4.4 DH

■ 디피(Diffie)와 헬먼(Hellman)

- 1976년 "공개키 암호(public-key cryptography)" 개념을 처음 세상에 발표
- 키 교환 알고리즘(key exchange algorithm)만을 제시

■ **Diffie–Hellman 키 교환 알고리즘이란?**

- 문제: 두 사람이 인터넷처럼 누구나 엿들을 수 있는 채널을 통해 대화하면서, 비밀 키를 공유하려고 한다.
- 조건: 미리 비밀을 공유하지 않았음.
- 해결책: 수학적 함정(이산 로그 문제)을 이용해서, 공개적으로 값들을 주고받아도 최종적으로는 둘만 아는 공유 비밀 키를 만들어낸다.
  - 수학적 함정 = 이산 로그 문제
    - Forward Path: $g^a \, mod \, p$ 계산은 빠름(지수승 + 나머지 연산은 컴퓨터가 금방함)
    - Backward Path : $g^a \, mod \, p$ 가 주어졌을 때, "지수 a가 뭐냐?" -> 이산 로그 문제(discrete logarithm problem)

4.4 DH

■ 이산 로그문제 예제

- 소수 $p = 23$, 밑 $g = 5$, $a = 6$
- Forward path
  - $g^a mod\, p = 5^6\, mod\, 23$
  - $5^6 = 15625$
  - $15625\ \div 23 = 679$ 나머지 $= 2$

- Backward Path
  - 이제 누군가 "결과값 2가 나왔는데, a가 뭐냐?"라고 묻는다면?
    결국 아래 같은 문제를 푸는 것:
  - $5^a \equiv 2\ (mod\ 23)$
  - $5^1 = 5\, mod\, 23 = 5$
  - $5^2 = 25\, mod\, 23 = 5$
  - a = 2일 때도 2가 나오기 때문에, 같은 결과값이 여러 지수에서 반복되므로 해 찾기는 훨씬 복잡해짐

- $5^a \equiv 2(mod 23)$에서 a를 찾아라 -> a = 6 거의 불가능

4.4 DH

■ Diffie–Hellman 키 교환 알고리즘 절차

1. 큰 소수 p 원시근 g 는 공개
2. Alice는 비밀 a 를 고르고 $A = g^a \, mod \, p$ 를 공개
   Bob는 비밀 b를 고르고 $B = g^b mod \, p$를 공개
3. 이제 Alice $B^a mod \, p$를 계산, Bob은 $A^b mod \, p$를 계산
   - 두 값은 같아서 $g^{ab} \, mod \, p$ 가 된다
4. 이 값이 두사람만 아는 공유키
   - 핵심은 서로의 공개값을 자기 비밀 지수로 한번 더 거듭 제곱한다는것

■ 예제

- $p = 23, g = 5$
- 각자 비밀 선택
  - $Alice \; a = 6, Bob \; b = 15$
- 각자 보내는 공개값
  - $Alice$가 보냄 $A \; \equiv g^a mod \, p = 5^6 \, mod \, 23 = 8$
  - Bob가 보냄 $B \; \equiv g^b mod \, p = 5^{15} \, mod \, 23 = 19$
- Alice가 계산하는 공유키
  - Alice $K = B^a \, mod \, p = 19^6 mod \, 23 = 2$
- Bob이 계산하는 공유키
  - $Bob \; K' = A^b mod \, p = 8^{15} \, mod \, 23 = 2$
- 엿듣는 사람이 보는것 $p = 23, g = 5, A = 8, B = 19$

4.4 디피-헬먼

■ 디피-헬먼 키 교환 알고리즘(DH)

- DH는 암호화나 서명을 위한 것이 아니라 사용자가 공유된 대칭키를 설정하는 데 사용
- DH의 보안은 이산 로그 문제의 계산 난이도에 따라 달라짐
- 예) g와 $x=g^k$
  - $k=\log g(x)$이기 때문에 k를 알아내려면 로그를 계산해야 함
  - g, p, $g^k \pmod{p}$가 주어지면 k를 찾는 문제는 이산 로그 문제와 유사

