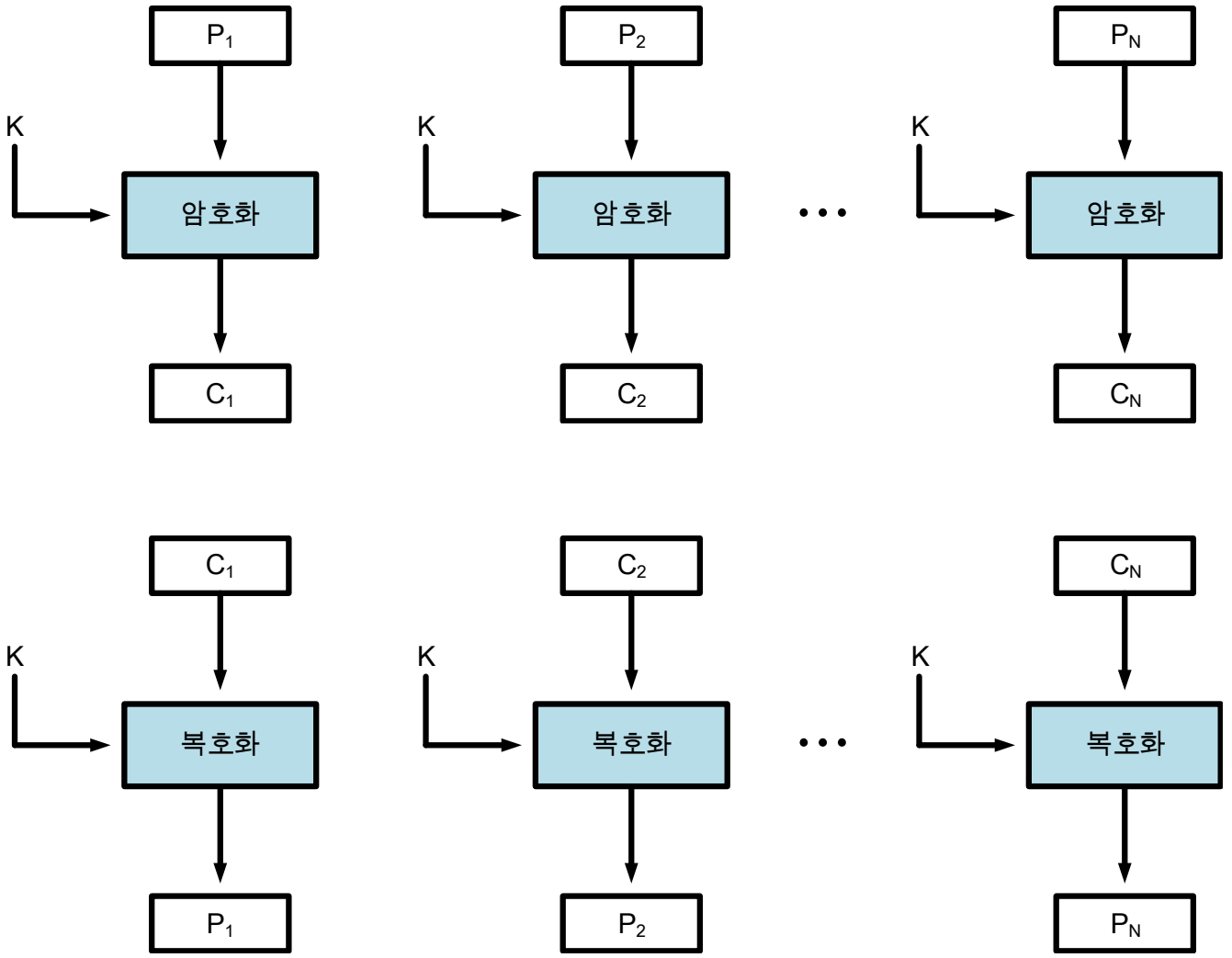
블록 암호 운용 모드

- 전자 코드북(ECB, Electronic Codebook) 모드
  - 평문을 일정한 블록으로 나누어 동일한 키로 암호화 하는 방식



블록 암호 운용 모드

- ECB
  - 구조
    - 암호화
    - 복호화

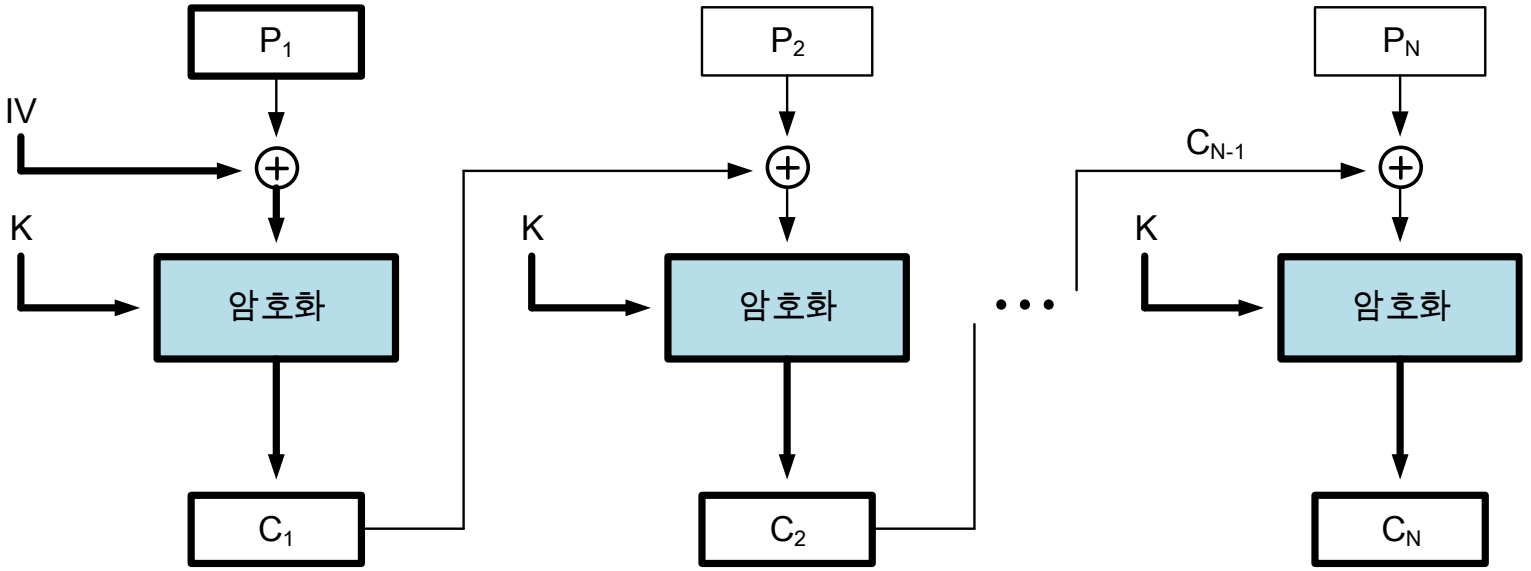


블록 암호 운용 모드

- ECB
  - 장점
    - 암·복호화 처리 속도가 빠름
    - 병렬 처리 가능
    - 오류가 확산되지 않음
  - 단점
    - 패딩이 필요함
    - 평문의 반복 패턴이 드러남
