18 Feb 2025

Cryptographic Hash functions:

Main security properties:

① **Preimage resistance:**

Given a random y, it should be computationaly hard to find an x s.t.

$$h(x) = y$$

② **Second Preimage Resistance**

Given x, it should be comp. hard to find an x' s.t.

$$x \neq x', \quad h(x) = h(x')$$

③ **Collision resistance:**

find x, x' s.t. $x \neq x'$,

$$h(x) = h(x')$$

Generic attacks:

① .. 2^n

② .. 2^n } calls to $h(\cdot)$

③ .. $2^{n/2}$ → "Birthday Paradox"

In reality, hash functions are "swiss army knife"

e.g. **near collision resistance:**

it should be hard to find x, x' s.t. $x \neq x'$ $(h(x) \oplus h(x'))$ has low Hamming weight

$x = 10100$

$y = 11010$

$\cdots$

$hw(x, y) = 3$

Syntax: $h(\cdot) : \{0,1\}^* \rightarrow \{0,1\}^n$

- produces an n-bit digest of an arbitrary length string
- output looks "random" & "unpredictable"

Random Oracle

even if you have made q queries to $h(\cdot)$

Queries
$x_1 \rightarrow y_1$
$x_2 \rightarrow y_2$
$\vdots$
$x_q \rightarrow y_q$

$\Pr[h(x^*) = y^*]$ ← any random y^*
where x^* has not been queried
$= 1/2^n$

the only way to know the response on a query is to compute $h(\cdot)$ on that query
→ no "shortcuts" are possible

How do we design such hash functions?

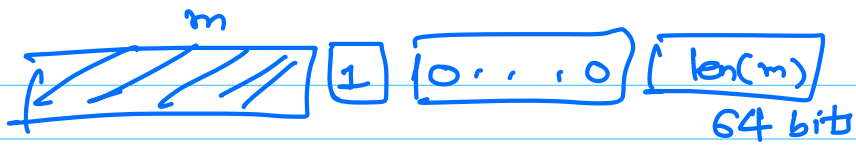
First problem: How to handle arbitrary length messages?

(i) Ensure by pre-processing that the input is multiple of some block length

→ $\text{padding}(m) = 10^* \,||\, \text{length}(m)$ in bits

↑ how many bits?

typically. block size = 512 bits



this padding permits only 64 bits for the length of the message.

⇒ the largest no which can be written in 64 bits $= (2^{64}-1)$

We can handle messages of length $(2^{64}-1)$ bits or less.

→ Not a practical restriction

Merkle-Damgard (MD) padding scheme

Bad example of designing a hash function:

padded message: $m =$ m_1 m_2 m_3 m_4 . . m_L

$h(m) = m_1 \oplus m_2 \oplus m_3 \oplus \cdots \oplus m_L$

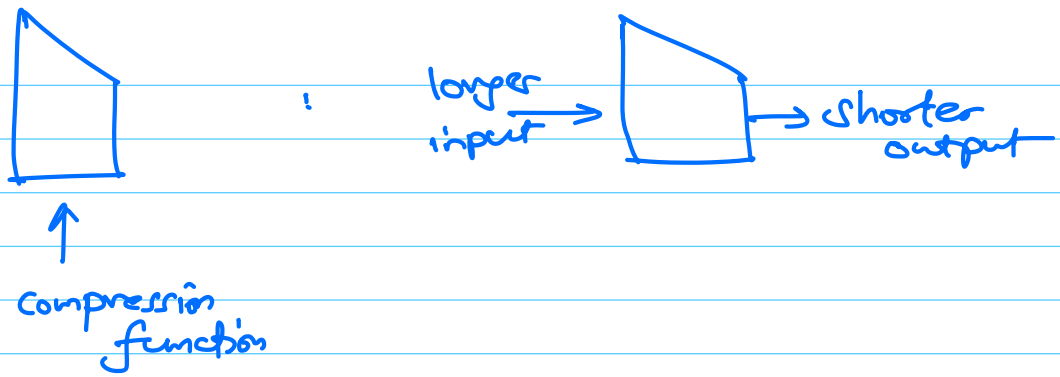
Preimage attack: output $= y$

input $= y \| x \| x \| z \| z \| \cdots$

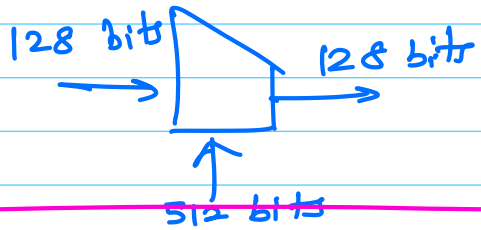
output $= y$ ...

