Hash functions

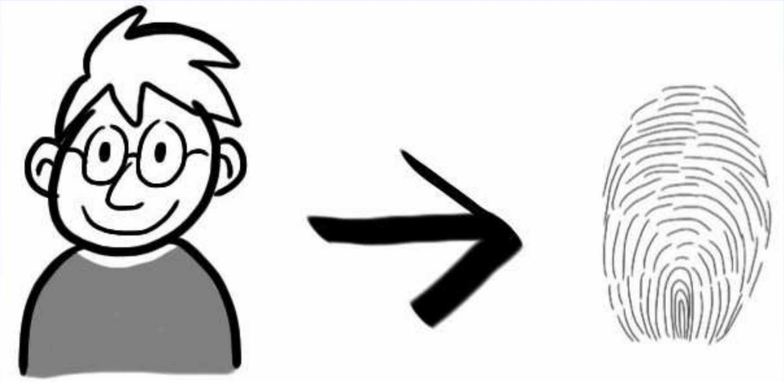
Chanathip Namprempre

Computer Science
Reed College

Agenda: Hash functions

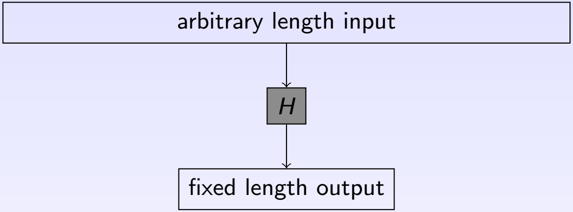
1. Concept
2. Hash function design
3. SHA1
4. Security definitions for hash functions: pre-image attacks
5. Security definitions for hash functions: second pre-image attacks
6. Security definitions for hash functions: collision attacks
7. Examples
8. Birthday paradox

A way to think about hash functions



A way to think about hash functions. The digest is supposed to capture the essense of the object being hashed.

Hash functions



The output is often called a message digest.

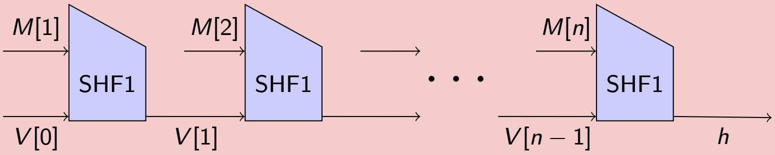
Example: SHA1 : $\{0,1\}^{<2^{64}} \to \{0,1\}^{160}$

Applications: how hash functions are used

- Password hashing: good idea, but be careful
- Message integrity: bad idea
- Fingerprint of large file: good idea, but make sure the fingerprint is the "real" one.
- Downline load security: same idea as above
- Digital signature efficiency: sign $H(M)$ instead of signing M

Hash function design: Merkle-Damgard

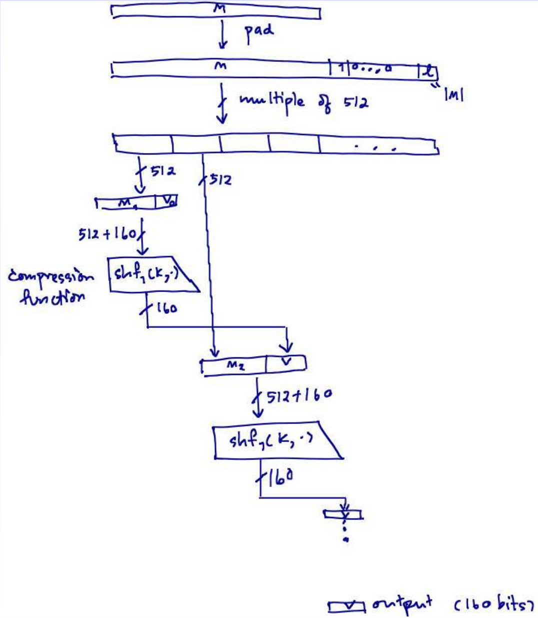
Merkle-Damgard: for example SHA1



The Merkle-Damgard construction.