      - 패턴 공격에 취약

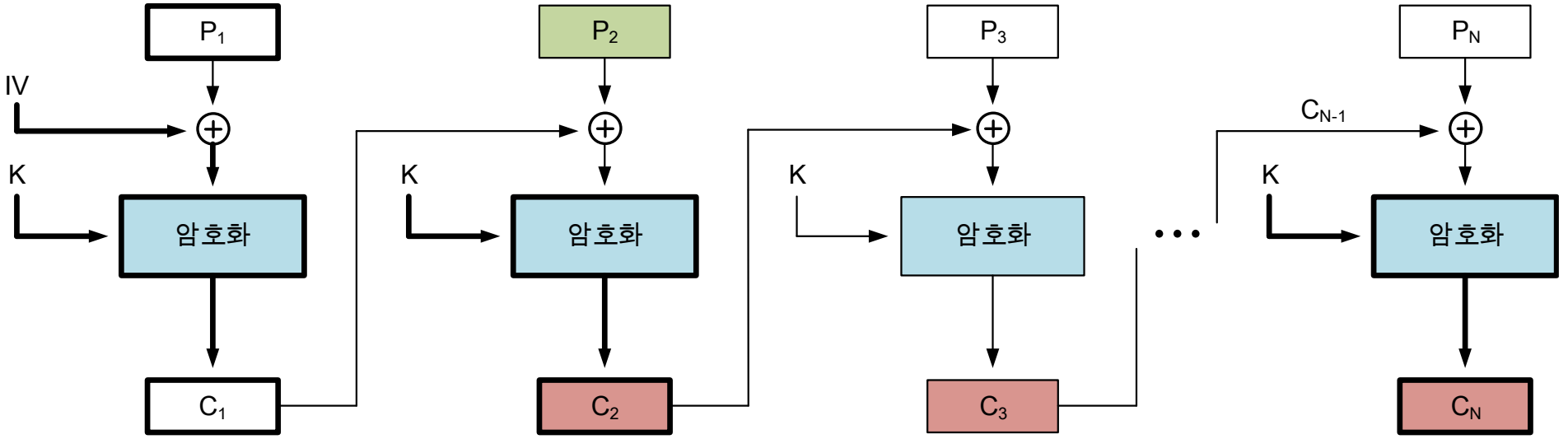
블록 암호 운용 모드

- 암호 블록 체인(CBC, Cipher Block Chaining) 모드
  - 체인 구조를 형성하여 각 암호문 블록이 이전 단계 암호문 블록의 영향을 받도록 만든 방식
  - 초기화 벡터(IV, Initial Vector) 사용



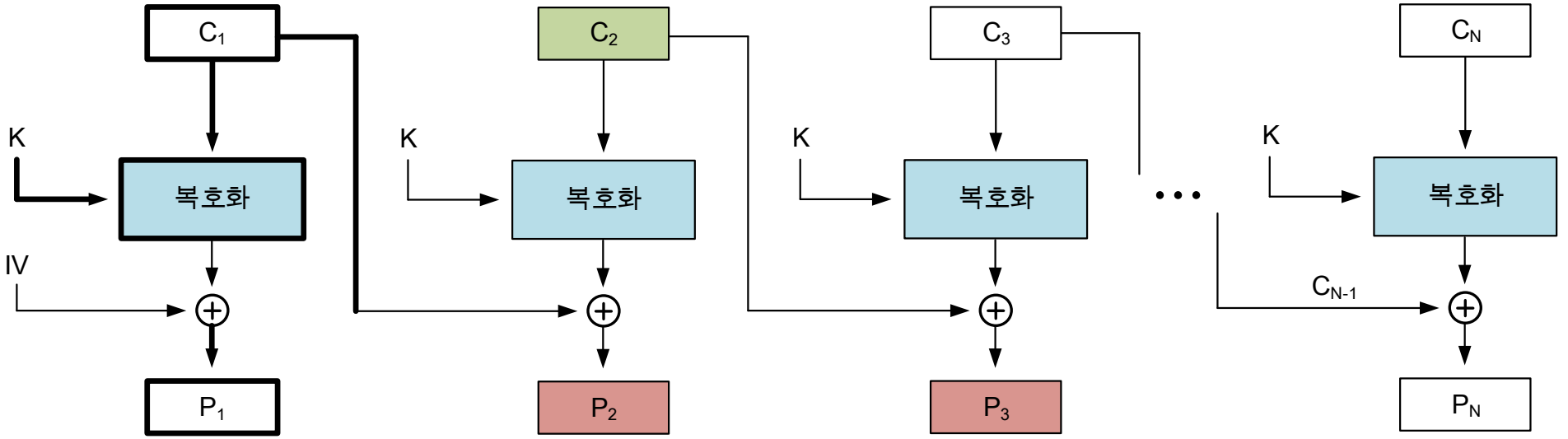
블록 암호 운용 모드

- CBC
  - 오류 확산(1/2)
    - 암호화 과정
      - 평문 블록 하나가 오류 났을 때, 현재 암호문 블록 이후 전체 암호문 블록에 영향



