Cryptographic Hash Function

Hardik Soni

17 March 2024

§1 Cryptographic Hash Functions

Speaking in Laymen terms, a **Cryptographic Hash Function** takes input of some length and compresses it into short fixed - length digest.

Let's begin with the definition of Cryptographic Hash Function. A cryptographic hash function (*CHF*) is an equation used to verify the validity of data. It has many applications, notably in information security (e.g. user authentication). A *CHF* translates data of various lengths — the message — into a fixed size numerical string — the hash. Without further ado, lets go to the formal definition of a *CHF*:

Definition 1.1 (Cryptographic Hash Functions). A hash function (*with output length l) is a pair of probabilistic polynomial-time algorithms* (***Gen***, H) satisfying the following:

- **(Gen:)** is a *probabilistic algorithm which takes as input a security parameter* 1^n *and outputs a key s. We assume that* 1^n *is implicit in s.*

- **(H:)** H *takes as input a key s and a string* $x \in \{0,1\}^*$ *and outputs a string* $H^s(x) \in \{0,1\}^{l(n)}$ *(where n is the value of the security parameter implicit in s).*

If H^s *is defined only for inputs* $x \in \{0,1\}^{l'(n)}$ *and* $l'(n) > l(n)$, then we say that (**Gen**, H) is a fixed-length hash function for inputs of length l'. In this case, H may be called a compression function.

In the fixed-length case we require that l' be greater than l. This makes sure that the function compresses its input. But in general case it must be able to take as input strings of arbitrary length. Thus, it also compresses (albeit only strings of length greater than $l(n)$). Note that without compression, collision resistance is trivial (since one can just take the identity function $H_s(x) = x$). To define security, we first define an experiment for a hash function $\Pi = (\mathbf{Gen}, H)$, an adversary A, and a security parameter n:

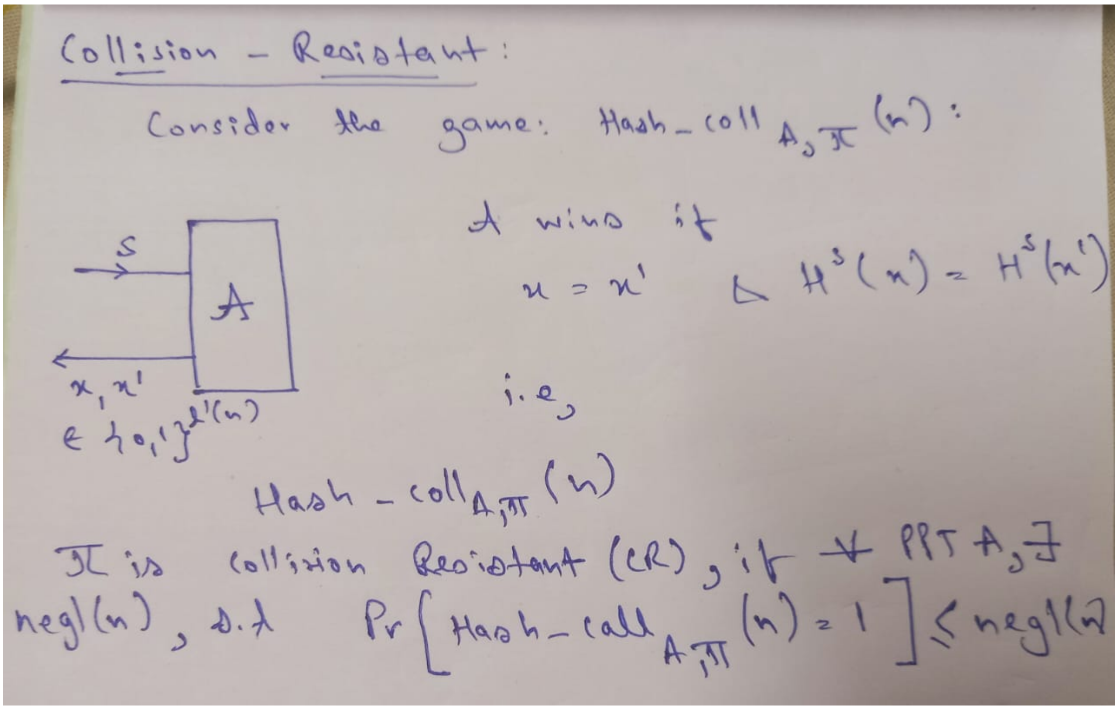
Definition 1.2 (The collision-finding experiment $HASH-COLL_{A,\Pi}(n)$:). The Collition Finding Experiment to determine if a hash function is a collision resistant function or not, is defined as:

1. A key s is generated by running **Gen**(1^n).

2. The adversary A is given s and outputs x, x'. (If Π is a fixed-length hash function for inputs of length $l'(n)$, then we require $x, x' \in \{0,1\}^{l'(n)}$.)

3. The output of the experiment is defined to be 1 if and only if $x \neq x'$ and $H_s(x) = H_s(x')$. In such a case we say that A has found a collision.

The definition of collision resistance states that no efficient adversary can find a collision in the above experiment except with negligible probability.



s
A
x, x'
$\in \{0,1\}^{l'(n)}$

Collision - Resistant :

Consider the game: $\text{Hash\_coll}_{A,\pi}(n)$:

A wins if
$x = x'$ & $H^s(x) = H^s(x')$

i.e,

$\text{Hash-coll}_{A,\pi}(n)$

π is collision Resistant (CR), if $\forall$ PPT A, $\exists$ negl(n), s.t $\Pr[\text{Hash-coll}_{A,\pi}(n) = 1] \leq$ negl(n

Theorem 1.3 (Collision Resistant Hash Function)
A hash function $\Pi = (Gen, H)$ is collision resistant if for all probabilistic polynomial-time adversaries A there is a negligible function **negl** such that:

$$Pr[Hash - coll_{A,\Pi}(n) = 1] \leq \text{negl(n)}$$

§1.1 Unkeyed hash functions:

Cryptographic hash functions used in practice generally have a fixed output length (just as block ciphers have a fixed key length) and are usually unkeyed, meaning that the hash function is just a fixed function $H : \{0,1\}^* \to \{0,1\}^l$. The problem from a theoretical standpoint is that, since for any such function there is always a constant-time algorithm that outputs a collision in H: the algorithm simply outputs a colliding pair (x, x') hardcoded into the algorithm itself.

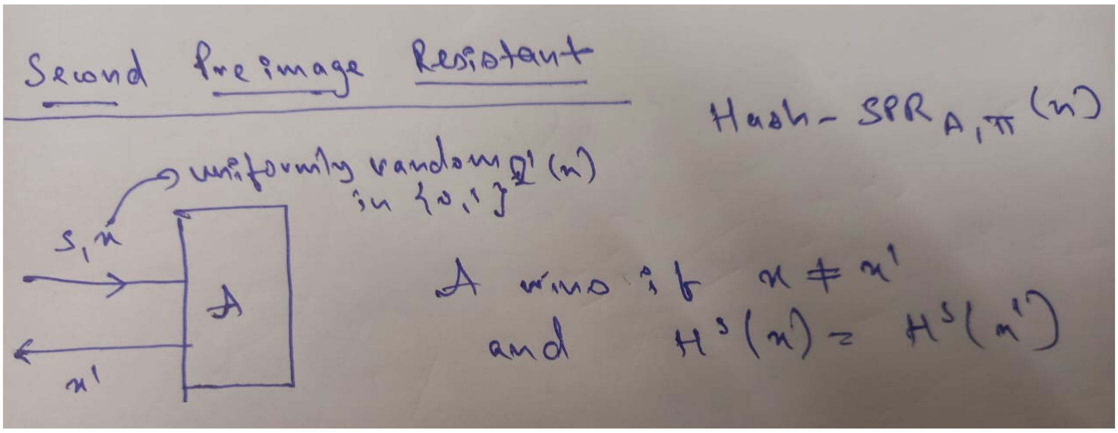
§2 Weaker Notions of Security

In some applications it suffices to rely on security requirements weaker than collision resistance. These include:

- *Second-preimage* or *target-collision resistance*: Informally, a hash function is second preimage resistant if given s and a uniform x it is infeasible for a PPT adversary to find $x' \neq x$ such that $H_s(x') = H_s(x)$.

