One Time Pad, Block Ciphers, Encryption Modes

Ahmet Burak Can
Hacettepe University
abc@hacettepe.edu.tr
Basic Ciphers

• Shift Cipher
  ◦ Brute-force attack can easily break

• Substitution Cipher
  ◦ Frequency analysis can reduce the search space

• Vigenere Cipher
  ◦ Kasiski test can reveal the length of key

• Enigma Machine
  ◦ The capture of the daily codebook

• How perfect secrecy can be satisfied?
One Time Pad

• Basic Idea: Extend Vigenère cipher so that the key is as long as the plaintext
  ◦ Key is a random string and is used only once
  ◦ Encryption is similar to Vigenère
  ◦ Cannot be broken by frequency analysis or Kasiski test

Plaintext \( P = (p_1 \ p_2 \ ... \ p_n) \)
Key \( K = (k_1 \ k_2 \ ... \ k_n) \)
Ciphertext \( C = (p_1 \ p_2 \ ... \ p_n) \)

\( E_k(X) = (p_1+k_1 \ p_2+k_2 \ ... \ p_n+k_n) \mod m \)
\( D_k(Y) = (c_1-k_1 \ c_2-k_2 \ ... \ c_n-k_n) \mod m \)
The Binary Version of One-Time Pad

- Plaintext space = Ciphertext space = Keyspace = \{0, 1\}^n
- Key is chosen randomly
- For example:
  - Plaintext 11011011
  - Key 01101001
  - Ciphertext 10110010
Security of One Time Pad

• How good is the security of one time pad?
  ◦ The key is random, so ciphertext is completely random
  ◦ Any plaintext can correspond to a ciphertext with the same length

• A scheme has perfect secrecy if ciphertext provides no “information” about plaintext
  ◦ C. E. Shannon, 1949

• One-time pad has perfect secrecy
  ◦ For example, suppose that the ciphertext is “Hello”, can we say any plaintext is more likely than another plaintext?
Importance of Key Randomness

- For perfect secrecy, key-length ≥ msg-length
- What if a One-Time Pad key is not chosen randomly, instead, texts from, e.g., a book is used.
  - this is not One-Time Pad anymore
  - this does not have perfect secrecy and can be broken
- The key in One-Time Pad should never be reused.
  - If it is reused, it is insecure!
  - How to send the key to the receiver of the ciphertext?

- These requirements make One Time Pad impractical.
Block Ciphers

1. Block Cipher = Symmetric key encryption = Conventional Encryption
2. Block ciphers can be considered as substitution ciphers with large block size ($\geq 64$ bits)
3. Map $n$-bit plaintext blocks to $n$-bit ciphertext blocks ($n$: block size).
   - For $n$-bit plaintext and ciphertext blocks and a fixed key, the encryption function is a one-to-one function
Block Ciphers

- **Block size**: in general larger block sizes mean greater security.
- **Key size**: larger key size means greater security (larger key space).
- **Number of rounds**: multiple rounds offer increasing security.
- **Encryption modes**: define how messages larger than the block size are encrypted, very important for the security of the encrypted message.
A Simple Block Cipher: Hill Cipher

- The key $k$ is a matrix. The message is considered as vectors. Encryption and decryption operations are matrix multiplication operations
  - Encryption: $C = k \cdot P \pmod{26}$
  - Decryption: $P = k^{-1} \cdot C \pmod{26}$

- Example: The plaintext is `CAT` converted to numeric values, namely 2, 0, 19.

- If the key is
  
  $\begin{pmatrix}
  6 & 24 & 1 \\
  13 & 16 & 10 \\
  20 & 17 & 15
  \end{pmatrix}$

- Encryption: $\begin{pmatrix}
  6 & 24 & 1 \\
  13 & 16 & 10 \\
  20 & 17 & 15
  \end{pmatrix} \begin{pmatrix}
  2 \\
  0 \\
  19
  \end{pmatrix} \equiv \begin{pmatrix}
  31 \\
  216 \\
  325
  \end{pmatrix} \equiv \begin{pmatrix}
  5 \\
  8 \\
  13
  \end{pmatrix} \pmod{26}$

- $C='FIN'$
An Insecure Block Cipher

- Hill cipher is insecure since it uses linear matrix operations.
  - Each output bit is a linear combination of the input bits
  - An insecure block cipher uses linear equations
- Hill Cipher can easily be broken by known-plaintext attack
  - An attacker knowing a plaintext and ciphertext pair can easily figure out the key matrix.
Feistel Network

- A Feistel Network is fully specified given
  - the block size: \( n = 2^w \)
  - number of rounds: \( d \)
  - \( d \) round functions \( f_1, f_2, \ldots, f_d: \{0,1\}^w \rightarrow \{0,1\}^w \)
  - Each \( f \) function is a SP cipher

- Used in DES, IDEA, RC5, and many other block ciphers.
- Not used in AES
Feistel Network

