Authentication

2025.08

자동차융합대학

GENERAL MOTORS
GM TECHNICAL CENTER KOREA

KMU 국민대학교
KOOKMIN UNIVERSITY

CONTENTS

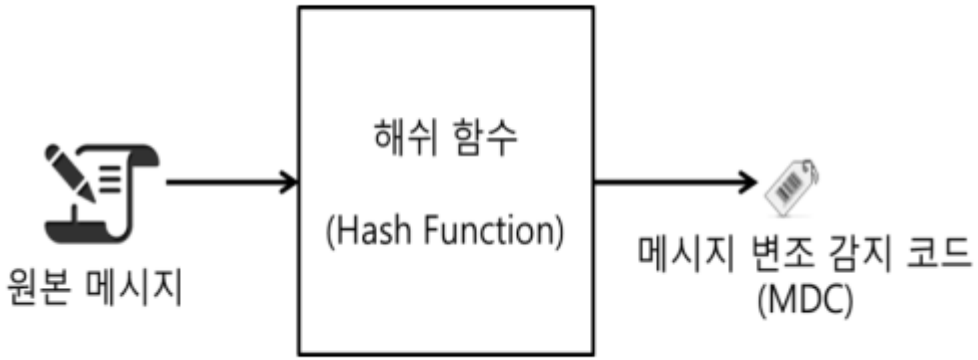
01

Hash Function

Message Integrity

■ 메시지 무결성

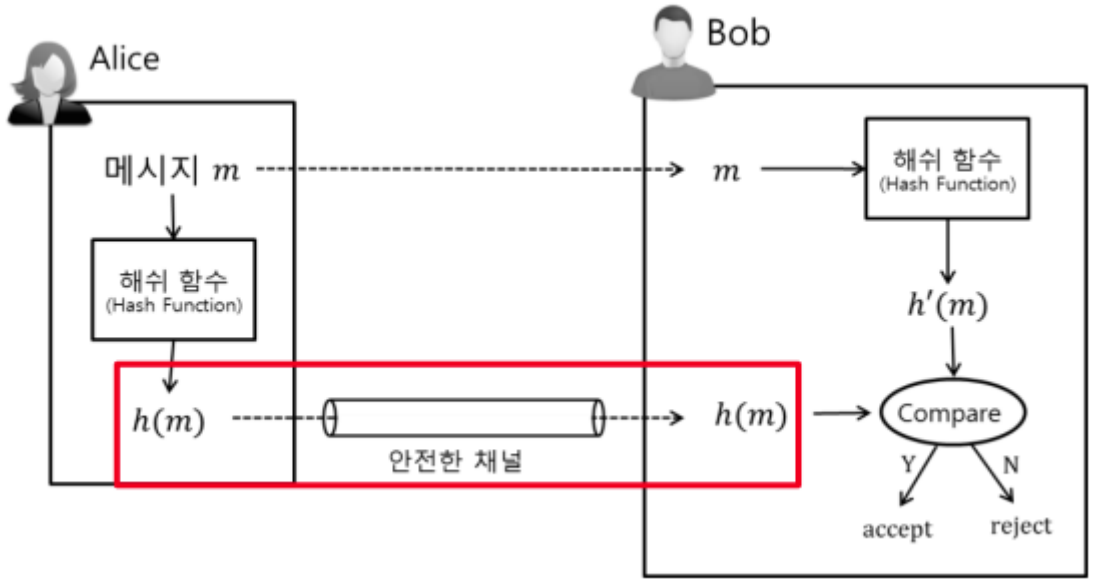
- 원본 메시지의 변조를 방지하기 위한 서비스
- 대칭키 & 공개키 암호시스템은 비밀성(Credentiality)을 제공
- 무결성을 위해 Modification Detection Code (MDC) 사용
  - 변조 감지 코드를 생성하기 우한 방법으로 해시 함수를 사용



Modification Detection Code (MDC)

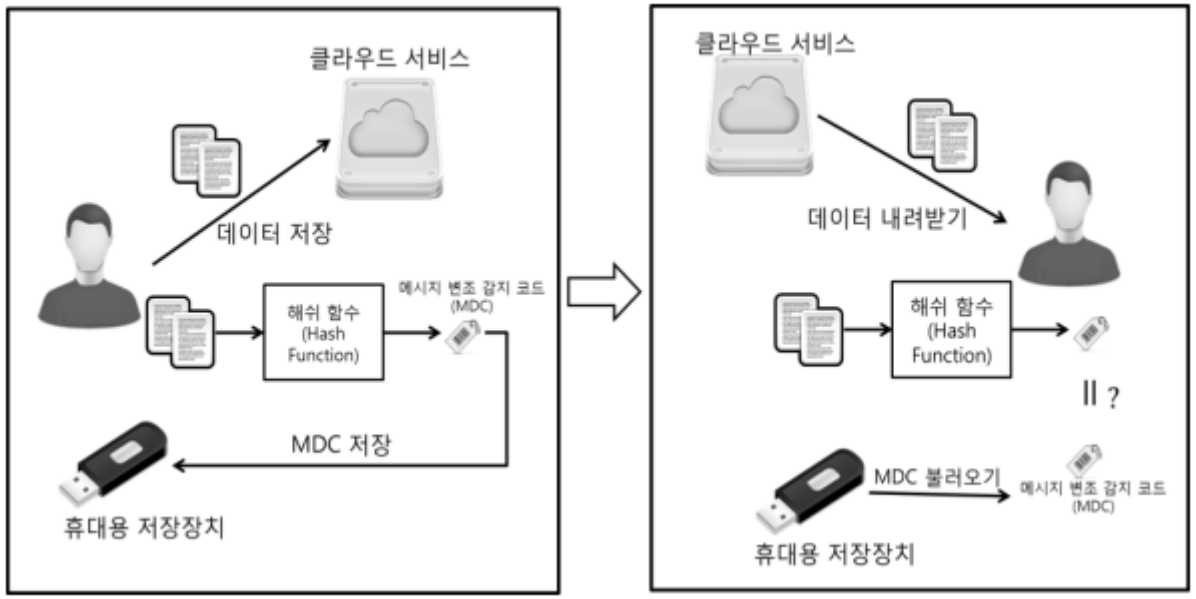
■ 메시지 변조 감지 코드 (MDC)

- 메시지 m과 MDC h(m)은 분리될 수 있으며, h(m)은 변조되지 않도록 하는 것이 중요
  - Alice는 메시지 m에 대한 MDC h(m) 생성
  - Alice → Bob : m, h(m), 단 h(m)은 안전한 채널로 전송



■ MDC를 이용한 메시지 변조 감지 코드 활용 예

- 클라우드 서비스에서 원본 메시지에 대한 무결성을 제공



■ 해시 함수

- h = H(M)
- 임의의 크기의 데이터를 입력 값
- 고정된 크기의 출력 값
  - 하드웨어 및 소프트웨어의 적용이 쉬워야 하며, 어떤 입력 데이터에 대해서도 출력 값을 계산하기 용이
  - 해시 함수는 공개된 함수이며 키가 사용되지 않음
    - ✓ 키를 사용한 해시 함수: h = H(k, M) → MAC (Message Authentication Code)
  - 전자 서명을 생성하기 위하여 사용됨
    - ✓ 메시지 m에 대한 서명은 h(m)을 생성 후 서명

Hash Function (cont'd)

■ 암호학적으로 안전한 해시 함수의 성질

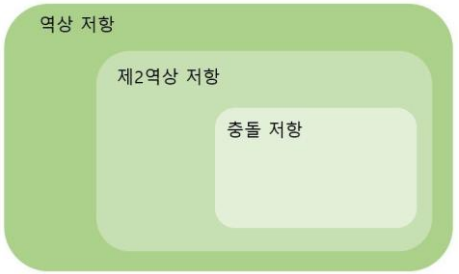
- 역상 저항성 (Preimage Resistance)
  - Given h, infeasible to find x s.t. H(x) = h

x	$h(x)$	x	$h(x)$
34A2BFE41	A535	BBDEE123	AEC7
53EF95	ABC4	5200DE	8DF9
824A74CD	7601	A6EEF1123	1B2A
2572AB4	AB12	A84F	0EF2
294AE9500D	5234	C1	52F3

- 제2 역상 저항성 (Second Preimage Resistance)
  - Given x, infeasible to find y s.t. H(y)=H(x)

x	$h(x)$	x	$h(x)$
180A	190D	AEF7D8C	8DF9
100F	58FE	32178FE88	802C
53EF95	8DF9	8000127D	EF19
C1	52F3	2572AB4	6A9D
824A74CD	1B2A	A6EEF1123	1B2A
A84F	1B2A	34A2BFE41	A535
BBDEE123	AEC7	5200DE	8DF9
294AE9500D	802C	A	ABC4

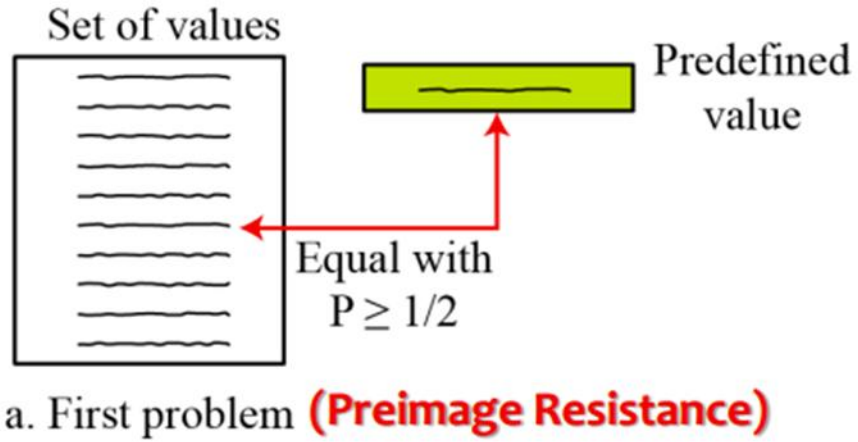
- 충돌 저항성 (Collision Resistance)
  - Infeasible to find any(x, y) s.t. H(y) = H(x)



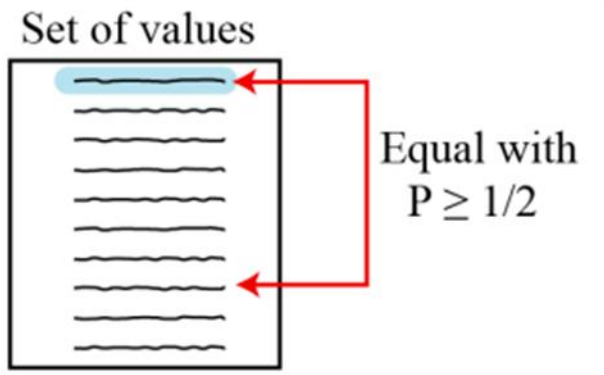
x	$h(x)$	x	$h(x)$
17AB8AF023	190D	9EFAD58	154A
102F	58FE	F2178FE88	A75F
53EF95	802C	50D00127D	535D
C1	52F3	AB	800E
824AA4CD	1B2A	A6EEF1123	AA5D
12A3	1B2A	34A2BFE41	C3F0
67BAD13	AEC7	320410DE	1189
A54F2	D812	112A	89DF

