Topic 2.2 Secret Key Cryptography

Agenda

- Generic block cipher
- Feistel cipher
- DES
- Modes of block ciphers
- Multiple encryptions
- Message authentication through secret key cryptography.
Generic Block Cipher

Plaintext block of length $N$ → Encrypt → Cipher block of length $N$ → Decrypt → Secret key

CSC 474/574 Dr. Peng Ning

NC State University Computer Science
Generic Block Encryption (Cont’d)

- Convert one block to another: one-to-one
- Long enough to avoid known-plaintext attack, but not too long (performance).
  - 64 bit typical
- Naïve: $2^{64}$ input values, 64 bits each
- Output should look random
  - No correlation between plaintext and ciphertext
  - Bit spreading

Generic Block Encryption (Cont’d)

- Achieve by substitution:
  - Need to know how to substitute each plaintext message.
  - How many bits for $k$-bit blocks: _________bits
- Achieve by permutation:
  - Need to know which position each bit is placed.
  - How many bits for $k$-bit blocks: _________bits
- Achieve by combinations of substitutions and permutations
  - How about $S \rightarrow P \rightarrow S \rightarrow S \rightarrow P \rightarrow \ldots$
  - How about $S \rightarrow P \rightarrow P \rightarrow S \rightarrow \ldots$
  - Lesson? ___________________________
Feistel Cipher

- Confusion
  - Make the relationship between the plaintext/key and the ciphertext as complex as possible
  - Achieved by complex substitution algorithm.

- Diffusion
  - Dissipate the statistical structure of the plaintext
  - Achieved by having each plaintext digit affect many ciphertext digit
  - Equivalently, having each ciphertext digit affected by many plaintext digit.
Feistel Cipher (cont’d)

- Alternate diffusion and confusion
- Equivalently, alternate substitution and permutation

Feistel Cipher Structure
Feistel Cipher Structure (cont’d)

Decryption:

\[
\begin{align*}
L_0 &\rightarrow R_0 \\
\text{Round 1} &\text{ F } \\
\text{Round } i &\text{ F } \\
\text{Round } n &\text{ F }
\end{align*}
\]

\[
\begin{align*}
K_n &
\end{align*}
\]

\[
\begin{align*}
L_i &\rightarrow R_i \\
K_i &
\end{align*}
\]

\[
\begin{align*}
L_n &\rightarrow R_n \\
K_1 &
\end{align*}
\]

\[
\begin{align*}
L_{n+1} &\rightarrow R_{n+1}
\end{align*}
\]

One Round Feistel Cipher

Encryption

\[
\begin{align*}
L_0 &\rightarrow R_0 \\
F &
\end{align*}
\]

\[
\begin{align*}
L_1 &\rightarrow R_1 \\
F &
\end{align*}
\]

\[
\begin{align*}
L_2 &\rightarrow R_2
\end{align*}
\]

\[
\begin{align*}
L'_{0} &\rightarrow R'_{0} \\
F &
\end{align*}
\]

\[
\begin{align*}
L'_{1} &\rightarrow R'_{1} \\
F &
\end{align*}
\]

\[
\begin{align*}
L'_{2} &\rightarrow R'_{2}
\end{align*}
\]

Decryption

\[
\begin{align*}
L_0 &\rightarrow R_0 \\
\text{Ciphertext (2w bits)} &\text{ F } \\
\text{Plaintext (2w bits)} &
\end{align*}
\]
Realization of Feistel Cipher

- Parameters
  - Block Size: typically 64 bits
  - Key Size: commonly 128 bits
  - Number of Rounds: 16
  - Subkey Generation algorithm
  - Round Function

DES (Data Encryption Standard)
DES (Data Encryption Standard)

- Published in 1977, standardized in 1979, expired in 1998.
- Similar structure to Feistel cipher
- Key: 64 bit quantity = 8-bit parity + 56-bit key
  - Every 8th bit is a parity bit.
- 64 bit input, 64 bit output.

