

Network Security Essential

- Chapter 2 대칭 암호와 메시지 기밀성(2) -

발표자: 박 재 형(jaehyoung@pel.smuc.ac.kr)

상명대학교 프로토콜공학연구실

목차

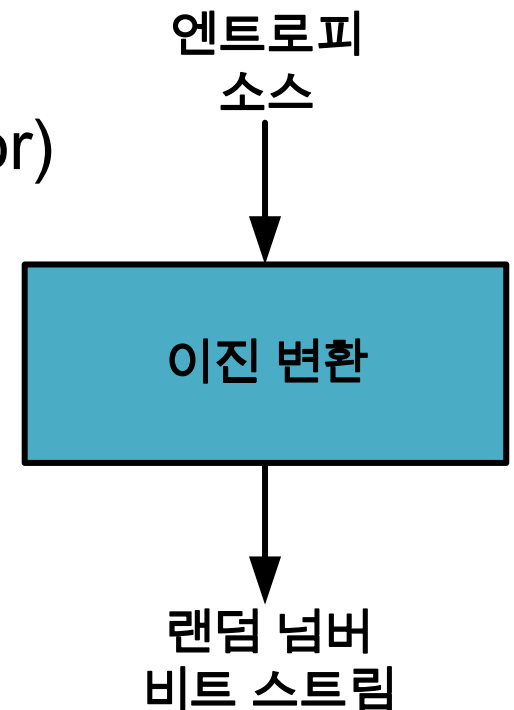
- 랜덤 넘버와 의사 랜덤 넘버
- 스트림 암호와 RC4
- 암호 블록 운용 모드

랜덤 넘버와 의사 랜덤 넘버

- 랜덤 넘버 (Random Number)
 - 정의
 - 정의된 범위 내에서 무작위로 추출된 수
 - 특징
 - 무작위성 (Randomness)
 - 수열이 어느 한 쪽으로 치우치지 않고 무작위로 분포 되어야 함
 - 균등 분포 (Uniform distribution)
 - 비트열에 나타나는 0과 1이 나타나는 빈도가 비슷해야 함
 - 독립성 (Independence)
 - 수열에서 추출한 부분 수열이 다른 수열로부터 추측할 수 없어야 함
 - 예측불가능성
 - 수열의 일부를 보고 이어지는 수를 예측할 수 없어야 함

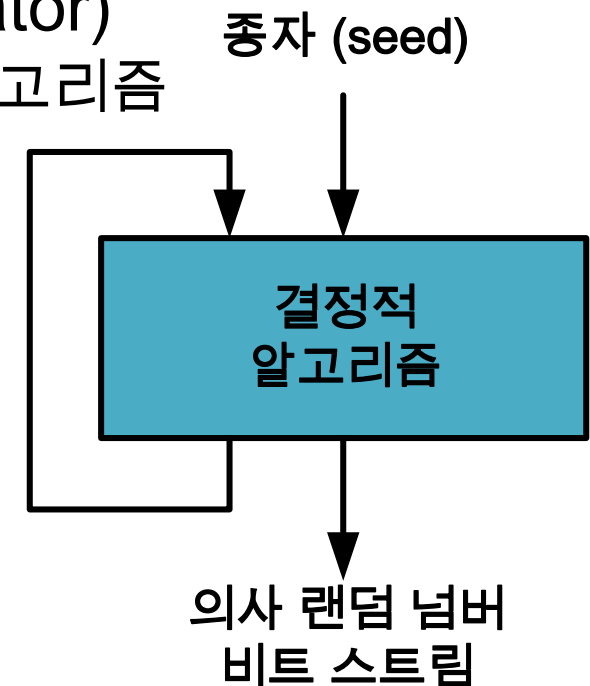
랜덤 넘버와 의사 랜덤 넘버

- 랜덤 넘버 (Random Number)
 - 진성 랜덤 넘버 (True random number)
 - 입력 값이 실제로 랜덤한 정보
 - 엔트로피 소스 (Entropy source)
 - 컴퓨터에서 물리적으로 얻을 수 있는 랜덤 정보
 - e.g., 키 입력 타이밍 패턴, 마우스 움직임 등
 - TRNG (True Random Number Generator)
 - 입력으로 엔트로피 소스를 사용하여 랜덤한 바이너리를 출력 함



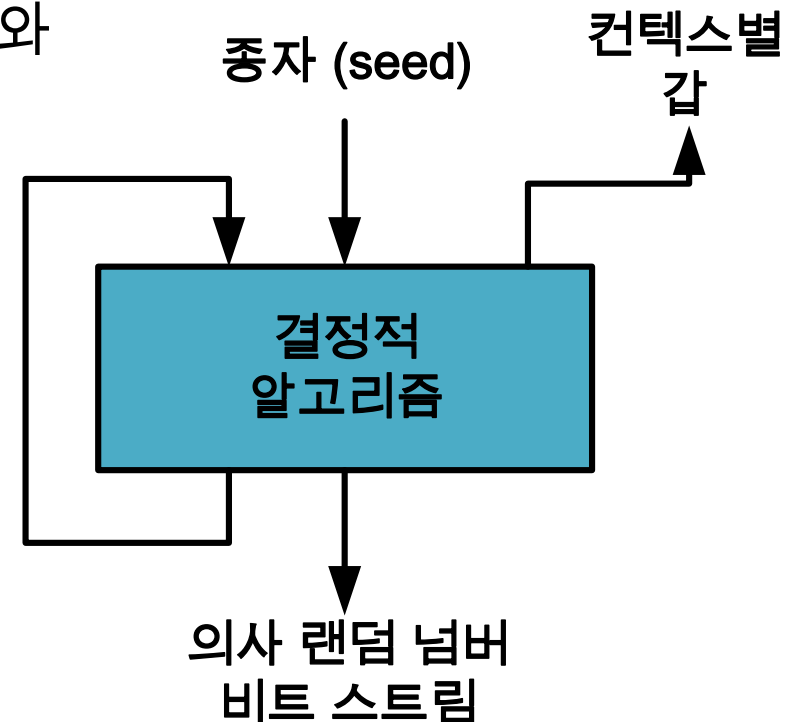
랜덤 넘버와 의사 랜덤 넘버

- 랜덤 넘버 (Random Number)
- 의사 랜덤 넘버 (Pseudo random number)
 - 알고리즘 기법에 기초하여 생성되고 무작위성 테스트를 통과 할 수 있는 수열
 - 철학 적으로 완전한 난수는 아님
- PRNG (Pseudo random number generator)
 - 무한 비트열을 생성 하기 위해 사용 되는 알고리즘
 - 종자 (Seed)는 고정된 값
 - 종자 값에 따라 난수 결정
 - 결정적 알고리즘 사용
 - 종자 값이 같으면 출력 값이 같은 알고리즘



랜덤 넘버와 의사 랜덤 넘버

- 랜덤 넘버 (Random Number)
- 의사 랜덤 넘버 (Pseudo random number)
 - PRF (Pseudo random)
 - 공정 길이 의사 랜덤 비트열을 생성 하는데 사용되는 함수
 - 컨텍스트별 값은 상황에 따른 입력 값
 - 생성되는 비트열만 다를 뿐 PRNG와 차이점이 없음



스트림 암호

- 스트림 암호 (Stream cipher)

