

정보보호론 #2

암호학적 해시 함수의 이해

Prof. Byung Il Kwak

Review

- 정보 보안의 이해

- 보안 3요소 + 2

- Hash function
- Message Authentication

CONTENTS

❑ Hash function

Cryptography - hash function

❑ 블록체인의 Trustless network

❑ 제 3자에 대한 신용이 필요 없음

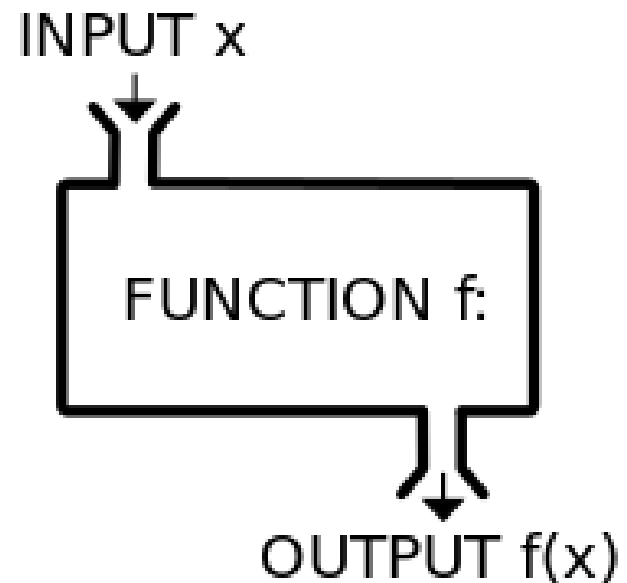
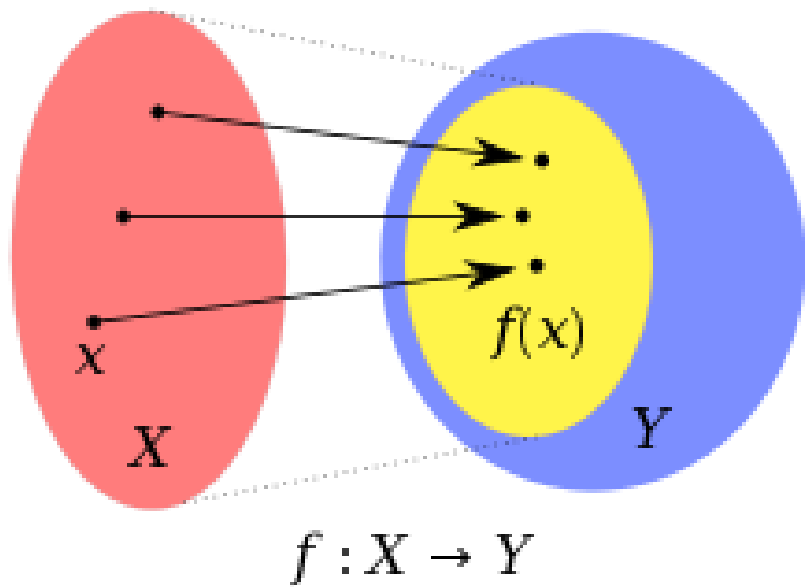
- 은행과 같은 중앙에서 관리하는 특정 노드가 필요하지 않음

❑ Trustless network에서의 신뢰성을 보장하기 위한 방법

- Hash function을 활용한 암호화
- 디지털 서명 (Public key & Private key)
- 메시지 인증 코드 (Message authentication code algorithms)
- ...

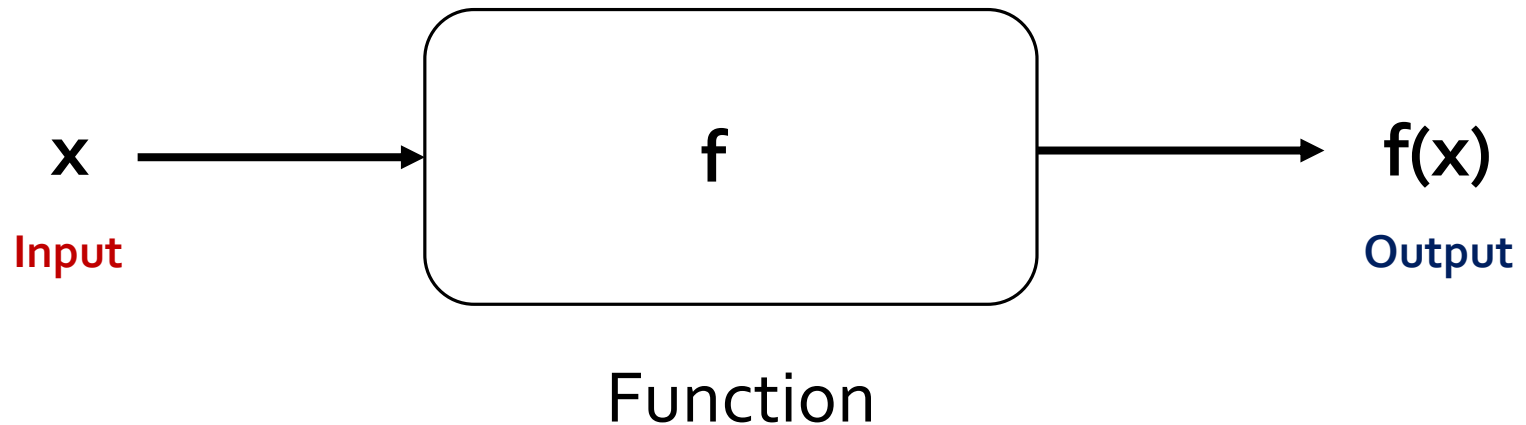
Cryptography - hash function

□ 함수 (Function) 이란?



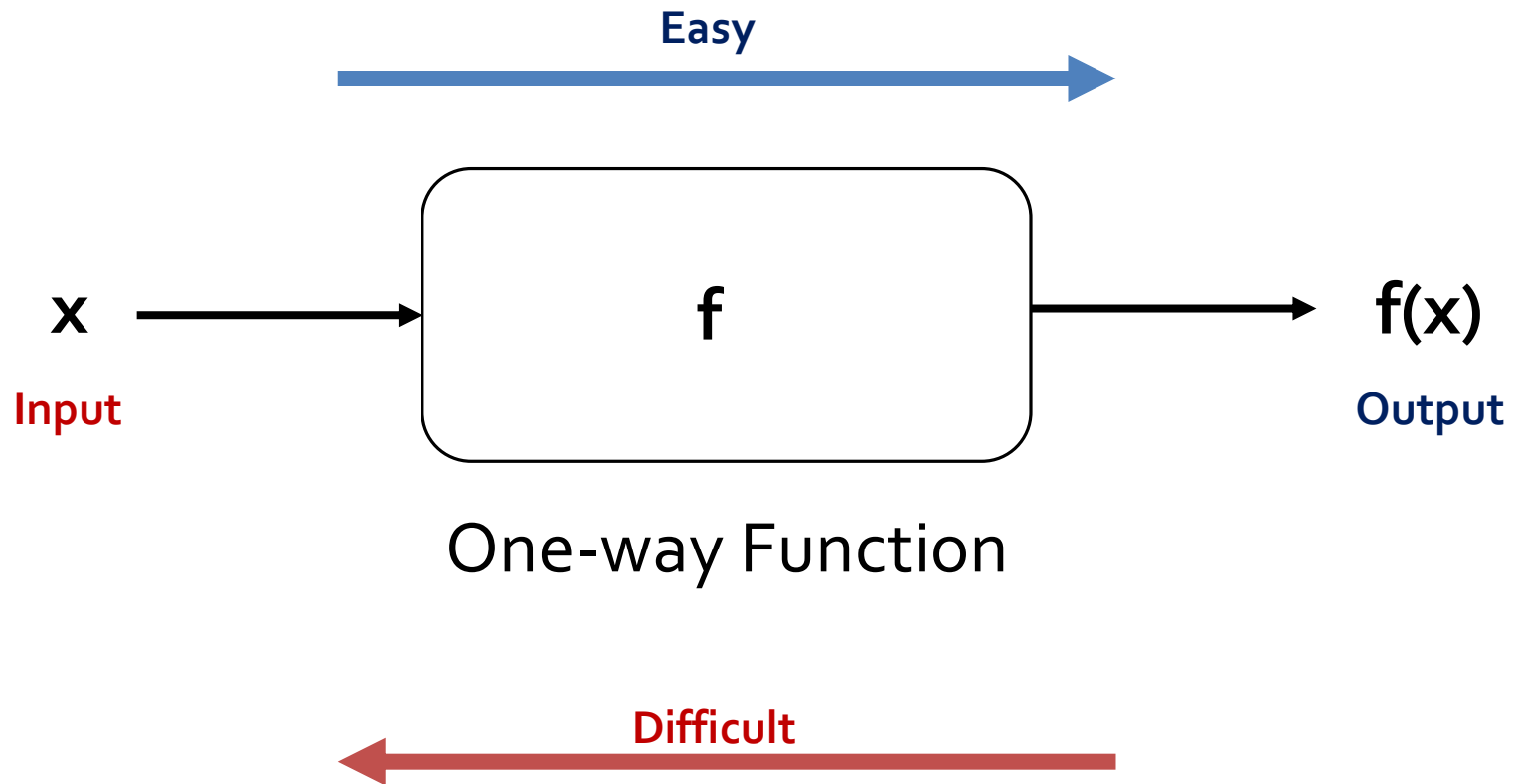
Cryptography - hash function

□ 함수 (Function) 이란?



