Computer and Network Security

Lecture 5

Hash Functions and Message Digest

Outline

- Hash Functions
- Basic properties
- Popular hash functions
- Applications

Scope

- Problem
  - Data integrity
    - Error in transmission
    - Malicious manipulation
- Solution
  - Error detection/correction codes
    - CRC
  - Hash functions

Properties

- H: $\{0,1\}^* \rightarrow \{0,1\}^n$
- Non-cryptographic hash functions
  - Arbitrary-length input
  - Fixed-length output
  - Efficient to compute
- Cryptographic hash functions
  - One-way
  - Strong Collision resistance
  - Weak Collision resistance

One-way

- Given ***x***
  - Computation of **y = f(x)** is **easy**
- Given **y**
  - Computation of **x = f^{-1}(y)** is **hard**

- Example -- DLP (Discrete Logarithm Problem)
  - p, prime; $Z_p^* = \{1,\ldots,p-1\}$; $g \in Z_p^*$, generator
  - $f(x) = g^x \bmod p$

  - $p = 17$, $Z_p^* = \{1,\ldots,16\}$; $g = 3$; $f(x) = 3^x \bmod 17$
  - x = 6 → $y = f(x) = 3^6 \bmod 17 = 15$
  - y = 15 → $x = f^{-1}(y) = ???$

Other properties

- Strong Collision resistance
  - It is hard to find a,b such that
    - a ≠ b
    - H(a) = H(b)
- Weak Collision resistance
  - Given a, it is hard to find b such that
    - a ≠ b
    - H(a) = H(b)

Security

- H: $\{0,1\}^* \rightarrow \{0,1\}^{64}$
- Given y = H(x)
  - How many trials for a collision?

- 2^{64} possible outputs
  - $2^{64}/2 = 2^{63}$?
  - Not really!

The Birthday paradox

- How many people are enough, so that the probability that two random people of them have the same birthday (month and day) is ≥ ½?

- Answer: 23

- Does it help attacking hash functions?

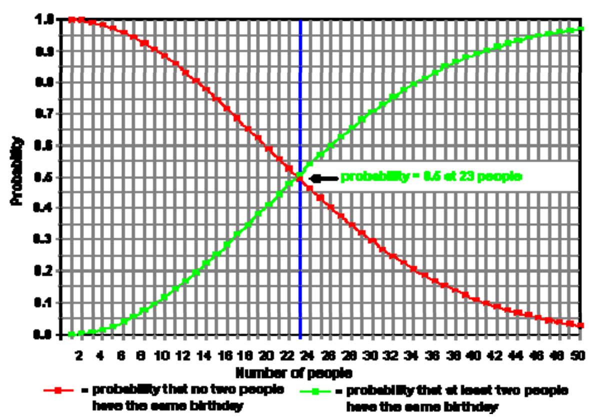
The Birthday paradox

- y = H(x)
  - x = person
  - H() = Birthday()
  - y ∈ {1,…,365}; let n be the size of the set
- How many people do we need to 'hash' to have a collision?

- Probability of having no collision
  - $P_0=1*(1-1/n)*(1-2/n)*…*(1-(k-1)/n)) \approx e^{k(1-k)/2n}$
- Probability of having at least one collision
  - $P_1=1-P_0$
  - Set P_1 to be at least 0.5 and solve for k
    - K ≈ 1.17 * SQRT(n)
    - k = 22.3 for n=365

- So what?

The Birthday paradox

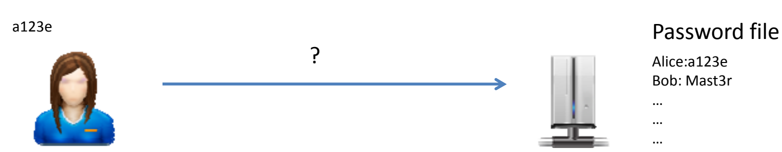


The Birthday paradox

- Assume that |H(x)| = n bits
  - $\sqrt{2^n} = 2^{\frac{n}{2}}$ trials are enough to find a collision with prob. ≥ 0.5
- How long should |H(x)| be?
  - Many input messages yield the same hash
  - E.g., 1024-bit message, 128-bit hash
    - On average, 2896 messages map into one hash
  - With n-bit hash, it takes about $2^{n/2}$ trials to find a collision with ≥ 0.5 prob.
  - When n = 64, it takes 2^{32} trials to find a collision (not 2^{63})
  - Today, need at least n = 128, requiring about 2^{64} trials