- DH를 위한 수학적 표현은 비교적 단순
- 예) p를 소수라 하고 g는 생성자라 가정
  - 어떤 $x \in \{1, 2, \ldots, p-1\}$에 대해 $x=g^n \bmod p$를 만족하는 지수 n이 존재
  - 앨리스는 $g^a \bmod p$를 계산하고 그 결과를 밥에게 보냄

$$(g^b)^a \bmod p = g^{ab} \bmod p$$

  - 밥은 $g^b \bmod p$를 계산하고 그 결과를 앨리스에게 보냄

$$(g^a)^b \bmod p = g^{ab} \bmod p$$

4.4 디피-헬먼

- g^{ab} mod p는 공유된 비밀이며 대칭키로 사용

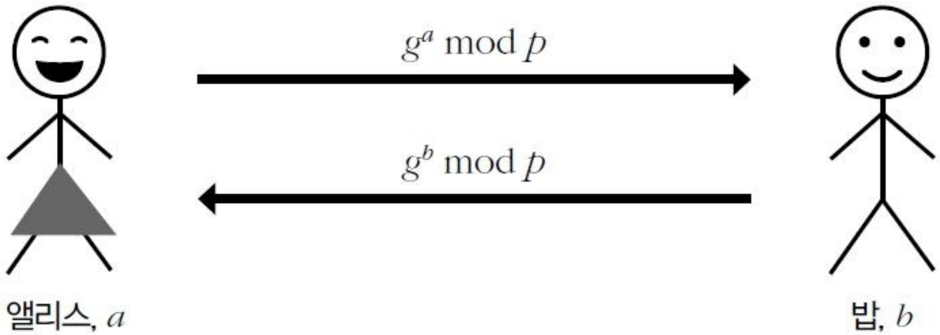


그림 4-1 디피–헬먼 키 교환

$g^a \cdot g^b = g^{a+b} \neq g^{ab}$ mod p (트루디는 비밀을 알 수 없음)

- DH 알고리즘과 관련된 근본적인 문제가 있는데 이는 중간자 공격 혹은 MiM 공격에 취약
- 트루디가 직접 앨리스와 밥의 중간에 서서 앨리스에서 밥으로 또는 그 역으로 오가는 메시지를 가로채는 능동적인 공격

4.4 디피-헬먼

- 트루디가 앨리스와 밥의 중간 위치에 있으면 앨리스와 밥 사이의 DH 공격은 쉽게 부서질 수 있음

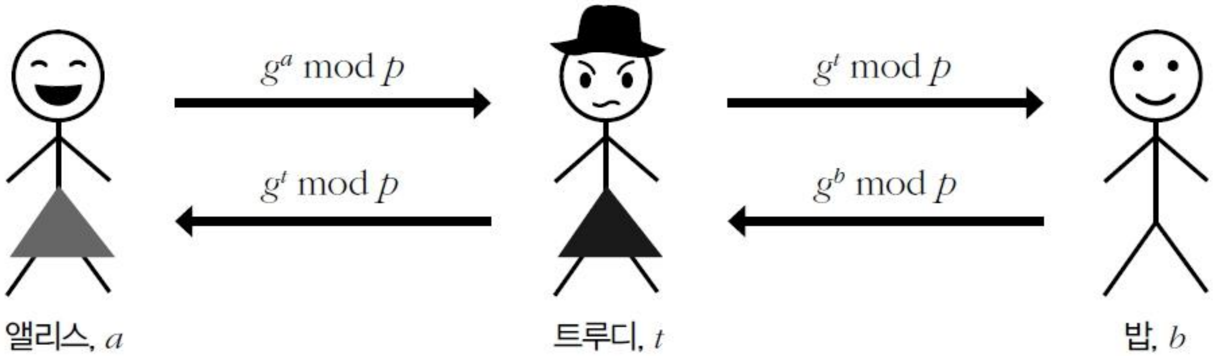


그림 4-2 디피-헬먼 중간자 공격

  - [그림 4-2]의 중간자 공격은 DH를 사용할 때 중요한 사항

- 이 공격을 방지하기 위한 여러 방법
  1. 공유된 대칭키로 DH 교환을 암호화
  2. 공개키로 DH 교환을 암호화
  3. 개인키로 DH 값을 서명

4.4 디피-헬먼

- 원래라면 공유키

$$g^{ab} mod\ p$$

- 하지만 트루디가 t 를 선택해 개입
  - 앨리스와는 g^{at}공유
  - 밥과는 g^{tb} 공유

- 따라서
  - 앨리스-트루디 공유키 $K_A = g^{at} mod\ p$
  - 밥-트루디 공유키 $K_B = g^{bt} mod\ p$
- 결과 : 앨리스와 밥은 서로 다른 키를 갖고 있음. 대신 트루디는 두 키 모두 알고 있음 -> 암호화가 깨짐

4.4 디피-헬먼

- $p = 23, g = 5$
- $Alice : a = 6 \ \rightarrow A = 8$
- $Bob : b = 15 \ \rightarrow B = 19$
- $Trudy: t = 13$

- $Trudy$가 개입
  - $Alice - Trudy : K_A = (g^t)^a = 5^{78} mod\ 23 = 12$
  - $Bob - Trudy: K_B = (g^t)^b = 5^{195} mod\ 23 \ - 7$
- 즉 트루디는 두 공유키를 알고, Alice-Bob의 대화를 복호화 가능