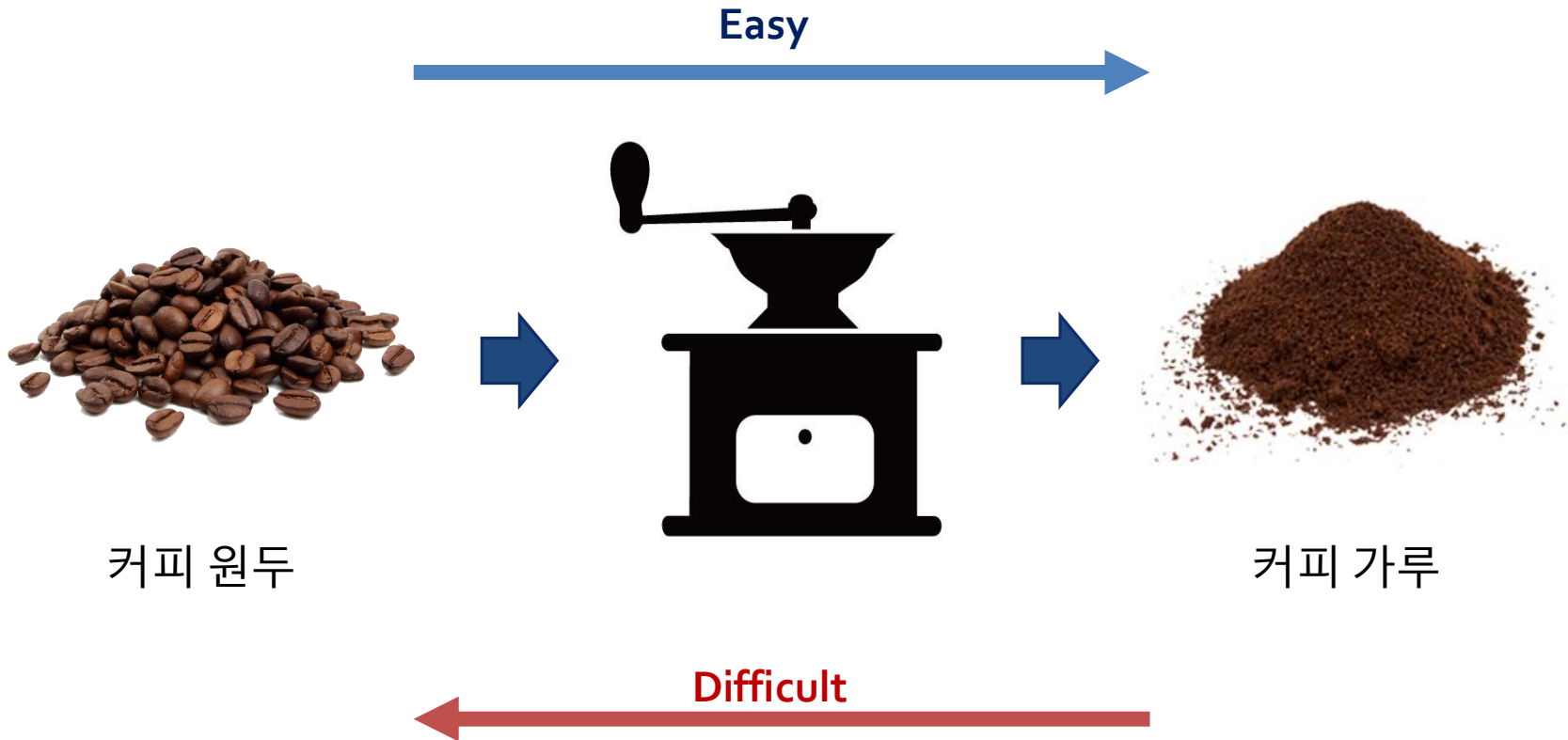
Cryptography - hash function

- 일 방향 함수 (One-way function) 이란?



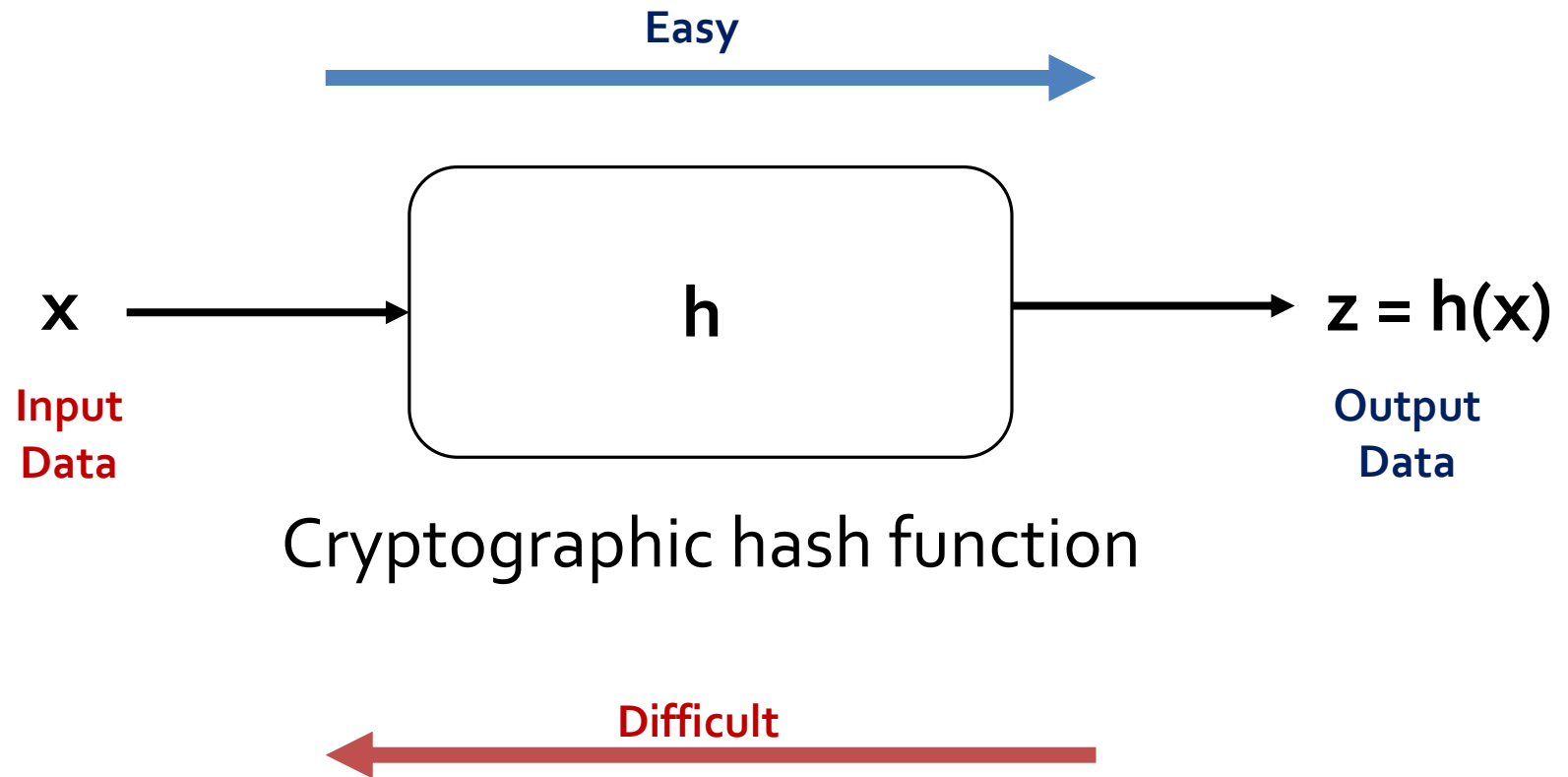
Cryptography - hash function

□ 일 방향 함수 (One-way function) 예시



Cryptography - hash function

□ Hash function (해시 함수)



Cryptography - hash function

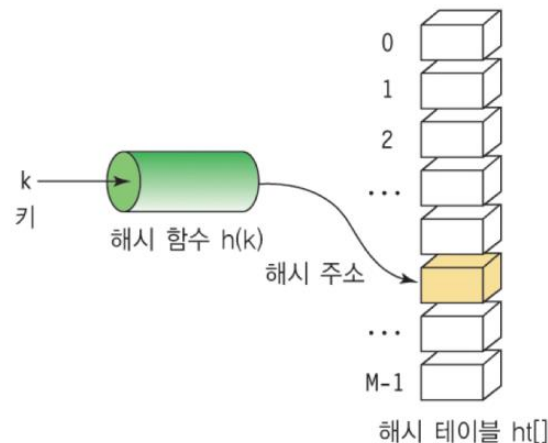
□ Hash function (해시 함수)

▣ 해시함수는 어떤 길이의 데이터를 입력해도 고정된 길이의 결과(해시 값)를 출력

▣ 일반적으로 Hash Table (해시 테이블)을 구성

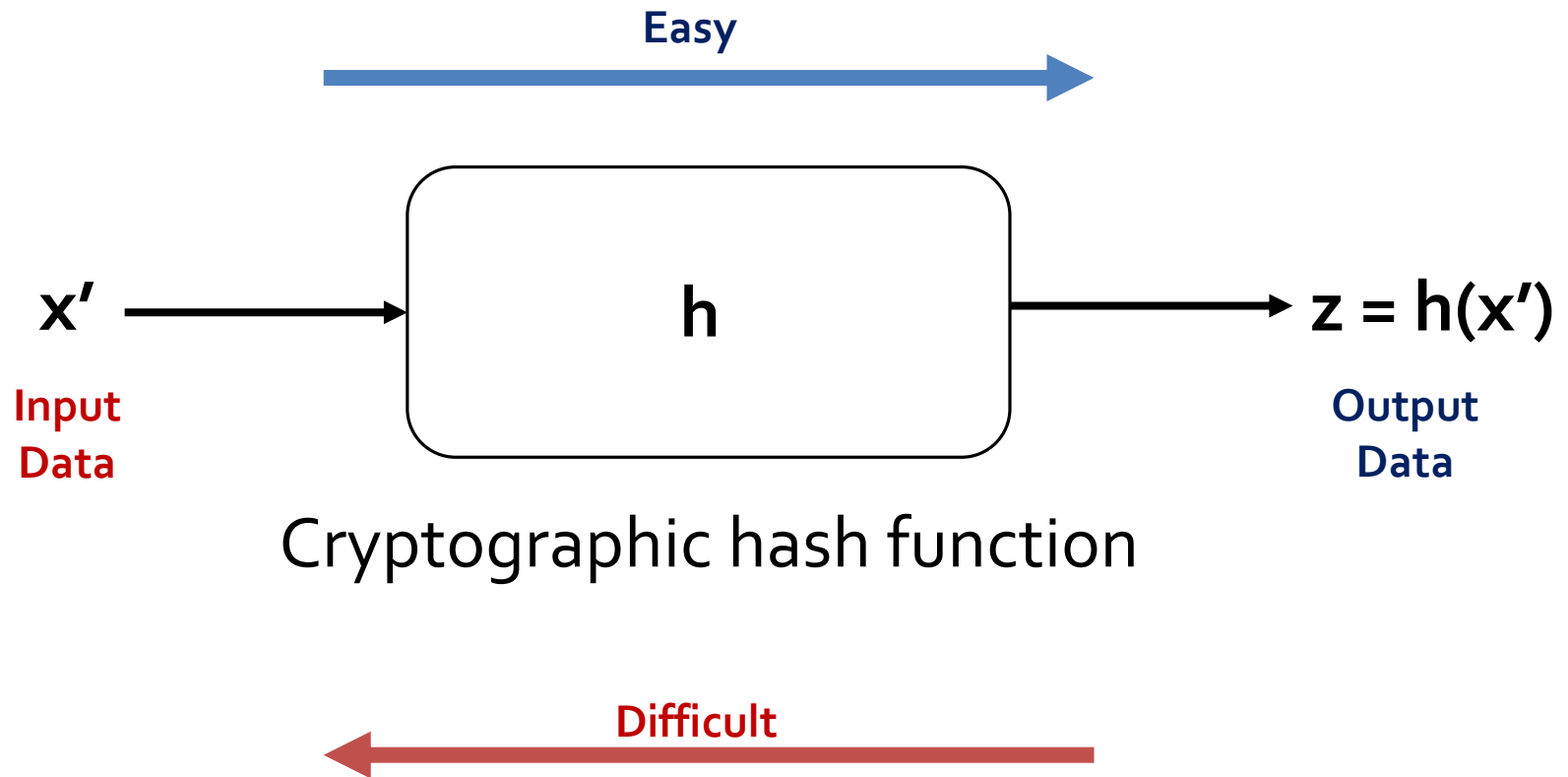
- 키(Key)에 데이터(Value)를 저장하는 데이터 구조

■ 예) Python에서 Dictionary 타입이 Hash Table의 예로 볼 수 있음



Cryptography - hash function

□ Hash function (해시 함수)



