2023/02/10, 2023 겨울방학 보안 기초 세미나

Network Security Essentials

- Chapter_3 공개키 암호와 메시지 인증 -

김 지 혜(jihye@pel.sejong.ac.kr)

세종대학교 프로토콜공학연구실

목 차

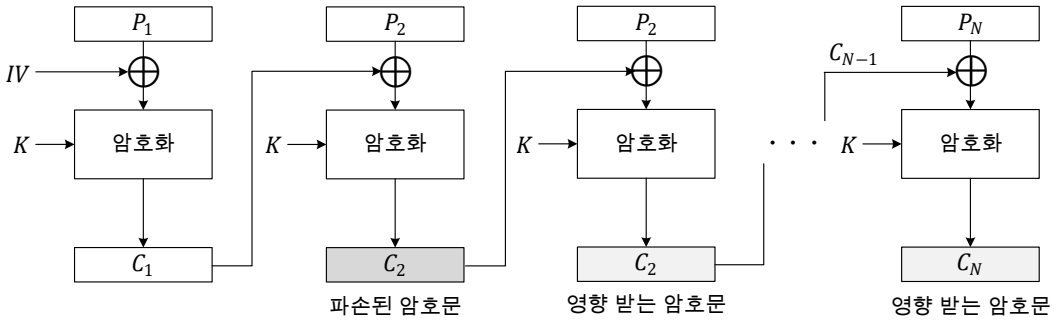
- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘

보충

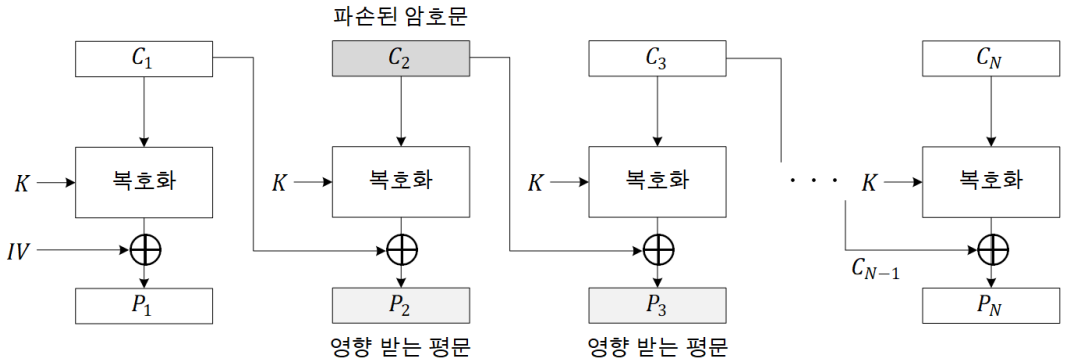
- 스트림 암호와 RC4
  - RC4 알고리즘
    - 정의
      - Ron Rivest가 설계한 바이트 단위의 가변적인 키를 사용하는 스트림 암호 알고리즘
    - 특징
      - 연산 속도가 빠름
        - 한 바이트 출력을 위해 8~16번의 연산 수행
      - SSL/TLS(Secure Socket Layer/Transport Layer Security) 표준에서 사용
        - 웹 브라우저와 서버 간 통신 표준
      - WEP(Wired Equivalent Privacy) 프로토콜과 WPA(WiFi-Protocol Access) 프로토콜에서 사용
        - 무선랜 표준(IEEE 802.11)

보충

- 암호 블록 운용 모드
  - CBC(Cipher-Block Chaining)
    - CBC 모드에 대한 공격
      - 암호화 시, 암호문 블록 1개가 파손된 경우, 다음 암호문 블록에 영향



      - 복호화 시, 암호문 블록 1개가 파손된 경우, 평문 블록 2개에 영향



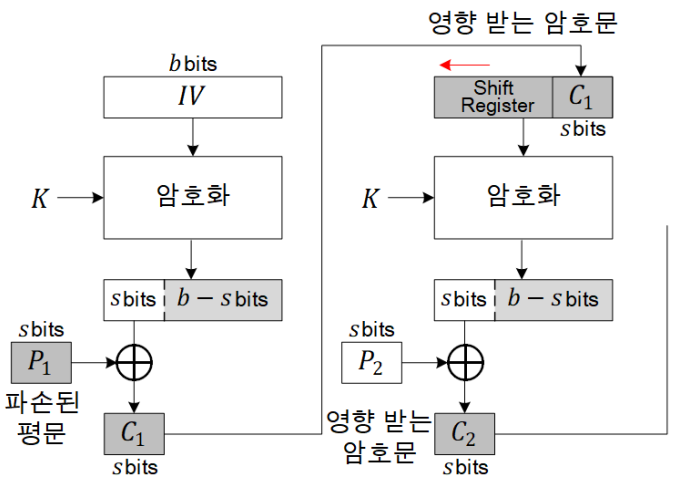
보충

- 암호 블록 운용 모드
  - CFB(Cipher-FeedBack)
    - 정의
      - 이전 암호문 블록을 암호화한 후 평문 블록과 XOR 연산하는 암호화 하는 방식
    - 특징
      - 초기화 벡터(IV, Initialization Vector)를 가짐
        - 키와 마찬가지로 송수신자가 서로 알고 있어야 함
      - 패딩(Padding) 불필요함
        - 스트림 암호화처럼 구성하므로 평문과 암호문 길이 동일
      - 암호 알고리즘만 사용
      - 암호화는 순차적, 복호화는 병렬적 처리
      - 오류가 확산됨

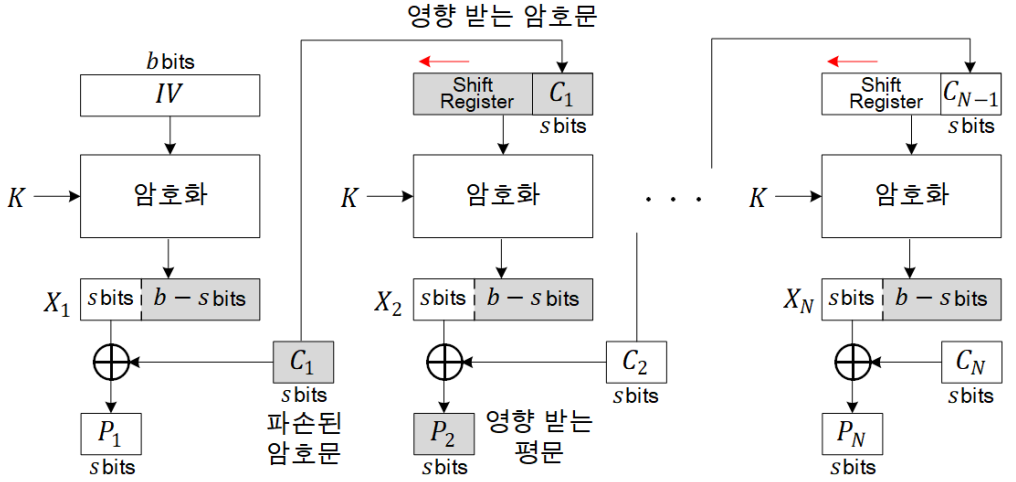
보충

- 암호 블록 운용 모드
  - CFB(Cipher-FeedBack)
    - CFB에 대한 공격
      - 암호화 시 평문 블록 1개가 파손된 경우, 이후의 입력 레지스터에서 해당 암호문 부분이 소멸되기 전까지 영향
      - 복호화 시 암호문 블록 1개가 파손된 경우도 동일함

<암호화 과정>



<복호화 과정>



목 차

- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘

메시지 인증 방법

- 메시지 인증(Message Authentication)
  - 정의
    - 송수신자가 주고받은 메시지의 신원을 확인하는 절차
  - 특징
    - 메시지 내용이 변조되지 않음을 확인 가능
    - 송신자의 신원 확인 가능
      - 메시지 재전송 여부 및 전송 순서
    - 메시지 인증 코드와 해시 함수 이용
  - 방법
    - 대칭키를 이용한 메시지 인증
    - 암호화 없이 메시지 인증

메시지 인증 방법

- 대칭키를 이용한 메시지 인증
  - 정의
    - 메시지를 암호화하여 인증하는 방식
  - 특징
    - 송수신자가 동일한 키를 가지고 있어야 함
    - 체크섬(Checksum) 필드를 통해 메시지 변조 여부 확인 가능
    - 순서 번호 필드를 통해 메시지 전송 순서 변경 여부 확인 가능
    - 타임스탬프(Timestamp) 필드를 통해 메시지가 고의적으로 지연되지 않음을 확인 가능
    - 무결성 및 기밀성 보장

메시지 인증 방법

- 암호화 없이 메시지 인증
  - 정의
    - 메시지 자체를 암호화하지 않고 인증하는 방식
  - 특징
    - 인증 태그(Authentication Tag)를 붙여서 전송
      - 메시지 인증 코드를 비교하여 인증
      - 해시 함수로 출력된 메시지 다이제스트를 이용하여 인증
    - 기밀성 보장하지 않음
      - 기밀성 없는 메시지 인증을 사용하는 경우 존재
        - e.g., 동일 메시지를 다수의 수신자에게 브로드캐스트하는 경우, 한 장비에 과부하로 인해 수신 메시지 복호화가 어려운 경우
    - 평문 메시지만으로 인증 가능
      - 인증에 대한 확신이 필요한 경우, 인증 태그로 무결성 검사

목 차

- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘

해시 알고리즘

- 해시 함수(Hash Function)
  - 정의
    - 임의의 길이의 데이터를 고정 길이의 데이터인 메시지 다이제스트(Message Digest)로 변환하는 함수
  - 특징

특징	설명
메시지 압축 (Message Digest)	• 가변 길이의 원본 메시지에 대해 고정 길이 해시값 생성
계산 용이성	• x가 주어진 경우, $h(x)$는 계산적으로 용이해야 함
일방향성 (One-Way Property)	• $h(x) = y$를 만족하는 x를 찾는 것은 계산적으로 불가능해야 함
약한 충돌 저항성 (Weak Collision Resistance)	• x가 주어진 경우, $h(x) = h(y)$인 $y(\neq x)$를 찾는 것은 계산적으로 불가능해야 함 • 한 입력 값을 알 때, 같은 해시 값을 생성하는 다른 입력 값을 찾는 것이 어려워야 한다는 의미
강한 충돌 저항성 (Strong Collision Resistance)	• $h(x) = h(y)$인 서로 다른 임의의 두 입력 x와 y를 찾는 것은 계산적으로 불가능함 • 같은 해시 값을 생성하는 두 개의 입력 값을 찾는 것이 어려워야한다는 의미

해시 알고리즘

- 단순 해시 함수(Simple Hash Function)
  - 정의
    - 비트 단위로 XOR 연산하는 간단한 형태의 해시 함수
  - 세로 덧붙임 검사(Longitudinal Redundancy Check)
    - 각 블록의 비트별로 XOR 연산
    - $C_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im}$
      - $C_i =$ 해시코드의 i번째 비트
      - $m =$ 입력의 n비트 블록의 수
      - $b_{ij} = j$번째 블록의 i번째 비트