- **Encryption**
  
  \[ \begin{align*}
  L_1 &= R_0 \\
  R_1 &= L_0 \oplus f_0(R_0) \\
  L_2 &= R_1 \\
  R_2 &= L_1 \oplus f_1(R_1) \\
  \vdots \\
  L_d &= R_{d-1} \\
  R_d &= L_{d-1} \oplus f_{d-1}(R_{d-1})
  \end{align*} \]

- **Decryption**

  \[ \begin{align*}
  R_{d-1} &= L_d \\
  L_{d-1} &= R_d \oplus f_{d-1}(L_d) \\
  \vdots \\
  R_0 &= L_1 \\
  L_0 &= R_1 \oplus f_0(L_1)
  \end{align*} \]
History of Data Encryption Standard (DES)

- 1967: Feistel at IBM
  - Lucifer: block size 128; key size 128 bit
- 1972: NBS asks for an encryption standard
- 1975: IBM developed DES (modification of Lucifer)
  - block size 64 bits; key size 56 bits
- 1975: NSA suggests modification
- 1977: NBS adopts DES as encryption standard in (FIPS 46-1, 46-2).
- 2001: NIST adopts Rijndael (AES) as replacement to DES.
DES Features

- Features:
  - Block size = 64 bits
  - Key size = 56 bits
  - Number of rounds = 16
  - 16 intermediary keys, each 48 bits
DES Structure

Diagram showing the structure of DES, including initial permutation, rounds (Round 1 to Round 16), permutation choices, and left circular shift.
Details of DES Rounds

- An initial permutation is applied on the plaintext
  \[ IP(x) = L_0 R_0 \]

- In each round:
  \[ L_i = R_{i-1} \]
  \[ R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \]
Details of DES Rounds

- After the last round

\[ y = \text{IP}^{-1}(R_{16}L_{16}) \]
DES f Function
DES S-boxes

- S-boxes are the only non-linear elements in DES design

\[ B(6\ \text{bits}) \rightarrow \text{S-Box} \rightarrow \ C(4\ \text{bits}) \]

- \( B = b_1b_2b_3b_4b_5b_6 \quad \text{row}=b_1b_6 \quad \text{column}=b_2b_3b_4b_5 \)
- Example: \( B = 011011 \quad \text{row}=01 \quad \text{column}=1101 \)

\[
\begin{array}{cccccccccccccccc}
S & 0000 & 0001 & 0010 & 0011 & 0100 & 0101 & 0110 & 0111 & 1000 & 1001 & 1010 & 1011 & 1100 & 1101 & 1110 & 1111 \\
00 & 0100 & 1100 & 0100 & 0001 & 0111 & 1010 & 1011 & 0110 & 1000 & 0101 & 0011 & 1111 & 1101 & 0000 & 1110 & 1001 \\
01 & 1110 & 1011 & 0010 & 1100 & 0100 & 0111 & 1101 & 0001 & 0101 & 0000 & 1111 & 1101 & 0110 & 0011 & 1001 & 1000 & 0110 \\
10 & 0100 & 0010 & 0001 & 1011 & 1010 & 1101 & 0111 & 1000 & 1111 & 1001 & 1100 & 0101 & 0110 & 0011 & 0000 & 1110 & 1110 \\
11 & 1011 & 1000 & 1100 & 0111 & 0001 & 1110 & 0010 & 1101 & 0110 & 1111 & 0000 & 1001 & 1010 & 0100 & 0101 & 0011 & \ \\
\end{array}
\]

\(C = 1001\)
DES Weak Keys

- **Weak keys**: keys make the same sub-key to be generated in more than one round.
  - Result: reduce cipher complexity
  - Weak keys can be avoided at key generation. DES has 4 weak keys:
    - \[0000000 \ 0000000\]
    - \[0000000 \ FFFFFFF\]
    - \[FFFFFFF \ 0000000\]
    - \[FFFFFFF \ FFFFFFF\]

- **Semi-weak keys**: A pair of DES semi-weak keys is a pair \((K_1,K_2)\) with \(E_{K_1}(E_{K_2}(x))=x\)

- There are six pairs of DES semi-weak keys
Dictionary Attack to DES

- Even without having weak/semi-weak keys DES is vulnerable to **dictionary attacks**:  
- Each plaintext may result in $2^{64}$ different ciphertexts, but there are only $2^{56}$ possible different key values.
- Given a PT/CT pair $(M,C)$
  - Encrypt the known plaintext $M$ with all possible keys.
  - Keep a look up table of size $2^{56}$.
  - Look up $C$ in the table
Double DES

- DES uses a 56-bit key, this raised concerns about brute force attacks.
- One proposed solution: double DES.
- Apply DES twice using two keys, K1 and K2.
  - $C = E_{K2} [ E_{K1} [ P ]]$
  - $P = D_{K1} [ D_{K2} [ C ]]$
- This leads to a $2 \times 56 = 112$ bit key, so it is more secure than DES. Is it?
Meet-in-the-middle Attack

- Goal: given the pair \((P, C)\) find keys \(K_1\) and \(K_2\).
- Based on the observation:
  \[
  C = E_{K_2}[E_{K_1}[P]]
  \]
  \[
  D_{K_2}[C] = E_{K_1}[P]
  \]