Cryptography - hash function

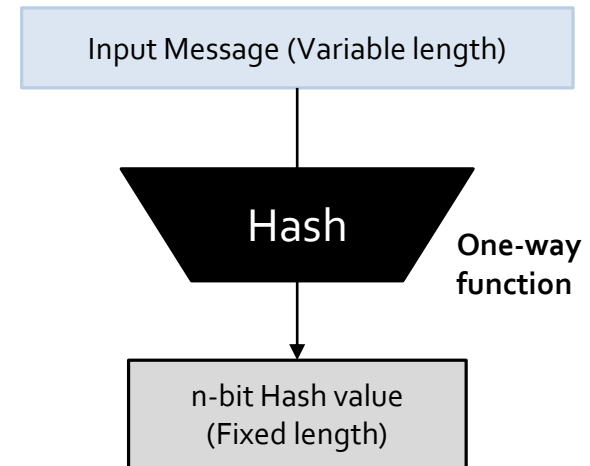
□ Hash function (해시 함수)

- ▣ Input data를 x , hash function 를 h , output을 z 라고 할 때,
 - x 는 z 의 프리이미지 (preimage)라고도 함
 - z 는 이미지 (image)라고 함
- ▣ 이러한 경우, 해시 함수 h 는 일대일의 함수가 아니기 때문에, 동일한 해시 값 z 를 가질 수 있는 Input data x 가 여러 개 존재할 수 있음

Cryptography - hash function

□ 해시 함수의 조건

- **Pre-image resistance:** Given a hash value h it should be difficult to find any message m such that $h = \text{hash}(m)$
- **Second pre-image resistance:** Given an input m_1 , it should be difficult to find a different input m_2 such that $\text{hash}(m_1) = \text{hash}(m_2)$
- **Collision resistance:** It should be difficult to find two different messages m_1 and m_2 such that $\text{hash}(m_1) = \text{hash}(m_2)$



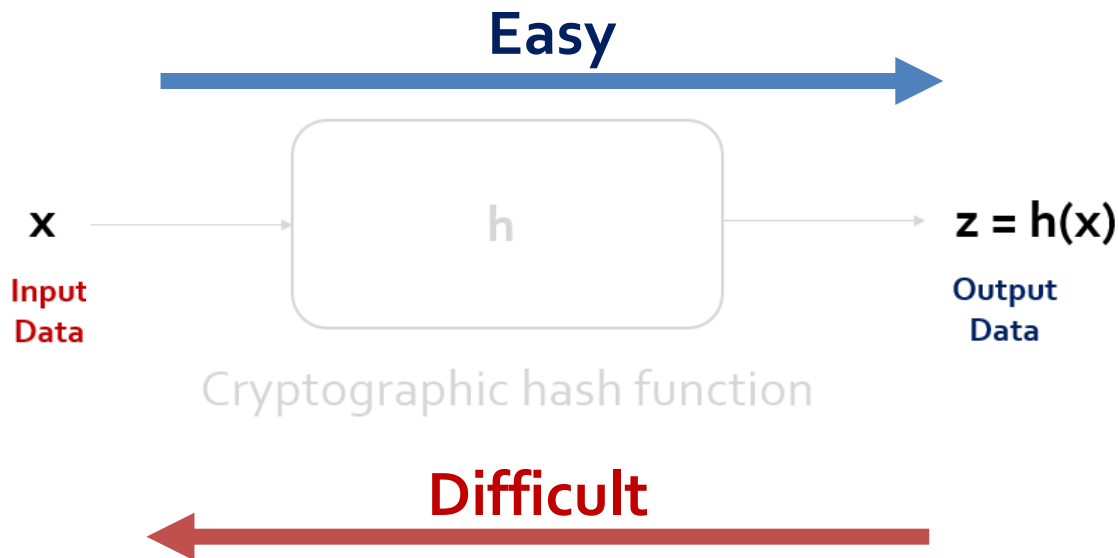
▣ 해시 함수는 'Digital fingerprints (디지털 지문)'으로도 사용됨

Cryptography - hash function

□ 해시 함수의 조건

▣ 프리 이미지 저항성 (Preimage resistance)

- 해시 값 z 로 문서 x 를 찾는 것이 어려워야 함
- 해시의 단방향적 성질을 의미
 - 주어진 z 를 통해 x 를 찾는 것은 어려움

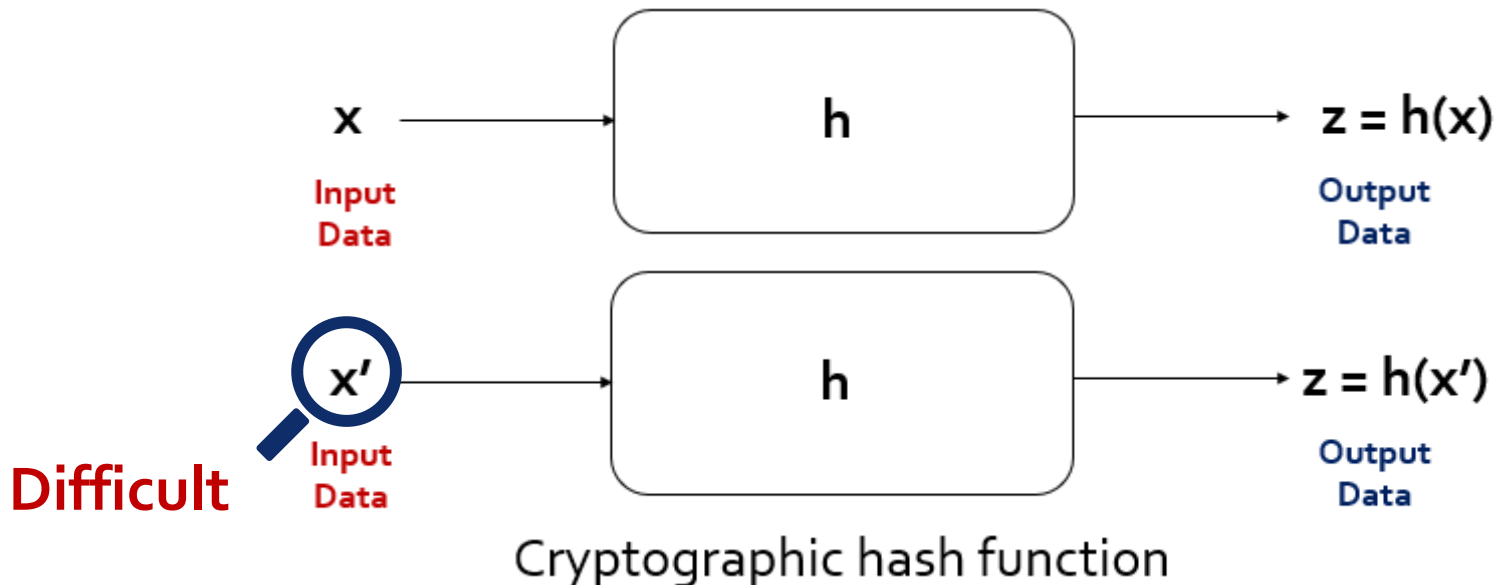


Cryptography - hash function

□ 해시 함수의 조건

▣ 제 2 프리 이미지 저항성 (Second preimage resistance)

- Input Data x 가 주어졌을 때, 해시 값 z 를 갖는 또 다른 Input Data x' 의 발견이 어려워야 함
 - 주어진 x, z 에 대해, x 를 $h()$ 에 적용한 $z = h(x)$ 가 있을 경우,
 $z = h(x) = h(x')$ 인 x' 를 찾기가 어려워함



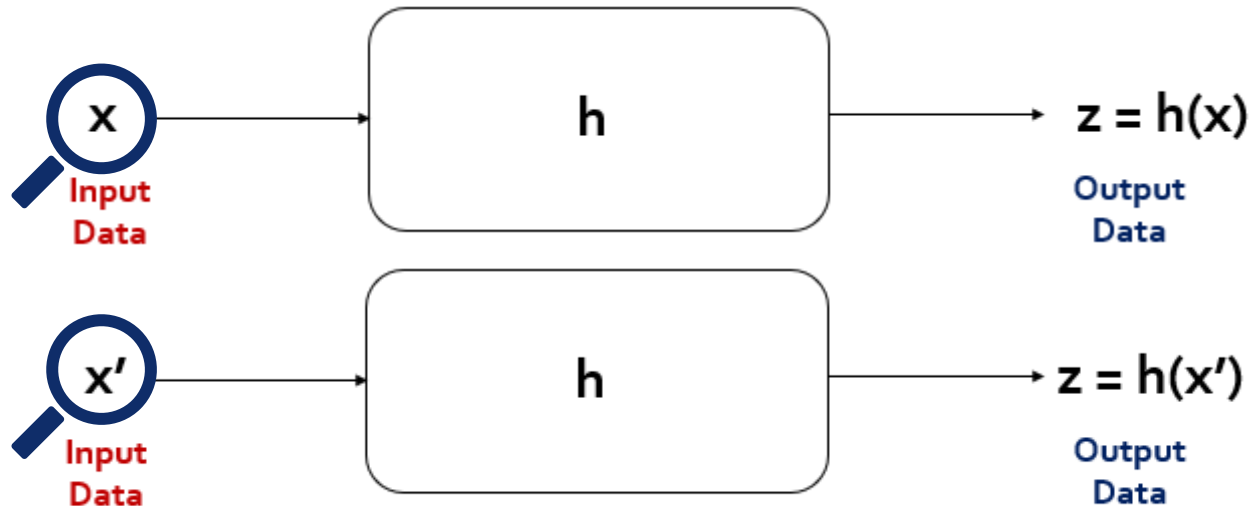
Cryptography - hash function

□ 해시 함수의 조건

▣ 충돌 저항성 (Collision resistance)

- 동일한 해시 값 z 를 갖는 임의의 두 Input Data x, x' 을 찾는 것이 어려워야 함
 - 주어진 정보가 없을 때, 해시 값 z 에 대한 $h(x) = h(x') = z$ 를 만족하는 (x, x') 을 찾는 것이 어려워야 함

Difficult



Cryptographic hash function

Cryptography - hash function

□ 해시 함수의 조건

- 프리 이미지 저항성 (Preimage resistance) -> **OK**
- 제 2 프리 이미지 저항성 (Second preimage resistance) ??
- 충돌 저항성 (Collision resistance) ??

