Chapters 12

Message Authentication Codes (MACs)

Dr. Shin-Ming Cheng

NTUST
CONNECTIVITY LAB

CS4003701

Introduction to Information Security

Content of this Chapter

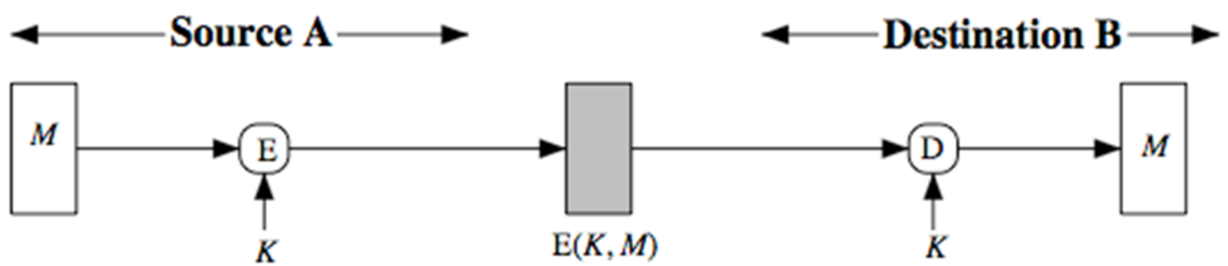
- The principle behind MACs
- The security properties that can be achieved with MACs
- How MACs can be realized with hash functions and with block ciphers

Message Authentication

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- will consider the security requirements
- Three alternative functions
  - hash function
  - message encryption
  - message authentication code (MAC)

Symmetric Message Encryption

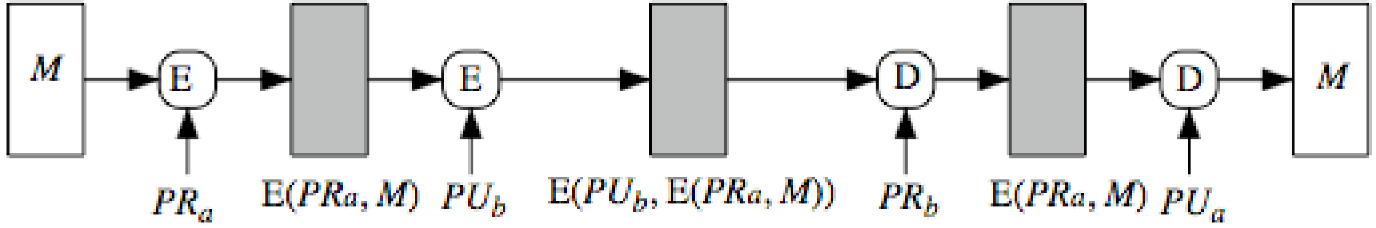
- encryption can also provides authentication
- if symmetric encryption is used then:
  - receiver know sender must have created it
  - since only sender and receiver now key used
  - know content cannot of been altered
  - if message has suitable structure, redundancy or a checksum to detect any changes



(a) Symmetric encryption: confidentiality and authentication

Public-Key Message Encryption

- if public-key encryption is used:
  - encryption provides no confidence of sender
    - since anyone potentially knows public-key
  - however if
    - sender signs message using their private-key
    - then encrypts with recipients public key
    - have both secrecy and authentication
  - again need to recognize corrupted messages
  - but at cost of two public-key uses on message



(d) Public-key encryption: confidentiality, authentication, and signature

Principle of MACs

› generated by an algorithm that creates a small fixed-sized block
  – depending on both message and some key
  – like encryption though need not be reversible

