제 III부

키, 난수, 응용 기술

박 종 혁

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

제 11장 키

비밀의 핵심

11.1 주요 내용

- 키란 무엇인가?
- 다양한 키
- 키 관리
- Diffie-Hellman 키 교환
- 패스워드를 기초로 한 암호(PBE)
- 안전한 패스워드를 만들려면?

11.2 키란 무엇인가

- 암호 알고리즘을 조정하는 정보 조각
- 평문을 암호문으로 전환
- 암호문을 복호화하는 역할
- 디지털 서명 구조
- 키를 이용한 해시 함수
- 인증

11.2.1 키는 대단히 큰 수

- 암호 기술을 사용하려면 반드시 **키**(key)라 불리는 대단히 큰 수가 필요

- 중요한 것은 수 그 자체의 크기보다도 **키 공간**(Key Space)의 **크기**, 즉 「가능한 키의 총수」
  - 키 공간이 크면 클수록 전사공격이 힘들다
  - 예: 키의 길이가 10비트이면 키의 종류는 총 2^{10} = 1,024개로서 키 공간의 크기가 1,024이다.
  - DES에서 사용하는 키의 길이가 64비트이므로 키 공간의 크기는 2^{64}

DES의 키

- 실질적으로 56비트 길이(7바이트 길이)
- 2진수로 표기된 DES 키의 예

  01010001 11101100 01001011 00010010
  00111101 01000010 00000011

- 16진수로 표기된 DES 키의 예

  51 EC 4B 12 3D 42 03

트리플 DES의 키

- DES-EDE2
  - DES의 키를 2개 사용
  - DES의 키 길이의 2배인 112비트 길이
  - 예:

  51 EC 4B 12 3D 42 03 30 04 D8 98 95 93 3F
- DES-EDE3
  - DES의 키를 3개 사용
  - DES의 키 길이의 3배인 168비트 길이
  - 예:

  51 EC 4B 12 3D 42 03 30 04 D8 98 95 93 3F 24 9F
  61 2A 2F D9 96

AES의 키

- 대칭 암호 AES의 키의 길이
  - 128, 192, 256 비트
  - 예

51 EC 4B 12 3D 42 03 30 04 D8 98 95 93 3F 24 9F
61 2A 2F D9 96 B9 42 DC FD A0 AE F4 5D 60 51 F1

11.2.2 키와 평문의 가치

- 암호문을 도청한 이브에게 「키가 넘어가는 것」은 「평문이 넘어가는 것」과 같은 것이다.
- 키는 평문과 같은 가치를 갖는다.

11.2.3 암호 알고리즘과 키

- Kerckhoffs 의 원리
- 안전한 시스템을 설계할 때는 사용하는 암호 알고리즘이 공격자들에게 알려져 있다는 것을 전제로 해야 한다.
- 즉,「**오직 키를 비밀로 하는 것만이 안전을 보장한다**」
- 「공격자는 시스템에 대해 알고 있다」는 것이다.

11.3 다양한 키

- 한 마디로 「키」라고 표현하기는 하지만 키에는 많은 종류가 있다.

11.3.1 대칭 암호의 키와 공개 키 암호의 키

- 대칭 암호
  - 암호화와 복호화에서 동일한 키 사용
  - 비밀키
- 공개 키 암호
  - 암호화와 복호화에서 다른 키를 사용
  - 개인키, 공개키