- The trapezoids represent a compression function.
- The initial vector $V[0]$ is a fixed, public value.
- The messages $M[1], \ldots, M[n]$ are fixed-length blocks of the padded input message.
- The output is the output h of the compression function on the last round.

SHA1: vertically



SHA1

repeatly apply compression function on each block of message

$$\begin{aligned} 160 \text{ bits} &\leftarrow V_0 = 37452301||\ldots||C3D2E1F0 \\ 128 \text{ bits} &\leftarrow K = 5A827999||\ldots||CA62C1D6 \end{aligned}$$

Observation: use the word "key" but really the "key" is <u>known</u>!

SHA1 in pseudocode

algorithm SHA1(M) $\quad // \; |M| < 2^{64}$
 $V \leftarrow$ SHF1($5A827999 \| 6ED9EBA1 \| 8F1BBCDC \| CA62C1D6, M$)
return V

---

algorithm SHF1(K, M) $\quad // \; |K| = 128$ and $|M| < 2^{64}$
 $y \leftarrow$ shapad(M)
 Parse y as $M_1 \| M_2 \| \ldots \| M_n$ where $|M_i| = 512 (1 \leq i \leq n)$
 $V \leftarrow 67452301 \| EFCDAB89 \| 98BADCFE \| 10325476 \| C3D2E1F0$
 for $i = 1, \ldots, n$ do
  $V \leftarrow$ shf1($K, M_i \| V$)
return V

---

algorithm shapad(M) $\quad // \; |M| < 2^{64}$
 $d \leftarrow (447 - |M|) \bmod 512$
 Let I be the 64-bit binary representation of $|M|$
 $y \leftarrow M \| 1 \| 0^d \| I$ $\quad // \; |y|$ is a multiple of 512
return y

The compression function shf1 in SHA1

algorithm shf1(K, $B \| V$) // $|K| = 128, |B| = 512, |V| = 160$

Parse B as $W_0 \| \ldots \| W_{15}$ where $|W_i| = 32 (0 \leq i \leq 15)$

Parse V as $V_0 \| \ldots \| V_4$ where $|V_i| = 32 (0 \leq i \leq 4)$

Parse K as $K_0 \| \ldots \| K_3$ where $|K_i| = 32 (0 \leq i \leq 3)$

for $t = 16, \ldots, 79$ do

$W_t \leftarrow \mathrm{ROTL}^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16})$

$A \leftarrow V_0$; $B \leftarrow V_1$; $C \leftarrow V_2$; $D \leftarrow V_3$; $E \leftarrow V_4$

for $t = 0, \ldots, 19$ do

$L_t \leftarrow K_0$; $L_{t+20} \leftarrow K_1$; $L_{t+40} \leftarrow K_2$; $L_{t+60} \leftarrow K_3$

for $t = 0, \ldots, 79$ do

if $(0 \leq t \leq 19)$ then $f \leftarrow (B \wedge C) \vee ((\neg B) \wedge D)$

if $(20 \leq t \leq 39)$ OR $(60 \leq t \leq 79)$ then $f \leftarrow B \oplus C \oplus D$

if $(40 \leq t \leq 59)$ then $f \leftarrow (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$

$temp \leftarrow \mathrm{ROTL}^5(A) + f + E + W_t + L_t$

$E \leftarrow D$; $D \leftarrow C$; $C \leftarrow \mathrm{ROTL}^{30}(B)$; $B \leftarrow A$; $A \leftarrow temp$

$V_0 \leftarrow V_0 + A$; $V_1 \leftarrow V_1 + B$; $V_2 \leftarrow V_2 + C$; $V_3 \leftarrow V_3 + D$; $V_4 \leftarrow V_4 + E$

$V \leftarrow V_0 \| V_1 \| V_2 \| V_3 \| V_4$

return V

[All pseudocode is from Bellare-Rogaway lecture notes.]