› appended to message as a signature

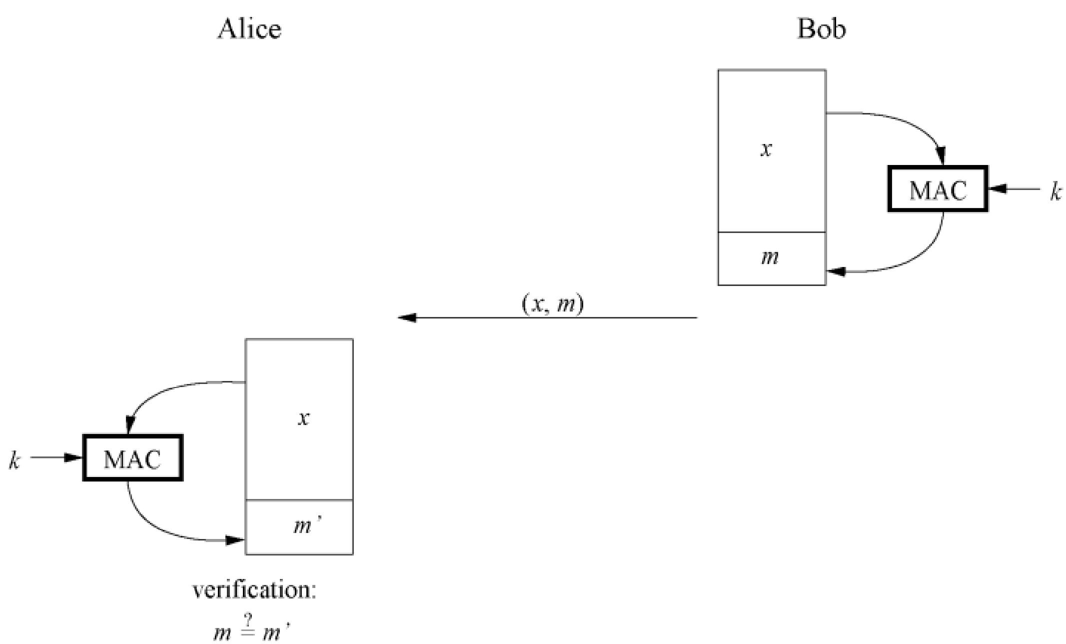
› receiver performs same computation on message and checks it matches the MAC

› provides assurance that message is unaltered and comes from sender

Principle of MACs

› a small fixed-sized block of data

- MACs use a symmetric key k for generation and verification
- Computation of a MAC: $m = MAC_k(x)$
- Appended to message when sent



Principle of MACs

- MAC provides authentication
- MAC can also use encryption for secrecy
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before
- Why use a MAC?
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption
- MAC is not a digital signature

Properties of MACs

- Cryptographic checksum
  - A MAC generates a cryptographically secure authentication tag for a given message.
- Symmetric
  - MACs are based on secret symmetric keys. The signing and verifying parties must share a secret key.
- Arbitrary message size
  - MACs accept messages of arbitrary length.
- Fixed output length
  - MACs generate fixed-size authentication tags.

Properties of MACs

- A many-to-one function
  - potentially many messages have same MAC
  - but finding these needs to be very difficult
- Message integrity
  - Any manipulations of a message during transit will be detected by the receiver.
- Message authentication
  - The receiving party is assured of the origin of the message.
- No nonrepudiation
  - Since MACs are based on symmetric principles, they do not provide nonrepudiation.

Requirements for MACs

- knowing a message and MAC, is infeasible to find another message with same MAC
- MACs should be uniformly distributed
- MAC should depend equally on all bits of the message

Types of MACs

- There are various types of MAC schemes
  - MACs based on block ciphers in CBC mode
  - MACs based on MDCs (hash functions)
  - Customized MACs
- Most widely used by far are the CBC-MACs
- CBC-MACs in various international standards
  - US Banking standards ANSI X9.9, ANSI X9.19
  - Specify CBC-MACs, date back to early 1980s
  - The ISO version is ISO 8731-1: 1987
  - Above standards specify DES in CBC mode to produce a MAC

MACs from Hash Functions

- MAC is realized with cryptographic hash functions
  - hash functions are generally faster
  - crypto hash function code is widely available
- Basic idea: Key is hashed together with the message
- Secret prefix MAC

$$m = MAC_k(x) = h(k||x)$$

- Secret suffix MAC

$$m = MAC_k(x) = h(x||k)$$

MACs from Hash Functions

- Envelope method with padding:

$$m = MAC_k(x) \;=\; h(\, k \| p \| x \| k \,)$$

  - p is a string used to pad k to the length of one block

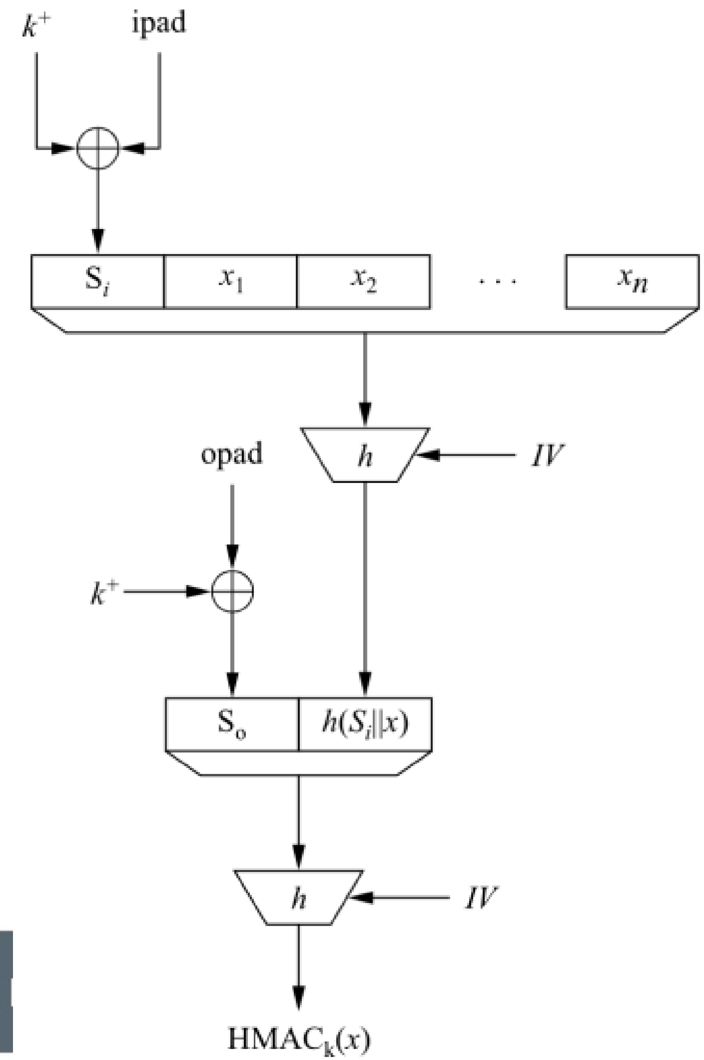
- HMAC

$$m = HMAC_k(x) \;=\; h(\, k \| p_1 \| h(\, k \| p_2 \| x \,))$$

  - with p_1, p_2 fixed strings used to pad k to full block
  - Proposed by Mihir Bellare, Ran Canetti and Hugo Krawczyk in 1996
  - any hash function can be used
    - MD5, SHA-1, RIPEMD-160, Whirlpool

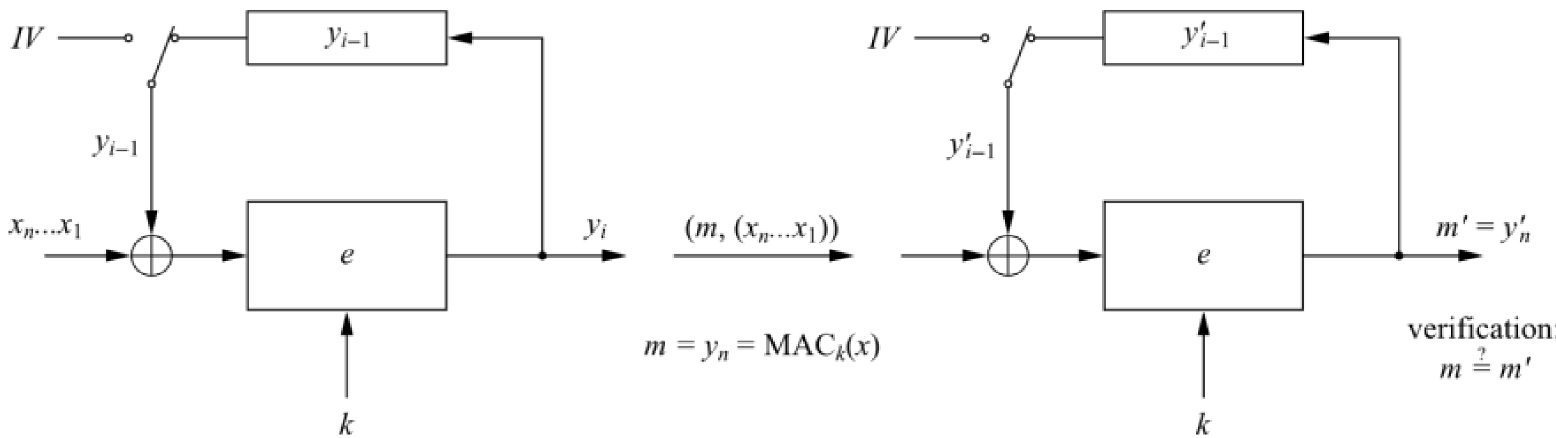
HMAC

- 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
  - $HMAC_k(x) = h[(k^+ \oplus opad)||h[(k^+ \oplus ipad)||x]]$



MACs from Block Ciphers

- can use any block cipher chaining mode and use final block as a MAC
- Operations of CBC-MAC

