**HASH FUNCTION**

**Introduction**

A hash function is a function which take a message as input and produce an output called: value. For example: An hash function *h* will map *t-*bit-strings of arbitrary finite length to *n* bits’ strings of fixed length. By consequence, there exists some collisions (inputs with identical outputs – probability of 2-n). A cryptographic hash function is basically a hash-value which is a representative image (digital fingerprint) of an input string and can be used as if it were uniquely identifiable with that string. The Hash Function is used for data integrity with the help of digital signature and must follows those properties: compression, ease of computation.

**Classes**

Hash functions may be split in **two classes**: **Unkeyed** hash functions (just a single input parameter – message – **MDC**) and **Keyed** hash functions (two input - message and secret key - **MAC**).

**MDC (modification detection codes) [OneWayHashFunction-OWHF] [CollisionResistantHashFunction-CRHF]:** provide a hash of message with additional mechanisms to facilitate integrity assurances – one-way hash functions (difficult to find the input) – collision resistant (finding two input with same output is difficult).

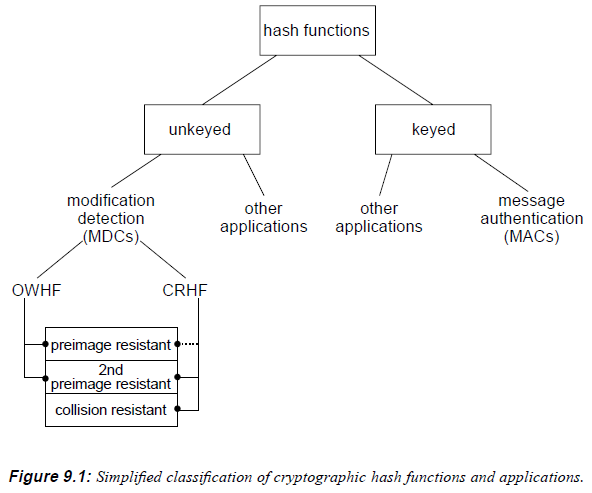
**MAC (message authentication code)**: provide assurances to the message and the integrity without any additional mechanisms.

**Properties**

**Properties** of a cryptographic hash function:

1. Preimage resistance: It is impossible to find the preimage of *x* such that *h(x) = y*.
2. 2nd-preimage resistance: It is impossible to find a second input with the same output, i.e. *x’ ≠ x* such that *h(x) = h(x’).*
3. Collision resistance: It is impossible to find two distinct inputs with the same output.

The use of non-cryptographic hash functions or cryptographic hash function is a question of needs. In a cryptographic hash function, we will provide a way to guarantee the security properties, it would be hard to find collisions or preimage and the output would be random. While in a non-cryptographic hash function, we will try to avoid collisions from malicious input, so there are less guarantees but it is much faster.



**MAC’s properties**

A message authentication code [MAC] algorithm is a family of functions  parameterized by a secret key *k*, with the following properties:

1. **Ease of computation** – For a known function , given a value *k* and an input *x*,  is easy to compute.
2. **Compression** -  maps an input *x* of arbitrary finite bit length to an output  of fixed bit length *n*.
3. **Computation-Resistance** – Given zero or more text-MAC pairs (), it is computationally infeasible to compute any text-MAC pair () for any new input .

If computation-resistance does not hold, a MAC algorithm is subject to MAC forgery. While computation-resistance implies the property of key non-recovery, key-recovery does not imply computation-resistance.

**Attacks on MAC and MDC**

The objective of an adversary who wishes to “attack” an MDC is:

1. **Attack OWHF** – Given a hash value , find a preimage  such that ;or given one such pair ,find a second preimagesuch that .
2. **Attack CRHF** – Find any two inputs , such that . (a CRHF must be designed to withstand standard birthday attacks)

The objective of an adversary for a MAC is:

1. **How to attack** - Without prior knowledge of a key *k*, compute a new text-MAC pair () for some text , given one or more pairs .

Computation-resistance here should hold whether the texts for which matching MACs are available are given to the adversary, ore may be chosen by the adversary. But the following attack scenarios exist for MACs for increasing the advantages of the adversary:

1. **Known-text attack** – One or more text-MAC pairs () are available.
2. **Chosen-text attack** – One or more text-MAC pairs () are available for chosen by the adversary.
3. **Adaptive chosen-text attack** – The may be chosen by the adversary as above, but now allowing successive choices based on the results of prior queries.

**Types of forgery (selective, existential)**

If MAC forgery is possible (if the MACs algorithm has been defeated), the consequences depends on the degree of control of an adversary over the value *x* for which a MAC may be forged, this is differentiated by types of forgery:

1. **Selective forgery [most damaging attack]** – Attacks whereby an adversary is able to produce a new text-MAC pair for a text of his choice.
2. **Existential forgery** – Attacks whereby an adversary is able to produce a new text-MAC pair but with no control over the value of that text.