4.5 타원곡선 암호

■ 타원곡선(elliptic curve)

- 공개키 암호에 필요한 수학 연산을 수행할 수 있는 또 다른 방법을 제공
- 타원곡선 암호(ECC)의 장점
  - 같은 수준의 보안을 성취하기 위해 필요한 비트 수가 적다는 것
- 타원곡선 수학을 깊게 다루어서 타원곡선 수학 연산에 비용이 많이 듦
- 타원곡선 E는 다음과 같이 함수 그래프의 형태

$$E : y^2 = x^3 + ax + b$$

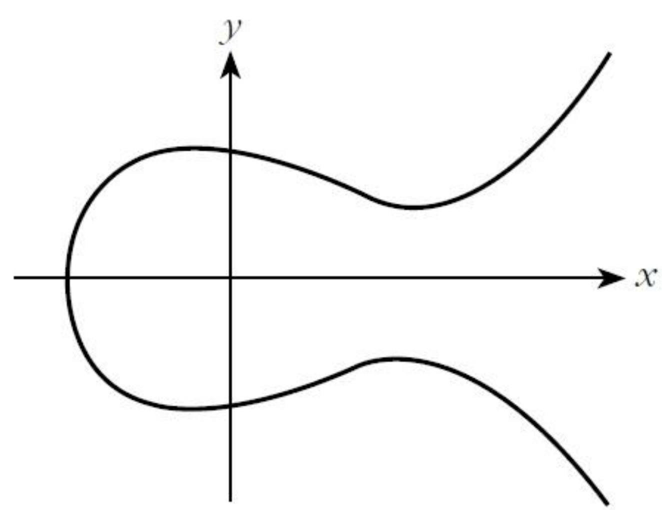


그림 4-3 $y^2 = x^3 - 2x + 2$ 타원곡선 그래프

4.5 타원곡선 암호

■ 타원곡선 수학

- 타원곡선상에서 필요한 유일한 수학적 연산은 덧셈
- $P_3 = P_1 + P_2$ P_3는 합의 대칭점

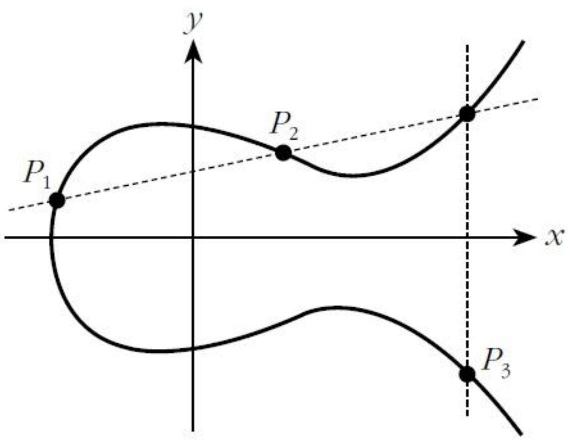


그림 4-4 타원곡선에서 점 추가

- 원래의 타원곡선 공식에 'mod *p*'를 추가

$$y^2 = x^3 + ax + b \pmod{p}$$

4.5 타원곡선 암호

■ 타원곡선 수학

- $E : y^2 = x^3 + ax + b\ (mod\ p)$위의 두 점 $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$에 대해 합 $P_3 = P_1 + P_2 = (x_3, y_3)$를 정의하려면
  - 서로 다른 점일 때($\mathrm{P}_1 \neq P_2$):
    - 기울기 m 을
    $$m \equiv (y_2 - y_1) \cdot (x_2 - x_1)^{-1}(mod\ p)$$
    로 정의한다. 여기서 $(x_2 - x_1)^{-1}$은 $(x_2 - x_1)$의 모듈러 역원 (즉 $(x_2 - x_1) \cdot (x_2 - x_1)^{-1} \equiv (mod\ p)$ 이다
    - 결과 좌표는
    $$x_3 \equiv m^2 - x_1 - x_2(mod\ p),$$
    $$y_3 \equiv m(x_1 - x_3) - y_1(mod\ p)$$
  - 같은 점을 두 번 더할 때 ($P_1 = P_2$, 즉 $2P$) – 접선의 기울기:
    - 기울기
    $$m \equiv (3x_1^2 + a) \cdot (2\ y_1)^{-1}(mod\ p)$$
    - 이후 위의 식과 동일한 식 사용
  - 특수 케이스
    - $x_2 \equiv x_1$이고 $y_2 \equiv -y_1$ 서로 역점이면 무한대 점