- DES Encryption
  - 64 bit M → 64 bit C
  - 56 bits

DES Top View

- 64-bit Input
- Permutation
- Initial Permutation
- 48-bit K1
- Round 1
- 48-bit K2
- Round 2
- ......
- Round 16
- 48-bit K16
- Swap
- Swap 32-bit halves
- Permutation
- Final Permutation
- 64-bit Output
- 56-bit Key
- Generate keys
Bit Permutation (1-to-1)

Initial and Final Permutations

- Initial permutation (IP)
- View the input as M: 8-byte X 8-bit matrix
- Transform M into M1 in two steps
  - Transpose row x into column (9-x), 0<x<9
  - Apply permutation on the rows:
    - For even row y, it becomes row y/2
    - For odd row y, it becomes row (5+y/2)
- Final permutation FP = IP⁻¹
  - Why?
Per-Round Key Generation

**Initial Permutation of DES key**

- $C_{i-1}$ 28 bits
- $D_{i-1}$ 28 bits

- Circular Left Shift
- Circular Left Shift

- Permutation with Discard

- Round 1,2,9,16: single shift
- Others: two bits

$48$ bits $K_i$

- $C_i$ 28 bits
- $D_i$ 28 bits

A DES Round

- 32 bits
- 32 bits

- One Round Encryption
- Mangler Function
- S-Boxes

- 48 bits $K_i$

- 32 bits

- 32 bits
E Box of DES

- How is the E Box defined?

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>12</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>16</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>25</td>
</tr>
<tr>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>1</td>
</tr>
</tbody>
</table>
Mangler Function

The permutation produces “spread” among the chunks/S-boxes!

S-Box (Substitute and Shrink)

- 48 bits ==> 32 bits. (8*6 ==> 8 *4)
- 2 bits used to select amongst 4 permutations for the rest of the 4-bit quantity

an integer between 0 and 15.

i = 1,…8.
S1: (p. 71)

Each row and column contain different numbers.

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & \ldots & 15 \\
0 & 14 & 4 & 13 & 1 & 2 & 15 & 11 & \\
1 & 0 & 15 & 7 & 4 & 14 & 2 & 13 & \\
2 & 4 & 1 & 14 & 8 & 13 & 6 & 2 & \\
3 & 15 & 12 & 8 & 2 & 4 & 9 & 1 & \\
\end{array}
\]

Example: input: 100110 output: ???

DES Standard

- Cipher Iterative Action
  - Input: 64 bits
  - Key: 48 bits
  - Output: 64 bits

- Key Generation Box
  - Input: 56 bits
  - Output: 48 bits

One round (Total 16 rounds)
Avalanche Effect

- A small change in either the plaintext or the key should produce a significant change in the ciphertext.
- DES has a strong avalanche effect.
- Example
  - Plaintexts: 0X0000000000000000 and 0X8000000000000000
  - Same key: 0X016B24621C181C32
  - 34 bits difference in cipher-texts
  - Similar result with same plaintext and slightly different keys

Concerns About DES

- Key space problem: 56 bit key ($2^{56}$)
  - DESCHALL recovered RSA challenge I key on June 17, 1997 (6 month into the contest)
  - $.25m (total cost), July 15, 1998, RSA DES challenge II key recovered in 56 hours
- Cryptanalysis
  - Sixteen Weak and semi-weak keys:
  - Differential cryptanalysis require less tries using chosen plaintext/ciphertext [Biham, 1993]
    - Effective up to 15 rounds
    - DES is well designed to defeat differential analysis
  - Linear cryptanalysis requires only known plaintext/ciphertext [Matsui, 1993]
DES Summary

- Simple, easy to implement:
  - Hardware/gigabits/second, software/megabits/second
- 56-bit key DES maybe acceptable for non-critical applications but triple DES (DES3) should be secure for most applications today
- Supports several operation modes: ECB CBC, OFB, CFB

Modes of Block Cipher Operations
Encrypting a Large Message

- Modes of block cipher operations
  - ECB (Electronic Code Book)
  - CBC (Cipher Block Chaining Mode)
  - OFB (Output Feedback Mode)
  - CFB (Cipher Feedback Mode)

Electronic Code Book (ECB)

Divide and conquer!
ECB Properties

ECB Properties (Cont’d)

