Encryption Algorithms & Protocols

Symmetric key Crypto
- Modes of Operation

Dr. Omar Abusada
E-mail: abossada1@gmail.com

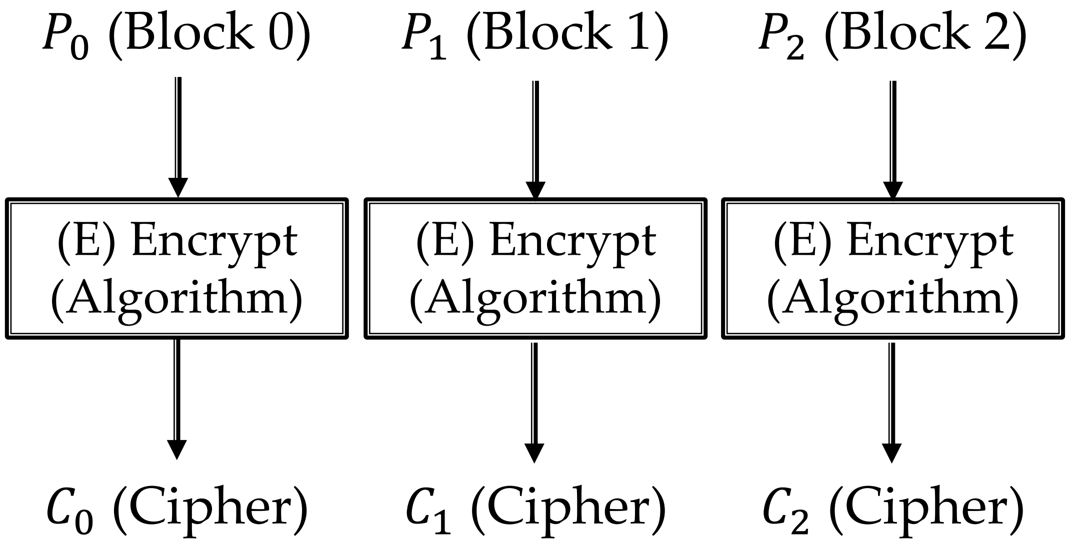
Modes of Operation

Many modes: we will discuss 3 most popular

1. ***Electronic Codebook*** (ECB) mode
   - Encrypt each block independently
   - Most obvious, but has a serious weakness
2. ***Cipher Block Chaining*** (CBC) mode
   - Chain the blocks together
   - More secure than ECB, virtually no extra work
3. ***Counter Mode*** (CTR) mode
   - Block ciphers acts like a stream cipher
   - Popular for random access

ECB Mode

- Notation: C = E(P,K)
- Given plaintext $P_0, P_1, \ldots \ldots \ldots \ldots P_m$
- Most obvious way to use a block cipher:
- Encrypt Decrypt
- $C_0 = E(P_0, K)$ $P_0 = D(C_0, K)$
- $C_1 = E(P_1, K)$ $P_1 = D(C_1, K)$
- $C_2 = E(P_2, K) \ldots$ $P_2 = D(C_2, K) \ldots$



For a fixed key K, this is an "electronic" version of a codebook cipher (without additive)

