Block Cipher

The Workhorse of Crypto

Block Cipher

- Block ciphers take as input a block of plaintext and generate the corresponding ciphertext.
- Fundamental goal of a block cipher is to provide data confidentiality.
- They also serve as a primary building block for other cryptographic primitives like MAC, hash functions etc.
- Symmetric key block ciphers are extremely fast.
- Example:
  1. Data Encryption Standard.
  2. Advanced Encryption Standard.

Definition

- A deterministic cryptosystem $\mathcal{E} = (E; D)$
  - Message space and ciphertext space: a finite set $\mathcal{X}$
  - Key space: $\mathcal{K}$.
  - $\mathcal{E}$ is a block cipher defined over $(\mathcal{K}; \mathcal{X})$.
- For every key $k \in \mathcal{K}$, define $f_k := E(k;)$

$$f_k : \mathcal{X} \longrightarrow \mathcal{X}$$

- For correctness of decryption what property f_k needs?

Definition

- A deterministic cryptosystem $\mathcal{E} = (E; D)$
  - Message space and ciphertext space: a finite set $\mathcal{X}$
  - Key space: $\mathcal{K}$.
  - $\mathcal{E}$ is a block cipher defined over $(\mathcal{K}; \mathcal{X})$.
- For every key $k \in \mathcal{K}$, define $f_k := E(k;)$

$$f_k : \mathcal{X} \longrightarrow \mathcal{X}$$

- For correctness of decryption what property f_k needs?
  - f_k must be a permutation on $\mathcal{X}$
  - $D(k;)$ is the inverse function f_k^{-1}.

Security

- What security property should $\mathcal{E}$ satisfy?

Security

- What security property should $\mathcal{E}$ satisfy?
- Computationally indistinguishable from a random permutation.
- Suppose: block-size = key-size = 128-bits
  - How many permutation functions are possible?
  - How many f_k are possible?

Security

- What security property should $\mathcal{E}$ satisfy?
- Computationally indistinguishable from a random permutation.
- Suppose: block-size = key-size = 128-bits
  - How many permutation functions are possible?
  - How many f_k are possible?
- Claim:
  - A secure block cipher is unpredictable.
  - Unpredictability implies key recovery is infeasible.

Block Cipher: Desirable Properties

- Security:
  1. Confusion: relationship between the key and the ciphertext should be complicated.
  2. Diffusion: every single ciphertext bit should depend on all the plaintext bits.
  3. Keysize: small enough to manage *but* large to make exhaustive search infeasible.
- Efficiency:
  1. Encryption and decryption rate should be high.
  2. Easy to implement (and analyze).
  3. Suitable for hardware and/or software.

Data Encryption Standard (DES)

- The first commercially available modern cipher with fully specified implementation details in the open literature.
  - Note: Design principles are still classified.
- In the early 70s US National Bureau of Standards (now NIST) solicited proposals for encryption algorithms to protect computer data.
- IBM's submission was later adopted as DES.
- In the early eighties DES was adopted as a US Banking Standard and used widely all over the world.
- DES has a block size of 64-bits and key size of 56-bits.
- NSA (allegedly) forced the keysize to be restricted to 56-bits.
  - In 1977 Diffie and Hellman suggested that a special purpose machine can be built to exhaustively search the keyspace of DES at an estimated cost of USD 20M.

Feistel Network/Cipher

- Parameters:
  1. Block length: $2n$-bits (divided into two equal halves).
  2. Key size: ℓ-bits.
  3. Number of rounds: r.
- $\mathcal{M} =?$, $\mathcal{C} =?$ and $\mathcal{K} =?$

Feistel Network/Cipher

- Parameters:
  1. Block length: $2n$-bits (divided into two equal halves).
  2. Key size: ℓ-bits.
  3. Number of rounds: r.
- $\mathcal{M} = ?$, $\mathcal{C} = ?$ and $\mathcal{K} = ?$
- Key Scheduling Algorithm: Derive ℓ'-bit "subkeys" $k_1, k_2, \ldots, k_r$ from the secret key k.
- Round function: Each subkey defines a round function:

$$f_i : \{0,1\}^n \times \{0,1\}^{\ell'} \rightarrow \{0,1\}^n$$

- Such a block cipher is called iterated block cipher.

