Lecture 4.1: Hash Functions, and Message Authentication Codes

CS 436/636/736
Spring 2015

Nitesh Saxena

---

Course Administration

- HW1 solution will be posted
- We are grading now
  - Should be done early next week
Course Administration

• Expect HW2 coming out by Monday
  – Covers mostly Public Key Cryptography
  – Due in 10-12 days

Outline of Today’s lecture

• Hash Functions
  – Properties
  – Birthday Paradox
  – Generic Design

• Message Authentication Codes
  – Based on block cipher designs
  – Based on hash functions
Cryptographic Hash Functions

• Requirements of cryptographic hash functions:
  – Can be applied to data of any length.
  – Output is fixed length, usually very short
  – Relatively easy to compute $h(x)$, given $x$
  – Function is **deterministic**
  – Infeasible to get $x$, given $h(x)$. **One-wayness property**
  – Given $x$, infeasible to find $y$ such that $h(x) = h(y)$.
    **Weak-collision resistance property**
  – Infeasible to find any pair $x$ and $y$ ($x \neq y$) such that $h(x) = h(y)$.
    **Strong-collision resistance property or just collision resistance**

Some Applications of Hash Functions

• In general, can be used as a checksum for large amounts of data
• Password hashing
• Digital signatures
• Message authentication codes (will study later)
• Used also in RSA-OAEP, and many other cryptographic constructions
Hash Output Length

- How long should be the output (n bits) of a cryptographic hash function?
- To find collision - randomly select messages and check if hash matches any that we know.
- Throwing k balls in N = 2^n bins. How large should k be, before probability of landing two balls in the same becomes greater than ½?
- **Birthday paradox** - a collision can be found in roughly \( \sqrt{N} = 2^{(n/2)} \) trials for an n bit hash
  - In a group of 23 (~ \( \sqrt{365} \)) people, at least two of them will have the same birthday (with a probability > ½)
- Hence n should be at least 160

Birthday Paradox

- Probability that hash values of k random messages are distinct is (that is, no collisions) is:

\[
= \left(1 - \frac{1}{N}\right) \left(1 - \frac{2}{N}\right) \cdots \left(1 - \frac{k-1}{N}\right) = \prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right)
\]

\[
\approx \prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right)
\]

(as for small \( x \), \( e^{-x} \approx 1 - x \), as \( e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} \cdots \))

\[= e^{-k(k-1)/2N} \]

So for at least one collision we have probability of \( 1 - e^{-k(k-1)/2N} \)

whose value is above 0.5 when \( k = 1.17\sqrt{N} \)
Generic Hash Function – Merkle-Damgard Construction

- This design for $H()$ is collision-resistant given that $h()$ is collision resistant.
- Intuitively, this is because there is an avalanche effect – even if the inputs differ in just 1 bit, the outputs will be completely different.
- IV is a known public constant.

An Illustrative Example from Wikipedia

- **Fox**
  - The red fox jumps over the blue dog
    - cryptographic hash function
    - 008E 46B8 FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC

- **The red fox jumps over the blue dog**
  - cryptographic hash function
  - 8FDB 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819

- **The red fox jumps over the blue dog**
  - cryptographic hash function
  - FCD3 7FDD 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45

- **The red fox jumps over the blue dog**
  - cryptographic hash function
  - 8ACA D682 D588 4C75 4BF4 1799 7D88 BC06 92B9 6A6C
Practical Examples

• SHA-1
  – Output 160 bits
  – B’day attack requires $2^{80}$ calls
  – Faster attacks $2^{69}$ calls

• MD5
  – Output is 128 bits, so B’day attack requires $2^{64}$ calls only
  – Faster attacks to find a collision:

• Better use stronger versions, such as SHA-256
• Although, these attacks are still not practical – they only find two random messages that collide

Further Reading

• Stallings Chapter 11
• HAC Chapter 9
Message Authentication Codes

- Provide integrity as well as authentication
- Send \((m, MAC)\); MAC is created on \(m\) using the key shared between two parties
- Has to be **deterministic** to enable verification  
  - Unlike encryption schemes
- We want MAC to be as small and as secure as possible
- Can not provide non-repudiation  
  - Why not?

MAC – Functions

- KeyGen – outputs a key
- MAC – creates a checksum on \(m\) using key \(K\)
- Verify – validates whether the checksum on \(m\) is computed correctly  
  - Just create MAC and compare
Security Notion for MAC

• Very similar to the security notion for a digital signature scheme
• Existential forgery under (adaptively) chosen message attack

Can encryption be used as a MAC?

• No, not in general
• Intuitively because encryption achieves properties different from MAC
• See a counter-example:
  – One-time pad as a MAC – bad idea!
• However, you can use 3-DES or AES as a building block for MAC
MAC Based on Block Cipher in the CBC mode – CBC-MAC

Note that this is deterministic
- IV = [0]
- Unlike CBC encryption

Only the last block of the output is used as a MAC

This is secure under CMA attack
- For pre-decided fixed-length messages
- Intuitively because of the presence of an avalanche effect
HMAC: MAC using Hash Functions

- Developed as part of IPSEC - RFC 2104. Also used in SSL etc.
- Key based hash but almost as fast as non-key based hash functions.
- Avoids export restrictions unlike DES based MAC.
- Provable security
- Can be used with different hash functions like SHA-1, MD5, etc.

HMAC

Block size b bits.

\[ K^+ - K \text{ padded with bits on the left to make } b \text{ bits.} \]

- ipad – 0110110 (0x36) repeated b/8 times.
- opad – 1011100 (0x5c) repeated b/8 times.

Essentially

\[ \text{HMAC}_K = H((K^+ \text{xor } \text{opad}) \ || \ H((K^+ \text{xor } \text{ipad}) \ || \ M)) \]
Security of HMAC

- Security related to the collision resistance of the underlying hash function

http://www.cse.ucsd.edu/~mihir/papers/hmac.html
Further Reading

• Stallings Chapter 12 (MAC)
• HAC Chapter 9 (MAC)