정보보호개론(CSE441)
www.facebook.com/KoreaTech.Crypto

암호알고리즘 개요 2부

NOTE 03

한국기술교육대학교 컴퓨터공학부 김상진

sangjin@koreatech.ac.kr
www.facebook.com/sangjin.kim.koreatech

교육목표

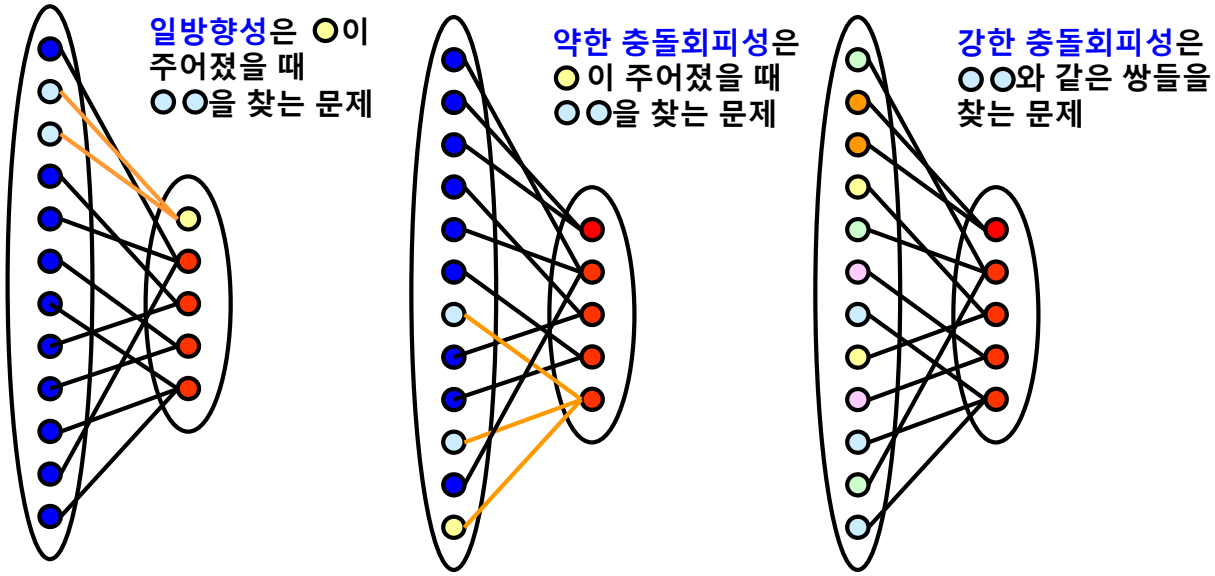
- 암호알고리즘에 대한 기초적 이해
  - 해시함수
  - MAC
  - 전자서명 알고리즘
- 암호해독과 암호알고리즘의 안전성 이해

해시함수

- 해시함수(hash function): 임의의 길이에 이진 문자열을 고정된 길이의 이진 문자열(해시값, 메시지 다이제스트, 메시지 지문)로 매핑하여 주는 결정적 함수
- 요구사항
  - 압축
  - 계산의 용이성: x가 주어지면 $H(x)$는 계산하기 쉬워야 함
  - 일방향성(one-wayness, preimage resistance): 입력을 모르는 해시값 y가 주어졌을 때, $H(x') = y$를 만족하는 x'를 찾는 것은 계산적으로 어려워야 함. (OWHF, One-Way Hash Function)
  - 약한 충돌회피성(weak collision-resistance, second-preimage resistance): x가 주어졌을 때 $H(x') = H(x)$인 $x'(\neq x)$을 찾는 것은 계산적으로 어려워야 함
  - 강한 충돌회피성(strong collision-resistance): $H(x') = H(x)$인 서로 다른 임의의 두 입력 x와 x'을 찾는 것은 계산적으로 어려워야 함. (CRHF, Collision-Resistant Hash Function)
- 대표적 알고리즘: SHA-1, SHA-2, SHA-3

충돌

- 서로 다른 x와 x'에 대해 $H(x) = H(x')$이면 충돌이 발생하였다고 함
- 해시함수의 정의역은 치역보다 훨씬 범위가 크기 때문에 충돌의 발생은 불가피함
  - 이 충돌을 찾는 것은 어려워야 함

