Shuffling and Cancelable Biometric chemes

JCCI 2012

1000000100000110000101000111100100010110011101010011111010000111000100100110110101101111011000110100101110111001100101010111111

2012 년 4 월 26 일

박정열, 박기현, 송홍엽

Coding & Crypto Lab.

연세대학교 전기전자공학과

Two Problems

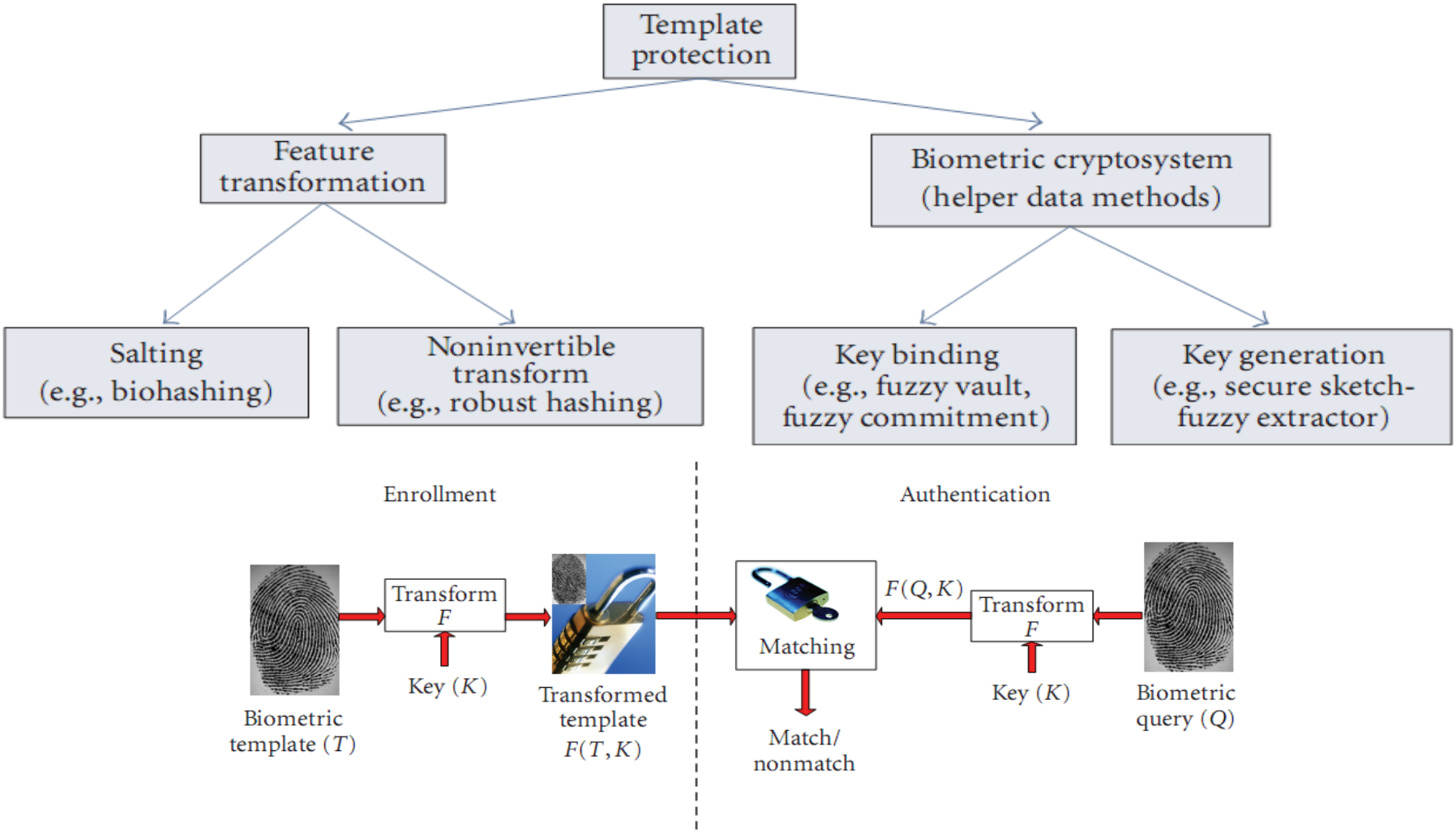
- **생체 정보를 인증에 사용하기 위한 여러 방법들이 국내외 오래동안 시도됨.**
  - 지문, 홍체, 얼굴인식, 정맥류 등 정보를 사용한 인증 방법이 예전부터 널리 사용됨
  - 생체정보 인식기술의 난해함과 생체정보의 재사용 불가로 상용화에 많은 어려움.
- **기밀성, 무결성, 인증, 부인방지를 위한 전통적 암호화 방식은 학문적으로 이미 상당한 수준.**
  - 비밀키의 보관/관리에 암호화 기술의 상용화 성공 여부가 달려있음.
  - 상당히 많이 활용되고 있으나 비밀키 보관/관리의 허술함으로 많은 문제점 발생.

동시에 해결?? ➔ 생체정보 암호화 기술

- **비밀키를 생체정보와 함께 암호화하여 보관하고 추후 입력되는 생체정보를 인증하여 비밀키를 찾아낼 수 있도록 함. (Secret Key 관리의 보안성 강화)**
  - 여기서 비밀키는 대칭키방식의 비밀키와 공개키방식의 비밀키를 모두 포함.

- **생체정보를 패스워드를 사용하여 암호화하여 저장(혹은 사용) 함으로써 생체정보가 도난/탈취되어도 동일한 생체정보를 새로운 패스워드를 사용하여 암호화 함으로써 다시 새롭게 사용 가능토록 함. (생체정보 재사용/무효화 가능)**