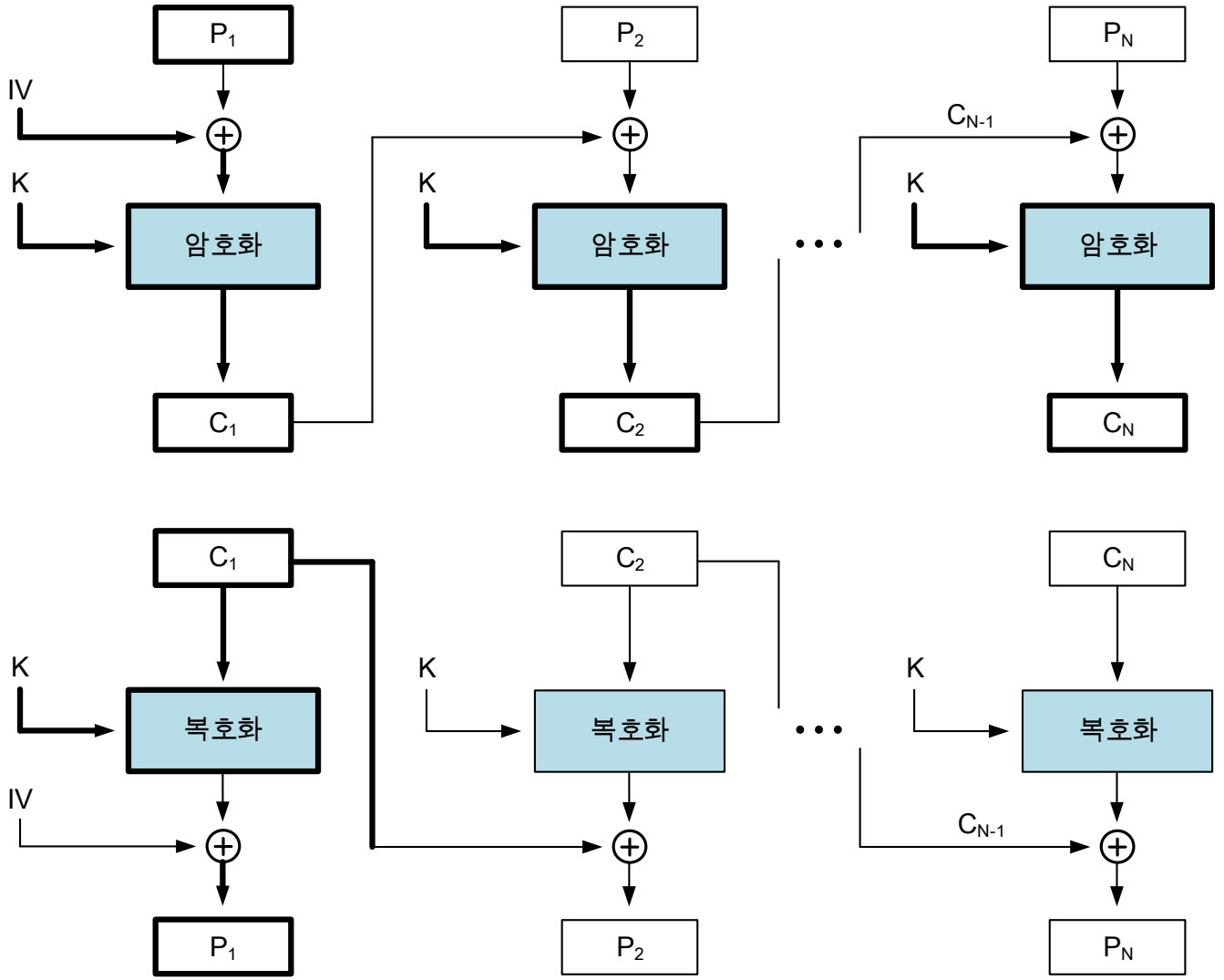
블록 암호 운용 모드

- CBC
  - 오류 확산(2/2)
    - 복호화 과정
      - 암호문 블록 하나가 오류 났을 때, 현재 평문 블록과 다음 평문 블록에만 영향



