**《现代密码学》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **20级信息安全专业01班** | | | **姓名** | **蔡欣彤** |
| **实验题目** | **对称密码算法DES** | | | | | |
| **实验时间** | **2023.5.14** | | **实验地点** | **DS3304** | | |
| **实验成绩** |  | | **实验性质** | **☑验证性 □设计性 □综合性** | | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 程序结构/实验步骤合理；  □实验结果正确； 语法、语义正确； 报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的  通过用DES算法对实际的数据进行加密和解密来深刻了解DES的运行原理。 | | | | | | |
| 二、实验项目内容  1、自己编写DES**子密钥的生成算法，并根据自己设定的种子密钥运行程序产生并显示出16轮的子密钥，同时完成DES算法弱秘钥和半弱秘钥对的验证；**  **弱秘钥**  **01 01 01 01 01 01 01 01**  **1F 1F 1F 1F 0F 0F 0E 0E**  **E0 E0 E0 E0 F1 F1 F1 F1**  **FE FE FE FE FE FE FE FE**  **k1和k2构成半弱密钥对**  **01 FE 01 FE 01 FE 01 FE**  **FE 01 FE 01 FE 01 FE 01**  **01 E0 01 E0 01 F1 01 F1**  **E0 01 E0 01 F1 01 F1 01**  2、应用DES算法程序实现对一个文件进行加密和解密，提交程序代码和执行结果。  3、用DES算法对输入的十六进制数加密，比较输入和输出，当改变一个位时比较输出的变化，并说明原因。 | | | | | | |
| 1. 实验过程或算法（源程序）   1.编程实现子密钥生成。  #include <iostream>  #include<string>  using namespace std;  int PC1[56] = {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};  int PC2[48] = { 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 };  int lshift[16] = {1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1};  //十六进制转二进制  string trans16to2(string hexDigit) {  string binaryDigit;  for (int i = 0; i < hexDigit.length(); ++i) {  char e = hexDigit[i];  if (e >= 'A' && e <= 'F') {  int a = static\_cast<int>(e - 'A' + 10);  switch (a) {  case 10:  binaryDigit += "1010";  break;  case 11:  binaryDigit += "1011";  break;  case 12:  binaryDigit += "1100";  break;  case 13:  binaryDigit += "1101";  break;  case 14:  binaryDigit += "1110";  break;  case 15:  binaryDigit += "1111";  break;  }  }  else if (isdigit(e)) {  int b = static\_cast<int>(e - '0');  switch (b) {  case 0:  binaryDigit += "0000";  break;  case 1:  binaryDigit += "0001";  break;  case 2:  binaryDigit += "0010";  break;  case 3:  binaryDigit += "0011";  break;  case 4:  binaryDigit += "0100";  break;  case 5:  binaryDigit += "0101";  break;  case 6:  binaryDigit += "0110";  break;  case 7:  binaryDigit += "0111";  break;  case 8:  binaryDigit += "1000";  break;  case 9:  binaryDigit += "1001";  break;  }  }  }  return binaryDigit;  };  //循环左移一位  string strol(string ss) {  int len = ss.length();  string ss1;  char temp = ss[0];  for (int i = 1; i < len; i++) {  ss1 += ss[i];  }  ss1 += temp;  return ss1;  };  int main() {  string hexDigit;  cin >> hexDigit;  string binaryDigit = trans16to2(hexDigit);  //cout << binaryDigit;  //cout << binaryDigit.length();  string sPC1;  string K[16];  for (int i = 0; i < 56; i++) {  sPC1 += binaryDigit[PC1[i]-1];  //cout << sPC1 << endl;  }  //cout << sPC1;  string temp = sPC1;  string c = temp.substr(0, 28);  string d = temp.substr(28, 55);  for (int i = 0; i < 16; i++) {  int numlshift = lshift[i];  //cout << c<<endl;  if (numlshift == 1) {  c = strol(c);  d = strol(d);  }  if (numlshift == 2) {  c = strol(c);  d = strol(d);  c = strol(c);  d = strol(d);  }    string ss;  ss += c;  ss += d;  //cout << ss.length() << endl;  //cout << ss<<endl;  for (int j = 0; j < 48; j++) {  K[i] += ss[PC2[j]-1];  }  }  for (int k = 0; k < 16; k++) {  //cout << K[k].length() << endl;  cout << K[k] << endl;  }  } | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程  1.密钥为0102030405060708    弱密钥验证：  密钥：01 01 01 01 01 01 01 01    1F 1F 1F 1F 0F 0F 0E 0E    E0 E0 E0 E0 F1 F1 F1 F1    FE FE FE FE FE FE FE FE    由上述结果可以看出，16轮每一轮生成的子密钥均相同。  弱密钥对验证：  01 FE 01 FE 01 FE 01 FE    FE 01 FE 01 FE 01 FE 01    01 E0 01 E0 01 F1 01 F1    E0 01 E0 01 F1 01 F1 01    由上述结果可以看出，弱密钥对16轮中只产出2个不同的子密钥。  2.   1. 用DES算法对输入的十六进制数加密，比较输入和输出，当改变一个位时比较输出的变化，并说明原因。     10101000011110100010101010100110110010101110110011011100000110000011000111  11101010011101110101001111110010100001100000001110001001111111010100010011101110101100100000110001000001101000001110010111100010000110001011000011010010110101110110001111101001111110  修改第二位为0    0001,0011,1111,1010,1101,1101,1111,0111,0001,0010,0000,1011,0001,0011,1000,0001  001100011111101010011101110101001111110010100001100000001110001001111111010100010011101110101100100000110001000001101000001110010111100010000110001011000011010010110101110110001111101001111110  改变后，除了前64位外，其他均为变化。  原因：DES是分组密码，以64位为一组进行加解密操作。 | | | | | | |