	비트 1	비트 2	· · ·	비트 n
블록 1	b_{11}	b_{21}		b_{n1}
블록 2	b_{12}	b_{22}		b_{n2}
	⋮	⋮	⋮	⋮
블록 m	b_{1m}	b_{2m}		b_{nm}
해시 코드	C_1	C_2		C_n

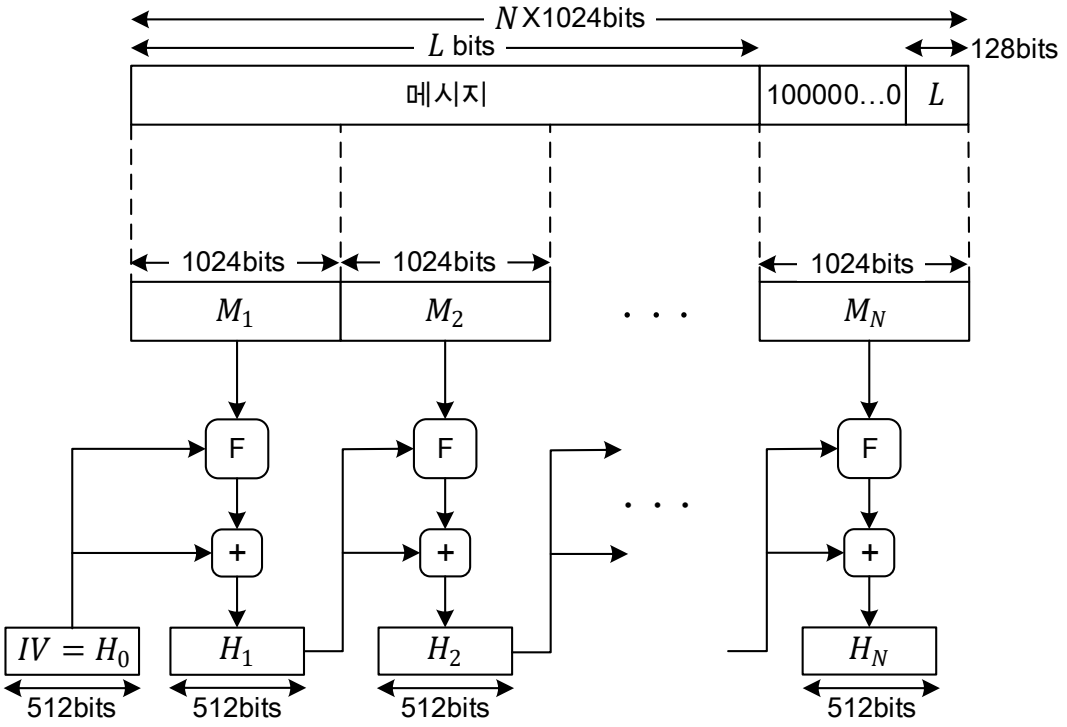
해시 알고리즘

- SHA(Secure Hash Algorithm) 안전 해시 함수
  - 정의
    - NIST에서 표준으로 채택한 암호학적 해시 함수 모음
  - 종류

<table>
<tr><th colspan="2">알고리즘</th><th>해시 길이</th><th>단계 수</th><th>최대 길이</th><th>충돌 발견</th><th>발표연도</th></tr>
<tr><td colspan="2">SHA-0</td><td>160</td><td rowspan="2">80</td><td rowspan="5">$2^{64}-1$</td><td rowspan="2">O</td><td>1993</td></tr>
<tr><td colspan="2">SHA-1</td><td>160</td><td>1995</td></tr>
<tr><td rowspan="4">SHA-2</td><td>SHA-224</td><td>224</td><td rowspan="2">64</td><td rowspan="4">X</td><td rowspan="4">2001</td></tr>
<tr><td>SHA-256</td><td>256</td></tr>
<tr><td>SHA-384</td><td>384</td><td rowspan="2">80</td><td rowspan="2">$2^{128}-1$</td></tr>
<tr><td>SHA-512</td><td>512</td></tr>
</table>

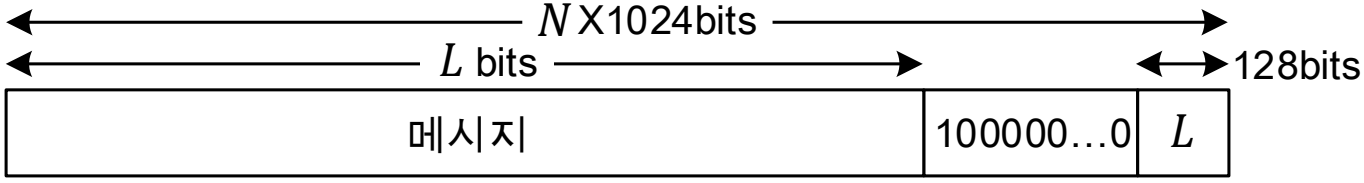
해시 알고리즘

- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(1/7)
    1. 메시지(M) 입력
    2. 메시지(M)를 1024비트 정수배로 패딩 처리
    3. 메시지(M)를 N개의 블록으로 구분
    4. 초기화벡터(IV) 생성
    5. 이전 해시값(H_{i-1})과 메시지 블록(M_i)을 F함수로 실행
    6. 해시값(H_i) 생성
    7. 4~5를 N번 수행
    8. 최종 해시값(H_i) 출력



해시 알고리즘

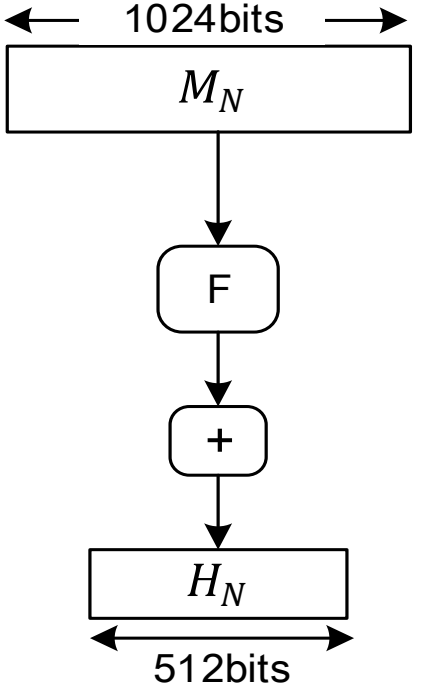
- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(2/7)
    1. 패딩 비트 붙이기(Appending Padding bits)
       - 메시지(M)를 1024비트로 정수배에서 128비트를 뺀만큼 패딩 처리
       - 비트 패딩 방식으로 처리됨
         - 첫 비트 1, 나머지 비트 0의 형태
    2. 길이 붙이기(Append Length)
       - 패딩된 메시지(M)에 L을 128비트로 추가
         - L은 메시지(M)의 길이



해시 알고리즘

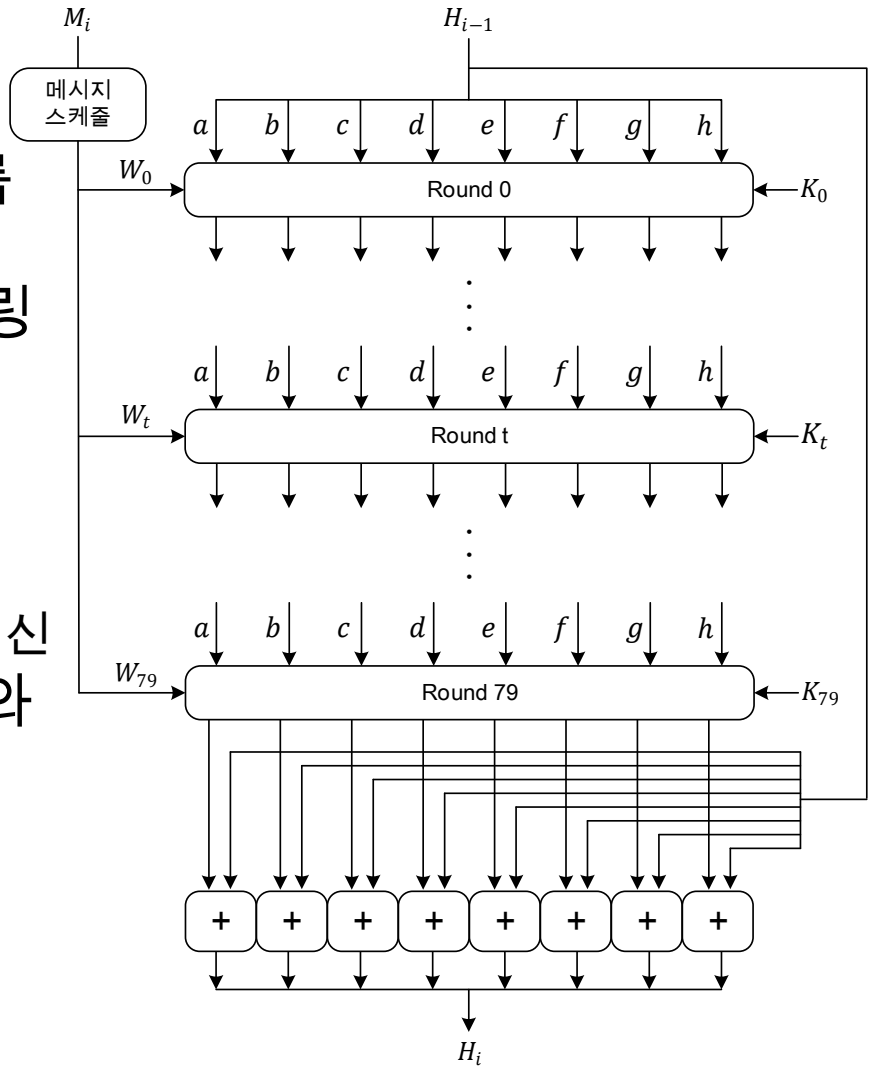
- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(3/7)
    3. MD 버퍼 초기화(Initialize Message Digest Buffer)
       - 512비트 MD 버퍼는 해시 함수의 중간 값(H_i)과 최종 값(H_N)을 저장하기 위해 사용
         - 64비트 레지스터(a,b,c,d,e,f,g,h) 8개로 버퍼 구성
       - 최초 MD 버퍼(H_0)는 소수의 제곱근의 소숫점 이하 처음 64비트를 16진수화하여 구성
         - e.g., a은 $\sqrt{2}$를, b는 $\sqrt{3}$

a = 6A09E667F3BCC908	e = 510E527FADE682D1
b = BB67AE8584CAA73B	f = 9B05688CEB3E6C1F
c = 3C6EF372FE94F82B	g = 1F83D9ABFB41BD6B
d = A54FF53A5F1D36F1	h = 5BE0CDI9137E2179