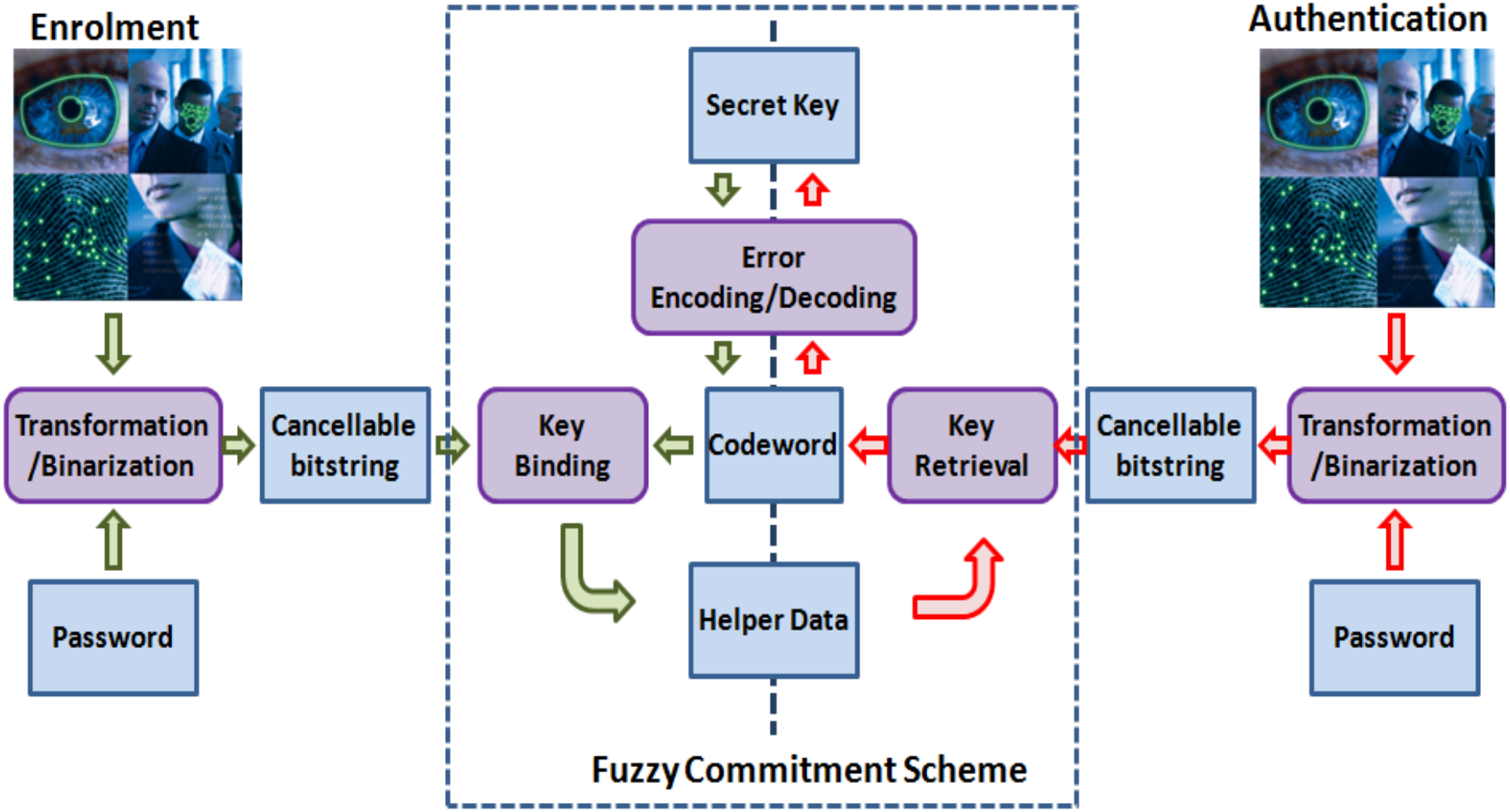
Biometric Encryption



Features of Biometric Encryption

- Diversity
- Re-usability (Cancellability)
- Non-invertibility (Privacy)
- Security
- Performance

