CS408
Cryptography & Internet Security

Lecture 18:

Cryptographic hash functions,

Message authentication codes

Functions

Definition

Given two sets, X and Y, a function f : X → Y (from set X to set Y), is a relation which uniquely associates members of set X with members of set Y.

Terminology

X is called domain

Y is called range, image, or co-domain.

For y = f(x) where x ∈ X and y ∈ Y, y is called the image of x and x is called the pre-image of y.

Cryptographic Hash Functions

- Takes as input a string of any size and outputs a fixed-size string (usually output is much smaller than input)
  - E.g., output can be 160 bits regardless of input size
- A hash is a many-to-one function, so **collisions can happen (but should be unlikely to happen)**.
- Two fundamental properties: **compression and easy to compute**.

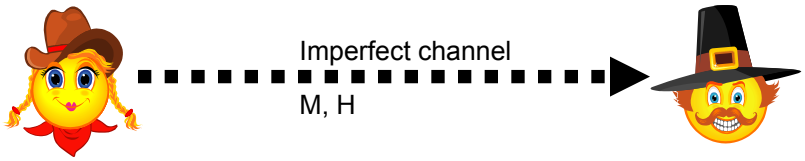
Cryptographic Hash Functions (continued)

- Informal requirements
  - One-way (non-invertible)
  - Produces different outputs for different inputs (with high likelihood)

Cryptographic Hash Functions (continued)

- Formally:
  - *First pre-image resistance*: given h(x), cannot find x
  - *Second pre-image resistance*: given x and h(x), cannot find y ≠x s.t. h(y) = h(x)
  - *Collision resistance*: cannot find any pair x,y, with x≠y s.t. h(x) = h(y)
- It is an unkeyed cryptographic primitive (publicly computable, no secret involved)
- Examples: SHA-1 (160 bits output), SHA-256 (256 bits output), SHA-512 (512 bits output), MD5 (128 bits output)

Data Integrity with Hash Functions



- Let h be a cryptographic hash function
- Alice computes H = h(M)
- Alice sends to Bob M and H
- Bob receives M, H, computes H1 = h(M) and checks if H1 = H
- If the check is true, then Bob accepts message; otherwise, reject message
- Why does this guarantee integrity?
  - Because of the second pre-image resistance property of h ! Given M, h(M), cannot find another M' s.t. h(M')=h(M).
  - This only provides integrity for a benign channel that can corrupt bits

Birthday Paradox

- What is the probability that that in a set of n randomly chosen people, two people have the same birthday?
- For a group of 23 people, the probability that two people have the same birthday is 50% !
- For a group of 57 people, the probability that two people have the same birthday is 99% !

Birthday Attack on Collision Resistance

- Goal: break collision resistance (find a collision)
- Let h be a hash function with the size of the output of m bits
- Birthday attack runs in $O(2^{m/2})$ and works against all the unkeyed hash functions
- Because of the birthday attack, the length of hash outputs in general should double the key length of block ciphers
  - SHA-256, SHA-384, SHA-512 to match the key lengths (128,192,256) in AES

Which Hash Function to Use?

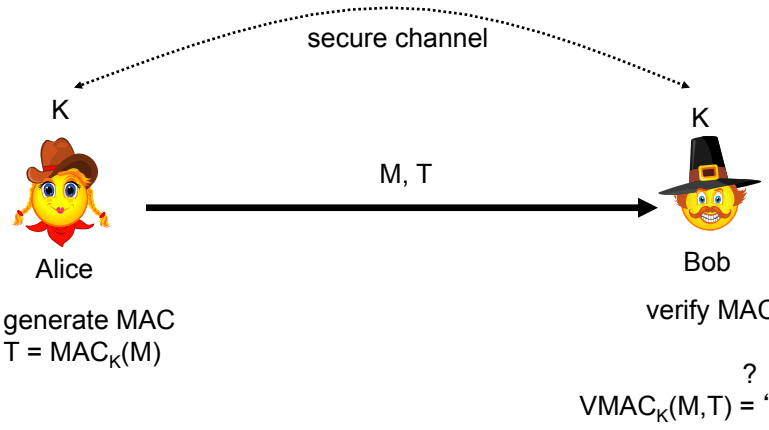
- In 2004, MD5 was shown not to be collision resistant
  - Attack was subsequently improved between 2005-2007
- Thus, MD5 should not be used if the goal is to have collision resistance
- Instead, the SHA-2 family of hash functions (SHA-256, SHA-384, SHA-512) is recommended