Note: Zip 등의 압축함수와 Checksum는 역상 저항성을 만족하지 못함

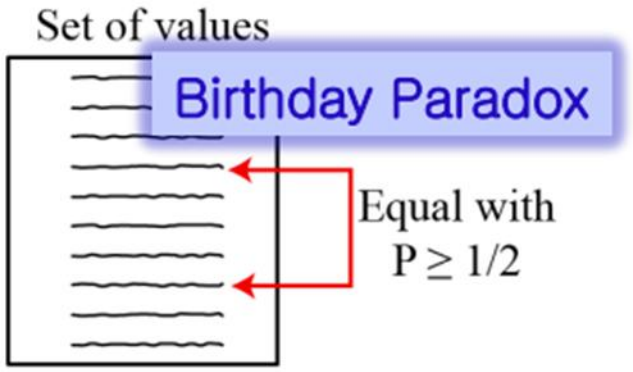
■ 암호학적으로 안전한 해시 함수



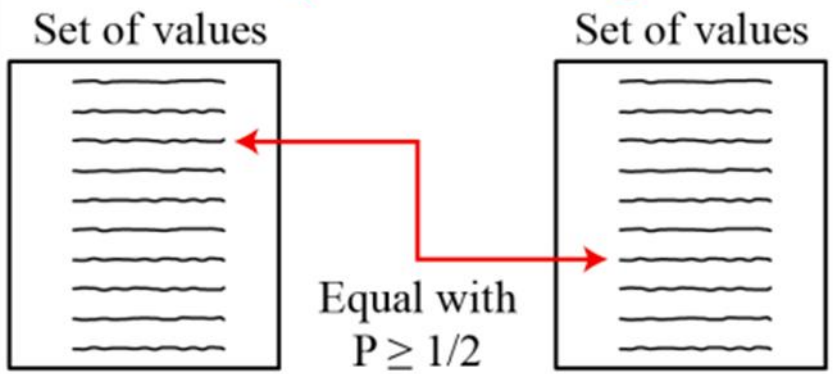
a. First problem **(Preimage Resistance)**



b. Second problem
(Second Preimage Resistance)



c. Third problem **(Collision Resistance)**

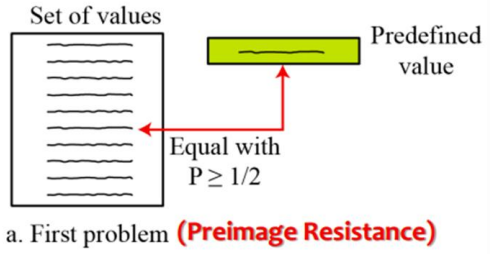


d. Fourth problem

Hash Function (cont'd)

■ Birthday Paradox

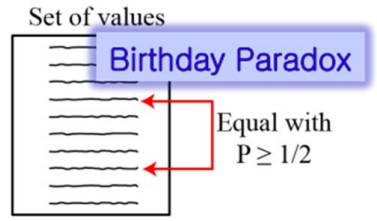
- 1st Problem
  - n 비트인 h(=H(x))가 주어진 경우 50% 이상의 확률로 x를 찾는 문제
    - ✓ "고정된 생일 h"를 갖는 학생이 50% 이상의 확률로 존재하기 위하여 필요한 학생의 수
    - ✓ 해시 함수를 평가해야 하는 횟수 → 0.69×2^n



a. First problem **(Preimage Resistance)**

- 2nd Problem
  - N비트인 h(=H(x))가 주어진 경우 50% 이상의 확률로 h = H(y)인 y(≠x)를 찾는 문제
    - ✓ "고정된 생일 h"를 갖는 학생이 있는 경우, 동일한 생일을 갖는 학생이 50% 이상의 확률로 존재하기 위하여 필요한 학생의 수
    - ✓ 해시 함수를 평가해야 하는 횟수 → $0.69 \times 2^n + 1$

- 3rd Problem
  - 50% 이상의 확률로 h = H(x) = H(y)인 쌍 (x, y) (y≠x)를 찾는 문제
  - 즉, 같은 생일을 갖는 두 학생이 50% 이상의 확률로 존재하기 위하여 필요한 학생의 수
    - ✓ $p = 1 - 1\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\ldots\left(1 - \frac{1}{n}\right) = 1.18 \times 2^{\frac{n}{2}}$ → $50\% = 1.18 \times 365^{\frac{1}{2}} \approx 23$
    - ✓ 120비트인 해시 값 h의 안전성은 대략 2^{60}



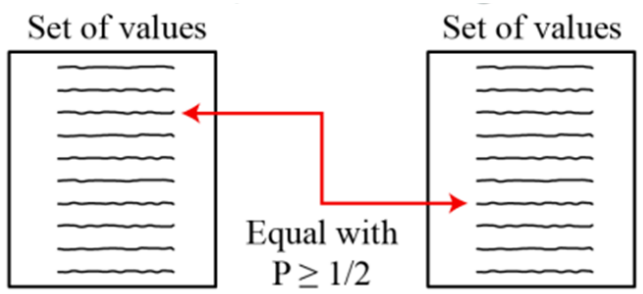
c. Third problem **(Collision Resistance)**

Running on a 1 GHz computer (with 10^9 cycles per second), 2^{60} CPU cycles requires $2^{60}/10^9$, or about 35 years. → BUT the fastest existing supercomputer (as of 2008) can execute 4.78×10^{14} per second, and 2^{60} would require only 40 minutes. Taking 2^{80} is more prudent choice (80 years on the supercomputer mentioned)

■ Birthday Problems

- 4th Problem
  - 서명 공격
    - ✓ m≠m′ 이면서 H(m) = H(m′)인 쌍 (m, m′)을 발견하여 공격
    - ✓ 서명자에게 S = Sig(H(m))를 받고 m′에 대한 서명이라 주장

1. 공격자는 정당한 문장 m과 동일한 의미를 갖는 $2^{\frac{n}{2}}$ 개의 문장을 생성 → M
2. 공격자는 자신이 원하는 문장 m′과 동일한 의미를 갖는 $2^{\frac{n}{2}}$ 개의 문장을 생성 → M′
3. M과 M′에서 동일한 해시 값을 갖는 (m, m′)을 50% 이상의 확률로 발견



d. Fourth problem

Hash Function (cont'd)

■ MD5 (considered harmful today)

- MD5를 사용하는 공개키 인증서 위조에 성공 → PKI 취약성으로 직결
- Web site phishing 공격 가능

■ SHA (Secure Hash Algorithm)

- 1993년 미국 국가 안전 보장국(NSA)과 미국 표준 기술연구소(NIST)가 함께 설계
  - SHA-1: SHA-0의 취약점을 보강, 160비트 해시 값
  - 2002년 SHA-2 계열 (SHA-256, SHA-284, SHA-512) 설계
  - In 2005, security flaws were identified in SHA-1. The SHA-2 variants are similar to SHA-1. → SHA-3 공모, 2012년 Keccak (pronounced "catch-ack") 알고리즘 최종 선정

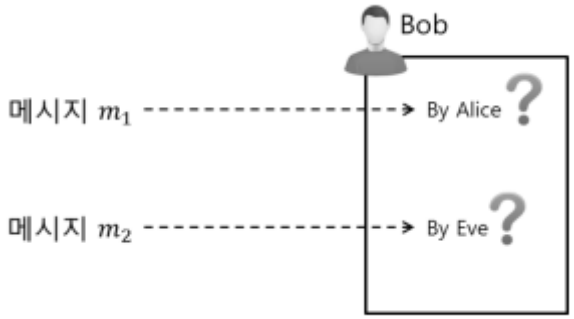
02

Message Authentication

Message Authentication Code (MAC)

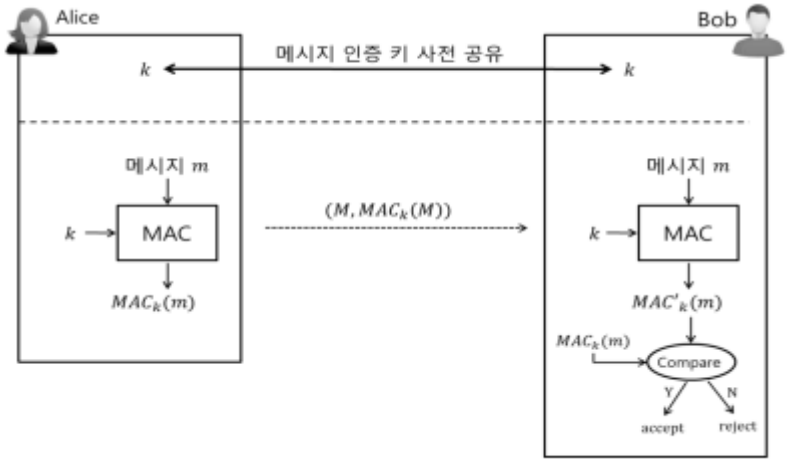
■ 메시지 인증 코드

- 메시지 근원 인증 (Message origin authentication)을 제공
  - 해시 함수를 이용한 MDC는 무결성 만을 제공 → 메시지의 출처를 확인할 수 없음
  - MAC은 무결성 제공 뿐 아니라 메시지의 출처 확인도 가능함



MAC (cont'd)

- $\boldsymbol{MAC_k(m) = h(k, m)}$
  - Alice와 Bob은 MAC Key k를 사전에 공유
  - Alice는 메시지 m에 대한 $MAC_k(m)$ 생성
  - Alice → Bob : m, $MAC_k(m)$
  - Bob은 수신 메시지 m′에 대한 $MAC_k(m')$ 생성
  - $MAC_k(m) =? MAC_k(m')$



MAC (cont'd)