블록 암호 운용 모드

- CBC
  - 구조
    - 암호화
    - 복호화

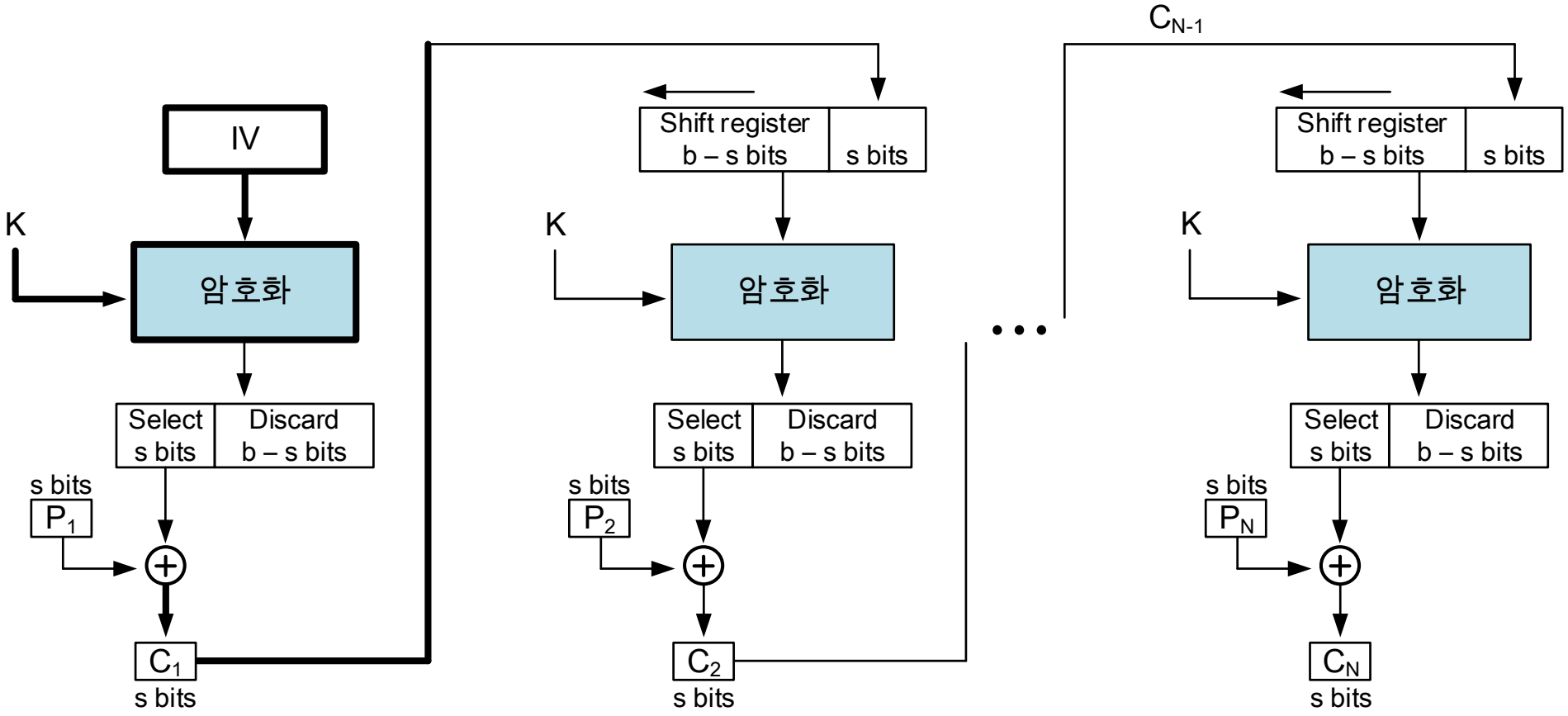


블록 암호 운용 모드

- CBC
  - 장점
    - 평문의 반복 패턴이 드러나지 않음
  - 단점
    - 오류가 확산됨
    - 암호화 과정은 병렬 처리 불가능
    - 패딩이 필요함