4.5 타

■ 타원

- $E : y$

  $P_3 =$

  - 서
    - 가

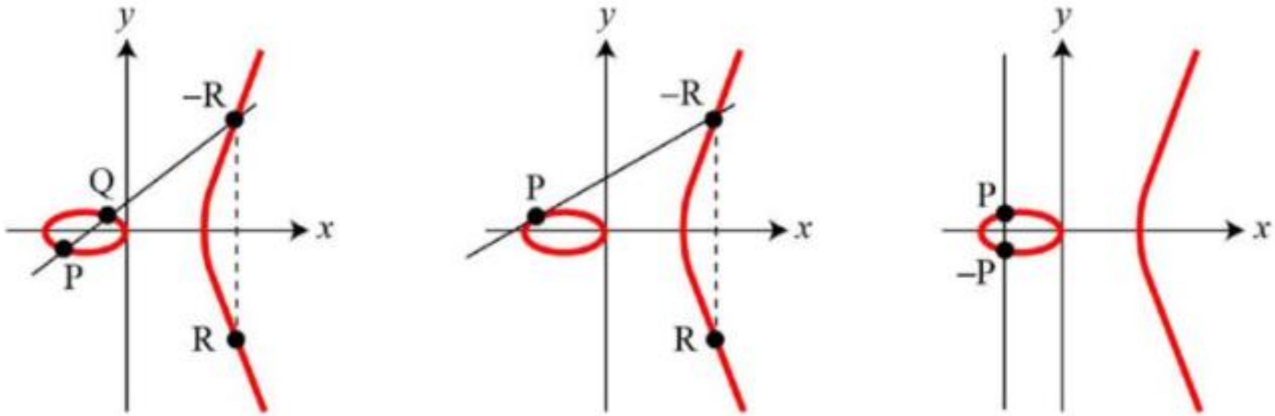
      0

    - 같

  - 같
    - 가
    - 0

해 합

$\equiv (mod\ p)$



a. (R = P + Q) b. (R = P + P) c. (O = P + (– P))

- Addition $(P \neq Q)$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$
$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = (x_1 - x_3)\lambda - y_1$$

- Doubling $(P = Q)$

$$\lambda = \frac{3x_1^{\,2} + a}{2y_1}$$
$$x_3 = \lambda^2 - 2x_1$$
$$y_3 = (x_1 - x_3)\lambda - y_1$$

4.5 타원곡선 암호

- 곡선상의 모든 점 (x,y)에 대해 가능한 값 x와 이와 일치하는 값 y를 계산해 나열하기

$$y^2 = x^3 + 2x + 1 \pmod 5 \qquad (4.5)$$

  - 모듈로 5를 작업하고 있기 때문에 오직 x=0, 1, 2, 3, 4라는 것만 고려할 필요가 있음

$$x = 0 \quad \Rightarrow y^2 = 1 \quad \Rightarrow y = 1,4 \pmod 5$$

$$x = 1 \quad \Rightarrow y^2 = 4 \quad \Rightarrow y = 2,3 \pmod 5$$

$$x = 2 \quad \Rightarrow y^2 = 13 = 3 \quad \Rightarrow \text{해가 없음} \pmod 5$$

$$x = 3 \quad \Rightarrow y^2 = 34 = 4 \quad \Rightarrow y = 2,3 \pmod 5$$

$$x = 4 \quad \Rightarrow y^2 = 73 = 3 \quad \Rightarrow \text{해가 없음} \pmod 5$$

  - 식 (4.5)에서 타원곡선상의 점을 다음과 같이 찾을 수 있음

$$(0,1),\ (0,4),\ (1,2),\ (1,3),\ (3,2),\ (3,3)\ \text{그리고}\ \infty \qquad (4.6)$$

4.5 타원곡선 암호

■ 곡선상에서 두 점을 합하는 문제

- 타원곡선상에서 두 점을 대수적으로 더하기 위한 알고리즘

표 4-1 타원곡선 mod p상에서의 덧셈

주어진 조건: 곡선 E: $y^2=x^3+ax+b \pmod{p}$

E상에서 $P_1=(x_1, y_1)$, $P_2=(x_2, y_2)$

구하는 해: $P_3=(x_3, y_3)=P_1+P_2$

알고리즘:

$x_3=m^2-x_1-x_2 \pmod{p}$

$y_3=m(x_1-x_3)-y_1 \pmod{p}$

$$\text{여기서 } m = \begin{cases} \text{만약 } P_1 \neq P_2 \text{ 이면, } (y_2 - y_1) \cdot (x_2-x_1)^{-1} \pmod{p} \\ \text{만약 } P_1=P_2 \text{ 이면, } (3x_1^2+a) \cdot (2y_1)^{-1} \pmod{p} \end{cases}$$

