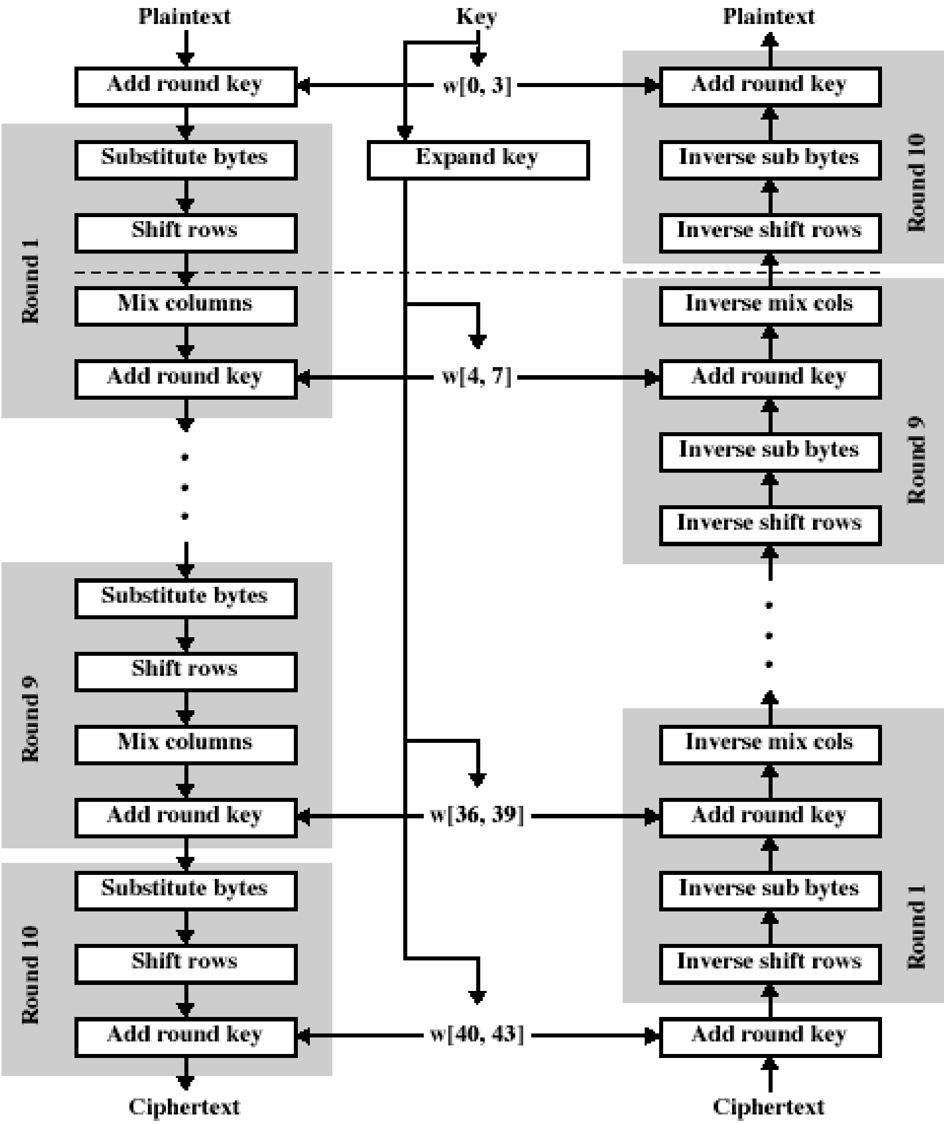
CS557: Cryptography

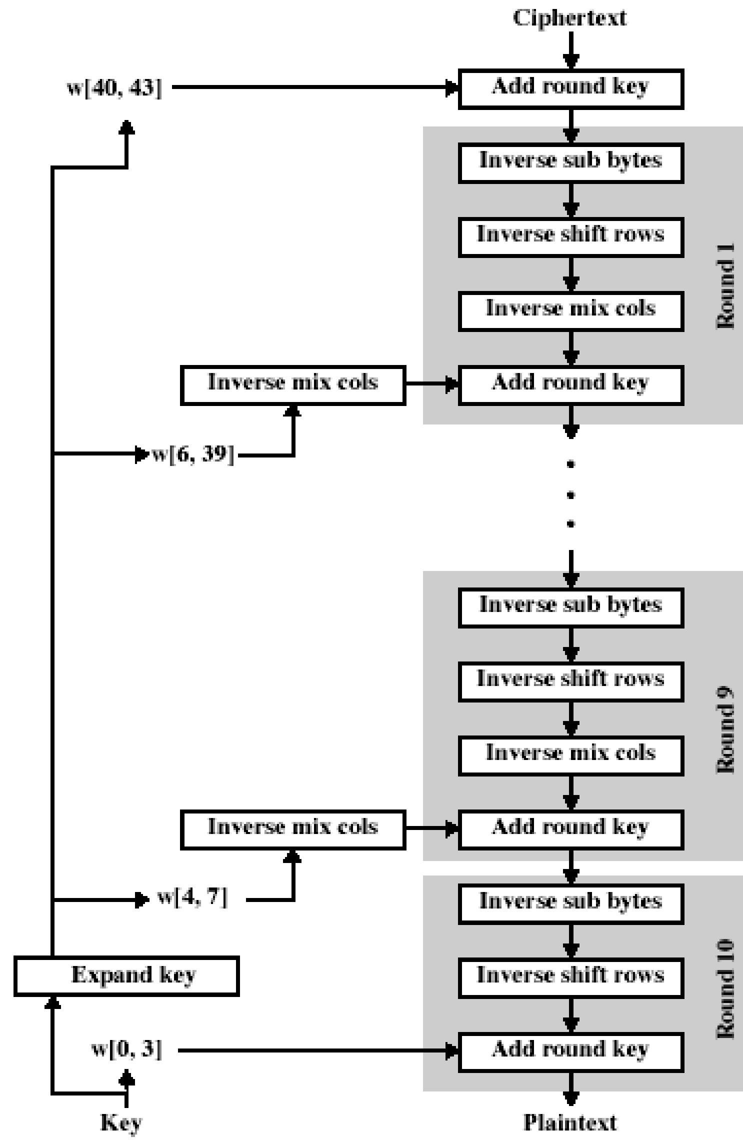
Modern Ciphers (AES-XTS)

S. Tripathy
IIT Patna

AES Encryption/ Decryption

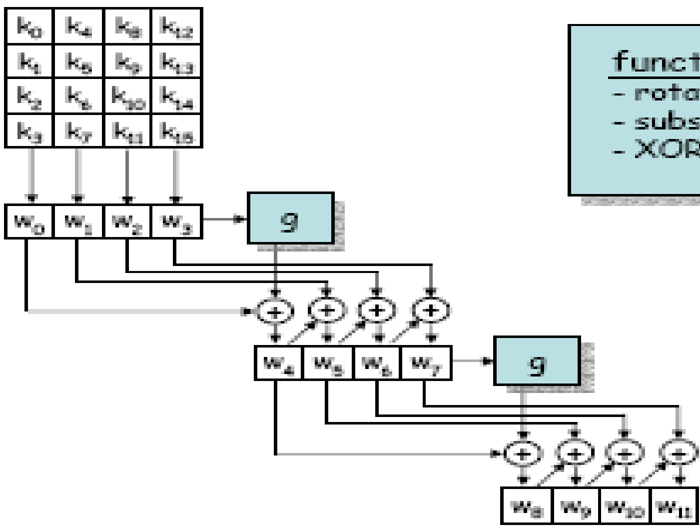


(a) Encryption (b) Decryption



AES Key Expansion

- takes 128/192/256-bit (16/24/32-byte) key and expands into array of 44/52/60 32-bit words
- start by copying key into first 4 words
- then loop creating words that depend on values in previous and 4 places back
  - in 3 of 4 cases just XOR these together
  - every 4^{th} has S-box + rotate + XO