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

Cryptography - hash function

□ 해시 함수

▣ 눈사태 효과 (Avalanche effect)

- 암호 알고리즘에서 입력 값에 미세한 변화를 줄 경우 출력 값에 상당한 변화가 일어나는 성질 [source: *Wikipedia*]
- 해시 함수에서의 눈사태 효과 (Avalanche effect)
 - 해시 함수에서 Input Data가 1 bit만 차이를 보여도 Output Data는 다르게 나타남

Hash Algorithm Types	Input Data	Hash values
SHA1	1	356A192B7913B04C54574D18C28D46E6395428AB
	2	DA4B9237BACCCDF19C0760CAB7AEC4A8359010B0
	3	77DE68DAECD823BABBB58EDB1C8E14D7106E83BB
	4	1B6453892473A467D07372D45EB05ABC2031647A

Cryptography - hash function

□ 해시 함수

▣ 눈사태 효과 (Avalanche effect)

```
I am Satoshi Nakamoto0 => a80a81401765c8eddee25df36728d732...
I am Satoshi Nakamoto1 => f7bc9a6304a4647bb41241a677b5345f...
I am Satoshi Nakamoto2 => ea758a8134b115298a1583ffb80ae629...
I am Satoshi Nakamoto3 => bfa9779618ff072c903d773de30c99bd...
I am Satoshi Nakamoto4 => bce8564de9a83c18c31944a66bde992f...
I am Satoshi Nakamoto5 => eb362c3cf3479be0a97a20163589038e...
I am Satoshi Nakamoto6 => 4a2fd48e3be420d0d28e202360cfbaba...
I am Satoshi Nakamoto7 => 790b5a1349a5f2b909bf74d0d166b17a...
I am Satoshi Nakamoto8 => 702c45e5b15aa54b625d68dd947f1597...
I am Satoshi Nakamoto9 => 7007cf7dd40f5e933cd89fff5b791ff0...
I am Satoshi Nakamoto10 => c2f38c81992f4614206a21537bd634a...
I am Satoshi Nakamoto11 => 7045da6ed8a914690f087690e1e8d66...
I am Satoshi Nakamoto12 => 60f01db30c1a0d4cbce2b4b22e88b9b...
I am Satoshi Nakamoto13 => 0ebc56d59a34f5082aaef3d66b37a66...
I am Satoshi Nakamoto14 => 27ead1ca85da66981fd9da01a8c6816...
I am Satoshi Nakamoto15 => 394809fb809c5f83ce97ab554a2812c...
I am Satoshi Nakamoto16 => 8fa4992219df33f50834465d3047429...
I am Satoshi Nakamoto17 => dca9b8b4f8d8e1521fa4eaa46f4f0cd...
I am Satoshi Nakamoto18 => 9989a401b2a3a318b01e9ca9a22b0f3...
I am Satoshi Nakamoto19 => cda56022ecb5b67b2bc93a2d764e75f...
```

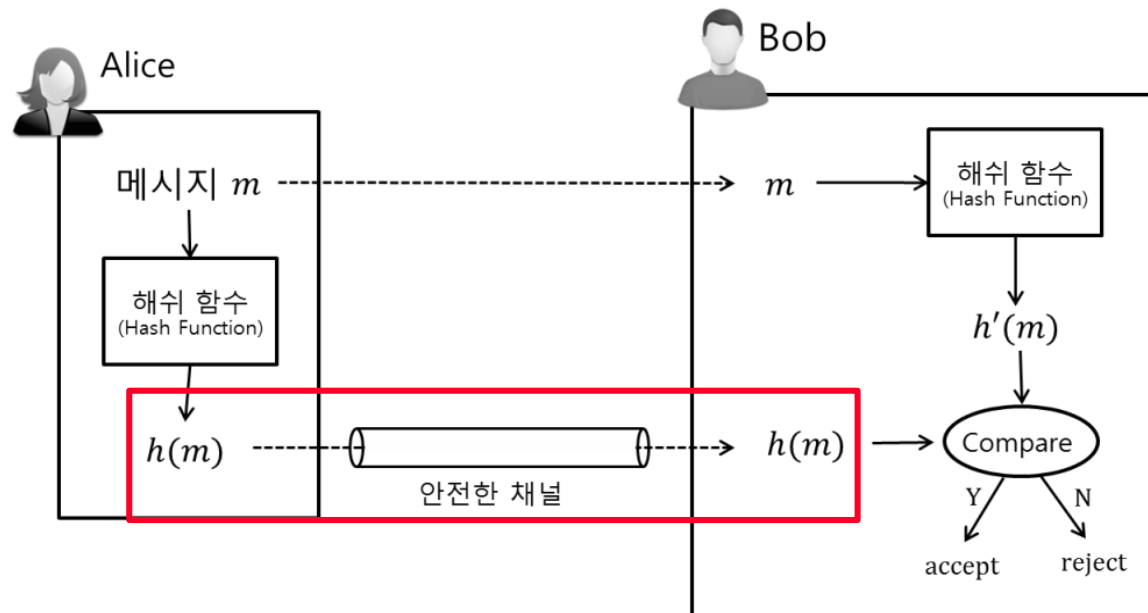
[Source: http://chimera.labs.oreilly.com/books/1234000001802/cho8.html#_proof_of_work_algorithm]

SHA: Secure Hash Algorithm

<https://passwordsgenerator.net/sha1-hash-generator/>

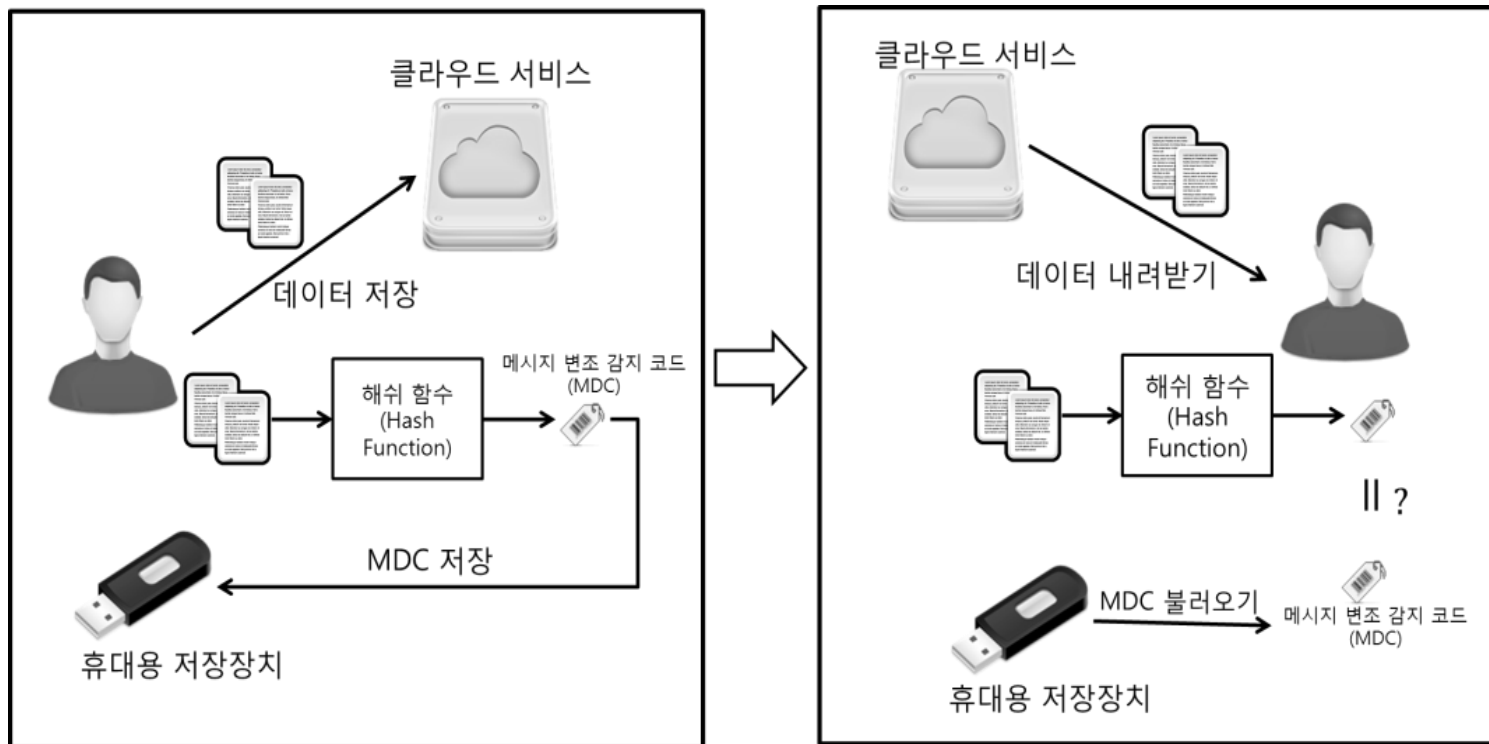
Cryptography - hash function

- 메시지 변조 감지 코드(Modification Detection Code - MDC)
 - ▣ 메시지 m 과 MDC $h(m)$ 은 분리될 수 있으며, $h(m)$ 은 변조되지 않도록 하는 것이 중요!
 - Alice는 메시지 m 에 대한 MDC $h(m)$ 생성
 - Alice \rightarrow Bob : m , $h(m)$, 단 $h(m)$ 은 안전한 채널로 전송