Encryption/Decryption

Encryption proceeds through r rounds.

- Divide the $2n$-bit message into two equal halves: $m = (m_0, m_1)$.
- Round 1: $(m_0, m_1) \rightarrow (m_1, m_2)$ where $m_2 = m_0 \oplus f_1(m_1, k_1)$.
- Round 2: $(m_1, m_2) \rightarrow (m_2, m_3)$ where $m_3 = m_1 \oplus f_2(m_2, k_2)$.
- $\ldots$
- Round r: $(m_{r-1}, m_r) \rightarrow (m_r, m_{r+1})$ where $m_{r+1} = m_{r-1} \oplus f_r(m_r, k_r)$.
- Ciphertext: $c = (m_{r+1}, m_r)$.

Decryption: ?

Encryption/Decryption

Encryption proceeds through r rounds.

- Divide the $2n$-bit message into two equal halves: $m = (m_0, m_1)$.
- Round 1: $(m_0, m_1) \to (m_1, m_2)$ where $m_2 = m_0 \oplus f_1(m_1, k_1)$.
- Round 2: $(m_1, m_2) \to (m_2, m_3)$ where $m_3 = m_1 \oplus f_2(m_2, k_2)$.
- $\dots$
- Round r: $(m_{r-1}, m_r) \to (m_r, m_{r+1})$ where $m_{r+1} = m_{r-1} \oplus f_r(m_r, k_r)$.
- Ciphertext: $c = (m_{r+1}, m_r)$.

Decryption: ?

Same process with keys reversed!

- Given $c = (m_{r+1}, m_r)$, compute $m_{r-1} = m_{r+1} \oplus f_r(m_r, k_r)$.
- Then compute $m_{r-2}, \dots, m_1, m_0$.

Feistel Cipher: Few Features

- The encryption must be invertible.
  - The round function f_i must be invertible. [True/False?]

Feistel Cipher: Few Features

- The encryption must be invertible.
  - The round function f_i must be invertible. [True/False?]
- Implementation:
  - Encryption: Implement just one round and then reuse the code for the other rounds.
  - Decryption: The same code for encryption can be reused with the subkeys used in reverse order.
- DES is an example of Feistel cipher with $n = 32$, $r = 16$ and $\ell = 56$.

Security: Small Key Size

- Exhaustive key search requires only 2^{56} steps and can be easily parallelized.
- DES Challenge from RSA Security: given three pairs of (m, c) find the corresponding key.
    1. [1997] The first challenge was broken in 96 days.
    2. [1998] The second challenge was broken in 56 hours by Deep Crack machine of Electronic Frontier Foundation (EFF).
    3. [1999] The third challenge was broken in 22 hours 15 minutes by Deep Crack and a network of around 100,000 computers.
- See www.distributed.net if you're interested!

Security: Small Block Size

- DES is a deterministic encryption scheme.
  - Same message encrypted under the same key always gives the same ciphertext.
- If plaintext blocks are distributed uniformly at random then we can expect a collision with a high probability after observing 2^{32} ciphertext blocks.
  - Ciphertext reveals some information about the underlying plaintext.

Multiple Encryption

- Re-encrypt the ciphertext once (or more) using independent keys.
- Double-DES:
  - Key: (k_1, k_2).
  - Encryption: $E_{k_2}(E_{k_1}(m))$, E is DES.
  - Key length is now double (112-bits) but block length remains 64-bits.

Multiple Encryption

- Re-encrypt the ciphertext once (or more) using independent keys.
- Double-DES:
  - Key: (k_1, k_2).
  - Encryption: $E_{k_2}(E_{k_1}(m))$, E is DES.
  - Key length is now double (112-bits) but block length remains 64-bits.
- Does multiple encryption always give increased security?

Multiple Encryption

- Re-encrypt the ciphertext once (or more) using independent keys.
- Double-DES:
  - Key: (k_1, k_2).
  - Encryption: $E_{k_2}(E_{k_1}(m))$, E is DES.
  - Key length is now double (112-bits) but block length remains 64-bits.
- Does multiple encryption always give increased security?
- Fix a DES key k, $E_k : \{0,1\}^{64} \rightarrow \{0,1\}^{64}$ defines a permutation.
- The 2^{56} keys define 2^{56} such potentially different permutations.
- What if given any two k_1, k_2 there exists a k_3 s.t $E_{k_3}(m) = E_{k_2}(E_{k_1}(m))$?