function g
- rotate word
- substitute by
- XOR with rou

Key Words	Auxiliary Function
w0 = 0f 15 71 c9	RotWord(w3)= 7f 67 98 af = x1
w1 = 47 d9 e8 59	SubWord(x1)= d2 85 46 79 = y1
w2 = 0c b7 ad	Rcon(1)= 01 00 00 00
w3 = af 7f 67 98	y1 ⊕ Rcon(1)= d3 85 46 79 = z1
w4 = w0 ⊕ z1 = dc 90 37 b0	RotWord(w7)= 81 15 a7 38 = x2
w5 = w4 ⊕ w1 = 9b 49 df e9	SubWord(x4)= 0c 59 5c 07 = y2
w6 = w5 ⊕ w2 = 97 fe 72 3f	Rcon(2)= 02 00 00 00
w7 = w6 ⊕ w3 = 38 81 15 a7	y2 ⊕ Rcon(2)= 0e 59 5c 07 = z2
w8 = w4 ⊕ z2 = d2 c9 6b b7	RotWord(w11)= ff d3 c6 e6 = x3
w9 = w8 ⊕ w5 = 49 80 b4 5e	SubWord(x2)= 16 66 b4 8e = y3
w10 = w9 ⊕ w6 = de 7e c6 61	Rcon(3)= 04 00 00 00
w11 = w10 ⊕ w7 = e6 ff d3 c6	y3 ⊕ Rcon(3)= 12 66 b4 8e = z3
w12 = w8 ⊕ z3 = c0 af df 39	RotWord(w15)= ae 7e c0 b1 = x4
w13 = w12 ⊕ w9 = 89 2f 6b 67	SubWord(x3)= e4 f3 ba c8 = y4
w14 = w13 ⊕ w10 = 57 51 ad 06	Rcon(4)= 08 00 00 00
w15 = w14 ⊕ w11 = b1 ae 7e c0	y4 ⊕ Rcon(4)= ec f3 ba c8 = 4

AES Diffusion: Single Byte

Round 1

Input

s00	s01	s02	s03
s10	s11	s12	s13
s20	s21	s22	s23
s30	s31	s32	s33

After ShiftRows

s00	s01	s02	s03
s11	s12	s13	s10
s22	s23	s20	s21
s33	s30	s31	s32

After MixColumns

s’00	s’01	s’02	s’03
s’11	s’12	s’13	s’10
s’22	s’23	s’20	s’21
s’33	s’30	s’31	s’32

Round 2

s’00	s’01	s’02	s’03
s’12	s’13	s’10	s’11
s’20	s’21	s’22	s’23
s’32	s’33	s’30	s’31

s”00	S”01	s”02	s”03
s”12	s”13	s”10	s”11
s”20	s”21	s”22	s”23
s”32	s”33	s”30	s”31

Note: AddRoundKey has no impact on diffusion

Avalanche effect

- Key: 0f1571c947d9e8590cb7add6af7f6798
- Plaintext:

  0123456789abcdeffedcba9876543210

  0023456789abcdeffedcba9876543210
- Ciphertext

  ffob844a0853bf7c6934ab4364148fb9

  612b89398d0600cde11627ce72433f0

  58-Bit
- Plaintext:

  0123456789abcdeffedcba9876543210
- Key:

  0f1571c947d9e8590cb7add6af7f6798

  0e1571c947d9e8590cb7add6af7f6798
- Ciphertext:

  ffob844a0853bf7c6934ab4364148fb9

  fc8923ee501a7d207ab670686839996b

  53-Bit

Strength against known attacks

- Brute-Force Attack
  - AES is definitely more secure than DES due to the larger-size key.

- Differential cryptanalysis(DC)
  - Necessary condition to be resistant against DC: No DT with predicated PR > 2^{-n+1}, n the block length.
  - For Rijndael: No 4-round DT with predicated PR above 2^{-150} (no 8-round trails with PR above 2^{-300})

- Linear cryptanalysis(LC)
  - Necessary condition to be resistant against LC: No LTs with a correlation coefficients > $2^{n/2}$
  - For Rijndael: No 4-round LTs with a correlation above 2^{-75} (no 8-round trails with a correlation above 2^{-150}).