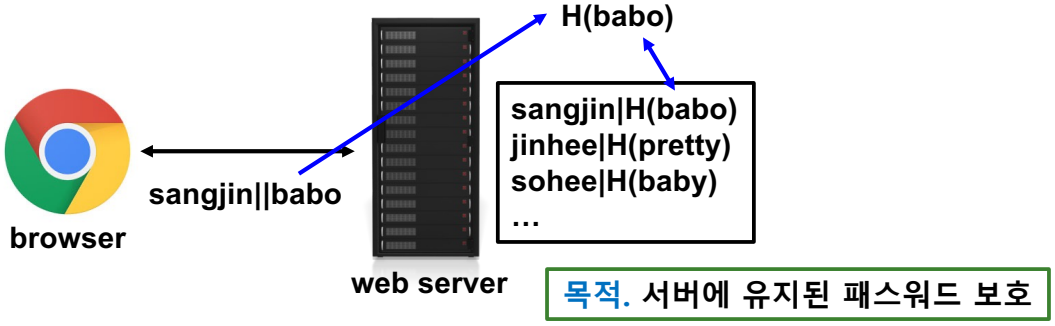


Birthday Paradox

- (생일 파라독스) N명 중 어떤 특정 사람을 선택하였을 때 그 사람과 생일이 같은 사람을 찾을 확률이 50%가 되기 위해서는 N이 183명 이상이 되어야 하지만 생일이 같은 임의의 두 사람을 찾을 확률이 50%가 되기 위해서는 23명만 있으면 됨
- 이 문제와 해시함수와의 관계
  - 해시함수 H: 사람의 생일 ⟹ {1, …, 365}
    - 생일이 같은 사람을 찾는 것은 이 해시함수에서 충돌을 찾는 것과 같음
  - birthday pardox 적용 결과: 23명만 검사해보면 충돌을 찾을 수 있음. 대략 $\sqrt{365}$임
- q개의 해시값(길이가 $n$bit)을 계산할 때 충돌을 찾을 확률: $\frac{q^2}{2^{n+1}}$
- SHA-1 해시함수의 해시값 길이: 160비트
  - 가능한 해시값: 2^{160}
    - x와 같은 값을 매핑되는 것을 찾기(50%): $2^{160}/2 = 2^{159}$
    - 같은 값으로 매핑되는 임의 쌍 찾기(50%): $\sqrt{2^{160}} = 2^{80}$

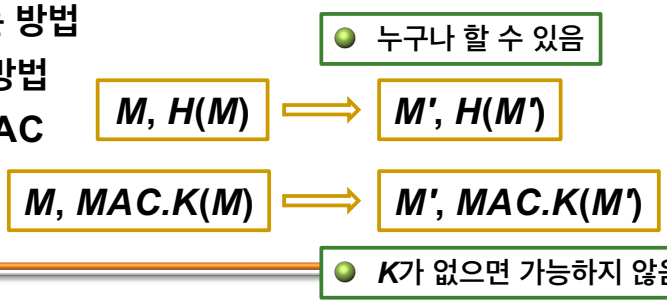
해시함수의 용도

- 해시함수의 용도
  - 전자서명: 서명 크기와 계산 비용을 줄이기 위해 전체 메시지 대신에 메시지의 해시값에 서명함
    - M, Sig.–K_A(M) ⟹ M, Sig.–K_A(H(M))
    - 해시함수의 충돌 회피성이 매우 중요
  - 무결성: M, H(M)
  - 패스워드 해싱 (Note 11)



MAC

- 해시함수는 보통 비밀키를 사용하지 않음
  - 단순히 메시지의 변경 여부를 판단하기 위해 사용하는 해시값을 조작 탐지 코드(MDC, Modification/Manipulation Detection Code)라 함
- 비밀키를 추가로 사용하면 송신자의 인증과 무결성을 동시에 제공할 수 있으며, 결과 값을 메시지 인증 코드(MAC, Message Authentication Code)라 함
  - MAC을 사용하기 위해서는 생성하는 자와 확인하는 자는 동일한 대칭키를 공유하고 있어야 함
- MAC도 해시함수와 같은 요구사항(일방향, 충돌회피 등)을 충족해야 하며, 추가로 위조가 가능하지 않아야 함
- MAC을 만드는 방법
  - MAC 전용 알고리즘을 이용하는 방법
  - 대칭 암호알고리즘을 이용하는 방법
  - 해시함수를 이용하는 방법: HMAC