- 정의

- 비트나 바이트 단위로 입력되는 요소를 연속적으로 처리하는 대칭키 암호 알고리즘

- 특징

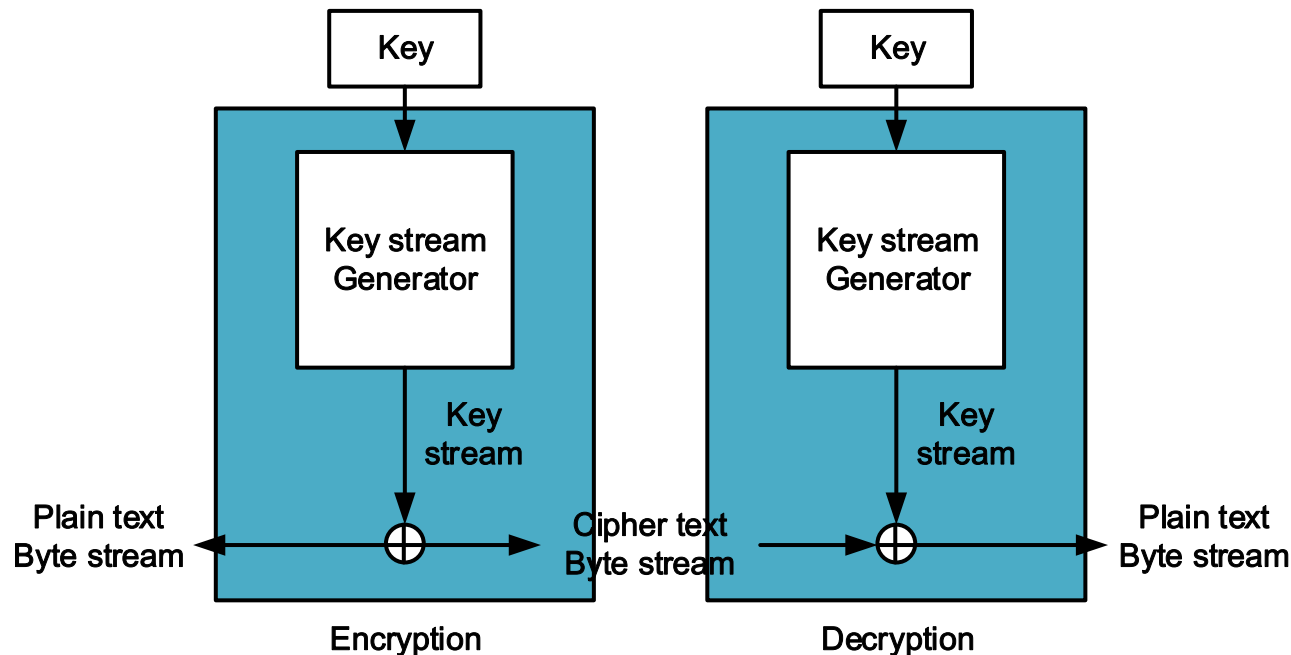
- 의사 랜덤 비트 생성기를 사용
- 키스트림 생성기 사용
- 전수 조사 공격에 대응하기 위해 최소 128 bits 이상의 키 길이를 가져야 함
- 평문과 키스트림에서 출력된 값을 XOR하여 암호문을 얻음

스트림 암호

- 스트림 암호 (Stream cipher)

- 스트림 암호의 구조

- PRNG의 출력 값인 키스트림과 평문을 XOR연산하여 암호문을 만들어냄
- 키는 공유된 키(대칭 암호)



스트림 암호

- 스트림 암호 (Stream cipher)

- e.g.,

11001100	평문	10100000	암호문
\oplus 01101100	키 스트림	\oplus 01101100	키 스트림
<hr/>		<hr/>	
10100000	암호문	11001100	평문

스트림 암호

- 스트림 암호 (Stream cipher)
 - 스트림 암호 설계시 주의 할 점
 - 키스트림 주기가 커야함
 - PRNG는 결국에는 반복적으로 나타나는 비트스트림을 만들어 냄
 - 반복 주기가 길 수록 해독이 어려워짐
 - 키스트림은 가능하면 진성랜덤 스트림의 특성에 근사
 - 0과 1의 개수가 거의 동일해야 함(균등 분포)
 - 입력 키의 길이가 충분히 길어야 함
 - 입력되는 키의 값이 전수공격을 차단해야 함(최소 128 bits)

스트림 암호

- 스트림 암호 (Stream cipher)
 - 스트림 암호의 장점
 - 패딩이 필요하지 않음
 - 블록 암호보다 속도가 빠르고 적은 양의 코드 사용
 - 실시간 사용 가능
 - 동일한 크기의 키를 사용하는 블록 암호만큼 안전
 - 스트림 암호의 단점
 - 서로 다른 평문을 동일한 키스트림으로 암호화 할 때 암호 해독이 단순해짐
 - 키스트림이 노출되면 모든 암호문은 공격자에게 노출 됨

스트림 암호

- 스트림 암호 (Stream cipher)
 - 두개의 평문을 같은 키로 암호화 한다면 암호해독이 단순해지는 경우가 간혹 생김
 - 두개의 암호문을 XOR시키면 원래 평문과의 XOR값
 - e.g., 평문1:10110111 암호문1:11011011
평문2:11001100 암호문2:10100000

$$\begin{array}{r} 10110111 \\ \oplus 11001100 \\ \hline 01111011 \end{array}$$

< 평문 XOR >

- 키스트림:01111011

$$\begin{array}{r} 11011011 \\ \oplus 10100000 \\ \hline 01111011 \end{array}$$

< 암호문 XOR >

스트림 암호

- 블록 암호와 스트림 암호의 실행 속도 비교

암호 알고리즘	키 길이(bits)	속도(Mbps)	블록 암호
DES	56	9	
3DES	168	3	
RC2	다양한 길이	0.9	스트림 암호
RC4	다양한 길이	45	

RC4 알고리즘

- RC4

- 특징

- 스트림 암호의 종류중 하나
- 바이트 단위로 작동됨, 다양한 크기의 키 사용
 - 1~256 byte
- 랜덤 치환 방식을 기반으로 함
- 하나의 바이트를 출력하기 위해 8에서 16번의 연산이 필요
 - 소프트웨어 상에서 매우 빠르게 동작 함

