# 전자서명에 사용되는 해시 함수 (Hash Functions)

https://youtu.be/cWEDyQhnjp0

HSU 한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB

해시 함수의 주요 특성

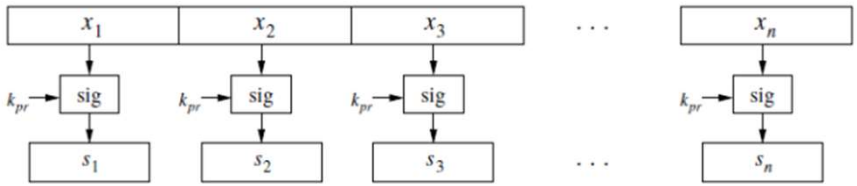
해시 함수의 안정성 분석

해시 함수 SHA-1의 작동 방법

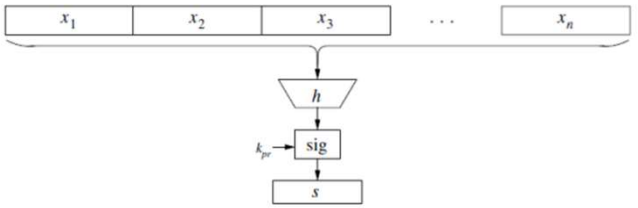
# 해시 함수의 주요 특성

## 긴 메시지의 서명을 할 때

가장 직관적으로 생각한 접근 방법: 블록 암호 사용




해결책




**Problem1: 높은 계산 부담**

**Problem2: 메시지 오버헤드**

예를 들어 1-MB의 메시지를 보낼 때, 1-MB의 RSA 서명도 보내야 하므로 총 2-MB의 전송이 요구됨.

**Problem3: 안정성의 한계**

공격자는 각 메시지와 그에 대응하는 서명을 삭제하거나, 메시지와 서명을 재배열할 수 있다.

또한 이전에 전송된 메시지나 서명의 일부를 이용해 새로운 메시지와 서명을 만들 수도 있다.

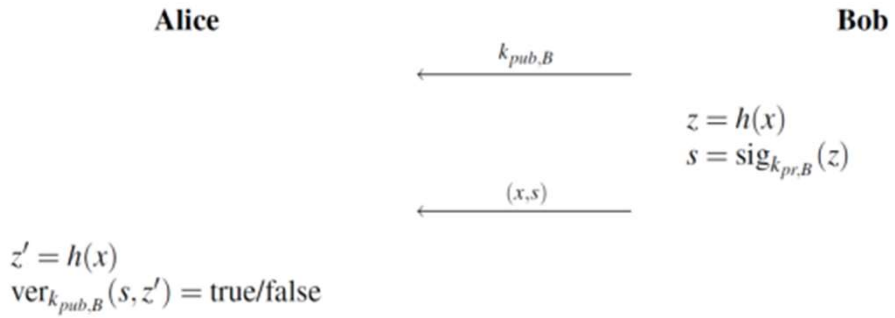
개별적인 블록 내부에서의 조작은 불가능하더라도 전체 메시지를 보호할 수는 없음.

보안 및 성능상의 이유로 임의의 길이를 갖는 전체 메시지에 대한 서명을 생성하기 보다는 **메시지 요약(Message Digest)**에 대한 서명을 생성하는 것이 효율적임.

따라서 해시 함수 필요!!

# 해시 함수의 주요 특성

**해시 함수를 이용한 전자서명의 기본 프로토콜**




Bob은 메시지 x에 대한 해시값 z를 계산하여 개인키(private-key)를 이용해 서명함.

Alice는 받은 메시지 x에 대한 해시값 z'을 계산한 후, Bob의 공개키(public-key)를 이용하여 서명 S를 검증함.

서명의 생성 및 검증은 메시지가 아니라 해시값을 이용하여 수행됨.

