Introduction to Cryptographic Hash Functions

Vincent Rijmen
http://www.iaik.tugraz.at/aboutus/people/rijmen/index.php

Hash&Stream, Salzburg, 2007/02/01
Overview

- Requirements
- Applications
- Constructions
The working principle of a hash function.

Input: data of variable length
Output: fixed-length fingerprint
Fast processing algorithm
Secure
Pre-image resistance

Given a fingerprint, Can’t find a corresponding input
Second pre-image resistance

Given an input,

Can’t find a second input with same fingerprint
Collision resistance

Can’t find two inputs with same fingerprint
What does collision-resistance mean?

Model:
- Hash function inputs: balls
- Hash function output space: set of bins
- Hash function: throws balls in bins
Pigeonhole principle

If there are more balls than bins, then at least one bin gets at least two balls.
Birthday paradox

More than \( \sqrt{N} \) balls

\[ \Rightarrow \text{more than 50\% chance that two balls go into the same bin.} \]
A collision attack

M < sqrt(N) balls

Less than sqrt(N) balls, but still two balls into the same bin (*academic attack*)

Feasible number M balls (*practical attack*)
Summary

Collision, 2\textsuperscript{nd} preimage, preimage resistance:

- \textit{Computational} properties

- Attack: exploiting internal structure to be more efficient than the generic attack

- Collision attack is easier than 2\textsuperscript{nd} preimage attack is easier than preimage attack
Extra: almost-collisions

What if two inputs give _almost_ the same output?

- Called: _near-collision_
- Strictly speaking: no attack
- Usually not wanted
  - May be first step towards collision
  - May interact badly with applications
Applications of hash functions

- Payment schemes
- Secure storage of passwords
- Generation of pseudo-random numbers
- Electronic (digital) signatures
- Commitments
Example 1: Coin flip by telephone

Generate random bit $b$
- 0 = heads; 1 = tails

Choice: head or tails

Compare $b$ with choice

Outcome

Sad face 😞
Commitment

- Generate random number $r$
- Heads/tails = first bit of $r$
- Compute commitment $x = \text{hash}(r)$

$\text{Choice: heads or tails}$

$r \rightarrow x$
Requirements

- Collision resistance
  - Otherwise commitment is not binding
- Preimage resistance
  - Otherwise commitment can be inverted
Example 2: Electronic Signature

Cut and paste is easy with digital documents

Electronic signatures are *different* for each document

\[ f(\text{content of document, secret of signer}) \]
Example 2: Electronic Signature

Cut and paste is easy with digital documents

Electronic signatures are *different* for each document

\[ f(\text{fingerprint}, \text{secret of signer}) \]
Attack on Electronic Signature

Two documents producing the same fingerprint.

Same fingerprints $\Rightarrow$ same signatures.

Substitute the original document by a forgery
Constructing hash functions

- With or without secret parameter (key)
  - Message Authentication Codes (MAC)
  - Universal hash functions
  - Manipulation Detection Codes (MDC, hash)

- Compression function
- Extension method
  - Tree
  - Merkle’s method
Compression function

- Fixed-size inputs
- Processes a part (block) of the message
- Outputs are inputs for next iteration
Tree

\[ m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8 \]

\[ f, f, f, f, f, f, f, f \]

\[ f, f, f, f \]

\[ f, f, f, f \]

\[ h = \text{output} \]
Merkle’s method

\[ \text{IV} \quad m_1 \quad m_2 \quad m_3 \]

\[ f \quad f \quad f \]

\[ h_1 \quad h_2 \quad h_3 = \text{output} \]
Merkle-Damgaard Theorem

If:
- Unique padding of message until length is multiple of block length
- Length of (original) message included into padding

Then:
Collision-resistant $f \implies$ collision-resistant hash
Remarks

- Padding with length = Merkle-Damgaard strengthening

- No-one uses construction with provably collision-resistant f
Compression function constructions

- Block cipher based
- Dedicated designs
- Modular arithmetic
Block-cipher based designs

- MDC-2, MDC-4 (DES)
- Davies-Meyer, Matyas-Meyer-Oseas, Miyaguchi-Preneel
Motivation

- Reduce amount of primitives to be implemented
- Block cipher DES was the only cryptographic primitive with a certification
- Block cipher + feed forward = noninvertible map without apparent structure
The MD4 family of hash functions

- MD4 [Rivest 1990]
  - Avoid export of encryption software
  - Fast in software on 32-bit CPU’s
- MD5 [Rivest 1992]
  - More secure version

- Extended MD4, RIPE-MD, RIPEMD-160
- SHA, SHA-1, SHA-256, SHA-384, SHA-512, SHA-224
- HAVAL-3, HAVAL-4, HAVAL-5, HAS-1, HAS-2, …
MD5 structure

Message expansion

State update

\[ m_i \]

\[ h_{i-1} \]

\[ h_i \]
MD5 step

MD5

- Chaining variable: 4x32 bits
- Message input: 512 bits (16x32)
- 4 rounds of 16 steps
- Each step processes one message word
- Rounds differ in
  - Order of message words
  - Shift constants
  - Constants $K_i$
  - Function $F$
MD5 functions $F$

- $IF(x, y, z) = xy \oplus (~x)z$
- $IF'(x, y, z) = xz \oplus y(\sim z)$
- $XOR(x, y, z) = x \oplus y \oplus z$
- $F4(x, y, z) = y \oplus (x \text{ OR } (~z))$

Other hash functions also use:

- $MAJ(x, y, z) = xy \oplus yz \oplus xz$

Always acting on 32 bits in parallel
Attacks on MD4

- Reduced versions
  - Bosselaers and den Boer (*)
  - Merkle
  - Vaudenay
- Full version, collisions and preimages:
  - Dobbertin (*)

(*) Also results on extended MD4, MD5
- Not further
The MASH functions

- Based on difficulty of factoring
  - Hence not really symmetric cryptography
  - Slow

- \( \text{MASH-1}(n) \)
  
  \[
  y_i = 1111x_{i_1}1111x_{i_2}1111...x_{i_t}
  
  h_{i+1} = (((h_i \oplus y_i) \lor 0xF00...0)^2 \mod pq) \oplus h_i
  
  \]

- \( \text{MASH-2}(n) \): exponent \( 2^8+1 \)
Conclusions

- Very little diversity in designs
- Everyone seemed quite confident about the security of SHA-1 and siblings
  - Also reflected in specification of applications