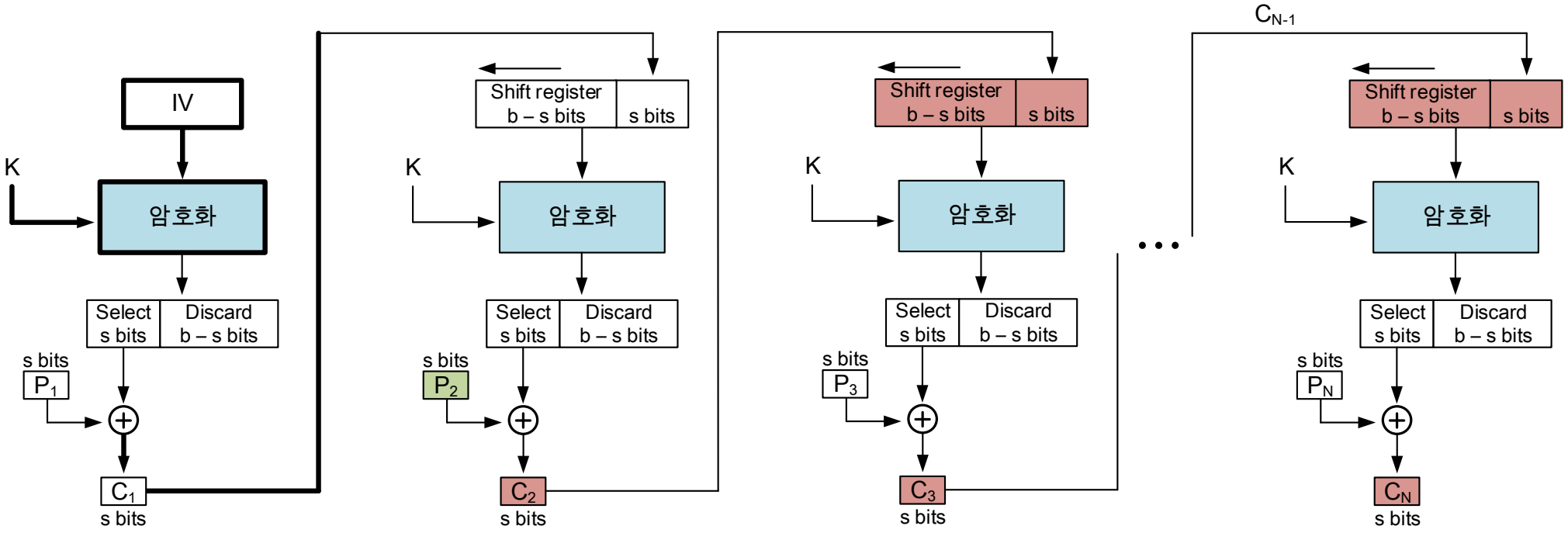
블록 암호 운용 모드

- 암호 피드백(CFB, Cipher Feedback) 모드
  - CBC의 변형으로, 블록 암호를 스트림 암호로 변환하여 암·복호화하는 방식



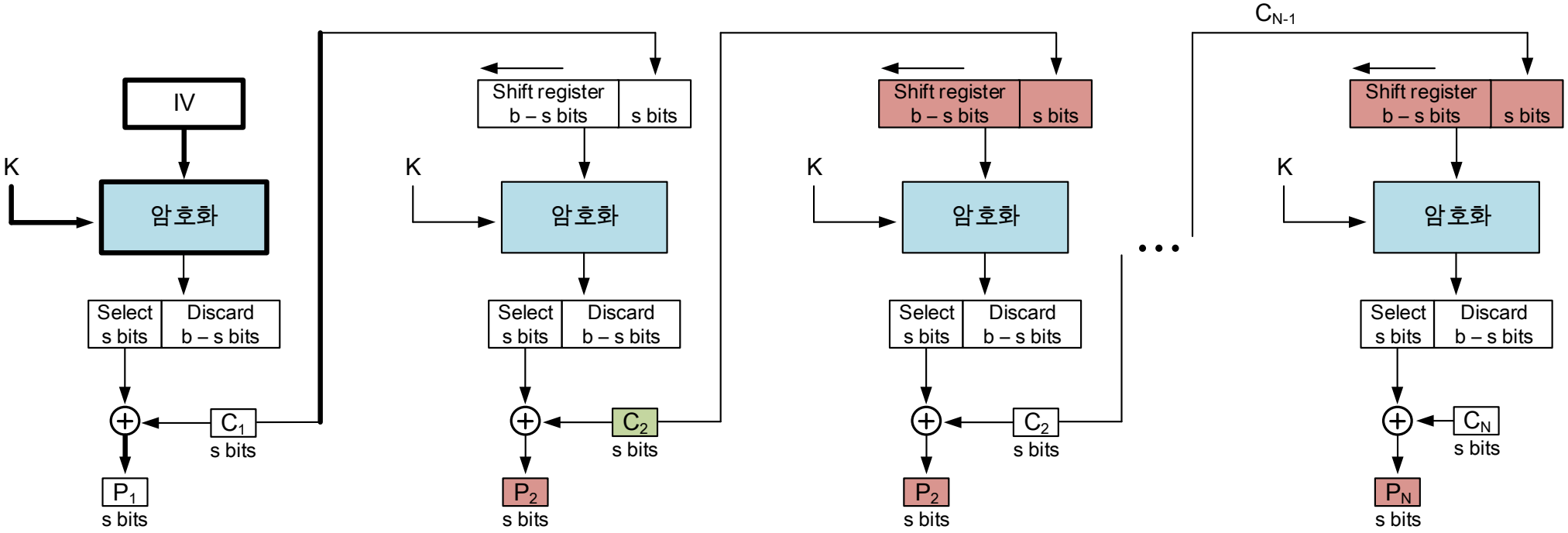
블록 암호 운용 모드

- CFB
  - 오류 확산(1/2)
    - 암호화 과정
      - 평문 블록 하나가 오류 났을 때, 현재 암호문 블록 이후 Shift register 에서 오류가 완전히 소멸될 때까지 암호문 블록에 영향



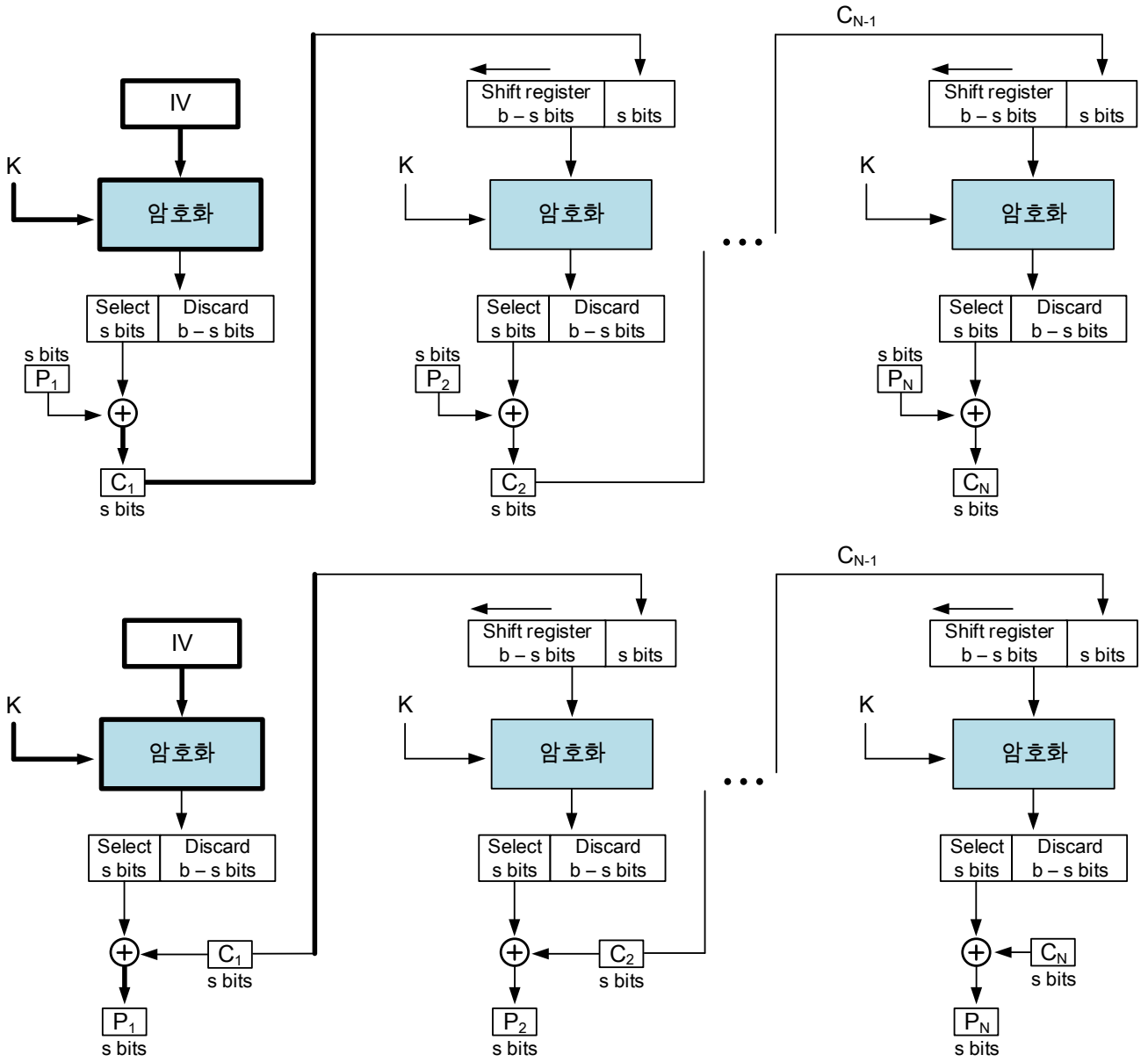
블록 암호 운용 모드

- CFB
  - 오류 확산(2/2)
    - 복호화 과정
      - 암호문 블록 하나가 오류 났을 때, 현재 평문 블록 이후 Shift register 에서 오류가 완전히 소멸될 때까지 평문 블록에 영향