1. Encrypt \(P\) with all \(2^{56}\) possible keys \(K_1\)
   - Store all pairs \((K_1, E_{K_1}[P])\), sorted by \(E_{K_1}[P]\).
2. Decrypt \(C\) using all \(2^{56}\) possible keys \(K_2\)
   - For each decrypted result, check to see if there is a match
     \(D_{K_2}(C) = E_{K_1}(P)\). If a match is found, \((K_1, K_2)\) is a possible match.
3. The attack has a higher chance of succeeding if another pair \((P', C')\) is available to the cryptanalysis.
Triple DES

- **Two key version is widely used and standard**
  - Key space is $56 \times 2 = 112$ bits
    - Encrypt: $C = E_{K1} \ [ \ D_{K2} \ [ \ E_{K1} \ [P] \ ] \ ]$
    - Decrypt: $P = D_{K1} \ [ \ E_{K2} \ [ \ D_{K1} \ [C] \ ] \ ]$

- **Three key version is possible but not standard**
  - Key space is $56 \times 3 = 168$ bits
    - Encrypt: $C = E_{K3} \ [ \ D_{K2} \ [ \ E_{K1} \ [P] \ ] \ ]$
    - Decrypt: $P = D_{K1} \ [ \ E_{K2} \ [ \ D_{K3} \ [C] \ ] \ ]$

- No known practical attack against it.
- Some protocols/applications use 3DES (such as PGP)
Encryption Modes

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Output Feedback Mode (OFB)
- Cipher Feedback Mode (CFB)
- Counter Mode (CTR)
Electronic Code Book (ECB)

- Message is broken into independent blocks of block_size bits.
  - Encryption: $C_i = E_k[P_i]$
  - Decryption: $P_i = D_k[C_i]$

\[ \begin{array}{ccc}
  P_1 & E & C_1 \\
  P_2 & E & C_2 \\
  P_3 & E & C_3 \\
\end{array} \]
Properties of ECB

- Deterministic: the same data block gets encrypted the same way.
  - This reveals patterns of data when a data block repeats.
- Malleable: reordering ciphertext results in reordered plaintext.
- Errors in one ciphertext block do not propagate.
- Usage: not recommended to encrypt more than one block of data.
Cipher Block Chaining (CBC)

- Cipher Block Chaining (CBC): next input depends upon previous output
  - Encryption: $C_i = E_k [P_i \oplus C_{i-1}]$, with $C_0=IV$
  - Decryption: $P_i = C_{i-1} \oplus D_k [C_i]$, with $C_0=IV$
Properties of CBC

- **Randomized encryption**: repeated text gets mapped to different encrypted data.
  - can be proven to be “secure” assuming that the block cipher has desirable properties and that random IV’s are used
- A ciphertext block depends on all preceding plaintext blocks
  - Sequential encryption, cannot use parallel hardware
- Errors in one block of ciphertext propagate to two blocks
  - one bit error in $C_j$ affects all bits in $M_j$ and one bit in $M_{j+1}$
Block Ciphers vs. Stream Ciphers

- A block cipher operates on blocks of fixed length.
- A stream cipher is a symmetric key cipher where plaintext bits are combined with a pseudorandom cipher bit stream (keystream), typically by an exclusive-or (xor) operation.
Output Feedback (OFB)

- Output feedback (OFB): construct a pseudorandom number generator (PRNG) to obtain a one time pad and XOR the message with the pad
  - Encryption: $X_0 = IV$, $X_i = E_k[X_{i-1}]$, $C_i = P_i + X_i$
  - Decryption: $X_0 = IV$, $X_i = E_k[X_{i-1}]$, $P_i = C_i + X_i$
Properties of OFB

- Randomized encryption
- Sequential encryption, but preprocessing possible
  - Generate the key before the message comes
- Error propagation limited
  - Only the changed bits are lost
- It can only be used as a stream cipher
Cipher Feedback (CFB)

- Cipher Feedback (CFB): the message is XORed with the feedback of encrypting the previous block
  - Encryption: $C_0 = IV$, $C_i = E_k[C_{i-1}] + P_i$
  - Decryption: $C_0 = IV$, $P_i = E_k[C_{i-1}] + C_i$
Counter Mode (CTR)

- Counter Mode (CTR): Another way to construct pseudo random number generator using DES
  - $X_i = E_k[\text{Counter}+i]$
  - $C_i = P_i \oplus X_i$
  - Sender and receiver share a counter value (does not need to be secret) and the secret key

![Diagram of Counter Mode (CTR)]
Properties of CTR

- **Software and hardware efficiency**: different blocks can be encrypted in parallel.
- **Preprocessing**: the encryption part can be done offline and when the message is known, just do the XOR.
- **Random Access**: decryption of a block can be done in random order, very useful for hard-disk encryption.
- **Messages of Arbitrary Length**: ciphertext is the same length with the plaintext (i.e., no IV).