Multiple Encryption

- Re-encrypt the ciphertext once (or more) using independent keys.
- Double-DES:
  - Key: (k_1, k_2).
  - Encryption: $E_{k_2}(E_{k_1}(m))$, E is DES.
  - Key length is now double (112-bits) but block length remains 64-bits.
- Does multiple encryption always give increased security?
- Fix a DES key k, $E_k : \{0,1\}^{64} \rightarrow \{0,1\}^{64}$ defines a permutation.
- The 2^{56} keys define 2^{56} such potentially different permutations.
- What if given any two k_1, k_2 there exists a k_3 s.t $E_{k_3}(m) = E_{k_2}(E_{k_1}(m))$?
- **[Fact:]** The set of 2^{56} permutations defined by 2^{56} DES keys is not closed under functional composition.

Double DES

- Key: (k_1, k_2).
- Encryption: $c = E_{k_2}(E_{k_1}(m))$, E is DES encryption.
- Decryption: $m = E_{k_1}^{-1}(E_{k_2}^{-1}(c))$.
- Key length is now double (112-bits) – exhaustive key search is infeasible.
- Block length remains 64-bits.

Note: DES is an endomorphic cryptosystem, $\mathcal{P} = \mathcal{C}$.

Is Double-DES more secure than DES?

Attacking 2-DES: *Meet-in-the-Middle*

$c = E_{k_2}(E_{k_1}(m))$ thus $E_{k_2}^{-1}(c) = E_{k_1}(m)$.

Attacking 2-DES: *Meet-in-the-Middle*

$c = E_{k_2}(E_{k_1}(m))$ thus $E_{k_2}^{-1}(c) = E_{k_1}(m)$.
Input: 3 known plaintext/ciphertext pairs $(m_1, c_1), (m_2, c_2), (m_3, c_3)$.
Output: The secret key (k_1, k_2).

Attacking 2-DES: *Meet-in-the-Middle*

$c = E_{k_2}(E_{k_1}(m))$ thus $E_{k_2}^{-1}(c) = E_{k_1}(m)$.

Input: 3 known plaintext/ciphertext pairs $(m_1, c_1), (m_2, c_2), (m_3, c_3)$.

Output: The secret key (k_1, k_2).

1. For each $h_2 \in \{0, 1\}^{56}$:
   - Compute $E_{h_2}^{-1}(c_1)$ and store $[E_{h_2}^{-1}(c_1), h_2]$ in a table $\mathbb{T}$ sorted by the first component.

Attacking 2-DES: *Meet-in-the-Middle*

$c = E_{k_2}(E_{k_1}(m))$ thus $E_{k_2}^{-1}(c) = E_{k_1}(m)$.
Input: 3 known plaintext/ciphertext pairs $(m_1, c_1), (m_2, c_2), (m_3, c_3)$.
Output: The secret key (k_1, k_2).

1. For each $h_2 \in \{0,1\}^{56}$:
   - Compute $E_{h_2}^{-1}(c_1)$ and store $[E_{h_2}^{-1}(c_1), h_2]$ in a table $\mathbb{T}$ sorted by the first component.
2. For each $h_1 \in \{0,1\}^{56}$ do the following:
   2.1 Compute $E_{h_1}(m_1)$
   2.2 Search for $E_{h_1}(m_1)$ in $\mathbb{T}$ ($E_{h_1}(m_1)$ matches table entry $[E_{h_2}^{-1}(c_1), h_2]$ if $E_{h_1}(m_1) = E_{h_2}^{-1}(c_1)$.)
   2.3 For each match $[E_{h_2}^{-1}(c_1), h_2]$ in the table, check whether $E_{h_2}(E_{h_1}(m_2)) = c_2$; if so then check whether $E_{h_2}(E_{h_1}(m_3)) = c_3$.
   2.4 If both checks pass, then output (h_1, h_2) and STOP.