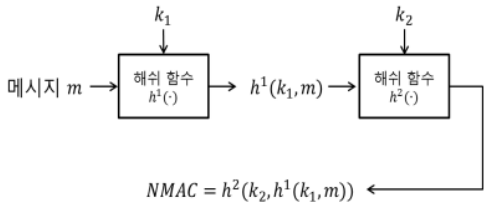
■ MAC 생성 방법

- 해시 함수 기반 메시지 인증 코드
  - Nested MAC
  - HMAC

- 블록 암호 알고리즘 기반 메시지 인증 코드
  - CBC-MAC
  - CMAC

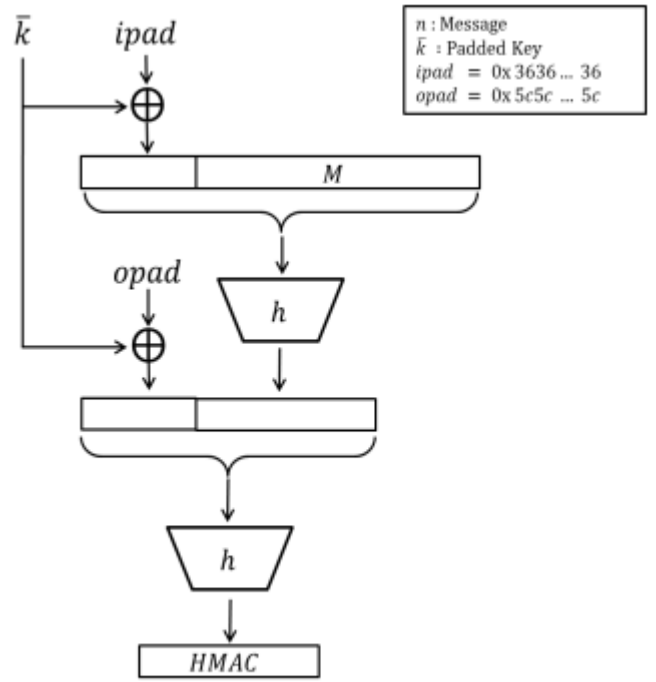
■ Nested MAC

- $NMAC_{k1,k2}(m) = h^2(k_2, h^1(k_1, m))$



■ HMAC (Hashed MAC)

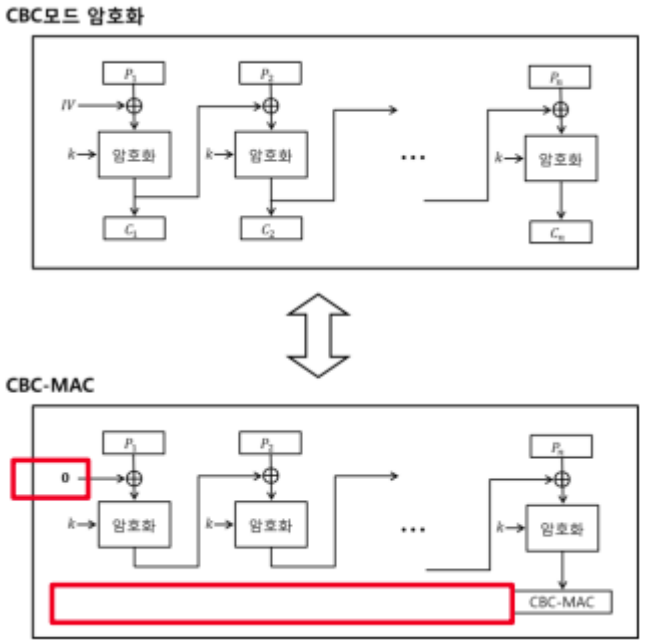
- NIST 표준 FIPS198
- 메시지 m을 b비트 사이즈로 분할
- key 왼쪽에 0으로 padding하여 b비트로 맞추어 $\bar{k}$ 생성



$HMAC = h(\bar{k} \oplus opad \,||\, h(\bar{k} \oplus ipad \,||\, m))$

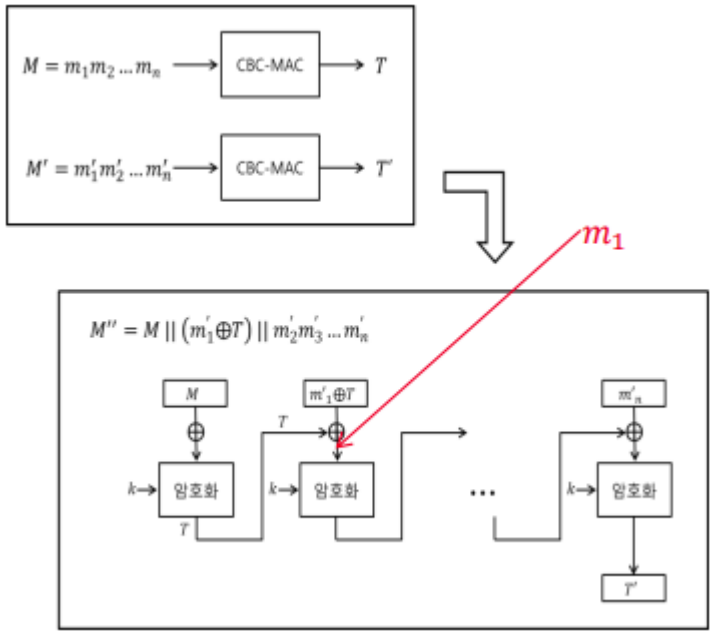
■ CBC-MAC (Cipher Block Chaining MAC)

- 블록 암호의 운영모드 중 CBC 모드를 사용하여 메시지 인증 코드를 생성 → 실제 환경에서 편리
  - 고정된 초기벡터(IV=0)를 사용
  - 고정된 메시지에 대한 MAC



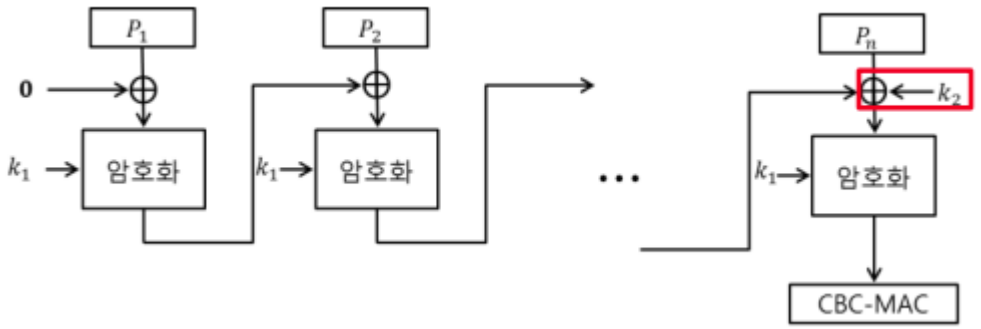
■ CBC-MAC

- 가변길이의 메시지에 대한 메시지 인증 코드를 생성하는 경우 → 제2역상을 계산할 수 있음
  - $M = m_1 m_2 \dots m_n$, $M' = m'_1 m'_2 \dots m'_n$인 (M, T), (M′, T′)를 수집
  - $M'' = M \| (m'_1 \oplus T) \| m'_2 m'_3 \dots m'_n$을 계산
  - $CBC\ MAC_k(M') = CBC\ MAC_k(M'')$



■ CMAC(Cipher-based MAC)

- NIST SP800-38B 표준
- 고정된 길이의 메시지만을 사용해야 하는 CBC-MAC에서의 제약 사항을 제거
  - 2개의 키 k_1, k_2 사용
  - 두 메시지 M′과 M″의 메시지 인증 코드 값이 같아지도록 하기 위해서는 k_2에 대한 값을 알고 있어야만 함



03

Digital Signature

■ 전자서명 vs. 종이서명

	종이 서명	전자 서명
작성 형태	문서 내에 서명이 포함	문서와 서명이 분리
검증 방법	서명 파일의 서명과 대조, 비교	별도의 검증기술을 적용
서명과 문서의 관계	One-to-Many	One-to-One
서명 검증	서명 파일이 필요	공개 검증

전자 서명의 기본 원리

■ 전자 서명 과정

- 공개키 암호시스템과 유사
  - (공개키 pk, 개인키 sk) → (검증키 pk, 서명키 sk)
  - $Sig_{sk}(m)$ = 서명 생성 알고리즘
  - $Ver_{pk}(m, s)$ = 검증 알고리즘
    - ✓ Alice → Bob : 메시지 m에 대한 서명 S = $Sig_{sk}(m)$
    - ✓ Bob : $Ver_{pk}(m, s)$로 검증

해시 함수를 이용한 전자 서명

■ **해시 후 서명 :** $s = Sig_{sk}(h(m))$

- 효율적
- 서명의 순서 변경이나 삭제를 방지
- 서명 위조 방지

전자 서명이 제공하는 보안 서비스

- **메시지 무결성 (Message Integrity)**
  - $Sig_{sk}\big(h(m)\big) \neq Sig_{sk}\big(h(m')\big)\ if\ m \neq m'$ {h는 충동저항성}
- **메시지 인증 (Message Authentication)**
  - 정당한 sk를 이용한 서명만이 $Ver_{pk}\Big(m, Sig_{sk}\big(h(m)\big)\Big)$을 통과
- **부인방지 (Non-Repudiation)**
  - $Sig_{sk}\big(h(m)\big)$을 생성할 수 있는 사람은 sk의 소지자 뿐!

MAC vs. 전자서명

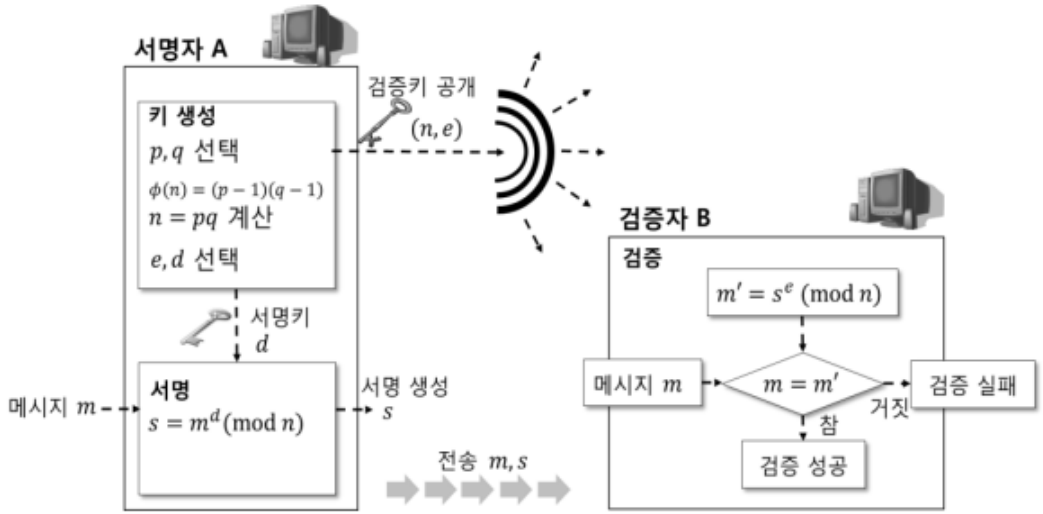
■ **MAC과 전자서명은 모두 무결성과 메시지 인증 제공**

■ **차이점**

- 공개 검증 (public verification)
  - pk 는 공개된 정보
- 키 관리
  - MAC의 경우, MAC Key를 사전에 공유해야함
- 부인방지
  - $Sig_{sk}(h(m))$을 생성할 수 있는 사람은 sk의 소지자
- Transferability
  - MAC의 경우, MAC Key를 사전에 공유한 경우에만 Transfer 가능 (단, MAC Key는 유일해야 하므로 2명 사이에서만 사용)
- 효율성
  - MAC은 전자서명보다 2~3배 효율적

