Cryptography and Network Security
Chapter 6

- Fifth Edition
  by William Stallings
- Lecture slides by
  Lawrie Brown

Block Cipher Operation

Arabic Language Audio Comments by
Prof. Mohamed Ashraf Madkour

Multiple Encryption & DES

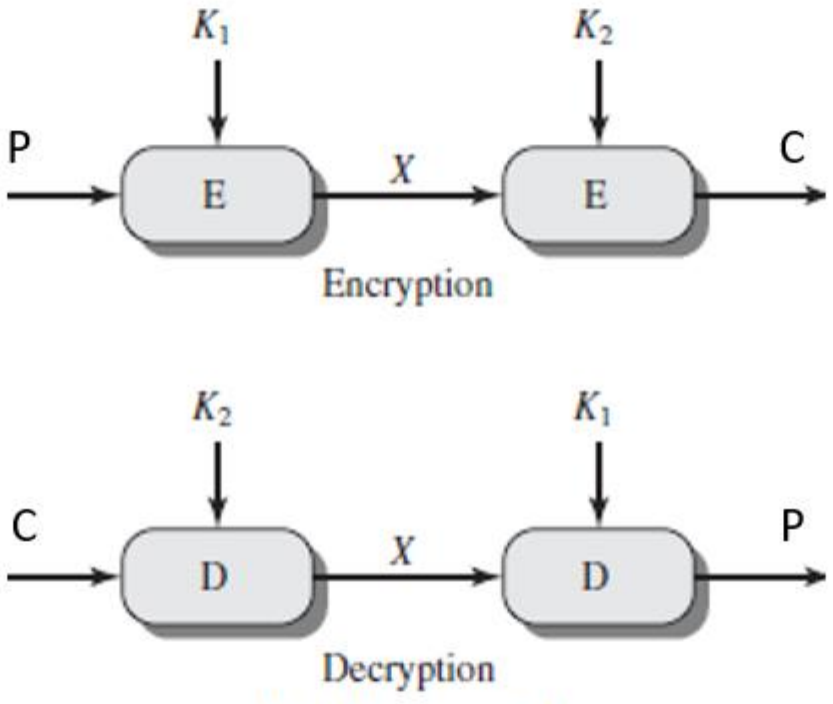
- a replacement for DES was needed
  - theoretical attacks that can break it
  - demonstrated exhaustive key search attacks
- AES is a new cipher alternative
- prior to this alternative was to use multiple encryption with DES implementations
- Triple-DES is the chosen form

Double-DES?

- The simplest form of multiple encryption has two encryption stages and two keys e.g. double-DES

$$C = E_{K2}(E_{K1}(P))$$

- However, it is subject to the meet in the middle (MitM) attack



Double encryption

“Meet-in-the-Middle” Attack (MitM)

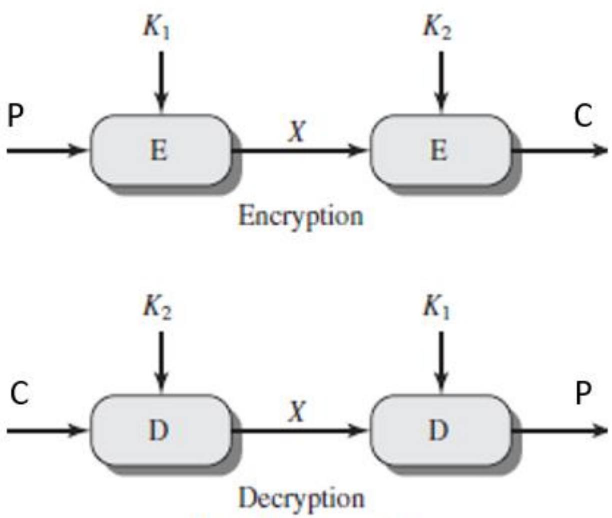
- The MitM attack on double encryption for any symmetric cipher such as DES and AES is possible with almost the same computing effort as breaking a single encryption.
- This effort is in the order of 2^N, where N is the number of bits in the encryption key.