블록 암호 운용 모드

- CFB
  - 구조
    - 암호화
    - 복호화

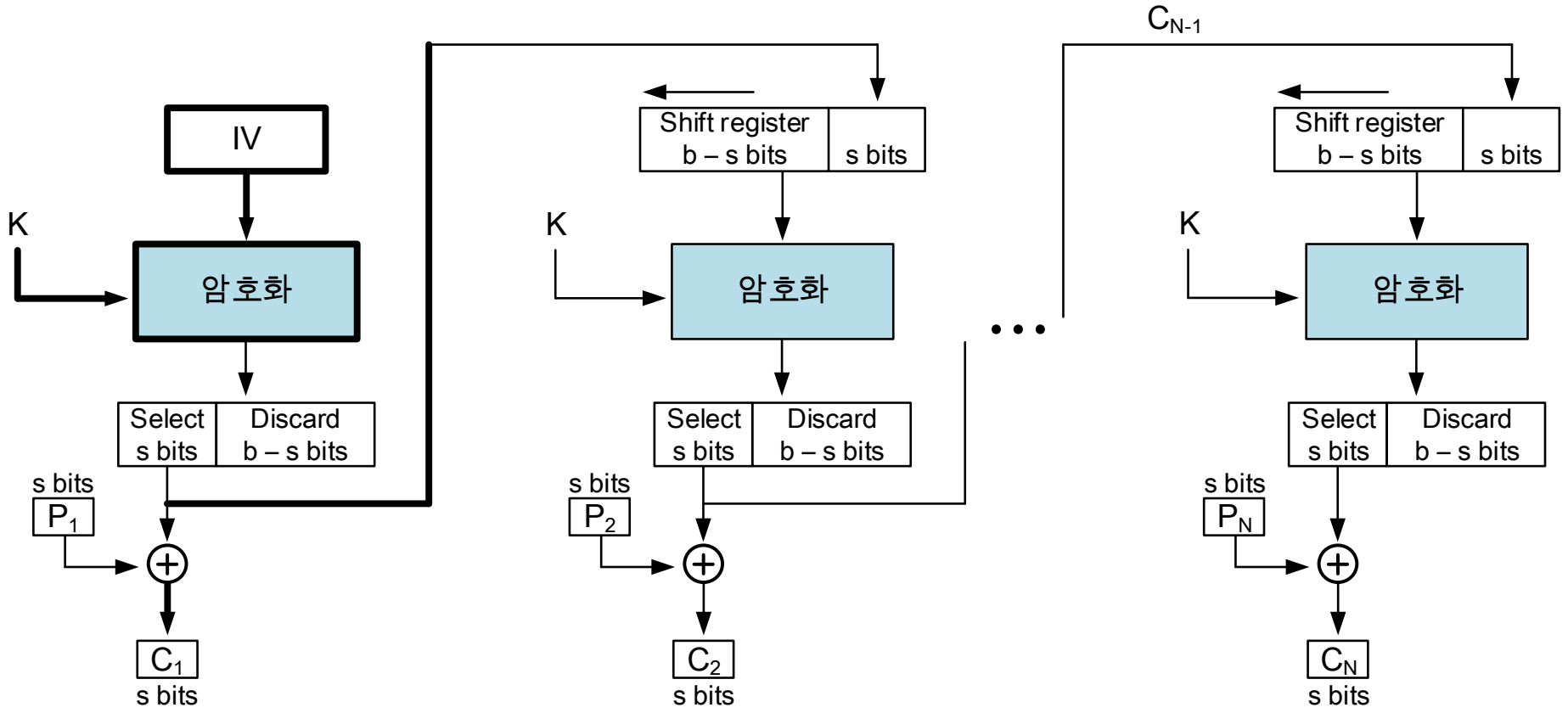


블록 암호 운용 모드

- CFB
  - 장점
    - 패딩이 필요하지 않음
  - 단점
    - 암호화 과정은 병렬 처리 불가능
    - 오류가 확산됨

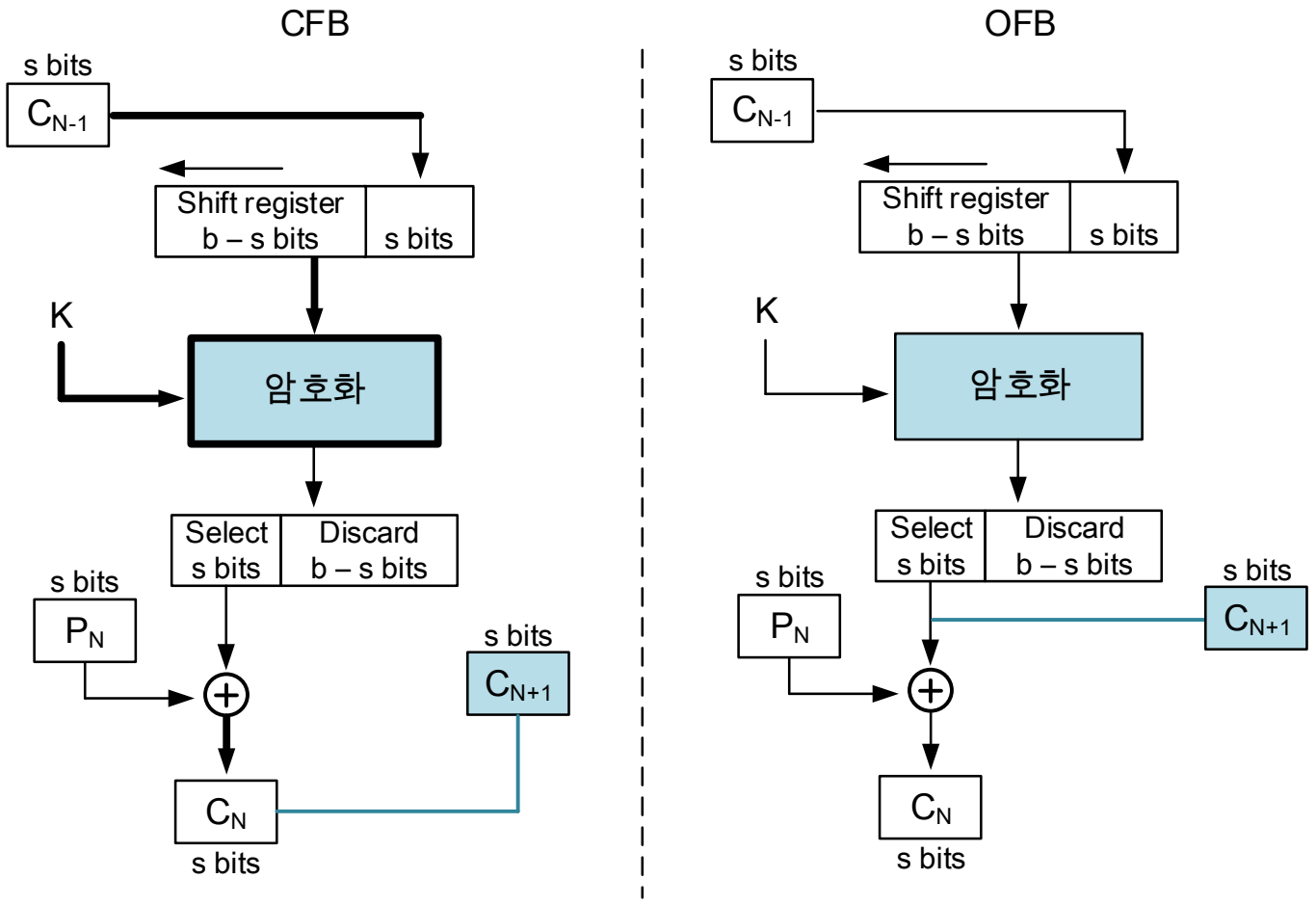
블록 암호 운용 모드

- 출력 피드백(OFB, Output Feedback) 모드
  - CFB의 변형으로, 각 암호문 블록이 이전 단계 암호문 블록들과 독립적인 방식