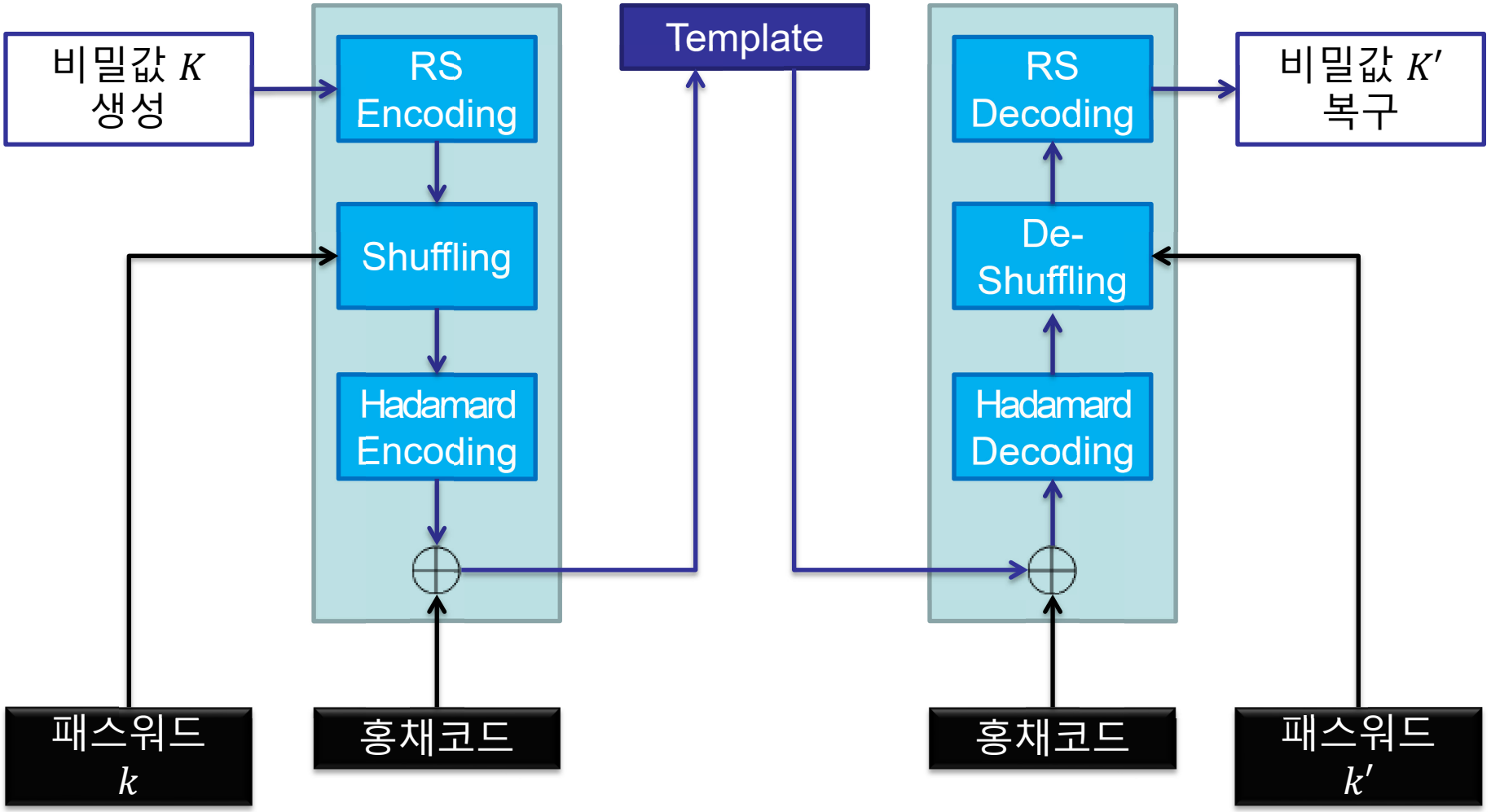
Key Binding Schemes



Content

- **Cancelable biometric scheme**
  - 개인의 생체정보를 template 형태로 저장한 뒤, 나중에 이 정보를 이용하여 개인을 식별하고 정보를 제공
  - Template는 언제나 폐기 후 재생성 가능
- **Kanade 등의 scheme은 다음 요소를 사용한다.**
  - 오류정정부호 : 생체정보 판독시 발생하는 오류를 제거
  - Shuffle functions (keyed permutations)
    : template의 각 비트를 뒤섞어 저장함
- **제안된 것만큼 안전하지 않음**
  - Permutation array
  - Permutation array가 shuffle에 비해 얼마나 좋은가?

Kanade 등의 CBS 전체 구조

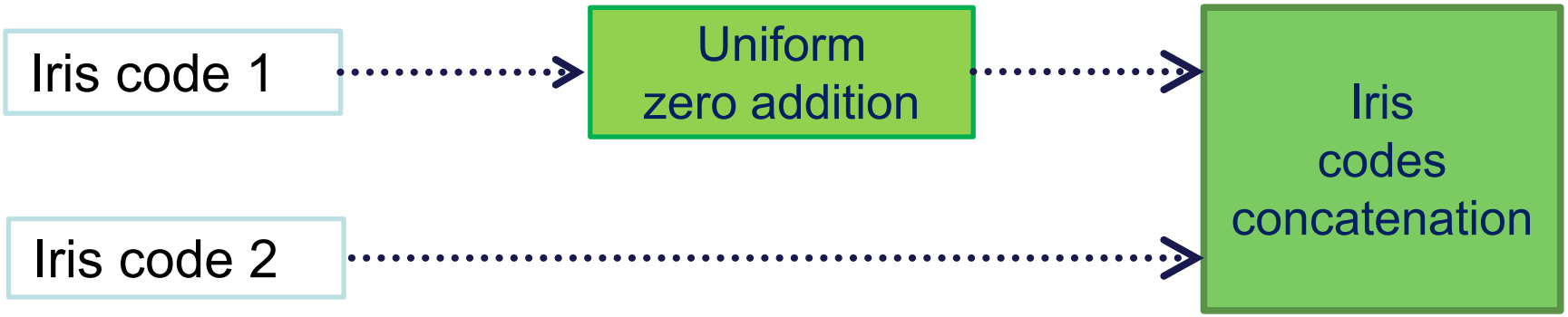


Kanade 등의 Cancelable Biometric Scheme

- **Error correcting code**
  - 홍채정보 내부의 오류 제거
- **양쪽 홍채정보를 사용**
  - 낮은 FRR (False Rejection Rate)
  - 높은 entropy
- **사용자 키를 통한 permutation 사용**
  - Cancelability
  - 낮은 FAR (False Accept Rate)
  - 추가적인 entropy

홍채코드

- **3136 bits**
- **Uniform zero additions**
  - More error correction for iris code 1 (1188 bits + 760 zeroes = 1948 bits)
  - Less error correction for iris code 2 (1188 bits)



Error Correcting Codes

- $\left(2^{k-1}, k, 2^{k-2}\right)$ **Hadamard code** $(k = 7)$
  - 64 bits/block x 49 blocks = 3136 bits
  - 배경 노이즈를 주로 제거

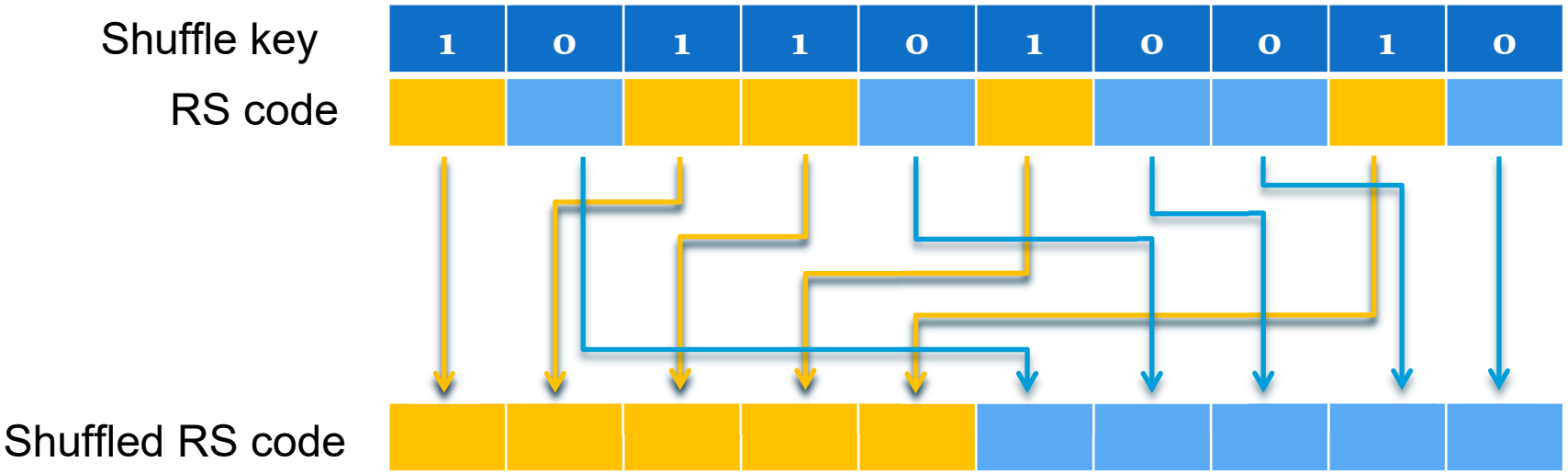
- **Reed-Solomon code**
  - $(127,127 - 2t)$ code over $F_{2^7} \rightarrow (49, 49 - 2t)$
  - $t = 12,13,14, \ldots$
  - 7 bits/block x 49 blocks = 343 bits
  - Burst error를 주로 제거