대칭 암호는 암호화와 복호화에서 공통의 키를 사용한다

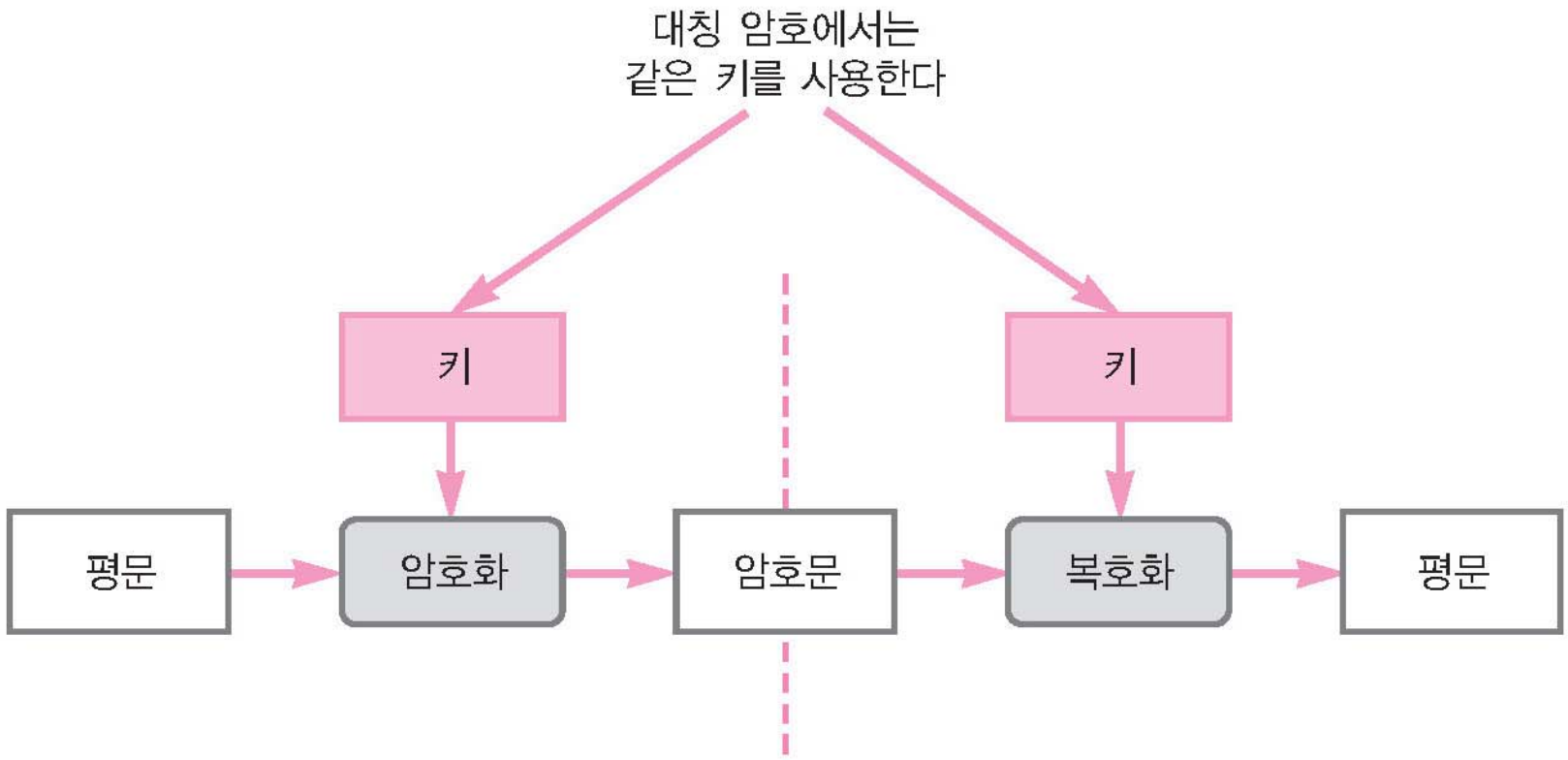

그림 11-1 대칭 암호는 암호화와 복호화에서 공통의 키를 사용한다

공개 키 암호는 공개 키로 암호화하고, 개인 키로 복호화한다

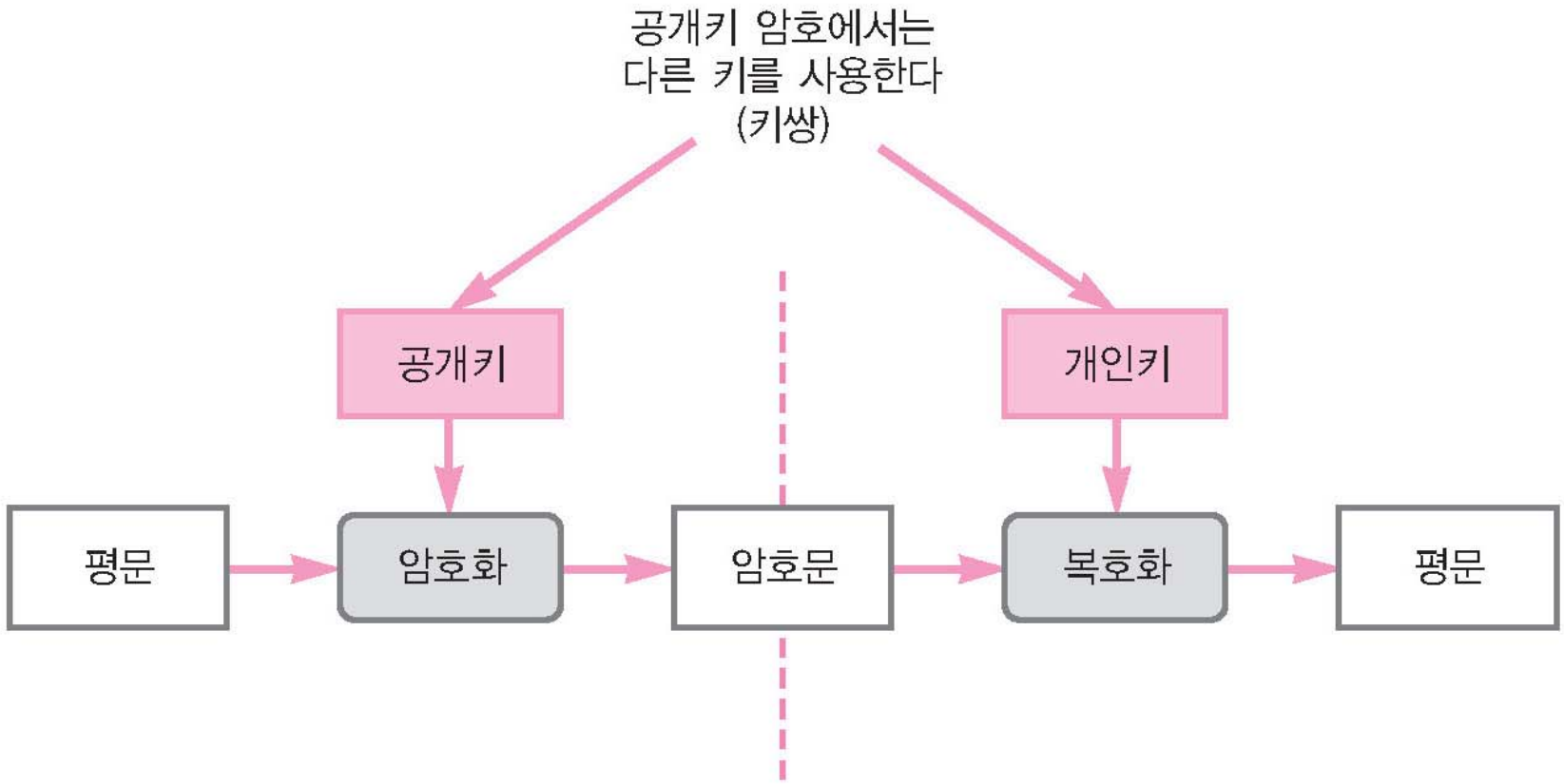

그림 11-2 공개 키 암호는 공개 키로 암호화하고, 개인 키로 복호화한다

11.3.2 메시지 인증 코드의 키와 디지털 서명의 키

- 메시지 인증 코드
  - 송신자와 수신자가 공통키를 사용해서 인증수행
- 디지털 서명
  - 서명작성과 서명검증에 서로 다른 키를 사용

메시지 인증 코드의 키

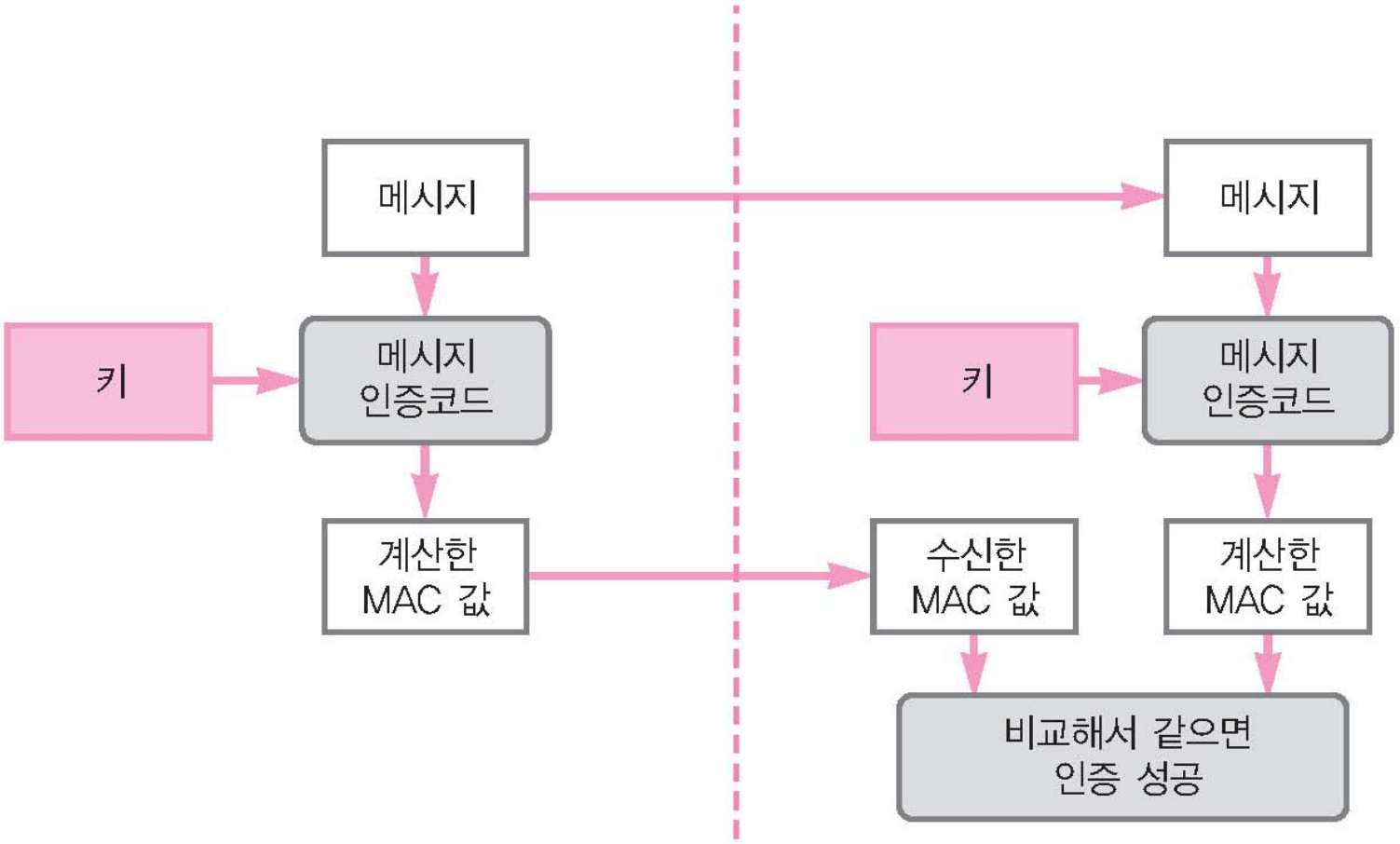