Why 3 Plaintext/Ciphertext Pairs?

- Suppose E is a block cipher with key space $\mathcal{K} = \{0,1\}^\ell$, and plaintext/ciphertext space $\mathcal{P} = \mathcal{C} = \{0,1\}^n$.
- Suppose $k' \in \mathcal{K}$ is the secret key and (m_i, c_i), $1 \leq i \leq t$ are the known plaintext/ciphertext pairs, where m_is are all distinct.
  - $c_i = E_{k'}(m_i)$ for all $1 \leq i \leq t$.
- What should be the value of t to ensure (with very high probability) that there is only one key $k' \in \mathcal{K}$ such that $E_{k'}(m_i) = c_i$ for all $1 \leq i \leq t$?

Why 3-Pairs of PT/CT is Enough

- For each $k \in \mathcal{K}$, $E_k : \{0,1\}^n \to \{0,1\}^n$ is a permutation.

Why 3-Pairs of PT/CT is Enough

- For each $k \in \mathcal{K}$, $E_k : \{0,1\}^n \to \{0,1\}^n$ is a permutation.
- [Heuristic Assumption] For each $k \in \mathcal{K}$, E_k is a random function (i.e., a randomly selected function).
  1. The assumption is not correct as E_k is not random and a random function is almost certainly not a permutation.
  2. However, the assumption turns out to be quite good for the analysis!

Why 3-Pairs of PT/CT is Enough

- For each $k \in \mathcal{K}$, $E_k : \{0,1\}^n \to \{0,1\}^n$ is a permutation.
- [Heuristic Assumption] For each $k \in \mathcal{K}$, E_k is a random function (i.e., a randomly selected function).
    1. The assumption is not correct as E_k is not random and a random function is almost certainly not a permutation.
    2. However, the assumption turns out to be quite good for the analysis!
- Fix any $k \in \mathcal{K}$ s.t. $k \neq k'$ (k' is the unknown key we want).
- Probability that $E_k(m_i) = c_i$ for all $1 \leq i \leq t$ is

$$\frac{1}{2^n} \cdot \frac{1}{2^n} \cdots \frac{1}{2^n} = \frac{1}{2^{nt}}$$

- The expected number of keys $k \in \mathcal{K}$ (excluding k') so that $E_k(m_i) = c_i$ for all $1 \leq i \leq t$ is:

$$E_{\mathcal{K}} = \frac{2^{\ell} - 1}{2^{nt}}$$

Analysis of *Meet-in-the-Middle* Attack

Double-DES Encryption: $c = E_{k_2}(E_{k_1}(m))$.

Goal: Find (k_1, k_2).

Here $\ell = 112$-bits and $n = 64$-bits.

1. For $t = 1$, $E_{\mathcal{K}} \approx 2^{48}$.
   - Expected number of Double-DES keys (h_1, h_2) s.t. $E_{h_2}(E_{h_1}(m_1)) = c_1$ is 2^{48}.

Analysis of *Meet-in-the-Middle* Attack

Double-DES Encryption: $c = E_{k_2}(E_{k_1}(m))$.
Goal: Find (k_1, k_2).
Here $\ell = 112$-bits and $n = 64$-bits.

1. For $t = 1$, $E_{\mathcal{K}} \approx 2^{48}$.
   - Expected number of Double-DES keys (h_1, h_2) s.t. $E_{h_2}(E_{h_1}(m_1)) = c_1$ is 2^{48}.
2. Among these 2^{48} keys, the expected number of keys that also satisfy $E_{h_2}(E_{h_1}(m_2)) = c_2$ is *approximately* $\frac{2^{48}}{2^{64}} = \frac{1}{2^{16}}$.

Analysis of *Meet-in-the-Middle* Attack

Double-DES Encryption: $c = E_{k_2}(E_{k_1}(m))$.
Goal: Find (k_1, k_2).
Here $\ell = 112$-bits and $n = 64$-bits.

