Digest functions

Digest functions

Give a fixed-length value from a variable-length text

- Sort of text "fingerprint"

Produce very different values for similar texts

- Cryptographic one-way hash functions

Relevant properties:

- Preimage resistance
  - Given a digest, it is unfeasible to find an original text producing it
- 2nd-preimage resistance
  - Given a text, it is unfeasible to find another one with the same digest
- Collision resistance
  - It is unfeasible to find any two texts with the same digest
  - Birthday paradox

Digest functions: size

Considering the similar, yet different texts:

- T1: “Hello User_A!”
- T2:“Hello User_XY!”

Different algorithms will result in values with different dimension, but independent of the dimension of the text

- MD5 (128 bits):
  - T1: `70df836fdaf02e0dfc990f9139762541`
  - T2: `a08313b553d8bf53ca7457601a361bea`
- SHA-1 (160 bits):
  - T1:`f591aa1eabcc97fb39c5f422b370ddf8cb880fde`
  - T2:`c28b0520311e471200b397eaa55f1689c8866f25`
- SHA-256 (256 bits):
  - T1: `9649d8c0d25515a239ec8ec94b293c8868e931ad318df4ccd0dffd67aff89905`
  - T2: `8fc49cde23d15f8b9b1195962e9ba517116f45661916a0f199fcf21cb686d852`

Digest functions: content

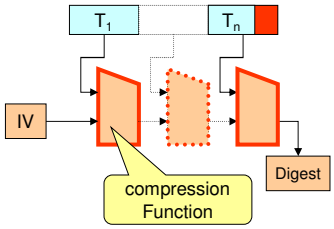
Considering the similar, yet different texts:

- T1: “Hello User_B!”
- T2: “Hello User_C!”

A small change in the text (1 bit) results in a completely different result

- MD5:
  - T1: `c32e0f62a7c9c815063d373acac80c37`
  - T2: `324a1bfc3041259480c6ad164cf0529f`
- SHA-1:
  - T1: `bab31eb62f961266758524071a7ad8221bc8700b`
  - T2: `bd758d82899d132cd2af66dc3402b948d98de62d`
- SHA-256:
  - T1: `e663a01d3bec4f35a470aba4baccece79bf484b5d0bffa88b59a9bb08707758a`
  - T2: `69f78345da90c6b8d4785b769cd6ae09e0531716fe5f5a392fde1bdc70a2bb7d`

Digest functions



Approaches

- Merkle-Damgård construction
  - Collision-resistant, one-way compression functions
  - Iterative compression
  - Length padding
- Sponge functions

Most common algorithms

- MD5 (128 bits)
  - No longer secure! It's easy to find collisions!
- SHA-1 (Secure Hash Algorithm, 160 bits)
  - Also no longer secure … (collisions found in 2017)
- SHA-2, aka SHA-256/SHA-512
- SHA-3

Message Integrity Code (MIC)

Provide the capability to detect changes by devices

- Communication/storage errors
- From a random process or without control

Send: Calculate MIC and send T + MIC

- T = Text
- MIC = digest(T)

Receive: Receive data (T') and check if H(T) = MIC

- Calculate MIC'=digest(T')
- Validate if MIC' = MIC

Doesn't protect from planned changes to the text

- Attacker can manipulate T into T'' and calculate a new MIC''

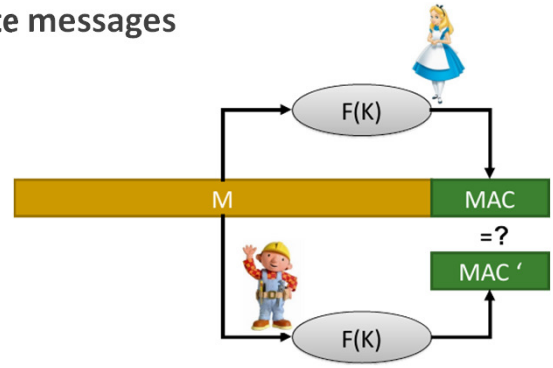
Message Authentication Code (MAC)

MIC computed with a key

- Only key holders can generate/validate the MAC

Used to authenticate messages

- M' = M | MAC(M)



MAC: approaches

Encryption of an ordinary digest

- Using, for instance, a symmetric block cipher

Using encryption with feedback & error propagation

- CBC-MAC

Adding a key to the hashed data

- Keyed-MD5 (128 bits)
  - MD5(K, keyfill, text, K, MD5fill)
- HMAC (output length depends on the function H used)
  - H(K, opad, H(K, ipad, text))
  - ipad = 0x36 B times opad = 0x5C B times B = size of H input block
    - HMAC-MD5, HMAC-SHA-1, etc.

Encryption + Authentication

Encrypt-then-MAC: MAC is computed from cryptogram
- Allows verifying integrity before (the longer) decryption

BAD

Encrypt-and-MAC: MAC is computed from plaintext
- MAC is not encrypted
- May give information regarding original text (if similar to other)

MAC-then-Encrypt: MAC is computed from plaintext
- MAC is encrypted
- Requires full decryption before MAC is validated

Example: GCM (Galois Counter Mode)