Modes of Encryption

- ECB:
  - Using the same key on multiple blocks makes it easier to break
  - Identical Plaintext Identical Ciphertext Does not change pattern:
- CBC:
  - Previous cipher blocks is chained with current plaintext block. Use an Initial Vector (IV) to start process.
  - Any change to a block affects all following ciphertext blocks
  - attacker can change bits of first block, and change IV to compensate
- OFB:
- CFB:
- CTR

Storage Encryption

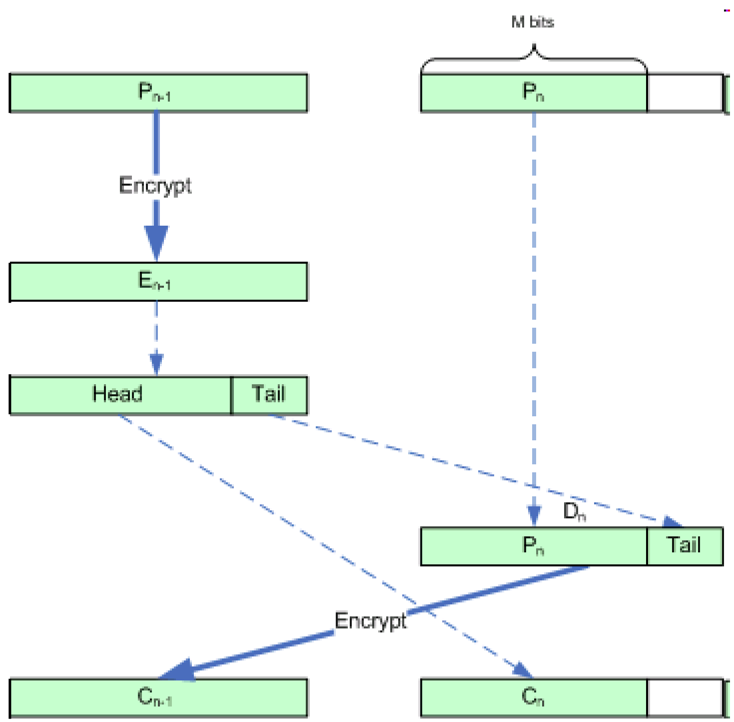
- If IV is predictable, CBC is not usable in storage because the plain text is chosen by the writer
  - Ciphertext is easily available to other users of the same disk
  - Two messages with the first blocks=b $\oplus$ IV1 and b $\oplus$ IV2 will both encrypt to the same ciphertext

- Last block may be shorter than others → Pad (size of CT ++)

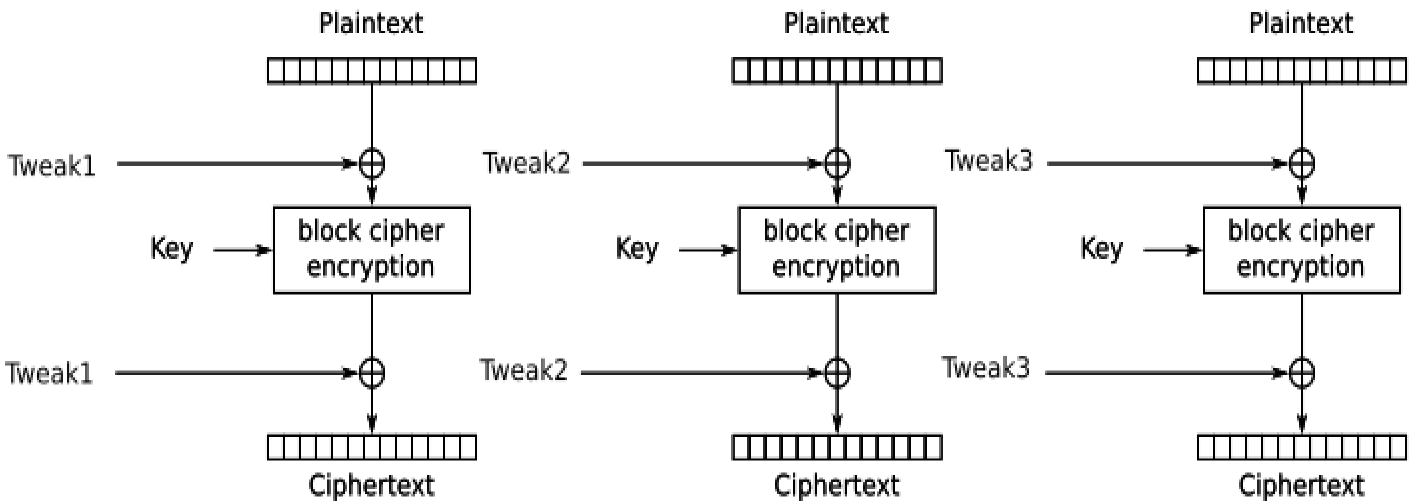
- Need to be able to read/write blocks without reading/writing other blocks