■ 해시를 이용한 RSA 전자서명

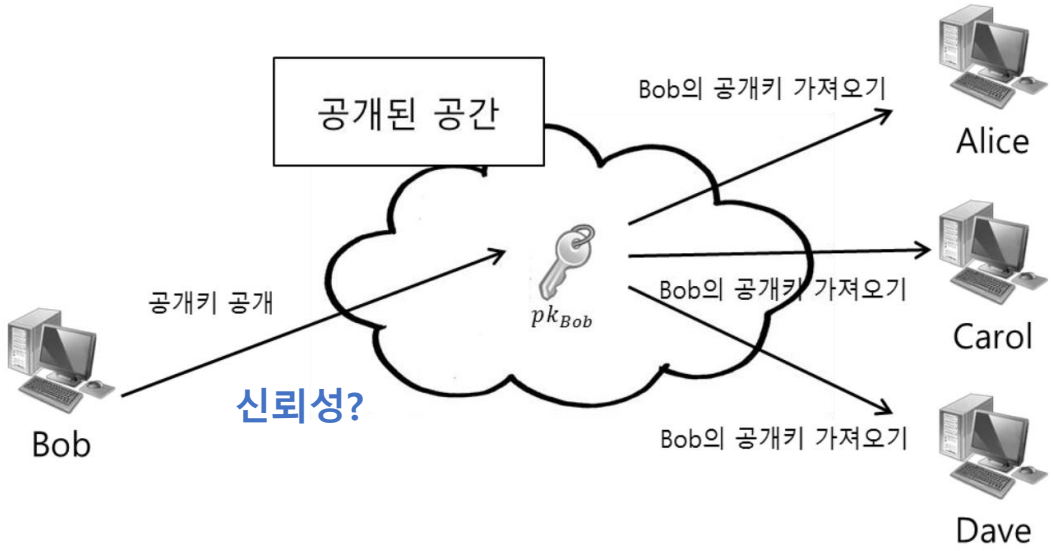
- 키 생성: RSA 암호와 동일
- 서명 생성
  - $Sig_{sk}(h(m)) \equiv \{h(m)\}^d \equiv s \ (mod\ n)$
- 서명 검증
  - $s^e (mod\ n) \equiv h(m)$
  - $h(m') =? h(m)$



Public Key Infrastructure (PKI)

■ PKI 정의

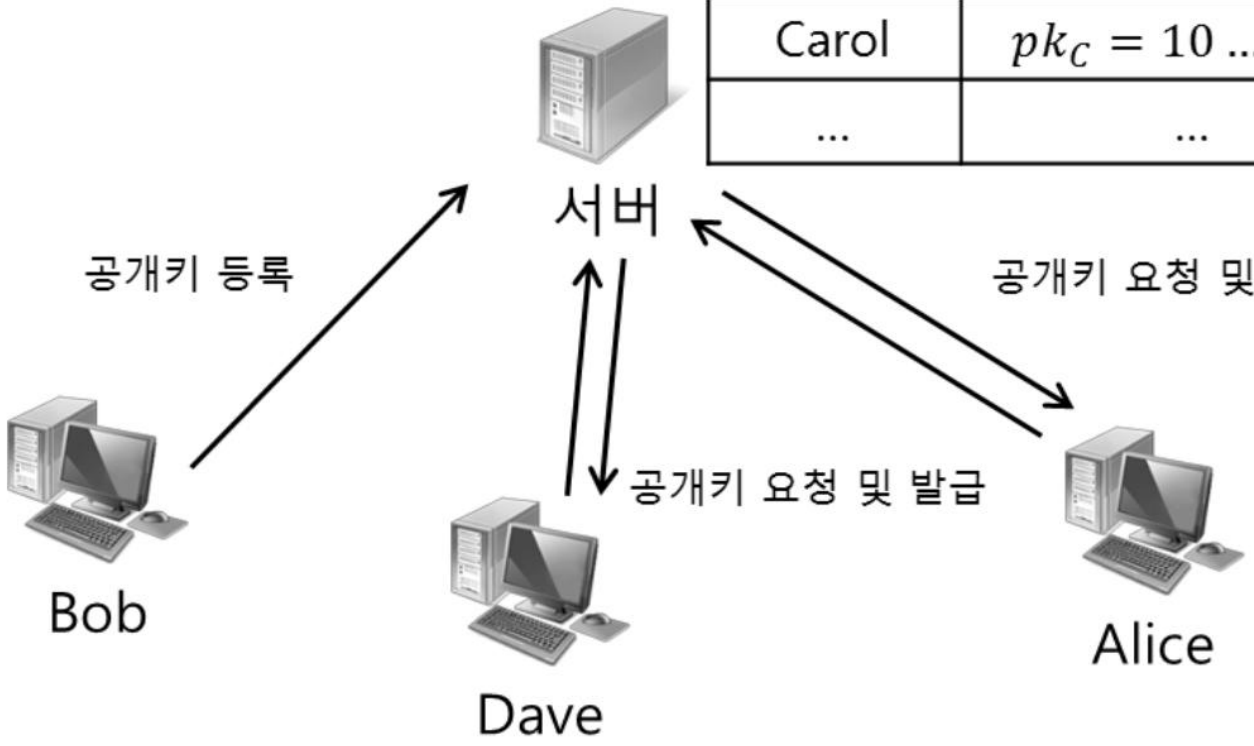
- 비대칭 암호시스템에 기초해서 디지털 인증서 생성, 관리, 저장, 배분, 취소에 필요한 하드웨어, 소프트웨어, 사람, 정책 및 절차라고 정의하고, 공개 키를 효과적으로 운용하기 위해 정해진 많은 규격이나 선택 사양의 총칭을 의미한다.
- 공개키를 이용하여 송수신 데이터를 암호화하고 디지털 인증서를 통해 사용자를 인증하는 시스템으로, 공개키 암호 알고리즘을 안전하게 사용하기 위해 필요한 서비스를 제공하는 기반 구조이다.
- 안전성과 투명성을 위해서 제3의 신뢰받는 기관이 디지털 인증서를 발급하고, 인증서가 순환하는 트러스트 모델을 가지고 있다.
- PKI 가 제공하는 서비스
  - 기밀성, 무결성, 인증, 부인방지, 접근통제



PKI 필요성

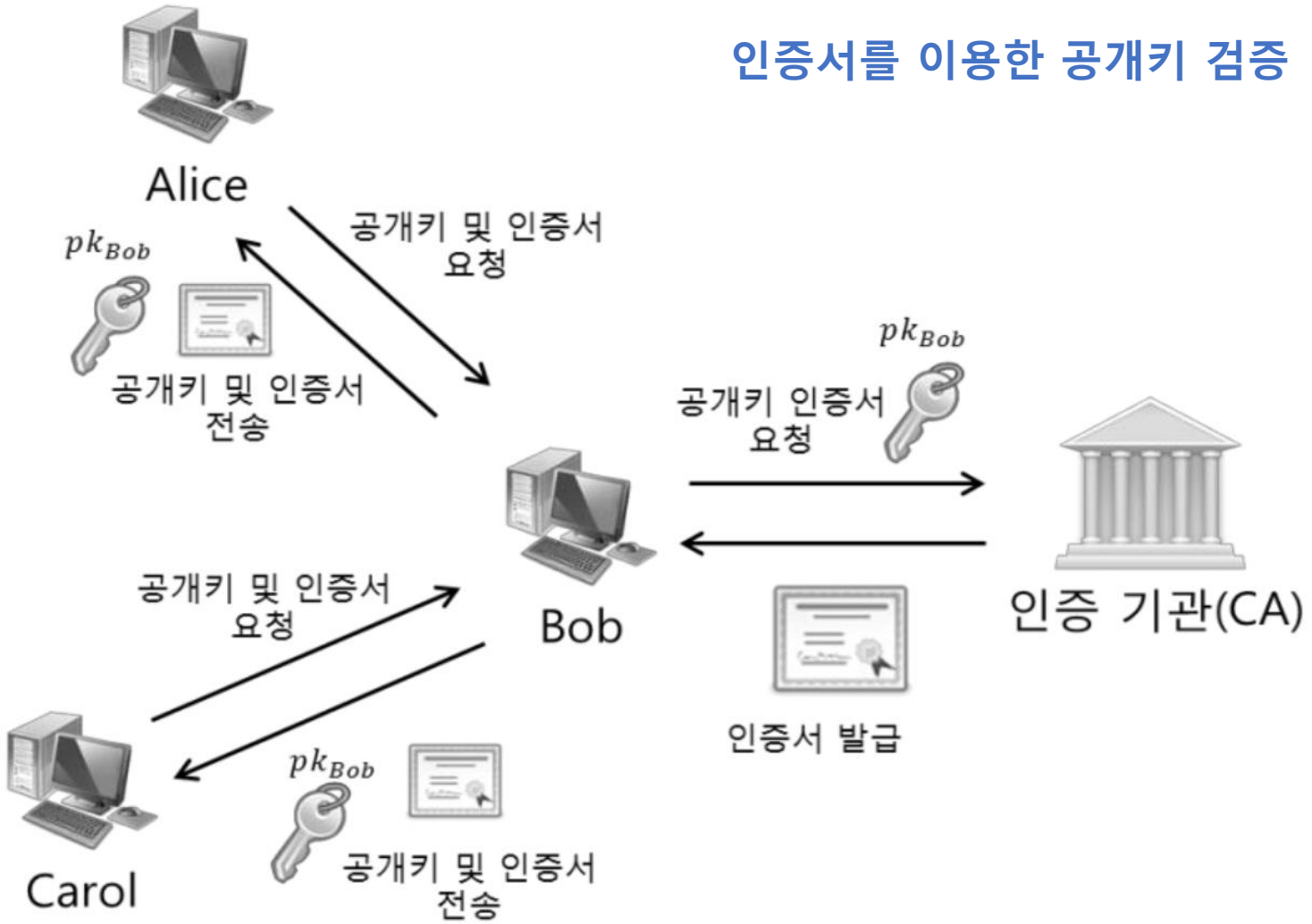
신뢰할 수 있는 서버를 이용 → 서버 과부하

ID	공개키
Alice	$pk_A = 11 \ldots 010$
Bob	$pk_B = 10 \ldots 010$
Carol	$pk_C = 10 \ldots 111$
…	…



PKI 필요성 (cont'd)

