Unit 4

LIKE
COMMENT
SHARE
SUBSCRIBE

1. Authentication requirement – Authentication function – MAC – Hash function – Security of hash function: HMAC, **CMAC** – SHA
2. Digital signature and authentication protocols – DSS – Schnorr Digital Signature Scheme – ElGamal cryptosystem
3. Entity Authentication: Biometrics, Passwords, Challenge Response protocols – Authentication applications – Kerberos MUTUAL
4. TRUST: Key management and distribution – Symmetric key distribution using symmetric and asymmetric encryption – Distribution of public keys – X.509 Certificates.

CMAC

CBC-MAC - fixed length

m ⇒

10110111 00011100 11111111 00000011

10110111 00011100 11111111 000

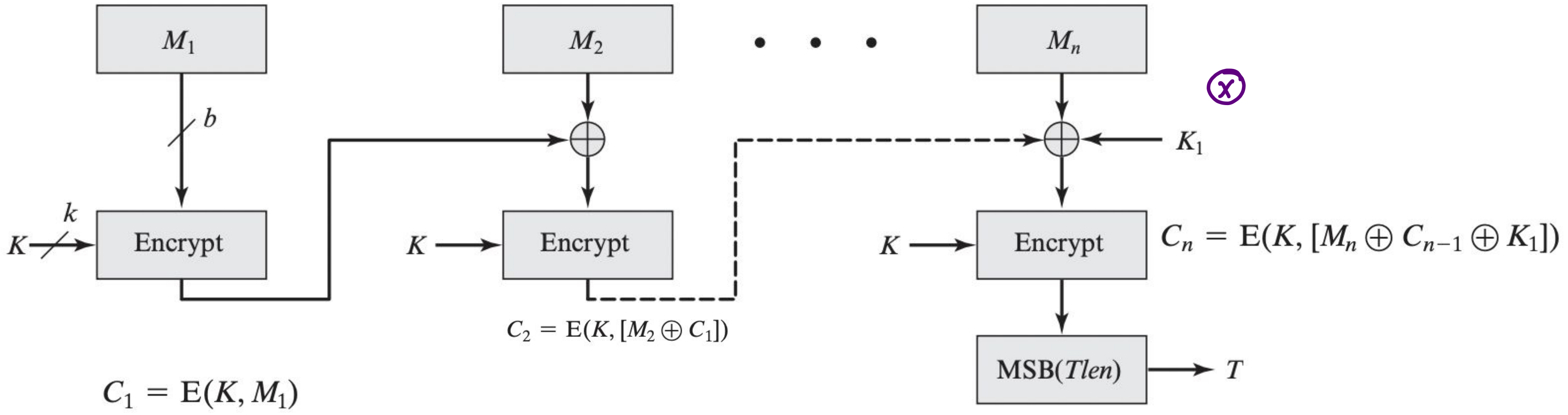
10110111	00011100	11111111	00000011
8 bits	8 bits	8 bits	8 bits

10110111	00011100	11111111	00010000

10 100000

Block Length	b	→ 8
Number of blocks	n	→ 4
Key length	k	⇒ K
Constant K1	b bits	
Constant K2 (Used when msg%b not equal to 0)	b bits	