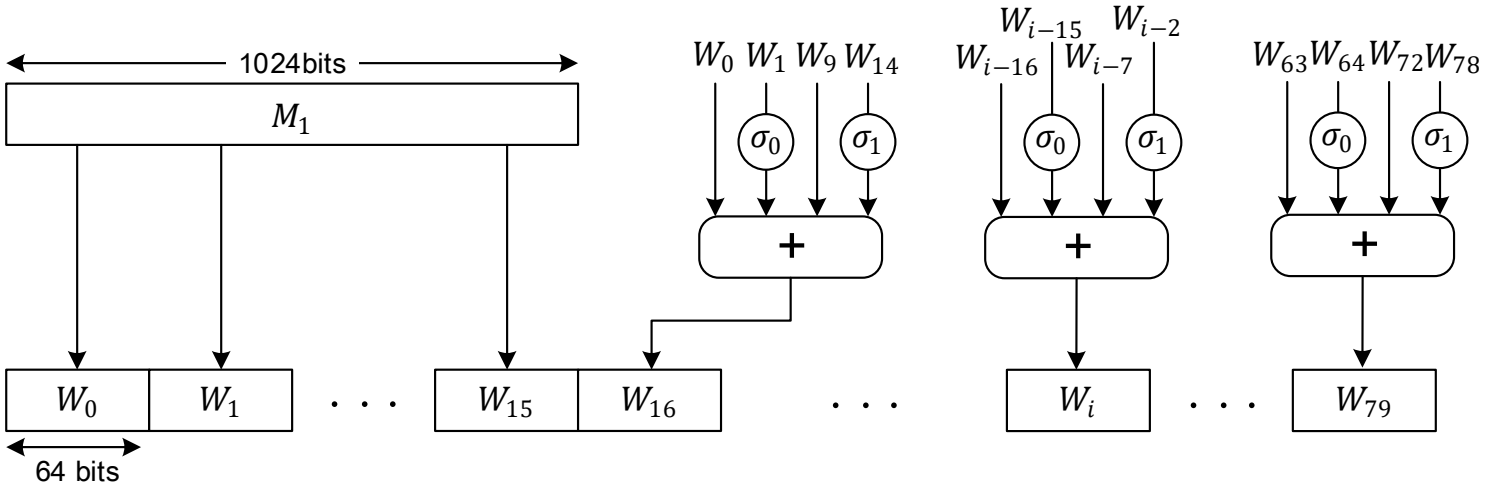
해시 알고리즘

- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(4/7)
    4. F함수 실행
       - 메시지(M)을 N개의 1024비트 블록으로 구분
       - 메시지 블록(M_i)을 메시지 스케줄링 하여 80개의 워드 블록(W_i) 생성
       - 메시지 블록(M_i), 워드 블록(W_i), 키(K_i), 이전 MD 버퍼(H_{i-1})로 80라운드 수행
         - 매라운드 사용되는 버퍼 값(a~h)은 갱신
       - 80라운드 후, 이전 MD 버퍼(H_{i-1})와 XOR 연산하여 MD 버퍼(H_i) 생성



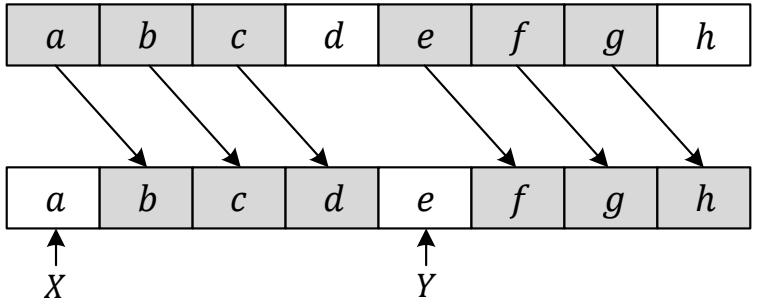
해시 알고리즘

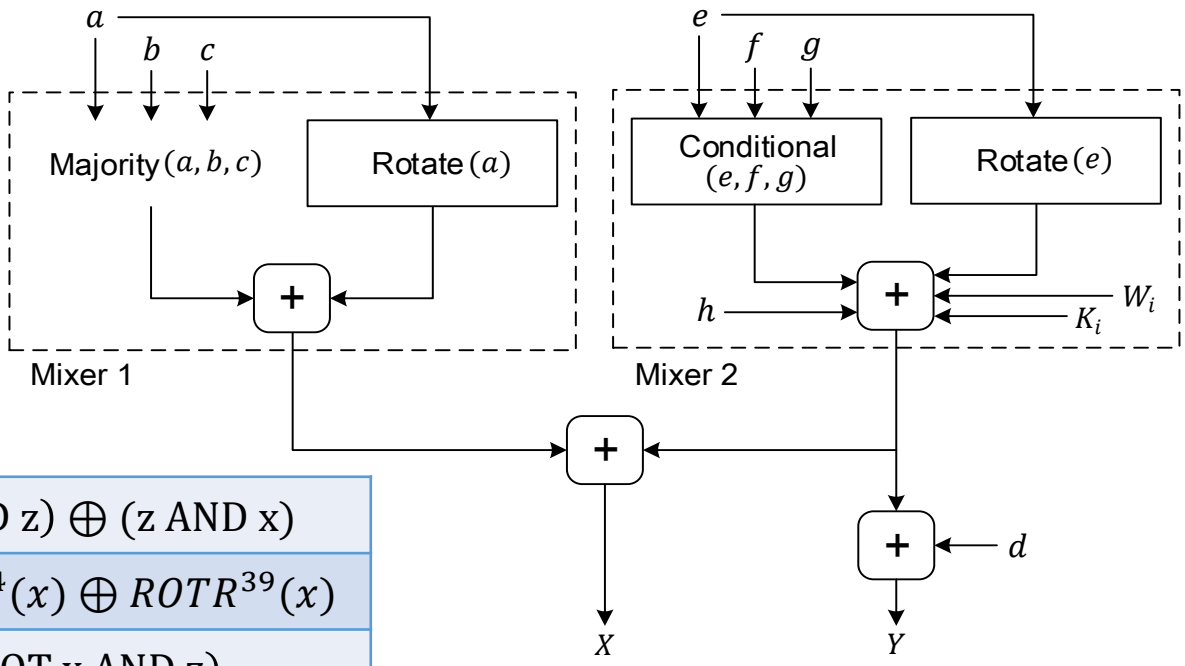
- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(5/7)
    4. F함수 실행
       - 메시지 스케줄링
         - 메시지 블록(M_i)으로 16개의 워드 블록(W_i) 생성
         - 나머지 64개의 워드 블록(W_i)은 $W_{i-16} \oplus W_{i-15} \times \sigma_0 \oplus W_{i-7} \oplus W_{i-2} \times \sigma_1$
           - $\sigma_0(x) = ROTR^1(x) \oplus ROTR^8(x) \oplus SHR^7(x)$
           - $\sigma_1(x) = ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^6(x)$
           - $ROTR^n(x)$은 x를 오른쪽으로 n번 순환 이동하는 것
           - $SHR^n(x)$은 x 를 왼쪽으로 n번 이동한 후, 남은 부분을 0으로 패딩



해시 알고리즘

- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(6/7)
    4. F함수 실행
       - 라운드 과정
         - 키(K_i)는 소수의 제곱근의 소숫점 이하 처음 64비트로 구성

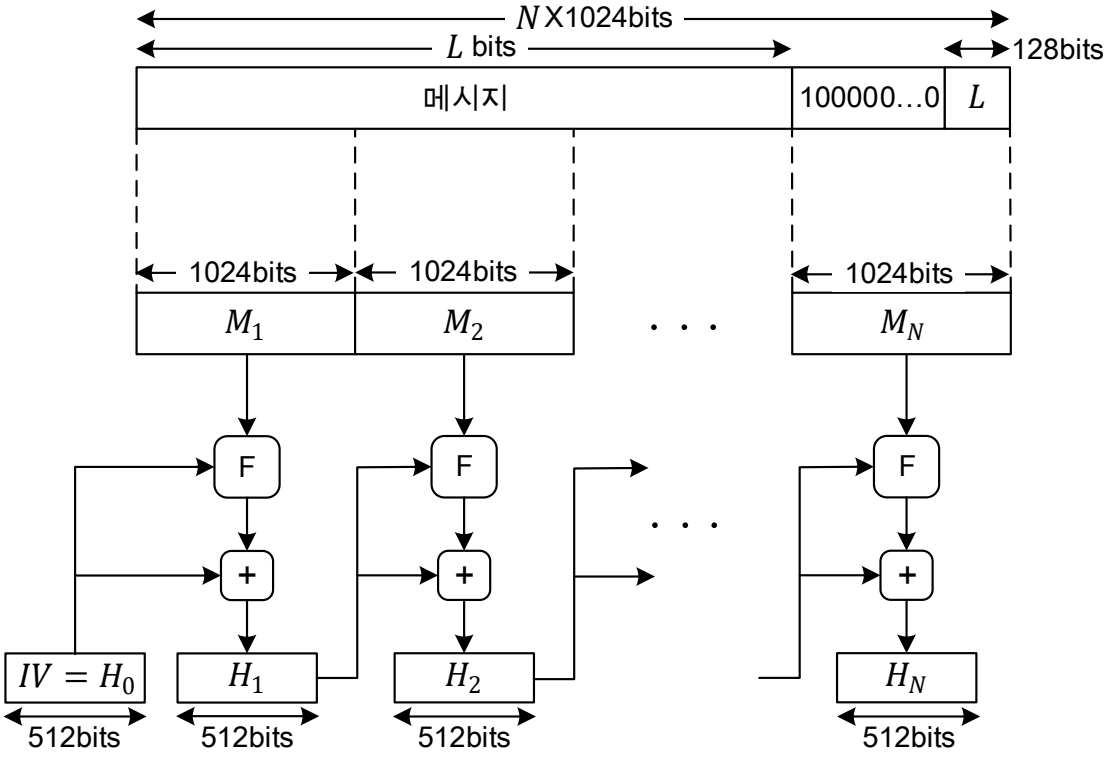




Majority(x, y, z)	$(x \text{ AND } y) \oplus (\text{y AND z}) \oplus (\text{z AND x})$
Rotate(x)	$ROTR^{28}(x) \oplus ROTR^{34}(x) \oplus ROTR^{39}(x)$
Conditional(x, y, z)	$(x \text{ AND } y) \oplus (\text{NOT x AND z})$

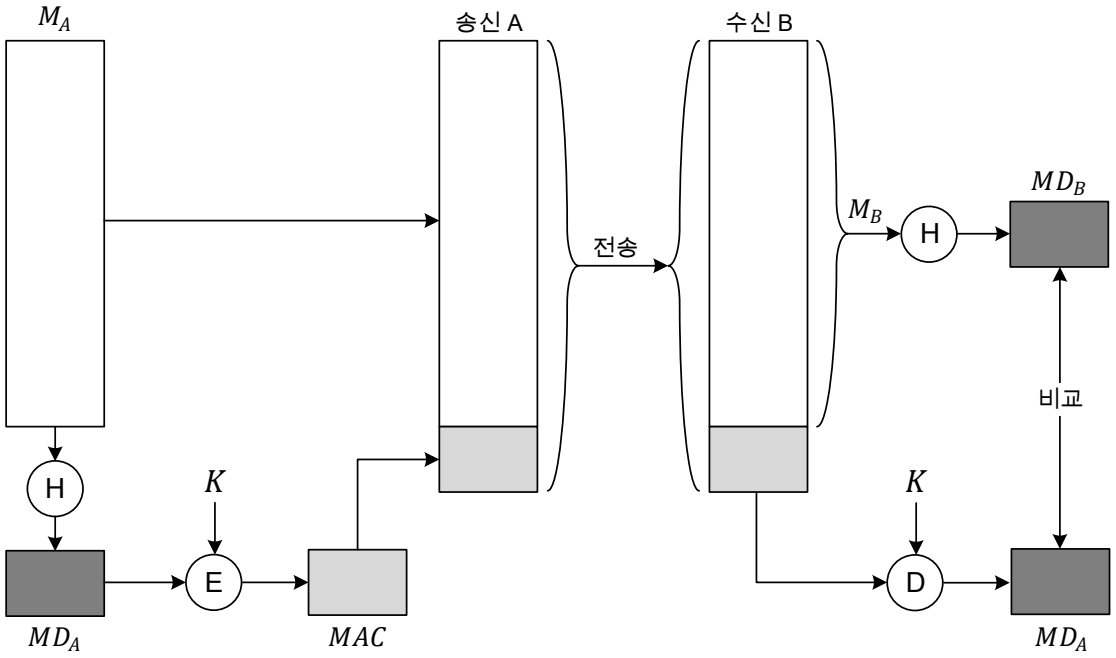
해시 알고리즘

- SHA(Secure Hash Algorithm) 안전 해시 함수
  - SHA-512 연산 과정(7/7)
    5. 출력(Output)
       - N개의 메시지 블록(M_i)을 연산한 후, 마지막에 512비트의 MD 버퍼(H_N) 출력