- Cipher block substitution and rearrangement attacks
  - fabrication of specific information
- No error propagation.
Cipher Block Chaining (CBC)

\[ M_1 = 64 \]
\[ M_2 = 64 \]
\[ M_3 = 64 \]
\[ M_4 = 46 \text{ pad} \]

IV
Initialization Vector

\[ C_1 \]
\[ C_2 \]
\[ C_3 \]
\[ C_4 \]

\( (M_1 = M_3) \text{ very unlikely leads to } (C_1 = C_3) \)

CBC Decryption

\[ M_1 \]
\[ M_2 \]
\[ M_3 \]
\[ M_4 \]

IV

\[ C_1 \]
\[ C_2 \]
\[ C_3 \]
\[ C_4 \]
CBC Properties

- **Chaining dependency**
  - Each ciphertext block depends on all preceding plaintext blocks

- **Error propagation**
  - Each error in \(c_j\) affects decipherment of \(c_j\) and \(c_{j+1}\).
  - Predictable bit change in \(m_{j+1}\) by alert corresponding bits of \(c_j\).

- **Error recovery**
  - An error in \(c_j\) doesn’t propagate beyond \(c_{j+1}\).
  - Can recover from loss of cipher text blocks.

Output Feedback Mode (OFB)

Like a Random Number Generator...

- IV
- \(k\)
- \(M_1\), \(M_2\), \(M_3\), \(M_4\)
- \(C_1\), \(C_2\), \(C_3\), \(C_4\)
- K bits
- ENC
OFB Properties

- Chaining dependencies
  - Key stream is plaintext-independent
  - Allow pre-computing of pseudo-random stream (One-Time Pad); XOR can be implemented very efficiently
- No error propagation problem as in CBC
- Error recovery
  - Can recover from bit error
  - But not from block loss.
- If the attacker knows the plaintext, he can change the ciphertext by XORing it with the plaintext and then XORing with whatever he wants to transmit.

General $k$-bit CFB
CFB Properties

- Chaining dependencies
  - Ciphertext block $c_j$ depends on all preceding plaintext blocks.

- Error propagation
  - Bit error in one ciphertext block affects the next several blocks

- Error recovery
  - Can recover from bit errors after several blocks
  - Can resynchronize after loss of blocks.

- Secure against known plaintext attack (plaintext substitution)

- Less vulnerable to tampering with ciphertext - cipher $C_j$’s impact on $m_{i+1}$ is subtle (through encryption function) and thus less predictable

Multiple Encryption
Triple DES

- Major limitation of DES
  - Key length is too short (56 bits).
- Question: Can we apply DES multiple times to increase the strength of encryption?
  - Advantage: preserve the existing investment in software and equipment.

Triple DES (Cont’d)

- Double DES
  - Encrypt the plaintext twice with two different DES keys
  - Key length increases to 112 bits
- Two concerns
  - Is DES a group?
    - $E_{k_2}(E_{k_1}(P)) = E_{k_3}(P)$
    - Implication?
  - Meet-in-the-middle attack
Meet-in-the-middle attack

Encryption

\[ \begin{align*}
    P & \xrightarrow{E} X & \xrightarrow{E} C \\
    K_1 & \downarrow & K_2
\end{align*} \]

Decryption

\[ \begin{align*}
    P & \xleftarrow{D} X & \xleftarrow{D} C \\
    K_1 & \uparrow & K_2
\end{align*} \]

Observation:
\[ X = E_{K_1}(P) = D_{K_2}(C) \]

- For a known pair (P,C)
  - Encrypt P for all 2^{56} values for K_1
  - Store the results in a table sorted by the value of X
  - Decrypt C for all 2^{56} values for K_2, and for each result check the table
  - A match reveals a possible combination of key

Meet-in-the-middle attack (Cont’d)

- Analysis
  - With one pair (P,C), #keys that can survive the test is 2^{112}/2^{64} = 2^{48}.
  - For each pair of keys (K_1, K_2), the probability that it can find a non-empty entry in the table is 2^{-8}.
  - With another pair (P’,C’), #keys that can survive both tests is 2^{-8} \times 2^{-8} = 2^{-16}.
  - The probability that the correct keys are determined is 1-2^{-16}.