1. For $t = 1$, $E_{\mathcal{K}} \approx 2^{48}$.
   - Expected number of Double-DES keys (h_1, h_2) s.t. $E_{h_2}(E_{h_1}(m_1)) = c_1$ is 2^{48}.
2. Among these 2^{48} keys, the expected number of keys that also satisfy $E_{h_2}(E_{h_1}(m_2)) = c_2$ is *approximately* $\frac{2^{48}}{2^{64}} = \frac{1}{2^{16}}$.
3. For $t = 3$, we have $E_{\mathcal{K}} \approx \frac{1}{2^{80}}$. If we find (h_1, h_2) s.t. $E_{h_2}(E_{h_1}(m_i)) = c_i$ for $1 \leq i \leq 3$ then with a very high probability $(h_1, h_2) = (k_1, k_2)$.

Resource Requirement

- Number of DES evaluation: $2^{56} + 2^{56} + 2 \times 2^{48} \approx 2^{57}$.
- Storage requirement: $2^{56}(64 + 56)\text{bits} \approx 10^6\text{TB}$.

Conclusion: The effective key-length for Double-DES is essentially same as DES.

- Double-DES is not significantly secure than DES.

Claim: The memory requirement in the attack can be reduced at the expense of time – Time Memory Trade-Off Attack:
Time: 2^{56+t} steps, Memory: 2^{56-t} units, $i \leq t \leq 55$.
Ref: A Cryptanalytic Time-Memory Trade-Off by Martin Hellman.

Triple-DES

- Key: (k_1, k_2, k_3), where $k_1, k_2, k_3 \in_R \{0,1\}^{56}$.
- Encryption: $c = E_{k_3}(E_{k_2}(E_{k_1}(m)))$ where E is the DES Encryption function.
- Decryption: $m = E_{k_1}^{-1}(E_{k_2}^{-1}(E_{k_3}^{-1}(c)))$.
- Key length is 168-bits and Block length is 64-bits.

Triple-DES

- Key: (k_1, k_2, k_3), where $k_1, k_2, k_3 \in_R \{0,1\}^{56}$.
- Encryption: $c = E_{k_3}(E_{k_2}(E_{k_1}(m)))$ where E is the DES Encryption function.
- Decryption: $m = E_{k_1}^{-1}(E_{k_2}^{-1}(E_{k_3}^{-1}(c)))$.
- Key length is 168-bits and Block length is 64-bits.
- Dictionary Attack: Adversary stores a large table ($\leq 2^{64}$) of plaintext-ciphertext pair.
  - Counter-measure: Change secret key periodically.
- Meet-in-the-Middle Attack: Takes approx. 2^{112} steps. [Exercise]
- Sweet32: Birthday attack demonstrated in 2016 exploiting the 64-bit block size.
- Triple-DES is still in use though it is suggested to be replaced by AES.

Iterated Block Cipher: Key Idea

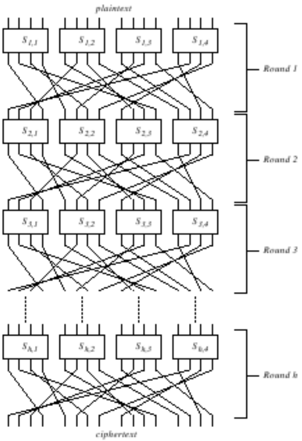
The basic design principle:

1. Round Cipher: a simple block cipher $(\hat{E}, \hat{D})$
   - DES: $\hat{E}_k(x||y) = (y||x \oplus f(k, y))$
   - This one round cipher is obviously insecure!
2. Key Expansion: Use a simple function to expand the key k to r round keys $k_1, k_2, \ldots, k_r$.

- Challenge: Design a round cipher which is very fast and gives a secure block cipher within a few rounds.
- A linear function cannot be used to get a secure block cipher.

Substitution-Permutation Network (SPN)

Iterated block cipher where each round consists of a substitution and a permutation.



SPN: Round Keys

- The secret key k is used to derive round keys $k_1, k_2, \ldots, k_h, k_{h+1}$ (h: number of rounds).
  - In the i-th round, k_i is XOR-ed with the input before applying the substitution.
- The output of the last round is XOR-ed with k_{h+1} to generate the ciphertext.
  - Prevents the adversary from attempting to decrypt the ciphertext by undoing the final substitution and permutation operations.
- Whitening: Internal state of the cipher is protected by k_1 and k_{h+1}.