무결성

- 예) 메시지 M을 전송할 때 M, $H(M)$을 전송
  - 공격자가 M을 M'으로 변경하면 수신자는 수신된 해시값과 새로 계산된 해시값을 비교하여 메시지가 변경되었다는 것을 알 수 있음
  - 공격자가 메시지만 변경하지 않고, 해시값까지 변경하면 (즉, M', $H(M')$) 수신자는 메시지가 변경되었다는 것을 알 수 없음
- 해결책 1. (MAC) $M, \mathrm{MAC}.K(M)$을 전송
  - 전제. 송신자와 수신자가 K를 공유하고 있어야 함
- 해결책 2. (전자서명) $M, \mathrm{Sig}.-K_A(M)$
- 해결책 1과 해결책 2의 비교
  - 비용(효율성) 측면에서는 해결책 1이 우수
  - 부인방지는 전자서명이 우수
  - 해결책 2는 대응되는 공개키를 가지고 있는 사용자는 누구나 무결성을 검증할 수 있음. 이들은 MAC과 달리 유효한 전자서명을 생성할 수는 없음

무결성과 비밀성을 동시에

- 인증 암호화(Authenticated Encryption): 비밀성과 무결성을 동시에 제공하는 방법
- 방법 1. 암호화할 때 해시값 또는 MAC 값을 포함하는 방법
  - $E.K(M||H(M))$, $E.K_1(M||MAC.K_2(M))$
- 방법 2. 메시지에 독립적으로 대칭 암호알고리즘과 MAC을 적용하는 방법

$$E.K_1(M), MAC.K_2(M)$$

  - 문제점. $MAC.K(M)$이 M에 대한 정보 노출. MAC은 비밀성을 제공하는 암호기술이 아님
- 방법 3. 메시지를 암호화한 후 암호문에 대한 MAC 값을 계산하는 방법

$$C = E.K_1(M), MAC.K_2(C)$$

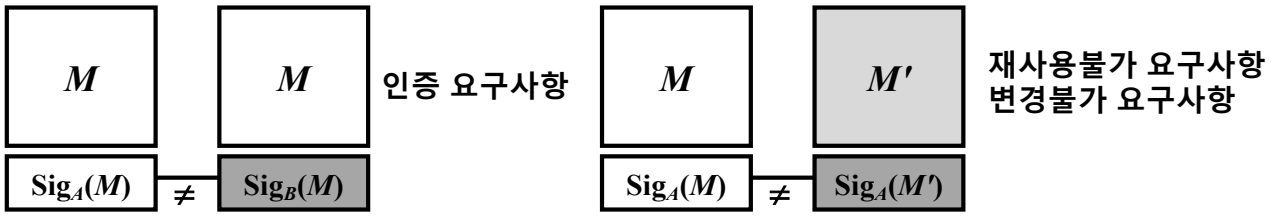
  - 이 방법을 encrypt-then-mac 방법이라 함
  - 방법 1과 2는 무조건 복호화를 해야 하지만 encrypt-then-mac은 MAC 값이 확인되지 않으면 복호화를 하지 않음
  - encrypt-then-mac이 인증 암호화 방식 중 가장 안전한 방법임

일반 서명과 전자서명의 차이점

- 전자서명은 수학적으로 서명자를 검증할 수 있음
  - 일반 서명은 보통 원 서명과 눈으로 대조하여 검증함
  - 위조될 확률이 높으므로 이 측면에서는 전자서명이 더 안전함
- 전자서명의 안전성은 동일한 알고리즘을 사용할 경우에는 사용자마다 같은 안전성을 제공함
  - 반면에 일반 서명은 사용자마다 다를 수 있음
- 일반 서명은 문서 위에 하지만 전자서명은 보통 문서와 별도로 존재함
  - 전자서명은 여전히 문서에 의존해야 함
  - 일반 서명은 서명마다 동일한 형태를 지니지만 전자서명은 서명하는 문서마다 다른 형태를 지님
- 전자서명은 원본과 복사본을 구분하기가 힘듦