그림 11-3 메시지 인증 코드의 키

디지털 서명 키

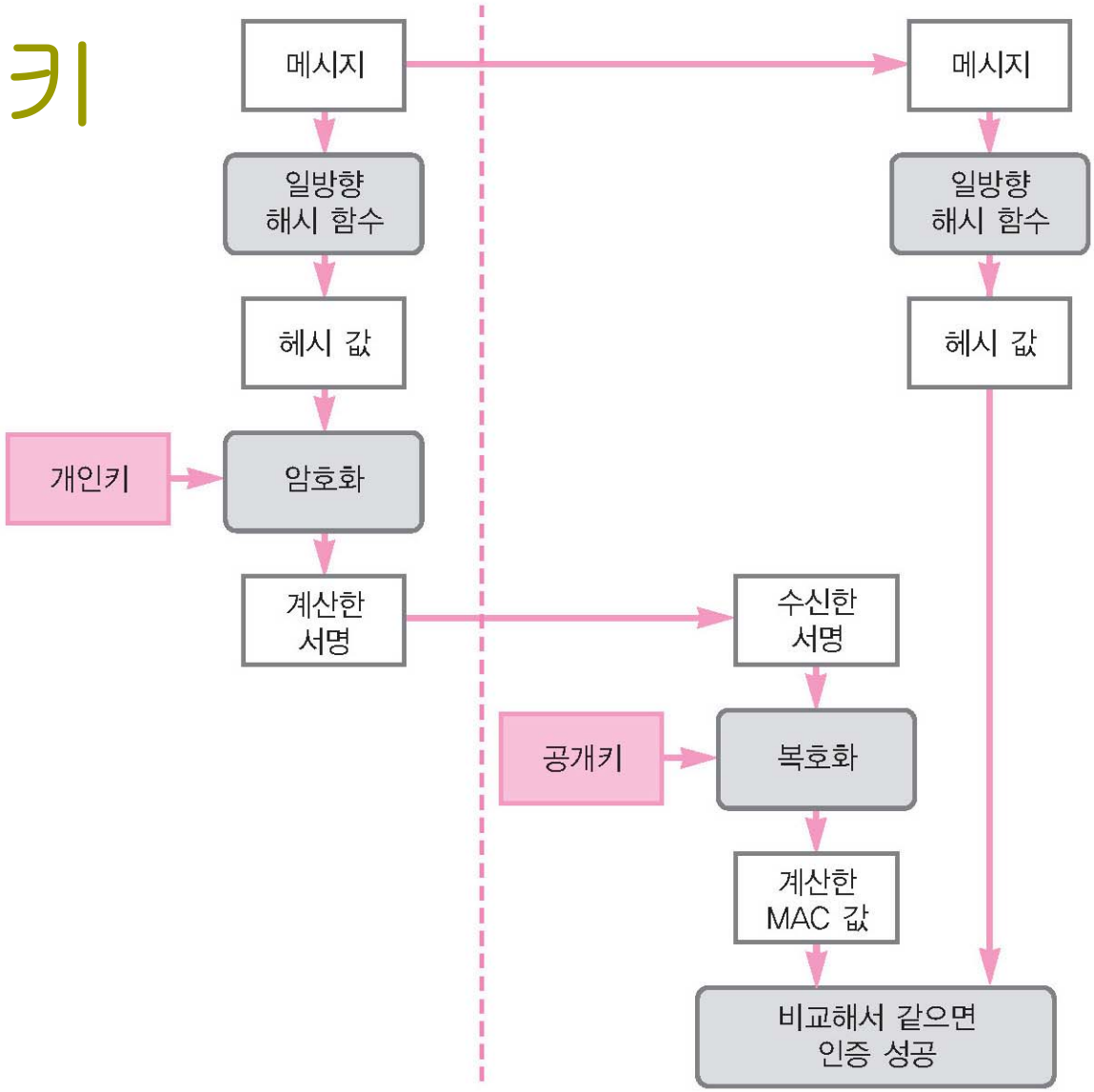

그림 11-4 디지털 서명 키

11.3.3 기밀성을 위한 키와 인증을 위한 키

- 기밀성
  - 대칭 암호나 공개 키 암호에서 사용하는 키
  - 복호화를 위한 바른 키를 모르는 사람은 평문의 내용을 알 수 없다.
- 인증
  - 메시지 인증 코드
  - 디지털 서명
  - 바른 키를 모르는 사람은 데이터를 수정하거나 거짓 행세를 할 수가 없다.

11.3.4 세션 키와 마스터 키

- 키가 사용되는 횟수에 주목
- 세션 키(session key)
  - 통신 때마다 한 번만 사용되는 키
  - 세션 키가 알려졌다고 하더라도 해독되는 것은 그 통신 1회분뿐임