특수한 경우 1: 만약 $m=\infty$이면, $P_3=\infty$

특수한 경우 2: 모든 P에 대해 $\infty+P=P$

$m=(3-1)\cdot(1-0)^{-1}=2 \pmod{5}$

$x_3=2^2 -1- 0 = 3 \pmod{5}$

$y_3=2(0-3)-1= -6 - 1= - 2 = 3 \pmod{5}$

- 따라서 곡선 $y^2=x^3+2x+1 \pmod{5}$상에서 총합이 (0,1)+(1,3)=(3,3)이라는 결론에 이름

4.5 타원곡선 암호

■ ECC 디피-헬먼

- 타원곡선상에서 덧셈을 할 수 있으니 디피-헬먼 키 교환의 ECC 버전을 고려

$$y^2 = x^3 + 11x + b \pmod{167} \qquad (4.7)$$

- 점 (x,y)를 선택하고 이 점이 결과로 나오는 곡선상에 존재하도록 b를 결정
- $(x,y)=(2,7)$이라고 할 경우,

  공개 정보: $y^2 = x^3 + 11x + 19 \pmod{167}$ 그리고 점 $(2,7)$ (4.8)

- 앨리스와 밥은 각각 자신의 비밀 승수를 무작위로 선택해야 함

  그 경우 앨리스는 $A(2,7)=15(2,7)=(102,88)$

  밥은 $B(2,7)=22(2,7)=(9,43)$
- 앨리스는 밥에게 받은 값을 다시 비밀 승수 A로 제곱을 함

  $A(9,43)=15(9,43)=(131,140)$

  밥은 $B(102,88)=22(102,88)=(131,140)$
- 그러면 앨리스와 밥은 대칭키로 사용하기에 적합한 공유된 비밀을 생성하게 됨
- 디피-헬먼의 이 타원곡선 버전이 작동하는 것은 $(AB)P=(BA)P$이기 때문

4.5 타원곡선 암호 – 실제 타원곡선 예시

- 예) 예시는 써티콤 ECCp-109 도전 문제의 일부분
  - 숫자는 쉼표 구분 없는 10진법으로 이루어짐

$$p = 564538252084441556247016902735257$$

$$a = 321094768129147601892514872825668$$

$$b = 430782315140218274262276694323197$$

  - 타원곡선 $E : y^2 = x^3 + ax + b \pmod{p}$
    - P는 타원곡선 E에 있는 다음과 같은 점

$$(97339010987059066523156133908935, 149670372846169285760682371978898)$$

    - k=281183840311601949668207954530684라고 하면 kP는

$$(44646769697405861057630861884284, 522968098895785888047540374779097)$$

4.6 공개키 표기법

- 앨리스의 공개키로 메시지 M을 암호화: $C=\{M\}_{\text{앨리스}}$
- 앨리스의 개인키로 암호문 C를 복호화: $M=[C]_{\text{앨리스}}$
- 앨리스가 메시지 M에 한 서명: $S=[M]_{\text{앨리스}}$
  - 중괄호는 공개키 연산, 대괄호는 개인키 연산, 아래첨자 이름은 누구의 키를 사용하는 것인지 보여줌

- 공개키와 개인키 연산은 역으로 함
  - $[\{M\}_{\text{앨리스}}]_{\text{앨리스}}=\{[M]_{\text{앨리스}}\}_{\text{앨리스}}=M$
  - 공개키는 항상 공개 → 누구나 $\{M\}_{\text{앨리스}}$를 계산 가능
  - 개인키는 비밀 → 오직 앨리스만이 $[C]_{\text{앨리스}}$ 또는 $[M]_{\text{앨리스}}$를 계산