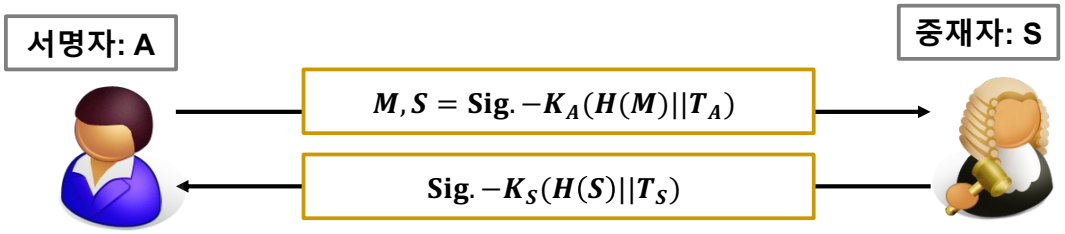
전자서명의 요구사항

- 인증(authentic): 누가 서명하였는지 확인이 가능해야 함
  ⟹ 각 서명자마다 서명이 독특해야 함 (서명키)
- 위조불가(unforgeable): 위조가 불가능해야 함
- 부인방지(non-repudiation): 나중에 부인할 수 없어야 함
- 재사용불가(not reusable): 서명을 다른 용도로 사용할 수 없어야 함
  (cut-and-paste) ⟹ 서명은 메시지에 의존해야 함
  - 전체의 재사용을 방지하기 위해서는 서명 시간이 포함되어야 함
- 변경불가(unalterable): 서명된 문서의 내용을 변경할 수 없어야 함



전자서명 방식의 종류

- 직접 서명 방식(direct digital signature): 서명자가 홀로 서명 알고리즘을 수행하여 서명하는 방식
  - 문제점. 서명키의 분실/도난
    - 서명자가 직접 서명 시간을 서명에 포함할 경우에는 서명키를 획득한 공격자의 부정을 증명할 방법이 없음
- 중재 서명 방식(arbitrated digital signature): 중재자와 프로토콜을 수행하여 서명하는 방식
  - 중재자는 서명의 증인 역할을 하게 됨
  - 서명자가 시간에 대한 부정을 할 수 없음
  - 중재자는 신뢰기관이어야 함



전자서명 알고리즘의 분류

- 메시지 복구 여부에 따른 분류
  - 메시지 복구 가능 전자서명 알고리즘(DSS with recovery)
    - 작은 메시지에 대해서만 사용 가능
  - 첨부 형태 전자서명 알고리즘(DSS with appendix)
    - 메시지와 서명 블록이 별도로 존재하며, 메시지와 서명 블록이 함께 제시되어야 검증 가능
    - 보통 해시함수를 이용함 (메시지 대신 해시값에 서명)
- 결정 vs. 확률 서명 알고리즘
  - 결정(deterministic) 서명 알고리즘: 메시지가 같으면 항상 결과 서명이 같은 알고리즘
  - 확률(probabilistic, randomized) 서명 알고리즘: 메시지가 같아도 서명할 때마다 그 결과가 달라지는 알고리즘
    - 암호화 함수와 달리 안전성 측면에서 서명 함수는 꼭 확률 알고리즘이 되어야 하는 것은 아님

암호알고리즘에 대한 안전성

- 암호알고리즘: 복호화키를 모르는 상태에서 암호문으로부터 평문이나 키를 얻을 수 없어야 함
  - 평문이나 키의 일부도 얻을 수 없어야 함
  - 공개키 암호알고리즘은 공개키로부터 개인키를 얻을 수 없어야 함
- 해시함수: 해시값의 역이나 충돌을 찾을 수 없어야 함
  - 역을 찾는 것은 보통 고려하지 않음
- MAC: MAC 키를 모르는 상태에서 MAC 값을 위조하거나 MAC 키를 알아낼 수 없어야 함
- 전자서명: 서명키를 모르는 상태에서 서명을 위조하거나 서명키를 알아낼 수 없어야 함
- 암호해독 외에 사회공학과 같은 다른 방법으로 키가 알려진 것을 노출(compromise) 되었다고 함