인증서를 이용한 공개키 검증



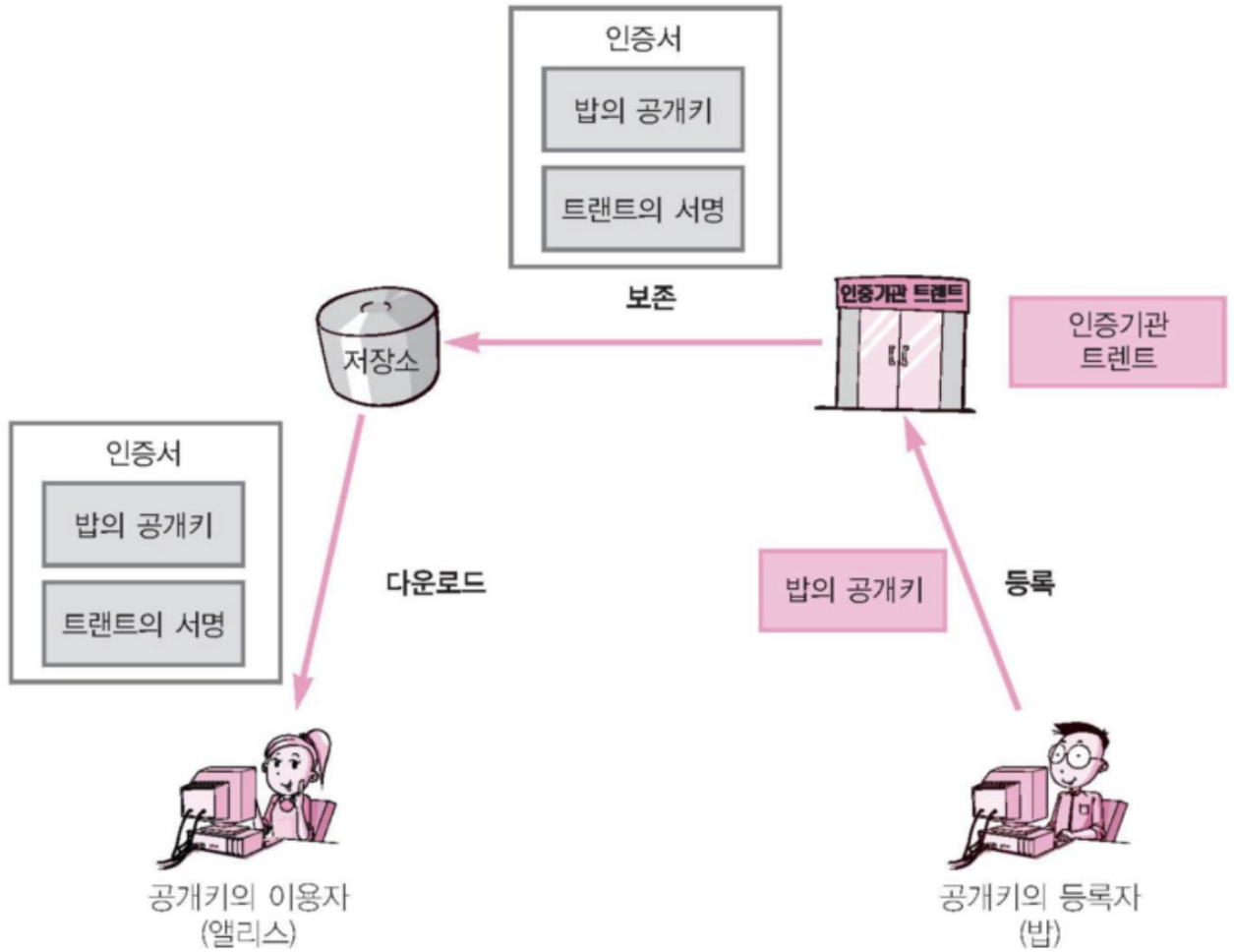
PKI 구성 요소

■ **이용자**
- PKI를 이용하는 사람

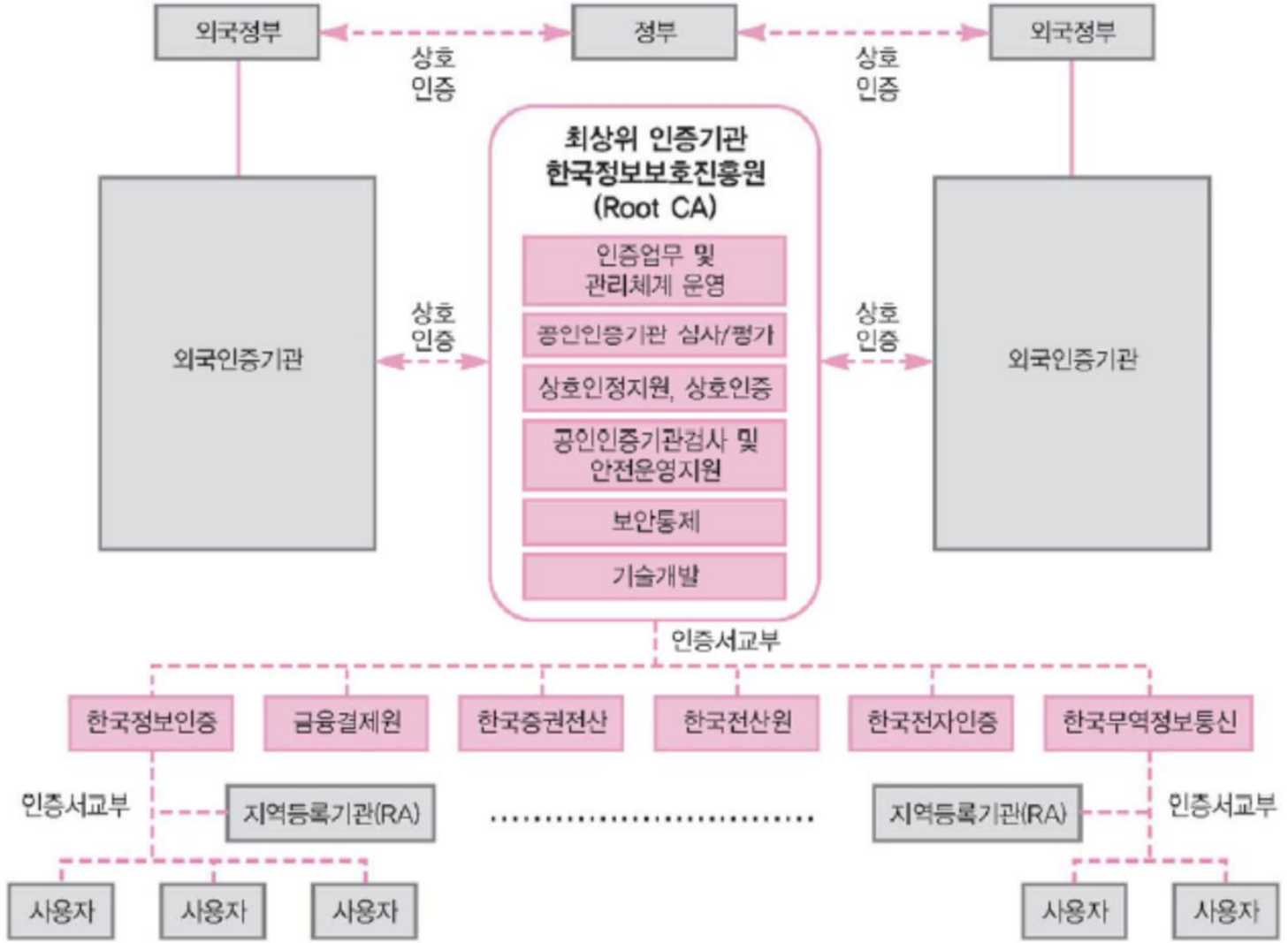
■ **인증기관**
- 인증서를 발행하는 기관

■ **등록기관**
- 인증서를 등록하는 기관

■ **저장소**
- 인증서를 보관하고 있는 데이터베이스



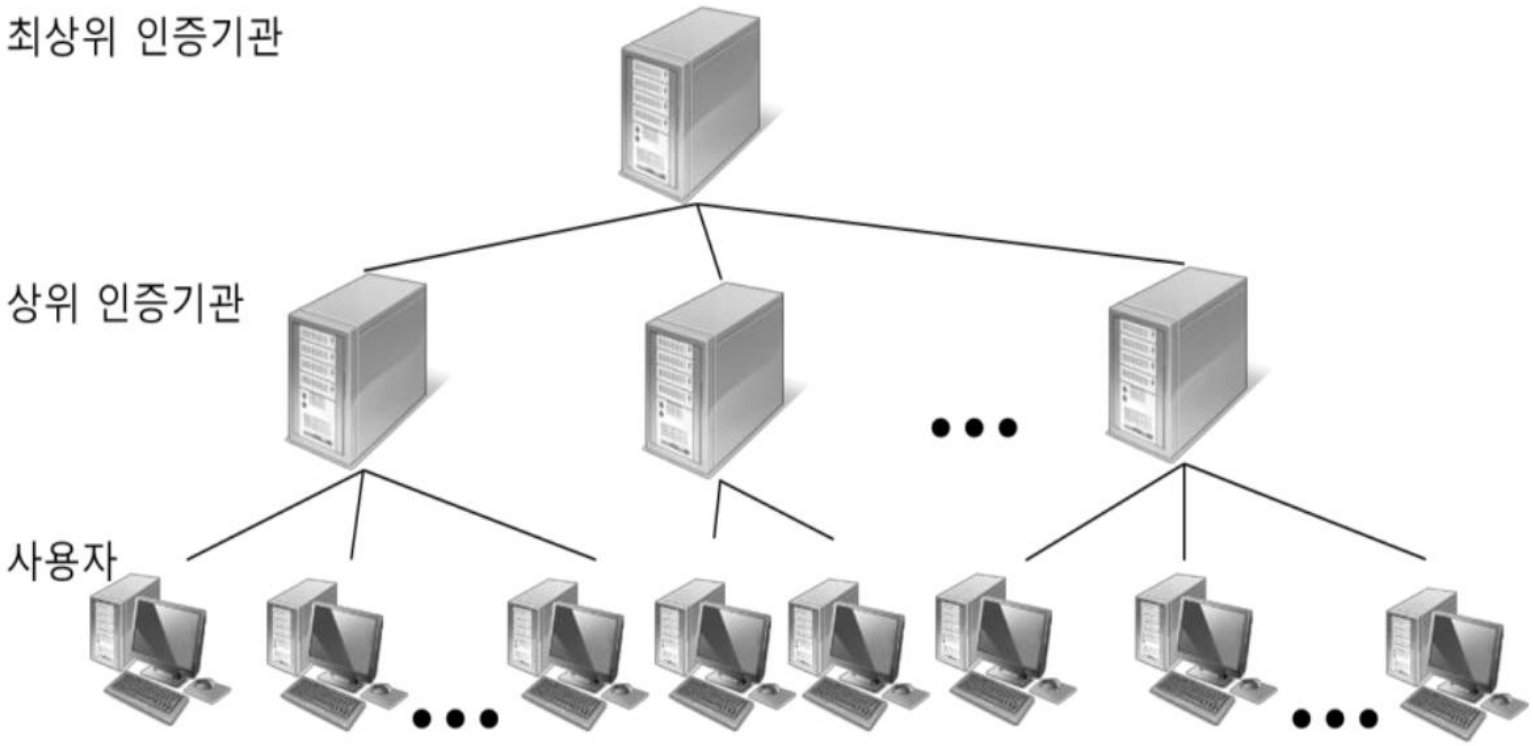
PKI 체계도


외국정부
정부
외국정부
상호 인증
상호 인증
최상위 인증기관
한국정보보호진흥원
(Root CA)
인증업무 및 관리체계 운영
공인인증기관 심사/평가
상호인정지원, 상호인증
공인인증기관검사 및 안전운영지원
보안통제
기술개발
외국인증기관
상호 인증
상호 인증
외국인증기관
인증서교부
한국정보인증
금융결제원
한국증권전산
한국전산원
한국전자인증
한국무역정보통신
인증서교부
지역등록기관(RA)
지역등록기관(RA)
인증서교부
사용자
사용자
사용자
사용자
사용자