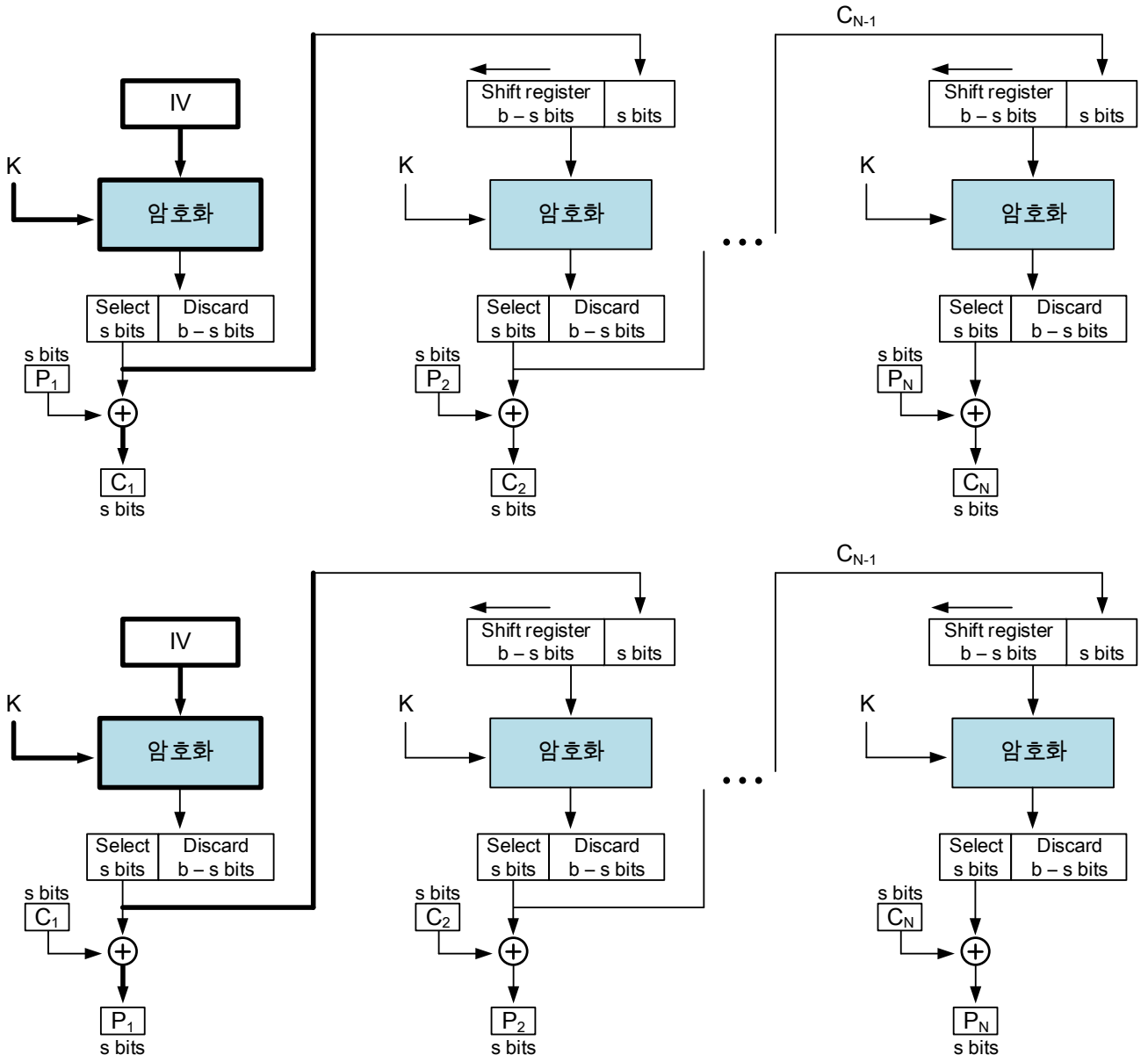
블록 암호 운용 모드

- OFB
  - CFB와 OFB의 비교



블록 암호 운용 모드

- OFB
  - 구조
    - 암호화
    - 복호화

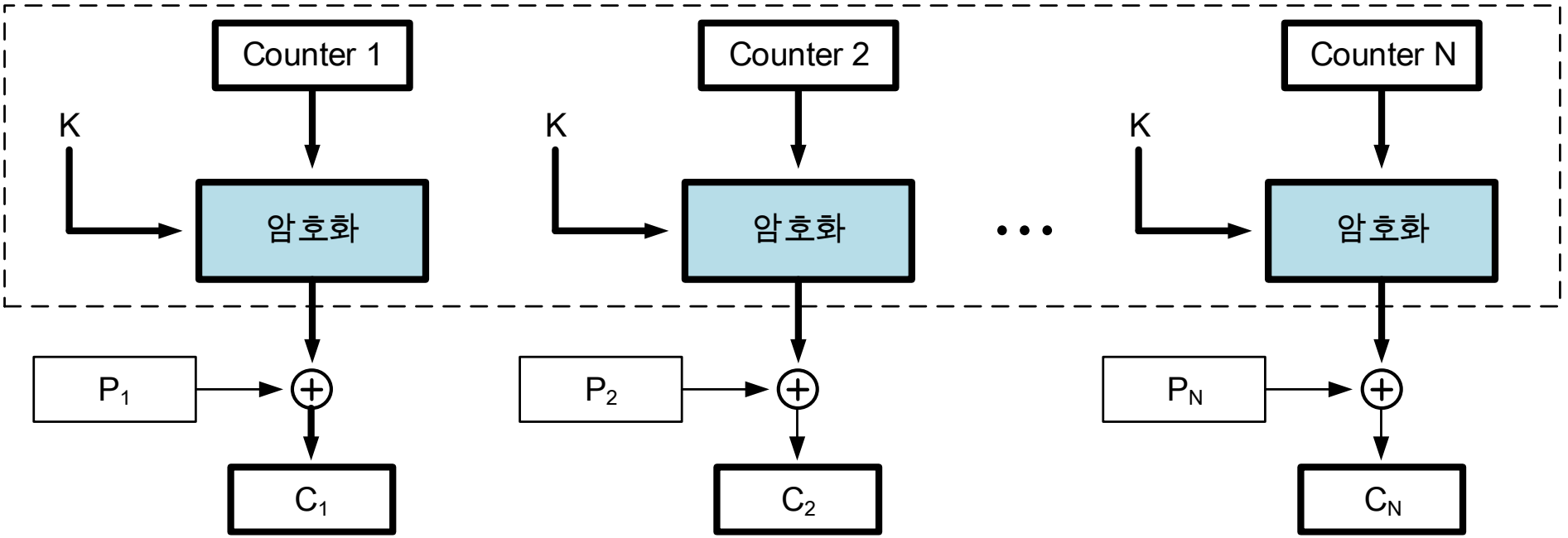


블록 암호 운용 모드

- OFB
  - 장점
    - 오류가 확산되지 않음
    - 패딩이 필요하지 않음
  - 단점
    - 병렬 처리 불가능

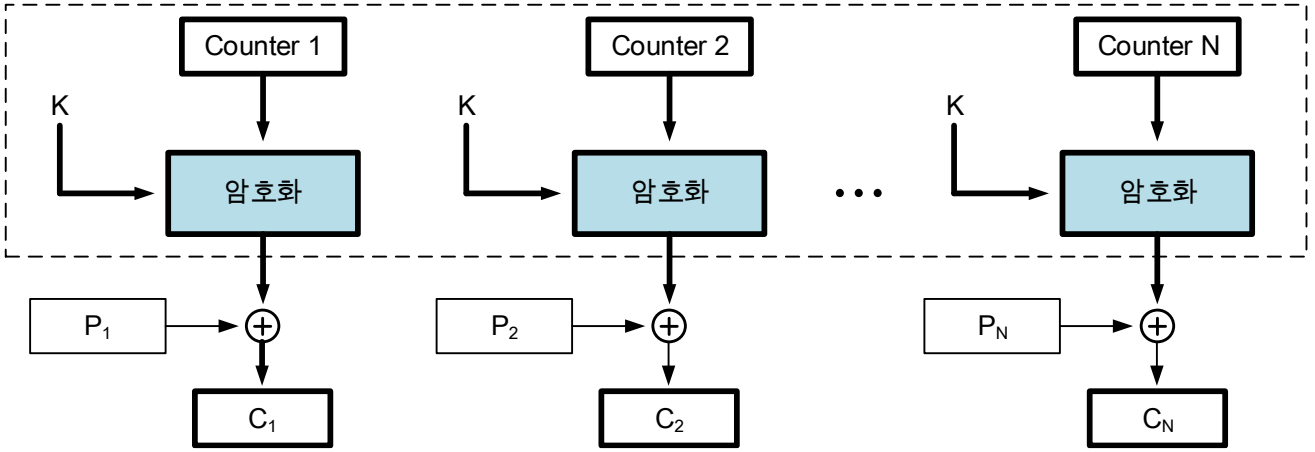
블록 암호 운용 모드

- 카운터(CTR, Counter) 모드
  - 모든 암호문 블록이 이전 단계 암호문 블록들과 독립적인 방식