- RC4동작 과정

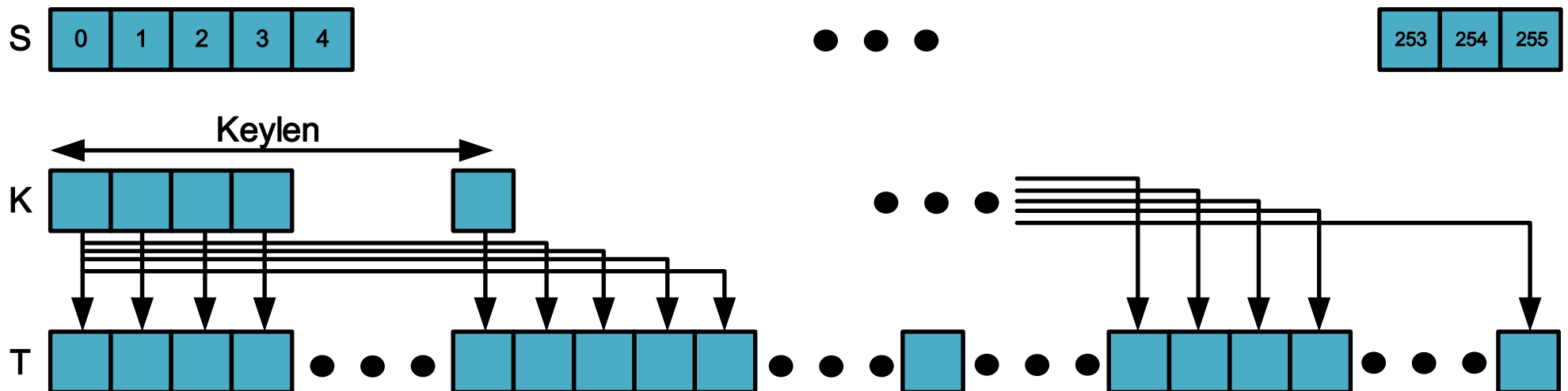
1. 벡터 S의 초기화
2. 벡터S의 초기치환
3. 스트림 생성

RC4 알고리즘

- 벡터 S의 초기화

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

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



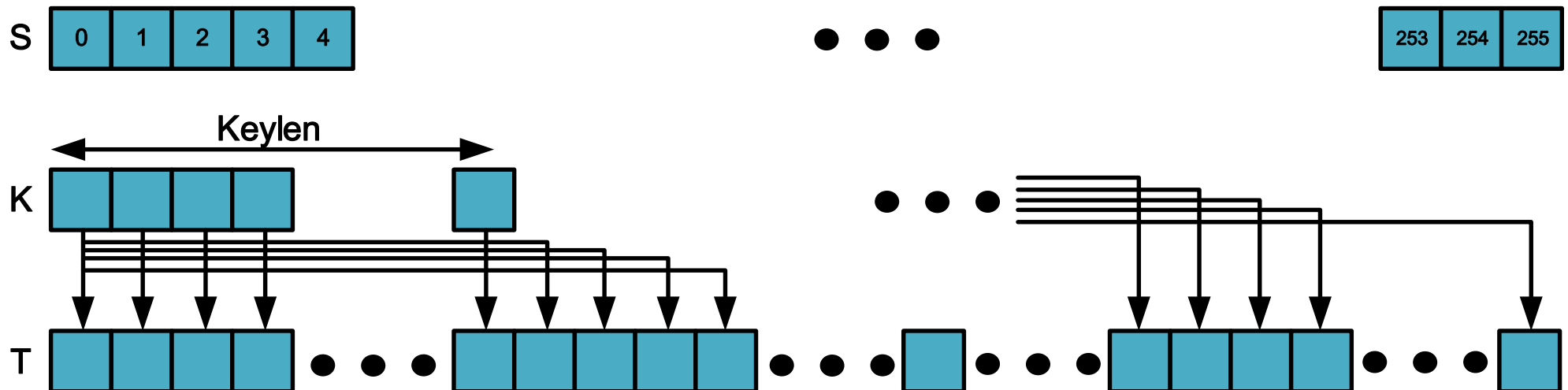
RC4 알고리즘

- 벡터 S의 초기화

- 임시 벡터 T 생성

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

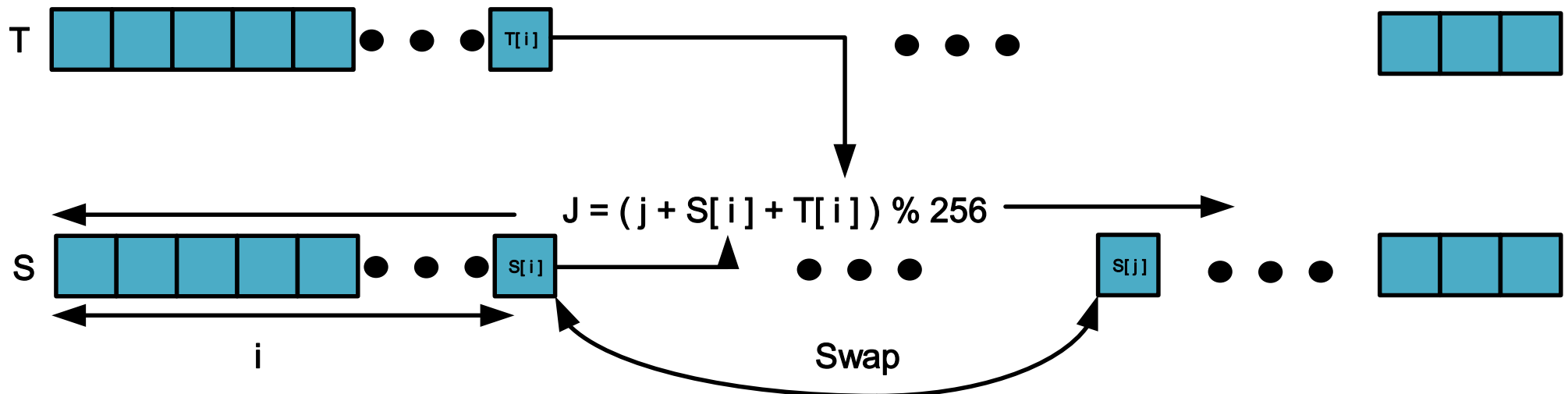
```
for i = 0 to 255 do  
  S[i] = i;  
  T[i] = K[i mod keylen];
```



RC4 알고리즘

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

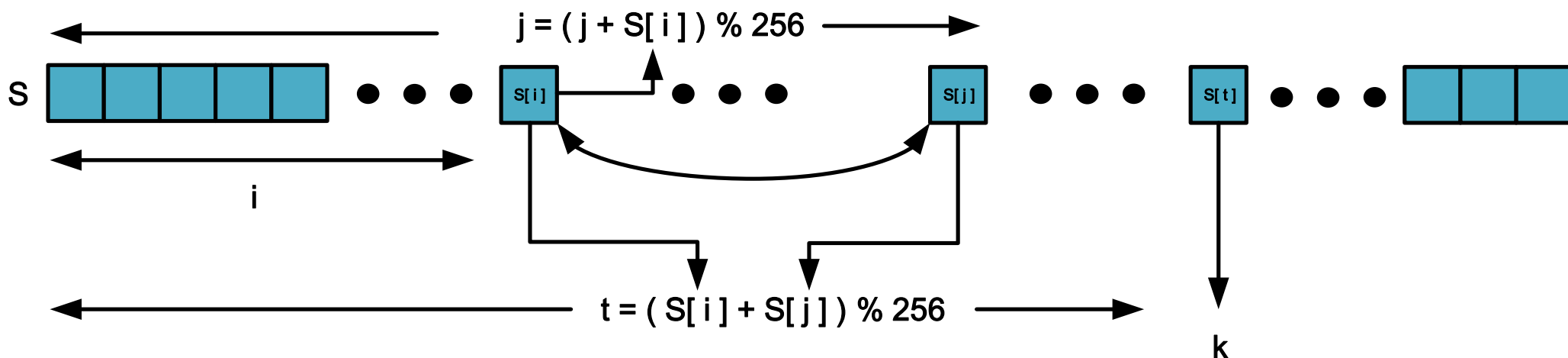
```
J = 0;  
for I = 0 to 255 do  
    j = (j + S[i] + T[i]) mod 256;  
    Swap( S[i], S[j] );
```



RC4 알고리즘

• 키 스트림 생성

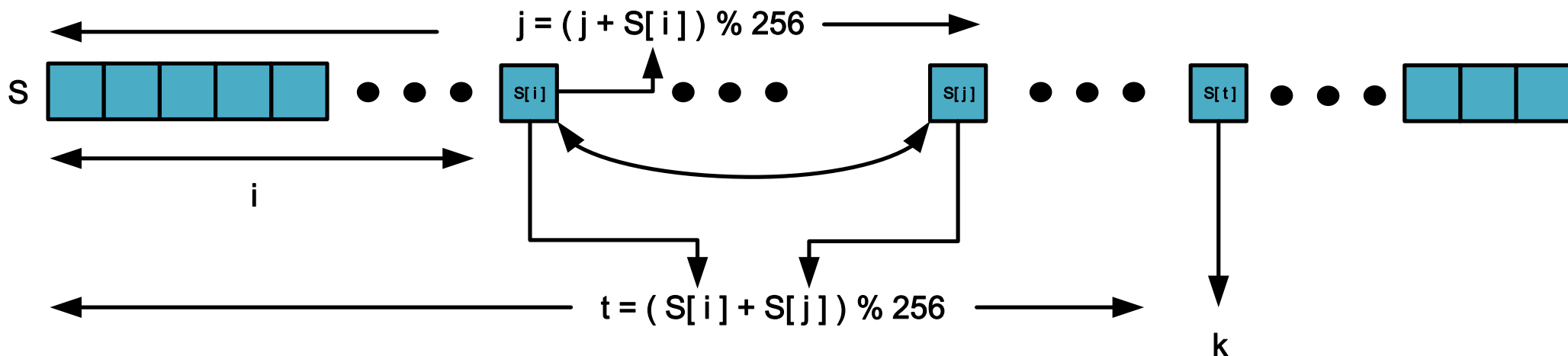
- 각 $S[i]$ 는 현재 상태 값에 따라 값이 교환 됨
- $S[i]$ 와 $S[j]$ 를 이용하여 인덱스 t 를 계산
- $S[t]$ 에 있는 값을 키 스트림으로 생성
- $S[255]$ 까지 도달하면 $S[0]$ 에서 처음부터 과정 반복



RC4 알고리즘

- 키 스트림 생성

```
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;
```



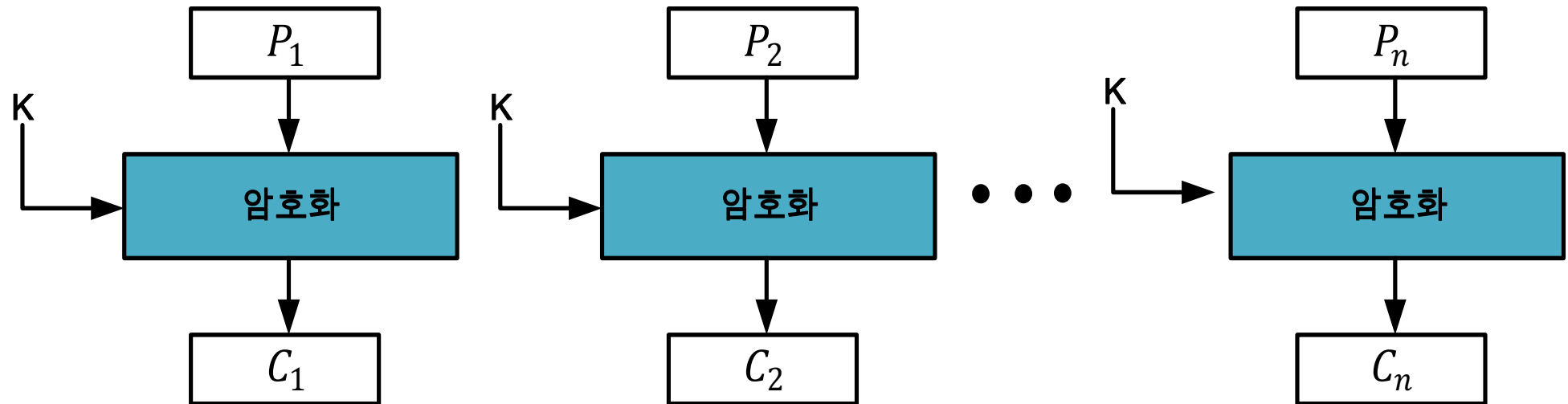
블록 암호 운용 모드

- 블록 암호 운용 모드

- 하나의 키를 사용하여 블록 암호를 반복적으로 이용하는 암호화 방식
- 길이가 가변적인 데이터를 암호화 하는 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