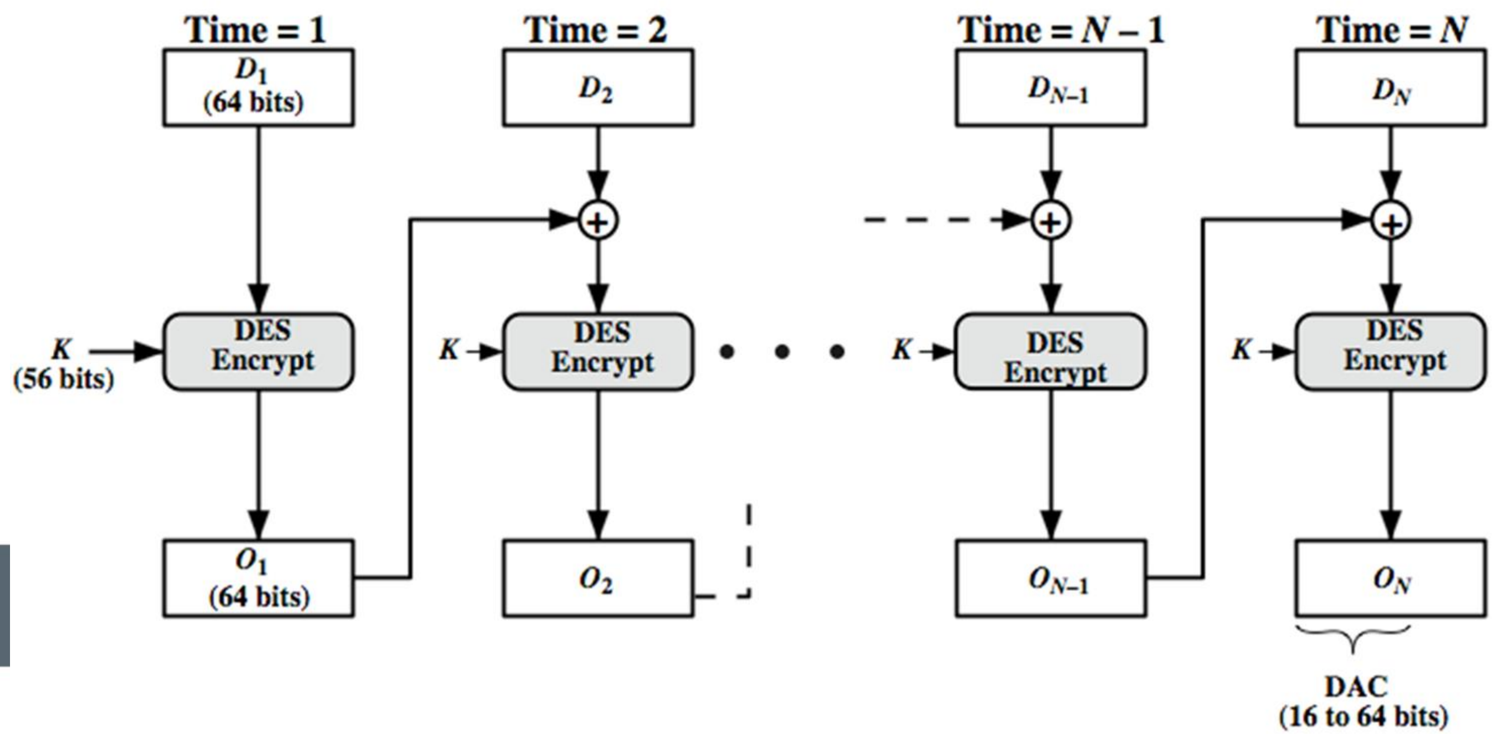


CBC-MAC

- 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: $m = MAC_k(x) = y_n$
- 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

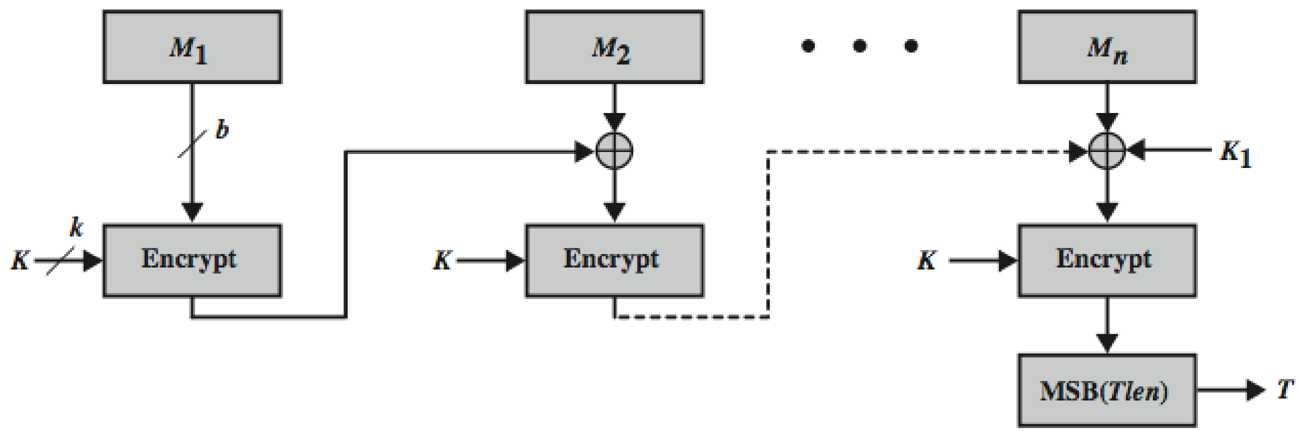
Data Authentication Algorithm

- using $IV = 0$ and zero-pad of final block
- encrypt message using DES in CBC mode
- and send just the final block as the MAC
  - or the leftmost M bits $(16 \le M \le 64)$ of final block