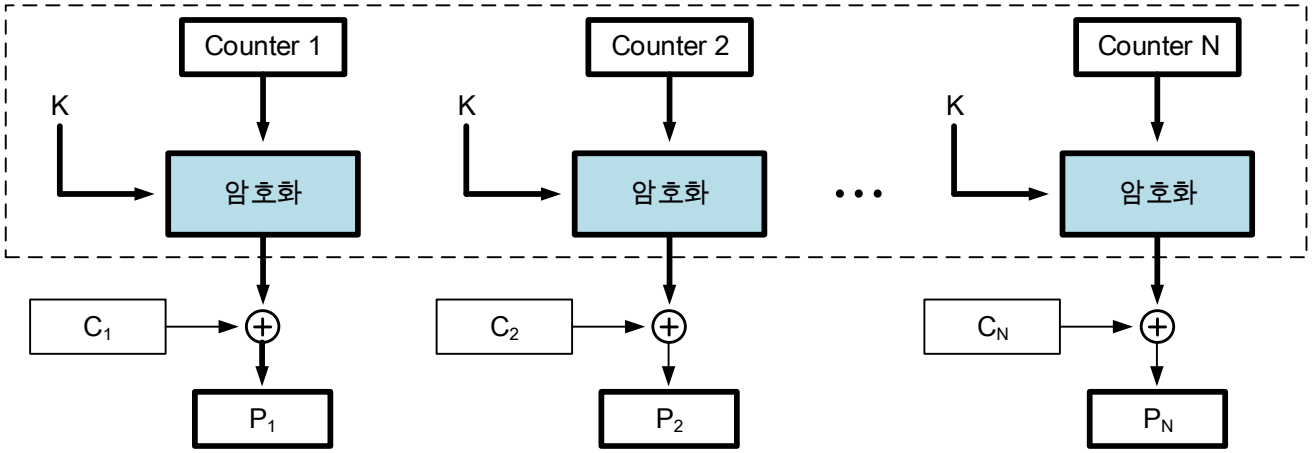


블록 암호 운용 모드

- CTR
  - 구조
    - 암호화



    - 복호화



블록 암호 운용 모드

- CTR
  - 장점
    - 병렬 처리 가능
    - 오류가 확산되지 않음
    - 패딩이 필요하지 않음

블록 암호 운용 모드

- 블록 암호 운용 모드 비교 표

암호 운용 모드	병렬 처리	패딩	랜덤 접근	초기화 백터	오류 확산
ECB	O	O	O	X	X
CBC	복호화만	O	복호화만	O	E: 해당 블록 이후 모든 블록 D: 해당 블록과 다음 블록
CFB	복호화만	X	복호화만	O	Shift register에서 오류가 완전히 소멸될 때까지
OFB	X	X	X	O	X
CTR	O	X	O	X	X

감사합니다!