인증 기관들 간 신뢰 모델

■ Hierarchical Model

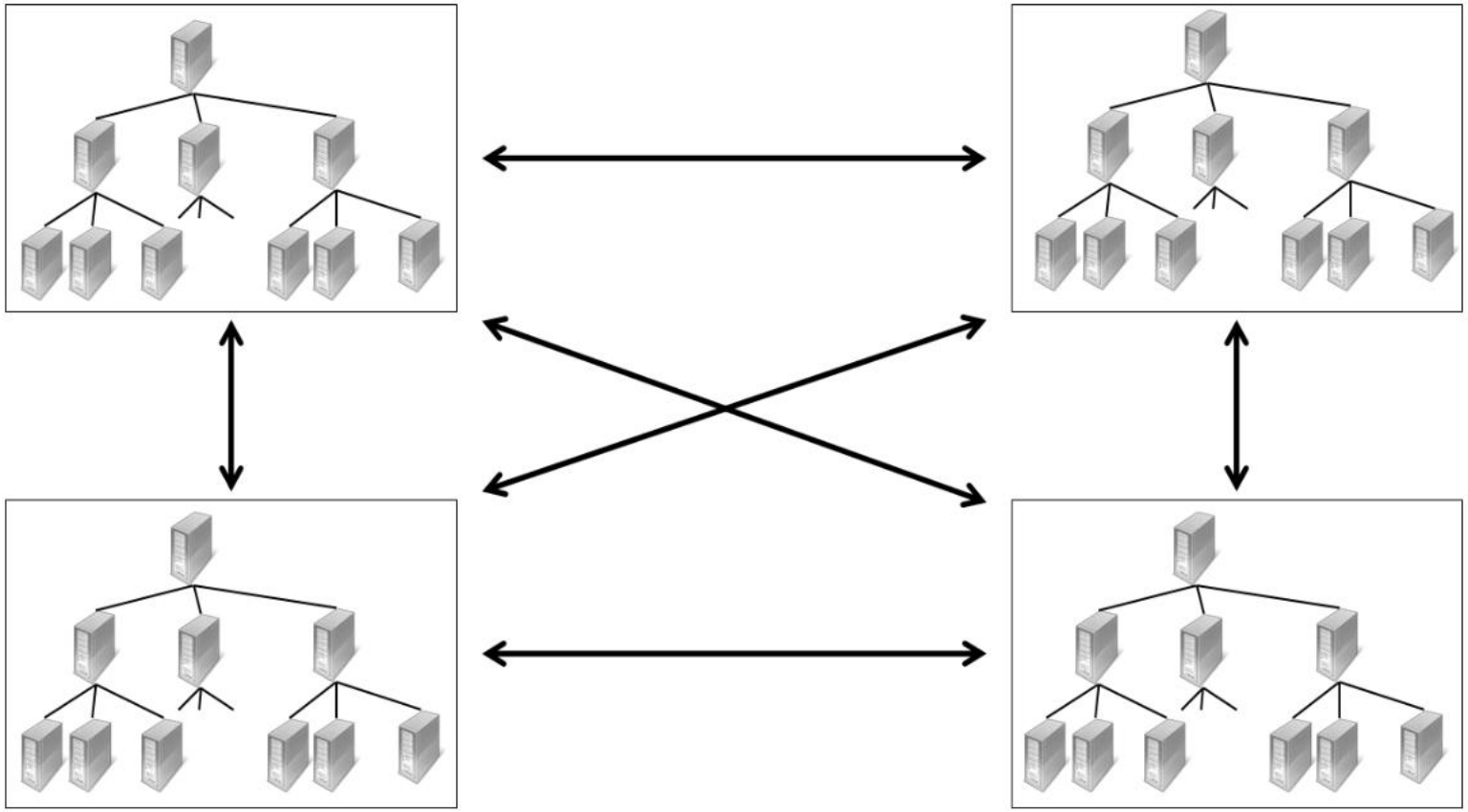
- 상위 계층이 바로 아래 계층의 인증서를 발급, 최상위 계층인 루트 인증 기관은 self-signing



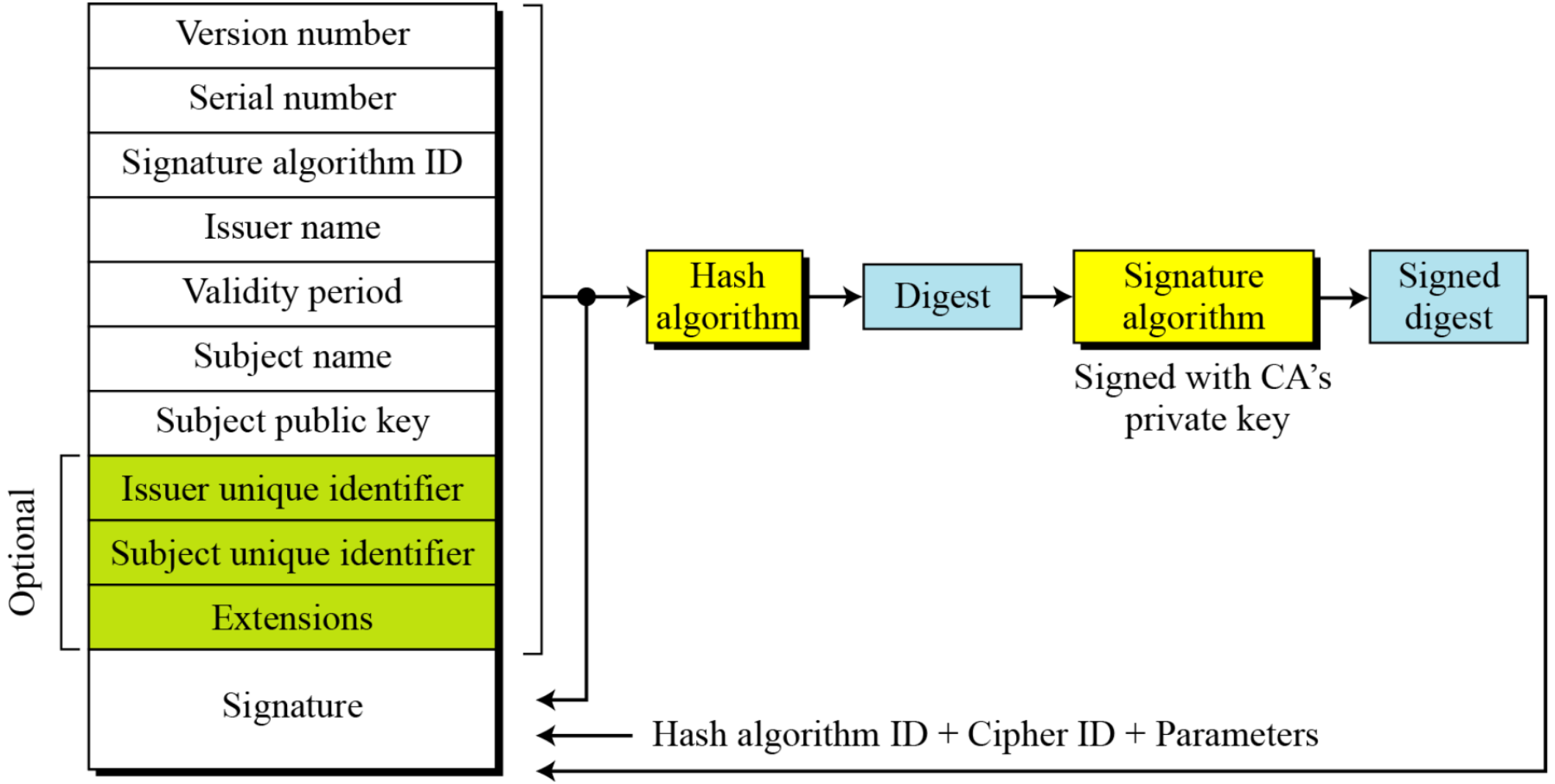
인증 기관들 간 신뢰 모델 (cont'd)

■ Mesh Model

- 국가 간 상호인증



인증서 형태


Version number
Serial number
Signature algorithm ID
Issuer name
Validity period
Subject name
Subject public key
Issuer unique identifier
Subject unique identifier
Extensions
Optional
Signature
Hash algorithm
Digest
Signature algorithm
Signed digest
Signed with CA's private key
Hash algorithm ID + Cipher ID + Parameters

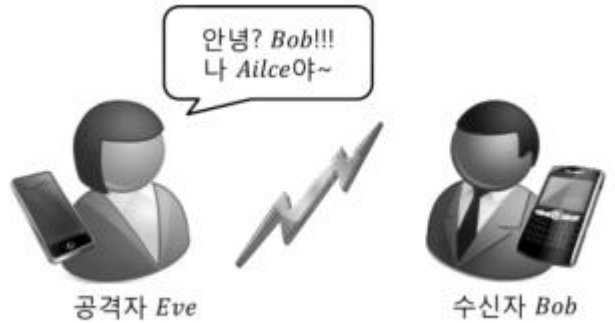
04

Object Authentication

■ 개체인증

- 개체의 신원을 증명하기 위한 일련의 과정
  - 개체(Entity): 사람(User) or 기기(Device)

- 인증 정보
  - What you are (voice, fingerprint, Iris)
  - What you know (password)
  - What you have (smart card, token card)



패스워드 방식

■ 패스워드

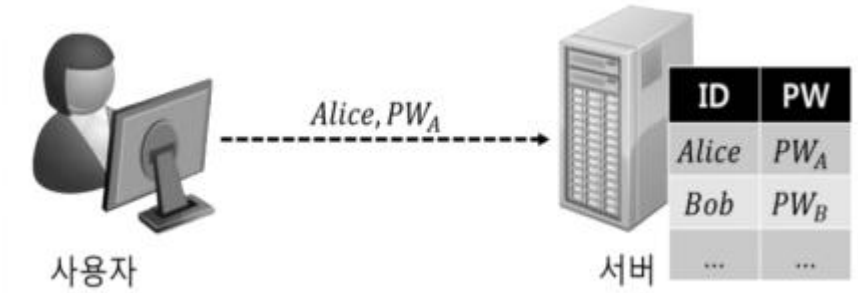
- Low entropy
  - 64비트 패스워드를 발견하기 위해서는 2^{64} 필요