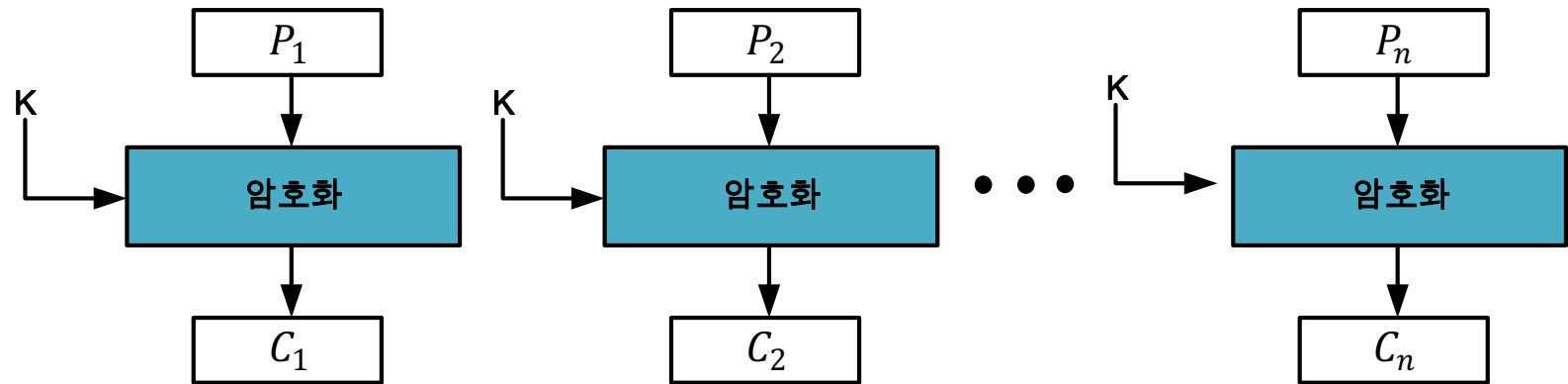


블록 암호 운용 모드

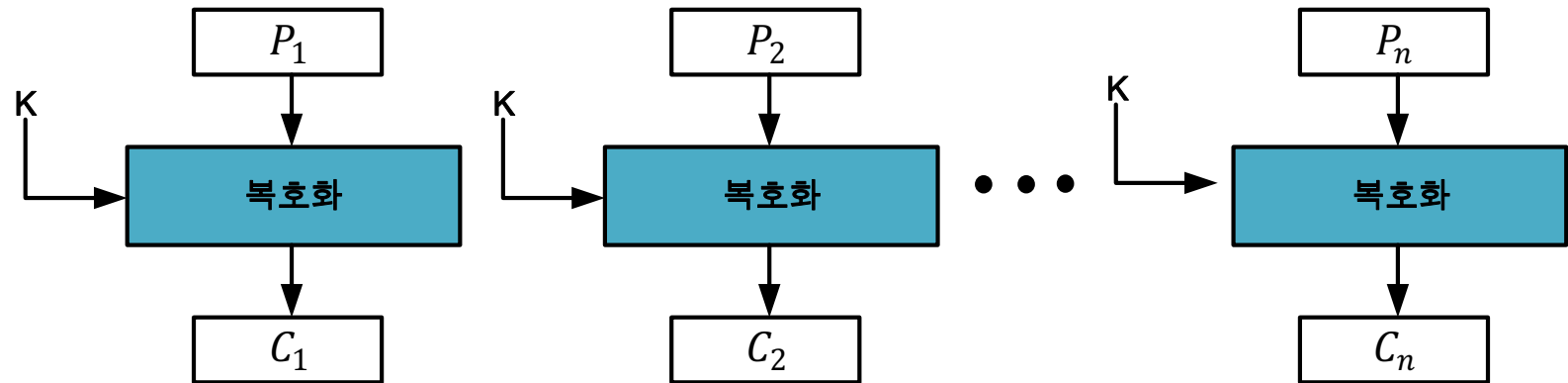
- ECB

- 구조

- 암호화



- 복호화



블록 암호 운용 모드

- ECB

- 장점

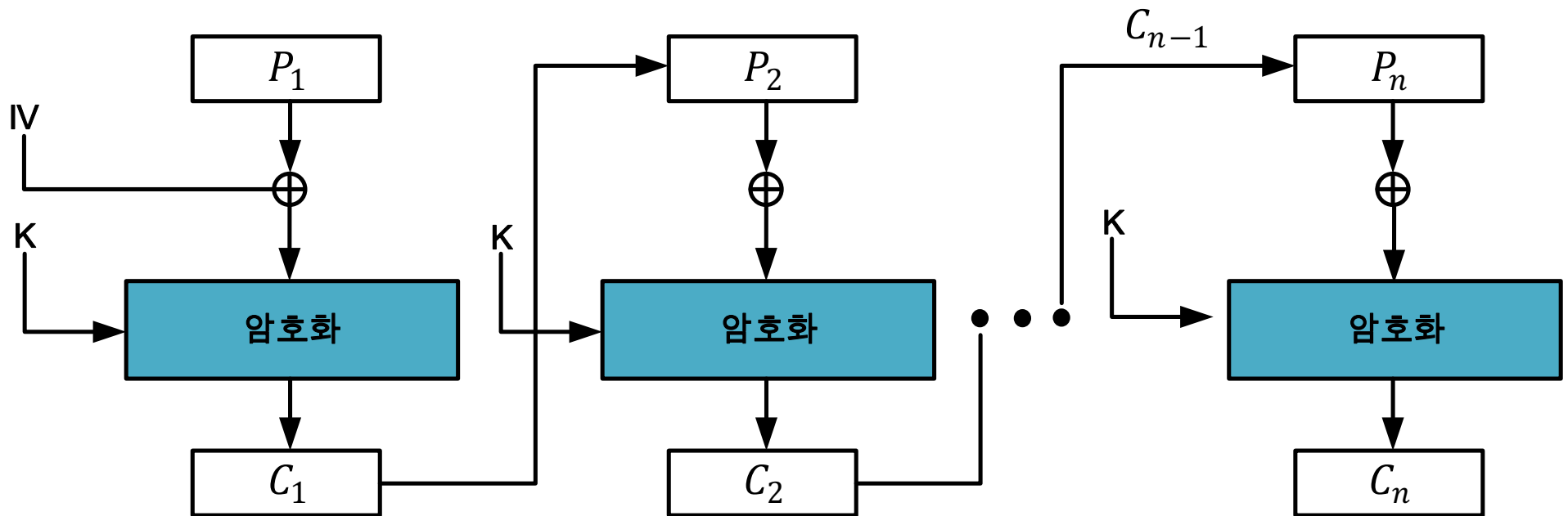
- 압/복호화 처리 속도가 빠름
 - 병렬 처리 가능
 - 오류가 확산되지 않음

