Hash Functions

10 April 2025
Lecture 4

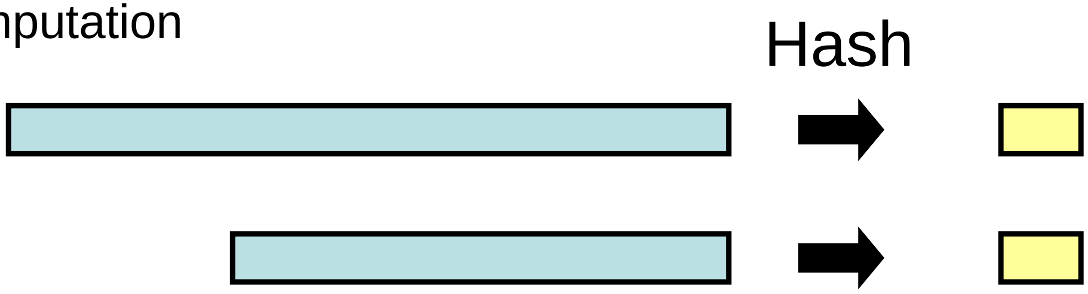
Slide Credits: Steve Zdancewic (UPenn)

Topics for today

- Cryptographic Hashes
- Hash Theory

Hash Algorithms

- Hash function defined by:
  - Compression
    - Take a variable length string, Produce a fixed length digest
    - Typically 128-1024 bits
  - Ease of Computation



- (Noncryptographic) Examples:
  - Parity (or byte-wise XOR)
  - CRC (cyclic redundancy check) used in communications
  - Ad hoc hashes used for hash tables
- Realistic Example
  - The NIST Secure Hash Algorithm (SHA) takes a message of less than 2^{64} bits and produces a digest of 160 bits

Cryptographic Hash History

Message Digest: MD4 – (128 bits)

- Invented by Rivest
- MD5 – successor
- Compromised completely – no security guarantees for it

Secure Hash Algorithm: SHA-0 – (160 bits)

- Developed by US National Security Agency (NSA) in 1993

Secure Hash Algorithm: SHA-1 – (160 bits)

- Also by NSA to fix a bug in SHA (1995)

Attacks have been found against both SHA-0 and SHA-1

Enter SHA-2

Also by NSA, in 2001

- Related to SHA-1 (same structure)

Family of algorithms with varying output sizes

- SHA-256 and SHA-512 are new algorithms

SHA-256

- 64 rounds
- SHA-224 (Truncated version)

SHA-512

- 80 rounds
- SHA-384 (Truncated version)

Latest Hash Algorithm: SHA-3

- 2007: US National Institutes of Standard and Technology (NIST) announces secure hash algorithm competition
  - Round 1: 51 entries
  - Round 2: 14 entries

- Chose 5 finalists (BLAKE, Grostl, JH, Keccak, Skein)

- 2012: Winner Keccak by *Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche*
  - Remember Daemen from AES?
  - Keccak is not at all like SHA-1 or SHA-2
  - Also varying output lengths (224, 256, 384, 512, or whatever)

Opinions about SHA-3

SHA-3 (Keccak) is a whole new family of hash algorithms

> “Thus I believe that SHA-3 should probably not be used. It offers no compelling advantage over SHA-2 and brings many costs. The only argument that I can credit is that it's nice to have a backup hash function…” – Adam Langley (Google)

Cryptographic Hashes

Creates a hard-to-invert summary of input data

Good for integrity properties:

Sender computes the hash of the data, transmits data and hash

Receiver uses the same hash algorithm, **checks** the result

Like a check-sum or error detection code

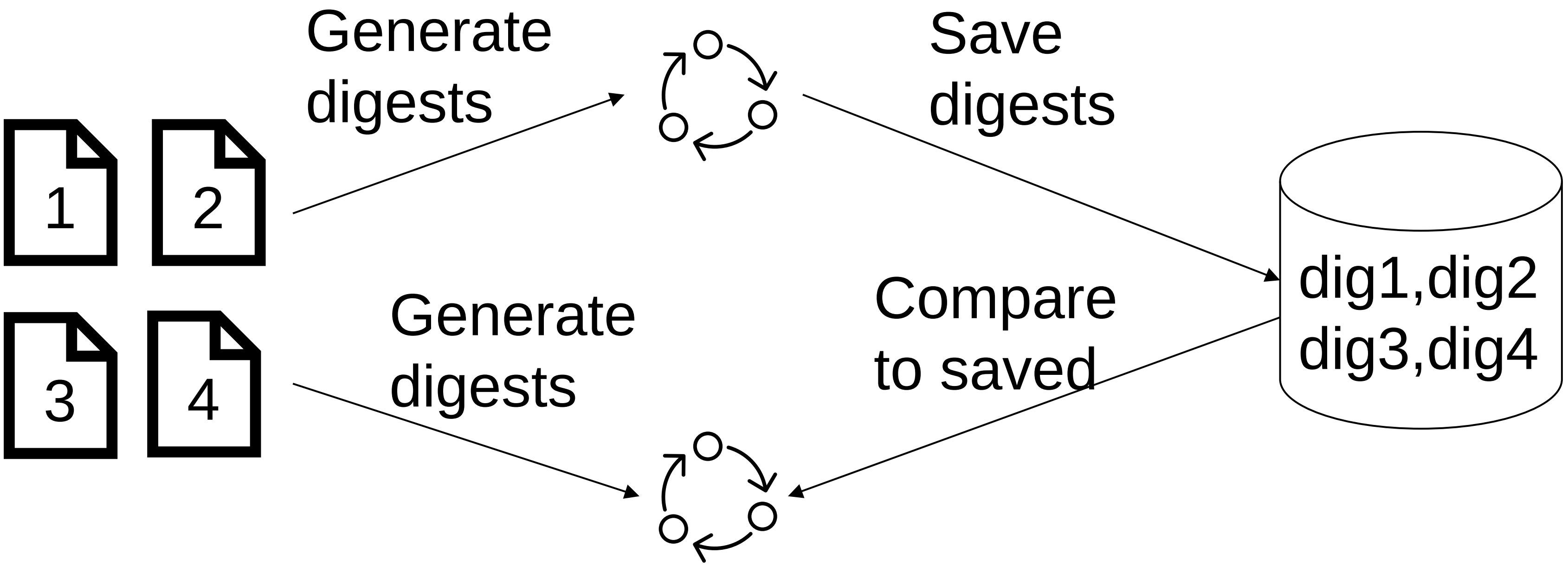
Uses a cryptographic algorithm internally

More expensive to compute

Also called a Message Digest

Uses of Hash Algorithms: #1

- Hashes are used to protect *integrity* of data
  - Virus Scanners
  - Program fingerprinting in general
  - Modification Detection Codes (MDC)



Uses of Hash Algorithms: #2

- Message Authenticity Code (MAC)
  - Hash with a cryptographic secret or component
  - Send $(msg, hash(msg, key))$
  - Attacker who doesn’t know the key can’t modify msg (or the hash)
  - Receiver who knows key can verify origin of message

- Make digital signatures more efficient (we'll see this later)

Using a MAC

$msg, mcode$

Alice

K_{AB}

Bart

K_{AB}

1. Wants to send
msg
2. Calculates
$mcode = MAC(K_{AB}, msg)$

3. Calculates
$v = MAC(K_{AB}, msg)$
4. Checks
$v == mcode$

MAC Standards: HMAC

Keyed-hash Message Authentication Code (HMAC) (FIPS 198-1)

1. Choose a message (msg)
2. Choose a secret (K)
3. Choose a good hash algorithm (ex. SHA2-256)
4. 0-pad or truncate K so it's $= 64$ Bytes $\rightarrow key$
5. $p_i = key \oplus ipad$
    - $ipad = 0x363636$ ...
6. $temp_1 = H(p_i \,||\, msg)$
7. $mcode = H((key \oplus opad)||temp_1)$
    - $opad = 0x5c5c5c5c$ ...

|| is concatenation

Summary:

$$HMAC(K, msg) = H((K \oplus opad)||H((K \oplus ipad)||msg))$$

- Uses the hash function twice to prevents some attacks

So Far

- Cryptographic Hashes
- Hash Theory

What's a good hash?