input $= y$

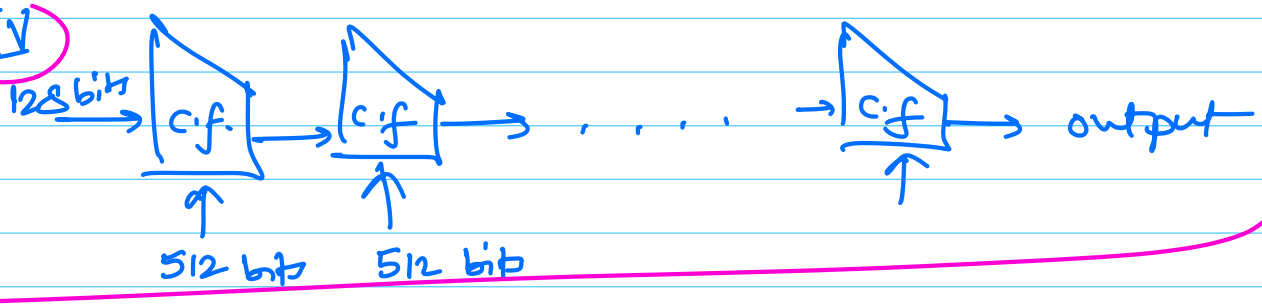
MD Design:



Example:



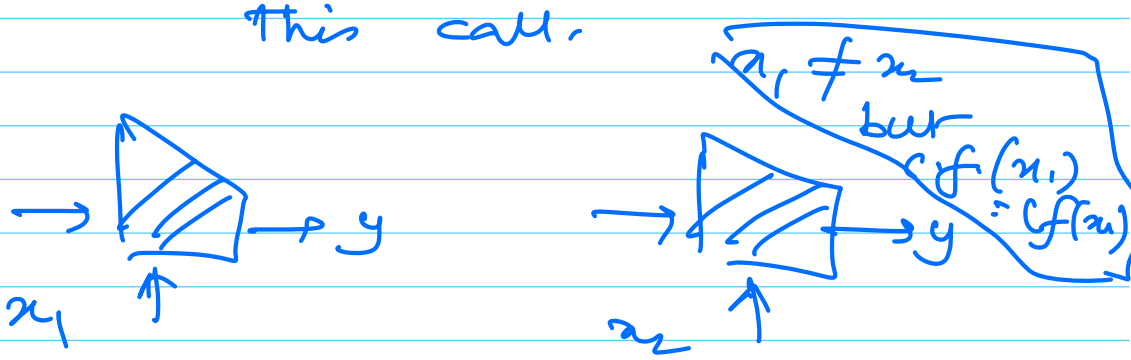
IV



MD theorem:

If $h(m_1) = h(m_2)$ for two different inputs m_1 & m_2

then there must be a collision for compression function somewhere in this call.



if compression function is collision resistant
then the hash function is collision resistant

Example functions

			Output size	No. of rounds
Rivest →	MD4	–	128	48
" →	MD5	–	128	64
NIST older	SHA-0	–	160	80
new	SHA-1	–	160	80
SHA-2	SHA-256		256	64
	SHA-512		512	80

Secure Hash Algorithm = SHA

SHA-1 collisions – Marc Stevens.

SHA-3 competition:

Keccak – Bertoni, ... Daemen

Sponge structure

As of today NIST → SHA-3

Proof of work in Bitcoin

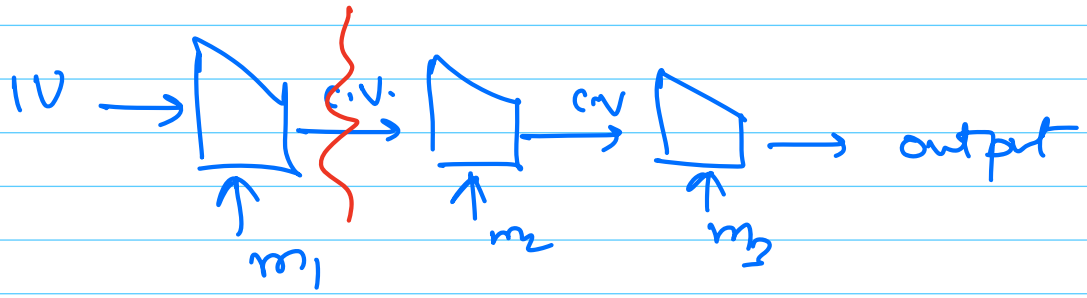
find an x s.t $h(x) = 0\ldots0$ ☐

What about IV in MD Design

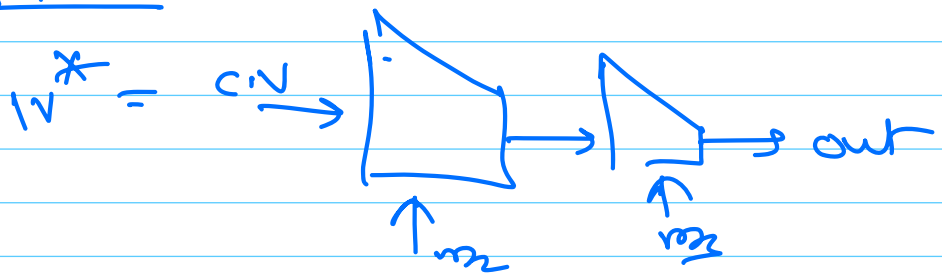
What if allow IV to be chosen like in block ciphers?

$h(m) =$ IV || val.

✗ Bad



Attack:



Solution:

Do not allow IV to be chosen

make IV to be fixed.