현대 암호학의 특징

- 과거는 “design-break-patch” 주기로 암호 기술이 개발됨
  - 오랫동안 허점이 발견되지 않은 것은 안전한 것으로 여김
- 하지만 오늘날에는 엄격한 안전성 증명을 중요하게 생각함
  - 관련 학문적 이론, 증명 기술의 발달이 중요 요인
  - Definition, Adversary Model, Assumption, Proof
  - 증명이 iron-clad guarantee 제공?
    - 아님
      - 정의가 잘못될 수 있음
      - 가정이 잘못될 수 있음
    - 그럼에도 불구하고 안전성 증명이 매우 중요

- 현재 현장에서는 엄격하게 증명된 것만 사용하고 있으며, 현장보다 더 안전한 알고리즘도 존재하지만 이들은 실용성 때문에 현장에서 사용하지 못하고 있음
- 향후 양자 컴퓨팅이 현실화되면 지금까지 사용한 여러 가정이 무의미해질 수 있음

암호알고리즘의 안전성 (1/2)

- 암호알고리즘을 해독하기 위해 요구되는 노력에 의해 측정
- 무조건적(절대적) 안전성(unconditionally secure): 무한한 컴퓨터 자원을 가져도 암호알고리즘을 해독할 수 없는 경우
  - 이 수준의 안전성을 제공하는 실용적인 알고리즘은 없음
- 계산적 안전성(computationally secure): 공격자의 능력이 가장 현실적으로 높다고 가정하였을 때 공격자가 암호알고리즘을 해독하기 위한 노력이 불합리하게 많은 컴퓨터 시간을 요구할 경우
  - 증명가능 안전성(provably secure): 어렵다고 알려진 문제와 등가임을 증명할 수 있는 경우
    - 예) 인수분해 문제
    - 안전성에 대한 절대적 증명은 아님
- 계산적 안전성을 증명하기 위해서는 다른 정의가 필요함

완벽한 안전성 vs. 의미론적 안전성

- 1949년에 C. Shannon은 암호알고리즘이 완벽한 안전성(perfect security)을 가지기 위한 조건을 제시함
- 확률을 이용하여 수학적으로 표현하면 다음과 같음

$$\Pr(M = m | C = c) = \Pr(M = m)$$

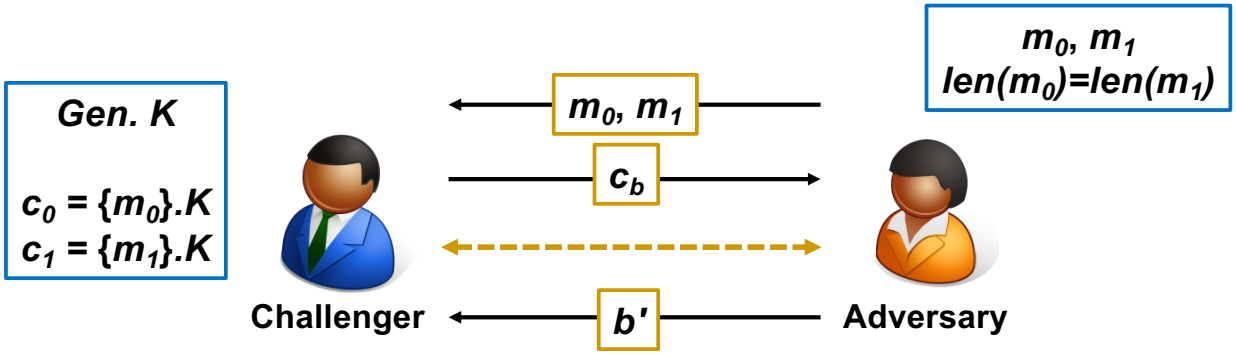
  - 의미. 암호문을 통해 이전에 몰랐던 평문에 대한 추가적인 정보를 전혀 얻을 수 없어야 함
- 완벽한 안전성을 가지기 위해서는 키 길이가 평문의 길이보다 커야 한다는 것을 증명함
- One time pad는 수동 공격에 대해 완벽한 안전성을 가지고 있다고 증명할 수 있음
- 완벽한 안전성을 가진 알고리즘은 키 길이 때문에 실제 현장에서 사용할 수 없음
- 무한한 컴퓨터 자원을 가져도 해독을 할 수 없으면 좋겠지만 계산적 안전성만 가져도 실제 현장에서 충분히 사용할 수 있음
  - 이 때문에 등장한 개념이 의미론적 안전성(semantic security)임