With a different codebook for each key

ECB Cut and Paste

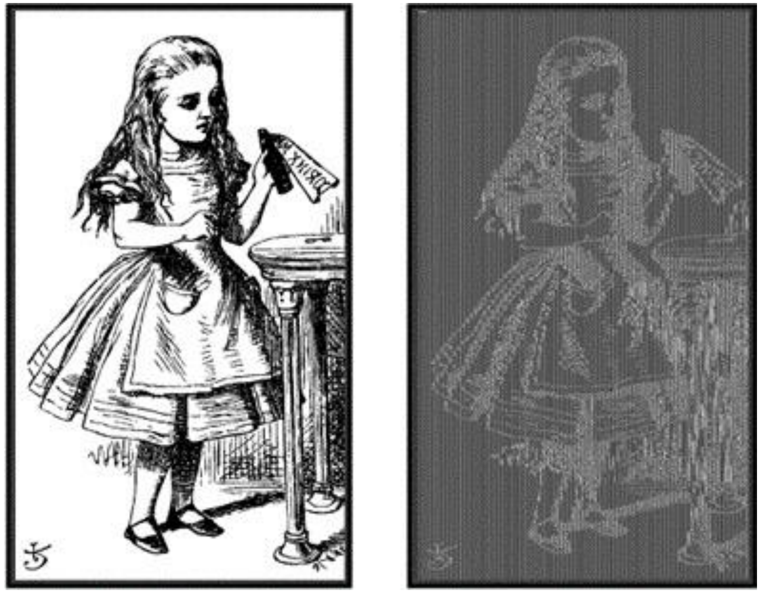
- Suppose plaintext is
  - Alice digs Bob. Trudy digs Tom.
- Assuming 64-bit blocks and 8-bit ASCII:
  - P_0 = "Alice di", P_1 = "gs Bob. ",
  - P_2 = "Trudy di", P_3 = "gs Tom. "
- Ciphertext: C_0, C_1, C_2, C_3
- Trudy cuts and pastes: C_0, C_3, C_2, C_1
- Decrypts as
  - Alice digs Tom. Trudy digs Bob.

ECB Weakness

- Suppose $P_i = P_j$
- Then $C_i = C_j$ and Trudy knows $P_i = P_j$
- This gives Trudy some information, even if she does not know P_i or P_j
- Trudy might know P_i
- Is this a serious issue?

Alice Hates ECB Mode

- Alice's uncompressed image, and ECB encrypted

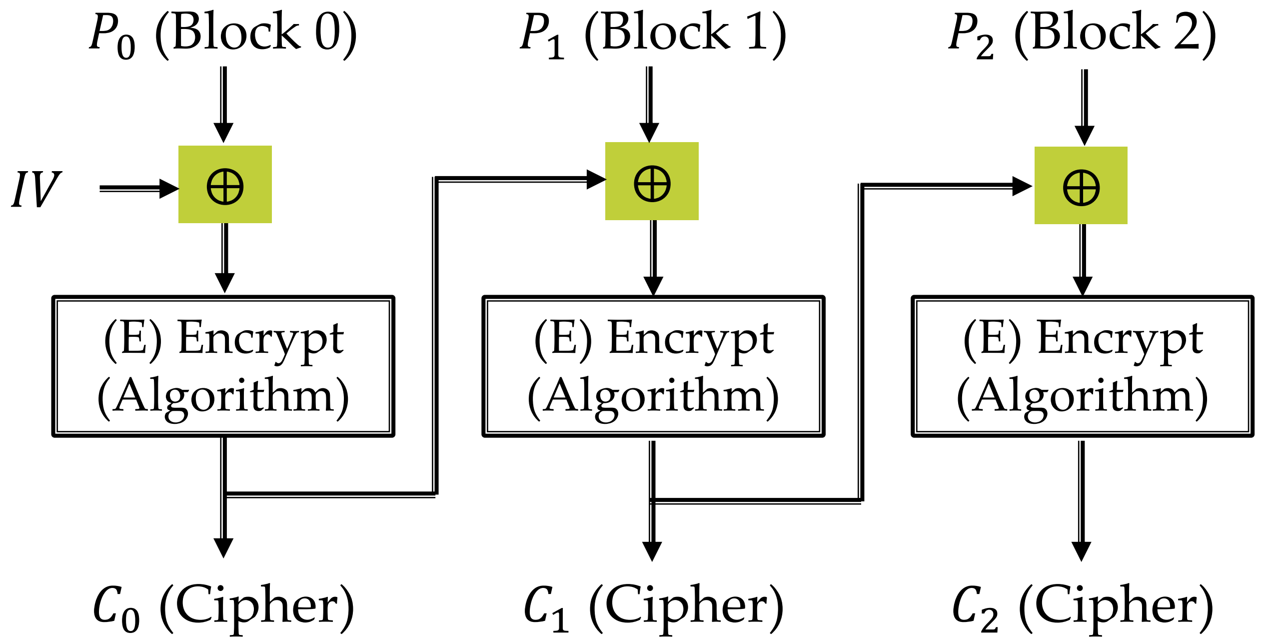


Cipher Block Chaining
CBC

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

CBC Mode

- Blocks are “chained” together
- A random Initialization Vector, or IV, is required to initialize CBC mode
- IV is random, but not secret



- Encrypt
- $C_0 = E(IV \oplus P_0, K)$
- $C_1 = E(C_0 \oplus P_1, K)$
- $C_2 = E(C_1 \oplus P_2, K)$ ...