Cryptography - hash function

- MDC를 이용한 메시지 변조 감지 코드 활용 예
 - ▣ 클라우드 서비스에서 원본 메시지에 대한 무결성을 제공



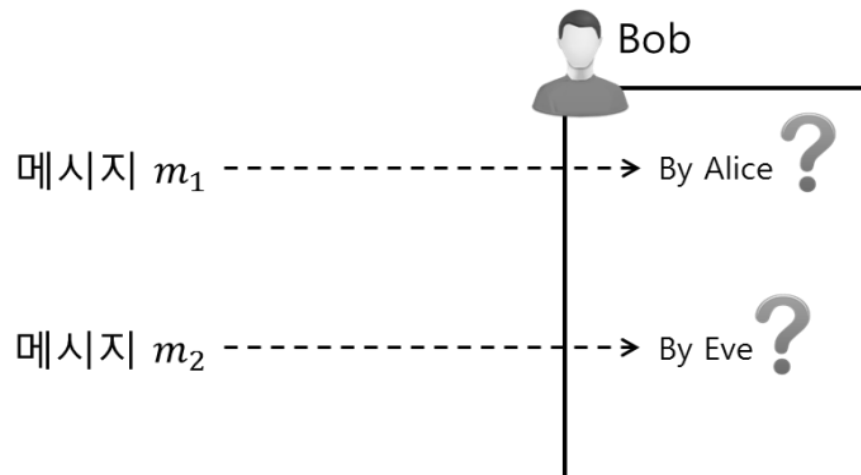
CONTENTS

- ❑ Message Authentication

Cryptography - hash function

□ Message Authentication Code (MAC)

- ▣ 메시지 근원 인증 (Message origin authentication)을 제공
 - 해쉬 함수를 이용한 MDC는 무결성만을 제공
 - 메시지의 출처를 확인할 수 없음
 - MAC은 무결성도 함께 제공



Cryptography - hash function

□ Message Authentication Code (MAC)

▣ MDC와 MAC의 특성 비교

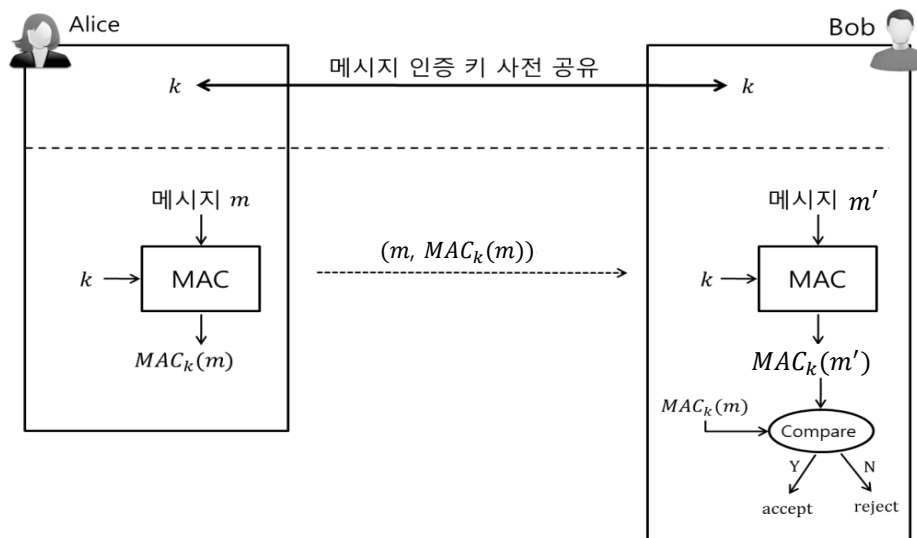
	변조 감지 코드(MDC)	메시지 인증 코드(MAC)
사전 키 공유	X	O
안전한 채널 사용	O	X
메시지의 무결성	O	O
데이터 출원 인증	X	O
부인 방지	X	X

Cryptography - hash function

□ Message Authentication Code (MAC)

▣ $MAC_k(m) = h(k, m)$

- Alice와 Bob은 MAC 에 사용할 키 k 를 사전에 공유함
- Alice는 메시지 m 에 대한 $MAC_k(m)$ 생성
- Alice \rightarrow Bob : $m, MAC_k(m)$
- Bob은 수신 메시지 m' 에 대한 $MAC_k(m')$ 생성
- $MAC_k(m) = ? MAC_k(m')$



Cryptography - hash function

- Message Authentication Code (MAC)
 - ▣ MAC 생성 방법

<u>해쉬 함수 기반</u> 메시지 인증 코드	Nested MAC
	HMAC
<u>블록 암호 알고리즘</u> 기반 메시지 인증 코드	CBC-MAC
	CMAC

□ Message Authentication Code (MAC)

▣ Nested MAC

- Nested MAC은 두 개의 해시 함수를 이용함
- 두 개의 해시 함수를 h^1, h^2 이라 한다면, 송/수신자는 각각의 해시 함수에 쓰일 키 k_1, k_2 를 공유하고 이 값을 이용하여 두 번의 해시를 통해 값을 계산함
- Nested MAC은 해시 함수 h^1, h^2 를 기반으로 하기 때문에 이 두 해시 함수의 안전성에 기반함

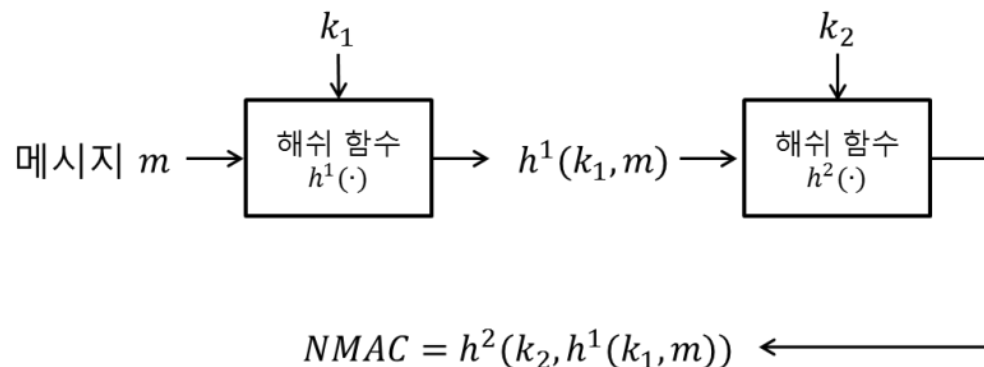
$$\text{NMAC}_{k_1, k_2}(m) = h^2(k_2, h^1(k_1, m))$$

Cryptography - hash function

□ Message Authentication Code (MAC)

▣ Nested MAC - 계산 방법

- 1. 키 k_1 의 오른쪽에 메시지 m 을 패딩함
- 2. 해시 함수 h^1 를 이용하여 1번의 결과값에 대한 해시값을 계산함
- 3. 키 k_2 의 오른쪽에 2번의 결과 값을 패딩함
- 4. 2와 같은 방법으로 해시 함수 h^2 를 이용하여 3번의 결과 값에 대한 해시값을 계산한 최종 값인 Nested MAC을 계산



Cryptography - hash function

□ Message Authentication Code (MAC)

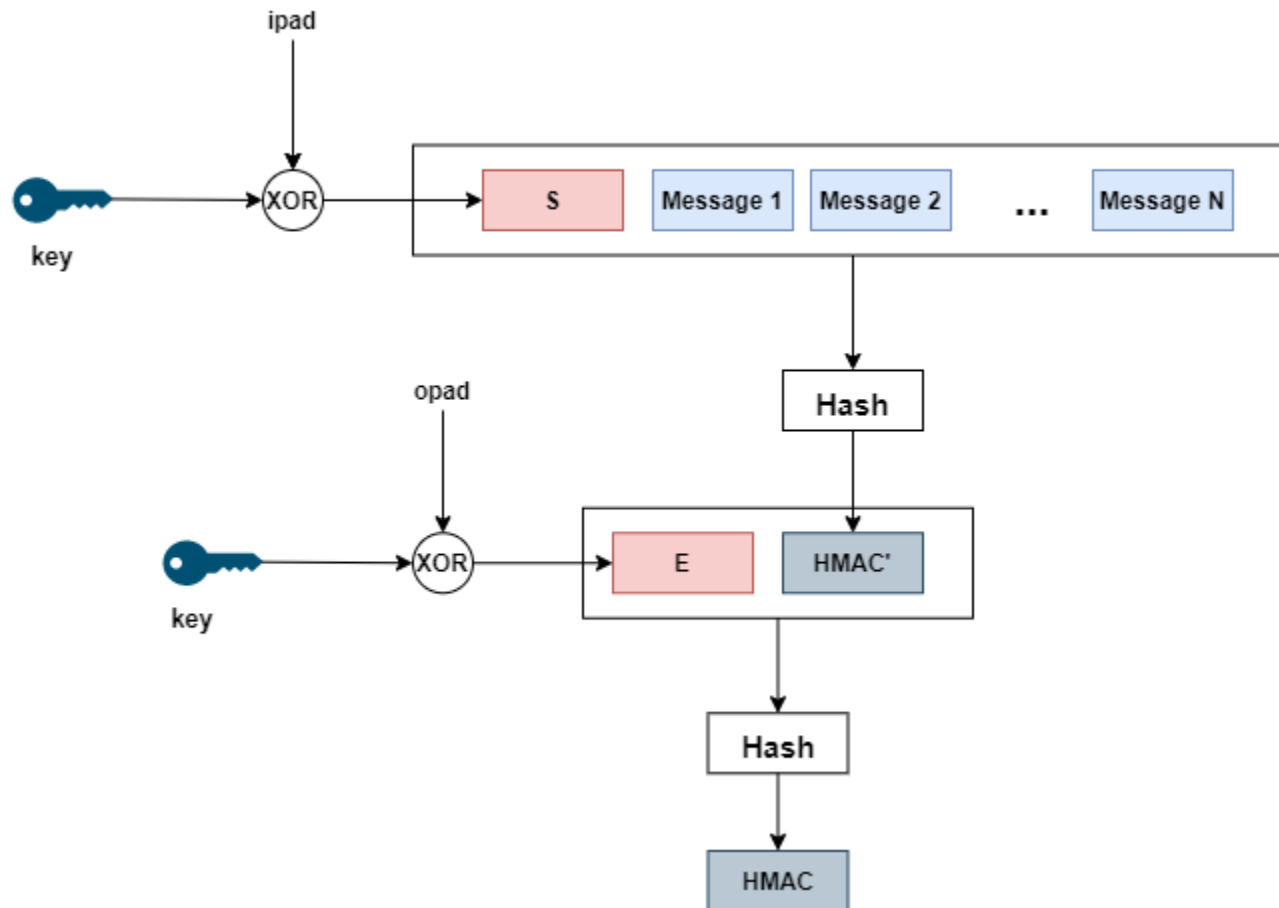
▣ HMAC

- Nested MAC을 기반으로 한 HMAC(Hashed MAC)은 NIST 표준 FIPS198으로 지정되어 사용됨
- HMAC을 구하는 과정은 Nested MAC과 유사하나 키와 해시 함수가 1개만 쓰이고 패딩과 같은 추가적인 연산을 필요로 한다는 차이가 있음