- 안전한 패스워드
  - 국내: 방통위, KISA (패스워드 선택 및 이용 안내서)
    - ✓ 세가지 종류 이상의 문자구성으로 8자리 이상의 길이로 구성된 문자열
    - ✓ 두 가지 종류 이상의 문자구성으로 10자리 이상의 길이로 구성된 문자열
    - ✓ 문자종류는 알파벳 대문자와 소문자, 특수문자, 숫자 4가지
  - 해외: NIST 800-63 (Digital Identity Guideline)

- Refer
  - Strong passwords: How to create and use them.
  - How strong is yours?: Password Checker

패스워드 방식 (cont'd)

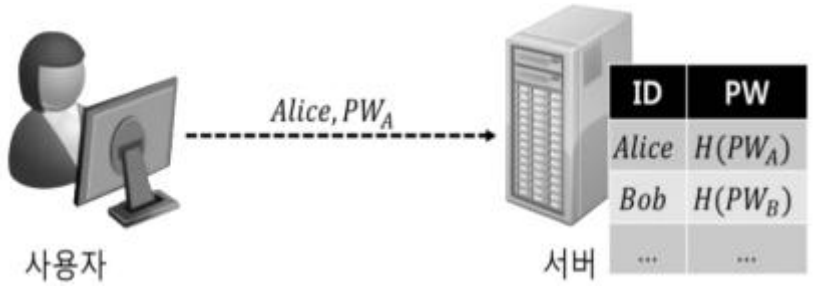
■ 고정된 패스워드



- 도청 등의 위협
- 패스워드 테이블의 유출 시 위험

패스워드 방식 (cont'd)

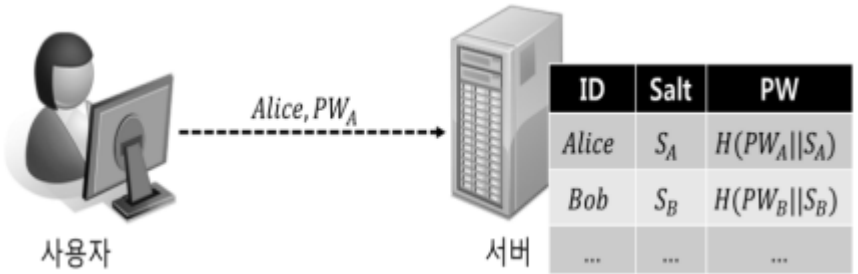
■ Hashed 패스워드



- 패스워드 테이블의 유출 시 해쉬 함수의 역상저항성으로 인하여 안전
  - 특정인 Alice의 패스워드를 알기 위해서는 $O(2^n)$번의 해쉬 평가 (n: 해쉬 함수의 출력 길이)
  - 임의 사용자의 패스워드를 알기 위해서는 offline 사전공격(dictionary attack)이 효과적
    - ✓ 추측된 패스워드 PW의 해쉬 값 H(PW)와 패스워드 테이블의 모든 해쉬 값과 비교

패스워드 방식 (cont'd)

■ Salt 사용



- 임의 사용자의 패스워드를 알기 위해서는 offline 사전공격(dictionary attack)을 방어
  - 추측된 패스워드 PW의 해쉬 값 H(PW)와 패스워드 테이블의 해쉬 값과 직접 비교 불가능
  - 모든 사용자 ID에 대하여 H(PW|S_{ID})와 테이블의 해쉬 값과 비교해야 함
  - ID의 개수가 t인 경우, 추측된 PW에 대하여 t번씩 증가
    - ✓ Salt가 공개된 경우, 특정인 Alice의 PW를 알기 위한 계산은 변동 없음
    - ✓ 여전히 $O(2^n)$번의 해쉬 평가 (n: 해쉬 함수의 출력 길이)

패스워드 방식 (cont'd)

■ OTP (One-Time Password)

- 매번 다른 난수 사용
  - 사전 공격(Dictionary Attack)이나 재전송 공격(Replay Attack) 등으로 부터 안전

- 동기화 방식의 일회용 패스워드 (Synchronized OTP)
  - 사용자와 서버는 시드(Seed)를 공유 후, 동일한 패스워드 생성
  - 시간 동기화 방식
    - ✓ $sk = h(seed,\ T)$: current time T
    - ✓ 적절한 오차 허용 → 시간 구간 설정
  - 이벤트 동기화 방식
    - ✓ $sk = h(seed,\ C)$: counter C
    - ✓ 전송 오류 시 C 동기화 필요
  - Hybrid 동기화 방식
    - ✓ 한 구간 내에 여러 번의 패스워드 생성, 카운터 값 증가
    - ✓ 각 구간마다 카운터 값 초기화

패스워드 방식 (cont'd)

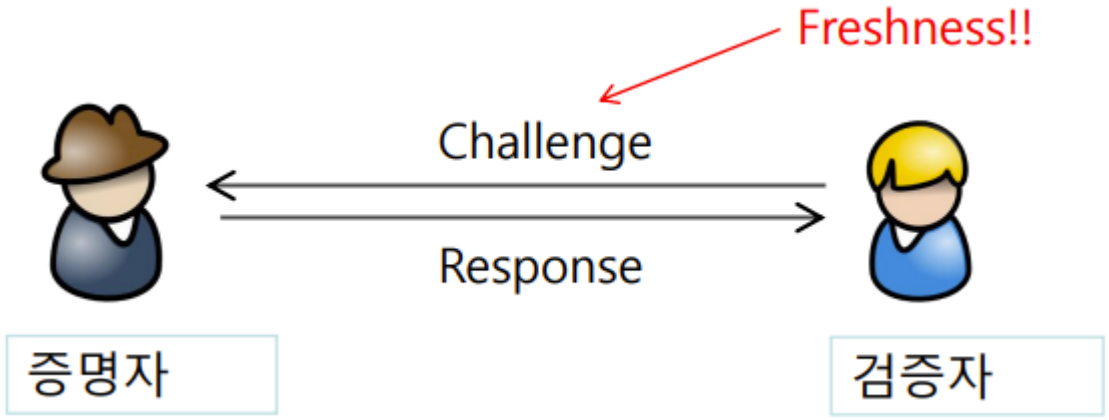
■ 비동기화 방식의 일회용 패스워드 (Non-Synchronized OTP)

- 질의-응답 (Challenge-Response) 방식
  - 동기화 불필요, 통신량 증가

- Lamport 방식 → Hash chain 사용
  - 사용자는 비밀 값 x를 생성, 서버의 접근 횟수 제한을 k
  - $x, h(x) = x_1, h\big(h(x)\big) = x_2, \ldots, h^k(x) = x_k$
  - 사용자와 서버는 초기 값 x_k 공유
  - 사용자 → 서버 : x_{k-1}
  - 서버는 $\mathrm{h}(x_{k-i}) = x_k$ 검증 후 x_{k-1} 저장
  - ....
  - 사용자 → 서버 : x_{k-i}

Challenge-Response 인증

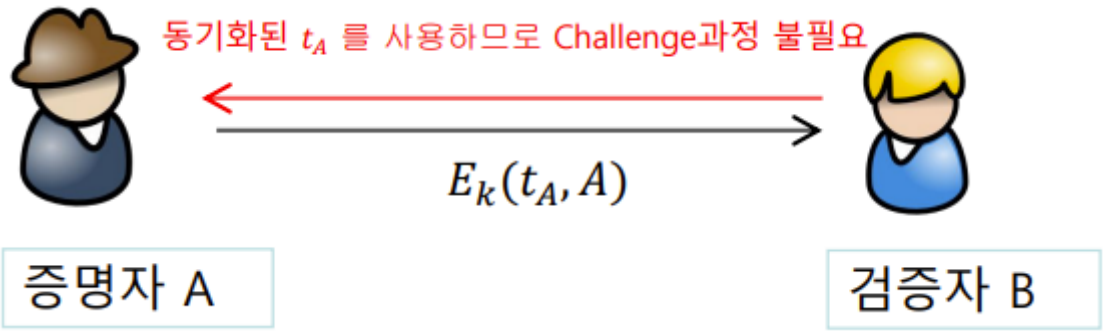
■ 검증자가 생성한 질의에 대하여 증명자가 응답



- 대칭키를 이용한 방식 & 공개키를 이용한 방식

■ 대칭키를 이용한 질의-응답 인증

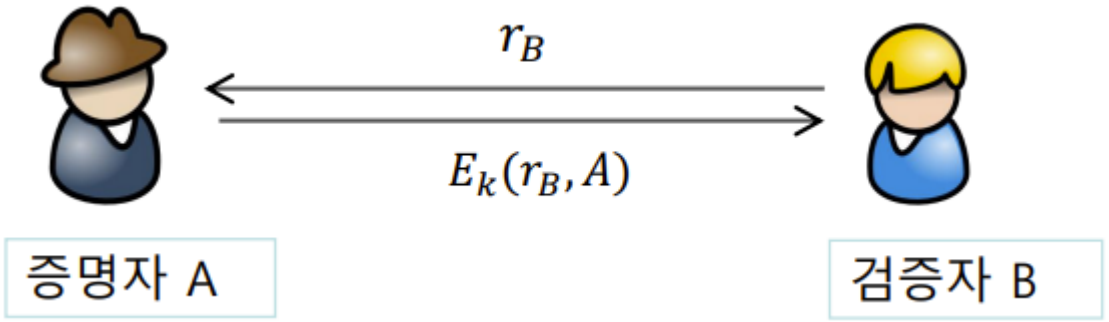
- 타임스탬프를 이용한 단방향 인증
  - A: 타임스탬프 t_A 생성
  - A → B : $E_k(t_A, A)$ {E_k는 A와 B가 사전 공유된 k로 암호}
  - B : $D_k(E_K(t_A, A))$ 후, t_A가 현재시간 구간에 들어오는지 확인



Challenge-Response 인증 (cont'd)

■ 대칭키를 이용한 질의-응답 인증

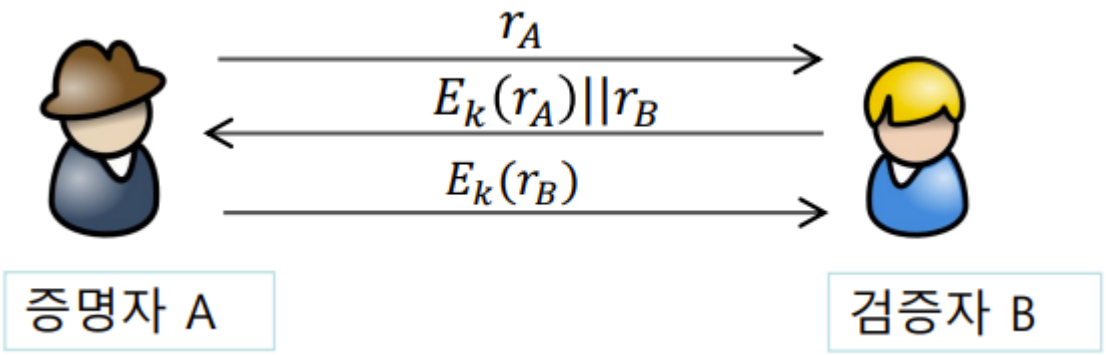
- 난수(Nonce)를 이용한 단방향 인증
  - B → A : r_B {challenge}
  - A → B : $\mathrm{E}_k(\mathrm{r_B}, \mathrm{A})$ {E_k는 A와 B가 사전 공유된 k로 암호}
  - 검증자 B : $\mathrm{D}_k(\mathrm{E}_k(\mathrm{r_B}, \mathrm{B}))$ 후, $\mathrm{r_B}$ 확인