Decrypt

$P_0 = IV \oplus D(C_0, K)$

$P_1 = C_0 \oplus D(C_1, K)$

$P_2 = C_1 \oplus D(C_2, K)$ ...

MAC

CBC Mode

- Identical plaintext blocks yield different ciphertext blocks ---- this is good!
- If C_1 is corrupted to, say, G then
  - $P_1 \neq C_0 \oplus D(G, K), P_2 \neq G \oplus D(C_2\ , K)$
  - But $P_3 = \ C_2 \oplus D(C_3, K), P_4 = \ C_3 \oplus D(C_4, K), \ldots \ldots$
- Automatically recovers from errors!
- Cut and paste is still possible, but more complex (and will cause garbles).

Alice Likes CBC Mode

- Alice's uncompressed image, Alice CBC encrypted

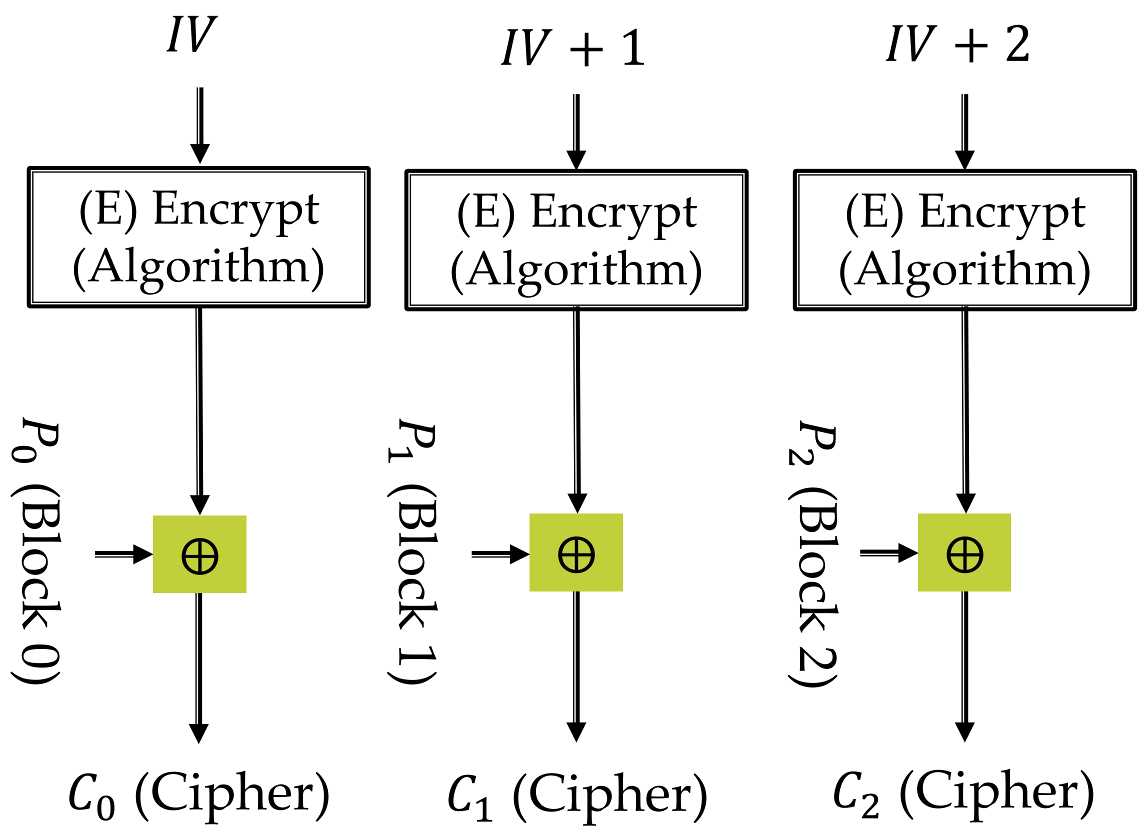

- Why does this happen?
- Same plaintext yields different ciphertext!