"Meet-in-the-Middle" Attack

Double DES is subject to the "meet-in-the-middle" (MitM) attack which works whenever a cipher is used twice

- The attacker obtains a pair (P, C) where C is the ciphertext of the plaintext P.
- in double DES we have: $X = E_{K1}(P) = D_{K2}(C)$
- attack by encrypting P with all keys and store then decrypt C with keys and match X value
- MitM takes $O(2^{56})$ steps to break the 2 keys
- a brute force attack takes $O(2^{112})$ steps to break the 2 keys of DES

Detailed steps to do a MitM attack:



Double encryption

1. Find a pair of plaintext and ciphertext “P,C”.
2. For all possible N values of the 1st unknown key K_1 prepare a list {X_e ; e= 1,2,3,…., N}, where X_e = E(k_e, P).
3. Generate X_d = D(k_d, C) for possible values of the 2nd unknown key k_2 , where {d= 1,2,3,….}
4. Compare each generated X_d with the prepared list until you find a matching value such that E(k_e, P) = D(k_d, C).
5. The unknown keys are: K_1 = K_e and K_2 = K_d.
6. Use a second pair of plaintext and ciphertext to verify the obtained keys

Triple-DES with Two-Keys

(standardized in ANSI X9.17 & ISO8732)

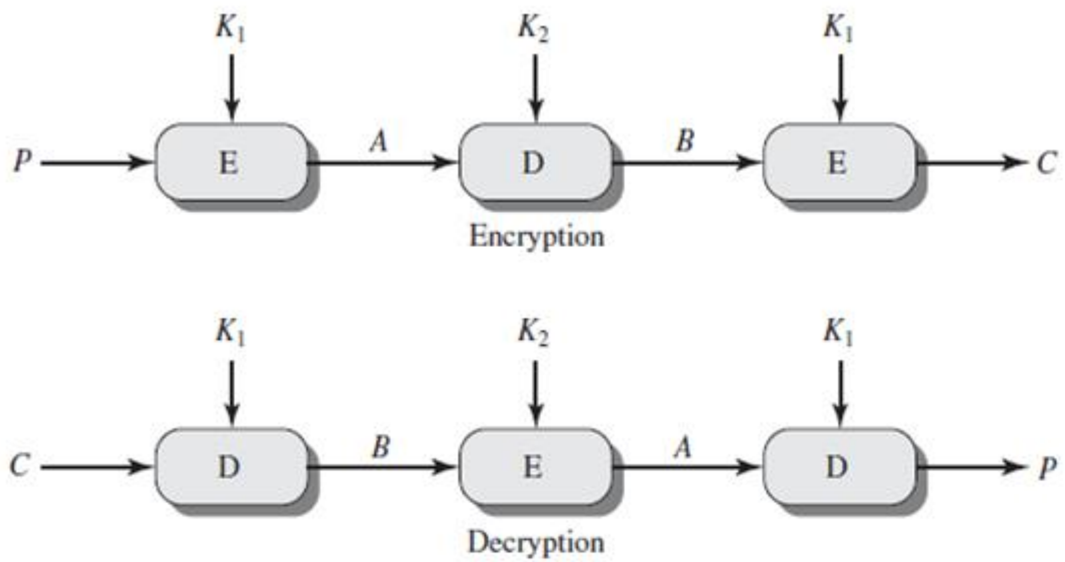
- to avoid MitM attack we must use 3 encryptions
- would seem to need 3 distinct keys but can use 2 keys with E-D-E sequence.

$$C = E_{K1}(D_{K2}(E_{K1}(P)))$$

- the use of encryption & decryption stages is equivalent in security, but the chosen E-D-E structure allows for compatibility with single-DES implementations.
- if K1=K2 then can work with single DES
- no current known practical attacks
- suffers from being 3 times slower to run

Triple-DES with Two-Keys

(The E – D – E Structure)



Triple encryption

To break this arrangement using MitM, the attacker needs a computing effort in the order of 2^{N+N}, where N is the number of bits in the encryption key.

Triple-DES with Three-Keys

- although are no practical attacks on two-key, Triple-DES have some concerns
- a preferred alternative is to use Triple-DES with Three-Keys to avoid any concerns

$$C = E_{K3}(D_{K2}(E_{K1}(P)))$$

- has been adopted by some Internet applications, e.g. PGP, S/MIME

Modes of Operation

- block ciphers encrypt fixed size blocks
  - e.g. DES encrypts 64-bit blocks with 56-bit key
- need some way to encrypt/decrypt arbitrary amounts of data in practice
- NIST SP 800-38A defines 5 modes
- have **block** and **stream** modes to cover a wide variety of applications
- can be used with any block cipher