패스워드

- **49 bits**
- **A shuffle key**
  - 49개의 블럭을 뒤섞는 Shuffle (permutation)을 생성
- **Cancelability**
  - Password에 맞추어 새로운 template 생성

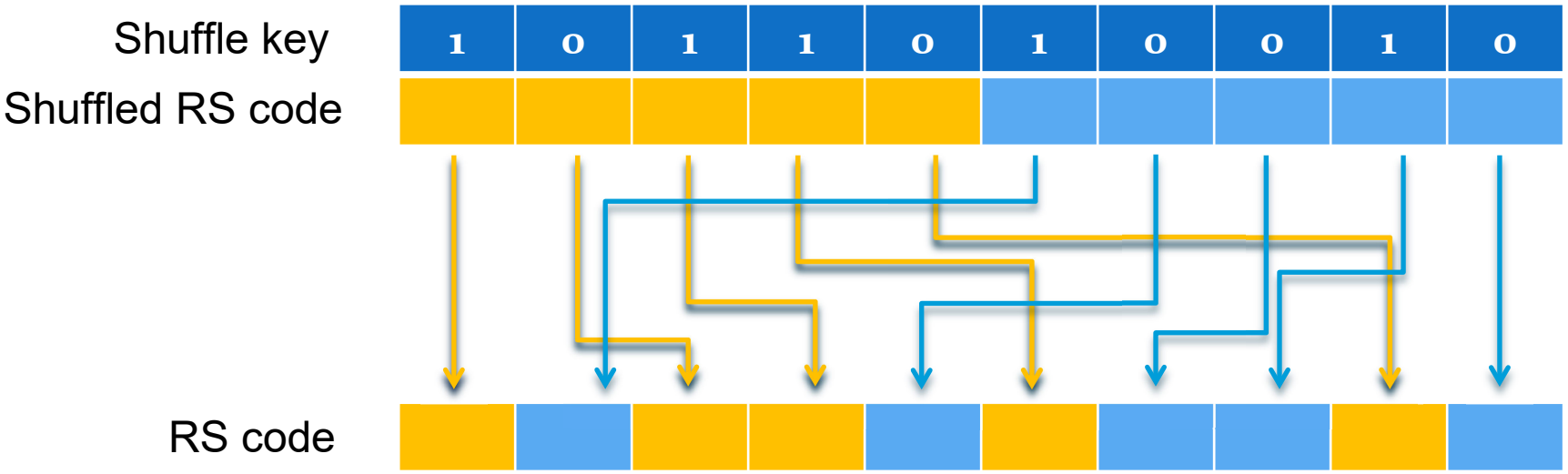
Shuffling

- 49개 블록에 대한 keyed permutation
- RS 코드의 systematic 성질 제거
- Quasi order-preserving

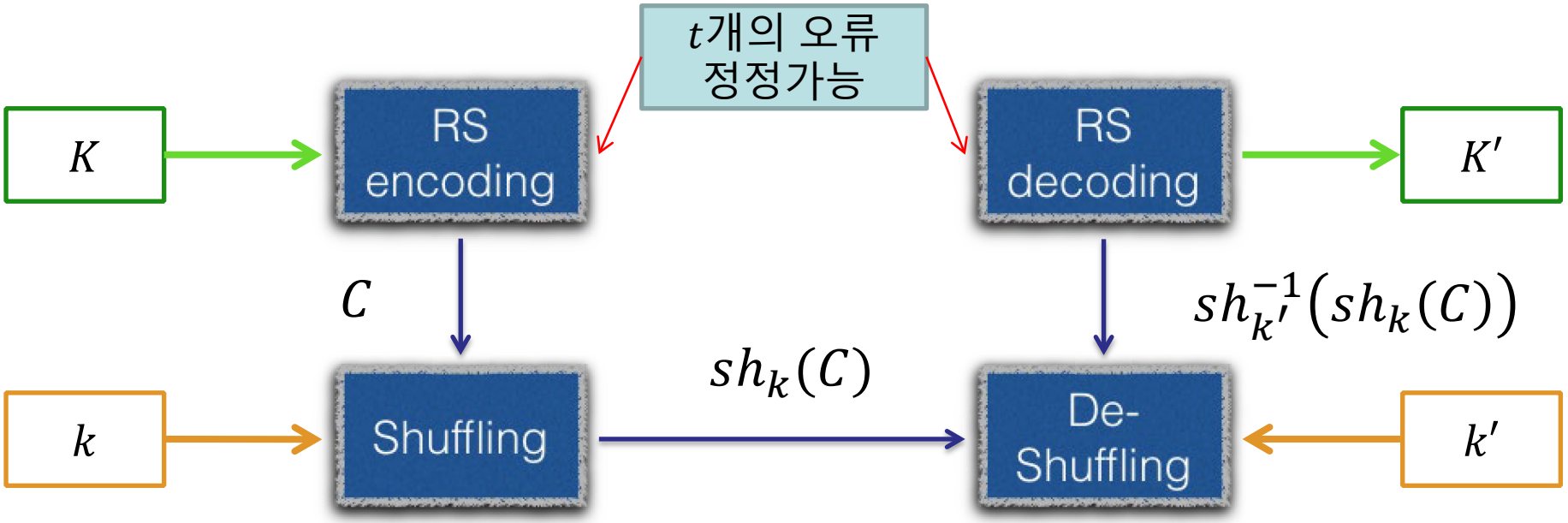


De-shuffling

- **Shuffling 의 역과정**



취약점 분석



- 만일 생체정보에 오류가 없다면,
  두 permutation sh_k와 $sh_{k'}$에 대해 $d(sh_{k'}, sh_k) \leq t$이면