- *Preimage resistance*: Informally, a hash function is preimage resistant if given s and a uniform y it is infeasible for a PPT adversary to find a value x such that $H_s(x) = y$.

Any hash function that is collision resistant is also second preimage resistant. This holds since if, given a uniform x, an adversary can find $x' \neq x$ for which $H_s(x') = H_s(x)$, then it can clearly find a colliding pair x and x'. Likewise, any hash function that is second preimage resistant is also preimage resistant. This is due to the fact that if it were possible, given y, to find an x such that $H_s(x) = y$, then one could also take a given input x', compute $y := H_s(x')$, and then obtain an x with $H_s(x) = y$. With high probability $x' \neq x$ (relying on the fact that H compresses, and so multiple inputs map to the same output), in which case a second preimage has been found.

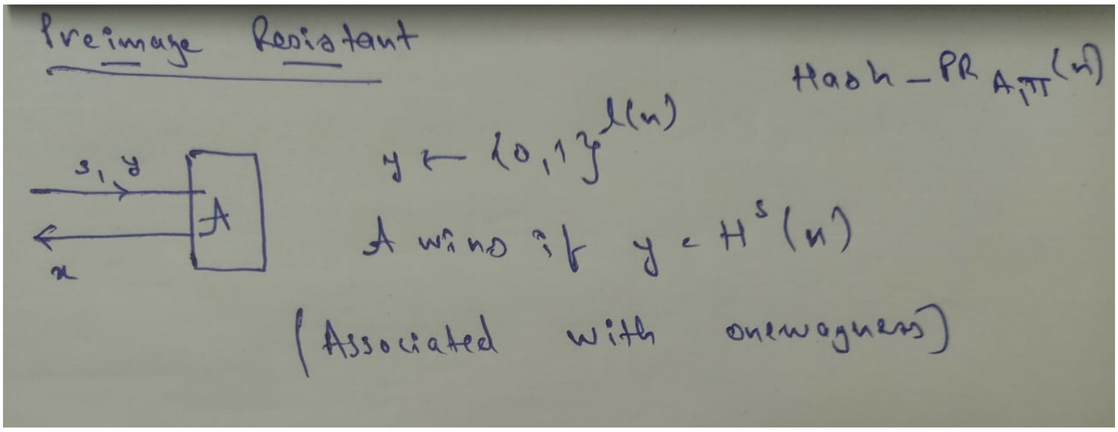


Second Preimage Resistant

$\text{Hash-SPR}_{A,\Pi}(n)$

uniformly random in $\{0,1\}^{\ell'(n)}$

A wins if $x \neq x'$ and $H^s(x) = H^s(x')$



Preimage Resistant

$\text{Hash-PR}_{A,\Pi}(n)$

$y \leftarrow \{0,1\}^{\ell(n)}$

A wins if $y = H^s(x)$

(Associated with onewayness)

§3 Domain Extension: The Merkle-Damgard Transform

Hash functions are often constructed by first designing a collision-resistant compression function handling fixed-length inputs, and then using *domain extension* to handle arbitrary-length inputs. The Merkle–Damgard transform is a common approach for extending a compression function to a full-fledged hash function, while maintaining the collision-resistance property of the former.