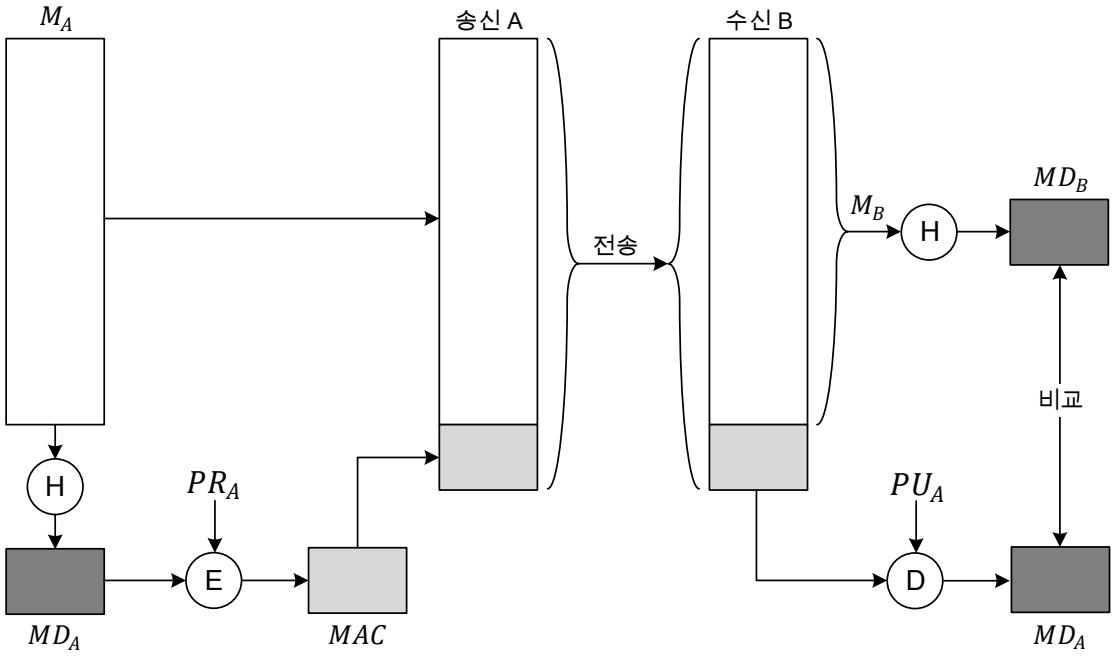
해시 알고리즘

- 해시 함수를 이용한 메시지 인증
  - 대칭키를 이용한 메시지 인증
    1. 송신 A에 인증 코드 추가
       - $MD_A = H(M_A)$
       - $MAC = E(K, MD_A)$
    2. 수신 B에서 인증 코드 추출
       - $MD_B = H(M_B)$
       - $MD_A = D(K, MAC)$
    3. 송신 A와 수신 B의 인증 코드 비교



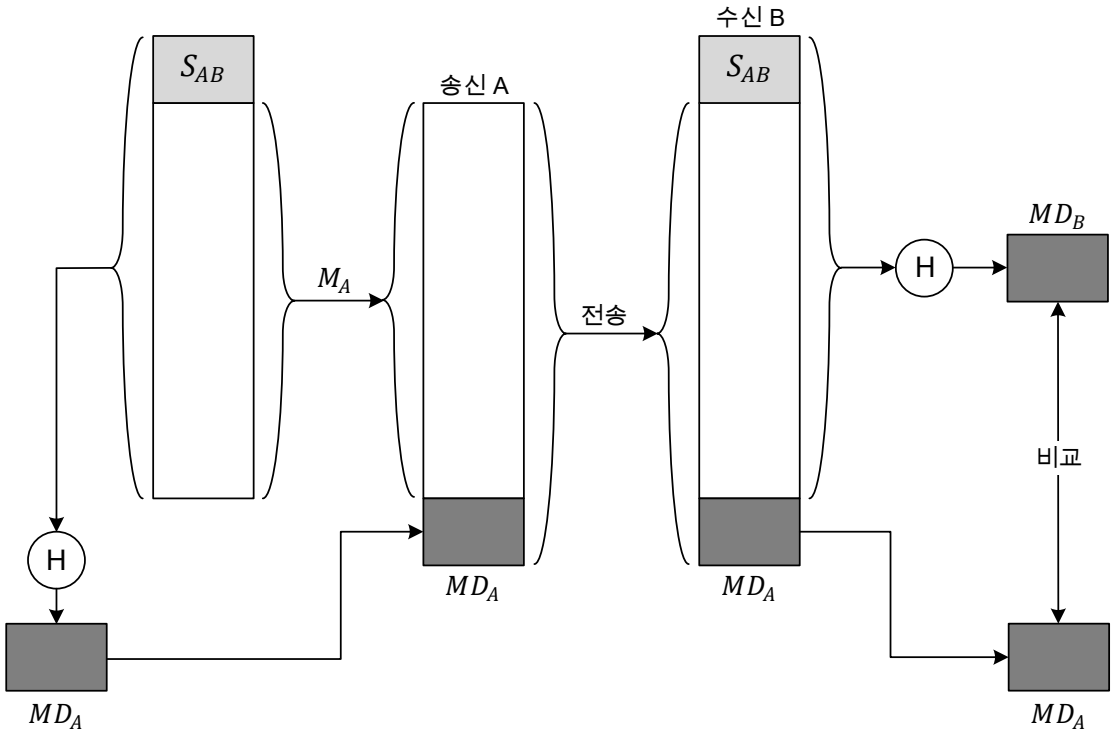
해시 알고리즘

- 해시 함수를 이용한 메시지 인증
  - 공개키를 이용한 메시지 인증
    1. 송신 A에 인증 코드 추가
       - $MD_A = H(M_A)$
       - $MAC = E(PR_A, MD_A)$
    2. 수신 B에서 인증 코드 추출
       - $MD_B = H(M_B)$
       - $MD_A = D(PU_A, MAC)$
    3. 송신 A와 수신 B의 인증 코드 비교



해시 알고리즘

- 해시 함수를 이용한 메시지 인증
  - 비밀키를 이용한 메시지 인증
    1. 송신 A에 인증 코드 추가
       - $MD_A = H(M_A || S_{AB})$
    2. 수신 B에서 인증 코드 추출
       - $MD_B = H(M_B || S_{AB})$
    3. 송신 A와 수신 B의 인증 코드 비교



목 차

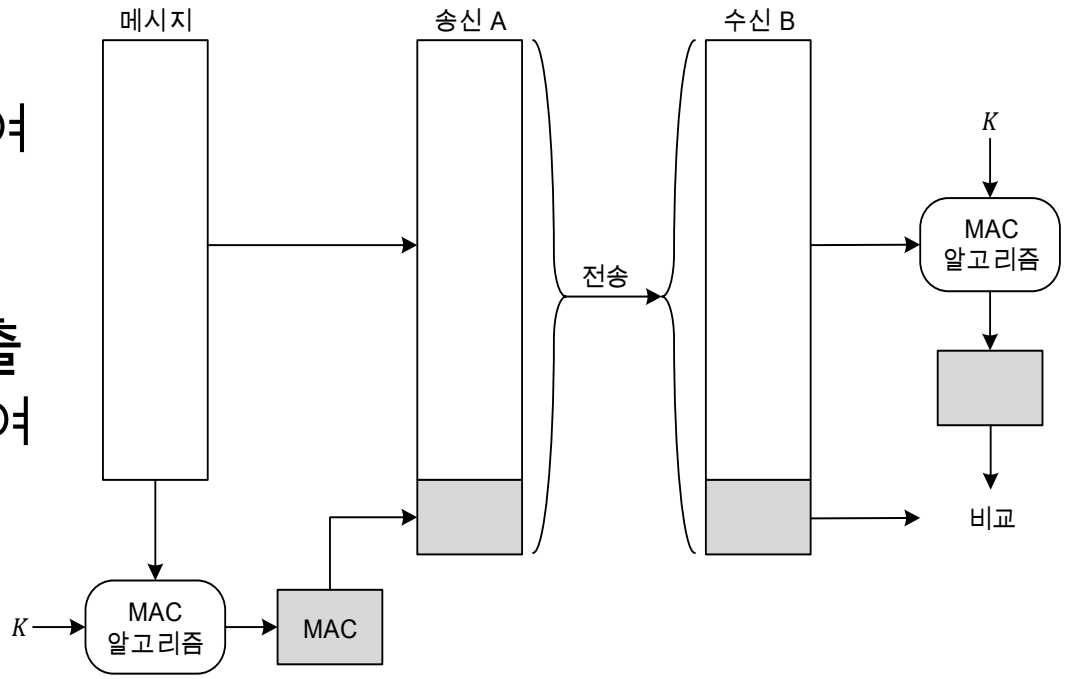
- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘

메시지 인증 코드

- 메시지 인증 코드(MAC, Message Authentication Code)
  - 정의
    - 해시 함수와 대칭키를 활용하여 인증하는 기술
  - 특징
    - 송신자가 생성하여 보낸 코드와 수신자가 계산한 코드가 동일한지 확인하여 인증
    - 메시지 무결성과 부인 방지를 보장

메시지 인증 코드

- 메시지 인증 코드(MAC, Message Authentication Code)
  - 연산 과정
    1. 송신 A에 인증 코드 추가
       - 메시지(M_A)와 키 K를 이용하여 MAC 알고리즘(F) 수행
       - $MAC = F(K, M_A)$
    2. 수신 B에서 인증 코드 추출
       - 메시지(M_B)와 키 K를 이용하여 MAC 알고리즘(F) 수행
       - $MAC = F(K, M_B)$
    3. 송신 A와 수신 B의 인증 코드 비교



메시지 인증 코드

- HMAC(Hashed Message Authentication Code)
  - 정의
    - 해시 함수를 사용하여 메시지를 인증하는 기술
  - 특징
    - 기존 해시 함수보다 안전한 해시 함수가 있는 경우, 해시 함수 변경 가능