  - 다음 통신에서는 또 다른 세션 키가 사용
- 마스터 키(master key)
  - 반복적으로 사용되는 키

11.4 콘텐츠를 암호화하는 키와 키를 암호화하는 키

- 암호화하는 대상에 주목
- 암호화의 대상
  - 통상적으로 사용자가 직접 이용하는 정보(콘텐츠)
- CEK(contents encrypting key)
  - 정보의 암호화에 사용하는 키
- KEK(key encrypting key)
  - 키를 암호화하는 키

콘텐츠를 암호화하는 키(CEK)와 키를 암호화하는 키(KEK)

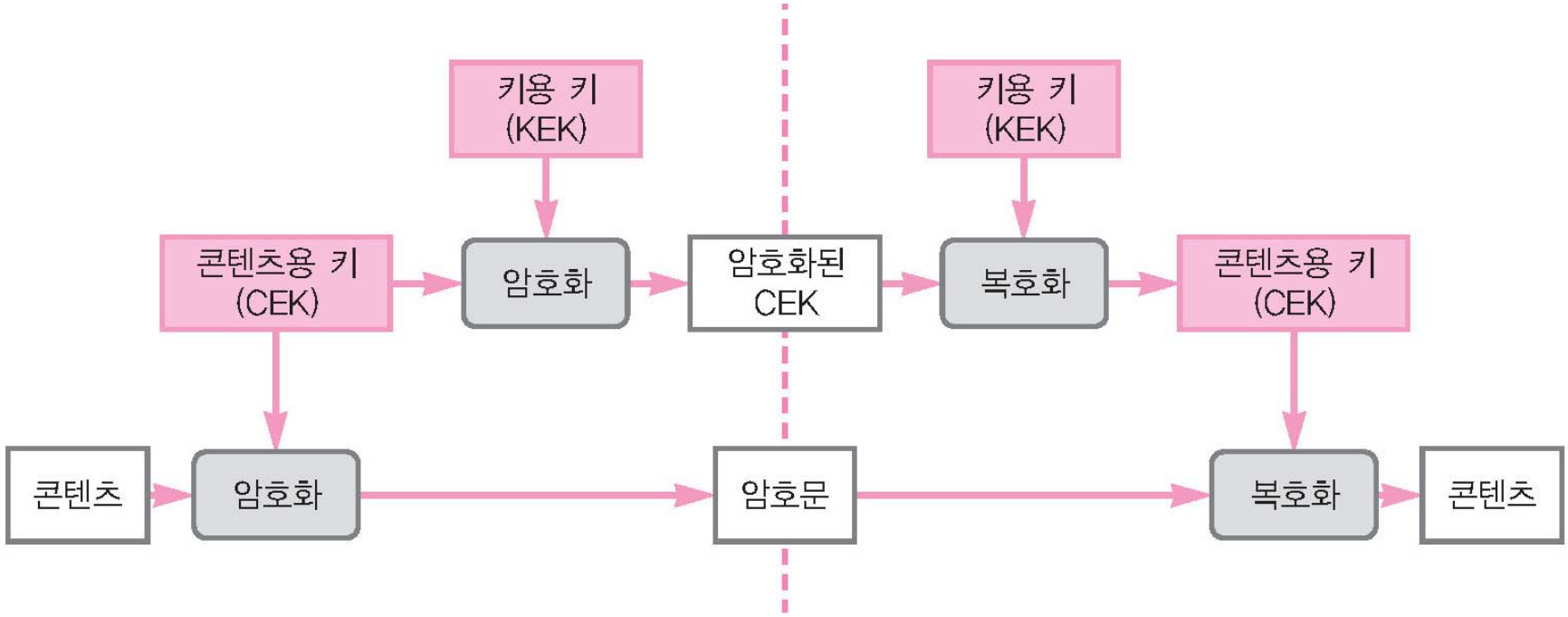

그림 11-5 콘텐츠를 암호화하는 키(CEK)와 키를 암호화하는 키(KEK)