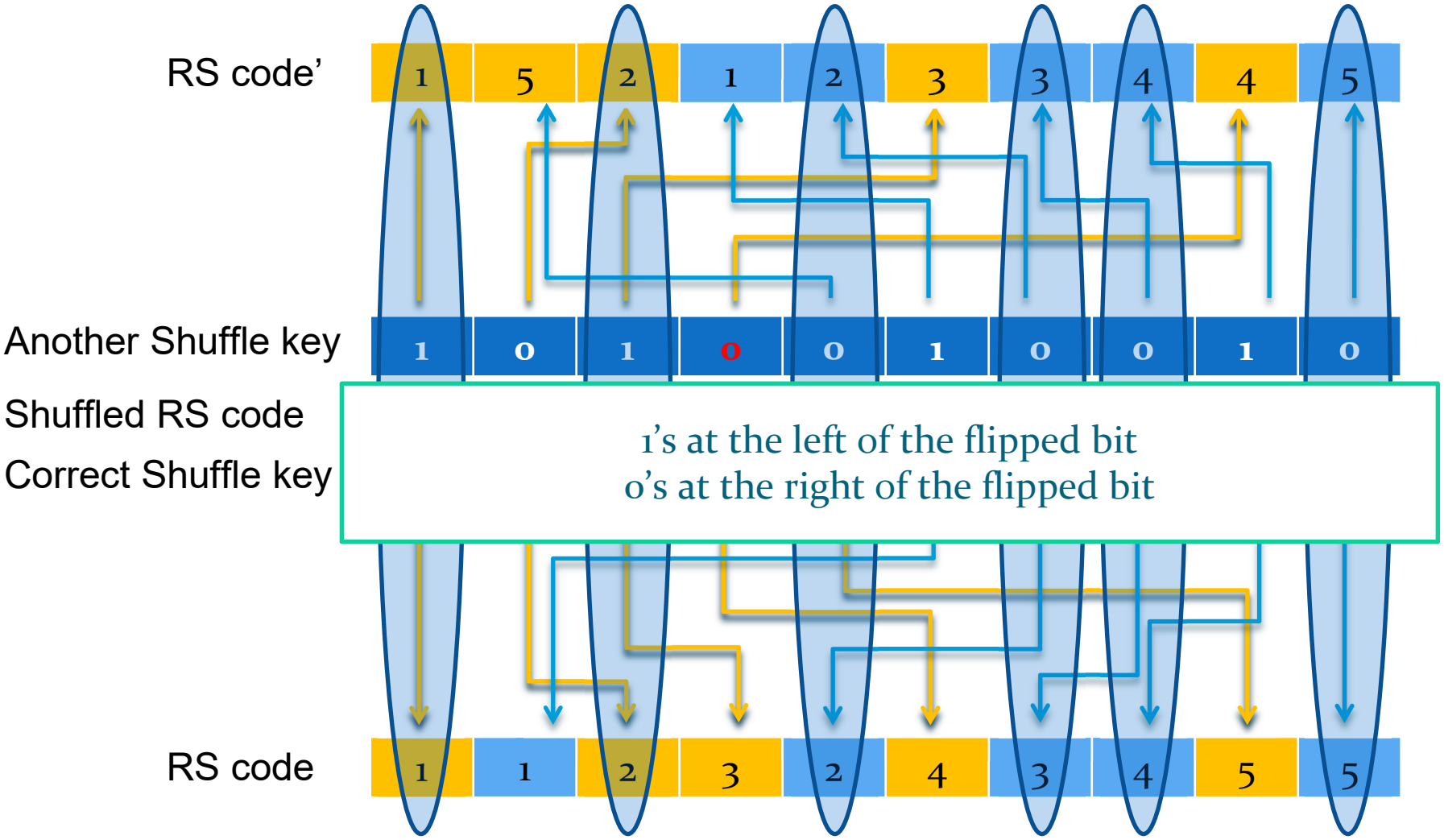
$$d\left(C, sh_{k'}^{-1}(sh_k(C))\right) \leq t$$

  이므로 $K = K'$가 된다.

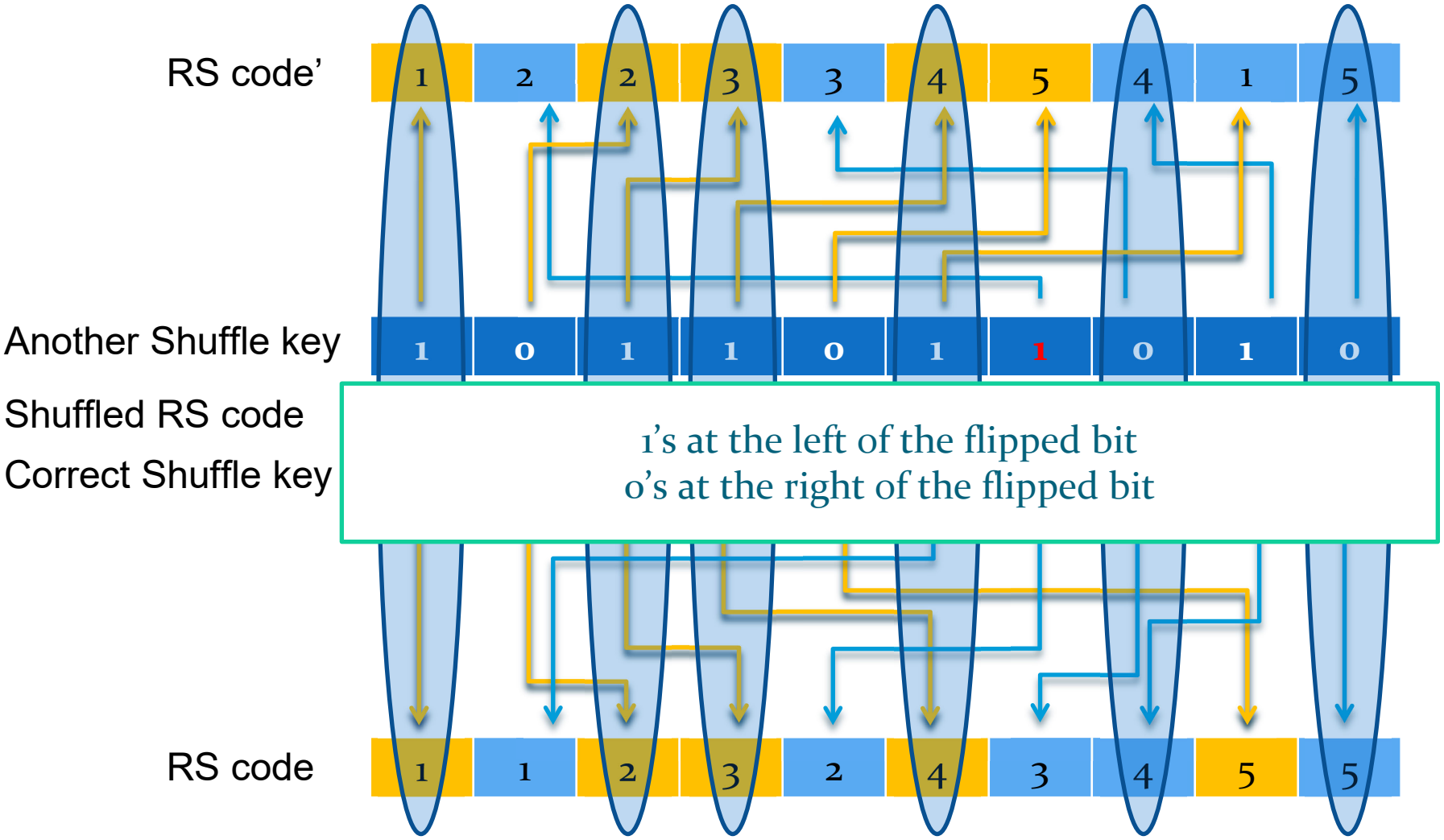
t-유사키 (t-similar key)

