Lecture 12: Hash Functions

Instructor: Omkant Pandey

Spring 2017 (CSE 594)
Last class

- Construct MAC using a PRF
- Today: compressing long messages into short ones
- Scribe notes volunteer?
Recall from algorithms/data-structures

- Hash tables?
- Idea: store a small number of elements coming from a large set.
- example: store $m = n^2$ values where each value is a string of length $n$.
- total strings to be stored are few in comparison to the full set of $2^n$ elements
- Want: deterministic method to quickly store and “look-up” elements $\Rightarrow$ get “look up” key from value/message.
- Want: low collisions (otherwise, useless)
Recall from algorithms/data-structures

- Use a hash function: \( \forall \text{ distinct } x, y: \)

\[
\Pr_{h \in \mathcal{H}} [h(x) = h(y)] \leq \frac{1}{|\mathcal{R}|}
\]

where \( h : \mathcal{D} \to \mathcal{R} \)

- These are actually universal hash functions, and \( \mathcal{H} \) a “family” of universal hash functions.

- Great for many applications such as data structures, randomness extraction, etc.

- Not always good enough for cryptographic purposes

- An adversary may easily find collisions given \( h \)
Cryptographic Hash Functions

- Want: $h$ should compress (say to half length)
- Want: given $h$, hard to find “collisions” $(x, y)$ s.t. $h(x) = h(y)$ (collision resistance)
- Want: given $(h, y)$, hard to find $x$ s.t. $h(x) = y$ (target collision resistance, “one-way”)
- Want: given $(h, x)$, hard to find $x'$ s.t. $h(x) = h(x')$ (second pre-image resistance)
- Today: focus only on “collision resistance”
Collision Resistant Hashing

- Compress large strings to short “message digests” s.t. hard to find collisions.
- Many uses:
  - Check if you received the same file over the network
  - Like an “error-detecting code” but much shorter
  - Version control and consistency
  - Many cryptographic applications
- How to define formally? Want: function $h$ such that:
  - $h$ is deterministic and efficiently computable
  - output length is shorter than input length, e.g., half (opposite of a PRG which stretches)
  - hard to find collisions: $(x_1, x_2)$ s.t. $h(x_1) = h(x_2)$ but $x_1 \neq x_2$.
- $h$ is called a collision resistant hash function (CRHF).
Collision Resistant Hashing

• Problem 1: if $|h(x)| < |x|$ for all $x$, then $h$ must have collisions!
  I.e., $\exists x_1 \neq x_2$ s.t. $h(x_1) = h(x_2)$ but $x_1 \neq x_2$.

• Problem 2: if $h$ is fixed, such $x_1, x_2$ could be known!
  Therefore, a non-uniform adversary $A$ can have these $x_1, x_2$ “hardwired” in the program.

• Idea 1: choose $h$ randomly from a family $\{h_i\}$ of CRHFs! (good for building a consistent theory)

• Idea 2: work with only uniform adversaries (probably good enough for all practical purposes: all the algorithms we write down, even those adversarially, are uniform).

• We focus on Idea 1.
Definition (Family of Collision-Resistant Hash Functions)

A set of functions $H = \{h_i : D_i \rightarrow R_i\}_{i \in I}$ is a family of collision-resistant hash functions (CRHF) if:

- (Easy to Sample) There is a PPT algorithm $Gen$ s.t. $Gen(1^n) \in I$.
- (Compression) $|R_i| < |D_i|$.
- (Easy to Evaluate) There is a PPT algorithm $Eval$ s.t. $\forall x \in D_i$ and $\forall i \in I$, $Eval(x, i) = h_i(x)$.
- (Collision-Resistance) $\forall$ non-uniform PPT $A$, there is a negligible function $\mu$ such that $\forall n \in \mathbb{N}$:

$$\Pr \left[ \begin{array}{c}
i \gets Gen(1^n), \\
(x, x') \gets A(1^n, i) \quad : \quad x \neq x' \land \\
h_i(x) = h_i(x')\end{array} \right] \leq \mu(n).$$
Remarks on CRHFs

- One-bit compression implies arbitrary compression. (why?)
- Ideally, we want $|h(x)| \leq |x|/2$.
- Merkle tree construction:
  - write string $x \in \{0, 1\}^*$ in “blocks”: $x = x_1 \| x_2 \| x_3 \| x_4 \| \ldots$
  - start with pairs to get “next level”: $y_1 = h(x_1 \| x_2), y_2 = h(x_2 \| x_4), \ldots$
  - do this all the way until to get the “root”;
- Root denotes the final “hash” written $h(x)$.

MAC with CRHF: use CRHF to compress a long message to short, then apply MAC to authenticate the short hash value.
- this gives MAC for long messages (from any MAC over short messages)
Remarks on CRHFs

- If a function is collision resistant for arbitrary length messages, it is also one-way.

- Proof: ?? (hint: ask collisions on \( h(x) \) for random \( x \))

- Unlikely that CRHF can be built from just OWF or even OWP [Simon-98]

- General attacks on CRHFs:
  - Enumeration attack: pick random \( x, x' \).
    
    Success probability \( \approx \frac{1}{\left|R_i\right|} - \frac{1}{\left|D_i\right|} \)
    
    \( \Rightarrow \) Range \( R_i \) cannot be too small. (Cannot compress a lot)
  - Birthday Attack: build a list as you go
    
    Start with random \( x \)s, look for collisions in the list
    
    Keep adding to the list until collisions are found.
    
    \( \approx \sqrt{\left|R_i\right|} \) tries needed.
Constructing CRHFs

- Many heuristic approaches, vibrant area of research!
  - MD5 (broken), SHA-1 family (broken just a few days back!)
  - SHA-2 family: SHA-256, SHA-384, SHA-512 (not yet “broken”)
- Provable construction from hard problems (slow)
  - constructions based on almost all interesting problems, e.g., DLP, factoring, LWE, Lattices, etc.
  - ... in a later class.