11.5 키 관리

11.5.1 키 생성

- 랜덤하게 만들어져야만 한다
- 랜덤하게 키를 만든다는 것은 매우 어려운 일
- 하드웨어적 난수생성기 이용
- 주사위를 이용
- 패스워드를 이용

난수를 이용한 키 생성

- 암호용 의사난수 생성기를 활용한다
- 암호용으로 설계되어 있지 않은 의사난수 생성기를 이용하여 키를 생성해서는 안 된다
  - 「예측 불가능」이라는 성질을 갖지 않기 때문

패스워드를 이용한 키 생성

- **패스워드**(password) 혹은 **패스 프레이즈**(passphrase)로부터 키를 만든다
  - 패스워드를 일방향 해시 함수에 입력해서 얻어진 해시 값을 키로 이용
  - 패스워드에 **솔트**(salt)라 불리는 난수를 부가해서 일방향 해시 함수에 입력
    - 사전공격 방어를 위해
- 「패스워드를 기초로 한 암호」(password based encryption; PBE)라 한다

11.5.2 키 배송

- 키를 사전에 공유하는 방법
- 키 배포 센터를 이용하는 방법
- 공개 키 암호를 사용하는 방법
- 키 배송 문제를 해결하는 또 하나의 방법
  - Diffie-Hellman 키 교환

11.5.3 키 갱신

- 키 갱신(key updating)
  - 통신의 기밀성을 높이는 테크닉
  - 정기적으로 키를 교환
    - 예를 들면 1000문자 통신할 때마다
  - 현재 키의 해시 값을 다음 키로 사용

Backward security

- 적어도 키가 노출된 특정 시점 이전의 통신에 대해서는 안전하다는 것을 보증하는 것

- 특정 시점에 키가 도청자에게 알려졌다고 하자.
  - 그러면 도청자는 그 키를 사용해서 그 시점 이후의 통신을 복호화할 수 있다.
  - 키 갱신을 행한 시점보다도 과거로 거슬러 올라간 시점에서 이루어졌던 통신을 복호화할 수는 없다.
    - 이유는 **일방향 해시 함수의 역산이 불가능**

- Forward Security ?

11.5.4 키 보존

- 세션 키는 통신 1회에 한해 사용하는 것이므로 키보존이 필요없다
- 그러나 키를 반복해서 사용하는 경우에는 **키의 보존**을 생각하지 않으면 안 된다.
- 인간의 기억으로는 불가능하다
- 금고 등의 안전한 장소에 보관
- 키를 암호화하여 보존
  - KEK 키를 이용하여 키를 암호화하여 보존한다

키를 암호화하는 수법은

- 지켜야 할 키의 개수를 적게 할 수 있다
- 많은 키(CEK)의 기밀성을 지키는 대신에 1개의 키(KEK)의 기밀성을 지키는 것

인증기관의 계층화와 키 계층

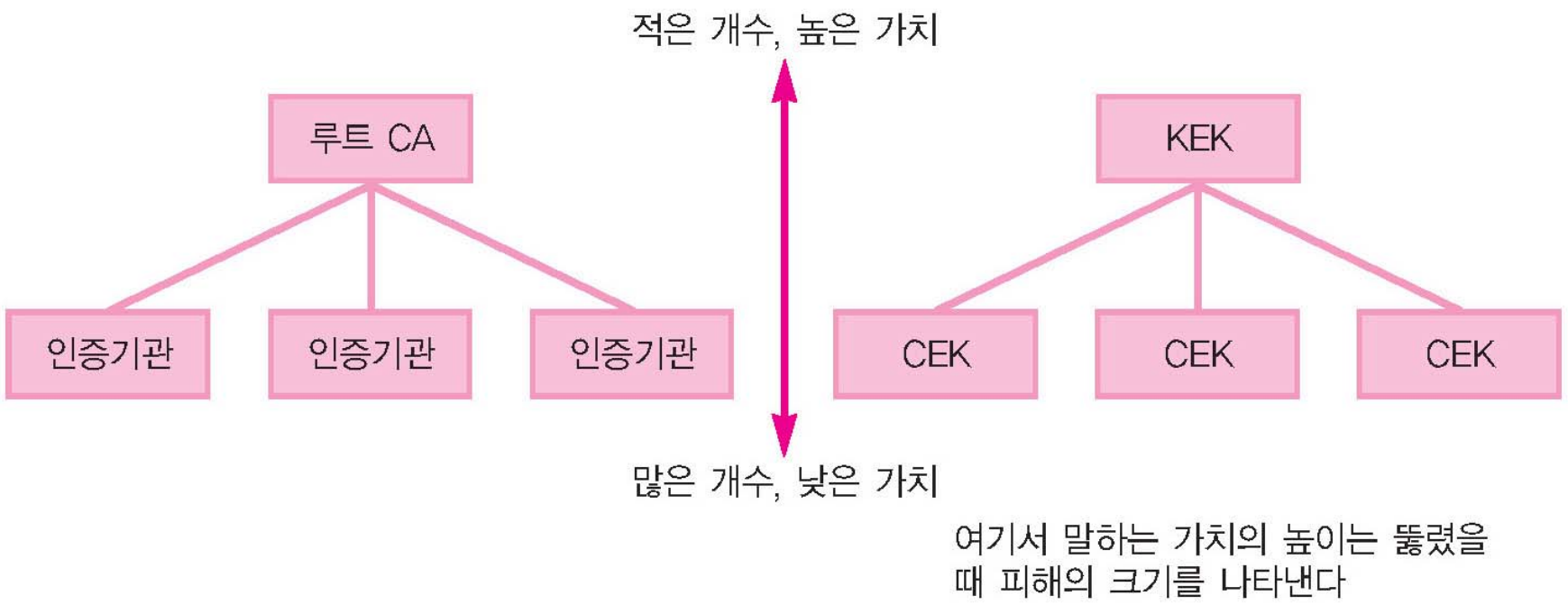

그림 11-6 인증기관의 계층화와 키 계층의 비교

11.5.5 키 폐기

- 불필요한 키는 폐기해야 한다
  - 만약 도청자 이브가 키를 입수하게 되면 이브가 보관하고 있던 암호 메일이 복호화 되기 때문
- 어떻게 버리는 것인가?
  - 컴퓨터 전체가 보안을 염두에 둔 설계 필요
- 키를 잃어버리면 어떻게 될까?
  - 공개 키 암호와 짝을 이루는 개인 키를 분실하면 안된다