- Goal of double DES
  - Increase the difficulty of exhaustive key search (2^{112} keys)
  - In effect, the effort is on the order of 2^{56}. 
Triple DES (Cont’d)

Encryption:

Decryption:

• Apply DES encryption/decryption three times.
  – With two keys or three keys
• Why E-D-E?
  – It’s not clear if DES is a group when this was proposed.
  – If one key is used, it’s equivalent to doing DES once.

Triple DES Is Not Ideal...

• Efficiency demands schemes with longer keys to begin with!
• Triple DES runs one third as fast as DES on the same platform
• New candidates are numerous - RC5, IDEA, two-fish, CAST, etc
• New AES
Message Authentication

- Message authentication is the process to verify that received messages come from the alleged source and have not been altered.
- The goals of message authentication is to prevent
  - Masquerade: insertion of messages from a fraudulent source.
  - Content modification: change of messages
  - Sequence modification: insertion, deletion and reordering of messages.
  - Timing modification: delay or replay of messages.
Message Authentication Functions

- Message encryption
- Message Authentication Code (MAC)
- Hash function

Encryption for Message Authentication

- Conventional cryptography
  - Use the structure or pattern in the plaintext
    - Accept the decrypted plaintext if it is in an intelligible form.
    - No guarantee!
  - Append an error-detecting code (Frame Check Sequence, or FCS) to the plaintext before encryption.
    - Encryption: $C = E_K(P || F(P))$
    - Decryption: $P' || F(P) = D_K(C)$, and then check if $F(P') = F(P)$
    - The order of FCS and encryption is critical
Message Authentication Code (MAC)

• MAC
  – Also known as cryptographic checksum, Message Integrity Code (MIC).
  – Assumption: the sender and the receiver share a common secret key.
  – A small fixed-size block generated from the message with secret key cryptography.
  – Usually appended to the original message.

MAC (Cont’d)

• Mode I
  – Message authentication
  – No confidentiality
MAC (Cont’d)

- **Mode II**
  - Message authentication and confidentiality
  - Authentication tied to plaintext

MAC (Cont’d)

- **Mode III**
  - Message authentication and confidentiality
  - Authentication tied to ciphertext
Requirements for MAC

- For M and $C_K(M)$, it is computationally infeasible to construct a message $M'$ such that $C_K(M') = C_K(M)$.
- $C_K(M)$ should be uniformly distributed in terms of M
  - For any two messages M and M', $\Pr[C_K(M) = C_K(M')] = 2^{-n}$, where $n$ is the number of bits in the MAC.
  - Intuition: prevent chosen plaintext attack.
- If $M'$ is equal to some known transformation on M, then $\Pr[C_K(M) = C_K(M')] = 2^{-n}$.
  - This requirement is subsumed by the above one.
  - Intuition: no weak spot with respect to certain bits of the message.

MAC Based on DES CBC Mode

- Known as Data Authentication Algorithm
- DES CBC mode with IV being zero.
- A message is padded with zeroes to form 64-bit blocks.
- The data authentication code (DAC, i.e., the MAC) consists of either the entire last ciphertext block or the left M bits with $16 \leq M \leq 64$. 

MAC Based on DES CBC Mode (Cont’d)

\[
\begin{align*}
    M_1 & \quad 64 \\
    & \quad \downarrow \\
    & \quad \text{ENC} \\
    & \quad \downarrow \\
    C_1 & \\
\end{align*}
\quad \quad
\begin{align*}
    M_2 & \quad 64 \\
    & \quad \downarrow \\
    & \quad \text{ENC} \\
    & \quad \downarrow \\
    C_2 & \\
\end{align*}
\quad \quad
\begin{align*}
    M_3 & \quad 64 \\
    & \quad \downarrow \\
    & \quad \text{ENC} \\
    & \quad \downarrow \\
    C_3 & \\
\end{align*}
\quad \quad
\begin{align*}
    M_4 & \quad 46\text{pad} \\
    & \quad \downarrow \\
    & \quad \text{ENC} \\
    & \quad \downarrow \\
    C_4 & \\
\end{align*}

DAC (16 to 64 bits)