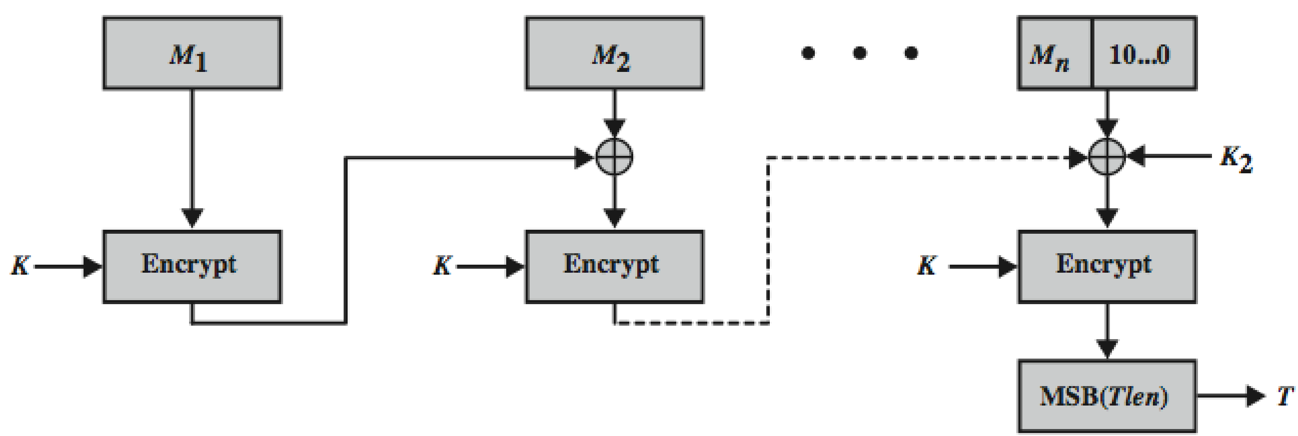


Cipher-based Message Authentication Code (CMAC)

- can overcome using 2 keys & padding



(a) Message length is integer multiple of block size



(b) Message length is not integer multiple of block size

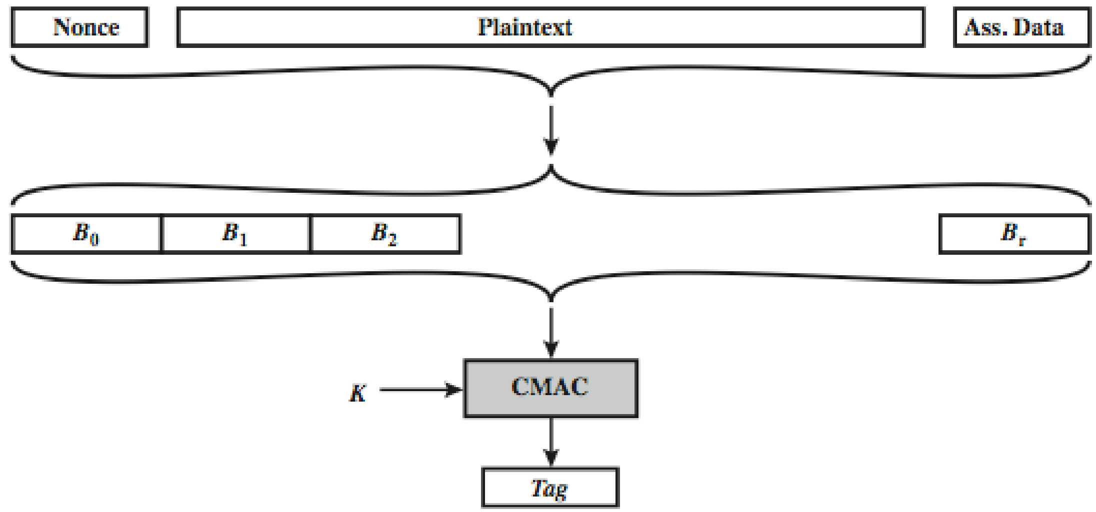
Authenticated Encryption

- Simultaneously protect confidentiality and authenticity of communications
  - often required but usually separate
- Approaches
  - Hash-then-encrypt: $E(k, (x||H(x))$
  - MAC-then-encrypt: $E(k_2, (x||MAC(k_1, x))$
  - Encrypt-then-MAC: $(C = E(k_2, x), m = MAC(k_1, C)$
  - Encrypt-and-MAC: $(C = E(k_2, x), m = MAC(k_1, x)$
- Decryption/verification straightforward