11.6 Diffie-Hellman 키 교환

- 키 배송 문제를 해결하는 방법의 하나로서 Diffie-Hellman 키 교환을 소개한다.

11.6.1 Diffie-Hellman 키 교환

- 타인에게 알려져도 상관없는 정보를 두 사람이 교환하는 것만으로 공통의 비밀 값을 만들어내는 방법
- 만들어낸 비밀 값을 대칭 암호의 키로 사용

11.6.2 Diffie-Hellman 키 교환의 수순

- 앨리스는 밥에게 2개의 소수 P와 G를 송신한다
- 앨리스는 난수 A를 준비한다
- 밥은 난수 B를 준비한다
- 앨리스는 밥에게 G^A mod P라는 수를 송신한다
- 밥은 앨리스에게 G^B mod P라는 수를 송신한다
- 앨리스는 밥이 보내 온 수를 A제곱해서 mod P를 취한다

앨리스가 계산한 키

- 앨리스가 계산한 키

$$(G^B \bmod P)^A \bmod P$$

- 단순화 하면

$$G^{B \times A} \bmod P$$

$$= G^{A \times B} \bmod P$$

밥이 계산한 키

- 밥이 계산한 키

$$(G^A \bmod P)^B \bmod P$$

- 단순화 하면

$$G^{A \times B} \bmod P$$

- 따라서

앨리스가 계산한 키 = 밥이 계산한 키

11.6.3 이브의 키 계산

- 도청자 이브가 알 수 있는 것
  - 두 사람이 교환한 P, G, G^A mod P, G^B mod P
- 이 4개의 수로부터 ($G^{A\times B}$ mod P)라는 값을 계산하는 것은 수학적인 면에서 대단히 어렵다.
- 유한체상의 **이산대수문제이다**
  - 유한체상의 이산대수문제를 수학적으로 풀기 어렵다는 것이 Diffie-Hellman 키 교환을 뒷받침한다

11.6.5 구체적 키 교환의 예

- 앨리스는 밥에게 2개의 수 P=13 과 G를 송신한다
- 앨리스는 1부터 P-2까지의 정수로서 랜덤한 수 A=9 를 준비한다
- 밥은 1부터 P-2까지의 정수 중에서 랜덤한 수 B=7 을 준비한다
- 앨리스는 밥에게 $G^A \bmod P = 2^9 \bmod 13 = 5$ 를 송신
- 밥은 앨리스에게 $G^B \bmod P = 2^7 \bmod 13 = 11$ 을 송신

앨리스가 계산한 키

- 앨리스는 밥이 보내 온 수 11을 A(=9) 제곱해서 P로 mod를 취한다

$$(G^B \bmod P)^A \bmod P$$
$$= 11^A \bmod P$$
$$= 11^9 \bmod 13$$
$$= 8$$

밥이 계산한 키

- 밥은 앨리스가 보내 온 수 5를 B(=7) 제곱해서 P로 mod를 취한다

$$(G^A \bmod P)^B \bmod P$$
$$= 5^B \bmod P$$
$$= 5^7 \bmod 13$$
$$= 8$$

11.7 패스워드를 기초로 한 암호 (PBE)

- 전사공격을 무력화시키기 위해서
  - 패스문(Passphrases)을 이용
  - PBKDF2(RFC 2898)같은 반복되는 패스워드 기반 키 변형함수를 사용

11.7.1 패스워드를 기초로 한 암호

- 공개 키 암호에 대한 **중간자 공격**은 디지털 서명에도 위협이 되는 공격이다.

- 디지털 서명의 중간자 공격은 적극적 공격자 맬로리가 송신자와 수신자의 사이에 들어가 송신자에 대해서는 수신자처럼, 수신자에 대해서는 송신자처럼 거짓 행세를 하는 공격이다.

- 이것은 디지털 서명의 알고리즘 자체를 깨지 않아도 가능하다.

PBE의 의의

- 중요한 메시지의 기밀성을 유지하고 싶다.
  ↓
- 메시지를 그대로 디스크에 보존하면 누군가에게 읽혀질지도 모른다.
  ↓
- 키(CEK)를 사용해서 메시지를 암호화하자.
  ↓
- 하지만 이번에는 키(CEK)의 기밀성을 유지하지 않으면 안된다.
  ↓
- 키(CEK)를 그 대로 디스크에 보존하는 것은 위험하다.
  ↓
- 다른 키(KEK)를 사용해서 키(CEK)를 암호화하자.
  ↓

PBE의 의의

- 그렇지만 이번에는 키(KEK)의 기밀성을 유지하지 않으면 안 된다. 이래 가지고는 빙빙 맴도는 것에 지나지 않는다.
  ↓
- 그럼 키(KEK)는 패스워드로부터 만들기로 하자.
  ↓
- 패스워드만으로 만들면 사전 공격을 받을 위험이 있다.
  ↓
- 그렇다면 키(KEK)는 솔트와 패스워드로부터 만들기로 하자.
  ↓
- 「솔트」는 암호화한 키(CEK)와 함께 보존해 두고, 키(KEK)는 버리기로 하자.
  ↓
- 「패스워드」는 자신의 머릿속에 보존해 두기로 하자.