- 거리가 t 이하인 permutation을 만드는 두 개의 키 k, k'
- Shuffle 함수의 경우
  - 만들어지는 sh_k 들의 거리가 매우 가까움
  - 매 키마다 최소 $2^{\lfloor t/2 \rfloor}$개의 유사키 존재
- 키 사용으로 얻는 안전성이 최소 $t/2$비트 감소
- 서로의 거리가 t보다 큰 permutation들이 필요

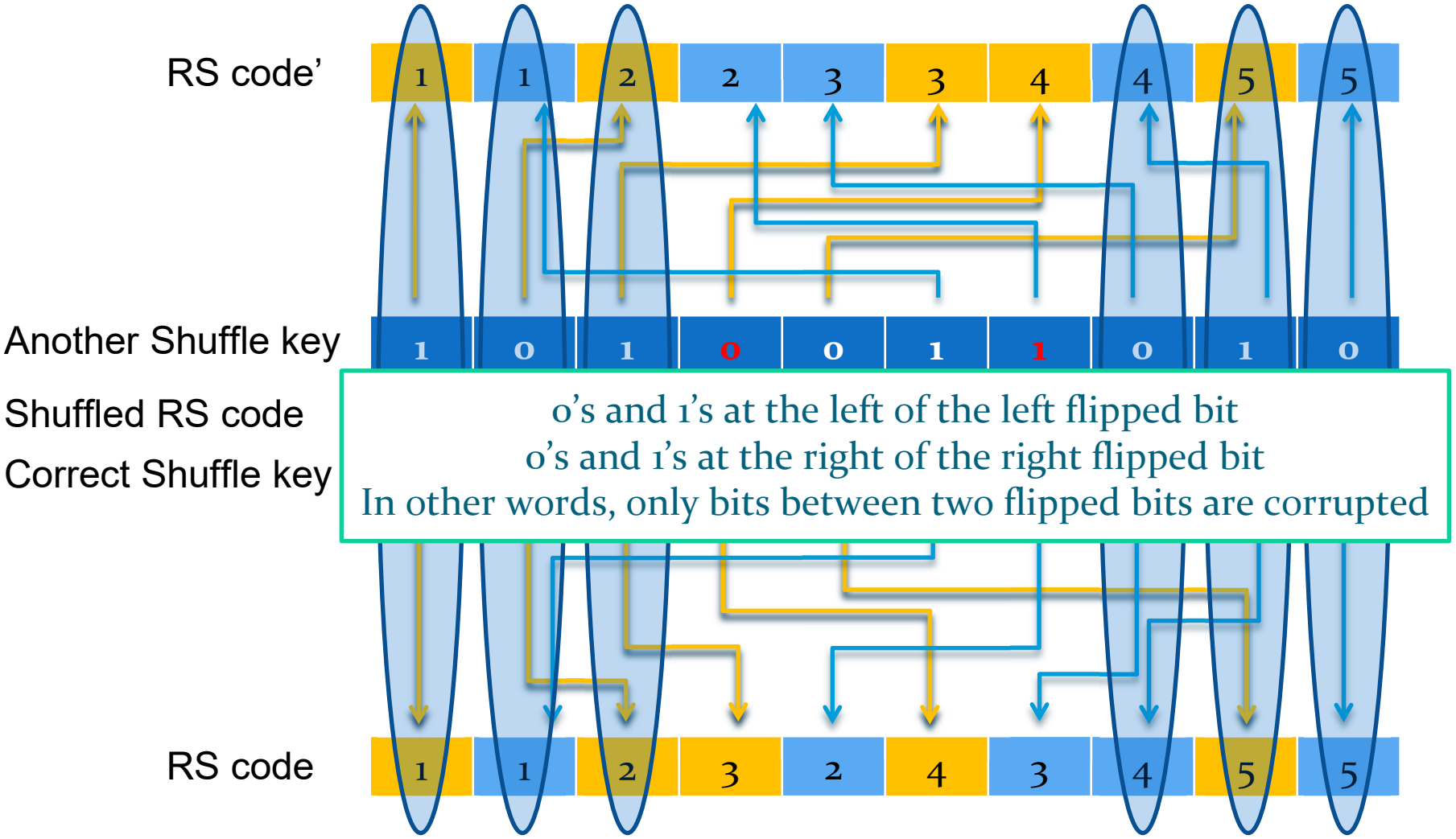
한 비트 뒤집기 : 1 → 0


RS code'
1 5 2 1 2 3 3 4 4 5
Another Shuffle key
1 0 1 0 0 1 0 0 1 0
Shuffled RS code
Correct Shuffle key
1's at the left of the flipped bit
0's at the right of the flipped bit
RS code
1 1 2 3 2 4 3 4 5 5