4.7 공개키 암호 용도

■ 공개키 암호의 중요한 장점

- 공개키 암호는 공유키를 미리 설정할 필요가 없음
- 무결성뿐만 아니라 부인봉쇄(non-repudiation)를 제공

■ 현실 세계에서의 보안성

- 효율성을 가지면서도 공유키를 미리 가질 필요가 없는 것이 가능할까?
  - ↳ 방법은 합성 암호체계를 이용하는 것

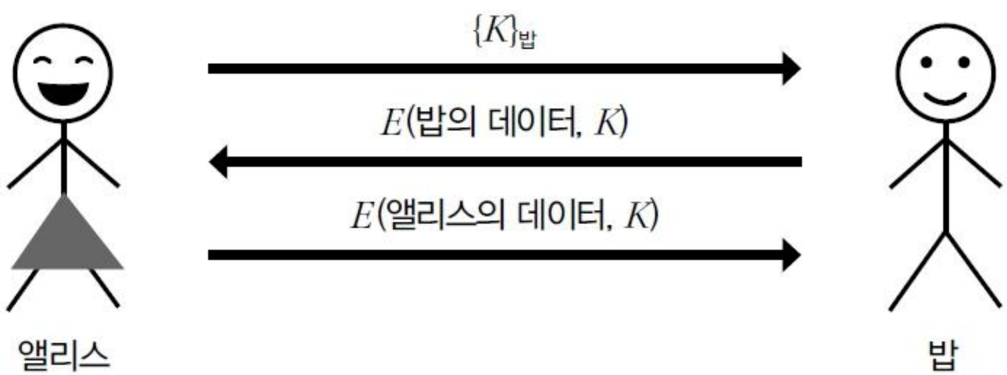


그림 4-5 합성 암호체계

4.7 공개키 암호 용도 – 서명과 부인봉쇄

■ 대칭키 암호의 무결성 예시를 살펴보기

- 앨리스가 단골 증권 중개인인 밥으로부터 주식 100주를 주문
  - 주문의 무결성을 확보하기 위해 앨리스는 공유된 대칭키 K_{AB}를 이용해 MAC를 계산
  - 앨리스가 주문을 넣은 지 얼마 지나지 않아 밥에게 돈을 내기 전에 주식의 가치가 급락
  - 이 시점에서 앨리스는 주문 전송을 부인할 수 있음

  - 밥은 앨리스가 주문을 넣었다는 것을 증명할 수 있을까?
  - 밥이 가진 것이 MAC밖에 없다면 증명할 방법이 없음

  - 이번에는 앨리스가 MAC 대신에 디지털 서명을 사용했다고 가정
  - MAC처럼 서명은 무결성 확인이 가능
  - 주가가 폭락하자 앨리스는 주문한 내용을 부인하려 한다고 가정
  - 밥은 그 주문이 앨리스로부터 온 것이라는 것을 증명할 수 있을까?
  - 증명할 수 있음!

    ↳오직 앨리스만이 개인키에 접근할 수 있기 때문

4.7 공개키 암호 용도 – 기밀성과 부인봉쇄

- 앨리스와 밥 모두 이용 가능한 공개키를 가지고 있고 앨리스가 메시지 M을 밥에게 보내려고 가정할 때
  - 보안에 매우 민감한 앨리스가 보안과 부인봉쇄 모두를 원한다고 가정
  - 그러면 앨리스는 단순히 M을 서명할 수는 없음
  - 해결책: 앨리스는 메시지를 밥에게 보내기 전에 서명하고 암호화하면 됨

$$\{[M]_{앨리스}\}_{밥}$$

  - 아니면 앨리스가 M을 먼저 암호화하고 서명하는 것이 더 나을까?

$$[\{M\}_{밥}]_{앨리스}$$

  - 순서가 영향을 미칠까? 두 가지 시나리오를 통해 살펴보기

4.7 공개키 암호 용도 – 기밀성과 부인봉쇄

■ 앨리스와 밥이 연인 관계라고 가정

- 앨리스는 M="사랑해" 메시지를 밥에게 보내려고 함

$$\{[M]_{앨리스}\}_{밥}$$

- 앨리스와 밥 사이에 작은 다툼이 일자, 밥은 홧김에 $[M]_{앨리스}$를 얻기 위해 서명된 메시지를 복호화한 뒤 찰리의 공개키를 이용해 다시 암호화함