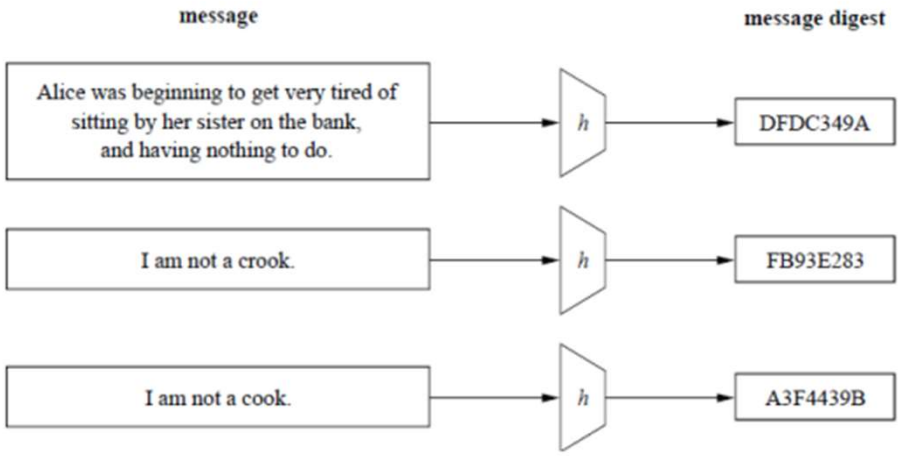
# 해시 함수의 주요 특성

## 해시 함수의 바람직한 입출력 동작 원리

1. 해시 함수는 임의의 크기를 갖는 메시지에 대해 적용 가능해야 하며, 이 때 함수 h()는 계산적으로 효율적이어야 함.

2. 해시 함수의 출력값은 고정된 크기이며 입력값의 크기에 독립적이어야 함.

3. 출력값은 입력값의 변화에 매우 민감하여, 입력이 약간 변경 되더라도 매우 달라져야 함.



# 해시 함수의 주요 특성

**해시 함수의 3가지 보안 특성**

1. 일방향성(One-Wayness)

   주어진 임의의 출력값 z에 대해, z = h(x)를 만족하는 입력값 x를 찾는 것이 불가능함.

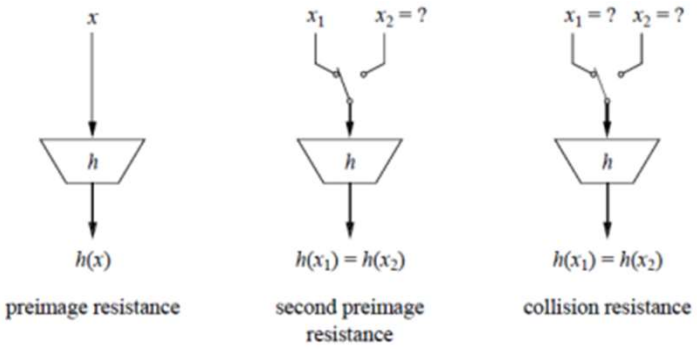
   즉 h()는 일방향 함수임.

2. 약한 충동 저항성

   주어진 입력값 x1에 대해 h(x1) = h(x2), x1 != x2를 만족하는 다른 임의의 입력값 x2를 찾는 것이 불가능함

3. 충돌 저항성

   h(x1) = h(x2)를 만족하는 임의의 서로 다른 두 입력값 x1, x2를 찾는 것이 불가능함.



# 해시 함수의 안정성 분석

**해시 함수의 경우 대부분의 문제가 충돌 저항성에 기인한다고 알려져 있음.**

50%의 확률로 충돌의 경우를 찾는 것이 얼마나 어려울까?

**생일 공격(Birthday Attack)** 문제 :

어떤 그룹에서 생일이 같은 두 명이 있을 확률이 50% 이상 되려면, 몇 명 정도의 그룹이 필요할까?

→ 직관적으로 365/2 ≈ 183일 것 같지만 실제로 x

$$P(m) = 1 - 1 \times \left(1 - \frac{1}{365}\right) \times \left(1 - \frac{2}{365}\right) \times \cdots \times \left(1 - \frac{m-1}{365}\right) = 1 - \frac{365!}{365^m (365-m)!} \geq 0.5 \rightarrow m = 23\text{명}$$

60명만 있어도 생일 같은 두 명이 있을 확률이 99.4%에 달함.

# 해시 함수의 안정성 분석

**해시 함수에 대한 충돌 탐색의 경우**

$2^n$에 대해 $t$개의 해시 함수 입력값 $(x_1, x_2, \ldots, x_t)$ 중 충돌이 없을 확률:

$$P(\text{No collision}) = 1 \times \left(1 - \frac{1}{2^n}\right) \times \left(1 - \frac{2}{2^n}\right) \times \cdots \times \left(1 - \frac{t-1}{2^n}\right) = \prod_{i=1}^{t-1}\left(1 - \frac{i}{2^n}\right) \approx \prod_{i=1}^{t-1} e^{-\frac{i}{2^n}} = e^{-\frac{t(t-1)}{2^{n+1}}}$$

**적어도 최소 한 번의 충돌이 있을 확률** $\lambda = P(\text{At least one collision}) = 1 - P(\text{No collision})$

→ $\lambda \approx 1 - e^{-\frac{t(t-1)}{2^{n+1}}} \approx 1 - e^{-\frac{t^2}{2^{n+1}}} \rightarrow t \approx 2^{\frac{n+1}{2}}\sqrt{\ln\left(\frac{1}{1-\lambda}\right)}$

위 식은 해시 출력값의 길이 n에 대해 충돌에 필요한 해시된 메시지 개수 t와 충돌 확률 $\lambda$ 와의 관계를 보여줌

즉, 충동을 찾기 위해 해시하는데 필요한 메시지의 수는 가능한 **출력값 개수의 제곱근**에 거의 근사함

따라서 x-bit의 보안 수준을 가지려면 해시 함수의 결과로써 출력값이 2x-bit 이상이어야 한다.