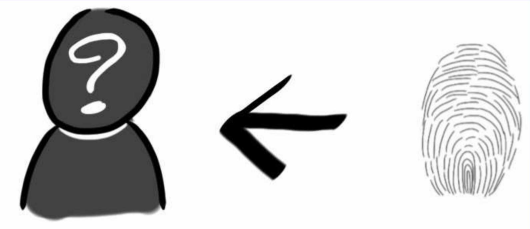
Security definitions for hash functions

What do we expect from hash functions?

Recall the common applications:

- password storage
- allowing people to check integrity of software that they download

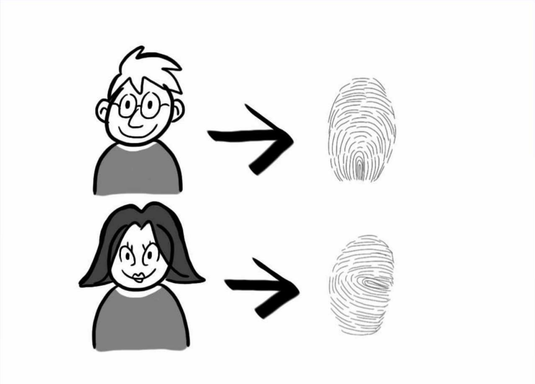
Security definition: Pre-image resistance



Preimage resistance means that it should be difficult to figure out what was hashed simply by looking at the hash value.

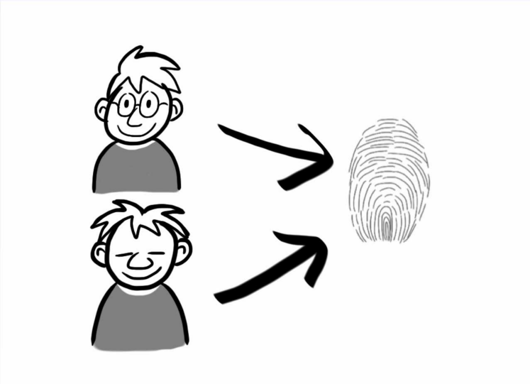
Example application: password storage

Security definition: collision resistance



For collision resistant hash functions, one would expect that, in most cases, hashing two distinct input values would result in two different hash values.

Security definition: collision resistance (cont.)



A collision can always occur, however, since a hash function maps values from a large set to those in a smaller set.

Example application: software integrity check

Example: SHA1

Definition

A collision for a function $h : \mathcal{D} \to \mathcal{R} \equiv$ a pair $x_1, x_2 \in \mathcal{D}$ such that $h(x_1) = h(x_2)$ but $x_1 \neq x_2$

Collision resistance of SHA1:

It is hard to find M and M' such that $\text{SHA1}(M) = \text{SHA1}(M')$ but $M \neq M'$

There are many such M and M' ! (by pigeonhole principle).

Security: resistance against pre-image attacks

Let m, n be integers such that $m > n$.
Let $H : \{0,1\}^m \to \{0,1\}^n$ be a hash function.

Subroutine *Initialize*	Experiment $\mathbf{Exp}_H^{\mathrm{pre}}(A)$
$x \xleftarrow{\$} \{0,1\}^m$; $h \leftarrow H(x)$ Return h	$h \leftarrow$ *Initialize* $x' \xleftarrow{\$} A(h)$ Return *Finalize*(x')
Subroutine *Finalize*(x') Return $(H(x') = h)$	

pre-image advantage

The pre-image advantage of an adversary A mounting a pre-image attack against H is

$$\mathbf{Adv}_H^{\mathrm{pre}}(A) = \Pr\left[\, \mathbf{Exp}_H^{\mathrm{pre}}(A) \Rightarrow \text{true} \,\right] .$$

Security: resistance against second pre-image attacks

Let m, n be integers such that $m > n$.
Let $H : \{0,1\}^m \to \{0,1\}^n$ be a hash function.