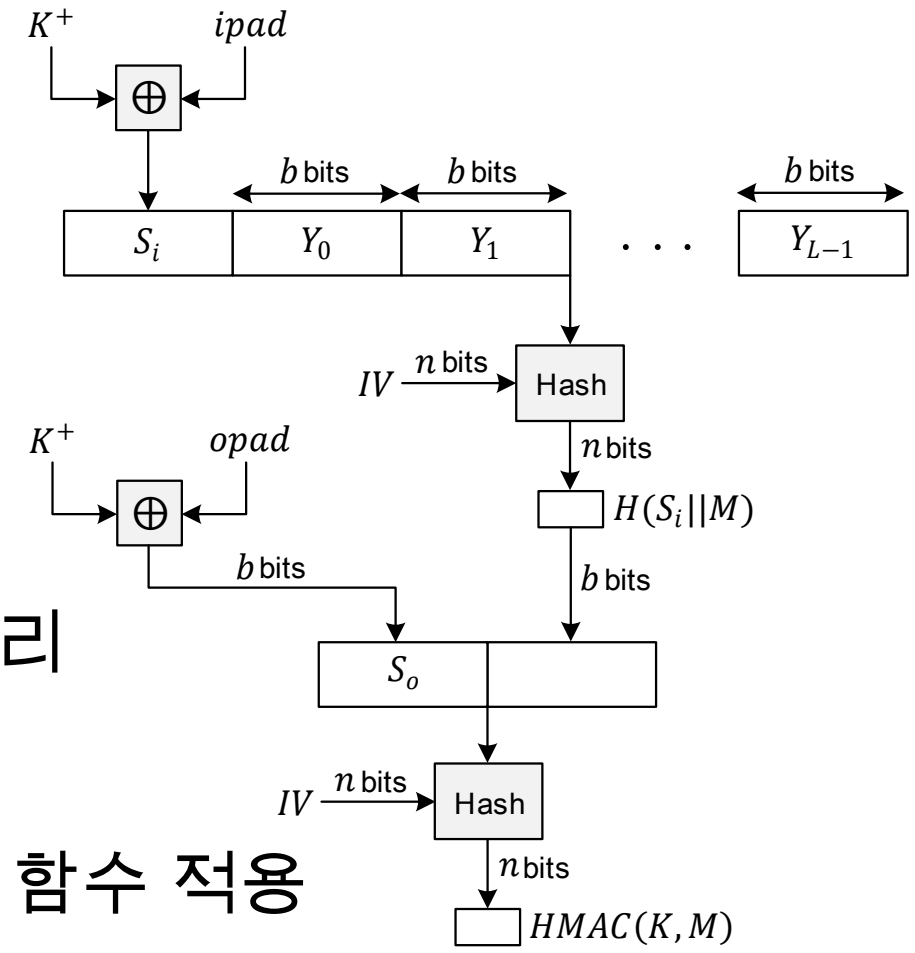
메시지 인증 코드

- HMAC(Hashed Message Authentication Code)
  - 동작 과정 용어

용어	설명
H	내장된 해시 함수
M	HMAC 입력 메시지
Y_i	M의 i번째 블록
L	M의 블록 수
b	블록 비트 수
n	H에 의해 생성된 해시 코드의 길이
K	비밀키(키 길이가 b보다 긴 경우, n비트 키 생성의 입력으로 사용)
K^+	K의 왼쪽에 0을 붙여서 길이가 b비트가 되도록 한 것
$ipad$	00110110을 b/8번 반복한 2진 수열
$opad$	01011100을 b/8번 반복한 2진 수열

메시지 인증 코드

- HMAC(Hashed Message Authentication Code)
  - 동작 과정
    1. b비트 K^+ 생성
       - K의 왼쪽을 0으로 패딩
    2. b비트 블록 S_i 생성
       - $S_i = K^+ \oplus ipad$
    3. 메시지(M)에 S_i 추가 후, 해시 함수 적용
       - $H(S_i||M)$
    4. $H(S_i||M)$를 b비트로 패딩 처리
    5. b비트 블록 S_o 생성
       - $S_o = K^+ \oplus opad$
    6. $H(S_i||M)$에 S_o 추가 후, 해시 함수 적용
       - $HMAC(K, M) = H[S_o||H(S_i||M)]$



메시지 인증 코드

- CMAC(Cipher-based Message Authentication Code)
  - 정의
    - 블록 암호 기술을 활용하여 메시지를 인증하는 기술
  - 특징
    - AES, 3DES와 같은 암호 기술 사용
    - 두 개의 키 사용
      - 기존 키 한 개와 서브키 한 개

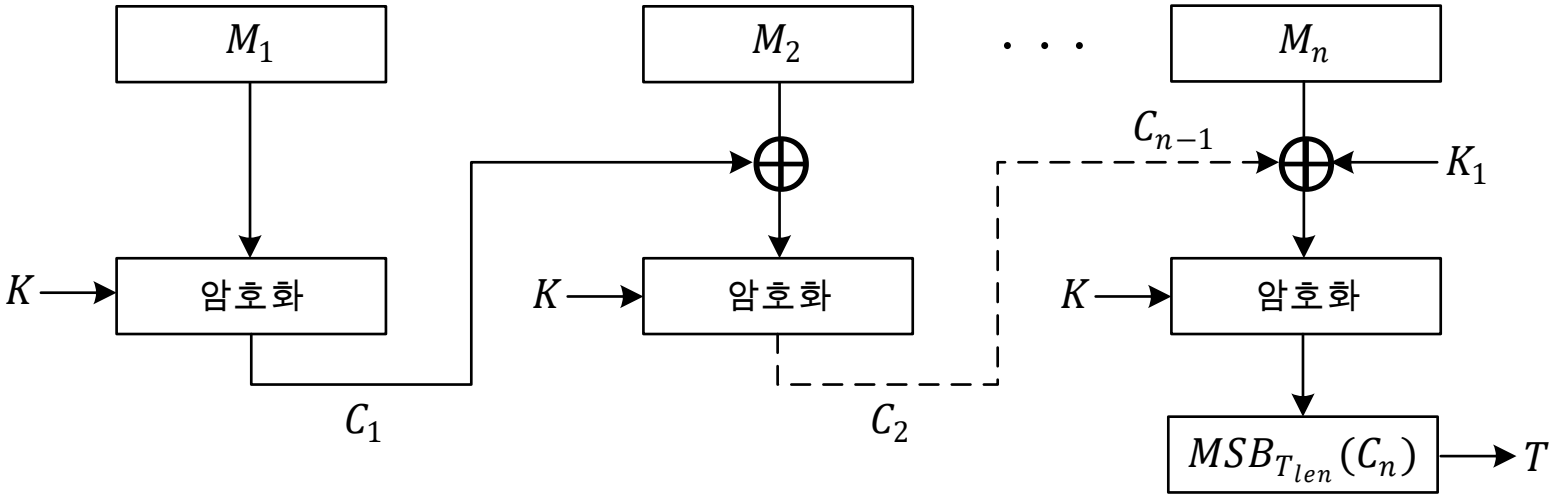
메시지 인증 코드

- CMAC(Cipher-based Message Authentication Code)
  - 동작 과정 용어

용어	설명
T	메시지 인증 코드를 의미하는 인증 태그(Tag)
T_{len}	T의 비트 길이
$MSB_s(X)$	비트열 X의 왼쪽부터 s개의 비트
n	블록 갯수
k	키 길이(e.g., AES: 128, 192, 256, 3DES: 112, 168)
b	블록 길이(e.g., AES: 128, 3DES: 64)
K	k비트의 키
K_1	b비트만큼의 0과 K를 암호화한 후, 왼쪽으로 1번 순환 이동한 서브키
K_2	b비트만큼의 0과 K_1을 암호화한 후, 왼쪽으로 1번 순환 이동한 서브키

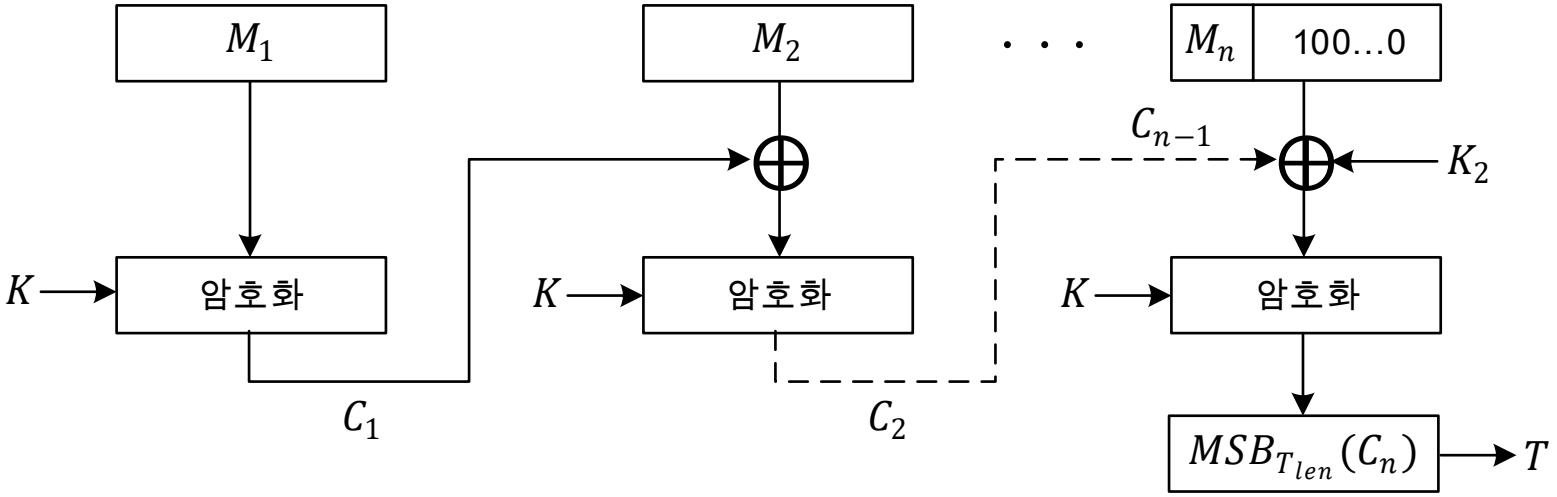
메시지 인증 코드

- CMAC(Cipher-based Message Authentication Code)
  - 동작 과정
    - 메시지 길이가 블록 길이의 정수배인 경우
      - k비트 키(K)와 b비트 서브키(K_1) 사용



메시지 인증 코드

- CMAC(Cipher-based Message Authentication Code)
  - 동작 과정
    - 메시지 길이가 블록 길이의 정수배가 아닌 경우
      - 마지막 메시지 블록을 b비트 만큼 패딩
      - k비트 키(K)와 b비트 서브키(K_2) 사용