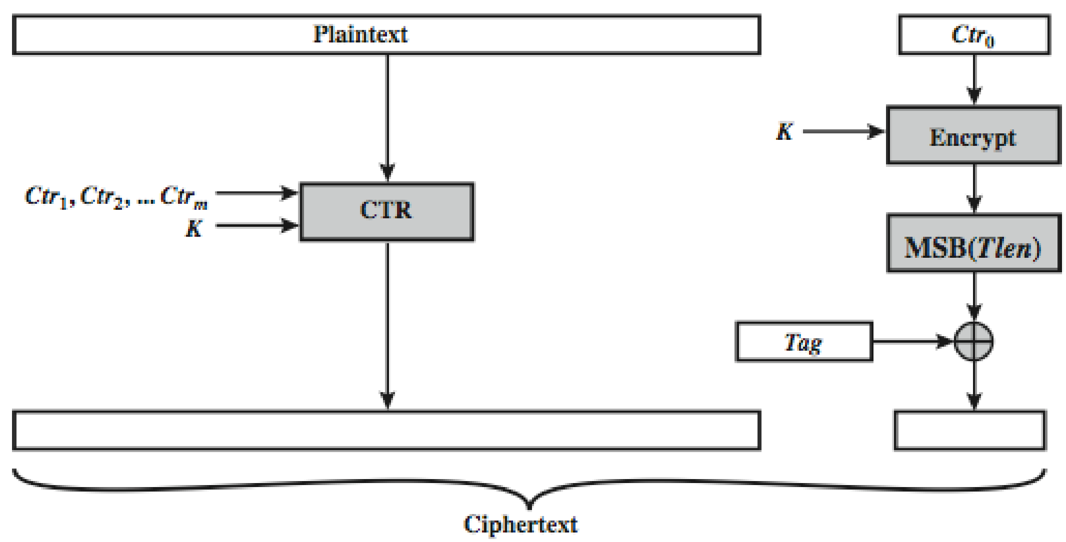
Counter with Cipher Block Chaining-Message Authentication Code (CCM)

- NIST standard SP 800-38C for WiFi
- Variation of encrypt-and-MAC approach
- Algorithmic ingredients
  - AES encryption algorithm
  - CTR mode of operation
  - CMAC authentication algorithm
- Single key used for both encryption and MAC