SPN: Encryption/Decryption

Encryption:

$A \leftarrow$ plaintext
For $i = 1 \ldots h$ do:
  $A \leftarrow A \oplus k_i$
  $A \leftarrow S(A)$
  $A \leftarrow P(A)$
$A \leftarrow A \oplus k_{h+1}$
ciphertext $\leftarrow A$

Decryption: Just the reverse.

Random Permutation

- A block cipher is a permutation: $F_k : \{0,1\}^n \to \{0,1\}^n$.
  - Ideally it should be a truly random permutation.
  - How many bits do you need to represent a random permutation on n-bits?

Random Permutation

- A block cipher is a permutation: $F_k : \{0,1\}^n \to \{0,1\}^n$.
  - Ideally it should be a truly random permutation.
  - How many bits do you need to represent a random permutation on n-bits?
    $\approx n \cdot 2^n$ (not practical when $n \geq 64$).

Random Permutation

- A block cipher is a permutation: $F_k : \{0,1\}^n \to \{0,1\}^n$.
  - Ideally it should be a truly random permutation.
  - How many bits do you need to represent a random permutation on n-bits?
    $\approx n \cdot 2^n$ (not practical when $n \geq 64$).
- Build a "random looking" permutation for large block-length from smaller random/random-looking permutations.
- Example:
  - For $n = 64$, $F_k(x) = f_1(x_1)f_2(x_2)\cdots f_8(x_8)$ where the secret key k determines 8 random permutations on 8-bits.
  - Can you distinguish $F_k(x)$ from the output of a truly random permutation?

Mixing

- The output bits of $F_k(x)$ are re-ordered/mixed.
  - $x' \leftarrow F_k(x)$, then permute the bits of x' to obtain x_1.

Mixing

- The output bits of $F_k(x)$ are re-ordered/mixed.
  - $x' \leftarrow F_k(x)$, then permute the bits of x' to obtain x_1.
- Recall one round of SPN:
  $A \leftarrow A \oplus k_i$
  $A \leftarrow S(A)$
  $A \leftarrow P(A)$

Mixing

- The output bits of $F_k(x)$ are re-ordered/mixed.
  - $x' \leftarrow F_k(x)$, then permute the bits of x' to obtain x_1.
- Recall one round of SPN:
  $A \leftarrow A \oplus k_i$
  $A \leftarrow S(A)$
  $A \leftarrow P(A)$
- Obviously one round is insecure. (Recover the secret key given one input/output pair.)
- Repeat the rounds several times. Hopefully small changes in input will have significant change in the output.

SPN Structure

- The smaller permutations f_i act as fixed substitution function or S-boxes.
- Unlike the Feistel Network, the S-boxes must be invertible in SPN.
- Avalanche effect: small change in input should result in large change to the output.
- SPN with several rounds achieve avalanche effect if
  1. S-boxes are so designed that changing a single input bit changes at least two bits in the output of the S-box.
  2. Output bits of any particular S-box is spread across different S-boxes in the next round by the mixing permutation.

Advanced Encryption Standard

- AES is an SPN but the permutation operation is replaced by two linear transformations (one of them is a permutation).
- All operations are byte oriented.
  - S-box maps 8-bits to 8-bits.
  - Allows efficient implementation on various platforms.
- Block size of AES is 128-bits.
- Each round key is also 128-bits.
- AES accepts three different key lengths and the number of rounds depends on the key length:
  1. 128-bit key: 10 rounds.
  2. 192-bit key: 12 rounds.
  3. 256-bit key: 14 rounds.

Performance

	Key Size (bits)	Block Size (bits)	No. of Rounds	Performance (MB/sec)
DES	56	64	16	80
3DES	168	64	48	30
AES-128	128	128	10	163
AES-256	256	128	14	115

On Intel Xeon CPU E5-2698 v3 at 2.30GHz.

- Several processors provide special instruction sets for AES, called AES-NI leading to significant speed-up.
- AES-128-NI $\approx$ 2400 MB/sec and AES-256-NI $\approx$ 1800 MB/sec

Encrypting Large Messages

- You've a secure block cipher and a confidential message.
  - Suppose the block cipher has a block length of n-bits and (for simplicity) the message length is some multiple of n, say, jn-bits.
- How will you generate the ciphertext?