Application

- Password storage



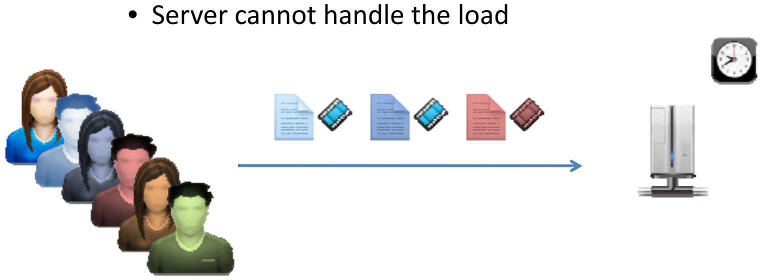
- Eavesdropping?
- Stolen password file

Application

- Digital Signatures
  - Alice computes signature σ message m
    - Forgery should not work
  - Anybody can verify (σ, m)
    - Signature schemes only sign m ≈ 160-bits

Application

- Electronic paper submission
  - Strict deadline: 9:45pm CET, March 21st
  - Last minutes are hectic
    - Servers slow down
  - Attachment might be several GBs
    - Videos
    - Server cannot handle the load

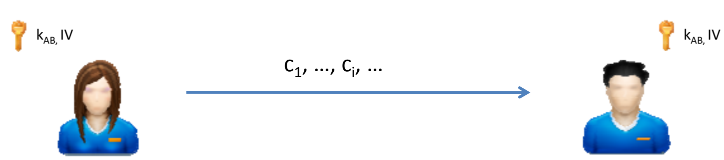


Popular Hash Functions

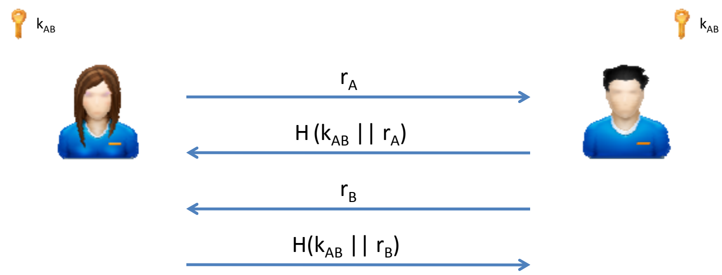
	SHA-256	MD5 (defunct)	RIPEMD-160
Digest length	256 bits	128 bits	160 bits
Block size	512 bits	512 bits	512 bits
# of steps	80	64	160
Max msg size	2^{64}-1 bits		

Hash Functions for Encryption

- (almost) One-time pad
  - $b_1 = H(K_{AB} \mid\mid IV)$, …, $b_i = H(K_{AB} \mid\mid b_{i-1})$, …
  - $c_1 = p_1$ XOR b_1, …, $c_i = p_i$ XOR b_i, …



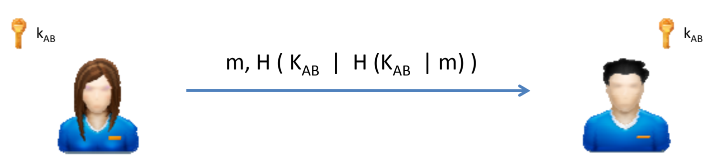
Hash Functions for Authentication



- Only requires hash computation

Hash Functions for Integrity

- Regular hash might be replace by any malicious party
- Requires HMAC
  - Prefix:
    - MAC = $H(K_{AB} | m)$
    - Allows concatenation with arbitrary message:
      $H(K_{AB} | m | m')$
  - Suffix:
    - MAC = $H(m | K_{AB})$
    - Collision in H() → Collision in HMAC
  - HMAC:
    - $H (K_{AB} | H (K_{AB} | m))$



HMAC

- **Main Idea**: Use a MAC derived from any cryptographic hash function
  - Note that hash functions do not use a key, and therefore cannot serve directly as a MAC
- **Motivations for HMAC:**
  - Cryptographic hash functions execute faster in software than encryption algorithms such as DES
  - No need for the reverseability of encryption
  - No export restrictions from the US (was important in the past)
- **Status**: designated as mandatory for IP security
  - Also used in Transport Layer Security (TLS), which will replace SSL, and in SET