한 비트 뒤집기 : 0 → 1

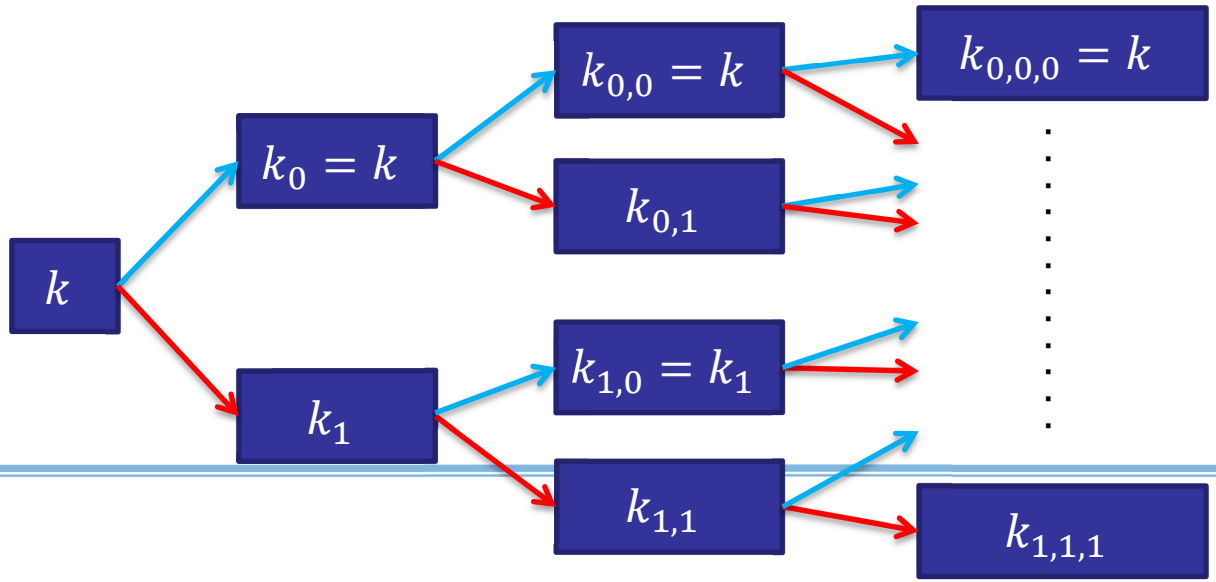

RS code'
1 2 2 3 3 4 5 4 1 5
Another Shuffle key
1 0 1 1 0 1 1 0 1 0
Shuffled RS code
Correct Shuffle key
1's at the left of the flipped bit
0's at the right of the flipped bit
RS code
1 1 2 3 2 4 3 4 5 5

두 비트 한 번에 뒤집기 : 한 쌍으로



t-유사키 생성 : Shuffle함수의 경우

- **1- 또는 2- 유사키 얻기**
  - 만일 키 k에 10 또는 01이라는 연속된 두 비트가 있다면
    - ✓ 해당 두 비트를 01 또는 10으로 뒤집어서 2-유사키 k'을 얻음
  - 만일 k가 000…000 또는 111…111꼴이면
    - ✓ k' = 100…000 또는 111…110 는 k의 1-유사키
- **t-유사키 얻기**
  - k_1이 k_0의 n_1-유사키, k_2가 k_1의 n_2-유사키이면 k_2는 k_0의 $(n_1 + n_2)$-유사키



문제점, 해결책, 문제점

- **문제점**
  - shuffle 함수를 사용하면, n비트 사용자키를 사용하지만 $n - t/2$ 비트 키를 사용하는 것과 다를 바 없음
  - 공격에 취약해짐
- **해결책**
  - 모든 permutation들이 최소 거리 $t + 1$을 가지면 됨
  - $PA(n, t + 1)$
- **다시 문제점**
  - $n = 49, t = 13$
  - 2^{49}개 이상의 row를 가지는 $PA(49,14)$의 construction?

Permutation arrays…

- **2^{49}개 이상의 row를 가지는 $PA(49,14)$의 construction**
  - 현재 생략
  - IWSDA11 등에서 발표한 자료 재활용 가능

t-유사키의 한계

- $\boldsymbol{d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right) \leq d(sh_{k'}, sh_k)}$
  - 따라서 $\boldsymbol{d(sh_{k'}, sh_k) \leq t}$이면 $\boldsymbol{d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right) \leq t}$
    즉, t-유사키는 위험함
  - 그러나 $\boldsymbol{d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right)} \leq t < \boldsymbol{d(sh_{k'}, sh_k)}$일 수 있음
    - ✓ $t < \boldsymbol{d(sh_{k'}, sh_k)}$ 을 만족하여 t-유사키는 없지만
    - ✓ $\boldsymbol{d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right)} \leq t$ 이어서 k'이 k의 역할을 할 수 있게 됨
  - 이러한 경우는 C에 의존적으로 발생
    - ✓ C의 모든 블록이 다르면
      $$\boldsymbol{d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right) = d(sh_{k'}, sh_k)}$$
    - ✓ 따라서 C의 블록 중 일부가 같으면 발생한다

t-유사키의 한계

- **C의 모든 블록이 서로 다른 경우는 거의 없음**
  - 최소 한 쌍의 같은 블록이 존재할 확률은,
    Birthday paradox $(n = 49, l = 128)$에 의해
    $$1 - \frac{l!}{l^n \cdot (l-n)!} \approx 0.999976$$
  - 실제 무작위추출로 실험해보면,
    Reed-Solomon code의 경우 블록이 모두 다를 확률은 약 0.002%
- **따라서 많은 경우**
  $$d\left(C, sh_{k'}^{-1}\left(sh_k(C)\right)\right) < d(sh_{k'}, sh_k)$$
- **$PA(n, t+1)$를 사용해도 높은 확률로 k'이 k의 역할 수행 가능**
- **Q) 실제로 그러한가?**

실험 : PA의 경우 t-유사키의 안전성

- **10^{12}개의 triple (C, k_1, k_2)에 대해 (C는 RS codeword)**
  - $PA(49,14)$를 이용하여 두 키 k_1, k_2에 대응하는 두 permutation p_{k_1}, p_{k_2}를 선택하고, $d\left(p_{k_1}(C), p_{k_2}(C)\right)$를 계산
  - 각 d에 대해 $d\left(p_{k_1}(C), p_{k_2}(C)\right) = d$인 triple의 비율 (%)

d	비율	d	비율	d	비율	d	비율	d	비율
0	0	10	0	20	0	30	0.00	40	0.00
1	0	11	0	21	0.00	31	0.00	41	0.02
2	0	12	0	22	0	32	0.00	42	0.05
3	0	13	0	23	0	33	0.00	43	0.25
4	0	14	0	24	0.00	34	0.00	44	1.05
5	0	15	0	25	0	35	0.00	45	3.80
6	0	16	0	26	0.00	36	0.00	46	11.04
7	0	17	0	27	0.00	37	0.00	47	23.95
8	0	18	0	28	0.00	38	0.00	48	34.48
9	0	19	0	29	0.00	39	0.00	49	25.33

실험결과 분석

- **RS codeword C의 모든 블록이 서로 다른 경우는 거의 없지만**
  - $PA(49,14)$를 이용하면

$$d\left(p_{k_1}(C), p_{k_2}(C)\right) \leq 14$$

    인 경우가 거의 없음
  - 따라서 크게 문제가 되지 않는 것으로 생각됨
  - (주: 이론적으로 이 사실을 보일 수 있을까?)
  - (주: 실험을 어떻게 다시 할 것인가?)

