**实验2： 加密算法**

**一、实验目的**

理解对称加密算法的原理和特点，通过用DES算法对实际的数据进行加密和解密来理解DES算法的加密原理及运行原理。

**二、实验内容**

理解对称加密算法的原理和特点；

通过用DES算法对实际的数据进行加密和解密来理解DES算法的加密原理及运行原理。

**三、实验原理、方法和手段**

**3.1 对称密钥体系**

对称密钥机制即对称密钥体系，也称为单钥密码体系和传统密码体系。对称密码体系通常分为两大类，一类是分组密码（如DES、AES算法），另一类是序列密码（如RC4算法）。

对称密码体系加密和解密时所用的密钥是相同的或者是类似的，即由加密密钥可以很容易地推导出解密密钥，反之亦然。同时在一个密码系统中，我们不能假定加密算法和解密算法是保密的，因此密钥必须保密。发送信息的通道往往是不可靠的或者不安全的，所以在对称密码系统中，必须用不同于发送信息的另外一个安全信道来发送密钥。

对称密码体系的优点：

加密效率高，硬件实现可达每秒数百兆字节（件实现略慢一些）。

密钥相对较短。

可以用来构造各种密钥机制。

可以用来建造安全性更强的密码。

对称密码体系的缺点：

通信双方都要保持密钥的秘密性。

在大型网络中，每个人需持有许多密钥。

为了安全，需要经常更换密钥。

**3.2 对称加密原理**

信息加密根据采用的密钥类型可以划分为对称密码算法和非对称密码算法。对称密码算法是指加密系统的加密密钥和解密密钥相同，或者虽然不同，但是可以从其中任意一个推导出另一个，更形象的说就是用同一把钥匙开锁和解锁。在对称密码算法的发展历史中曾出现过多种优秀的算法，包括DES、3DES、AES等。下面我们以DES算法为例介绍对称密码算法的实现机制。

DES算法是由美国IBM公司在20世纪70年代提出，并被美国政府、美国国家标准局和美国国家标准协会采纳和承认的一种标准加密算法。它属于分组加密算法，即在明文加密和密文解密过程中，信息都是按照固定长度分组后进行处理的。混淆和扩散是它采用的两个最重要的安全特性。混淆是指通过密码算法使明文和密文以及密钥的关系非常复杂，无法从数学上描述或者统计。扩散是指明文和密钥中每一位信息的变动，都会影响到密文中许多位信息的变动，从而隐藏统计上的特性，增加密码的安全。

DES算法将明文分成64位大小的众多数据块，即分组长度为64位。同时用56位密钥对64位明文信息加密，最终形成64位的密文。如果明文长度下足64位，则将其扩展为64位（如补零等方法）。具体加密过程首先是将输入的数据进行初始换位（IP），即将明文M中数据的排例顺序按一定的规则重新排列，生成新的数据序列，以打乱原来的次序。然后将变换后的数据平分成左右两部分，左边记为L0，右边记为R0，然后对R0实行在子密钥(由加密密钥产生）控制下的变换f，结果记为f（Ro，K1)，再与L0做逐位异或运算，其结果记为R1，R0则作为下一轮的L1。如此循环16轮，最后得到L16、R16，再对L16、，R16实行逆初始置换IP-1，即可得到加密数据。解密过程与此类似，不同之处仅在于子密钥的使用顺序正好相反。

DES的加密算法包括3个基本模块。

1. 初始换位(IP)

它的作用是把输入的64位数据块的排列顺序打乱，每位数据按照下面换位规则重新组合，即将第58位换到第1位，第50位换到第2位，……，依次类推。重组后的64位输出分为L0、R0（左、右）两部分，每部分分别为32位。

58，50，42，34，26，18，10，2，60，52．44，36，28，20，12，4

62，54，46，38，30，22，14，6，64，56，48，40，32，24，16，8

57，49，41，33，25，17，9，1，59，51，43，35，27，19，11，3

61，53，45，37，29，21，13，5，63，55，47，39，31，23，15，7

R0和K1经过f(Ro，K1)变换后的输出结果，再和L0进行异或运算，输出结果做为R1，R0则赋给L1。L1和R1同样再做类似运算生成L2和R2，……，经过16次运算后生成L16和R16。