Counter Mode (CTR)

- CTR is popular for random access
- Use block cipher like a stream cipher

- Encrypt — Decrypt
- $C_0 = P_0 \oplus E(IV, K)$ — $P_0 = C_0 \oplus D(IV, K)$
- $C_1 = P_1 \oplus E(IV + 1, K)$ — $P_1 = C_1 \oplus D(IV + 1, K)$
- $C_2 = P_2 \oplus E(IV + 2, K)$ … — $P_2 = C_2 \oplus D(IV + 2, K)$, …

- CBC can also be used for random access
  - With a significant limitation…



Data Integrity

Block cipher can also show data integrity…

Data Integrity

- Integrity (Reliability, Truthfulness), detect unauthorized writing (i.e., modification of data)
- **If C_1 is garbled to, say, G then**
  - $P_1 \neq C_0 \oplus D(G, K), P_2 \neq G \oplus D(C_2, K)$
- Example: Inter-bank fund transfers
  - Confidentiality (Privacy) may be nice, integrity is critical
- Encryption provides confidentiality (prevents unauthorized expose)
- Encryption alone does not provide integrity
  - One-time pad, ECB cut-and-paste, etc.

Message Authentication Code

- Message Authentication Code (MAC)
  - Used for data integrity (preventing, or at least detecting, unauthorized "writing“)
  - Integrity not the same as confidentiality
- MAC is computed as CBC residue (***Cipher Block Chaining***)
  - That is, compute CBC encryption, saving only final ciphertext block, the MAC

CBC

MAC Computation

- MAC computation (assuming N blocks), from **CBC Mode**
- $C_0 = E(IV \oplus P_0, K),$
- $C_1 = E(C_0 \oplus P_1, K),$
- $C_2 = E(C_1 \oplus P_2, K),$
- $C_{N-1} = E(C_{N-2} \oplus P_{N-1}, K) = MAC,$
- MAC sent with IV and plaintext
- Receiver does same computation and verifies that result agrees with MAC
  - Note: receiver must know the key K

Does a MAC work?

- Suppose Alice has 4 plaintext blocks
- Alice computes
  - $C_0 = E(IV \oplus P_0, K)$, $C_1 = E(C_0 \oplus P_1, K)$,
  - $C_2 = E(C_1 \oplus P_2, K)$, $C_3 = E(C_2 \oplus P_3, K) = MAC$,
- Alice sends IV, P_0, P_1, P_2, P_3 and MAC to Bob
- Suppose Trudy changes P_1 $to\ X$
- Bob computes
  - $C_0 = E(IV \oplus P_0, K)$, $C_1 = E(C_0 \oplus X, K)$,
  - $C_2 = E(C_1 \oplus P_2, K)$, $C_3 = E(C_2 \oplus P_3, K) = MAC \neq MAC$,
- That is, error propagates into MAC, so Trudy can't make $MAC == MAC$ without K

Confidentiality and Integrity

- Encrypt with one key, MAC with another key
- Why not use the same key?
  - Send last encrypted block (MAC) twice?
  - This cannot add any security!
- Using different keys to encrypt and compute MAC works, even if keys are related
  - But, twice as much work as encryption alone
  - Can do a little better
  - Confidentiality and integrity with same work as one encryption is a research topic

Uses for Symmetric Crypto

- Confidentiality
  - Transmitting data over insecure channel
  - Secure storage on insecure media
- Integrity (MAC)
- Authentication protocols
- Anything you can do with a hash function

... Thank you ...