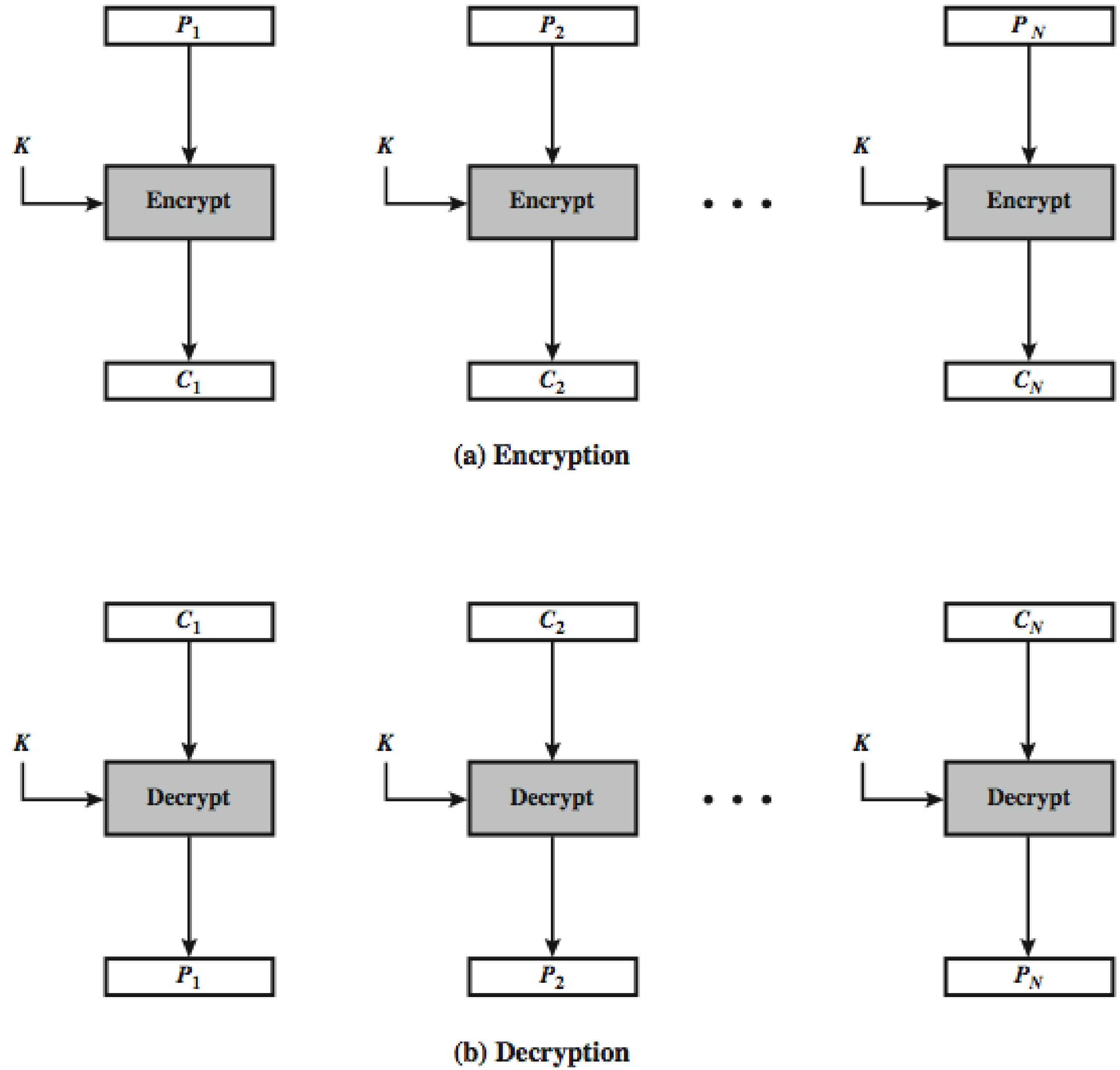
Electronic Codebook Book (ECB)

- message is broken into independent blocks which are encrypted
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks

  $C_i = E_K(P_i)$

- this mode is used for secure transmission of single values

Electronic Codebook Mode (ECB)



Advantages and Limitations of ECB

- message repetitions may show in ciphertext
  - if aligned with message block
  - particularly with data such as graphics
  - or with messages that change very little, which become a codebook analysis problem
- weakness is due to the encrypted message blocks being independent
- main use is sending a few blocks of data, e.g. a session encryption key

ECB Limitations

- Using the same key on multiple blocks makes it easier to break
- Identical Plaintext Identical Ciphertext
  Does not change pattern:

Original

ECB

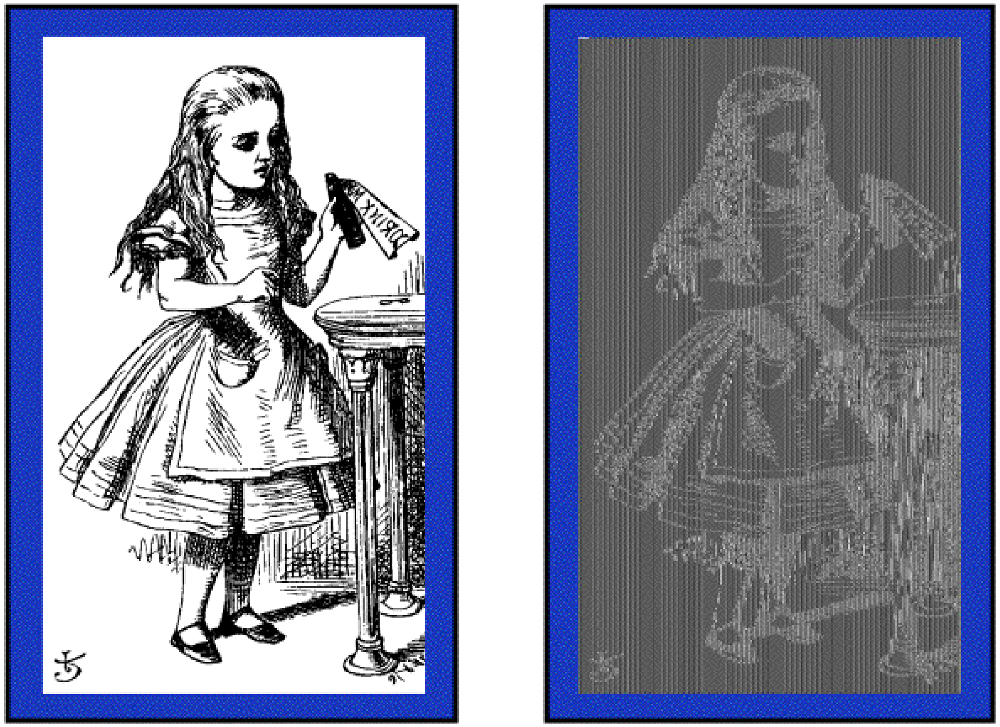
Better

- NIST SP 800-38A defines 5 modes **that** can be used with any block cipher

Ref: http://en.wikipedia.org/wiki/Modes_of_operation

Alice Hates ECB Mode

Alice's uncompressed image, and ECB encrypted



- Why does this happen?
- Same plaintext yields same ciphertext!

Cipher Block Chaining (CBC)

- message is broken into blocks
- linked together in encryption operation
- each previous cipher blocks is chained with current plaintext block, hence name
- use Initial Vector (IV) to start process; i=1,2,3,….