$$\{[M]_{앨리스}\}_{찰리}$$

- 밥은 이 메시지를 찰리에게 보냄

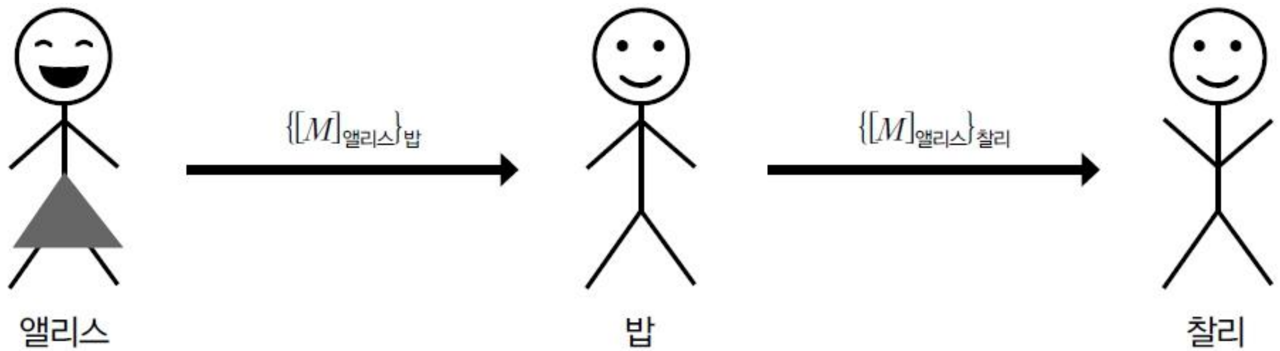


그림 4-6 서명 후 암호화의 함정

- 앨리스는 다시는 서명 후에 암호화를 하지 않겠다고 다짐
- 이제부터 앨리스는 기밀성과 부인봉쇄를 원할 때 항상 암호화를 한 후 서명할 것임

4.7 공개키 암호 용도 – 기밀성과 부인봉쇄

■ 앨리스와 밥이 연인 관계라고 가정

- 앨리스와 밥은 화해했고 앨리스는 새로 개발한 이론을 메시지로 밥에게 보내려 함

  M= "브론토사우루스는 한쪽 끝은 가늘고 가운데는 두껍다가 다른 쪽 끝에서 다시 가늘어진다."

- 앨리스는 밥에게 메시지를 보내기 전에 다음과 같이 암호화한 뒤 서명

  $[\{M\}_{\text{밥}}]_{\text{앨리스}}$

- 찰리는 자신을 중간자로 설정하고 앨리스와 밥 사이에 오가는 모든 트래픽을 가로챔
- 찰리는 가로챈 $[\{M\}_{\text{밥}}]_{\text{앨리스}}$에서 앨리스의 공개키를 이용해 $\{M\}_{\text{밥}}$을 계산한 뒤 이를 밥에게 보내기 전에 서명함

  $[\{M\}_{\text{밥}}]_{\text{찰리}}$

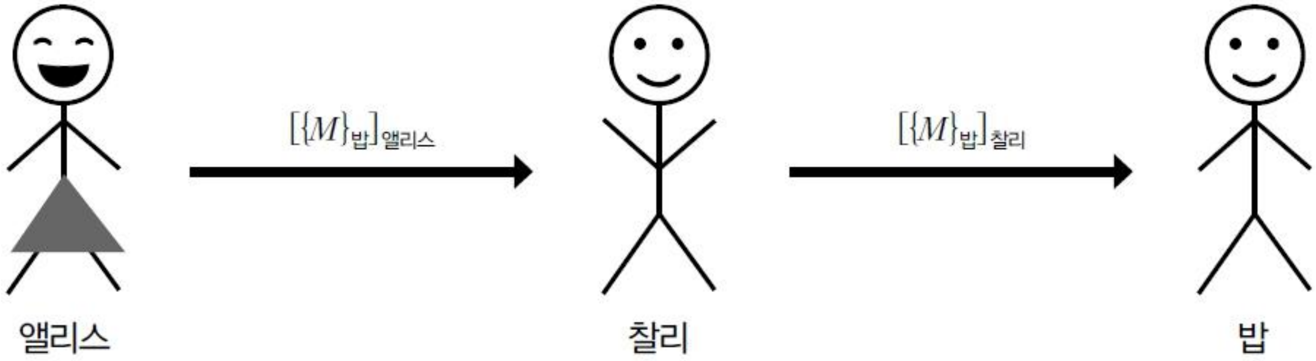


그림 4-7 암호화 후 서명의 함정

4.8 인증서와 PKI

- 공개키 기반 구조(PKI)
  - 공개키를 안전하게 사용하기 위해 필요한 모든 것을 모아 놓은 것

- 전자 인증서(digital certificate)
  - 사용자의 공개키와 사용자 이름을 함께 포함. 이는 인증 기관(CA)에 의해 서명
  - 예) 앨리스의 인증서는 다음 내용을 포함

M=("앨리스", 앨리스의 공개키) 그리고 $S=[M]_{CA}$

- 이 인증서를 검증하기 위해 밥은 $\{S\}_{CA}$를 계산하고 이것이 메시지 M과 일치하는지 검증
- 밥은 CA를 믿어야만 하고 인증서가 유효하다는 것을 판단하기 전에 서명을 검증해야 함

4.8 인증서와 PKI

- 인증서에는 공개키 외에도 유용한 여러 정보들을 포함할 수 있음
  - 하지만, 더 많은 정보가 있을수록 인증서는 유효성을 잃게 될 것임
- 만약 CA가 실수를 저지른다면 대단히 심각한 결과를 초래할 수 있음
  - 예) 베리사인(VeriSign)이 마이크로소프트를 위해 서명된 인증서를 다른 누군가에게 발행한 적이 있음
  - ↳ 중요한 PKI 이슈(인증서 폐기 이슈)를 불러옴
- PKI는 다음 문제를 처리해야만 함
  - 키 생성과 관리
  - 인증 기관(CAs)
  - 인증서 폐기