기타 사항 : PA가 shuffle보다 좋은 이유

- **Biometric Scheme에서 사용된 permutation들은**
  - Kanade 등의 주장에 따르면 이 permutation들은 일종의 encryption처럼 작동해야 함
    - ✓ 패스워드(키) k를 사용하여 추가적인 안전성을 보장받고자 함
  - 그러기 위해서는 Pseudo-random permutation이어야 함

- **F가 a family of pseudo-random permutations라면**
  - 임의의 n비트 binary string B와 $f \in F$인 permutation f에 대해,

$$\Pr\big(d\big(B, f(B)\big) = 2\epsilon\big) = \frac{1}{2^n}\binom{n+1}{2\epsilon+1}$$

    을 만족해야 함 (필요조건)

- **PA의 경우는 위 식을 만족하나, shuffle은 만족하지 못한다.**
  - PA가 Kanade 등의 의도에 더 부합하는 permutation들임

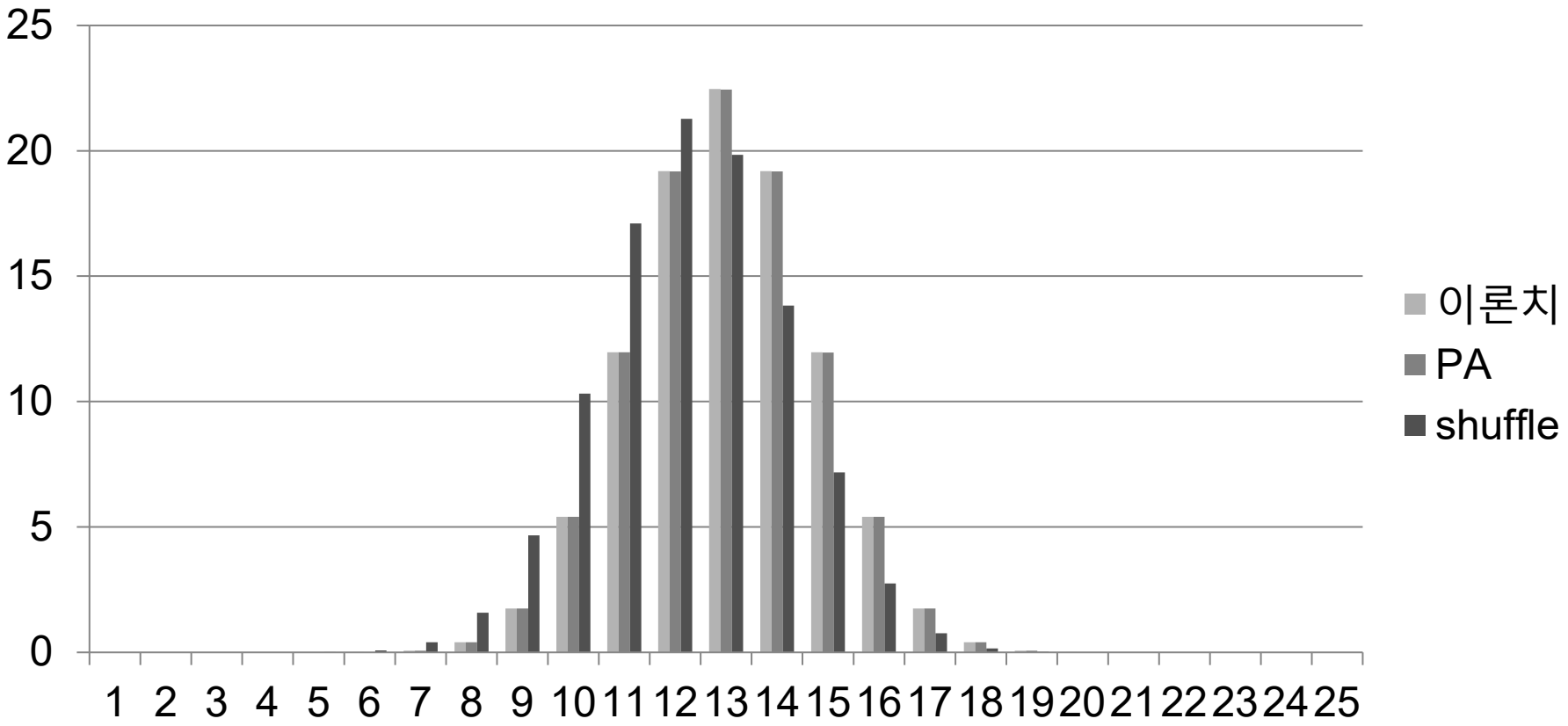
기타 사항 : PA가 shuffle보다 좋은 이유

- **10^{12}개의 pair (B, f)에 대해 $\Pr\left(d\left(B, f(B)\right) = 2\epsilon\right)$를 계산**
  - f는 각각 무작위 키를 생성하여 shuffle과 PA에서 추출함

ϵ	이론치	shuffle	PA	ϵ	이론치	shuffle	PA
0	0.00	0.00	0.00	13	19.19	13.83	19.18
1	0.00	0.00	0.00	14	11.96	7.18	11.95
2	0.00	0.00	0.00	15	5.40	2.74	5.40
3	0.00	0.00	0.00	16	1.75	0.76	1.75
4	0.00	0.01	0.00	17	0.40	0.15	0.40
5	0.01	0.08	0.00	18	0.06	0.02	0.06
6	0.06	0.40	0.06	19	0.01	0.00	0.00
7	0.40	1.58	0.40	20	0.00	0.00	0.00
8	1.75	4.66	1.75	21	0.00	0.00	0.00
9	5.40	10.31	5.40	22	0.00	0.00	0.00
10	11.96	17.10	11.96	23	0.00	0.00	0.00
11	19.19	21.27	19.18	24	0.00	0.00	0.00
12	22.46	19.84	22.44				

기타 사항 : PA가 shuffle보다 좋은 이유

- **10^{12}개의 pair (B, f)에 대해 $\Pr\left(d\left(B, f(B)\right) = 2\epsilon\right)$를 계산**
  - f는 각각 무작위 키를 생성하여 shuffle과 PA에서 추출함



결론

- Biometric Encryption에 대한 설명 (논문에 없음)
- Kaneda 등이 제안한 Fuzzy Committment Scheme의 보안성 분석
- 특히 shuffling 방식의 문제점 파악
- 새로운 shuffle 방식 제안