Cryptographic Hash Functions

Lecture 13, CS 234, Autumn 2021
David Cash
Outline

1. Hash function basics, definitions
2. Hash function constructions
3. Small-space collision finding
Hash Functions in Computer Science

Data structures frequently use "hash functions".

\[ H : \text{Labels} \rightarrow \text{Indexes} \]

- \( H(x) \) "looks random"
- \( H \) can take a "key" (like \( H_k(x) = k \cdot x \mod \ell \))
- Collisions happen and are handled (\( H(x) = H(y) \))

Ex: \( \text{Labels} = \{ 0, 1, 3 \}^* \), \( \text{Indexes} = \{ 1, \ldots, 10,000 \} \)
Cryptographic Hash Functions

- Super strong version of a hash function

- Syntax: \( H : k \times D \rightarrow R \)
  - \( k \) is set of keys (ex: \( 0113^{123} \))
  - \( D \) is domain (ex: \( 0113^* \))
  - \( R \) is range (ex: \( 0113^{256} \))
Application: File Integrity

New version of MacOS! Patch file should have hash e97h01...

Apple Inc

Inet

=file

=file (MacOS w/ malware)

H(file) = H(file)

"collision"
Application: Commitments

I predict this weekend’s NFL scores are:

CHI 6  NYJ 10
DAL 28  SF 3

Before games: Publish \( y = H(\text{predictions || random}) \) used to prevent finding pred early.

After games: Reveal predictions || random. Everyone checks \( y \).

Security threats? – \( A \) might find input from \( y \) alone.

Collisions: \( y = H(\text{pred || rand}) = H(\text{pred’ || rand’}) \)
"One-wayness": Given $y = H(x)$, find $x$.

"Collision resistance": Find $x, x'$ such that $x = x'$, $H(x) = H(x')$.

... many possible properties.
Hash Security Goal: Collision Resistance

- Assume attacker knows key $K$! Key is not secret.

Goal: Design $H$ so that it is very hard to find $x \neq x' \in D$ such that

$$H(K, x) = H(K, x').$$

Bad news: For $H: K \times D \to \mathcal{R}$, if $|D| > |\mathcal{R}|$ ...

pigeonhole: \exists x \neq x' : H(x) = H(x')
Definition of Collision Resistance

Definition: Let $H : \mathcal{D} \times \mathcal{D} \rightarrow \mathcal{R}$ be a hash function and $A$ be an adversary. Define

$$\text{Expt}_H^\text{cr}(A)$$

1. Pick $k \in \mathcal{K}$ at random.

2. Give $k$ to $A$, which outputs $x, x'$.

3. If $x, x' \in \mathcal{D}$, $x \neq x'$, and $H(k, x) = H(k, x')$:
   - Output 1
   - Else Output 0

and $\text{Adv}_H^\text{cr}(A) = \Pr[\text{Expt}_H^\text{cr}(A) = 1]$. 

Obvious attack: time $\mathcal{O}(1)$
Example 1

\[ H(k, x_1 \| x_2) = AES(k, x_1) \oplus \ldots \oplus AES(k, x_t) \]

\[ H(k, x_1 \| x_2) = H(k, x_2 \| x_1) \]

\[ x = x_1 \| x_2 \]

\[ x^c = x_2 \| x_1 \]

\[ \text{Adv}_{H}^c(\Theta) = 1 \]
Example 2

\[ H(k, x_1, || x_2) = k \oplus \text{AES}(x_1, x_2) \]

\[ x_1 = 0^{128} \]
\[ x_2 = 0^{128} \]

\[ z = \text{AES}(x_1, x_2) \]

\[ x_1' = 1^{128} \]
\[ x_2' = 0^{128} \]

\[ z' = \text{AES}^{-1}(x_1', x_2') \]

\[ \text{Adv}^- = 1 \]
Recall: Collision Probabilities

Suppose we draw $q$ independent, uniform samples from a set of size $N$. Let $C(N,q)$ be the probability that a value is repeated in our samples.

\[ X_1, X_2, X_3, \ldots, X_i, \ldots, X_j, \ldots, X_q \]

