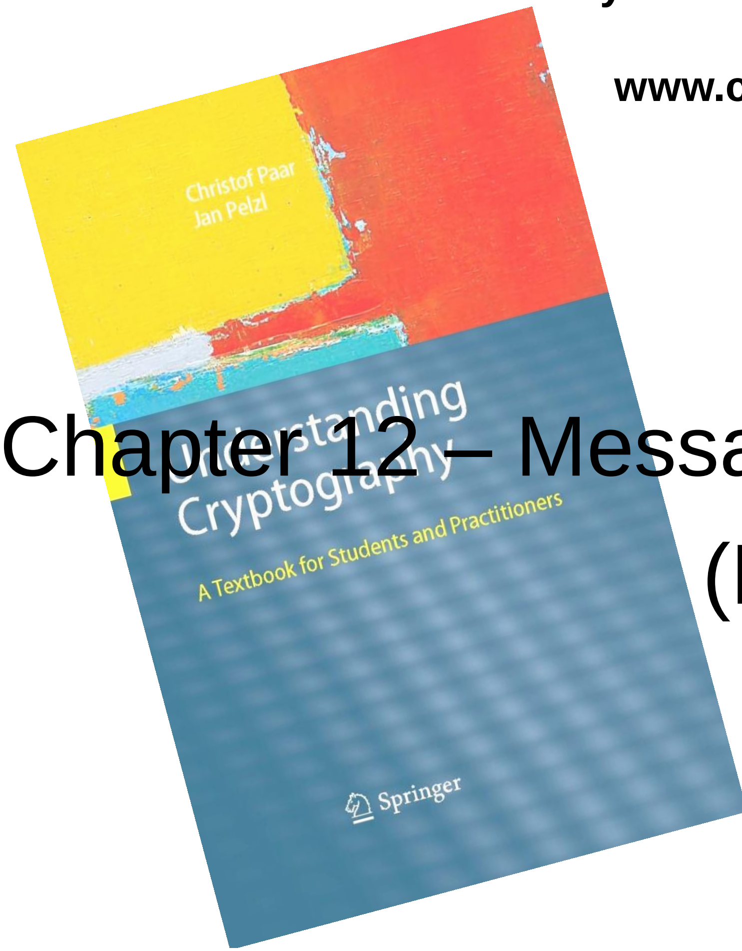
Understanding Cryptography

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 12 – Message Authentication Codes (MACs)