- 단점

- 패딩이 필요함
 - 평문의 반복 패턴이 드러남
 - 패턴 공격에 취약

블록 암호 운용모드

- 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

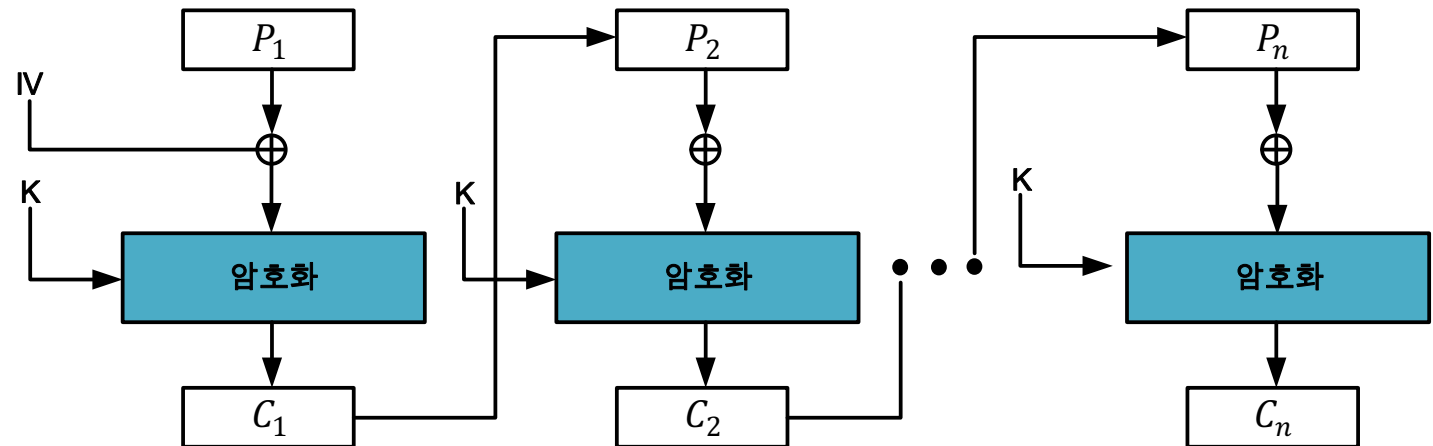


블록 암호 운용모드

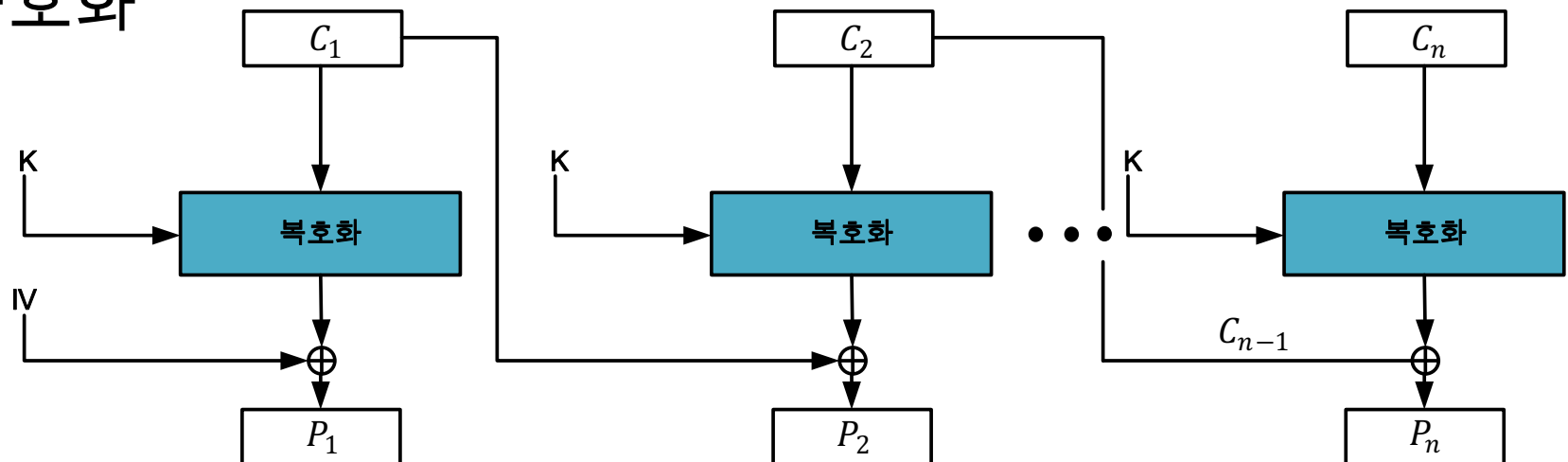
- CBC

- 구조

- 암호화



- 복호화



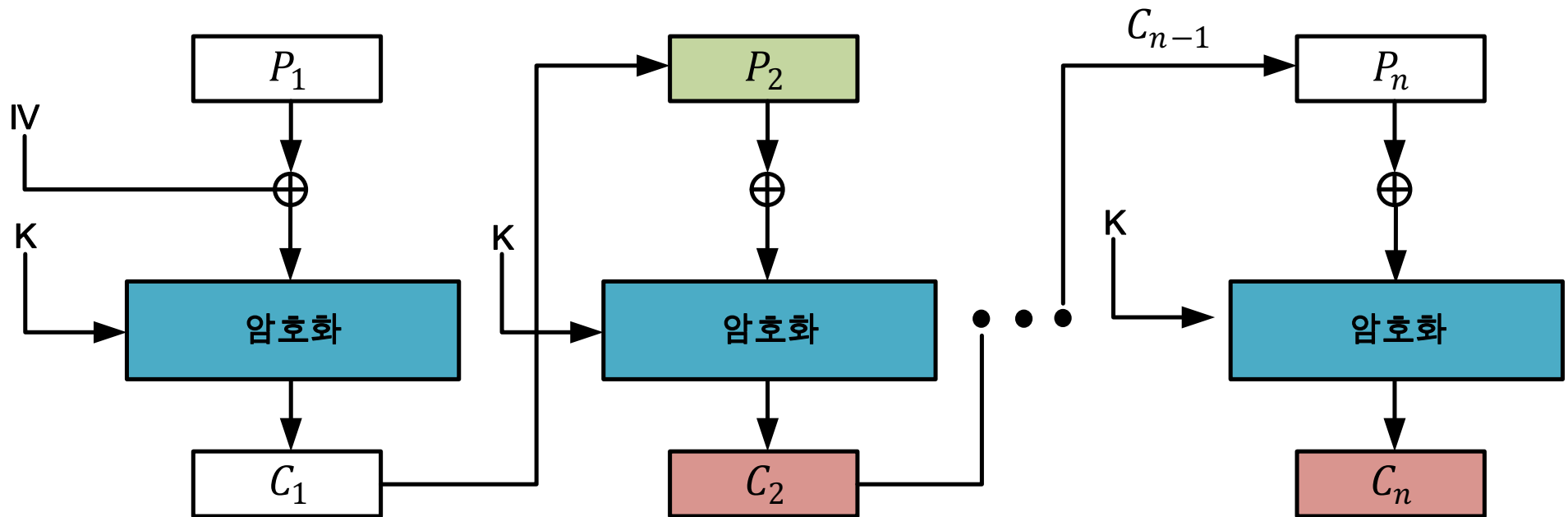
블록 암호 운용모드

- CBC