**Theorem** For $q \leq \sqrt{2N}$,

\[ 0.3 \frac{q(q-1)}{N} \leq C(N,q) \leq 0.5 \frac{q(q-1)}{N} \]

\[ e^2 \frac{q^2}{N} \]
Birthday Attack: Collision Finding Against any $H$

Let $H: k \times D \rightarrow \Phi Z$

$A(K) \quad // Input: key K$

$\textbf{Output: Collision } x, x' (x \neq x' \text{ but } H(K,x) = H(K,x'))$

Initialize hash table $Y$

For $x = 1, \ldots, q_b$:

$y \leftarrow H(K,x) \quad // \text{Treat number } x \text{ as bit string }$  

If $Y[y] \neq \perp$:

$x' \leftarrow Y[y]$

$\underbrace{x, x'} \quad \text{output } x, x'$  

Else: $Y[y] \not\in x$

Heuristically, $\text{Adv}^{cr}_H(A) = \text{Col}(|R|, q) \approx \frac{q^2}{N}$
# Popular Hash Functions, Past and Present

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Output Length</th>
<th>Security Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA 1</td>
<td>1995</td>
<td>160</td>
<td>Collisions found in 2017</td>
</tr>
<tr>
<td>SHA 2 family</td>
<td>2001</td>
<td>224 or 256 or 384 or 512</td>
<td>Lookin' good :D</td>
</tr>
<tr>
<td>SHA 3 family</td>
<td>2015</td>
<td>Same</td>
<td>Lookin' good :D</td>
</tr>
</tbody>
</table>
Outline

1. Hash function basics, definitions
2. Hash function constructions
3. Small-space collision finding
Two steps:

1. Design a fixed-length function

   \[ h : \{ 0113 \}^{512} \times \{ 0113 \}^{256} \rightarrow \{ 0113 \}^{256} \]

2. Chain \( h \) together to build

   \[ H : \{ 0113 \}^* \rightarrow \{ 0113 \}^{256} \]

Finally, reason that \( H \) has good C.R.E. as long as \( h \) does.
Step 1: Design compression function $h$

Want: $h : \{0,1\}^{512} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$

- Build $h$ from a block cipher! SHA-256 uses a custom block cipher

$$E : \{0,1\}^{512} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$$

and defines

$$h(x,v) = E(x,v) \oplus v$$

Why not $h(x,v) = E(x,v)$? See problem set.
Step 2: Chain h together to build H

Assume we have \( h: \{0,1\}^{512} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256} \).

Now build \( H: \{0,1\}^{*} \rightarrow \{0,1\}^{256} \).

\[
H(x)
\]

\[ l \leftarrow \text{length}(x) \]

\[ \overline{x} \leftarrow \text{pad}(x) \] // add zeros

Parse \( x_1 \parallel \cdots \parallel x_t \leftarrow \overline{x} \) (\( x_t \in \{0,1\}^{512} \))

Set \( v_0 \) to a magic number \( 6a69e6\ldots \)

For \( i=1 \ldots t \): \( v_i \leftarrow h(x_i, v_{i-1}) \)

\( v_{t+1} \leftarrow h(x_t, v_t) \)

Output \( x_{t+1} \)

512-bit encoding of \( l \)
Step 2 in a Picture

\[ x_1 \| x_2 \| \cdots \| x_t \]

Merkle-Damgård Chaining/Transform

\[ \text{output} \]

\[ H(x) \]
Claim: Given a collision $x, x'$ for $H$, one can easily find a collision for $h$.

$\Rightarrow$ If attackers can't find collisions in $h$, then they can't find collisions in $H$ either!
Analysis

Given a collision $x, x'$ for $H$, need to find collision for $h$

Case 1. $\text{length}(x) = \text{length}(x')$

$h(x, y) = h(x', y')$
Analysis continued

Case 2 \( \text{length}(x) = \text{length}(x') \)
Outline

1. Hash function basics, definitions
2. Hash function constructions
3. Small-space collision finding
Birthday Attacks and Space

Let $H: k \times D \to \mathbb{Z}$

$A(k)$ // Input: key $k$
Output: Collision $x, x'$ ($x \neq x'$ but $H(k, x) = H(k, x')$)

Initialize hash table $Y$

For $x = 1, \ldots, q_b$:

$y \leftarrow H(k, x)$ // Treat number $i$ as bit string input

If $Y[y] \neq \perp$

$x' \leftarrow Y[y]$

output $x, x'$

Size of $Y$:

