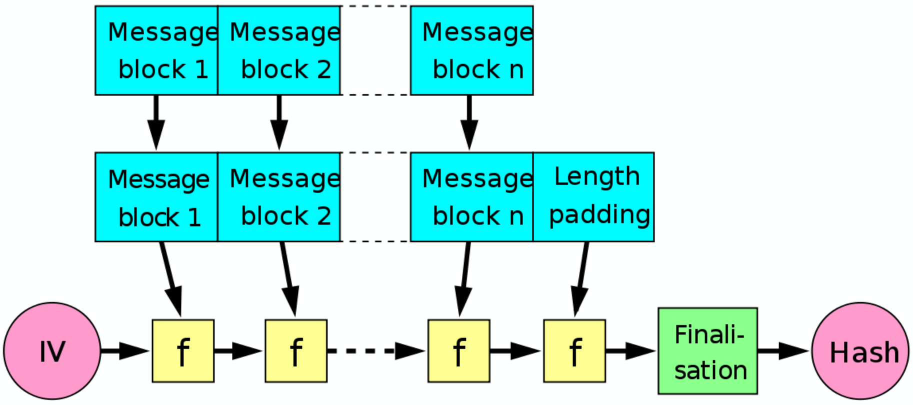
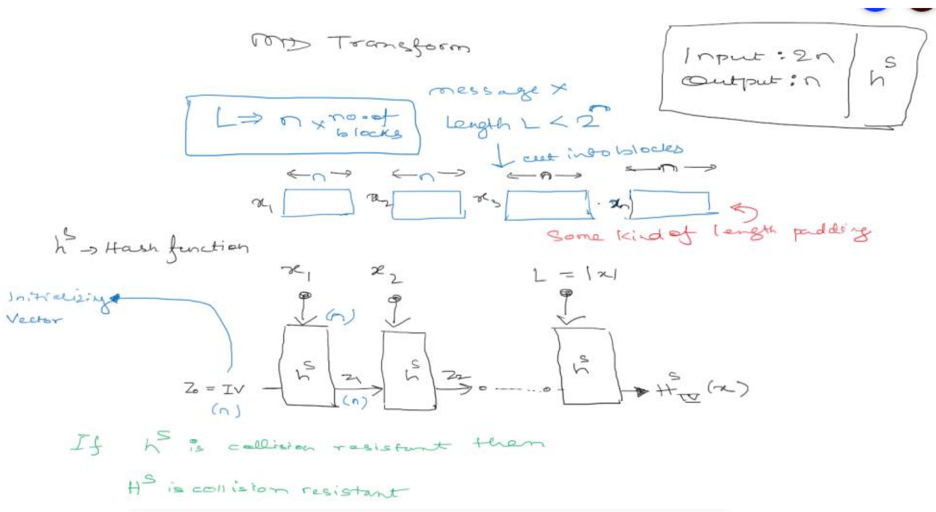
Soumodipta Bose| 2021201086

Using MD Transform to obtain a (provably secure) Collision Resistant Hash Function (Theory)

MD Transform or Merkle Damgard Transform is a methodology that is used to constructing a Hash function for longer inputs using a fixed length hash function.



Sketches/Ideas:



Construction:

Once our Fixed length collision resistant hash function is ready to be used as an api. We can further dive in to just use it and turn it into a Provably secure Collision resistant Hash function.

We take two inputs to merkle damgard, an initialization vector and another the message whose hash is to be found.

Let l(n)=|iv|

CONSTRUCTION 4.11 The Merkle-Damgård Transform.

Let (Gen_h, h) be a fixed-length hash function with input length $2\ell(n)$ and output length $\ell(n)$. Construct a variable-length hash function (Gen, H) as follows:

- $\mathsf{Gen}(1^n)$: upon input 1^n, run the key-generation algorithm Gen_h of the fixed-length hash function and output the key. That is, output $s \leftarrow \mathsf{Gen}_h$.
- $H^s(x)$: Upon input key s and message $x \in \{0,1\}^*$ of length at most $2^{\ell(n)} - 1$, compute as follows:
  1. Let $L = |x|$ (the length of x) and let $B = \lceil \frac{L}{\ell} \rceil$ (i.e., the number of blocks in x). Pad x with zeroes so that its length is an exact multiple of ℓ.
  2. Define $z_0 := 0^\ell$ and then for every $i = 1, \ldots, B$, compute $z_i := h^s(z_{i-1} \| x_i)$, where h^s is the given fixed-length hash function.
  3. Output $z = H^s(z_B \| L)$

At first the message is augmented with zeros at the end to make it a multiple of the size of the initialization vector. Then we encode the length of the message at the end of the message as a binary string. Now all we have to do is setting up the parameters for our fixed length hash function x1 and x2 in a loop. We will set x2 as our message fragment and x1 is initially our iv and in later stages of propagation becomes the output of our hash function.Both x1 and x2 are of size ln(), so total data feeded to our hash function hash(x1,x2) is of size 2*l(n). Now our gen algorithm as already discussed in previous document generates the group parameters which will then be used along with DLP to generate a hash of 16 bits.After a number of iterations equal to the number of message fragments we will get our final hash.

This version has a limitation because of our fixed length hash function which supports upto 16bits.