$$C_i = E_K(P_i \oplus C_{i-1})$$

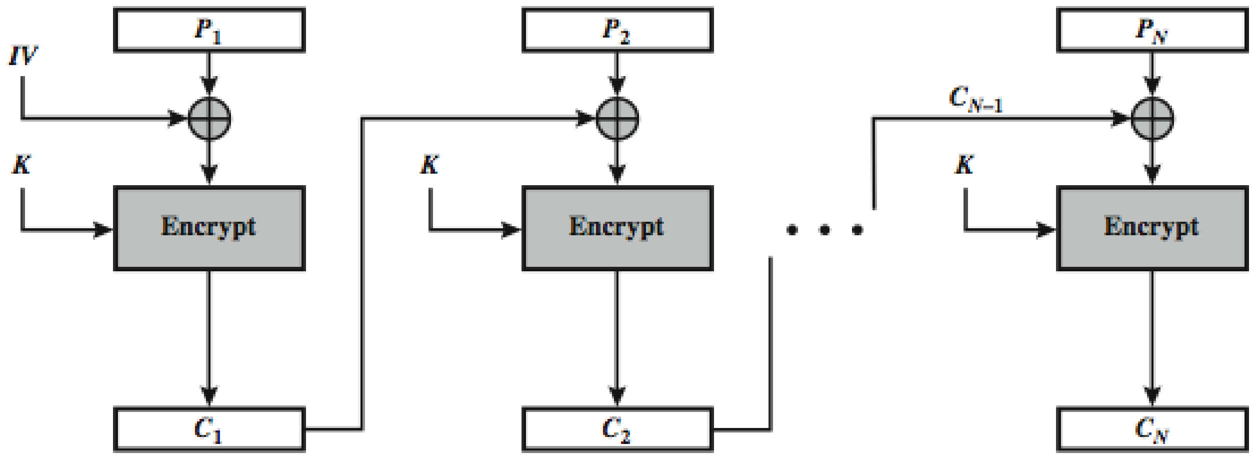
$$C_0 = IV$$

- Uses: bulk data encryption, authentication

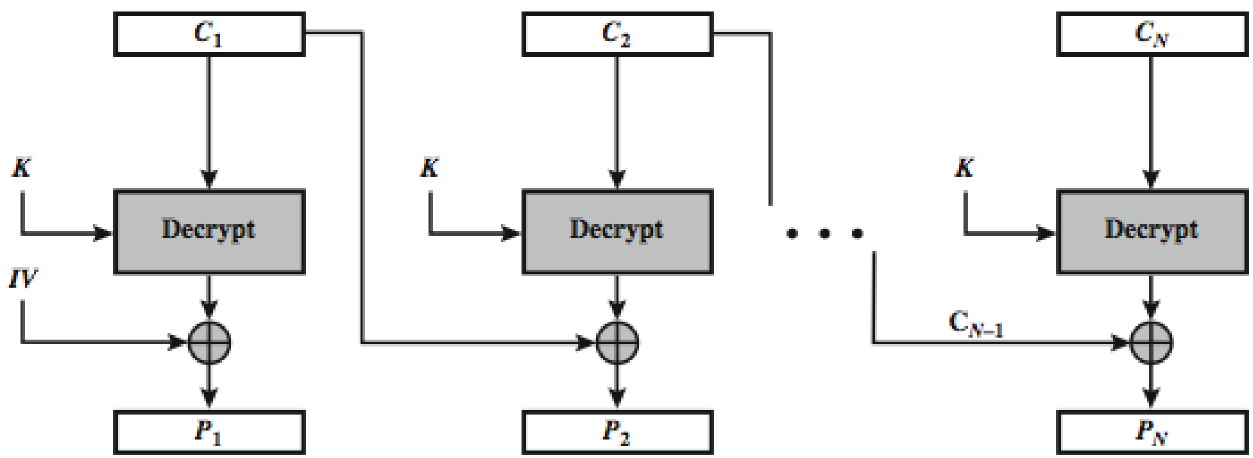
Cipher Block Chaining (CBC)