CMAC (Cipher Based MAC) - Secure for variable length messages



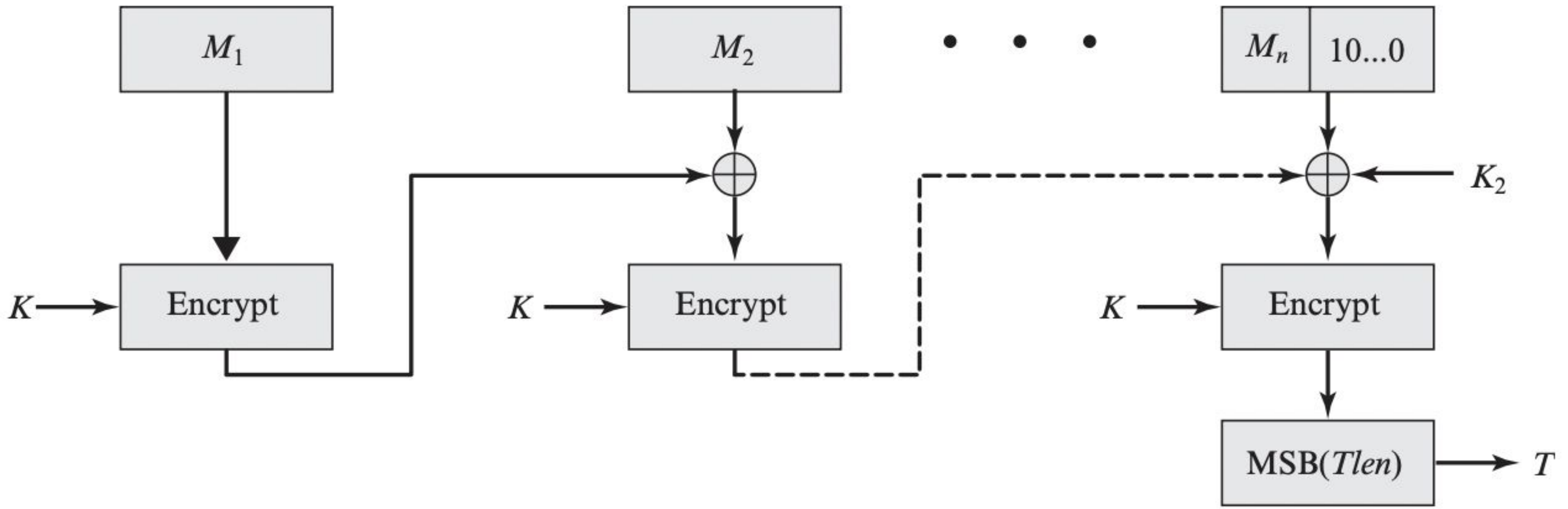
(a) Message length is integer multiple of block size

Block Length	b
Number of blocks	n
Key length	k
Constant K1	b bits
Constant K2 (Used when msg%b not equal to 0)	b bits

T = message authentication code, also referred to as the tag

$Tlen$ = bit length of T

$MSB_s(X)$ = the s leftmost bits of the bit string X



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

Calculation of Keys K_1 and K_2

To generate an ℓ-bit CMAC tag (t) of a message (m) using a b-bit block cipher (E) and a secret key (k), one first generates two b-bit sub-keys (k_1 and k_2) using the following algorithm (this is equivalent to multiplication by x and x^2 in a finite field $GF(2^b)$). Let $\ll$ denote the standard left-shift operator and $\oplus$ denote bit-wise exclusive or:

1. Calculate a temporary value $k_0 = E_k(0)$.
2. If $\text{msb}(k_0) = 0$, then $k_1 = k_0 \ll 1$, else $k_1 = (k_0 \ll 1) \oplus C$; where C is a certain constant that depends only on b. (Specifically, C is the non-leading coefficients of the lexicographically first irreducible degree-b binary polynomial with the minimal number of ones: 0x1B for 64-bit, 0x87 for 128-bit, and 0x425 for 256-bit blocks.)
3. If $\text{msb}(k_1) = 0$, then $k_2 = k_1 \ll 1$, else $k_2 = (k_1 \ll 1) \oplus C$.
4. Return keys (k_1, k_2) for the MAC generation process.

3 ~~4~~ 5 6 7

Key Points

First, let us define the operation of CMAC when the message is an integer multiple n of the cipher block length b. For AES, $b = 128$, and for triple DES, $b = 64$. The message is divided into n blocks $(M_1, M_2, \ldots, M_n)$. The algorithm makes use of a k-bit encryption key K and a b-bit constant, K_1. For AES, the key size k is 128, 192, or 256 bits; for triple DES, the key size is 112 or 168 bits. CMAC is calculated as follows (Figure 12.8).

$$
\begin{aligned}
C_1 &= \mathrm{E}(K, M_1) \\
C_2 &= \mathrm{E}(K, [M_2 \oplus C_1]) \\
C_3 &= \mathrm{E}(K, [M_3 \oplus C_2]) \\
&\vdots \\
C_n &= \mathrm{E}(K, [M_n \oplus C_{n-1} \oplus K_1]) \\
T &= \mathrm{MSB}_{Tlen}(C_n)
\end{aligned}
$$

where

T	=	message authentication code, also referred to as the tag
$Tlen$	=	bit length of T
$\mathrm{MSB}_s(X)$	=	the s leftmost bits of the bit string X

If the message is not an integer multiple of the cipher block length, then the final block is padded to the right (least significant bits) with a 1 and as many 0s as necessary so that the final block is also of length b. The CMAC operation then proceeds as before, except that a different b-bit key K_2 is used instead of K_1.


thank you


LIKE
COMMENT
SHARE
SUBSCRIBE