Computer Security
CS 526
Topic 5

Cryptography: Cryptographic Hash Functions And Message Authentication Code
Announcements

• Quiz 1 will be on Sept 16, covering topics 1-5

• Mid-term exam tentatively scheduled to be Tuesday Oct 21, during lecture time
Readings for This Lecture

• Wikipedia
  • Cryptographic Hash Functions
  • Message Authentication Code
Data Integrity and Source Authentication

- Encryption does not protect data from modification by another party.
- Most encryption schemes are malleable:
  - Modifying ciphertext result in (somewhat) predictable change in plaintext
- Need a way to ensure that data arrives at destination in its original form as sent by the sender.
Hash Functions

- A hash function maps a message of an arbitrary length to a \( m \)-bit output
  - output known as the fingerprint or the message digest

- What is an example of hash functions?
  - Give a hash function that maps Strings to integers in \([0, 2^{32} - 1]\)

- Cryptographic hash functions are hash functions with additional security requirements
Using Hash Functions for Message Integrity

• Method 1: Uses a Hash Function \( h \), assuming an authentic (adversary cannot modify) channel for short messages
  – Transmit a message \( M \) over the normal (insecure) channel
  – Transmit the message digest \( h(M) \) over the authentic channel
  – When receiver receives both \( M' \) and \( h \), how does the receiver check to make sure the message has not been modified?

• This is insecure. How to attack it?
• A hash function is a many-to-one function, so collisions can happen.
Security Requirements for Cryptographic Hash Functions

Given a function $h: X \rightarrow Y$, then we say that $h$ is:

- **preimage resistant (one-way):**
  if given $y \in Y$ it is computationally infeasible to find a value $x \in X$ s.t. $h(x) = y$

- **2-nd preimage resistant (weak collision resistant):**
  if given $x \in X$ it is computationally infeasible to find a value $x' \in X$, s.t. $x' \neq x$ and $h(x') = h(x)$

- **collision resistant (strong collision resistant):**
  if it is computationally infeasible to find two distinct values $x', x \in X$, s.t. $h(x') = h(x)$
Usages of Cryptographic Hash Functions

• Software integrity
  – E.g., tripwire

• Timestamping (cryptographic commitment)
  – How to prove that you have discovered a secret on an earlier date without disclosing the context of a secret?

• Covered later
  – Message authentication
  – One-time passwords
  – Digital signature
Bruteforce Attacks on Hash Functions

- Attacking one-wayness
  - Goal: given $h: X \rightarrow Y$, $y \in Y$, find $x$ such that $h(x) = y$
  - Algorithm:
    - pick a random value $x$ in $X$, check if $h(x) = y$, if $h(x) = y$, returns $x$; otherwise iterate
    - after failing $q$ iterations, return fail
  - The average-case success probability is
    $$\varepsilon = 1 - \left(1 - \frac{1}{|Y|}\right)^q \approx 1 - e^{-\frac{q}{|Y|}} \approx \frac{q}{|Y|}$$
    - The first approximation holds when $|Y|$ is large,
    - The second roughly holds when $q/|Y|$ is small (e.g., < 0.5)
  - Let $|Y| = 2^m$, to get $\varepsilon$ to be close to 0.5, $q \approx 2^{m-1}$
Brute-force Attacks on Hash Functions

- Attacking collision resistance
  - Goal: given $h$, find $x, x'$ such that $h(x)=h(x')$
  - Algorithm: pick a random set $X_0$ of $q$ values in $X$
    for each $x \in X_0$, computes $y_x = h(x)$
    if $y_x = y_{x'}$ for some $x' \neq x$ then return $(x,x')$ else fail
  - The average success probability is
    $$1 - \left(1 - \frac{1}{|Y|}\right)^\frac{q(q-1)}{2} \approx 1 - e^{-\frac{q(q-1)}{2|Y|}}$$
  - Let $|Y|=2^m$, to get $\varepsilon$ to be close to 0.5, $q \approx 2^{m/2}$
  - This is known as the birthday attack.
Choosing Parameters

• The level of security (for collision resistance) of a hash function that outputs n bits, is about $n/2$ bits
  – i.e., it takes $2^{n/2}$ time to bruteforce it
  – Assuming that no better way of attacking the hash function is known

• Longer outputs often means more computation time and more communication overhead

• The level of security for encryption function using $k$-bit key is about $k$ bits
Choosing the length of Hash outputs

• **The Weakest Link Principle:**
  – A system is only as secure as its weakest link.
  – Hence all links in a system should have similar levels of security.