Theorem 3.1

The Merkle–Damgard transform Let (Gen, h) be a fixed-length hash function for inputs of length 2n andwith output length n. Construct hash function (Gen, H) as follows:

- **Gen**: remains unchanged.
- **H**: on input a key s and a string $x \in \{0,1\}^*$ of length $L < 2n$, do the following:
  1. 1. Set $B := \lceil \frac{L}{n} \rceil$ (i.e., the number of blocks in x). Pad x with zeros so its length is a multiple of n. Parse the padded result as the sequence of n-bit blocks $x_1, ..., x_B$. Set $x_{B+1} := L$, where L is encoded as an n-bit string.
  2. Set $z_0 := 0^n$. (This is also called the IV .)
  3. For i = 1,...,B + 1, compute $z_i := h^s(z_{i-1}||x_i)$.
  4. Output z_{B+1}.

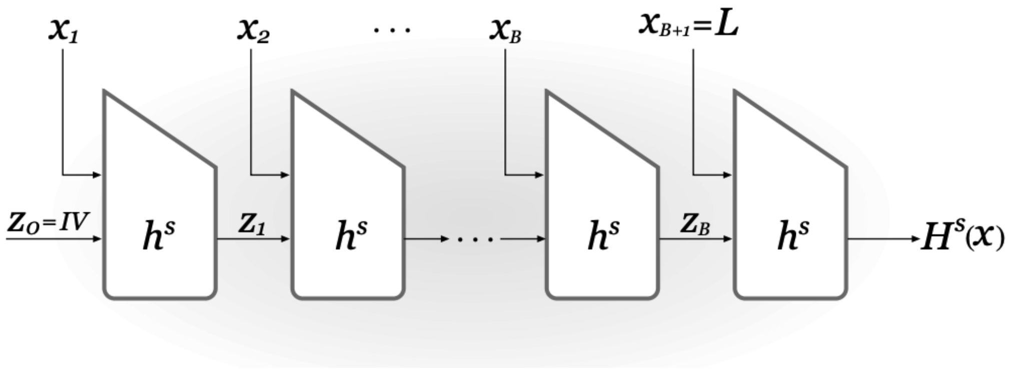


FIGURE 5.1: The Merkle–Damgård transform.

Theorem 3.2

If $\Pi = (\mathbf{Gen}, \mathbf{h})$ is collision resistant, then so is $(\mathbf{Gen}, \mathbf{H})$.

PROOF: We show that for any s, a collision in H^s yields a collision in h^s. Let x and x' be two different strings of length L and L', respectively, such that $H^s(x) = H^s(x')$. Let $x_1, ..., x_B$ be the B blocks of the padded x, and let $x'_1, ..., x'_{B'}$ be the B' blocks of the padded x'. Recall that $x_{B+1} = L$ and $x'_{B'+1} = L'$. There are two cases to consider:

1. *Case 1*: $L \neq L'$. In this case, the last step of the computation of $H^s(x)$ is $z_{B+1} := h^s(z_B||L)$, and the last step of the computation of $H^s(x')$ is $z'_{B'+1} := h^s(z'_{B'}||L')$. Since $H^s(x) = H^s(x')$ it follows that $h^s(z_B||L) = h^s(z'_{B'}||L')$. However, $L \neq L'$ and so $z_B||L$ and $z'_B{}'||L'$ are two different strings that collide under h^s.

2. *Case 2*: $L = L'$. This means that $B = B'$. Let $z_0, ..., z_{B+1}$ be the values defined during the computation of $H^s(x)$, let $I_i \overset{def}{=} z_{i1}||x_i$ denote the $i_t h$ input to h^s, and set $I_{B+2} \overset{def}{=} z_{B+1}$. Define $I'_1, ..., I'_{B+2}$ analogously with respect to x'. Let N be the largest index for which $I_N \neq I'N$. Since $|x| = |x'|$ but $x \neq x'$, there is an i with $x_i \neq x'_i$ and so such an N certainly exists. Because

$$I_{B+2} = z_{B+1} = H^s(x) = H^s(x') = z'_{B+1} = I'_{B+2}$$

, we have $N \leq B + 1$. By maximality of N, we have $I_{N+1} = I'_{N+1}$ and in particular $z_N = z'_N$. But this means that I_N , I'_N are a collision in h^s.