11.7.2 PBE의 암호화

(1) KEK의 생성

(2) 세션 키의 생성과 암호화

(3) 메시지의 암호화

PBE의 암호화

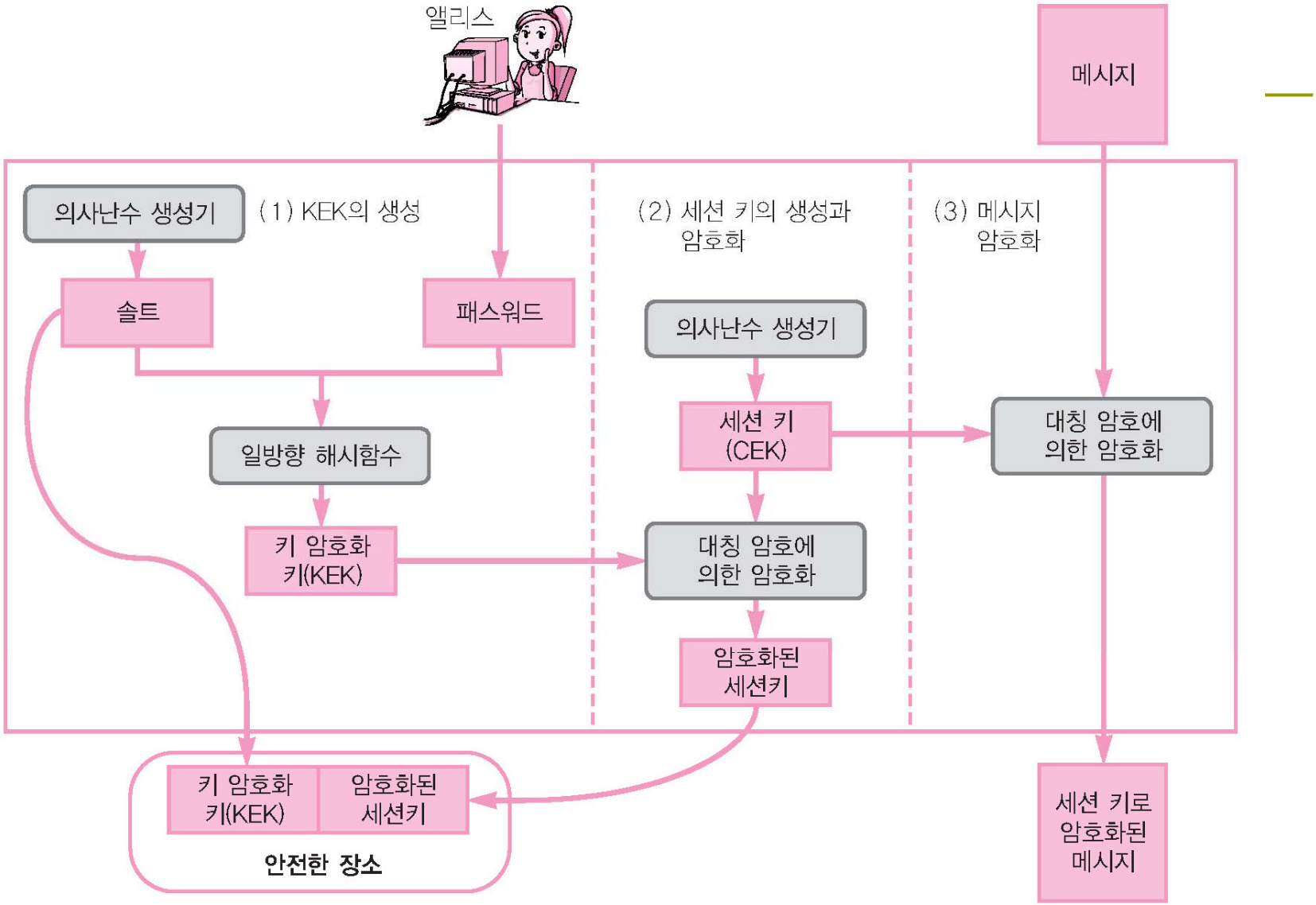

그림 11-8 PBE의 암호화

11.7.3 PBE의 복호화

(1) KEK의 복원

(2) 세션 키의 복호화

(3) 메시지의 복호화

PBE의 복호화

- 복호화에 의사난수 생성기는 등장하지 않는다

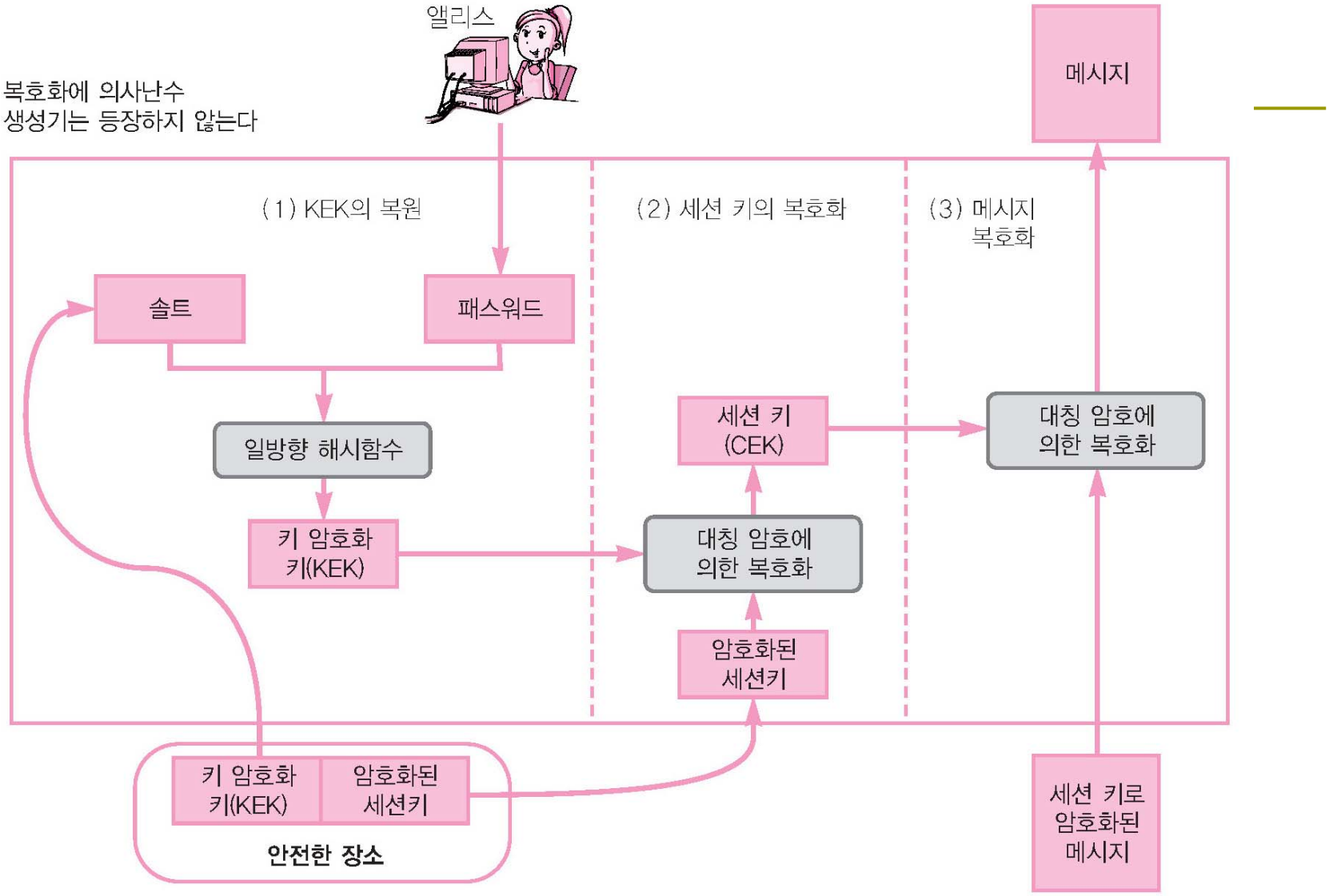

그림 11-9 PBE의 복호화

11.7.4 솔트의 역할

- 솔트(Salt)는 사전 공격을 막기 위해 있다.
- KEK를 만들 때에 솔트를 사용하지 않았다고 가정하면
  - 적극적 공격자 맬로리는 사전 데이터 등을 기초로 해서 KEK의 후보를 미리 대량으로 만들어 두는 것이 가능하다.

사전 공격(dictionary attack)

- 맬로리는 암호화된 세션 키를 훔친 다음 복호화를 시도한다
- KEK의 후보들을 미리 만들어 둠으로써 시도해보는 시간을 대폭 단축할 수가 있다.
- 이것이 **사전 공격**(dictionary attack)이다.

솔트(salt)의 역할은?

- 솔트를 사용하면,
  - 가능한 KEK 후보들의 종류 수가 솔트의 비트 길이만큼 늘어난다
  - 따라서 그 수가 방대해지기 때문

  ➔ KEK의 후보를 미리 만들어 놓는다는 것이 매우 어렵다.

11.7.5 패스워드의 역할

- 충분한 길이를 갖는 키는 기억할 수 없다
- 패스워드도 마찬가지로 우리들은 충분한 비트 수를 갖는 패스워드를 기억하지 못한다.
- 패스워드를 기초로 한 암호(PBE)를 이용하는 경우
  - 솔트와 암호화한 CEK를 **물리적으로 지키는 방법을 병용**해야 한다.

11.7.6 PBE의 개선

- KEK를 만들 때 일방향 해시 함수를 여러 번 통과하도록 하면 안전성을 높일 수 있다
