

Cryptographic Hashing

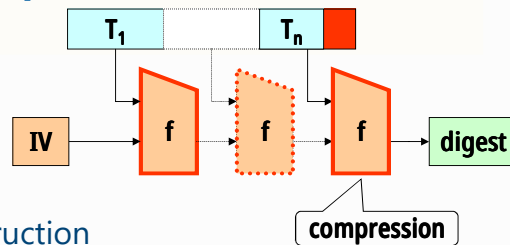


Digest functions

- ▷ 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 → collisions exist
- ▷ Relevant properties:
 - ♦ **Preimage resistance**
 - Given a digest, it is infeasible to find an original text producing it
 - ♦ **2nd-preimage resistance**
 - Given a text, it is infeasible to find another one with the same digest
 - ♦ **Collision resistance** → can happen but difficult
 - It is infeasible to find any two texts with the same digest
 - Birthday paradox



Digest functions: approaches



▷ Merkle-Damgård construction

- Iterative compression
- Collision-resistant, one-way compression functions
- Length padding (1, followed by zeros, followed by length)

▷ Sponge functions

- **Absorption**: update a finite internal state (entropy pool) from a variable-length, padded input stream
- **Squeezing**: produce an arbitrary-length output from the internal state



© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

3

Digest functions: common algorithms

▷ MD5 (128 bits)

- No longer secure! It's easy to find collisions!
- Disclaimer: it can be used when collisions are not an issue

▷ SHA-1 (Secure Hash Algorithm, 160 bits)

- Also no longer secure ... (collisions found in 2017)

▷ RIPEMD (128 and 160)

▷ SHA-2, aka SHA-256 / SHA-384 / SHA-512

▷ SHA-3 (Keccak)



© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

4

Rainbow tables

- ▷ We can invert a digest function with a table
 - ♦ For all possible input, we compute and store the digest
 - ♦ But the table size is given by the digest length
 - Not usually applicable
- ▷ Solution: rainbow tables
 - ♦ Trade space with time
 - ♦ Store only part of the outputs
 - For direct matching
 - ♦ Find for more matches using computation



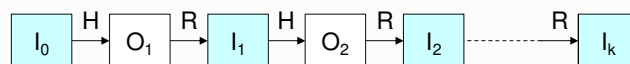
© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

5

Rainbow tables

- ▷ They are based on a reverse function R
 - ♦ Which is not the inverse of H
 - ♦ The goal of R is to produce a new input given a hashing result



- ▷ R functions are likely to produce collisions
 - ♦ But we can use many different R functions
 - ♦ Collisions can still occur
 - But will not create a problem unless occurring at the exact same column
 - And that case can be identified (and discarded) by identical outputs
- ▷ A table with m k -length rows can invert $k \times m$ hashes
 - ♦ At most
 - ♦ Only I_0 and I_k is stored per row



© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

6

	H	R	R ₂
I ₁	O ₁	I' ₁	
I ₂	O ₂	I	
I ₃	O ₃	I	I'' ₃

Rainbow tables: exploitation

- ▷ A set of m random inputs is generated
 - ♦ $I_0 = \{I_{0,1}, \dots, I_{0,m}\}$
- ▷ A set of m k -length chain outputs is computed
 - ♦ $I_k = \{I_{k,1}, \dots, I_{k,m}\}$
- ▷ Given a target o
 - ♦ Look for $R(o)$ in I_k
 - ♦ If found in row r , compute chain from $I_{0,r}$
 - until finding i such that $H(i) = o$
 - ♦ If not found, compute o_r from o using H and R for each row r
 - and see if $o_r = I_{k,r}$
 - H and R are applied 1 to k times, using different R functions



© André Zúquete /
Tomás Oliveira e Silva

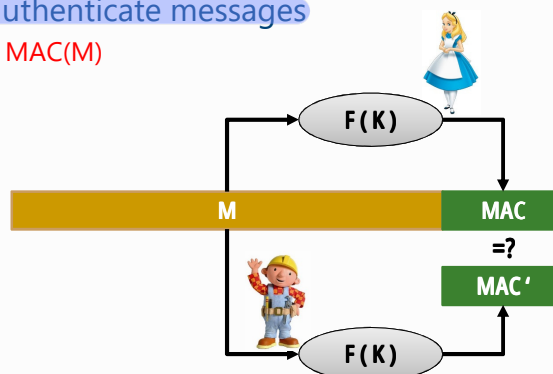
Applied Cryptography

7

Do Digest by itself it's not secure anyone can do it. It needs a key

Message Authentication Codes (MAC)