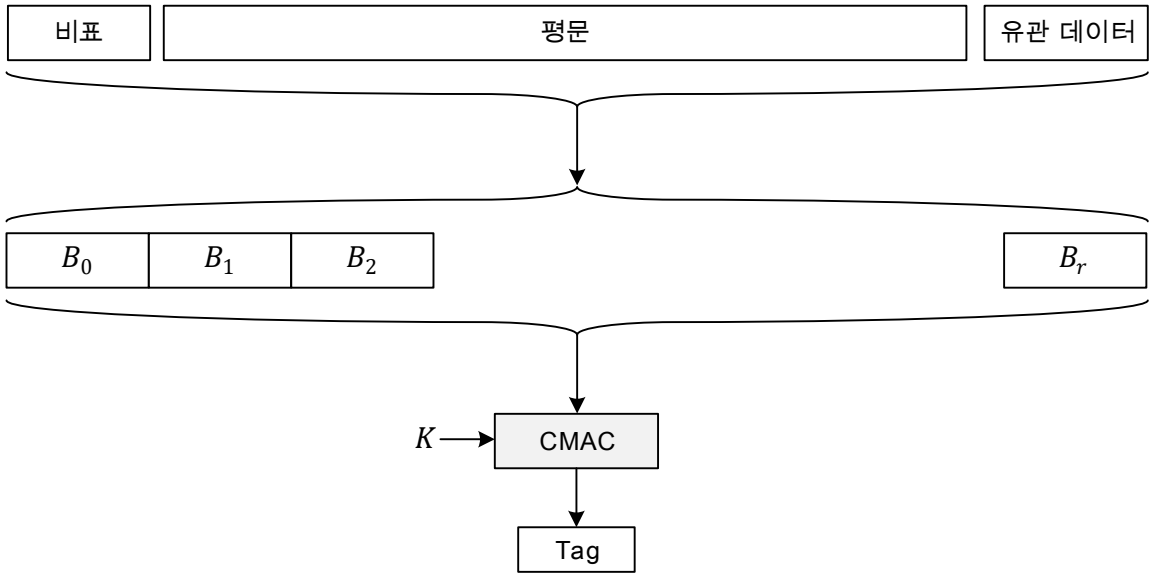


메시지 인증 코드

- CCM(Counter with Cipher Block Chaining-MAC)
  - 정의
    - 기밀성과 무결성을 동시에 보호하는 블록 암호 기반 메시지 인증 기술
  - 특징
    - AES, CTR 모드, CMAC 사용
    - 인증과 암호화에 동일한 하나의 키 사용

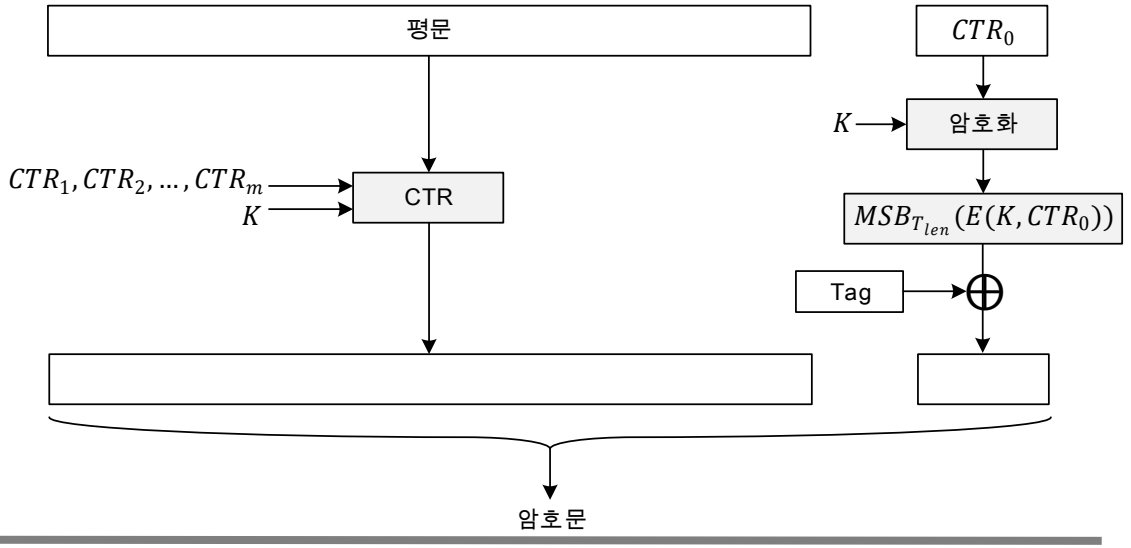
메시지 인증 코드

- CCM(Counter with Cipher Block Chaining-MAC)
  - 인증 과정
    1. 비표, 평문, 유관 데이터를 입력
       - 비표는 매 순간 달라지는 임의의 값
       - 유관 데이터는 인증은 수행하나 암호화는 수행하지 않는 값
    2. r개의 블록으로 구분
    3. CMAC을 통해 메시지 인증 코드(Tag) 생성



메시지 인증 코드

- CCM(Counter with Cipher Block Chaining-MAC)
  - 암호화 과정
    1. m개의 카운터열 생성
    2. 첫 번째 카운터열(CTR_0) 암호화
       - 키 K를 이용하여 CTR 모드로 암호화
       - 왼쪽 T_{len}만큼을 인증 태그(Tag)와 XOR
    3. 평문 블록과 나머지 카운터열(CTR_i) 암호화
       - 키 K를 이용하여 CTR 모드로 암호화
    4. 암호문 출력



목 차

- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘

공개키 암호 원리

- 공개키 암호
  - 정의
    - 송수신자가 서로 다른 키를 가지고 암호화/복호화하는 방식
  - 특징
    - 암호화와 복호화에 사용되는 키가 다름
    - 송수신자 간 키 전송이 불필요함
    - 대칭키 암호보다 키 길이가 김
    - 대칭키 암호보다 암호화/복호화 속도가 느림

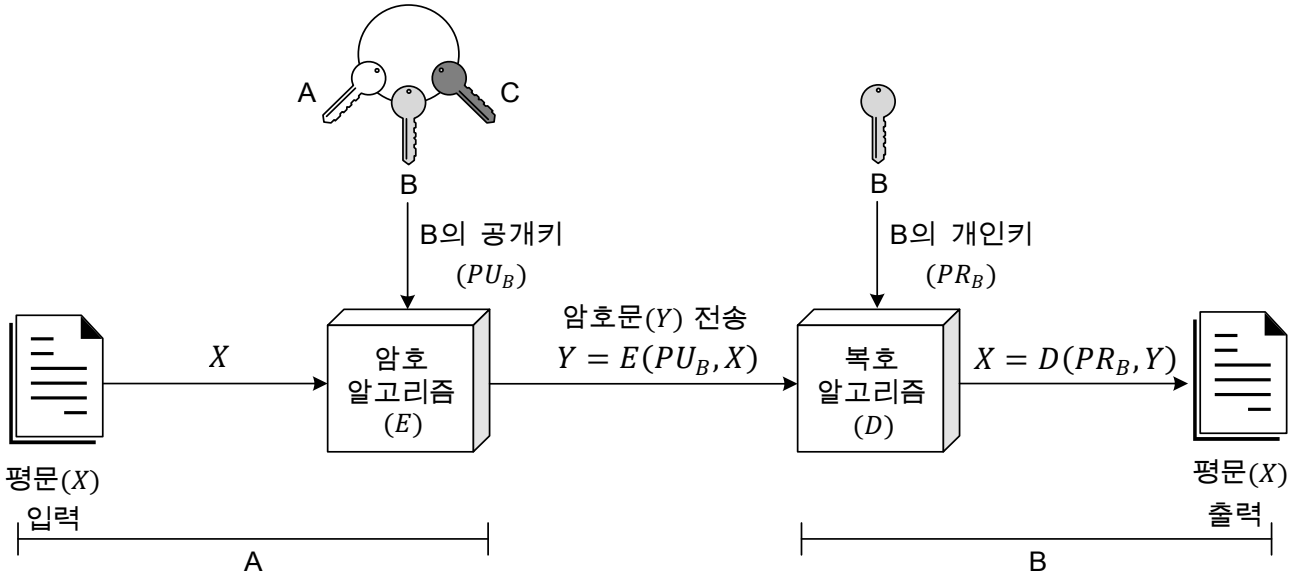
공개키 암호 원리

- 공개키 암호
  - 용어
    - 평문(Plaintext)
      - 전달할 내용을 담은 일반적인 데이터
    - 공개키(Public Key)
      - 개인키와 한 쌍을 이루며, 암호화나 복호화에 사용되는 공개된 키
    - 개인키(Private Key)
      - 공개키와 한 쌍을 이루며, 암호화나 복호화에 사용되는 비공개된 키
    - 암호 알고리즘(Encryption Algorithm)
      - 공개키 혹은 개인키를 활용하여 평문을 암호화하는 방법
    - 암호문(Ciphertext)
      - 암호 알고리즘을 통해 암호화된 평문
    - 복호 알고리즘(Decryption Algorithm)
      - 공개키 혹은 개인키를 활용하여 암호문을 복호화하는 방법

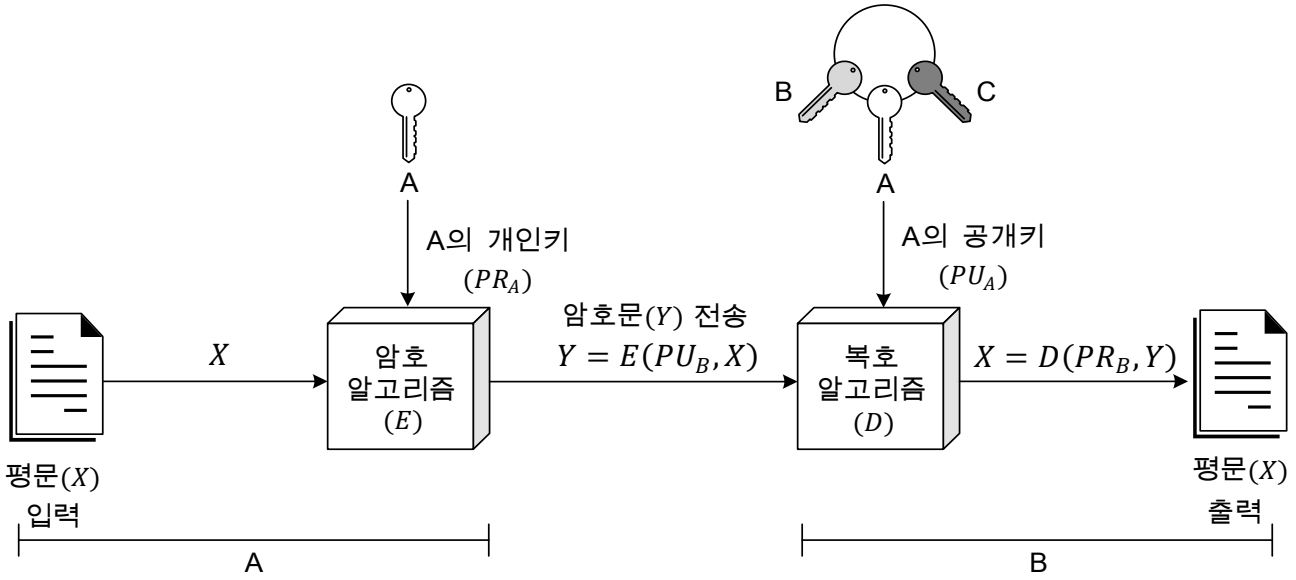
공개키 암호 원리

- 공개키 암호
  - 구조

공개키에 의한 암호화



개인키에 의한 암호화



공개키 암호 원리

- 공개키 암호
  - 요건
    - B가 한 쌍의 공개키(PU_B), 개인키(PR_B)를 생성하기 쉬워야 함
    - 공개키(PU_B)와 평문(M)을 알고 있는 송신자 A는 암호문을 쉽게 구할 수 있어야 함
    - 수신자 B는 암호문을 자신의 개인키(PR_B)로 복호화하는 것이 계산적으로 쉬워야 함
    - 공개키(PU_B)를 아는 공격자가 개인키(PR_B)를 알기 어려워야 함
    - 공개키(PU_B)와 암호문(C)을 아는 공격자가 평문(M)을 알기 어려워야 함