These slides were prepared by Christof Paar and Jan Pelzl

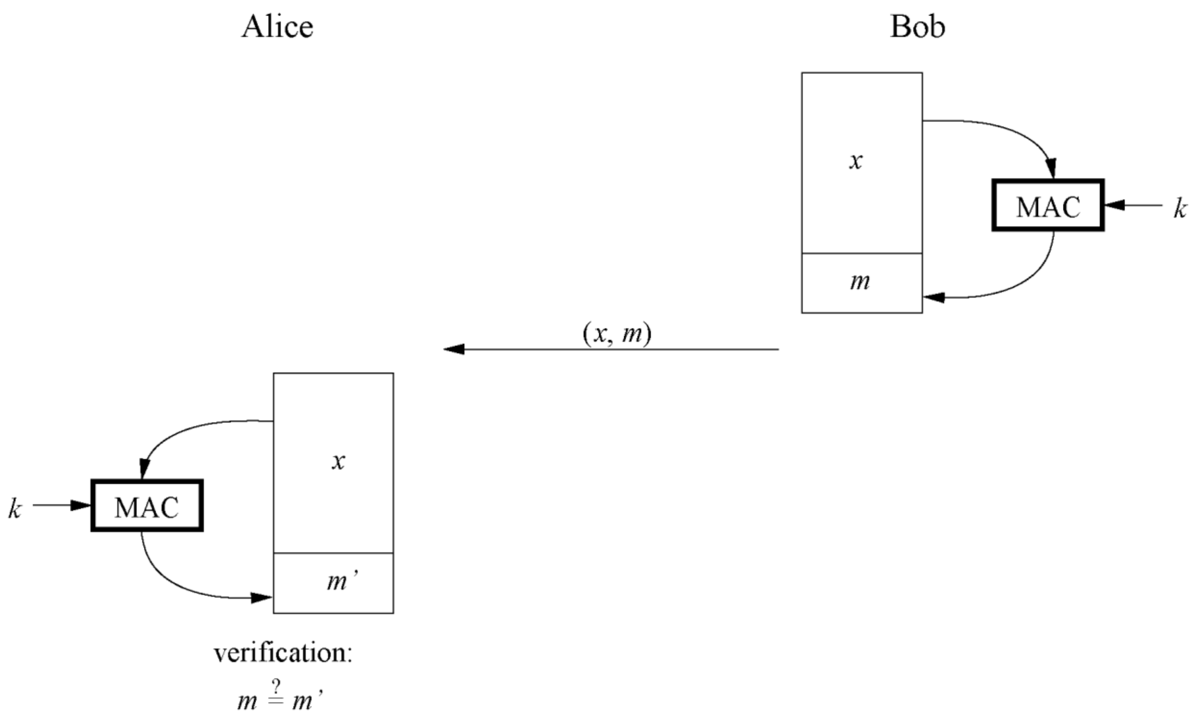
Some legal stuff (sorry): Terms of Use

Content of this Chapter

- The principle behind MACs
- The security properties that can be achieved with MACs
- How MACs can be realized with hash functions and with block ciphers

Principle of Message Authentication Codes

- Similar to digital signatures, MACs append an authentication tag to a message
- MACs use a symmetric key k for generation and verification
- Computation of a MAC: $m = \mathrm{MAC}_k(x)$



Properties of Message Authentication Codes

1. **Cryptographic checksum**

   A MAC generates a cryptographically secure authentication tag for a given message.

2. **Symmetric**

   MACs are based on secret symmetric keys. The signing and verifying parties must share a secret key.

3. **Arbitrary message size**

   MACs accept messages of arbitrary length.

4. **Fixed output length**

   MACs generate fixed-size authentication tags.

5. **Message integrity**

   MACs providemessage integrity: Any manipulations of a message during transit will be detected by the receiver.

6. **Message authentication**

   The receiving party is assured of the origin of the message.

7. **No nonrepudiation**

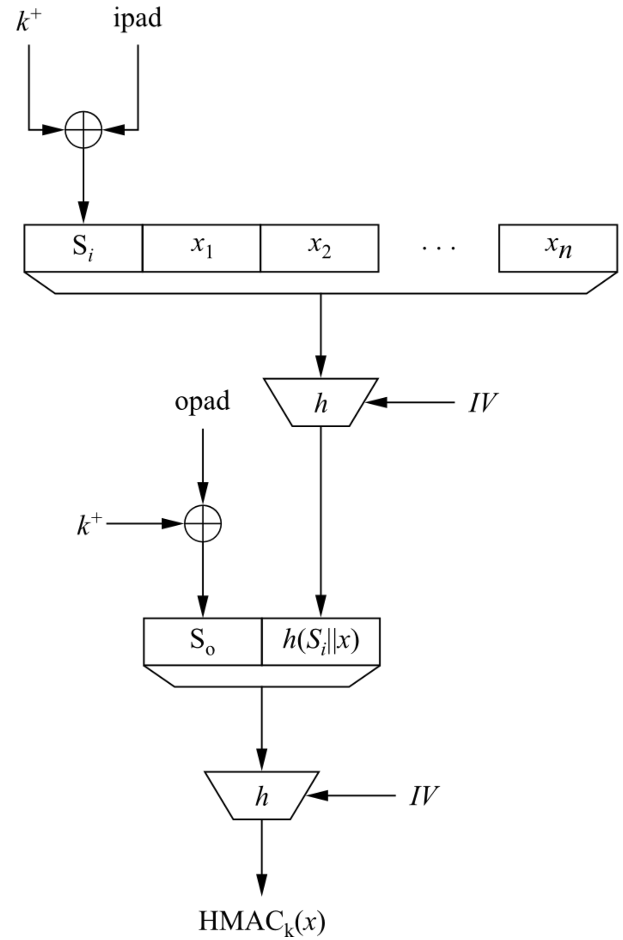
   Since MACs are based on symmetric principles, they do not provide nonrepudiation.