암호알고리즘의 안전성 (3/4)

- 의미론적 안전성: 암호문이 주어졌을 때 효율적으로 계산할 수 있는 모든 것을 암호문이 없어도 계산할 수 있어야 함
  - 계산적 안전성을 고려한 것임
  - 이 자체를 증명하기 어렵기 때문에 구별 불가 안전성(indistinguishability)이라는 개념을 사용함

암호알고리즘의 안전성 (3/4)

- 구별 불가 안전성: 아래와 같은 게임을 진행함
  - 이 게임은 $b = 0$ 게임과 $b = 1$ 게임으로 나누어짐
    - $b = 0$ 게임에서 공격자는 항상 m_0를 암호화함
  - 공격자는 자신이 어떤 게임에 참여하고 있는지 구분하는 것이 목표
  - 이 게임을 여러 번 진행할 수 있음 (분석하는 상황에 따라)
  - 게임은 두 종류이기 때문에 이길 확률은 50%임
  - 실제 이길 확률이 그것보다 높더라도 그 정도가 무시할 정도로 작으면 안전한 알고리즘임



암호알고리즘의 안전성 (4/4)

- NM 특성(Non-Malleability): NM 특성을 만족하는 알고리즘에 의해 생성된 암호문은 그것의 평문을 알고 있는지 여부와 상관없이 의미 있는 평문으로 복호화되도록 이 암호문을 다른 암호문으로 변경할 수 없음
  - 의미론적으로 안전하더라도 NM 특성을 만족하지 않을 수 있음
  - 예) XOR 연산을 이용한 암호화: $C = M \oplus K$, $M = C \oplus K$
    - NM 특성을 만족할까?
      - C의 특정 위치에 있는 비트를 토글하면 M도 같은 위치에서 변화함
      - One-time pad는 NM 특성을 만족 못함
- 안전한 암호알고리즘과 안전한 MAC 함수를 이용하여 인증 암호화를 하면 NM 특성을 제공할 수 있음
  - 변경할 수 있지만 변경한 사실을 알 수 있음

암호해독 공격의 분류 (1/2)

- 전사 공격(brute-force attack): 가능한 모든 키를 검사하는 방법 (exhaustive key search)
  - 암호해독 공격은 아님
- 암호해독 공격의 분류
  - 암호문 단독 공격(ciphertext-only attack): 공격자가 암호문만 얻을 수 있는 경우
  - 기지 평문 공격(known-plaintext attack): 공격자가 특정한 개수의 평문과 암호문 쌍만을 얻을 수 있음
  - 선택 평문 공격(chosen-plaintext attack): 특정한 개수의 평문과 암호문 쌍을 얻을 수 있지만 공격자는 자신이 원하는 평문을 선택할 수 있음
    - 이 선택은 한번에 이루어져야 함
  - 선택 암호문 공격(chosen-ciphertext attack): 특정한 개수의 평문과 암호문 쌍을 얻을 수 있지만 공격자는 자신이 원하는 암호문을 선택할 수 있음
    - 공개키 암호알고리즘과 관련된 공격 (선택 평문 공격 능력 포함)

암호해독 공격의 분류 (2/2)

- 적응적(adaptive) 선택 평문 공격: 이전에 얻은 쌍들을 바탕으로 추가적으로 원하는 평문을 선택할 수 있음
- 적응적 선택 암호문 공격: 이전에 얻은 쌍들을 바탕으로 추가적으로 원하는 암호문을 선택할 수 있음
- 이들 공격은 실제 이루어질 수 있는 공격이지만 암호알고리즘이 얼마나 안전한지 검사하기 위한 가상의 공격으로 생각할 수 있음