Cipher Text Stealing (CTS)

- Alternative to padding
- Last 2 blocks are specially coded
- Tail bits of $(n\text{-}1)^{st}$ encoded block are added to nth block and order of transmission of the two blocks is interchanged



XEX (xor–encrypt–xor)



Xor Encrypt Xor (XEX) mode encryption

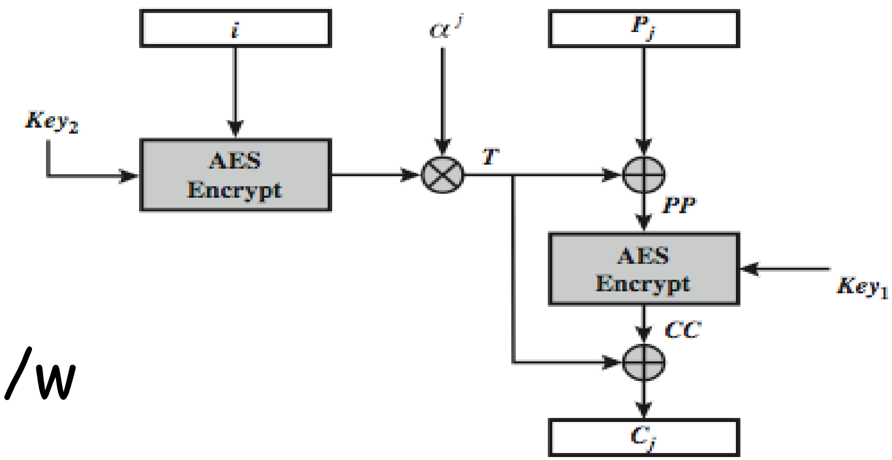
AES-XTS

- XTS = XEX-based Tweaked Codebook mode with Cipherstealing
  - A mode, for block oriented storage use
  - in IEEE Std 1619-2007
  - Stealing (XEX = Xor-Encrypt-xor)

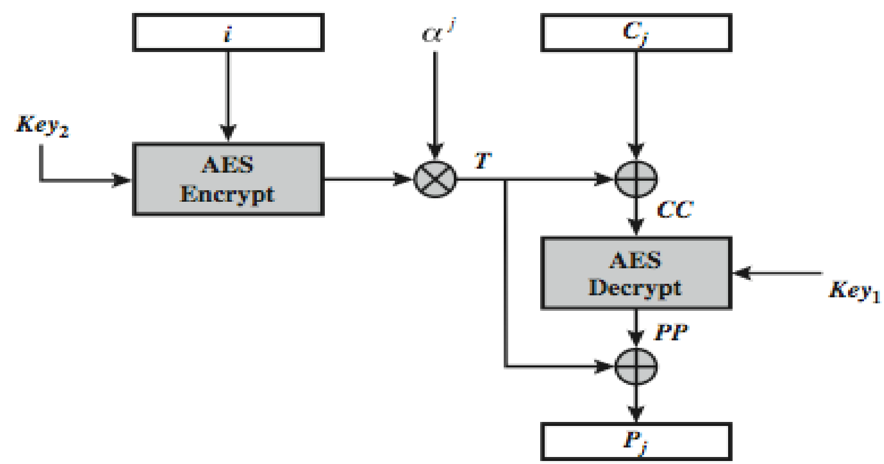
- Creates a unique IV for each block using AES and 2 keys
  - Tj = EK2(i) $\oplus$ α^j Size of K2 = size of block
  - Cj = EK1(Pj $\oplus$ Tj) $\oplus$ Tj K1 256 bit for AES-256
    - where i is logical sector # & j is block # (sector = n blocks)
  - α = primitive element in GF(2^{128}) defined by polynomial x