• Because of the birthday attack, the length of hash outputs in general should double the key length of block ciphers
  – SHA-224 matches the 112-bit strength of triple-DES (encryption 3 times using DES)
  – SHA-256, SHA-384, SHA-512 match the new key lengths (128,192,256) in AES
  – If small output size is highly important, and one is sure that collision-resistance is not needed (only one-wayness is needed), then same size should be okay.
Well Known Hash Functions

- **MD5**
  - output 128 bits
  - collision resistance completely broken by researchers in China in 2004

- **SHA1**
  - output 160 bits
  - no collision found yet, but method exist to find collisions in less than $2^{80}$
  - considered insecure for collision resistance
  - one-wayness still holds

- **SHA2 (SHA-224, SHA-256, SHA-384, SHA-512)**
  - outputs 224, 256, 384, and 512 bits, respectively
  - No real security concerns yet

- **SHA3 (224, 256, 384, 512)**
Merkle-Damgard Construction for Hash Functions

- Message is divided into fixed-size blocks and padded
- Uses a compression function $f$, which takes a chaining variable (of size of hash output) and a message block, and outputs the next chaining variable
- Final chaining variable is the hash value

$$M = m_1 m_2 \ldots m_n; \ C_0 = IV, \ C_{i+1} = f(C_i, m_i); \ H(M) = C_n$$
NIST SHA-3 Competition

• NIST completed a competition for SHA-3, the next generation of standard hash algorithms
• 2007: Request for submissions of new hash functions
• 2008: Submissions deadline. Received 64 entries. Announced first-round selections of 51 candidates.
• 2009: After First SHA-3 candidate conference in Feb, announced 14 Second Round Candidates in July.
• 2011: Public comment for final round
• 2012: October 2, NIST selected SHA3
  – Keccak (pronounced “catch-ack”) created by Guido Bertoni, Joan Daemen and Gilles Van Assche, Michaël Peeters
Each round in the input phase, the next $r$ bits of message is XOR’ed into the first $r$ bits of the state, and a function $f$ is applied to the state.

In the output phase, output $r$ bits of each round as the hash output; continue applying $f$ to get new states

- Only a portion of the state is affected in each round, and only a portion of the state is revealed in each output round

SHA-3 uses 1600 bits for state size
Limitation of Using Hash Functions for Authentication

- Require an authentic channel to transmit the hash of a message
  - Without such a channel, it is insecure, because anyone can compute the hash value of any message, as the hash function is public
  - Such a channel may not always exist

- How to address this?
  - use more than one hash functions
  - use a key to select which one to use
Hash Family

- A hash family is a four-tuple \((X, Y, K, H)\), where
  - \(X\) is a set of possible messages
  - \(Y\) is a finite set of possible message digests
  - \(K\) is the keyspace
  - For each \(K \in K\), there is a hash function \(h_K \in H\). Each \(h_K : X \rightarrow Y\)
- Alternatively, one can think of \(H\) as a function \(K \times X \rightarrow Y\)
Message Authentication Code (MAC)

- A MAC scheme is a hash family, used for message authentication
- \( \text{MAC}(K,M) = H_K(M) \)
- The sender and the receiver share secret \( K \)
- The sender sends \((M, H_K(M))\)
- The receiver receives \((X,Y)\) and verifies that \( H_K(X) = Y \), if so, then accepts the message as from the sender
- To be secure, an adversary shouldn’t be able to come up with \((X',Y')\) such that \( H_K(X') = Y' \).

**MAC:** Using a shared secret (or a limit-bandwidth confidential channel) to achieve authenticity/integrity.
Security Requirements for MAC

• Secure against the “Existential Forgery under Chosen Plaintext Attack”
  – Challenger chooses a random key $K$
  – Adversary chooses a number of messages $M_1, M_2, .., M_n$, and obtains $t_j=\text{MAC}(K, M_j)$ for $1 \leq j \leq n$
  – Adversary outputs $M'$ and $t'$
  – Adversary wins if $\forall j \ M' \neq M_j$, and $t'=\text{MAC}(K, M')$

• Basically, adversary cannot create the MAC value for a message for which it hasn’t seen an MAC
Constructing MAC from Hash Functions

• Let $h$ be a one-way hash function

• $\text{MAC}(K,M) = h(K \ || \ M)$, where $\ || \$ denote concatenation
  - Insecure as MAC with a hash function that uses the Merkle-Damgard construction:
    - given $M$ and $t=h(K \ || \ M)$, adversary can compute $M'=M||\text{Pad}(M)||X$ and $t'$, such that $h(K||M') = t'$
HMAC: Constructing MAC from Cryptographic Hash Functions

\[ \text{HMAC}_K[M] = \text{Hash}[(K^+ \oplus \text{opad}) \ || \ \text{Hash}[(K^+ \oplus \text{ipad}) || M))] \]

- \( K^+ \) is the key padded (with 0) to B bytes, the input block size of the hash function
- \( \text{ipad} = \) the byte 0x36 repeated B times
- \( \text{opad} = \) the byte 0x5C repeated B times.

At high level, \( \text{HMAC}_K[M] = H(K \ || \ H(K \ || \ M)) \)

Hash function is used twice, in nested fashion.
HMAC Security

- If used with a secure hash functions (e.g., SHA-256) and according to the specification (key size, and use correct output), no known practical attacks against HMAC
Coming Attractions …

- Cryptography: Public Key Cryptography