- 오류 확산

- 암호화 과정

- 평문 블록 하나가 오류 났을 때, 현재 암호문 블록 이후 전체 암호문 블록에 영향



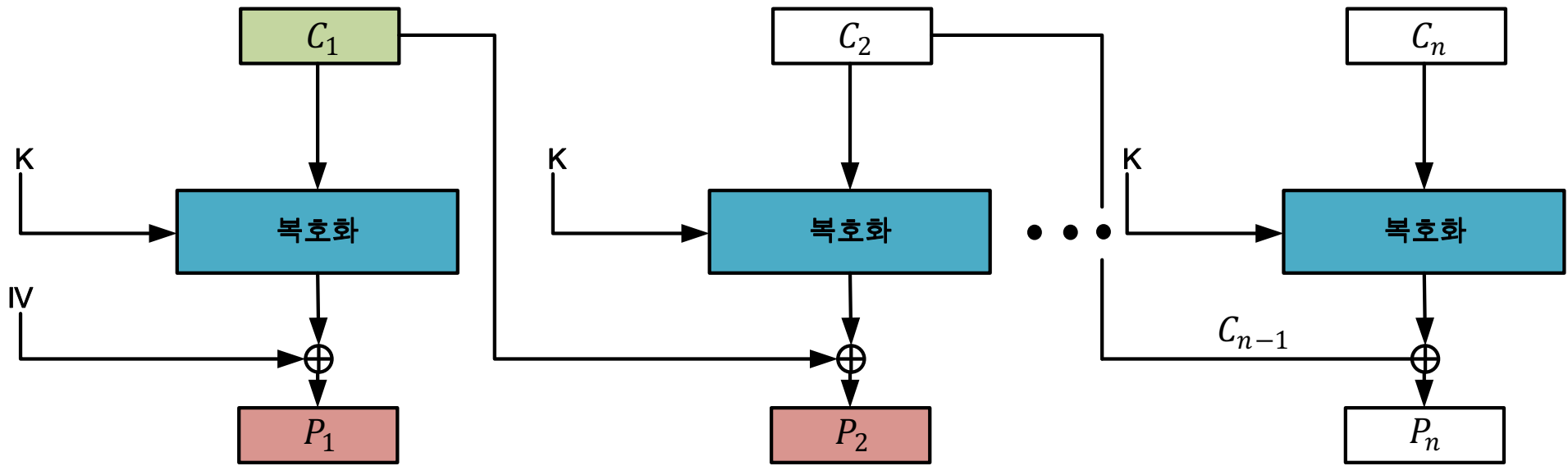
블록 암호 운용모드

- CBC

- 오류 확산

- 복호화 과정

- 암호문 블록 하나가 오류 났을 때, 현재 평문 블록과 다음 평문 블록에만 영향



블록 암호 운용모드

- CBC

- 장점

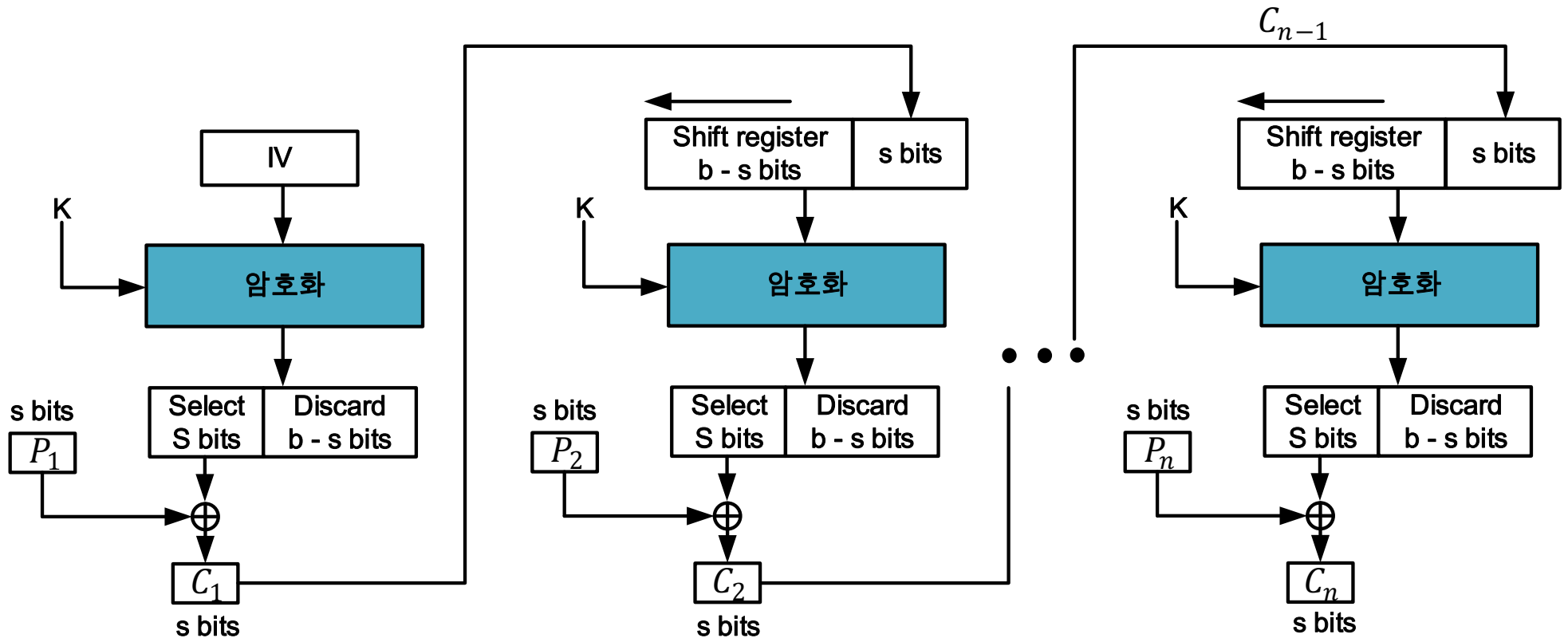
- 평문의 반복 패턴이 드러나지 않음
 - 체인구조로 연속해서 XOR연산