Challenge-Response 인증 (cont'd)

■ 대칭키를 이용한 질의-응답 인증

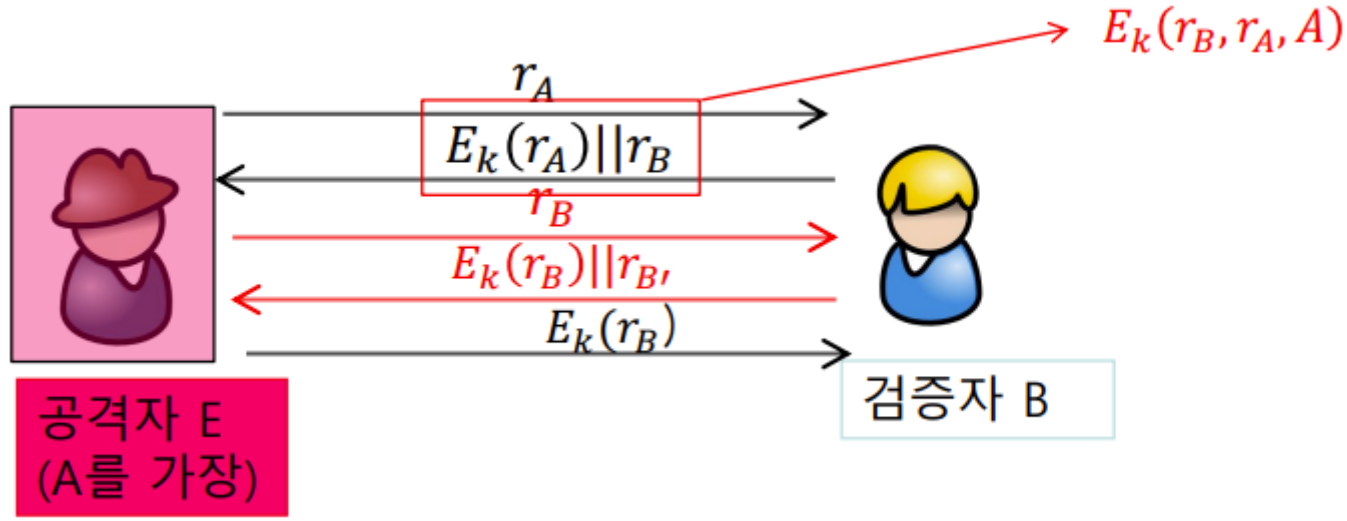
- 난수(Nonce)를 이용한 양방향 인증
  - A → B : k_A {A의 challenge}
  - B → A : $\mathrm{E}_k(\mathrm{r_A})|\mathrm{r_B}$ {응답 & B의 challenge}
  - A : $\mathrm{D}_k(\mathrm{E_k}(\mathrm{r_A}))$ 후, $\mathrm{r_A}$ 확인
  - A → B : $\mathrm{E}_k(\mathrm{r_B})$
  - B : $\mathrm{D}_k(\mathrm{E_k}(\mathrm{r_B}))$ 후, $\mathrm{r_B}$ 확인



Challenge-Response 인증 (cont'd)

■ 대칭키를 이용한 질의-응답 인증

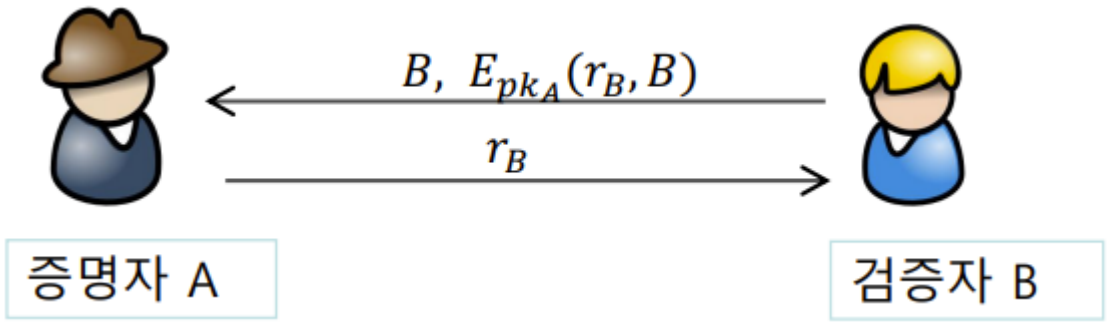
- 반사 공격 (Reflection Attack)
  - A(E) → B : r_A {A(E)의 challenge}
  - B → A(E) : $E_k(r_A)||r_B$ {응답 & B의 challenge}
  - A(E) → B : r_B {E의 challenge, 두 번째 세션 open}
  - B → A(E) : $E_k(r_B)||r_{B'}$ {응답 & B의 challenge}
  - A(E) → B : 첫 번째 세션의 정당한 응답 $E_k(r_B)$ 전송



Challenge-Response 인증 (cont'd)

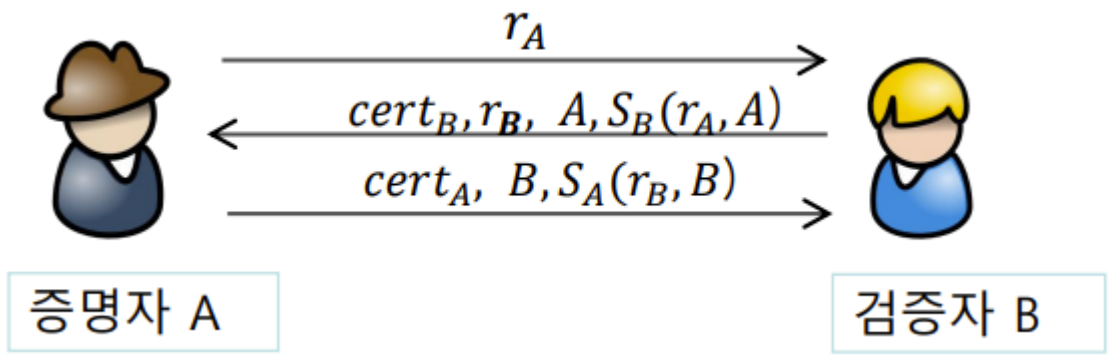
■ 공개키 암호를 이용한 단방향 인증

- 난수를 이용한 인증
  - B → A : $B, E_{pk_A}(r_B, B)$ {B의 challenge}
  - A → B : r_B



■ 전자 서명을 이용한 양방향 인증

- 난수를 이용한 양방향 인증
  - A → B : r_A {A의 challenge}
  - B → A : $cert_B, r_B, A, S_B(r_A, A)$ {B의 응답 & challenge}
  - A → B : $\mathrm{cert_A}, \mathrm{B}, \mathrm{S_A}(\mathrm{r_B}, \mathrm{B})$ {A의 응답 & challenge}



KMU 국민대학교 KOOKMIN UNIVERSITY

■ Performance index of biometry techniques

- Accuracy
  - 올바르게 예측된 데이터의 수를 전체 데이터의 수로 나눈 값

$$\frac{TruePositives + TrueNegatives}{TruePositives + TrueNegatives + FalsePositives + FalseNegatives}$$

- Recall
  - 실제로 True인 데이터를 모델이 True라고 인식한 데이터의 수

$$\frac{TruePositives}{TruePositives + FalseNegatives}$$

- Precision
  - 모델이 True로 예측한 데이터 중 실제로 True인 데이터의 수

$$\frac{TruePositives}{TruePositives + FalsePositives}$$

- F1 Score
  - Precision과 Recall의 조화평균 (Recall과 Precision은 Trade-off 관계에 있음)

$$2 * \frac{Precision * Recall}{Precision + Recall}$$

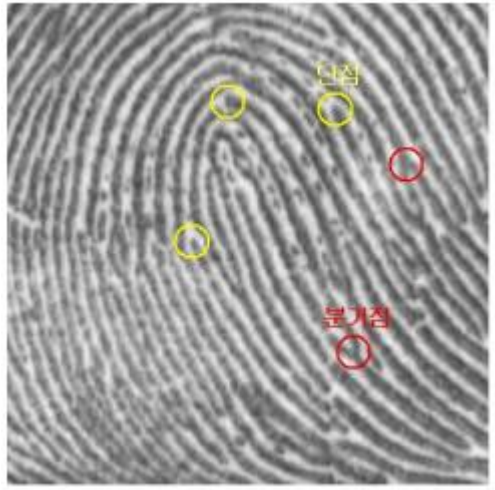
- False Rejection Rate(FRR) → FP / (FP + TN)
  - 원래는 정상이나 이상치 탐지 모델에 의해 이상치로 잘못 판별된(rejected) 비율
- False Acceptance Rate(RAR) → FN / (TP + FN)
  - 원래는 이상치라서 탐지가 되어야 하나 모델에 의해 탐지가 되지 못하고 정상으로 판별된(accepted) 비율

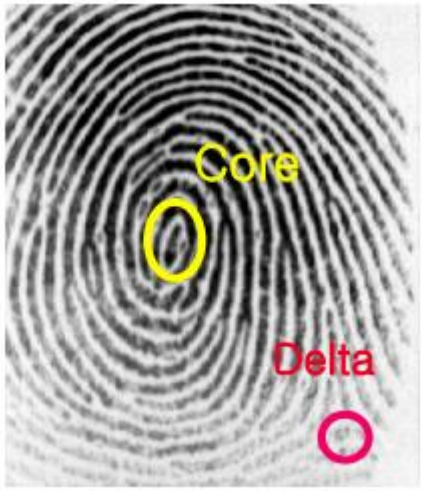
		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

생체 인증 (cont'd)

■ **지문**

- 다른 두 손가락의 지문은 상이
- 지문의 모양은 평생 바뀌지 않음
- 특징점(minutiae) 추출
  - 단점(ending point): 융선이 끊어지는 점
  - 분기점(bifurcation point): 융선이 갈라지는 점
  - Core: 융선 회전의 끝 부분
  - Delta: 융선의 흐름이 세 방향으로 모아져 하나가 되는 점





Thank You

- **Lab:** https://mose.kookmin.ac.kr
- **Email:** sh.jeon@kookmin.ac.kr