목 차

- 보충
  - 스트림 암호와 RC4
  - 암호 블록 운용 모드
- 메시지 인증 방법
  - 해시 알고리즘
  - 메시지 인증 코드
- 공개키 암호 원리
- 공개키 암호 알고리즘
- 디지털 서명

공개키 암호 알고리즘

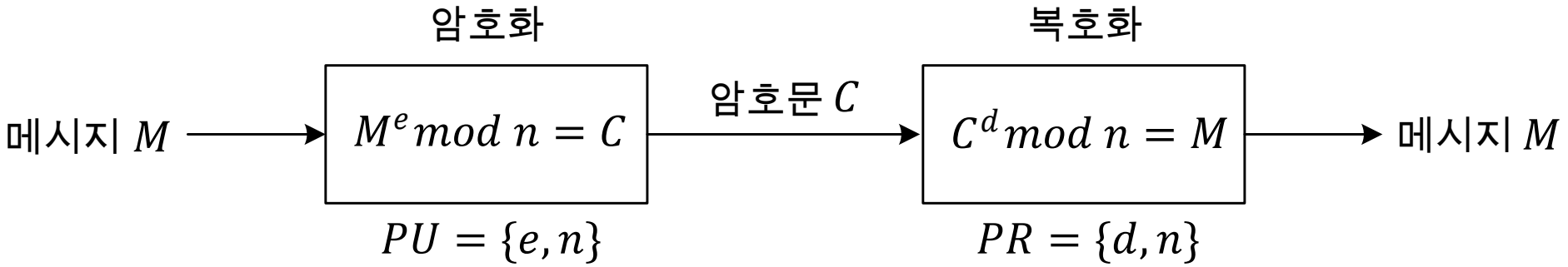
- RSA(Rivest-Shamir-Adleman)
  - 정의
    - 큰 정수를 소인수분해하기 어렵다는 점을 이용한 공개키 암호 알고리즘
  - 특징
    - 공개키 암호화 방식
      - 두 개의 키 사용
    - 디지털 서명이 가능한 최초의 알고리즘
    - SSL/TLS에서 가장 많이 사용
      - 전세계 대부분의 인터넷 뱅킹에서 RSA 암호화 사용

공개키 암호 알고리즘

- RSA(Rivest-Shamir-Adleman)
  - 동작 과정(1/2)
    - 키 생성($PU = \{e, n\},\ PR = \{d, n\}$)
      1. 서로 다른 임의의 두 개의 소수 p와 q 선택
      2. n 계산
         - $n = p \times q$
      3. $\varphi(n)$ 계산
         - $\varphi(n)$은 1부터 $n-1$까지의 양의 정수 중, n과 서로소인 정수의 개수 의미
         - $\varphi(n) = (p-1) \times (q-1)$
      4. 정수 e 계산
         - $\varphi(n)$와 서로소인 정수 e를 구함
         - 서로소는 최대공약수(GCD, Greatest Common Multiple)가 1인 수, $\gcd(\varphi(n), e) = 1$
         - 공개키 $PU = \{e, n\}$ 생성
      5. d 계산
         - $d = e^{-1}$ 이므로 $de \bmod \varphi(n) = 1$
         - 개인키 $PR = \{d, n\}$ 생성

공개키 암호 알고리즘

- RSA(Rivest-Shamir-Adleman)
  - 동작 과정(2/2)
    - 암호화 과정
      - 공개키 $PU = \{e, n\}$ 사용
      - $C = M^e \ mod \ n$
    - 복호화 과정
      - 개인키 $PR = \{d, n\}$ 사용
      - $M = C^d \ mod \ n$



공개키 암호 알고리즘

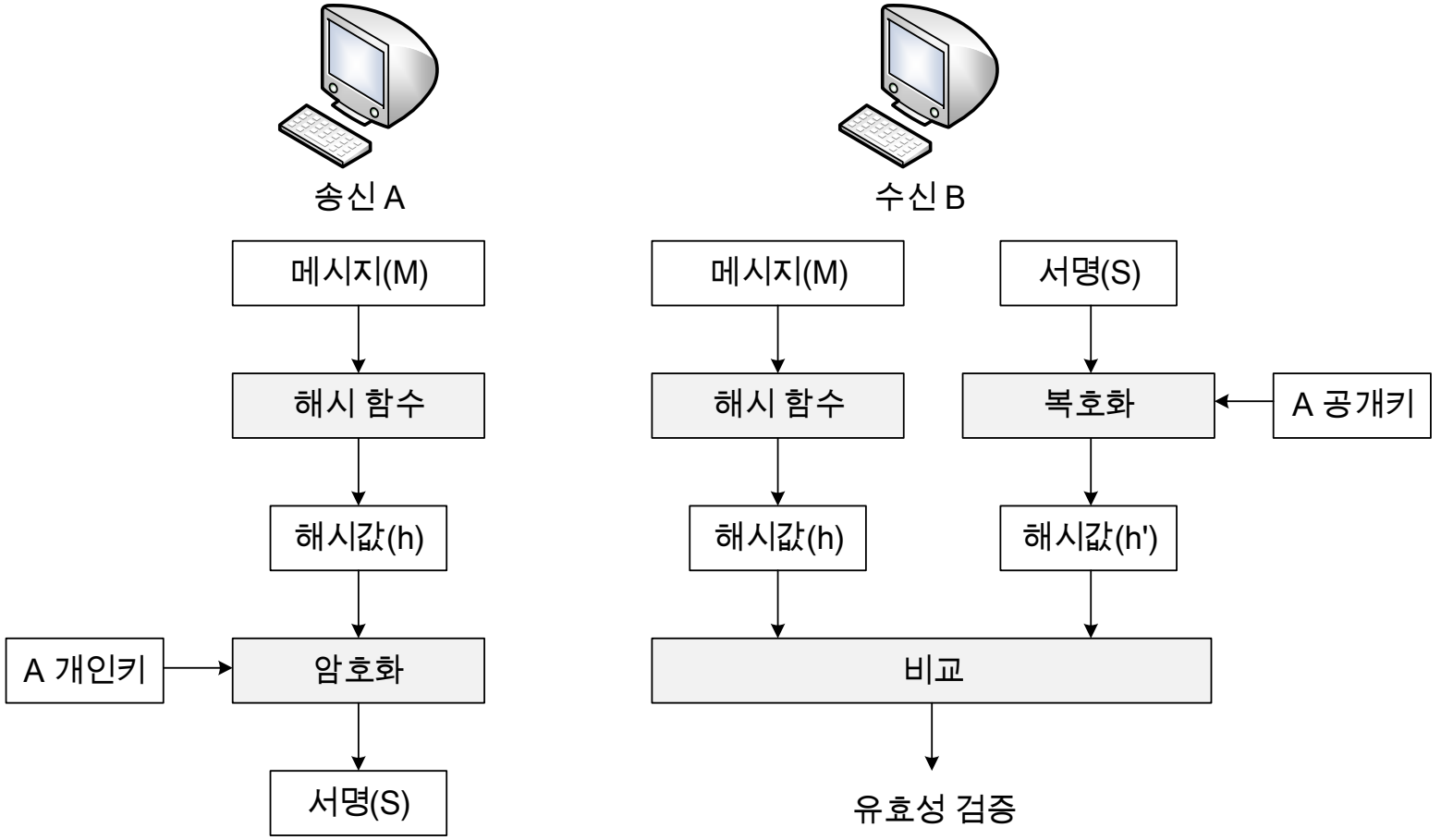
- RSA(Rivest-Shamir-Adleman)
  - 시도 가능한 공격
    - 전수 공격(Brute-Force Attack)
      - 가능한 모든 경우의 개인키로 시도하는 공격
      - e와 d가 클수록 공격이 어려워짐
    - 소인수분해 공격
      - n을 소인수분해하여 소수 p와 q를 통해 키를 구하는 공격
        - 개인키가 유출된 것을 의미
      - n이 클수록 공격이 어려워짐

공개키 암호 알고리즘

- 디지털 서명(Digital Signature)
  - 정의
    - 송신자의 신원을 증명하기 위한 검증 알고리즘
  - 특징
    - RSA 암호화 사용
    - 해시 함수 사용
    - 출처 인증 및 데이터 무결성 검증
      - 서명자의 신분 증명
      - 서명자의 서명 사실에 대한 부인봉쇄 증명
      - 서명한 데이터의 무결성 증명
    - 재사용 불가능
      - 다른 문서의 서명으로 사용 불가