2．f函数

f函数是多个置换函数和替代函数的组合函数，它将32位比特的输入变换为32位的输出。Ri经过扩展运算E变换后扩展为48位的E(Ri)，与Ki+1进行异或运算后输出的结果分成8组，每组6比特的并联B，B=B1B2B3B4B5B6B7B8，再经过8个S盒的选择压缩运算转换为4位，8个4位合并为32位后再经过P变换输出为32位的f(Ri，Ki+1)。其中，扩展运算E与置换P主要作用是增加算法的扩散效果。

3．逆初始量换函数IP-1

它将L16和R16作为输入，进行逆初始换位得到密文输出。逆初始换位是初始换位的逆运算，换位规则如下所列：

40， 8，48，16，56，24，64，32，39，7，47，15，55，25，63，31

38，6，46，14，54，22，62，30，37，5，45，13．53．21，61，29

36，4，44，12，52，20，60，28，35，3，43，11，51，19，59，27

34，2，42，10，50，18，58，26，33，1，41，9，49，1?，57，25

DES的加密算法中除了上面介绍的3个基本函数，还有一个非常重要的功能模块，子密钥的生成模块。

输入的初始密钥值为64位，但DES算法规定，其中第8、16、……、64位是奇偶校验位，不参与DES运算。所以，实际可用位数只有56位，经过缩小选择换位表1(表l—2)即密钥置换PC-1的变换后，初始密钥的位数由64位变成了56位，将其平分为两部分C0、D0，然后分别进行第1次循环左移，得到C1、 D1，将C1（28位）、D1(28位）合并后得到56位的输出结果，再经过缩小选择换位表2(表1—3)即密钥置换PC-2，从而得到了密钥K1(48位)。依次类推，便可得到K2、……、K16需要注意的是，16次循环左移对应的左移位数要依据表1的规则进行。

表1 左移位数规则

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| LSi | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |

表2缩小选择换位表1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 |
| 1 | 58 | 50 | 42 | 34 | 26 | 18 |
| 10 | 2 | 59 | 51 | 43 | 35 | 27 |
| 19 | 11 | 3 | 60 | 52 | 44 | 36 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 |
| 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 |
| 21 | 13 | 5 | 28 | 20 | 12 | 4 |

表3缩小选择换位表2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 14 | 17 | 11 | 24 | 1 | 5 |
| 3 | 28 | 15 | 6 | 21 | 10 |
| 23 | 19 | 12 | 4 | 26 | 8 |
| 16 | 7 | 27 | 20 | 13 | 2 |
| 41 | 52 | 31 | 37 | 47 | 55 |
| 30 | 40 | 51 | 45 | 33 | 48 |
| 44 | 49 | 39 | 56 | 34 | 53 |
| 46 | 42 | 50 | 36 | 29 | 32 |

