COMPUTER SECURITY CS 526

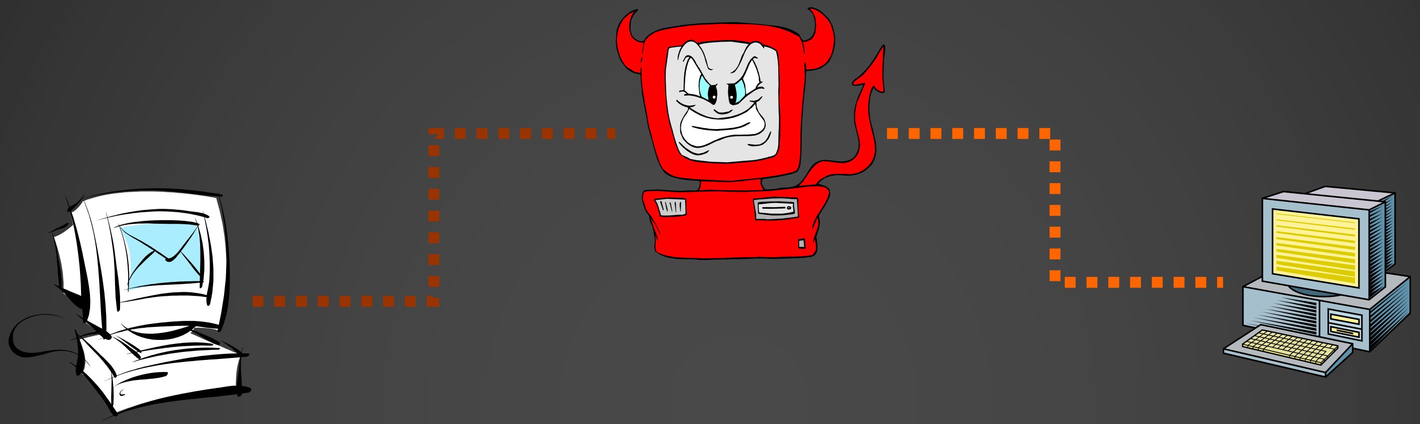
TOPIC 5

CRYPTOGRAPHY: CRYPTOGRAPHIC HASH FUNCTIONS AND MESSAGE AUTHENTICATION CODE

READINGS FOR THIS LECTURE

- Wikipedia
  - Cryptographic Hash Functions
  - Message Authentication Code

INTEGRITY AND AUTHENTICATION



- Encryption does not protect data from modification by another party.
  - Why?
- Need a way to ensure that data arrives at destination in its original form as sent by the sender and it is coming from an authenticated source.

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 secure 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 $h: X \to Y$, 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
  - How to prove that you have discovered a secret on an earlier date without disclosing it?
- 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

$$\varepsilon = 1 - \left(1 - {}^{1}\!/_{|Y|}\right)^{q} \approx \frac{q}{|Y|}$$

  - Let $|Y|=2^m$, to get ε to be close to 0.5, $q \approx 2^{m-1}$

BRUTEFORCE 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 ε to be close to 0.5, $q \approx 2^{m/2}$
  - This is known as the birthday attack.