XTS-AES Mode per block

- Efficiency
  - Can do parallel encryptions in h/w or s/w
  - Random access to encrypted data blocks
- Has both nonce & counter

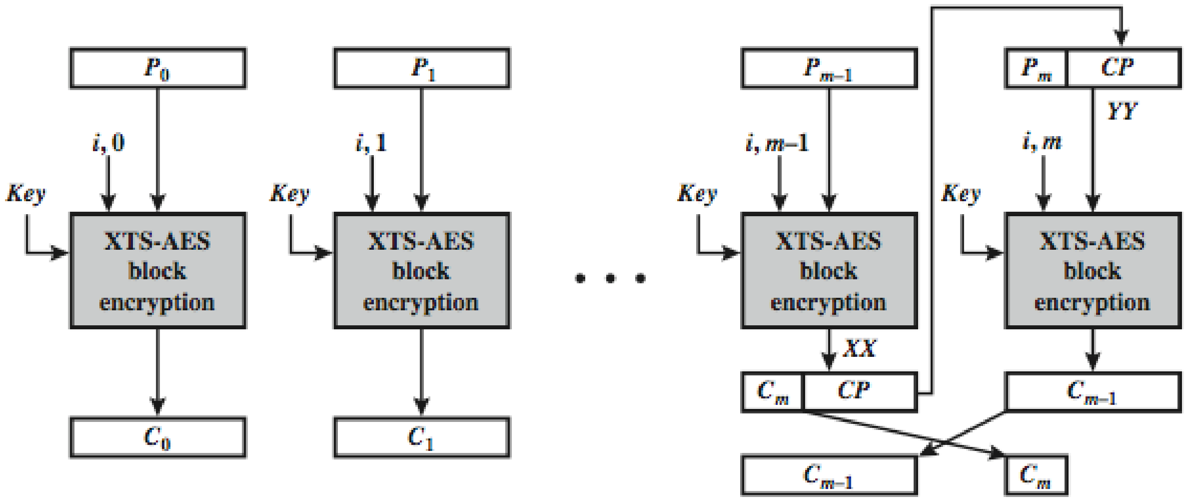


(a) Encryption

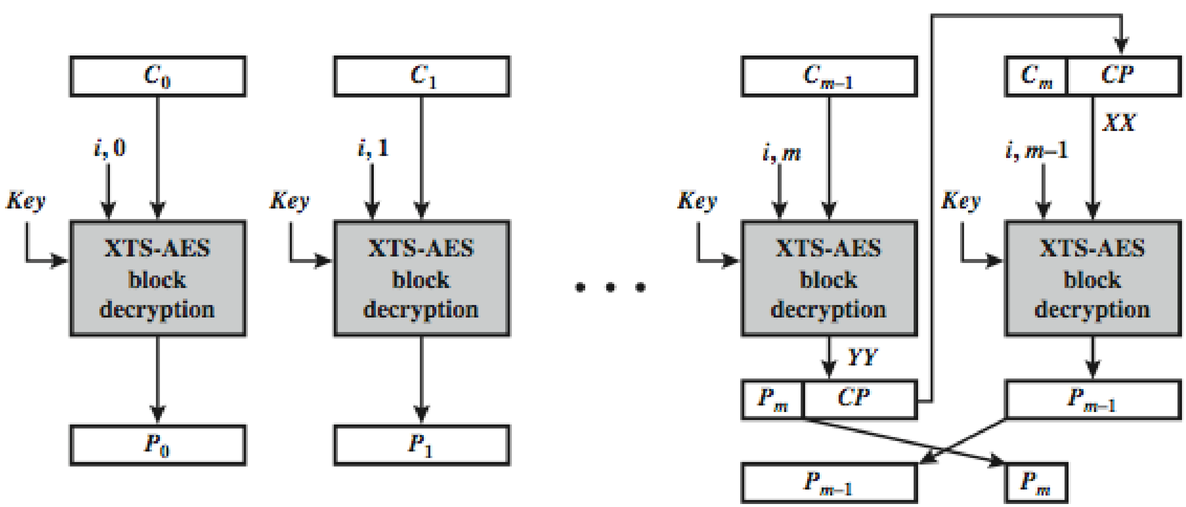


(b) Decryption

XTS-AES Mode Overview



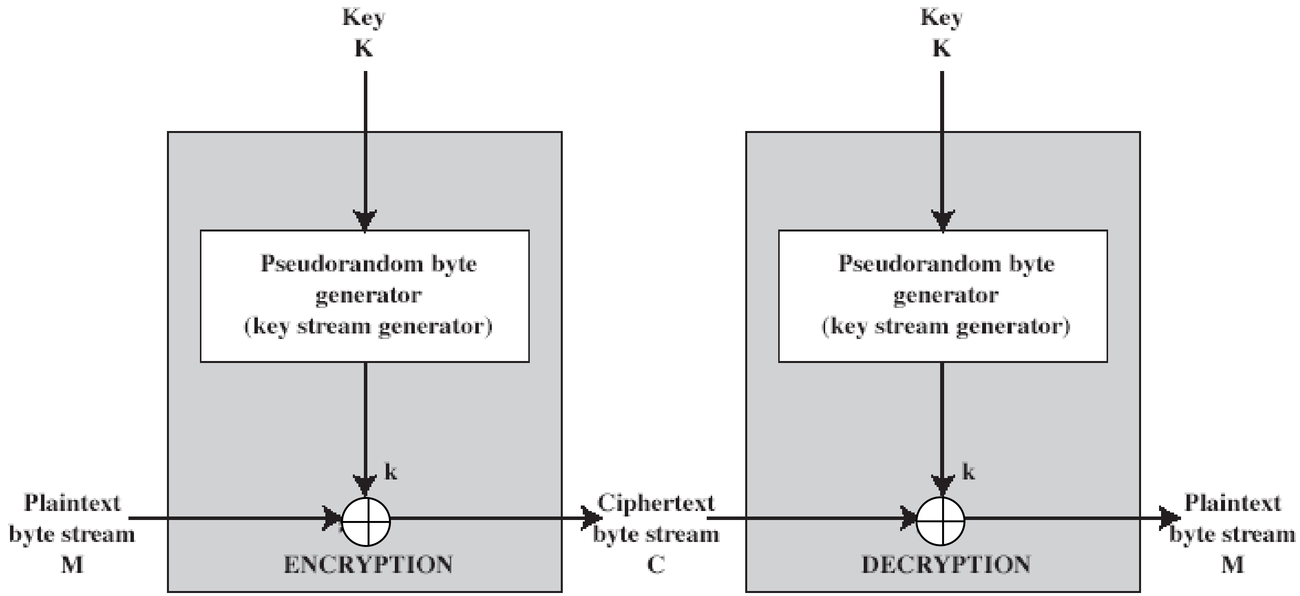
(a) Encryption



(b) Decryption

Stream Ciphers

- process the message bit by bit (as a stream)
- typically have a (pseudo) random stream key
- combined (XOR) with plaintext bit by bit
- randomness of stream key completely destroys any statistically properties in the message
  - $C_i = M_i \text{ XOR } StreamKey_i$



Never reuse stream key

- Stream cipher outputs keystream, KS
  - KS produced by a function, F, that is initialized with a key, k
  - $C = E_k(P) = P \oplus KS$
  - $P = C \oplus KS$
- k can be used only once
  - $C1 = E_{k1}(P1)$; $C2 = E_{k2}(P2)$
  - $C1 \oplus C2 = P1 \oplus KS1 \oplus P2 \oplus KS2 = P1 \oplus P2$ if $KS1 = KS2$
  - Will know when P1 and P2 have identical bits
  - If know part of P1 (if packet headers, format information), then can obtain part of P2

- Period – how long is KS before it starts repeating?
  - repeating is equivalent to reusing a key

- Thanks