CS 120/ E-177: Introduction to Cryptography

Salil Vadhan and Alon Rosen Dec. 4, 2006

Lecture Notes 18:

Collision-Resistant Hashing

- Recommended reading: Katz-Lindell 10.5

1 Definition

Idea: Sign (or MAC) a long message m by first hashing it. What properties will we want from the hash function h?

- $||h(x)|| \ll ||x||$.
- h easy to evaluate.
- Hard to find *collisions*, i.e. (x, x') s.t. $x \neq x'$ and $h(x) = h(x')$.

Definition 1 (collision-resistant hash functions) *$\mathcal{H} = \{h_i : \{0,1\}^{\ell_d(i)} \to \{0,1\}^{\ell_r(i)}\}_{i \in \mathcal{I}}$ is a collection of* collision-resistant hash functions *if*

- *(generation) There is a PPT $G(1^n)$ which outputs $i \in \mathcal{I}$.*
- *(hashing) $\ell_d(i) < \ell_r(i)$.*
- *(easy to evaluate) Given x, i, can compute $h_i(x)$ in poly-time.*
- *(hard to form collisions) For every PPT A, there is a negligible function ε such that*

  $$\Pr\left[A(I) = (X, X') \text{ s.t. } X \neq X' \text{ and } h_I(X) = h_I(X')\right] \leq \varepsilon(n), \qquad \forall n$$

  where the probability is taken over $I \stackrel{R}{\leftarrow} G(1^n)$ and the coin tosses of A.
- *(technical condition) $n \leq \mathrm{poly}(|i|)$ for any $i \leftarrow G(1^n)$.*

1.1 Comments

- Typically, we want the range to be much smaller than the domain, we can think of $\{0,1\}^{\ell_d(i)} = \{0,1\}^*$, $\{0,1\}^{\ell_r(i)} = \{0,1\}^n$.

1.2 Attacks

There are different attacks on collision-resistant hash functions:

- Random guessing: Suppose $\ell_d = 2n$. pick m, m' randomly from $\{0,1\}^{2n}$. The probability of success is greater than $\frac{1}{2^n} - \frac{1}{2^{2n}}$.

- Birthday attack: pick random messages to find a collision. We choose t messages randomly from $\{0,1\}^{2n}$ and the expected number of collisions is:

$$\begin{aligned}
\mathrm{E}[\#\text{ collisions}] &= \#\text{ pairs} \cdot \Pr[\text{any one pair collide}] \\
&\geq \binom{t}{2} \cdot \left(\frac{1}{2^n} - \frac{1}{2^{2n}}\right) \\
&\sim \frac{t^2}{2^{n+1}}
\end{aligned}$$

  If we pick $t = \Theta(2^{k/2})$, we expect to find a collision. Quadratic savings over exhaustive search (though still exponential in n).

1.3 Universal One-Way Hash Functions

Note that our definition of collision-resistantness is a strong notion. There exists a weaker notion called *universal one-way hash functions*, where:

1. A first picks $x \in \{0,1\}^{\ell_d(i)}$.
2. $i \overset{r}{\leftarrow} G(1^n)$
3. A has to find $x' \neq x$ s.t. $h_i(x') = h_i(x)$.

Universal one-way hash functions can be constructed from one-way functions.

1.4 Shrinking by more than One Bit

The definition of Collision -Resistant Hash Functions only requires shrinking by one bit. To shrink more may apply "Merkle–Damgård" methodology:

- First design a collision-resistant "compression function" $h : \{0,1\}^{\ell+n} \to \{0,1\}^n$.
- For a message $M \in \{0,1\}^*$ (eventually padded appropriately), break into ℓ-bit blocks $M_1 M_2 \cdots M_t$, where M_t contains length of M, and define $H(M) = h(M_t \circ h(M_{t-1} \circ h(M_{t-2} \cdots h(M_1 \circ \mathrm{IV}))))$, where IV is a fixed initial vector (e.g. $\mathrm{IV} = 0^n$).