- 단점

- 오류가 확산됨
 - 암호화 과정은 병렬 처리 불가능
 - 패딩이 필요함

블록 암호 운용모드

- 암호 피드백 (CFB, Cipher feedback)모드
- CBC의 변형으로, 블록 암호를 스트림 암호로 변환하여 암호/복호화하는 방식

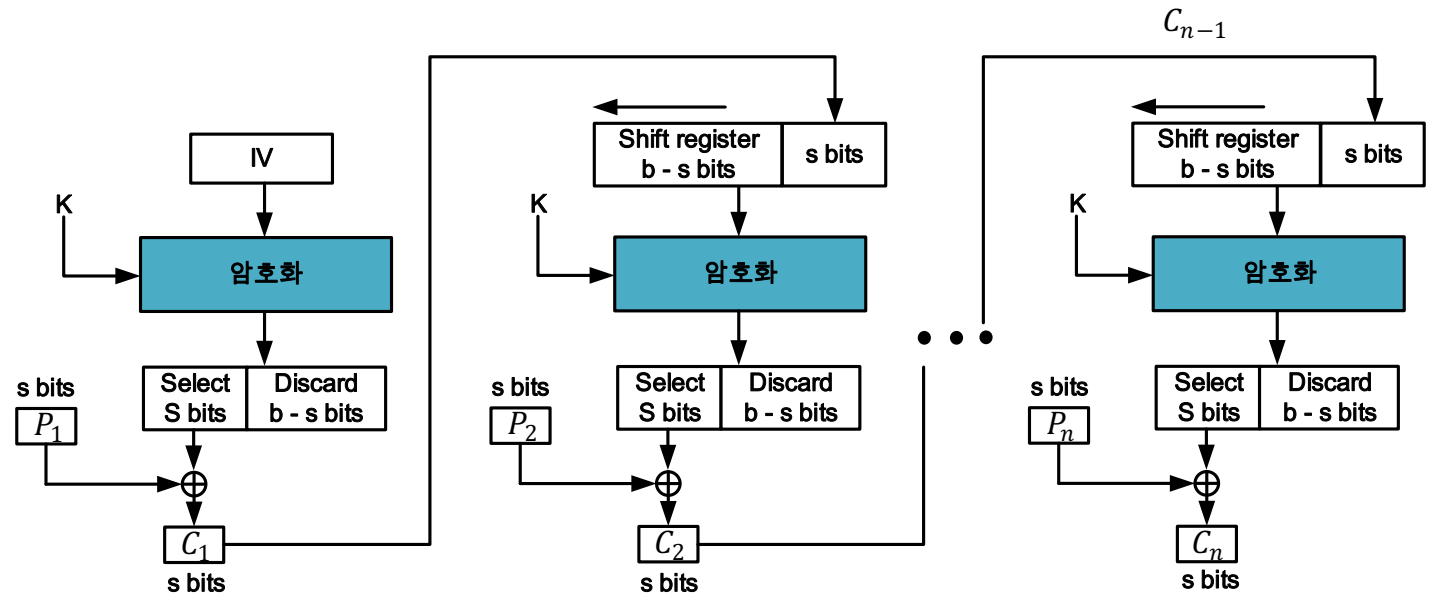


블록 암호 운용모드

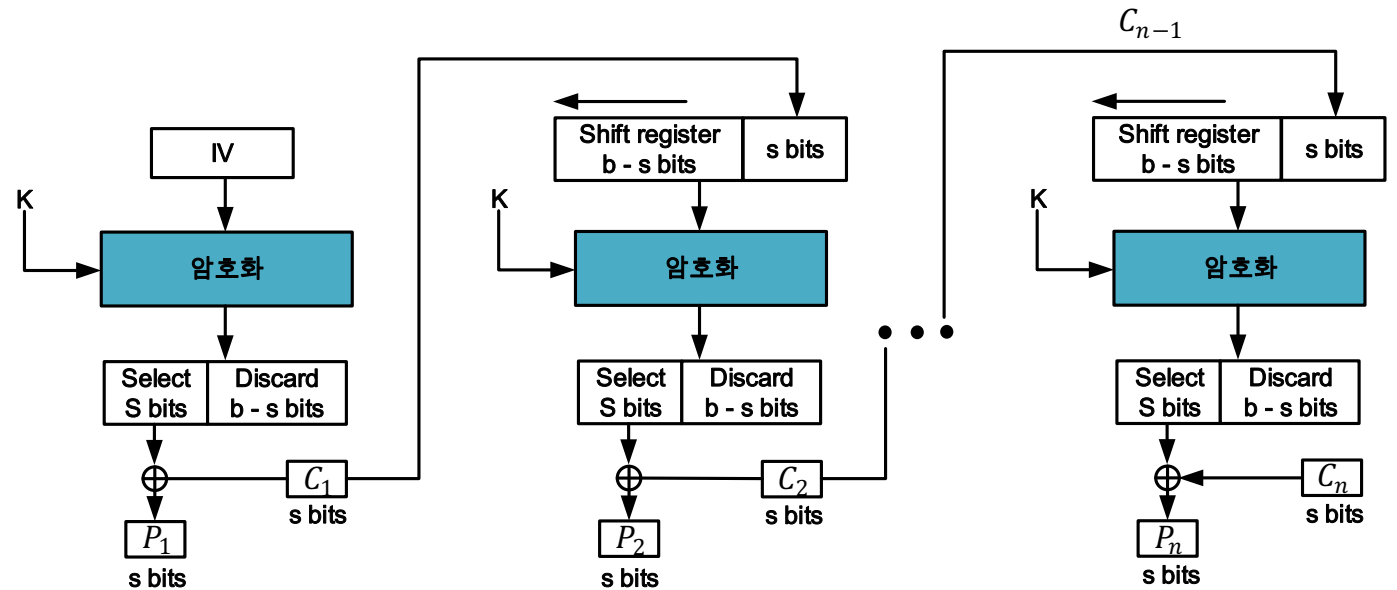
- CFB

- 구조

- 암호화



- 복호화



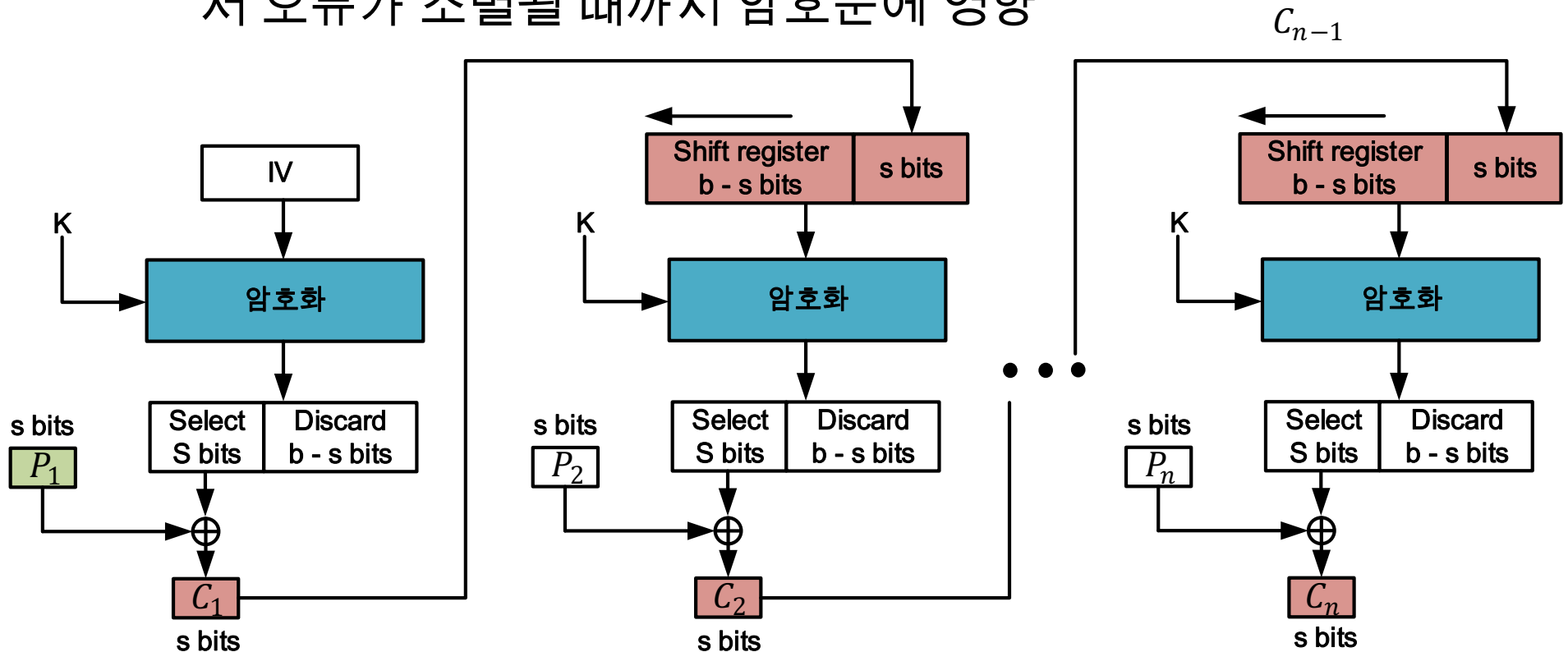
블록 암호 운용모드

- CFB

- 오류 확산

- 암호화 과정

- 평문 블록 하나가 오류 나면, 현재 암호문 블록 이후 Shift register에서 오류가 소멸될 때까지 암호문에 영향



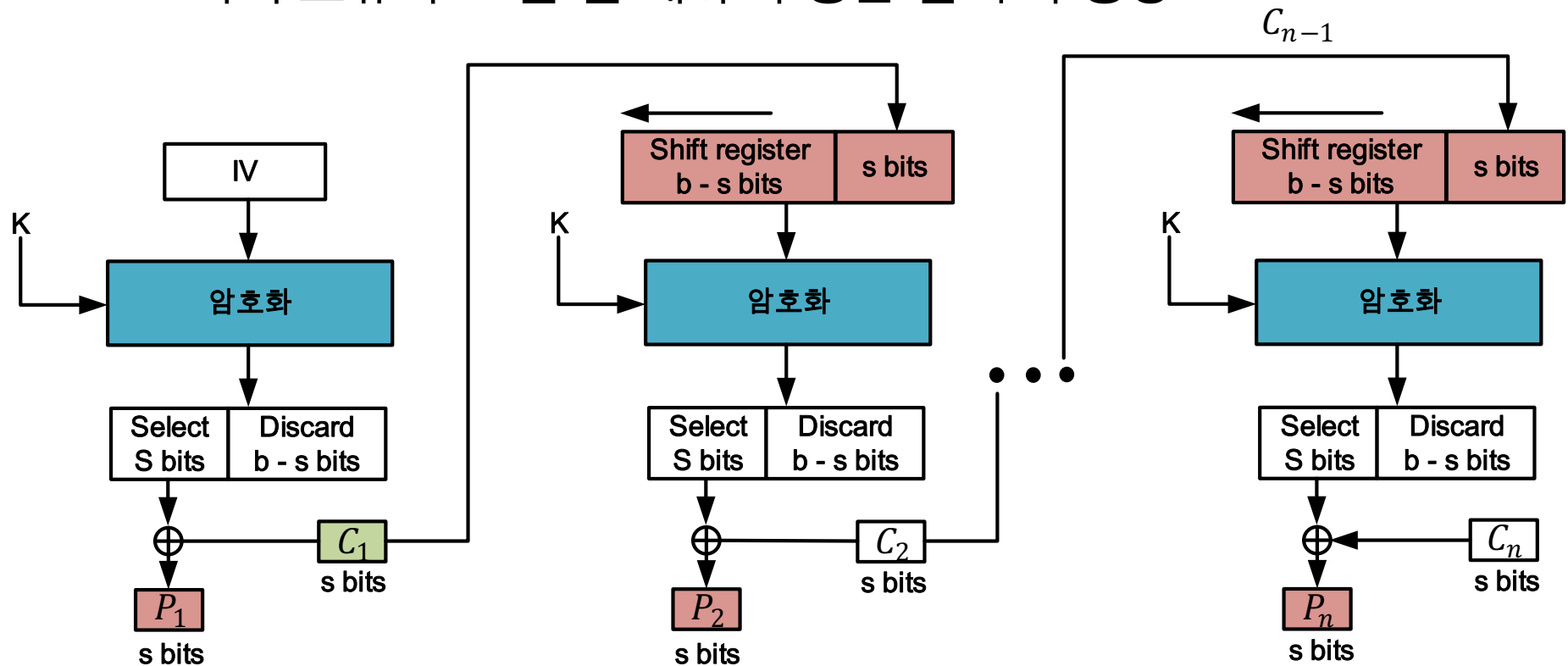
블록 암호 운용모드

- CFB

- 오류 확산

- 복호화 과정

- 암호문 블록 하나가 오류 났을 때, 현재 평문 블록 이후 Shift register 에서 오류가 소멸 될 때까지 평문 블록에 영향



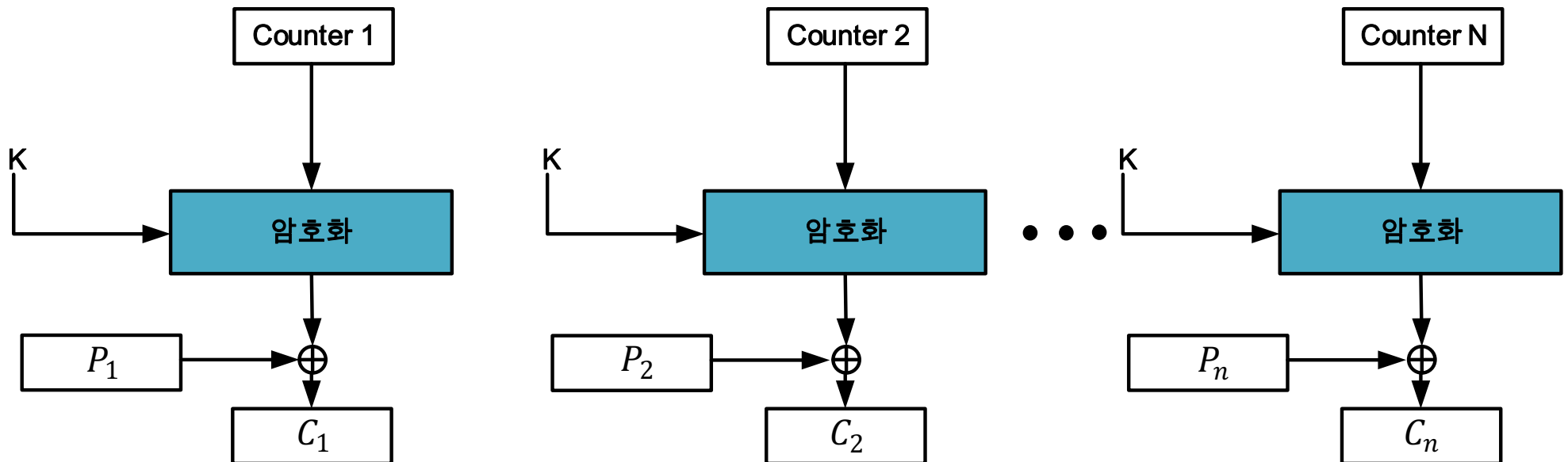