Why Hash is Not Enough?

- Hash functions can provide data integrity, but no indication about where is data coming from or who generated the hash output (hash function is public)
- Data source authentication (also referred as message authentication) is needed, otherwise anybody can inject traffic
- Mechanism? Involve a secret key

Message Authentication Code (MAC)

- Alice and Bob already have a trust relationship (i.e., they share a secret key K) (the dotted line)
- Then, they can exchange messages "securely"



Message Authentication Code (continued)

- (informal) Requirements for MAC:
  - Involve a secret key
  - Computation is easy if secret key k is known
  - Similar to hash functions requirements:
    - Compression: M has n bits, $MAC_k(M)$ has fixed length m bits, m < n
    - Knowing a message and its MAC, it is infeasible to find another message with same MAC
    - Unforgeability: Given a valid MAC on a message, it is infeasible to find another valid MAC (on a different message), without knowing K:
      Given $(M_1, MAC_K(M_1))$, it is hard to find $(M_2, MAC_K(M_2))$, with $M_1 \neq M_2$
    - MACs should be uniformly distributed
    - MAC should depend equally on all bits of the message

Message Authentication Code (continued)

- A message authentication code is a collection of three algorithms (G, MAC, VMAC)
  - Key generation algorithm G: generates a key K
  - Authentication tag generation algorithm MAC: $T = MAC_K(M)$
  - Authentication tag verification algorithm VMAC: "result" = $VMAC_K(M,T)$, where "result" is "valid" or "invalid"
- The following should always hold true:
  - $VMAC_K(M,T)$ = "valid", if $T = MAC_K(M)$
    = "invalid", otherwise
- Provides: authentication, integrity
- Does not provide: confidentiality, non-repudiation
- It is a keyed cryptographic primitive
- Example: HMAC-SHA1, HMAC-SHA256

Keyed Hash Functions as MACs

- Create a MAC using a hash function
- Uses a public hash function and a secret symmetric key
- Current standard is HMAC, specified in FIPS 198 (2002)

HMAC (Hash-based Message Authentication Code)

- Let h be a cryptographic hash function

(Simplified) definition of HMAC:

$HMAC_K(m) = h(K \| h(K \| m))$

(Full) definition of HMAC:

$HMAC_K(m) = h((K^+ \oplus opad) \| h((K^+ \oplus ipad) \| m))$

where:

- || denotes concatenation
- opad and ipad are fixed, public strings
- K^+ is the key padded with extra 0's to the input block size of the hash function
  - A hash function also has a input block size (similar with block ciphers)

HMAC Security

- Security of HMAC depends on the security of the underlying hash function
  - this has been formally proven
- What is the output length of HMAC?
  - It depends on which hash function is used
  - Is the same as the output length of the underlying hash function
- If used with a secure hash function (like SHA1) and according to the specification (key size, and use correct output), there are no known practical attacks against HMAC
- If HMAC is used with SHA1, it is referred to as *HMAC-SHA1*

What About Integrity of Communication in a Non-malicious Environment?

- Goal: protect against accidental or non-malicious errors on noisy channels subject to transmission errors
  - This is different than the insecure channel we have been considering so far
- Error detection codes and error correction codes
- NOTE: with these methods, the requirement is different and anybody can forge packets
  - Why?
- Methods:
  - Checksum
  - CRC (cyclic redundancy codes)