$s \approx q_b$

$s \approx \sqrt{|R|}$

$Y = \sqrt{|R|}$

$2^{64}$ space 😞
An Interview Question

Suppose you are given a pointer \( p \) to a linked list, and told that the list contains a cycle. Show how to find the "colliding pointers" using as little memory as possible.
Solving the Puzzle: Floyd’s Cycle Detecting Alg

Finds the cycle in “constant space”!

tortoise ← head
hare ← head
While tortoise ≠ hare:
  tortoise ← next(tortoise)
  hare ← next(next(hare))  # double speed

Output tortoise, hare
Analysis of Floyd's Alg

Tail length: $l_{tail}$

Loop length: $l_{loop}$

Tail: $a_3$

Hare: $a_6$

$a_6 = a_{14}$

$l_{tail} = 6$

$l_{loop} = 8$
Analysis of Floyd's Alg

Claim 1  For any $i \geq l_{\text{tail}}$, 
\[ a_i = a_{i+l_{\text{loop}}} = a_{i+2 \cdot l_{\text{loop}}} = a_{i+3 \cdot l_{\text{loop}}} = \ldots \]

Moreover, if $i > l_{\text{tail}}$ and $a_i = a_{i+j}$, then $j$ is a multiple of $l_{\text{loop}}$. 
Claim 1. For any $i > \ell_{\text{tail}}$,

$$a_i = a_{i+\ell_{\text{loop}}} = a_{i+2\ell_{\text{loop}}} = a_{i+3\ell_{\text{loop}}} = \ldots$$

Moreover, if $i > \ell_{\text{tail}}$ and $a_i = a_{i+j}$, then $j$ is a multiple of $\ell_{\text{loop}}$.

Corollary. $a_i = a_{2i}$ if and only if $i > \ell_{\text{tail}}$ and $i$ is a multiple of $\ell_{\text{loop}}$.

Proof ($\Rightarrow$) If $i = k\ell_{\text{loop}}$, $a_{2i} = a_{i+i} = a_{i+k\ell_{\text{loop}}} = a_i$.

($\Leftarrow$) If $a_i = a_{2i}$, then $i > \ell_{\text{tail}}$ because there are not early repeats. Moreover, since $a_i = a_{2i} = a_{i+i}$, $i$ is a multiple of $\ell_{\text{loop}}$ by second part of Claim 1.
Analysis of Floyd's Alg

Claim 2. For some $1 \leq i \leq l_{\text{tail}} + l_{\text{loop}}$, $a_i = a_{2i}$.

Proof. Take $i = \text{smallest multiple of } l_{\text{loop}} \text{ that is at least } l_{\text{tail}}$.

Apply corollary: $a_{2i} = a_{i+i} = a_{i+i} + l_{\text{loop}} = a_i$.

Claim 3. If $i$ is a multiple of $l_{\text{loop}}$, then $a_{i+l_{\text{tail}}} = a_{i+l_{\text{tail}}}$.

Proof. This is just the original claim 1 (but with $i = l_{\text{tail}}$ there!)
Analysis of Floyd's Algorithm

Putting it together:

① Alg will find tortoise = hare in at most $l_{tail} + l_{loop}$ iterations
   - By claim 2, there is $i \leq l_{tail} + l_{loop}$ such that $a_i = a_{2i}$

② When tortoise = hare, this pointer is exactly $l_{tail}$ steps from the collision.
   - By claim 3, $a_{l_{tail}} = a_{i + l_{tail}}$ for this $i$.

③ Restarted walk will arrive at collision in $l_{tail}$ steps as well!
Putting it together

Define a "list" $H(\emptyset), H(H(\emptyset)), H(H(H(\emptyset)))$...

```
tortoise ← 0
hare ← 0
While tortoise ≠ hare:
    tortoise ← H(tortoise)
    hare ← H(H(hare))  # double speed

Output tortoise, hare
```

```
tortoise ← head  # restart
While H(tortoise) ≠ H(hare):
    tortoise ← H(tortoise)
    hare ← H(hare)  # single speed
```

Heuristically, expect a collision after about $\sqrt{|R|}$ steps.
Rest of analysis is the same!

Space: $\sim \emptyset$

Run Time: $\sim \sqrt{|R|}$
3Day w/ 10,000 machines: \[ \frac{\sqrt{121}}{10,000} \]

Floyd's w/ 10,000 machines: \[ \sqrt{121} \]

→ more advanced algo, time/space "parallelism"