Subroutine *Initialize* $x \xleftarrow{\$} \{0,1\}^m$ Return x Subroutine $\mathit{Finalize}(x')$ Return $(H(x) = H(x') \wedge x \neq x')$	Experiment $\mathbf{Exp}_H^{\mathrm{sec}}(A)$ $x \leftarrow \mathit{Initialize}$ $x' \xleftarrow{\$} A(x)$ Return $\mathit{Finalize}(x')$

second pre-image advantage

The second pre-image advantage of an adversary A mounting a second pre-image attack against H is

$$\mathbf{Adv}_H^{\mathrm{sec}}(A) = \Pr\left[\, \mathbf{Exp}_H^{\mathrm{sec}}(A) \Rightarrow \mathrm{true} \,\right] .$$

Security: collision resistance

Let m, n be integers such that $m > n$.
Let $H : \{0,1\}^m \to \{0,1\}^n$ be a hash function.

Subroutine *Initialize*	Experiment $\mathbf{Exp}_H^{\mathrm{coll}}(A)$
Return	
	Initialize
Subroutine $\mathit{Finalize}(x, x')$	$(x, x') \xleftarrow{\$} A$
Return $(H(x) = H(x') \wedge x \neq x')$	Return $\mathit{Finalize}(x, x')$

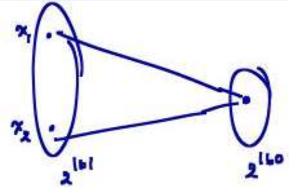
collision advantage

The collision advantage of an adversary A mounting a collision attack against H is

$$\mathbf{Adv}_H^{\mathrm{coll}}(A) = \Pr\left[\, \mathbf{Exp}_H^{\mathrm{coll}}(A) \Rightarrow \mathrm{true} \,\right] .$$

Collision-finding attacks: Example

Example: Suppose $|\mathcal{D}| = 2|\mathcal{R}|$



Collision-finding attack: Example (cont.) – strategy #1

Strategy # 1

Pick a point, go through all elements in the domain in some order until collide.

Worst case :

- the last one is the one and
- evenly distributed.
  [i.e. only 2 points collide for each element in $\mathcal{R}$, i.e. $\max_{y \in \mathcal{R}} |H_K^{-1}(y)| = 2$]

$$\text{number of trials needed} = 2^{161}$$

Collision-finding attack: Example (cont.) – strategy #2

Strategy # 2

Pick a point, pick another point at random from $\mathcal{D}$ until collide.

Find collision with probability about 1 in $|\mathcal{R}|$. So,

$$\text{number of trials needed} = 2^{160}$$

Collision-finding attack: Example (cont.) – strategy #3

Strategy # 3: Birthday attack

Pick random points from $\mathcal{D}$ until find a pair that collides.

$$\text{number of trials needed} = O(\sqrt{|\mathcal{R}|}) = O(2^{80}) < 2^{160}$$

Birthday paradox

Let $x_1, \ldots, x_n \in \mathcal{D}$ be independent identically distributed elements of $\mathcal{D}$.

Birthday bound

If $n = 1.2 \times |\mathcal{D}|^{1/2}$, then

$$\Pr[\,\exists i \neq j \;:\; x_i = x_j\,] \geq \frac{1}{2}\,.$$

Proof of birthday bound (for uniform independent x's)

Denote $|\mathcal{D}|$ by D.

$$
\begin{aligned}
\Pr[\,\exists i \neq j \;:\; x_i = x_j\,] &= 1 - \Pr[\,\forall i \neq j \;:\; x_i \neq x_j\,] \\
&= 1 - \left(\frac{D-1}{D}\right)\left(\frac{D-2}{D}\right)\cdots\left(\frac{D-n+1}{D}\right) \\
&= 1 - \prod_{i=1}^{n-1}\left(1 - \frac{i}{D}\right) \\
&\geq 1 - \prod_{i=1}^{n-1} e^{-\frac{i}{D}} \qquad\qquad [\text{because } 1 - x \leq e^{-x}] \\
&= 1 - e^{-\frac{1}{D}\sum_{i=1}^{n-1} i} \\
&\geq 1 - e^{-\frac{n^2}{2D}} \\
&\geq 1 - e^{-0.72} \\
&= 0.53
\end{aligned}
$$

So we get this bound when $n^2/2D = 0.72$, i.e. $n = 1.2 \times |\mathcal{D}|^{1/2}$.