Cryptography - hash function

□ Message Authentication Code (MAC)

▣ HMAC



Cryptography - hash function

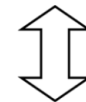
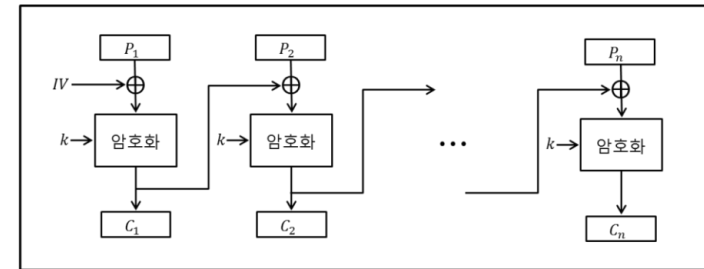
□ Message Authentication Code (MAC)

▣ CBC-MAC (Cipher Block Chaining MAC)

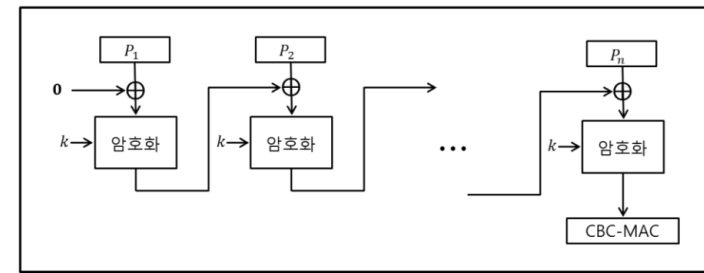
- CBC-MAC은 블록암호의 CBC 모드를 사용하여 메시지 인증 코드를 생성함

- 실제 환경에서 편리하게 적용될 수 있음
- 메시지 인증 코드 생성을 위한 추가적인 작업 없이 CBC 모드 암호화를 이용하여 그대로 메시지 인증 코드를 생성할 수 있음
- 고정된 초기벡터(IV=0)를 사용
- 고정된 메시지에 대한 MAC 추출

CBC모드 암호화



CBC-MAC

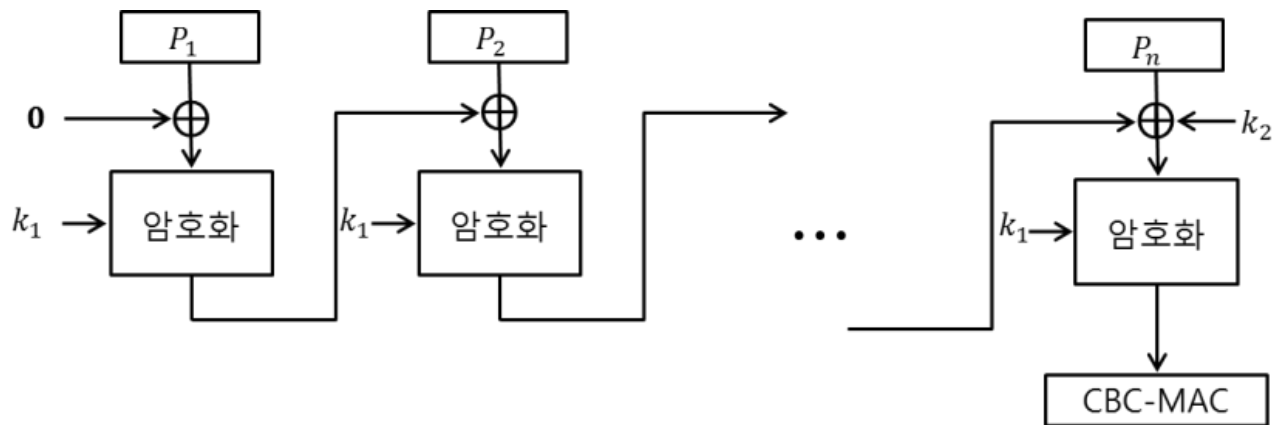


Cryptography - hash function

□ Message Authentication Code (MAC)

▣ CMAC (Cipher-based MAC)

- 고정된 길이의 메시지만을 사용해야 하는 CBC-MAC에서의 제약 사항을 제거함
 - 2개의 키 k_1, k_2 사용
 - 두 메시지 M' 과 M'' 의 메시지 인증 코드 값이 같아지도록 하기 위해서는 k_2 에 대한 값을 알고 있어야만 함



Cryptography - hash function

□ Message Authentication Code (MAC)

▣ 반복 구조의 해쉬 함수(Iterated Hash Function)

– Merkle-Damgard 구조: MD5, SHA-1, SHA-2 등

■ 취약점 발견 → SHA-3은 다른 구조를 채택

▣ 계산 순서

1. n 비트의 배수가 되도록 ($m \parallel$ 패딩)

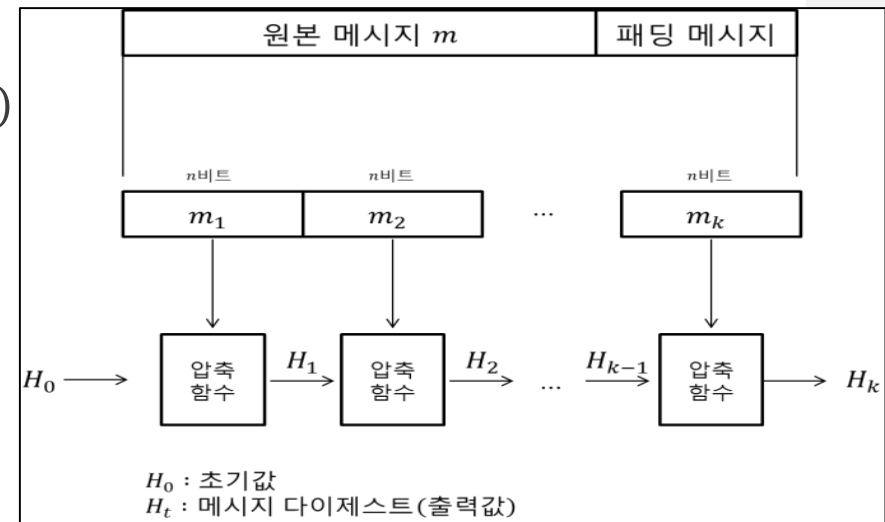
2. ($m \parallel$ 패딩)을 한 블록이 n 비트가 되도록 분할 m_1, m_2, \dots, m_k

3. $f(H_0, m_1) = H_1$

■ 초기 값 H_0 , 압축 함수 $f(\cdot)$

4. k 번 수행하여 H_k 를 생성

→ $h(m) = H_k$



Cryptography - hash function

□ 해시 알고리즘 종류


- ▣ MD5: 1991. Rivest개발, 128비트 알고리즘
- ▣ SHA-1: 1995. 미국 NIST표준 채택, 160비트 알고리즘
- ▣ RIPEMD160: 1996. Han Dobbertin개발, 160비트 알고리즘
- ▣ SHA-256, 384, 512: 2002. 미국 NIST 표준 채택, 256, 384, 512 비트 알고리즘
- ▣ MD5, SHA-1은 충돌 저항성이 낮음
- ▣ 현재에는 SHA-256이 널리 사용됨

Cryptography - hash function

□ 해시 함수 사용 케이스



Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community

FILE	URL	SEARCH
		
<input type="text" value="URL, IP address, domain, or file hash"/>		

<https://support.virustotal.com/hc/en-us/articles/115002739245-Searching>

Cryptography - hash function

▣ 해시 함수 사용 케이스

Download Kali Linux Images

We generate fresh Kali Linux image files every few months, which we make available for download. This page provides the links to **download Kali Linux** in its latest official release. For a release history, check our [Kali Linux Releases](#) page. Please note: You can find unofficial, untested weekly releases at <http://cdimage.kali.org/kali-weekly/>.

Image Name	Download	Size	Version	sha256sum
Kali Linux 64 Bit	HTTP Torrent	2.8G	2018.2	56f677e2edfb2efcd0b08662ddde824e254c3d53567ebbbcd9b5c03efd9bc0f
Kali Linux Light 64 Bit	HTTP Torrent	865M	2018.2	554f020b0c89d5978928d31b8635a7eaddf0a3900abcacdbc39616f80d247f86
Kali Linux E17 64 Bit	HTTP Torrent	2.6G	2018.2	be0a858c4a1862eb5d7b8875852e7d38ef852c335c3c23852a8b08807b4c3be8
Kali Linux Lxde 64 Bit	HTTP Torrent	2.6G	2018.2	449ecca86b0f49a52f95a51acdde94745821020b7fc0bd2129628c56bc2d145d
Kali Linux Xfce 64 Bit	HTTP Torrent	2.6G	2018.2	0e94035a0a56fccc49961b0da56b9243ed3da6a3f8d696884e6f0b936f74dbfb