**四、实验内容**

**4.1 画原理图**

画出DES加密的算法原理图。

**4.2 加密代码**

编写DES加密代码。

**4.3 解密代码**

编写DES加密代码。

**五、实验组织运行要求**

单人实验，单独完成。

**六、实验条件**

（1）计算机（win、linux、osx不限）

（2）MATLAB或其他任何编程工具/语言，工具不是我们实验目的。

**七、实验结果**

**7.1 画原理图**

画出DES加密的算法原理图。

|  |
| --- |
|  |

**7.2 加密**

**实验代码**

|  |
| --- |
| **generate\_keys.m函数**  function [ keys ] = generate\_keys( keyword )  % 根据密钥生成16轮子密钥  if length(keyword)~=8  disp(['密码不是八位的'])  return  end  binkwd='';  bin\_keyword = dec2base(keyword,2,8); % 转8位二进制  for i=1:8  binkwd = [binkwd,bin\_keyword(i,:)];  end  bkwd = PC\_1(binkwd); % pc-1打乱次序  mov\_bit=[1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1]; % 每一轮循环移位的位数  C0 = bkwd(1:28);  D0 = bkwd(29:56);  for i=1:16  C0=[C0(1+mov\_bit(i):end),C0(1:mov\_bit(i))];  D0=[D0(1+mov\_bit(i):end),D0(1:mov\_bit(i))]; % 循环移位  bfkey = [C0,D0];  keys(i,:)=PC\_2(bfkey);  end  end  function [Bkwd]=PC\_1(bkwd)  % PC\_1置换  PC1\_tab = [57,49,41,33,25,17, 9,1,58,50,42,34,26,18,10, 2,59,51,43,35,27,19,11, 3,60,52,44,36,63,55,47,39,31,23,15,7,62,54,46,38,30,22,14, 6,64,53,45,37,29,21,13, 5,28,20,12,4];  for i=1:56  Bkwd(i)=bkwd(PC1\_tab(i));  end  end  function [key]=PC\_2(bfkey)  % PC\_2置换  PC2\_tab = [14,17,11,24,1,5,3,28,15,6,21,10,23,19,12,4,26,8,16,7,27,20,13,2,41,52,31,37,47,55,30,40,51,45,33,48,44,49,39,56,34,53,46,42,50,36,29,32];  for i=1:48  key(i)=bfkey(PC2\_tab(i));  end  end  **convert\_mess2bin.m函数**  function [ text ] = convert\_mess2bin( mess )  %  if length(mess)~=8  disp(['input is not 8 byte'])  return  end  text='';  mess = dec2base(mess,2,8); %  for i=1:8  text = [text,mess(i,:)];  end  end  **DES.m函数**  function [ ciphertext ] = DES( ciphertext,keys )  % 对称密码算法 分组加密  % ciphertext 待加密明文  % key 密钥  % ciphertext 加密以后的密文  % 按照字符串处理  ciphertext\_ip = transform( ciphertext ); % 进行IP置换  text = ciphertext\_ip;  Ltext = text(:,1:1:32); % 左半部分  Rtext = text(:,33:1:64); % 右半部分  for i=1:16 % 16轮结构变换  NextL = Rtext; % 下一轮的L是上一轮的R  now\_key = keys(i,:); % 这一轮加密的密钥  Rtext\_E = Entend\_box(Rtext); % 32位数据放入E盒扩展成48位  for j=1:48  Rtext\_E(j)=dec2bin(xor(bin2dec(Rtext\_E(j)),bin2dec(now\_key(j)))); % 与密钥异或处理  end  Rtext = Substitute\_box(Rtext\_E); % S盒压缩  Rtext = P\_box(Rtext) ; % 对S盒输出进行IP置换  for j=1:32  Rtext(j)=dec2bin(xor(bin2dec(Rtext(j)),bin2dec(Ltext(j)))); % 与上一轮L异或处理  end  Ltext = NextL;  end  % text = [Ltext,Rtext]; 这样是错的  text = [Rtext,Ltext];  ciphertext = anti\_transform(text);  end  function [ ciphertext\_ip ] = transform( ciphertext )  % 对明文进行IP置换  % ciphertext 明文  % ciphertext\_ip IP置换后的明文  ciphertext\_ip = blanks(64); % 预分配内存64bit  IP\_table = [58, 50, 42, 34, 26, 18, 10, 2,60, 52, 44, 36, 28, 20, 12, 4,62, 54, 46, 38, 30, 22, 14, 6,64, 56, 48, 40, 32, 24, 16, 8,57, 49, 41, 33, 25, 17, 9, 1,59, 51, 43, 35, 27, 19, 11, 3,61, 53, 45, 37, 29, 21, 13, 5,63, 55, 47, 39, 31, 23, 15, 7]; % IP置换表  for i=1:64  ciphertext\_ip(i)=ciphertext(IP\_table(i));  end  end  function [Rtext\_E] = Entend\_box(Rtext)  % 对右半部分扩展成48位便于异或操作  % Rtext 待扩展数据  % Rtext\_E E扩展后的数据  Rtext\_E = blanks(48); % 预分配内存48bit  ex\_tab = [32, 1, 2, 3, 4, 5,4, 5, 6, 7, 8, 9,8, 9,10,11,12,13,12,13,14,15,16,17,16,17,18,19,20,21,20,21,22,23,24,25,24,25,26,27,28,29,28,29,30,31,32, 1]; % E扩展表  for i=1:48  Rtext\_E(i)=Rtext(ex\_tab(i)); % 32位扩展成48位  end  end  function [Rtext] = Substitute\_box(Rtext\_E)  % S盒压缩成32位  stab(:,:,1)=[14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7;  0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8;  4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0;  15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13];  stab(:,:,2)=[15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10;  3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5;  