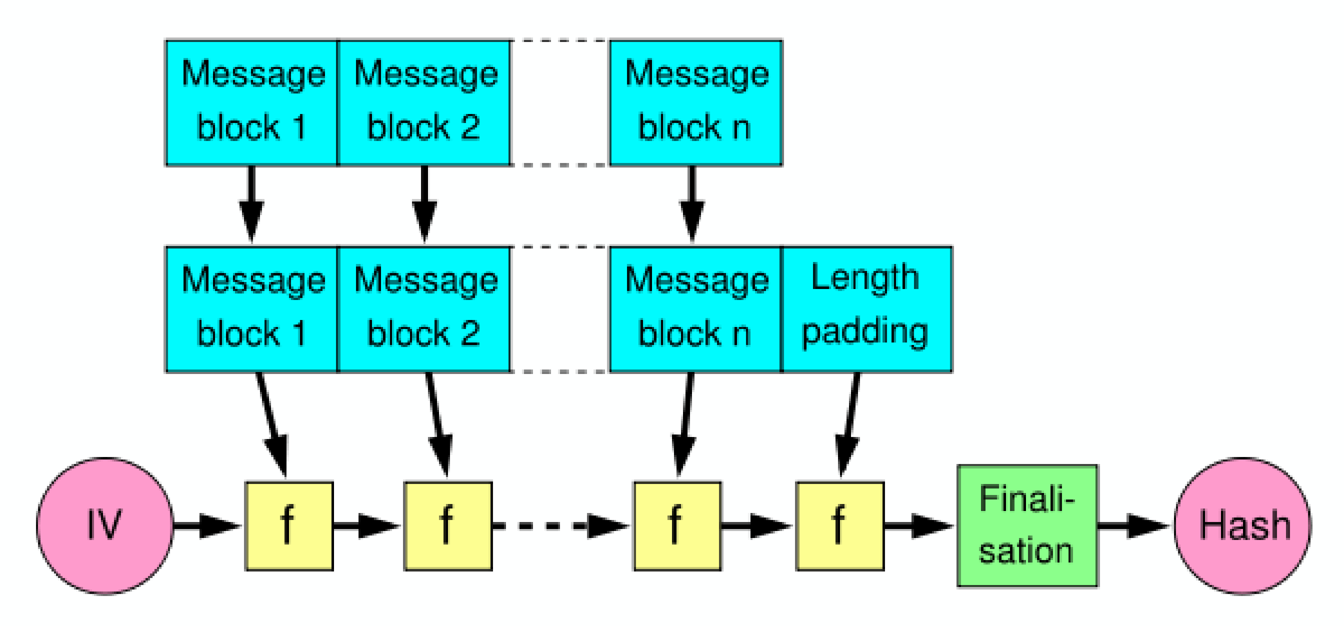
WELL KNOWN HASH FUNCTIONS

- MD5
  - output 128 bits
  - collision resistance completely broken by researchers in China in 2004
- SHA1
  - output 160 bits
  - https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html
- SHA2 (SHA-224, SHA-256, SHA-384, SHA-512)
  - outputs 224, 256, 384, and 512 bits, respectively
  - No real security concerns yet

WELL KNOWN 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

MERKLE-DAMGARD CONSTRUCTION FOR HASH FUNCTIONS



$M=m_1m_2\ldots m_n$; $C_0=IV$, $C_{i+1}=f(C_i,m_i)$; $H(M)=C_n$

NIST SHA-3 COMPETITION

- NIST is having an ongoing 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.
- 2010: After one year public review of the algorithms, hold second SHA-3 candidate conference in Aug. Announced 5 Third-round candidates in Dec.
- 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

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

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 $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, where
  - $\mathcal{X}$ is a set of possible messages
  - $\mathcal{Y}$ is a finite set of possible message digests
  - $\mathcal{K}$ is the keyspace
  - For each $K \in \mathcal{K}$, there is a hash function $h_K \in \mathcal{H}$. Each h_K: $\mathcal{X} \to \mathcal{Y}$
- Alternatively, one can think of $\mathcal{H}$ as a function $\mathcal{K} \times \mathcal{X} \to \mathcal{Y}$

MESSAGE AUTHENTICATION CODE

- A MAC scheme is a hash family, used for message authentication
- $MAC(K,M) = H_K(M)$
- 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 $H_K(X')=Y'$.

SECURITY REQUIREMENTS FOR MAC

- Resist 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=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'=MAC(K,M')

- Basically, adversary cannot create the MAC for a message for which it hasn't seen an MAC

CONSTRUCTING MAC FROM HASH FUNCTIONS

- Let h be a one-way hash function

- MAC(K,M) = h(K || M), where || denote concatenation
  - Insecure as MAC
  - Because of the Merkle-Damgard construction for hash functions, given M and t=h(K || M), adversary can compute M’=M||Pad(M)||X and t’, such that h(K||M’) = t’

HMAC: CONSTRUCTING MAC FROM CRYPTOGRAPHIC HASH FUNCTIONS

$$HMAC_K[M] = Hash[(K^+ \oplus opad) \,||\, Hash[(K^+ \oplus ipad)||M)]]$$

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

At high level, $HMAC_K[M] = H(K \,||\, H(K \,||\, M))$

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

NEXT CLASS

- Cryptography: Public Key Cryptography