  - 사용자 입장에서 해시 함수를 1000회 반복하는 것은 큰 부담이 되지 않는다
  - 적극적 공격자 맬로리에게는 작은 차이가 큰 부담이 된다.
  - 맬로리는 바른 KEK를 찾을 때까지 대량의 패스워드를 시도하지 않으면 안 되기 때문이다.

11.8 안전한 패스워드

- 안전한 패스워드를 만들어내는 힌트
  - 자신만이 알 수 있는 정보를 사용할 것
  - 복수의 패스워드를 나누어 쓸 것
  - 메모를 유효하게 사용할 것
  - 패스워드의 한계를 알 것

11.8.1 자신만이 알 수 있는 정보 사용

- 중요한 것의 이름을 사용해서는 안 된다
- 자신에 관한 정보를 사용해서는 안 된다
- 타인이 보기 쉬운 정보를 사용해서는 안 된다

11.8.2 복수 패스워드의 사용

- 정보의 가치에 따라 패스워드를 나누어 써야 한다
- 좋지 않은 사례
  - 회사 컴퓨터의 로그인용 tUniJww1
  - 집 컴퓨터의 로그인용 tUniJww2
  - 메일의 디지털 서명용 tUniJww3
  - 온라인 쇼핑용 tUniJww4

11.8.3 메모의 유효한 사용

- 의사난수 생성기로 만든 랜덤한 문자열을 패스워드로 하고, 그 패스워드를 쓴 메모를 안전하게 보존하는 것이 더 좋을 수 있다.

11.8.4 패스워드의 한계

- 패스워드가 영어 알파벳과 숫자열중의 8문자로 한정되어 있다고 가정해보면
- 영어 알파벳과 숫자 8문자로 된 문자열의 가능성을 계산해보면

$$62\times 62\times 62\times 62\times 62\times 62\times 62\times 62$$
$$= 62^8$$
$$= 218340105584896$$

- 이정도의 길이는 전사공격이 가능한 길이이다.
- 만약 적극적 공격자의 컴퓨터가 1초간에 1억 개의 패스워드를 만들어서 시험할 수 있다면, 약 25일에 모든 패스워드를 체크할 수 있다는 것이 된다.

질의 및 응답

– 끝–