CCM Operation



(a) Authentication

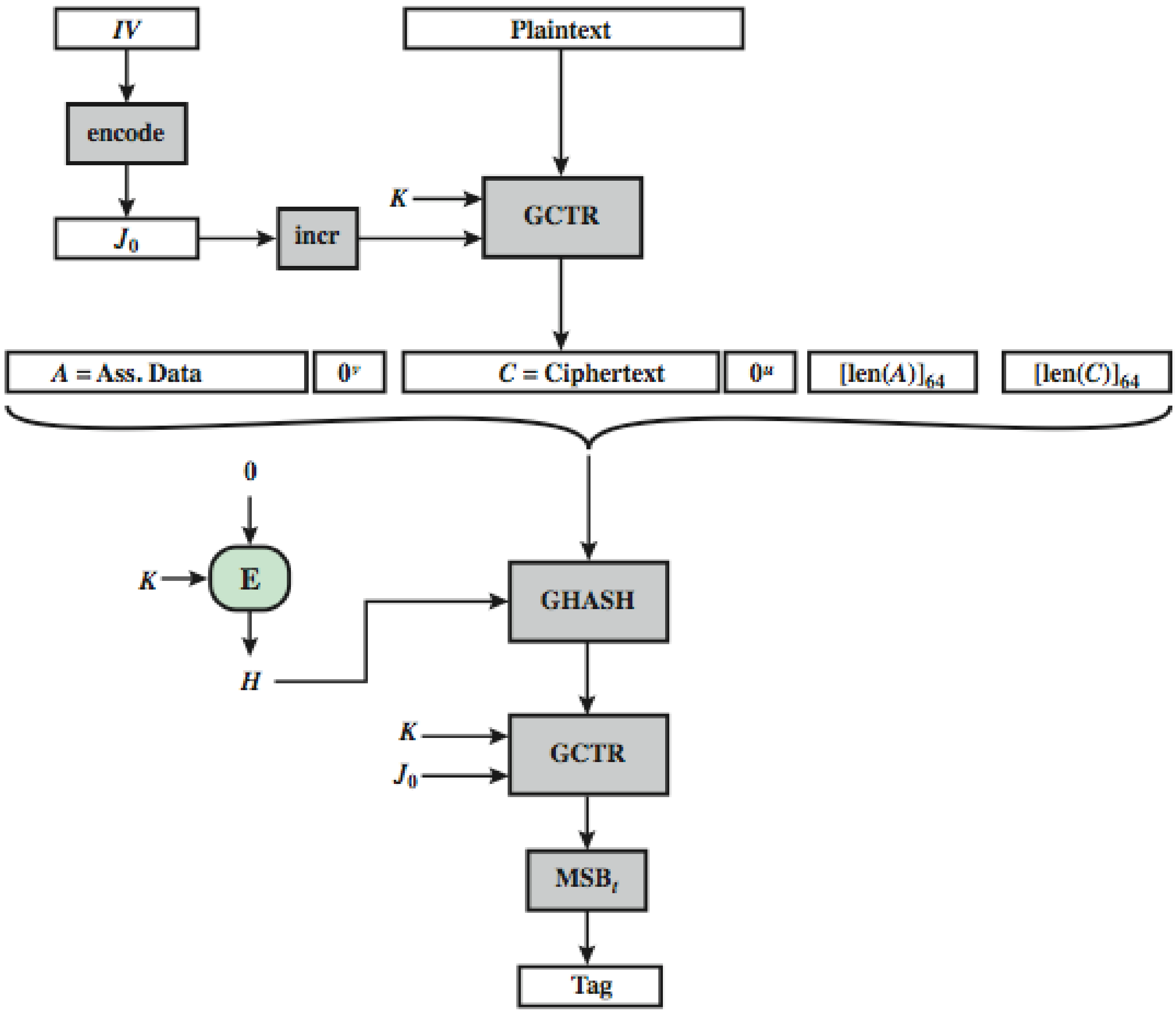


(b) Encryption

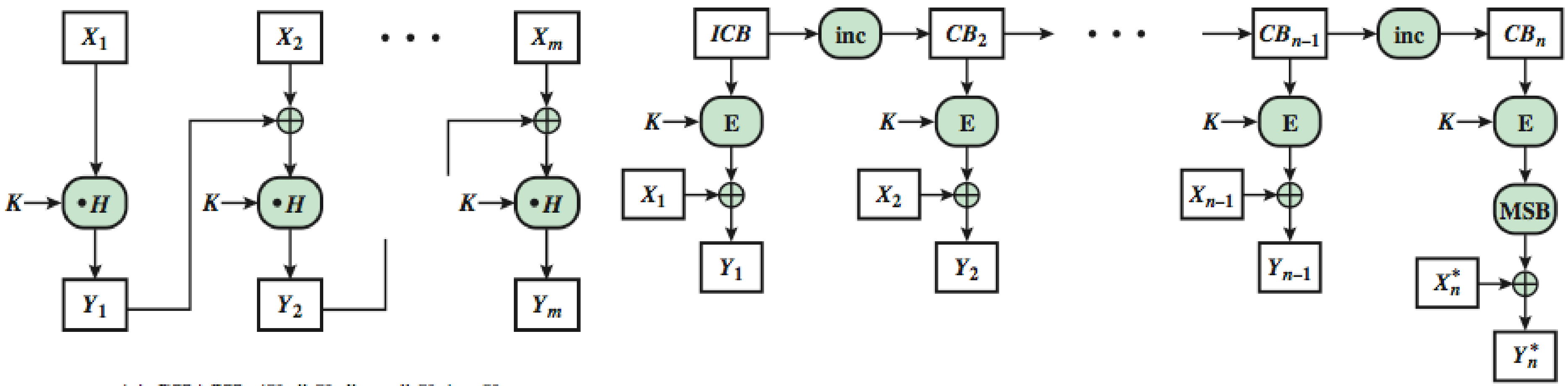
Galois/Counter Mode (GCM)

- NIST standard SP 800-38D, parallelizable
- Nessage is encrypted in variant of CTR
- Ciphertext multiplied with key & length over in (2^{128}) to generate authenticator tag
- Have GMAC MAC-only mode also
- Uses two functions:
  - GHASH - a keyed hash function
  - GCTR - CTR mode with incremented count