현재 최소 160-bit 이상이어야 한다고 보고 있다.

# 해시 함수 SHA-1의 작동 방법

## SHA(Secure Hash Algorithm) 계열

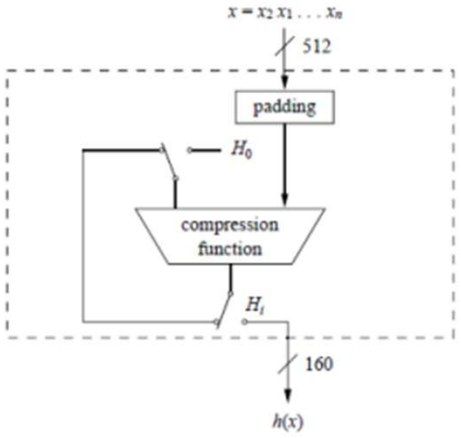
SHA-1: 512-bit 입력값과 160-bit 출력값을 가지며, 분석 공격(Analtical Attack)을 고려하지 않을 시 약 $2^{80}$의 충돌 저항성을 갖음.

MD4 계열, Merkle-Damgard 구조에 기반을 두고 있으며. 압축 함수(Compression Function)가 블록 암호처럼 작동함.

최대 $2^{64} - 1$ bit의 길이를 가지는 입력 메시지에 대해 160-bit의 출력을 만들어 냄.

실제 해시 계산 전에, 압축 함수는 입력 메시지를 512-bit 길이의 메시지 단위(Chunk)로 패딩(Padding)하는 전처리를 하며 각 20개의 라운드를 갖는 4개의 단계로 이루어진 총 80라운드로 구성됨.

| Algorithm | | Output [bit] | Input [bit] | No. of rounds | Collisions found |
|---|---|---|---|---|---|
| **MD5** | | 128 | 512 | 64 | yes |
| **SHA-1** | | 160 | 512 | 80 | not yet |
| **SHA-2** | **SHA-224** | 224 | 512 | 64 | no |
| | **SHA-256** | 256 | 512 | 64 | no |
| | **SHA-384** | 384 | 1024 | 80 | no |
| | **SHA-512** | 512 | 1024 | 80 | no |



# 해시 함수 SHA-1의 작동 방법

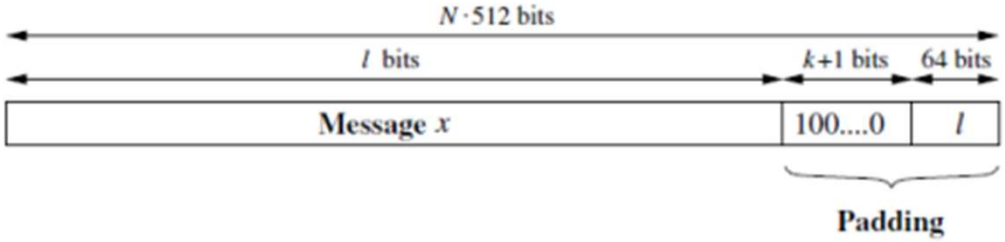
**SHA-1의 전처리**

1. 입력 메시지 x는 512-bit의 배수가 되도록 패딩됨.
2. 내부 처리를 위해 패딩된 메시지는 블록 단위로 분할됨.
3. 연산을 위한 초기값 H0는 사전에 정의됨

**패딩**

메시지 x의 길이가 ℓ-bit라고 가정

전제 메시지의 길이가 512-bit의 배수가 되도록, k + 1bit의 100…0을 패딩하고 64-bit의 길이를 갖는 ℓ의 이진 표현을 추가함. 즉, $k = 512 - 64 - 1 - \ell = 448 - (\ell + 1) \bmod 512$.



# 해시 함수 SHA-1의 작동 방법

**SHA-1의 해시 계산**

각 메시지 블록 xi는 20개의 라운드로 구성된 4단계를 거쳐 처리.