Cryptography - hash function

□ 해시 함수 사용 케이스

```
명령 프롬프트
C:\Users\Kwak\hash test>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 8A20-40A2

C:\Users\Kwak\hash test 디렉터리

2021-08-19 오후 07:02 <DIR>          .
2021-08-19 오후 07:02 <DIR>          ..
2019-12-07 오후 06:09                27,648 calc.exe
2021-06-25 오후 02:43        651,468,992 VMware-workstation-full-16.1.2-17966106.exe
                2개 파일          651,496,640 바이트
                2개 디렉터리 494,360,707,072 바이트 남음

C:\Users\Kwak\hash test>
C:\Users\Kwak\hash test>
C:\Users\Kwak\hash test>certutil -hashfile VMware-workstation-full-16.1.2-17966106.exe sha256
SHA256의 VMware-workstation-full-16.1.2-17966106.exe 해시:
71b44f2fcfde663195b833ba19f2f70d9ed21a78f9bce35cf13c7f780418a336
CertUtil: -hashfile 명령이 성공적으로 완료되었습니다.

C:\Users\Kwak\hash test>
C:\Users\Kwak\hash test>
C:\Users\Kwak\hash test>certutil -hashfile calc.exe sha256
SHA256의 calc.exe 해시:
58189cbd4e6dc0c7d8e66b6a6f75652fc9f4afc7ce0eba7d67d8c3feb0d5381f
CertUtil: -hashfile 명령이 성공적으로 완료되었습니다.

C:\Users\Kwak\hash test>
```

□ 윈도우 cmd

- > certutil -hashfile 파일명 해시알고리즘

- MD2, MD4, MD5, SHA1, SHA256, SHA384, SHA512

CONTENTS

- ❑ Digital signature

Cryptography - digital signature

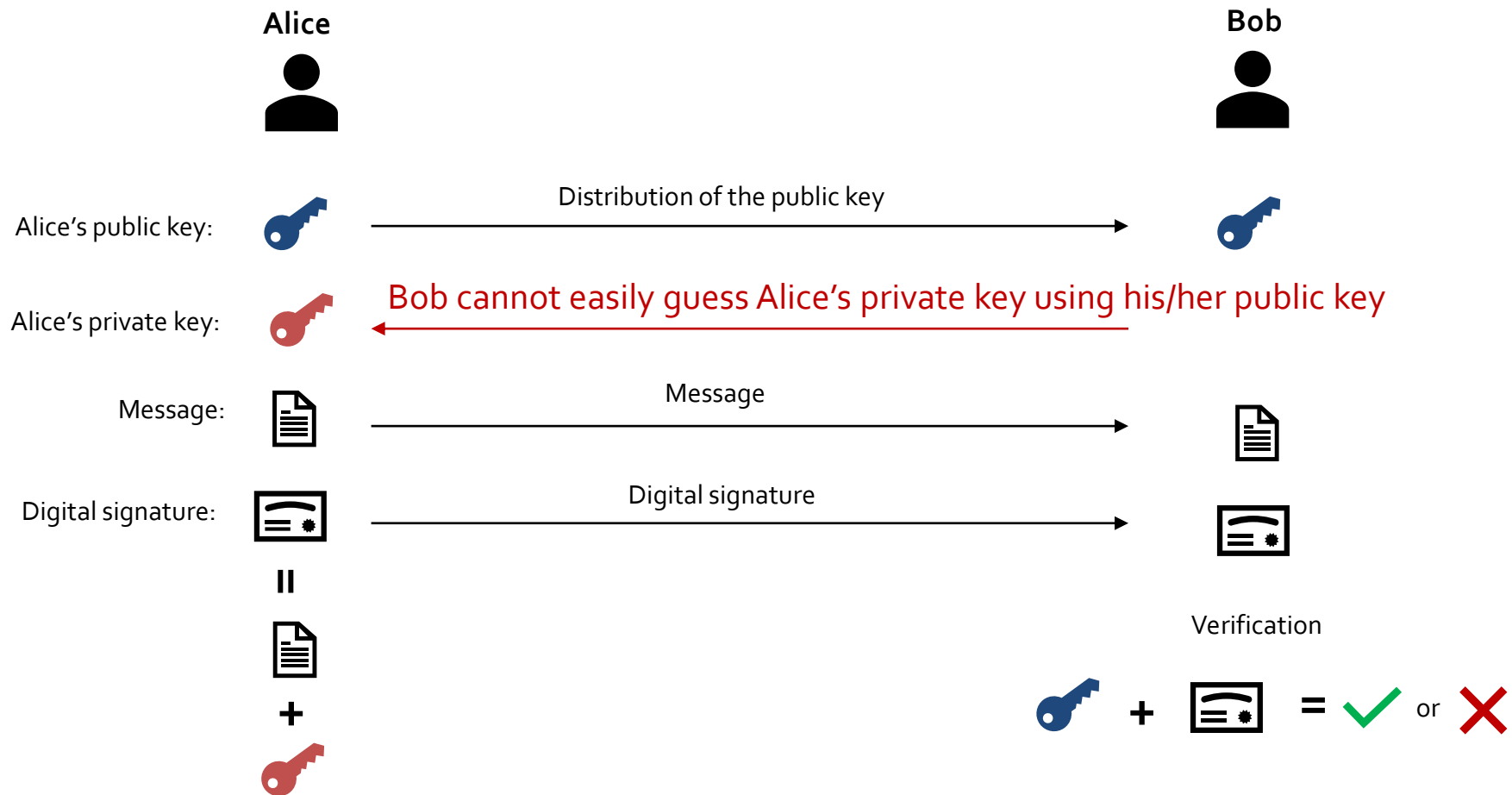
- 전자 서명 (Digital signature)
 - ▣ 인증 (Authentication)
 - 전자 서명을 통해 서명자를 검증할 수 있음
 - ▣ 메시지 무결성 (Integrity)
 - 메시지는 서명 후에 변경될 수 없음
 - ▣ 부인방지 (Non-repudiation)
 - 발신자는 서명한 사실을 부인할 수 없음
 - ▣ 전자 서명 알고리즘의 종류
 - RSA, ECDSA, ...