$C_i = E_K(P_i \oplus C_{i-1})$

$C_0 = IV$



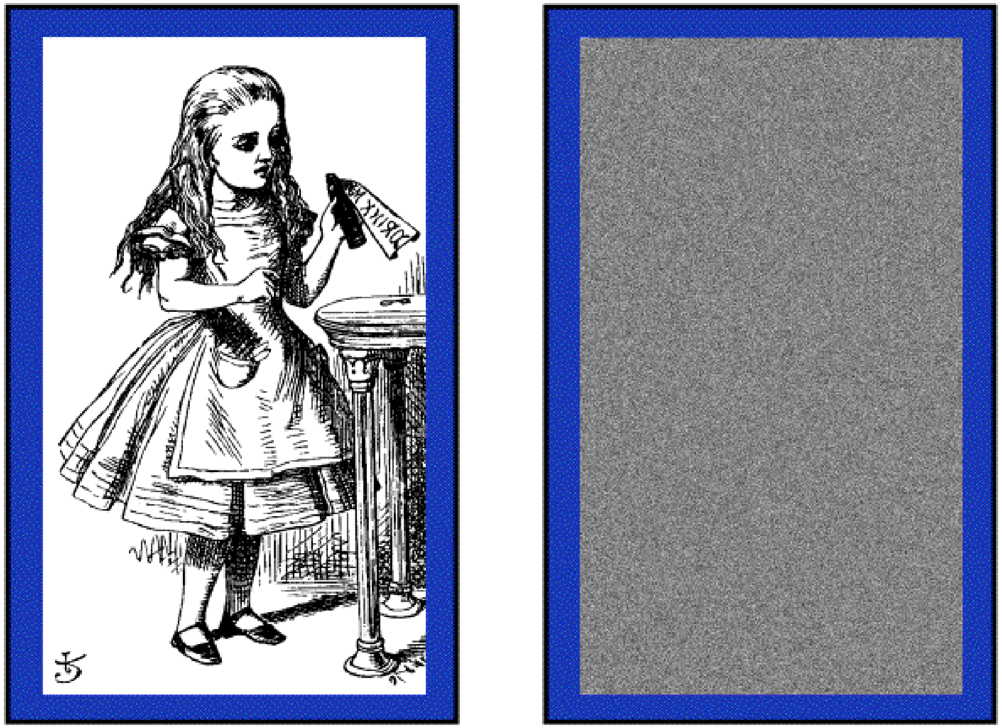
(a) Encryption



(b) Decryption

Alice Likes CBC Mode

Alice's uncompressed image, Alice CBC encrypted



- Why does this happen?
- Same plaintext yields different ciphertext!

Message Padding

At end of message we must handle a possible last short block.

- which is not as large as block size of cipher
- pad either with known non-data value (e.g. nulls)
- or pad last block along with count of pad size
  - Consider the last block containing only 3 bytes of data: [D1] , [D2] , and [D3].
  - To complete the block to be 8 bytes (64 bits) we must add 5 more bytes.
  - The padded block will be: {[D1] [D2] [D3] [0] [0] [0] [0] [5]}
  - This means we have 3 data bytes, then 4 bytes EACH CONTAINS ZERO + last byte contains the count of added padding bytes (which is 5).
- this method may require an extra entire block over those in message. (Why??)

Advantages and Limitations of CBC

- a ciphertext block depends on **all** blocks before it
- any change to a block affects all following ciphertext blocks
- need **Initialization Vector** (IV)
  - which must be known to sender & receiver
  - if sent in clear, attacker can change bits of first block, and change IV to compensate
  - hence IV must either be a fixed value (as in EFTPOS)
  - or must be sent encrypted in ECB mode before rest of message

Stream Modes of Operation

- Block modes encrypt entire block
- Stream modes convert block cipher into stream cipher
  1. cipher feedback (CFB) mode
     - allows to operate on smaller plaintext units, e.g. real time data
     - if a transmission error occurs in one ciphertext block, there will be an error in several blocks of the decrypted plaintext (this is called error propagation)
     - it uses a complex structure
  2. output feedback (OFB) mode
     - is simpler than CFB but preferred to operate on entire data blocks
  3. counter (CTR) mode
     - is like OFB but provides more advantages
- Encryption is done by XORing plaintext blocks with random bits
- Use block cipher as some form of **pseudo-random number** generator to generate the required random bits