MACs from Hash Functions

- MAC is realized with cryptographic hash functions (e.g., SHA-1)
- HMAC is such a MAC built from hash functions
- Basic idea: Key is hashed together with the message
- Two possible constructions:
  - secret prefix MAC: $m = MAC_k(x) = h(k||x)$
  - secret suffix MAC: $m = MAC_k(x) = h(x||k)$
- Attacks:
  - secret prefix MAC: Attack MAC for the message $x = (x_1, x_2, \ldots, x_n, x_{n+1})$, *where* x_{n+1} is an arbitrary additional block, can be constructed from *m without knowing the* secret key
  - secret suffix MAC: find collision x and x_O such that $h(x) = h(x_O)$, then $m = h(x||k) = h(x_O||k)$
- Idea: Combine secret prefix and suffix: HMAC (cf. next slide)

HMAC

- Proposed by Mihir Bellare, Ran Canetti and Hugo Krawczyk in 1996
- Scheme consists of an inner and outer hash
  - k^+ is expanded key k
  - expanded key k^+ is XORed with the inner pad
  - ipad = 00110110,00110110, . . .,00110110
  - opad = 01011100,01011100, . . .,01011100
  - $HMAC_k(x) = h[(k^+ \oplus opad) || h[(k^+ \oplus ipad) || x]]$

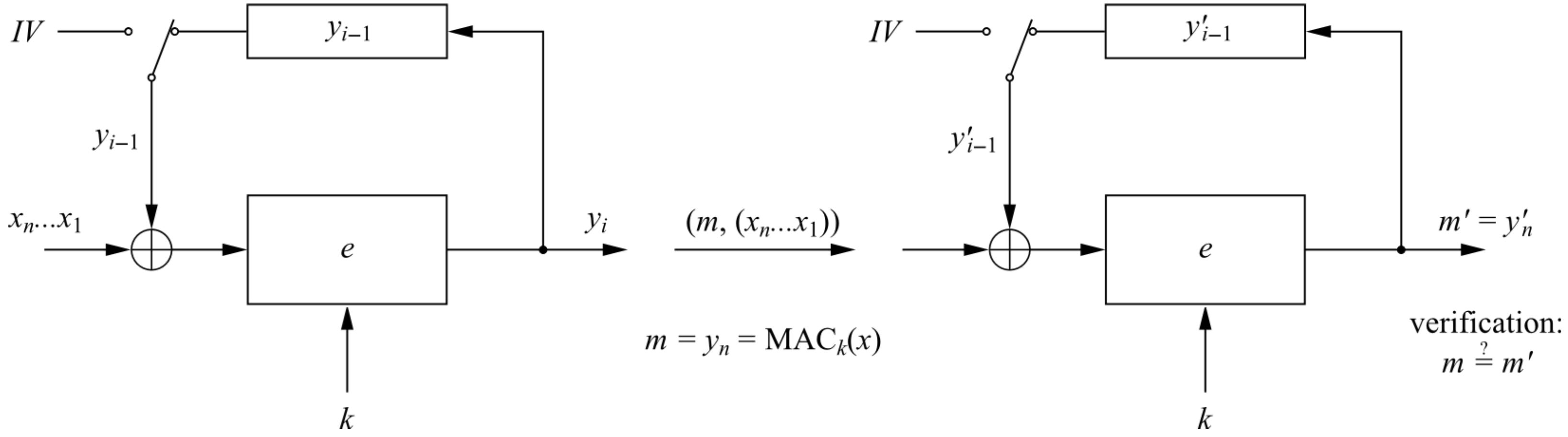


- HMAC is provable secure which means (informally speaking): The MAC can only be broken if a collision for the hash function can be found.

MACs from Block Ciphers

- MAC constructed from block ciphers (e.g. AES)
- Popular: Use AES in CBC mode

- CBC-MAC:



Chapter 12 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

CBC-MAC

- MAC Generation
  - Divide the message x into blocks x_i
  - Compute first iteration $y_1 = e_k(x_1 \oplus IV)$
  - Compute $y_i = e_k(x_i \oplus y_{i-1})$ for the next blocks
  - Final block is the MAC value: $m = \mathrm{MAC}_k(x) = y_n$

- MAC Verification
  - Repeat MAC computation (m')
  - Compare results:In case $m' = m$, the message is verified as correct
  - In case $m' \neq m$, the message and/or the MAC value m have been altered during transmission

Lessons Learned

- MACs provide two security services, *message integrity and message authentication,* using symmetric techniques. MACs are widely used in protocols.
- Both of these services also provided by digital signatures, but MACs are much faster as they are based on symmetric algorithms.
- MACs do not provide nonrepudiation.
- In practice, MACs are either based on block ciphers or on hash functions.
- HMAC is a popular and very secure MAC, used in many practical protocols such as TLS.