블록 암호 운용모드

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

블록 암호 운용모드

- 카운터 (CTR, Counter)모드

- 모든 암호문 블록이 이전 단계 암호문 블록들과 독립적인 방식

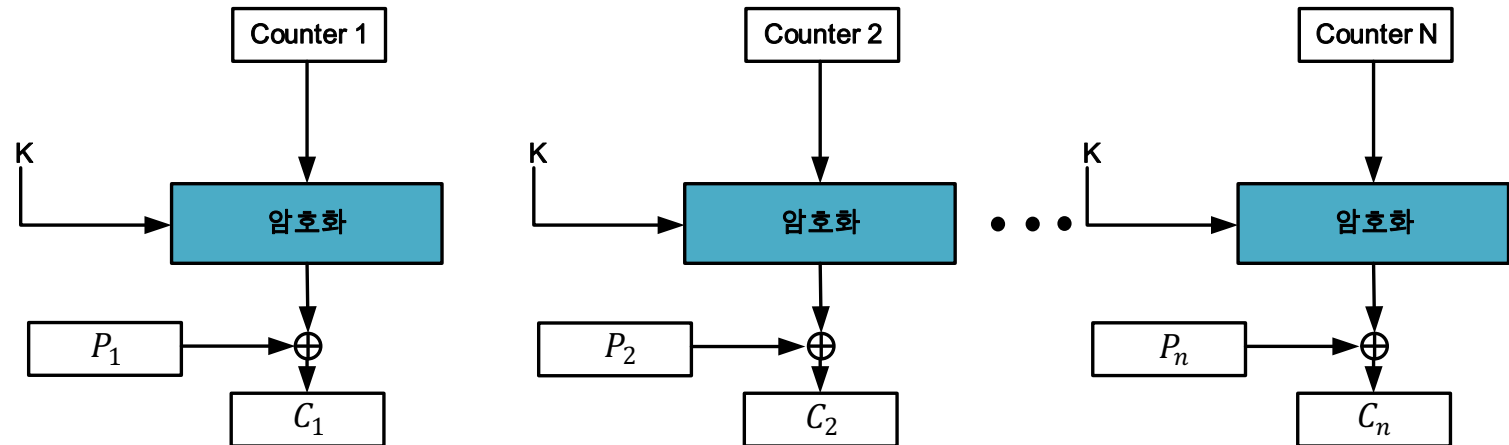


블록 암호 운용모드

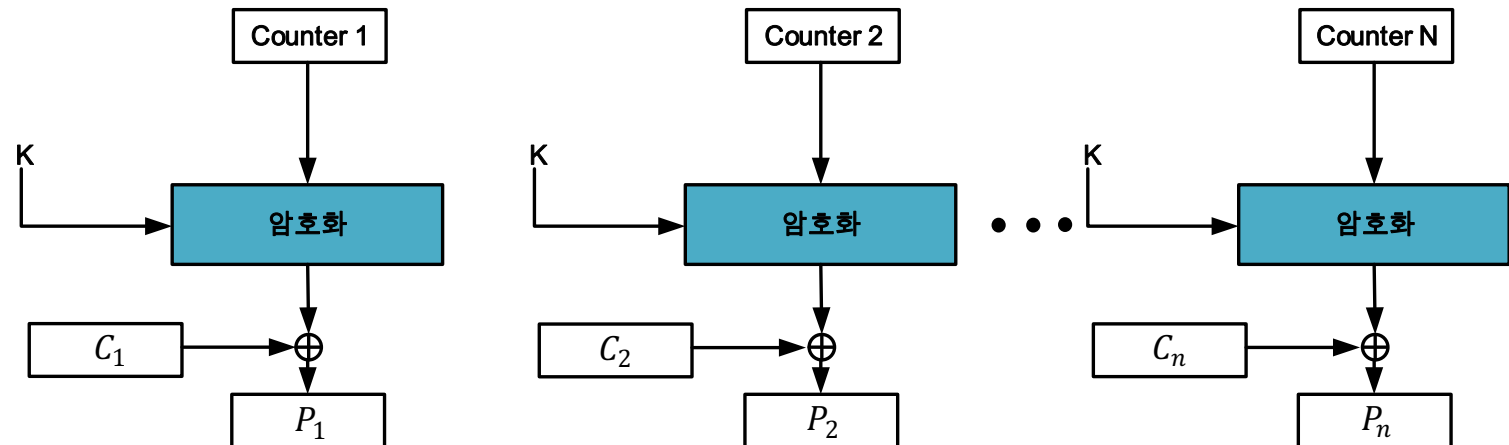
- 카운터 (CTR, Counter)모드

- 구조

- 암호화



- 복호화



블록 암호 운용모드

- 카운터 (CTR, Counter)모드

- 장점

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

- 단점

블록 암호 운용모드

- 블록 암호 운용 모드 비교 표

암호 운용 모드	병렬 처리	패딩	랜덤 접근	초기화 벡터	오류 확산
ECB	O	O	O	X	X
CBC	복호화만	O	복호화만	O	E : 해당 블록이후 모든 블록 D : 해당 블록과 다음 블록
CFB	복호화만	X	복호화만	O	Shift registe에서 오류가 완전히 소멸 될때까지
CTR	O	X	O	X	X

Thanks!

박재형 (jaehyoung@pel.smuc.ac.kr)