공개키 암호 알고리즘

- 디지털 서명(Digital Signature)
  - 동작 과정

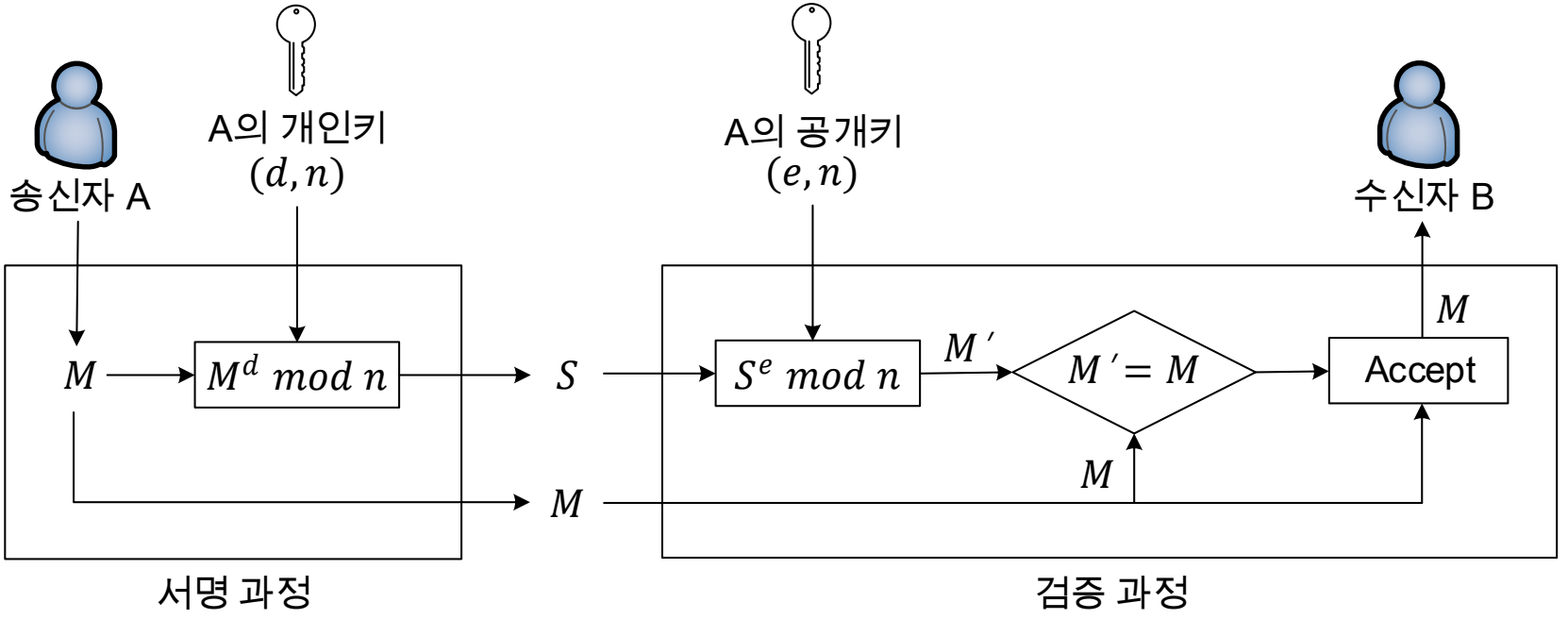


공개키 암호 알고리즘

- RSA 기반 디지털 서명(Digital Signature)
  - 동작 과정(1/2)
    1. 키 생성
       - 두 개의 소수 p, q를 선택
       - $n = p \times q$
       - $\varphi(n) = (p-1) \times (q-1)$
       - 정수 e 선택 후, $de \ mod \ \varphi(n) = 1$를 통해 d 계산
    2. 서명
       - 송신자는 자신의 개인키(d, n)로 메시지(M)의 서명(S) 생성
         - $S = M^d \ mod \ n$
       - 수신자에게 송신자의 공개키(e, n) 전달

공개키 암호 알고리즘

- RSA 기반 디지털 서명(Digital Signature)
  - 동작 과정(2/2)
    3. 검증
       - 수신자는 송신자의 공개키(e, n)를 이용하여 메시지(M') 계산
         - $M' = S^e \ mod \ n$
       - 수신자는 M과 M'를 비교하여 유효성 검증 가능



공개키 암호 알고리즘

- Diffie-Hellman 키 교환
  - 정의
    - 송수신자가 대칭키를 안전하게 교환하기 위해 사용하는 알고리즘
  - 특징
    - 키를 암호화하기 위함이 아닌 교환을 위한 알고리즘
    - 본인의 개인키와 상대방의 공개키를 이용하여 대칭키 생성
    - 이산 대수 문제(Discrete Logarithms Problems) 활용
    - 중간자 공격(Man-in-the-Middle Attack) 시도 가능

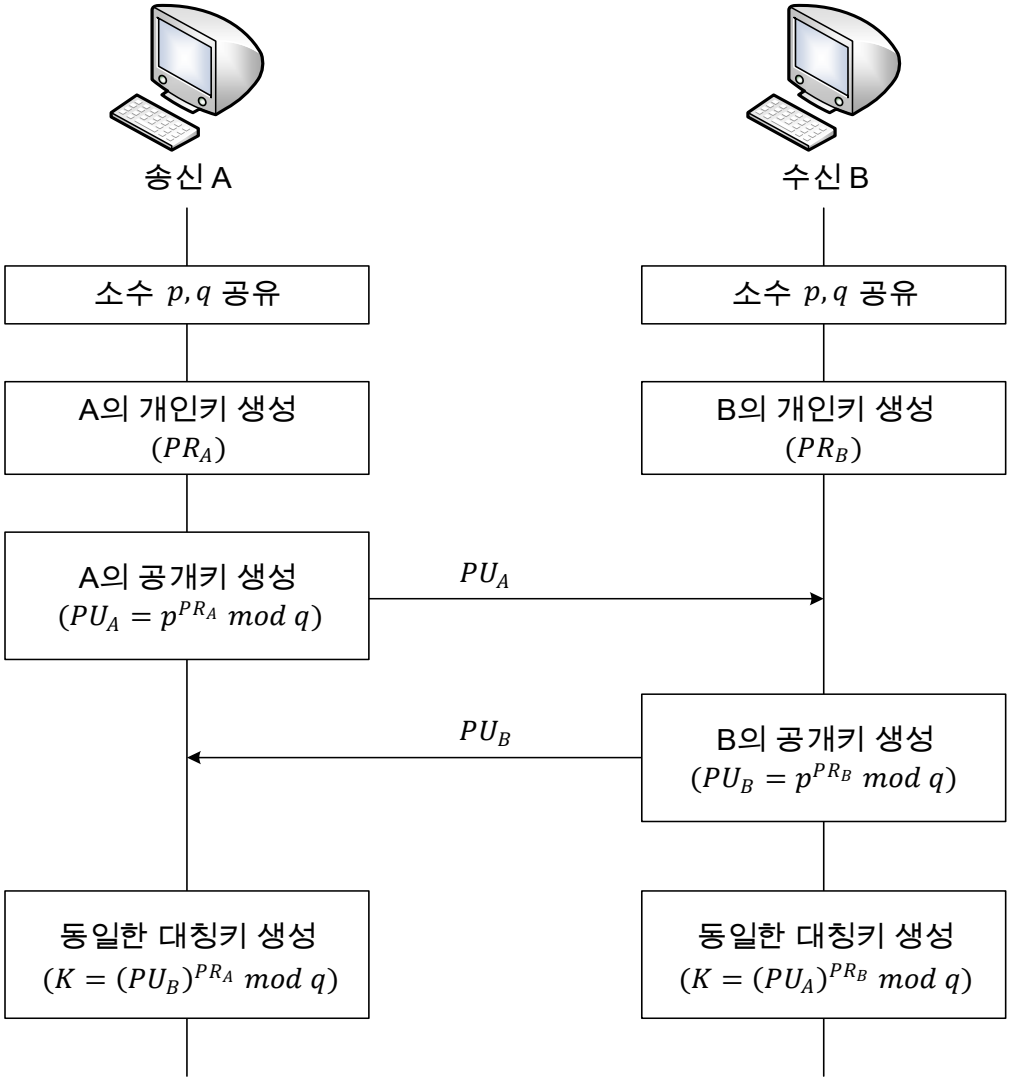
공개키 암호 알고리즘

- Diffie-Hellman 키 교환
  - 이산 대수 문제(Discrete Logarithms Problems)
    - 정의
      - $y = x^{n}(mod\ m)$에서 x, y를 알고 있는 경우, 지수 n을 구하기 어렵다는 문제
    - 원시근(Primitive Root)
      - 자신의 거듭제곱으로 1부터 p-1까지의 모든 정수를 생성할 수 있는 수를 의미
      - 어떤 수 a가 소수 p의 원시근
        - $a\ mod\ p, a^2\ mod\ p, \ldots, a^p - 1\ mod\ p$
        - e.g., $a^i\ mod\ p,\ \ p = 13$인 경우, 7은 p의 원시근 중 하나임

	1	2	3	4	5	6	7	8	9	10	11	12
7	7	10	5	9	11	12	2	9	8	10	6	1

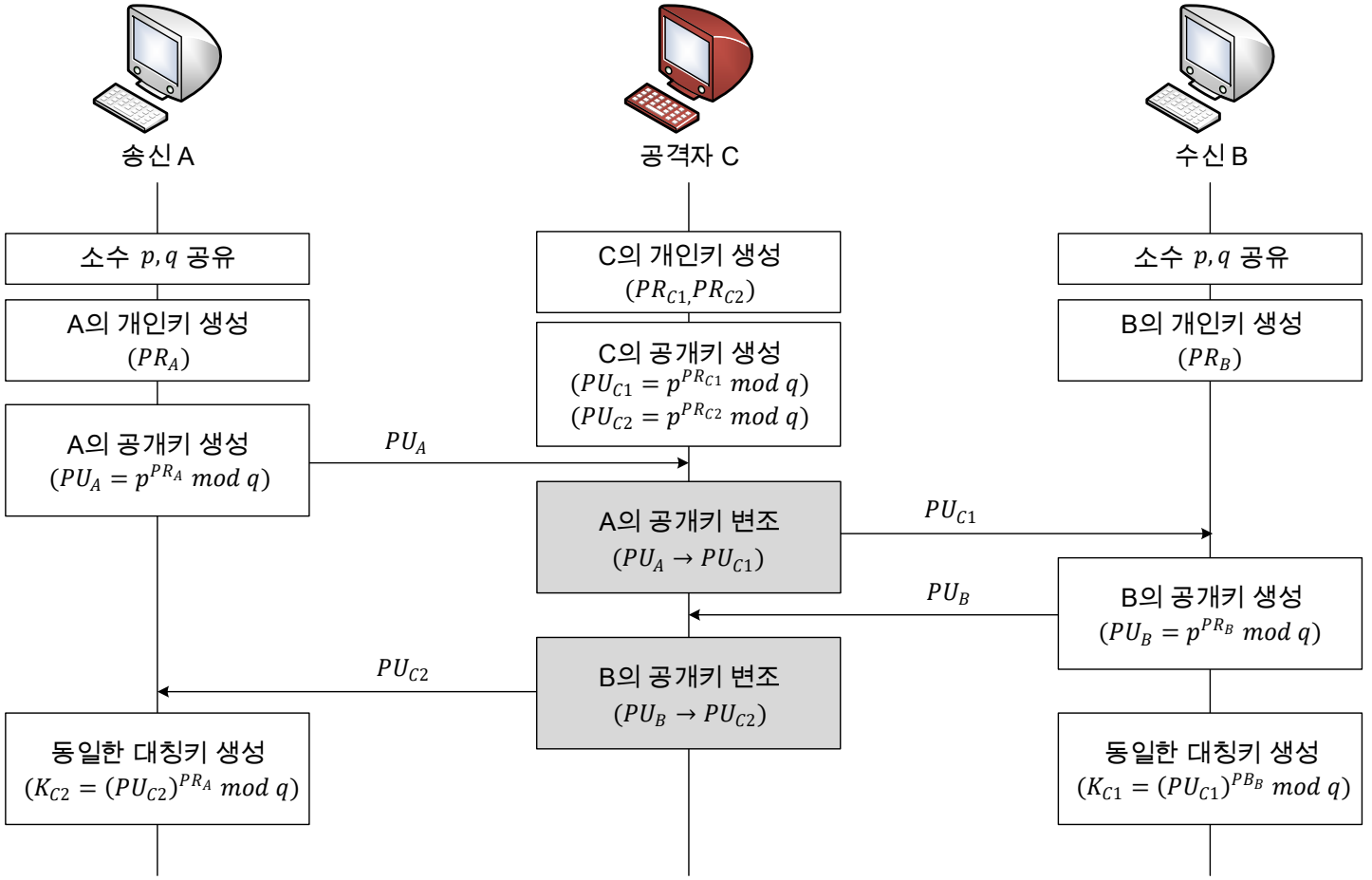
공개키 암호 알고리즘

- Diffie-Hellman 키 교환
  - 키 교환 알고리즘



공개키 암호 알고리즘

- Diffie-Hellman 키 교환
  - 중간자 공격(Man-in-the-Middle Attack)
    - 공격자가 송수신자 간 전달 메시지를 변조하는 공격



Thanks!

김 지 혜 (jihye@pel.sejong.ac.kr)