GCM Mode Overview



GCM Functions



(a) $\mathrm{GHASH}_H(X_1 \parallel X_2 \parallel \ldots \parallel X_m) = Y_m$

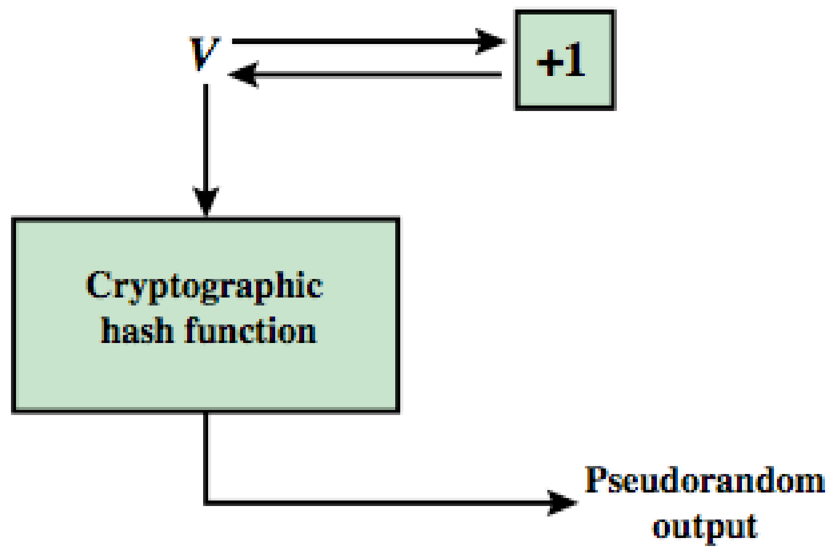
(b) $\mathrm{GCTR}_K(ICB, X_1 \parallel X_2 \parallel \ldots \parallel X_n^*) = Y_n^*$

PRNG using Hash Functions and MACs

- Essential elements of PRNG are
  - seed value
  - deterministic algorithm
- seed must be known only as needed
- Can base PRNG on
  - encryption algorithm (Chs 7 & 10)
  - hash function (ISO18031 & NIST SP 800-90)
  - MAC (NIST SP 800-90)

PRNG using Hash Functions

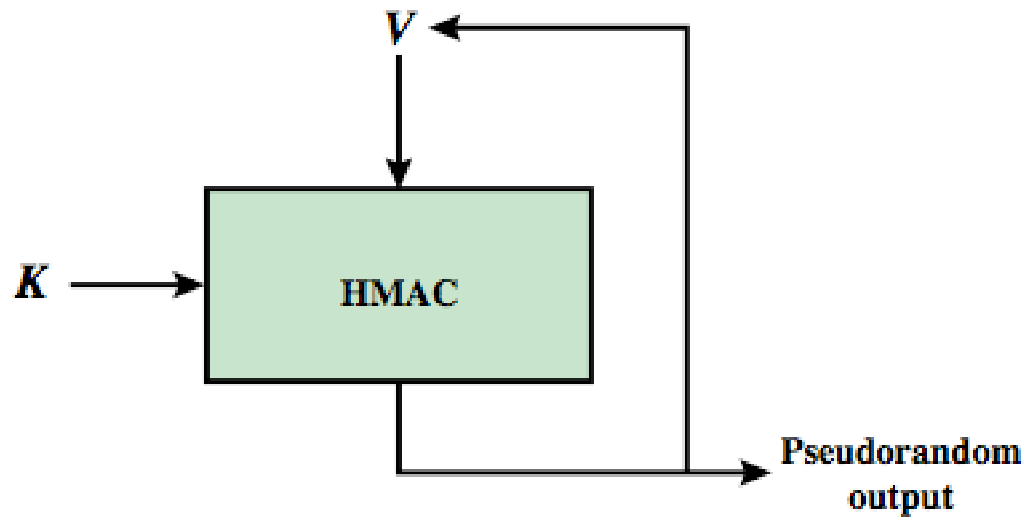
- hash PRNG from SP800-90 and ISO18031
  - take seed V
  - repeatedly add 1
  - hash V
  - use n-bits of has as random value
- secure if good hash used



(a) PRNG using cryptographic hash function

PRNG using MACs

- MAC PRNGs in SP800-90, IEEE 802.11i, TLS
  - use key
  - input based on last hash in various ways



(b) PRNG using HMAC

MDC vs. MAC

- Data integrity without confidentiality
  - MAC: compute $MAC_k(x)$ and send $x||MAC_k(x)$
  - MDC: send x and compute $MDC(x)$, which should be sent over an authenticated channel
- Data integrity with confidentiality
  - MAC: need two different keys k_1 and k_2
    - One for encryption and one for MAC
    - Compute $MAC_{k_1}(x)$ and sends $c = e_{k_2}(x||MAC_{k_1}(x))$
  - MDC: only needs one key k for encryption
    - One computes $MDC(x)$ and sends $c = e_k(x||MDC(x))$

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 are also provided by digital signatures, but MACs are much faster.
- MACs do not provide nonrepudiation.
- In practice, MACs are either based on block ciphers or on hash functions.
- HMAC is a popular MAC used in many practical protocols such as TLS.