Output FeedBack (OFB) Mode

- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance

  $O_i = E_K(O_{i-1})$

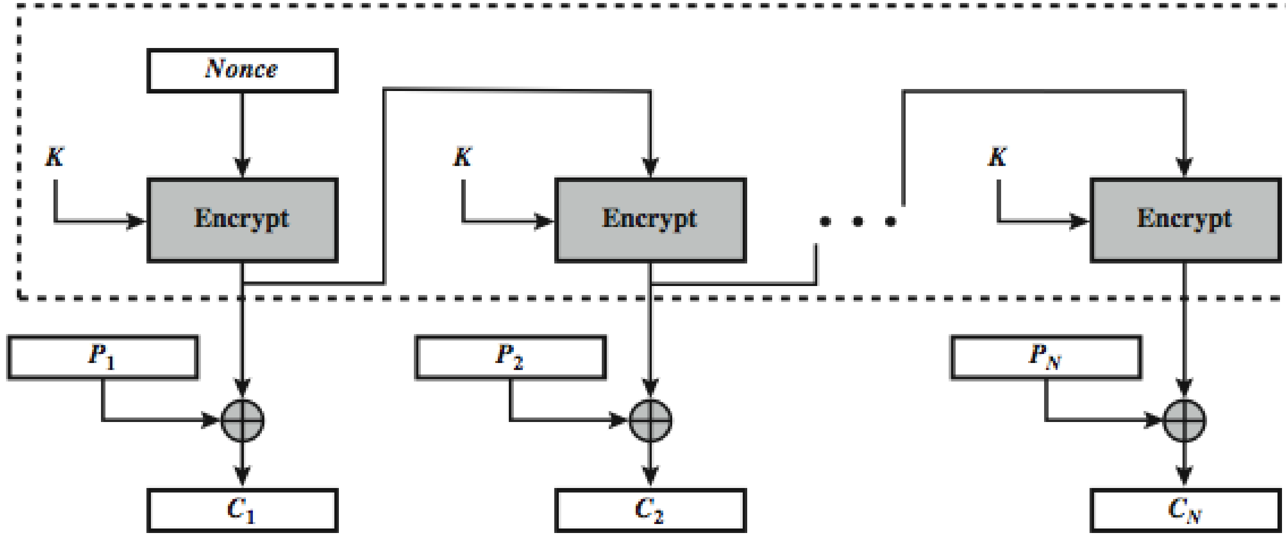
  $C_i = P_i \oplus O_i$

  $O_0 = IV$
- Uses: stream encryption on noisy channels

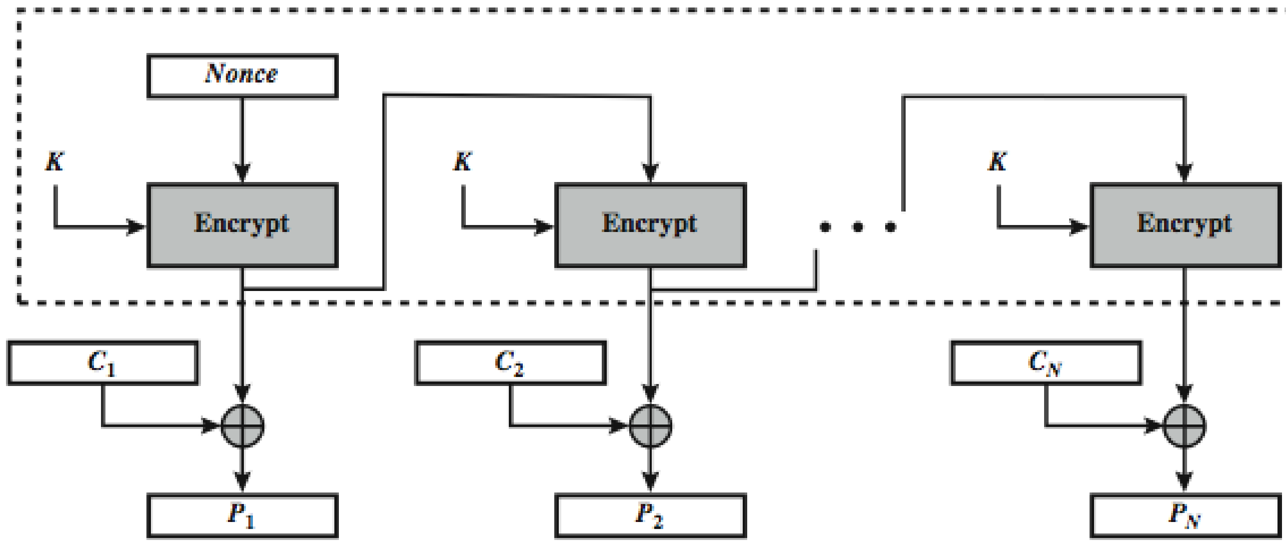
Output FeedBack (OFB)

$O_i = E_K(O_{i-1})$
$C_i = P_i \oplus O_i$
$O_0 = IV$

(IV is the nonce which is a random number used only once)



(a) Encryption



(b) Decryption

Advantages and Limitations of OFB

- needs an IV (nonce) which is unique for each use
- if ever reuse IV, attacker can recover outputs
- bit errors do not propagate
- more vulnerable to message stream modification attack than is CFB
- sender & receiver must remain in synchronism, or all data is lost
- only use with full block feedback, where typically a block is 64 or 128 bits

Counter (CTR) Mode

- the Counter (CTR) mode is a variant of OFB, but which encrypts a counter value (hence name) rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