0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15;  13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9];  stab(:,:,3)=[10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8;  13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1;  13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7;  1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12];  stab(:,:,4)=[7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15;  13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9;  10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4;  3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14];  stab(:,:,5)=[2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9;  14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6;  4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14;  11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3];  stab(:,:,6)=[12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11;  10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8;  9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6;  4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13];  stab(:,:,7)=[4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1;  13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6;  1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2;  6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12];  stab(:,:,8)=[13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7;  1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2;  7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8;  2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11];  % 8个S盒的表  Rtext='';  for i=1:8  s = stab(:,:,i);  str = Rtext\_E(:,i\*6-5:i\*6);  m = bin2dec([str(1),str(6)]); % S盒的行列计数都是从0开始  n = bin2dec(str(:,2:5));  temp = dec2base(s(m+1,n+1),2,4); % s盒第m行n列,4位二进制  Rtext = [Rtext,temp];  end  end  function [ stext\_ip ] = P\_box( stext )  % 对S盒的输出进行IP置换  % ciphertext 明文  % ciphertext\_ip IP置换后的明文  stext\_ip = blanks(32); % 预分配内存32bit  IP\_table = [16, 7,20,21,29,12,28,17,1,15,23,26, 5,18,31,10,2, 8,24,14,32,27, 3, 9,19,13,30, 6,22,11, 4,25]; % IP置换表  for i=1:32  stext\_ip(i)=stext(IP\_table(i));  end  end  function [ text\_ip ] = anti\_transform( text )  % 对最终结果进行IP逆置换  text\_ip = blanks(64); % 预分配内存64bit  IP\_table = [40,8,48,16,56,24,64,32,39,7,47,15,55,23,63,31,38,6,46,14,54,22,62,30,37,5,45,13,53,21,61,29,36,4,44,12,52,20,60,28,35,3,43,11,51,19,59,27,34,2,42,10,50,18,58 26,33,1,41, 9,49,17,57,25,];% IP置换表  for i=1:64  text\_ip(i)=text(IP\_table(i));  end  end  **主函数**  clc;clear all;close all;  prompt= '输入密钥：\n';  keyword = input(prompt,'s');  keys = generate\_keys(keyword);  prompt='输入明文：\n';  text =input(prompt,'s');%要加密的字符串  len=length(text);  result='';  a=mod(len,8);  for i=1:(8-a)  text=[text ' '];  end  r=length(text)/8;  for i=1:r%循环r次  a=text(1:8);  bin\_text = convert\_mess2bin(a);  bin\_result = des(bin\_text,keys);  text=text(9:length(text));  result=[result bin\_result];  end  disp(['密文：',result]) |

**实验结果**

|  |
| --- |
|  |

**7.3 解密**

**实验代码**

|  |
| --- |
| **convert\_bin2mess.m函数**  function [ mess ] = convert\_bin2mess( binn )  mess = '';  for i=1:8  a =(i-1)\*8+1;  str = binn(a:a+7);  mess = [mess,char(bin2dec(str))];  end  end  **主函数**  clc;clear all;close all;  prompt= '输入密钥\n';  keyword = input(prompt,'s');  keys = generate\_keys(keyword);  de\_keys = flipud(keys);  prompt='输入密文：\n';%要解密的字符串  text =input(prompt,'s');  result='';  r=length(text)/64;  for i=1:r  a=text(1:64);  bin\_de\_result = des(a,de\_keys);  de\_result = convert\_bin2mess(bin\_de\_result);  text=text(65:length(text));  result=[result de\_result];  end  disp(['明文：'result]) |

**实验结果**

|  |
| --- |
|  |