공격의 복잡성 척도

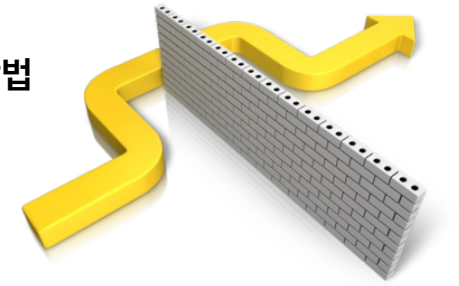
- 데이터 복잡성(data complexity): 공격이 성공하기 위해 필요한 데이터의 양
  - 공격에 성공하기 필요한 평문/암호문 쌍
- 처리 복잡성(processing complexity): 공격이 성공하기 위해 필요한 시간 (work factor)
  - 병렬처리 가능 여부
  - precomputation: 예) 2G 이동 통신: 2개월 동안 2TB 데이터를 생성한 후 몇 초 내에 세션키를 계산하였음
- 저장공간 요구사항(storage requirement): 공격하기 위해 필요한 메모리 공간

알고리즘에 대한 공격 결과

- 완전 성공(total break): 암호키를 발견
- 광역 성공(global deduction): 암호키를 발견하지 못하였지만 복호화할 수 있는 알고리즘을 발견
  - 목표하는 사용자의 서명을 무조건적으로 위조가 가능
- 인스턴스 성공(instance deduction): 어떤 한 암호문으로부터 그것의 평문을 얻어냄
  - 목표하는 사용자의 일부 서명을 위조 가능
  - 존재 위조(existential forgery): 어떤 메시지에 대한 서명을 위조할 수 있지만 공격자는 위조할 수 있는 메시지에 대한 어떤 선택권이 없는 경우
  - 선택적 위조(selective forgery): 특정 종류의 메시지에 대한 서명을 위조할 수 있음
- 정보 추출(information deduction): 암호문으로부터 평문의 일부나 암호키와 관련된 정보를 얻어냄

부채널 공격

- 부채널(side channel): 알고리즘의 입력과 출력 외에 다른 정보를 이용하는 것
- 예1) 시간(timing) 정보: 입력이 주어졌을 때 출력이 계산되는데 소요되는 시간
- 예2) 전력소비(power consumption) 정보: 입력이 주어졌을 때 출력을 계산하기 위해 소요된 전력
  - 키 비트 값에 따라 소요되는 시간 또는 소비되는 전력의 차이가 있음
- 심화 학습을 이용한 분석 기술까지 등장
- 방어하는 방법
  - 부채널로 노출되는 정보를 얻을 수 없도록 하는 방법
  - 부채널로 노출되는 정보를 줄이는 방법
    - 편차 또는 연관성을 줄이는 방법



```
bool equals(const byte* a, const byte* b, size_t size){
    for(size_t i{0}; i<size; ++i)
        if(a[i]!=b[i]) return false;
    return true;
}
```

```
bool equals(const byte* a, const byte* b, size_t size){
    byte ret{0};
    for(size_t i{0}; i<size; ++i)
        ret |= (a[i]^b[i])
    return ret==0;
}
```


양자 컴퓨팅과 암호기술

- 양자 컴퓨팅은 아직 현실화되어 있지는 않음
- 양자 컴퓨팅 원리에 대해서는 오래전부터 이해하고 있음
- 두 가지 중요한 발견
  - Glover 알고리즘. 양자 컴퓨팅을 이용하면 $O(n)$ 검색 비용이 필요한 것을 $O(n^{1/2})$에 할 수 있음
    - 대칭키의 길이를 2배로 확대해야 함
    - 해시의 강한 충돌은 양자 컴퓨팅을 이용하면 $O(n^{1/3})$이 되며, 약한 충돌은 $O(n^{1/2})$임
  - Shor 알고리즘. 다차시간에 인수분해, 이산대수 문제를 해결할 수 있음
    - 현재 사용되고 있는 공개키 기술은 사용할 수 없음
- 양자내성암호(post-quantum cryptography)에 대한 연구가 활발하게 진행되고 있음 (NIST는 2017년부터 양자내성암호 표준을 진행하고 있음)
  - 대안 1. NP-Hard 문제 기반 공개키 암호알고리즘
  - 대안 2. 해시함수 기반 공개키 암호알고리즘

https://csrc.nist.gov/projects/post-quantum-cryptography