4.8 인증서와 PKI

- 최근 사용되고 있는 높은 수준의 PKI 신뢰 모델
  - 완전 독점 모델(monopoly model)
    - 개념: 단 하나의 루트 인증기관(CA)만 존재하고, 모든 사용자는 이 기관을 신뢰해야 함.
    - 장점: 단순함, 관리가 명확하고 일관성 있음.
    - 단점: 단일 실패 지점(Single Point of Failure). 만약 이 루트 CA가 해킹되거나 잘못된 인증서를 발급하면, 전체 인터넷 보안이 무너질 수 있음.
  - 소수 독점 모델(oligarchy model)
    - 개념: 여러 개의 루트 CA가 존재하고, 운영체제나 브라우저 벤더가 이들을 신뢰 루트로 미리 탑재.
    - 장점: 독점보다는 분산되어 신뢰성이 올라감. 특정 CA가 문제가 생겨도 다른 CA를 통해 운영 가능.
    - 단점: 어떤 CA를 신뢰할지 결정하는 권한이 운영체제·브라우저 업체에 집중됨. 사용자는 실제로 '소수의 선택'을 따를 수밖에 없음.
  - 완전 자유 모델(anarchy model)
    - 개념: 중앙 권위가 없고, 각 사용자가 스스로 어떤 공개키를 신뢰할지 결정.
    - 장점: 절대적 권위가 없으므로 권력 집중 문제 없음. 완전한 분산 구조.
    - 단점: 확장성 부족, 관리가 어려움. 잘못된 키를 쉽게 신뢰할 위험.

  - 그러나 모두가 동의하는 신뢰 모델이 없다는 사실 자체가 PKI가 가지는 중요한 문제 중 하나

4.9 양자 컴퓨터와 공개키

- 양자 알고리즘(quantum algorithm)
  - 1994년에 피터 쇼어(Peter Shor)는 쇼어 알고리즘을 개발
    - 인수분해 문제에 대해 효율적인 해결책 제공
  - 인수분해는 계수를 인수분해하여 RSA를 해독할 수 있음
    - RSA 계수는 정수 $N=pq$이며, p와 q는 소수
  - 양자 컴퓨터를 사용할 수 있다고 가정하면, 쇼어 알고리즘은 다음과 같은 순서로 암호를 깨기 위해 노력

$$(\log_2 N)^2(\log_2(\log_2 N))(\log_2(\log_2(\log_2 N)))$$

$$n^2\log_2(n)\log_2(\log_2(n))$$

    - 여기서 $n=\log_2 N$은 N이라는 비트 수, n-비트의 RSA 계수를 적용했을 때 쇼어 알고리즘의 암호를 깨는 데 드는 노력은 다음과 같은 길이를 가진 대칭키를 찾기 위한 검색과 거의 같음

$$2\log_2 n+\log_2\log_2 n+\log_2\log_2\log_2 n$$

4.9 양자 컴퓨터와 공개키

- 가장 좋은 고전 인수분해 알고리즘은 하위 지수 작업 계수(work factor)를 가지는 수체 체(number field sieve) 알고리즘
- 수체 체 작업 계수는 다음 크기의 대칭키를 찾기 위한 검색과 같음

$$1.9223 n^{1/3} (\log 2n)^{2/3}$$

- 예) n=2048비트인 RSA 계수를 인수분해하려면?
  - 수체 체 알고리즘은 125비트 이상의 대칭키를 찾기 위한 검색과 비슷한 노력을 해야 함
  - 반면에 같은 크기의 RSA 계수에 대해 쇼어 알고리즘은 30비트 대칭키를 찾기 위한 검색보다 더 적은 노력을 들여도 됨

- 충분한 크기의 양자 컴퓨터가 현실이 되면 현재 사용하는 RSA는 사라질 것임
- 다행히 포스트 양자 시대에도 사용 가능한 암호 시스템이 몇 가지 있음
  - 예) NTRU 공개키 체계는 격자에서 가장 짧은 벡터를 찾는 수학 문제를 기반으로 함

4.10 요약

- 배낭암호를 시작으로 RSA와 디피-헬먼 암호를 자세하게 다루었음
- 타원곡선 암호화(ECC)는 미래에 그 역할이 점점 더 증가할 것으로 보임
- 공개키 암호의 가장 큰 장점이라 할 수 있는 서명과 부인봉쇄
- 공개키 암호가 실제로 기밀성을 위해 사용될 수 있는 방법인 합성 암호체계라는 아이디어 제시
- 중요하지만 종종 혼돈을 겪는 전자 인증
- 공개키 암호를 전개해서 사용할 때 주된 장애물이 되는 PKI
- 쇼어 알고리즘
- 성공적인 양자 컴퓨터가 공개키 영역에 미칠 수 있는 영향