각 80 라운드에 대해 각 32-bit 워드 W0, …, W79를 계산하는 스케줄을 가짐.

$$W_j = \begin{cases} x_i^{(j)} & 0 \le j \le 15 \\ (W_{j-16} \oplus W_{j-14} \oplus W_{j-8} \oplus W_{j-3}) \lll 1 & 16 \le j \le 79 \end{cases}$$

32-bit 크기의 5개의 작업 레지스터 $A, B, C, D, E$가 이용됨.

해시값 $H_i$는 5개의 32-bit 워드 $H_i^{(0)}, H_i^{(1)}, \ldots, H_i^{(4)}$로 구성되며, 초기 해시값은 $H_0$으로 설정됨.

최종 해시값 $H_n$은 SHA-1의 출력값 $h(x)$과 동일함.

$t$ 단계의 $j$ 라운드에서의 계산은 다음과 같음.

$$A, B, C, D, E = \left(E + f_t(B,C,D) + (A)_{<<<5} + W_j + K_t\right), A, (B)_{<<<30}, C, D$$

| Stage $t$ | Round $j$ | Constant $K_t$ | Function $f_t$ |
|---|---|---|---|
| 1 | 0…19 | $K_1 =$ 5A827999 | $f_1(B,C,D) = (B \wedge C) \vee (\bar{B} \wedge D)$ |
| 2 | 20…39 | $K_2 =$ 6ED9EBA1 | $f_2(B,C,D) = B \oplus C \oplus D$ |
| 3 | 40…59 | $K_3 =$ 8F1BBCDC | $f_3(B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$ |
| 4 | 60…79 | $K_4 =$ CA62C1D6 | $f_4(B,C,D) = B \oplus C \oplus D$ |

$A = H_0^{(0)} =$ 67452301

$B = H_0^{(1)} =$ EFCDAB89

$C = H_0^{(2)} =$ 98BADCFE

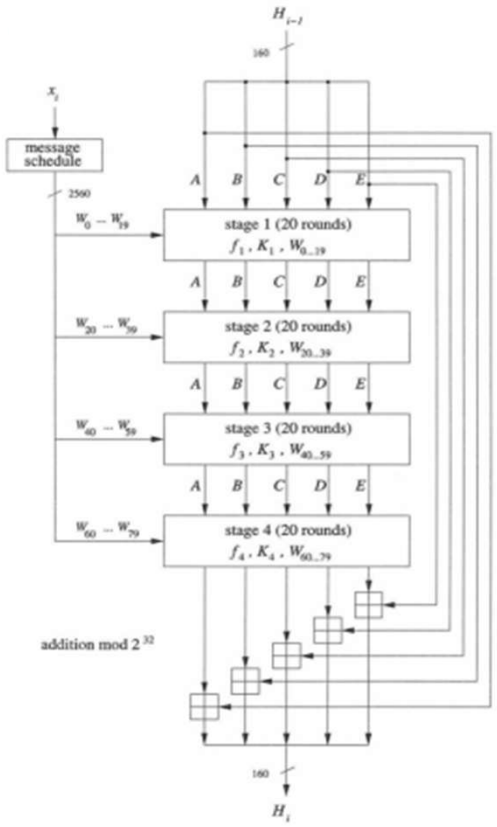
$D = H_0^{(3)} =$ 10325476

$E = H_0^{(4)} =$ C3D2E1F0

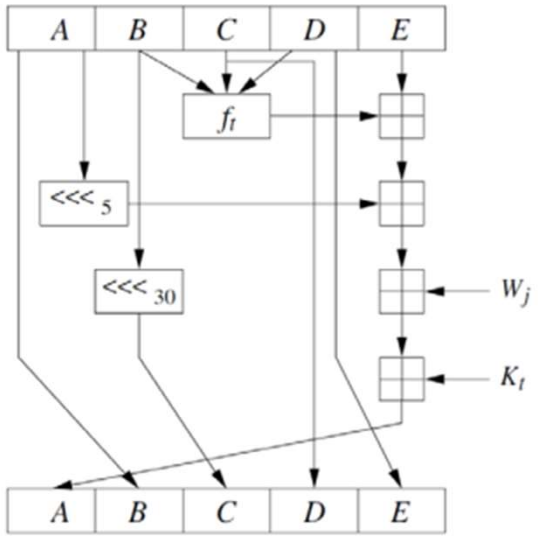
다음과 같은 5개의 32-bit 길이를 갖는 고정된 워드가 주어짐.

# 해시 함수 SHA-1의 작동 방법

**SHA-1 구조**




**SHA-1의 t단계의 j 라운드**



# Q & A