RSA: MIT's Rivest, Shamir and Adleman

ECDSA: Elliptic Curve Digital Signature Algorithm

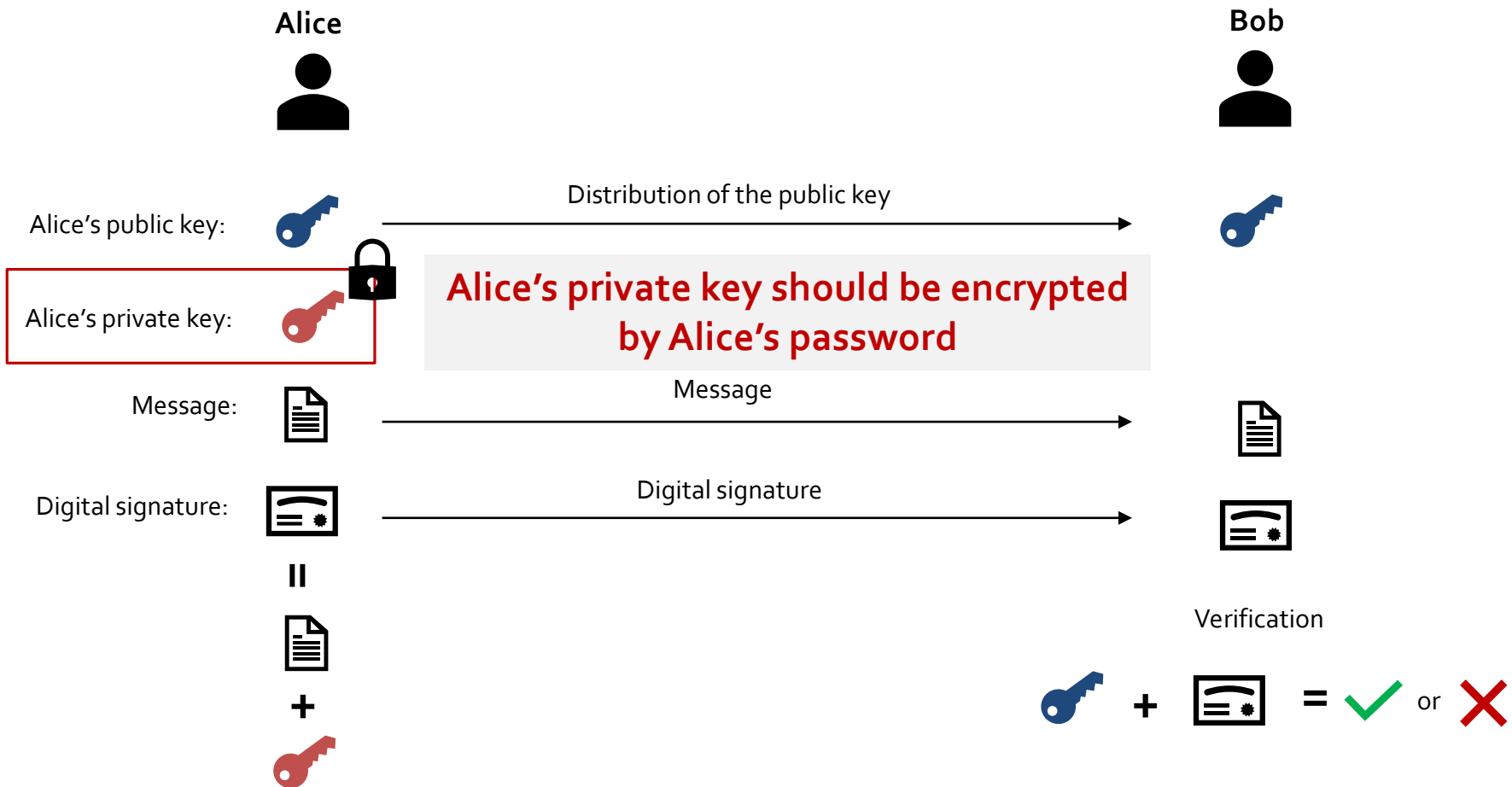
Cryptography - digital signature

□ 전자 서명 (Digital signature) 기본 개념 및 흐름



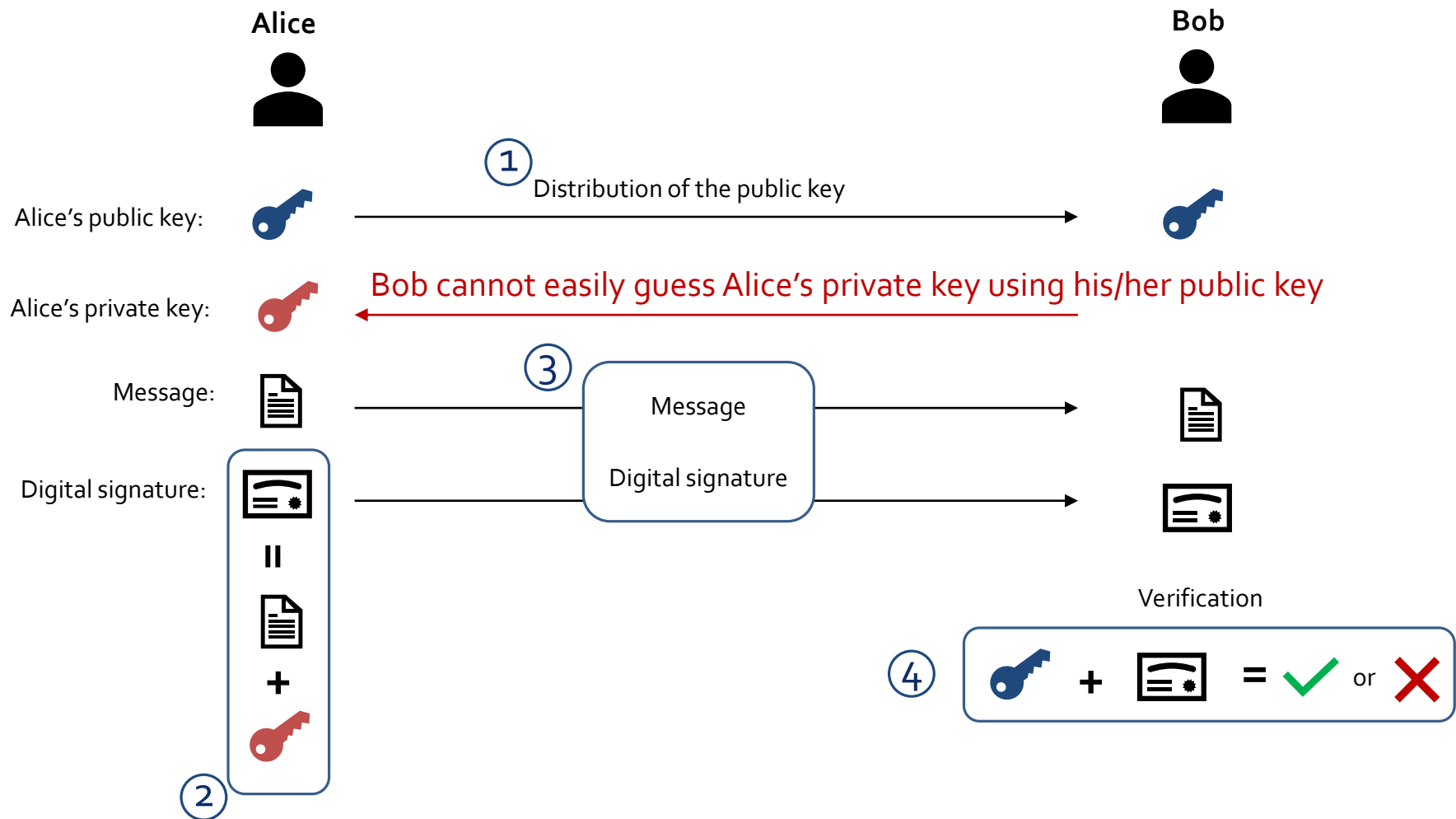
Cryptography - digital signature

□ 전자 서명 (Digital signature) 기본 개념 및 흐름



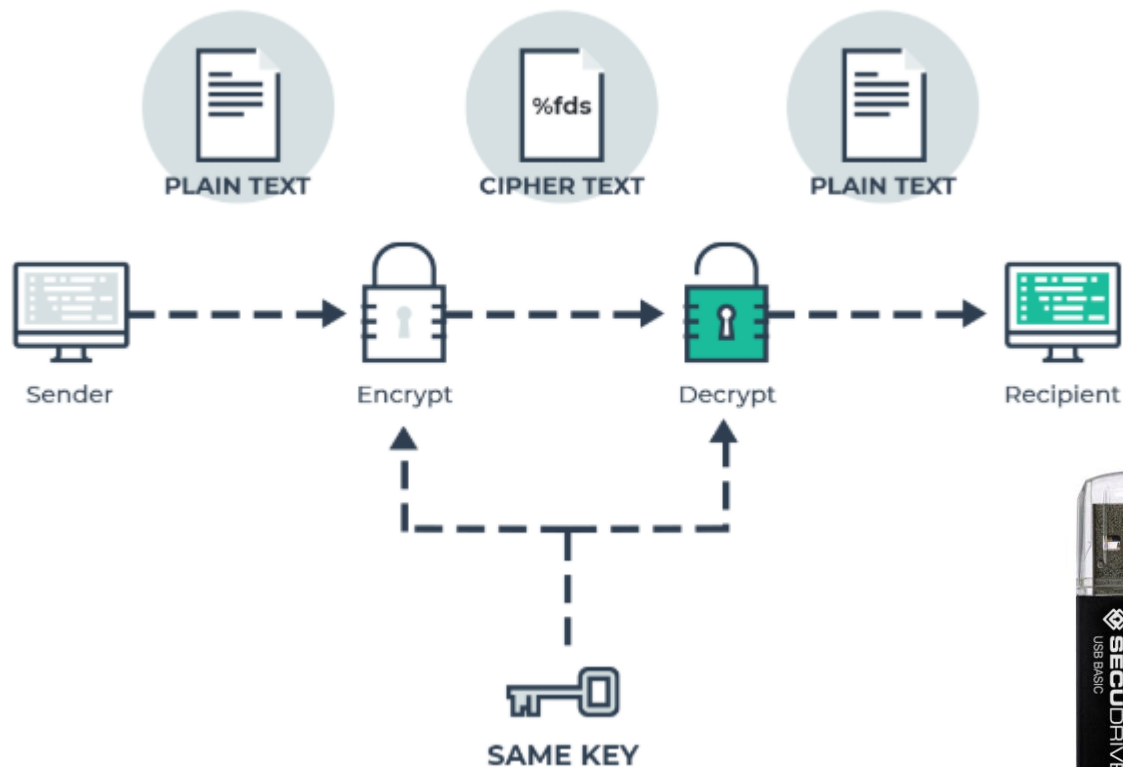
Cryptography - digital signature

□ 전자 서명 (Digital signature) 기본 개념 및 흐름



Appendix. Encryption

□ (Symmetric) Encryption (e.g., AES)



[Source: <https://pixelprivacy.com/resources/what-is-encryption/>]



SECUDRIVE
USB BASIC

SECUDRIVE USB BASIC SD300

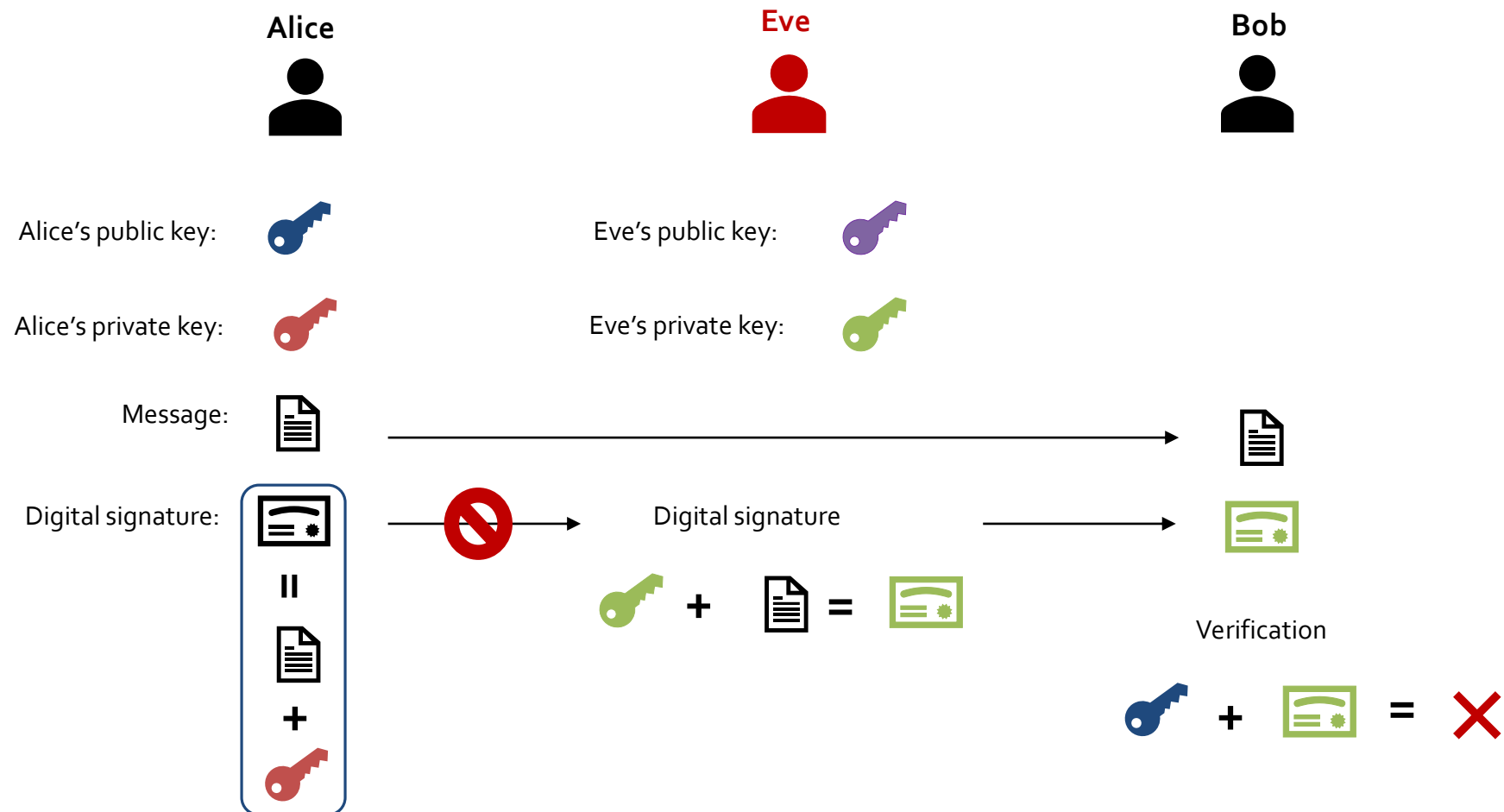
256비트 AES 암호화 칩을 탑재한
USB 3.0 보안 USB

4GB **8GB** 16GB / 32GB / 64GB

AES: Advanced Encryption Standard

Cryptography - digital signature

□ 전자 서명 (Digital signature) 기본 개념 및 흐름



정보보호의 이해

- 암호학적 해시 함수 (Hash function)
- 메시지 인증 (Message Authentication)

참고문헌

- ▣ 정보 보안 개론 - 한권으로 배우는 핵심 보안 이론, 양대일, 한빛 아카데미

Q & A