- Probability that a randomly chosen message maps to an n-bit hash digest should be $\left(\frac{1}{2}\right)^{n}$.
  - Meaning: Attacker must work hard to modify the source message without altering the hash digest value

- Properties for (unkeyed) hashes:
  1. Preimage Resistance
  2. 2^{nd}-preimage Resistance
  3. Collision Resistance

What’s a good hash?

Hash functions h for cryptographic use as MDC’s fall in one or both of the following classes.

– *One Way Hash Function:* Given a specific hash value *y*, it should be computationally infeasible to find an input x such that $h(x) = y$. (1&2)

– *Collision Resistant Hash Function:* It should be computationally infeasible to find two distinct inputs that hash to a common value (i.e. $h(x) = h(y)$). (3)

Design Objective for n-bit Hash

Hash Type	Design goal	Ideal Strength	Adversary's goal
One Way Hash Function (OWHF)	Preimage resistance	2^n	Produce preimage
	2nd-preimage resistance	2^n	Find 2nd input, same image
Collision Resistant Hash Function (CRHF)	Collision resistance	$2^{\frac{n}{2}}$	Produce any collision

Secure Hash Algorithm (SHA)

- Pad message so it can be divided into 512-bit blocks, including a 64 bit value giving the length of the original message.
- Process each block as 16 32-bit words called $W(t)$ for t from 0 to 15.
- Expand from these 16 words to 80 words by defining as follows for each t from 16 to 79:
  - $W(t) = W(t-3) \oplus W(t-8) \oplus W(t-14) \oplus W(t-16)$
- Constants $H0, \ldots, H4$ are initialized to special constants
- Result is final contents of $H0, \ldots, H4$