Proposition 2 *If h is collision-resistant, then H is collision-resistant.*

2 Constructions

2.1 Number-Theoretic Constructions

Theorem 3 *Collections of collision-resistant hash functions exist under either the Factoring Assumption or the Discrete Log Assumption.*

Proof Sketch: Construction based on Discrete Log: First construct $h_{p,g,y} : \mathbb{Z}_{p-1} \times \{0,1\} \to \mathbb{Z}_p^*$ by $h_{p,g,y}(x,b) = y^b \cdot g^x \bmod p$. A collision for $h_{p,g,y}$ yields the discrete log of y. □

2.2 Hash Functions in Practice

Typical design features:

- Tailor-designed functions $H : \{0,1\}^* \to \{0,1\}^n$, with e.g. $n = 128$ or $n = 160$. (Note that n is larger than for block ciphers to protect against birthday attacks)
- Very fast.
- Designed to be collision-resistant (in strong sense), have "random looking" output.
- Confusion & diffusion
- Not related to any nice complexity assumption.
- Not a "collection" — think of "design choices" as generation algorithm.

Some "popular hash-functions:

- MD4 — Message Digest 4
  - Designed by Ron Rivest (1990), $n = 128$, $\ell = 512$.
  - Collisions have been found (1995). Design is basis for stronger hash functions (MD5, SHA-1).
  - Follows Merkle–Damgård with compression function $h : \{0,1\}^{512+128} \to \{0,1\}^{128}$.
- MD5 — improvement to MD4 (Rivest, 1992). Collisions have been found (1998).
- SHA-1 — another improvement to MD4 (NIST w/NSA, 1994)
  - hash size $n = 160$, so compression function is $h : \{0,1\}^{512+160} \to \{0,1\}^{160}$.
  - Collisions can be found in time 2^{60} (better than "birthday attack") (2005).

3 Hash-then-Sign

We present it for signatures, but it also works for MACs. Let (G, S, V) be a signature scheme for message space $\{0,1\}^n$, and let $\mathcal{H}$ be a collection of hash functions with domain $\{0,1\}^*$ and range $\{0,1\}^n$. Define a new signature scheme (G', S', V') for message space $\{0,1\}^*$ by setting

- $PK' = (PK, i)$, $SK' = (SK, i)$.
- $\mathrm{S}'_{SK'}(m) = \mathrm{S}_{SK}(h_i(m))$.
- $V'_{PK'}(m, \sigma) = V_{PK}(h_i(m), \sigma)$.

Theorem 4 *If (G, S, V) is a secure one-time signature scheme for message space $\{0,1\}^n$ and $\mathcal{H}$ is collision resistant, then (G', S', V') is a secure one-time signature scheme for message space $\{0,1\}^*$.*

Proof: Suppose there is a PPT A which breaks the new signature scheme with probability at least ε. $A(PK')$ makes one query m, gets back $\sigma \overset{R}{\leftarrow} \mathrm{S}'_{SK'}(m)$, and outputs (m', σ'). For this to be a successful forgery, $m \neq m'$ and $V'_{PK'}(m', \sigma') = \texttt{accept}$. One of the following two cases must hold.

- $h_i(m) = h_i(m')$ and $m \neq m'$. This means that A has found a collision for h.

  A PPT B that breaks h is given i as input:

- generate (PK, SK) for the original signature shceme
- run $A(PK, i)$ (B can answer A's query because B has SK) to obtain (m', σ').
- B outputs (m, m')

- $h_i(m) \neq h_i(m')$ and $V_{PK}(h_i(m'), \sigma') = \texttt{accept}$. This means that A has forged in the original signature scheme.

  A PPT C that breaks the original signature scheme is given PK as input:

  - B picks i at random and runs $A(PK, i)$. Note that B can ask for one query in the original scheme so B will ask for the signature of $h_i(m)$, where m is A's query.
  - A produces a forgery (m', σ')
  - B outputs the forgery $(h_i(m'), \sigma')$.

Each of the above happens with only negligible probability, by reducibility arguments. ■

Hash-then-sign also works for general (i.e. many-time) signatures and MACs.