Hashing

(Cryptography)

Hashing

- a cryptographic **hash function** is a mathematical algorithm that maps data of arbitrary size (message) to a bit array of a fixed size (message digest or hash)
- all operations are public – such as the **h(x)** hash-function
- there are no private keys here
- it is **deterministic** and random (pseudo-random)

Hashing

fixed length
*string (**d** bits)*

$$h: \{0,1\}^* \rightarrow \{0,1\}^d$$

ASSOCIATIVE ARRAYS
NEED O(1) RUNING TIME
BUT HERE IT IS NOT THE CASE

*data of **arbitrary size***
(string with arbitrary length)

Hashing

fixed length
*string (**d** bits)*

***MD5** – **d** is **128** bits*
***SHA1** – **d** is **160** bits*
***SHA2** – **d** is **256** bits*

$$h: \{0,1\}^* \rightarrow \{0,1\}^d$$

ASSOCIATIVE ARRAYS
NEED O(1) RUNING TIME
BUT HERE IT IS NOT THE CASE

*data of **arbitrary size***
(string with arbitrary length)

Hashing

- a cryptographic **hash function** is a mathematical algorithm that maps data of arbitrary size (message) to a bit array of a fixed size (message digest or hash)
- all operations are public – such as the **h(x)** hash-function
- there are no private keys here
- it is **deterministic** and random (pseudo-random)

Properties of Hashing

– PRE-IMAGE RESISTANCE –

it is exponentially hard (computationally infeasible) to determine the **m** message from the **h(m)** message digest

Properties of Hashing

– PRE-IMAGE RESISTANCE –

it is exponentially hard (computationally infeasible) to determine the **m** message from the **h(m)** message digest

– SECOND PRE-IMAGE RESISTANCE –

if $\mathbf{m_1}$ is given then it is infeasible to find $\mathbf{m_2}$ such that $\mathbf{h(m_1) = h(m_2)}$ so the hashes are the same

Properties of Hashing

– PRE-IMAGE RESISTANCE –
it is exponentially hard (computationally infeasible) to determine the **m** message from the **h(m)** message digest

– SECOND PRE-IMAGE RESISTANCE –
if $\mathbf{m_1}$ is given then it is infeasible to find $\mathbf{m_2}$ such that $\mathbf{h(m_1) = h(m_2)}$ so the hashes are the same

– COLLISION RESISTANCE –
it is infeasible to find any $\mathbf{m_1}$ and $\mathbf{m_2}$ messages such that $\mathbf{h(m_1) = h(m_2)}$. It is easier to break collision resistance than the second pre-image resistance.

Properties of Hashing

1.) ***deterministic***: it means that if we apply to same hash-function (**SHA256**) on
the exact same input then the output must be the same

2.) ***one-way***: it is easy to generate the hash with the given hashing algorithm but
on the other hand it is extremely hard (time-consuming) to restore the original input
~ it is like a trap-door function

3.) ***collision-free***: there are no collisions in **SHA256** (ok there are but with extremely low probability)
It means that no two different inputs share the same output hash
~ and this is good: we want to make these hashes unique, this
is how we identify a block in the blockchain

4.) ***avalanche effect***: a little change in the input results in a completely different output hash
~ otherwise a cryptoanalyst can make predictions about the input
based on the output exclusively

Breaking Second Pre-Image Resistance

If $\mathbf{m_1}$ is given then we have to find $\mathbf{m_2}$ such that $\mathbf{h(m_1) = h(m_2)}$
so the hashes are the same

→ we can use brute-force approach: pick a $\mathbf{m_2}$ message and hash it then compare it with $\mathbf{h(m_1)}$

→ what is the **running time** of this approach?

Best-case scenario: we find $\mathbf{m_2}$ in the first iteration

Worst-case scenario: there are $\mathbf{2^{128}}$ possible messages all together and we have to try them all

Average-case scenario: $\mathbf{2^{128} / 2 = 2^{127}}$