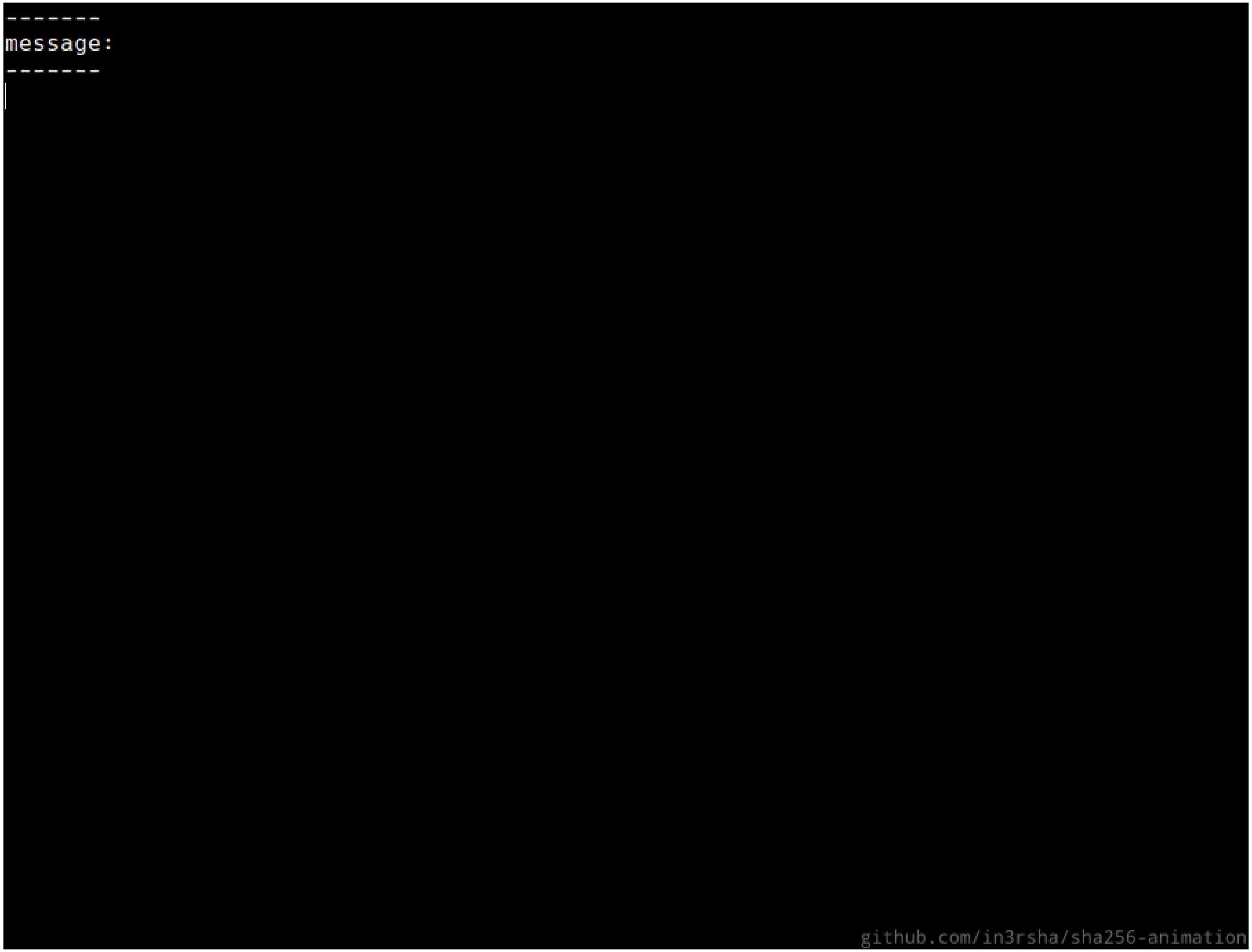
```
for each 16-word block begin
    A := H0; B := H1; C := H2; D := H3; E := H4
    for I := 0 to 19 begin
        TEMP := S(5,A) + ((B ∧ C) ∨ (¬ B ∧ D)) + E + W(I) + 5A827999;
        E := D; D := C; C := S(30,B); B := A; A := TEMP
    end
    for I := 20 to 39 begin
        TEMP := S(5,A) + (B ⊕ C ⊕ D) + E + W(I) + 6ED9EBA1;
        E := D; D := C; C := S(30,B); B := A; A := TEMP
    end
    for I := 40 to 59 begin
        TEMP := S(5,A) + ((B ∧ C) ∨ (B ∧ D) ∨ (C ∧ D)) + E + W(I) + 8F1BBCDC;
        E := D; D := C; C := S(30,B); B := A; A := TEMP
    end
    for I := 60 to 79 begin
        TEMP := S(5,A) + (B ⊕ C ⊕ D) + E + W(I) + CA62C1D6;
        E := D; D := C; C := S(30,B); B := A; A := TEMP
    end
    H0 := H0+A; H1 := H1+B; H2 := H2+C; H3 := H3+D; H4 := H4+E
end
```

Chaining Variables

Shift A left 5 bits

How SHA2-256 Works

https://www.youtube.com/watch?v=f9EbD6iY9zI



Attacks against SHA-1

- Early 2005: Rijmen and Oswald publish attack
  - Reduced version of SHA-1 (53 out of 80 rounds)
  - Finds collisions with less than 2^{80} operations.

- Feb 2005: Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu public attack
  - Find collisions in the full version of SHA-1
  - Fewer than 2^{69} operations (brute force is 2^{80})
- Aug 2005: Same group, threshold now 2^{63}

- 2012: Best attack thought to be by Marc Stevens with complexity 2^{61}

SHA-1 has been deprecated

- NIST says to stop using in 2010
- MS and Google stopped accepting it in 2017

SHATTERED https://shattered.io/

Full SHA1 Collision
Stevens (2017)

Cost: $100,000
6,500 CPU Years
100 GPU Years

Built on PDF image headers (collision is two PDFs which look different)

Used GPUs → Longer than theoretical time ($2^{63.1}$ instead of 2^{61})

Summary

- Cryptographic Hashes
- Hash Theory