$O_i = E_K(\mathit{Counter}_i)$

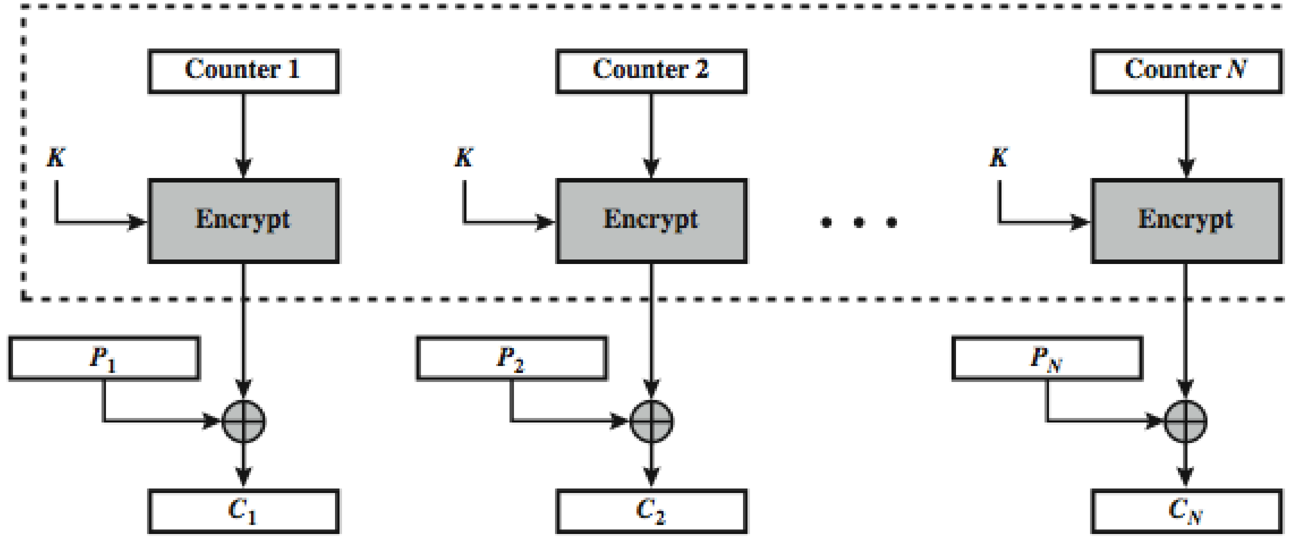
$C_i = P_i \oplus O_i$

Uses: high-speed network encryptions

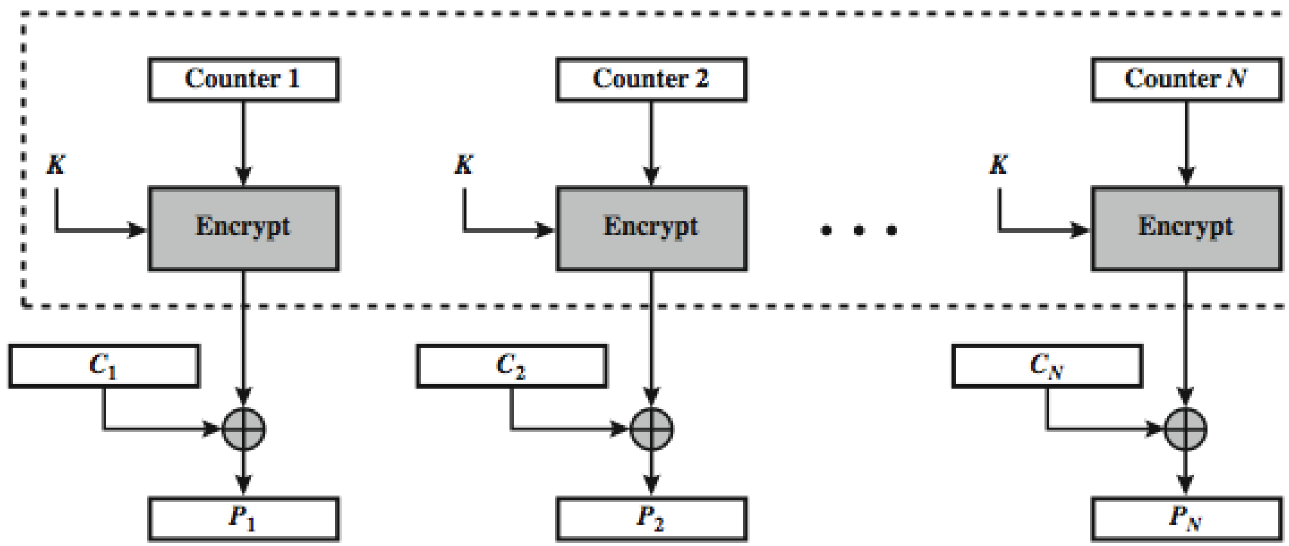
Counter (CTR)

$O_i = E_K(\mathit{Counter}_i)$
$C_i = P_i \oplus O_i$

- *$Counter_1$ is a random number*
- *$Counter_{i+1} = Counter_i + 1$*



(a) Encryption



(b) Decryption

Advantages and Limitations of CTR

- efficiency
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance of need
  - good for bursty high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break (like OFB)

Conclusion

Have considered:

- 2DES and 3DES
- Encryption modes