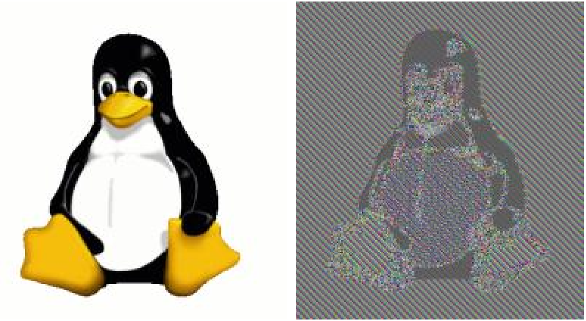
Encrypting Large Messages

- You've a secure block cipher and a confidential message.
  - Suppose the block cipher has a block length of n-bits and (for simplicity) the message length is some multiple of n, say, jn-bits.
- How will you generate the ciphertext?
  - Divide the message into j blocks each of length n-bits.
  - Encrypt the message blocks in sequence and return the ciphertext.
  - This is called Electronic Codebook mode (ECB).

Encrypting Large Messages

- You've a secure block cipher and a confidential message.
  - Suppose the block cipher has a block length of n-bits and (for simplicity) the message length is some multiple of n, say, jn-bits.
- How will you generate the ciphertext?
  - Divide the message into j blocks each of length n-bits.
  - Encrypt the message blocks in sequence and return the ciphertext.
  - This is called Electronic Codebook mode (ECB).
- Is ECB a secure mode of encrypting your confidential data?

A picture is worth a thousand words!



Encryption by ECB mode [Source: Wikipedia].

Zoom at the beginning of the Pandemic used ECB for encryption!
See: Move Fast and Roll Your Own Crypto: A Quick Look at the Confidentiality of Zoom Meetings [The Citizen Lab report, April 2020]

Cipher Block Chaining (CBC) Mode

Encrypt:

- Start with an initialization vector $\mathrm{IV} \in_R \{0,1\}^n$ and set $c_0 = \mathrm{IV}$.
- Compute $c_i = E_k(m_i \oplus c_{i-1})$ for $1 \leq i \leq j$.
- Ciphertext: $c_0, c_1, \ldots, c_j$. Note: IV is sent in the clear.

Decrypt:

- Compute $m_i = E_k^{-1}(c_i) \oplus c_{i-1}$, for $1 \leq i \leq j$.

Cipher Block Chaining (CBC) Mode

Encrypt:

- Start with an initialization vector $\mathrm{IV} \in_R \{0,1\}^n$ and set $c_0 = \mathrm{IV}$.
- Compute $c_i = E_k(m_i \oplus c_{i-1})$ for $1 \leq i \leq j$.
- Ciphertext: $c_0, c_1, \ldots, c_j$. Note: IV is sent in the clear.

Decrypt:

- Compute $m_i = E_k^{-1}(c_i) \oplus c_{i-1}$, for $1 \leq i \leq j$.

Note:

1. The IV shouldn't be predictable.
2. Identical plaintexts with different IV results in different ciphertexts.
3. Any change in a plaintext block (m_i) affects $c_i, c_{i+1}, \ldots$.
    - Useful in the construction of a message authentication code (MAC).

Other Modes of Operation

- Output feedback mode (OFB): An initialization vector IV is encrypted repeatedly using the block cipher to generate a key stream – actually a stream cipher.
- Cipher feedback mode (CFB): Also a stream cipher.
  - $c_0 = \mathrm{IV}$.
  - $z_i = E_k(c_{i-1})$.
  - $c_i = m_i \oplus z_i$.
- Counter mode
  - Select a counter ctr and increment it as $T_i = ctr + i - 1 \bmod 2^n$ where n is the block length.
  - $z_i = E_k(T_i)$ and $c_i = m_i \oplus z_i$.
  - Allows parallelism.

Reference

1. Video Lectures by Alfred Menezes:
   https://cryptography101.ca/crypto101-building-blocks
   - Chapter V2 for Symmetric Key Encryption
2. Boneh and Shoup [draft]: Chapter 3 for Stream Cipher and Chapter 4 for Block Cipher.