- ▷ Hash, or digest, computed with a key
 - ♦ Only key holders can generate and validate the MAC
- ▷ Used to authenticate messages
 - ♦ $M' = M \parallel \text{MAC}(M)$



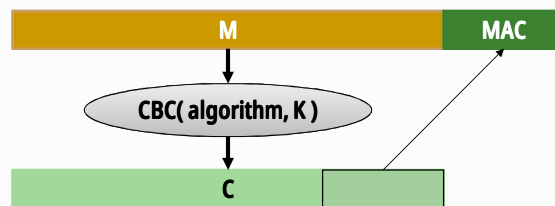
© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

8

Message Authentication Codes (MAC): Approaches

- ▷ Encryption of an ordinary digest
 - Using, for instance, a symmetric block cipher
- ▷ Using encryption with feedback & error propagation
 - ANSI X9.9 (or DES-MAC) with DES CBC (64 bits)
 - CBC-MAC



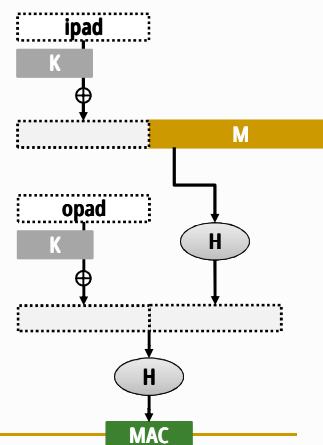
© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

9

Message Authentication Codes (MAC): Approaches

- ▷ Adding a key to the hashed data
 - Keyed-MD5 (128 bits)
 - $\text{MD5}(K, \text{keyfill}, \text{text}, K, \text{MD5fill})$
 - HMAC (Hashed-based MAC)
 - Generic construction, uses a hash function H
 - Output length depends on H
 - HMAC-MD5, HMAC-SHA, etc.
- $H(K, \text{opad}, H(K, \text{ipad}, \text{text}))$
- $\text{ipad} = 0x36$ B times
- $\text{opad} = 0x5C$ B times



© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

10

Authenticated encryption

▷ Encryption mixed with integrity control

- ♦ Error propagation
- ♦ Authentication tags

▷ Examples

- ♦ GCM (Galois/Counter Mode)
- ♦ CCM (Counter with CBC-MAC)

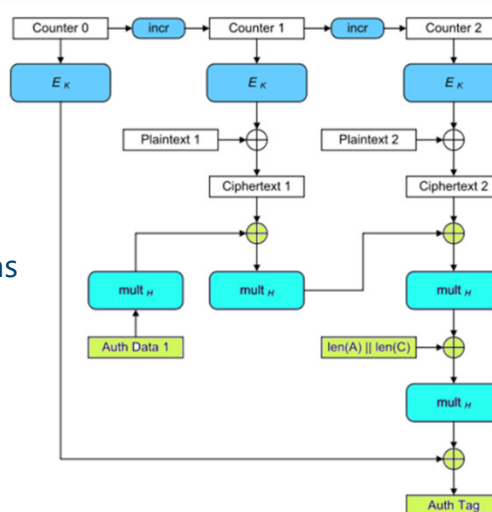


GCM

▷ CTR mode encryption

▷ Successive multiplications for integrity control

- ♦ Multiplications in $GF(2^n)$
- ♦ $H = E_k(0)$



Encryption + authentication

Das ist die best

▷ Encrypt-then-MAC

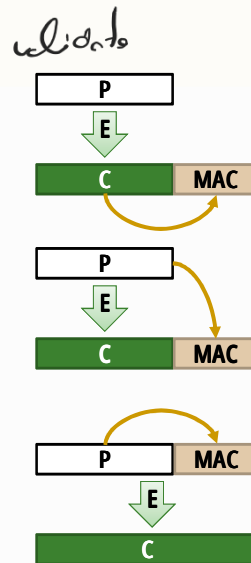
- MAC is computed from cryptogram
- Should use two different keys
- IPSec uses it
- No parallel computation

▷ Encrypt-and-MAC

- MAC is computed from plaintext
- MAC is not encrypted
- SSH uses it

▷ MAC-then-Encrypt

- MAC is computed from plaintext
- MAC is encrypted
- TLS uses it



© André Zúquete /
Tomás Oliveira e Silva

Applied Cryptography

13

