

# 开发手册&用户指南

—— written by Superstars

## 1.简介

### 1.1 系统概述

该系统是我们“Superstars”团队为用户设计的一款简单、开源的 DES 算法加解密系统,能够 满足用户简单的加解密需要(二进制加解密和 ASCII 码字符串加解密),同时可以帮助用户在已知明密文的条件下暴力破解出密钥,以及向用户展示 DES 算法的雪崩效应。

### 1.2 主要功能和特点

该系统主要功能包括对二进制数和 ASCII 码进行 DES 算法的加解密、根据明密文对进行暴力破解计算、验证 DES 算法的雪崩效应等。 该系统的特点在于简单易懂、容易上手,可以帮助不太了解 DES 算法的用户快速获取加解密结果以及相应的密钥信息,也可以帮助正在学习 DES 算法学生更直观的接触到简单的 DES 应用,验证 DES 的雪崩效应特性,感受 DES 加密算法的伟大和奇妙之处。

### 1.3 目标用户群体

初步学习和希望了解 DES 算法的用户。

### 1.4 使用说明

该系统可以在 QT 上运行,具体页面及功能详见下文

## 2. 开发者团队

**2.1 开发者队名:** Superstars

**2.2 开发者姓名:** 张家玮、季俊杰

## **3. 开发环境**

### **3.1 开发工具和技术栈**

#### **3.1.1 开发工具**

集成开发环境（IDE）：VS Code、QtCreator4.11.1

版本控制：Github

#### **3.1.2 技术栈**

前端技术：语言：C++ 框架和库：Qt

后端技术：语言：C++

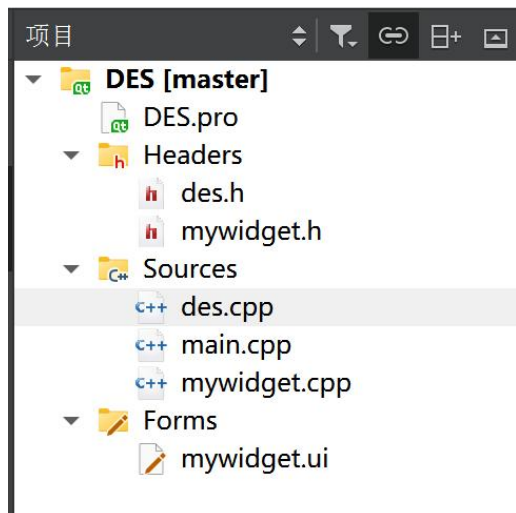
### **3.2 运行**

在 Qt 中运行 "DES.pro"

## **4.代码结构**

### **4.1 项目的目录结构**

项目的目录结构图如下：

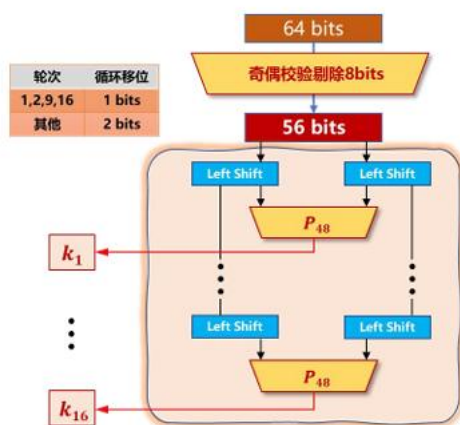


- `des.cpp` 实现一个基于 DES（数据加密标准）的加密和解密算法，包括生成密钥、处理明文和密文的各种变换操作。
- `main.cpp` 负责创建应用程序的主窗口，设置界面布局。
- `mywidget.cpp` 处理用户输入和与加密算法的交互。

## 4.2 关键组件和模块的描述

### 4.2.1 密钥生成函数

1、原理：



2、代码演示：

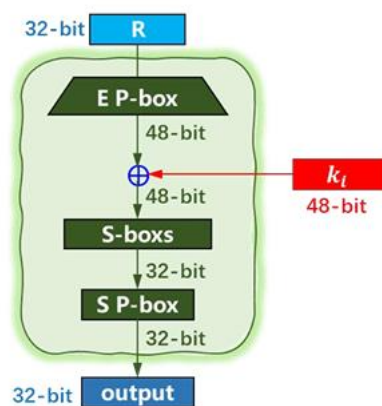
该代码首先执行置换操作。再将密钥分成两部分 `left_half` 和 `right_half`，分别执行左移一位的操作，然后执行第一次压缩置换的操作，生成第一个密钥 `K1`。接着

重复拆 分左移和压缩置换的操作，得到第二个密钥 K2。 所用函数如下所示：

- `getKey(string input)`: 通过输入字符串生成 10 位的密钥。
- `P10Box(vector<int> input)`: 对输入进行 P10 置换，用于密钥生成。
- `P8Box(vector<int> input)`: 对输入进行 P8 置换，生成最终的子密钥。
- `subKey(vector<int> input)`: 生成两个子密钥，分别用于加密的不同轮次。
- `leftShift1(vector<int> input)` 和 `leftShift2(vector<int> input)`: 对密钥的两部分进行左移操作，以生成子密钥。

### 4.2.2 轮函数

1、原理：



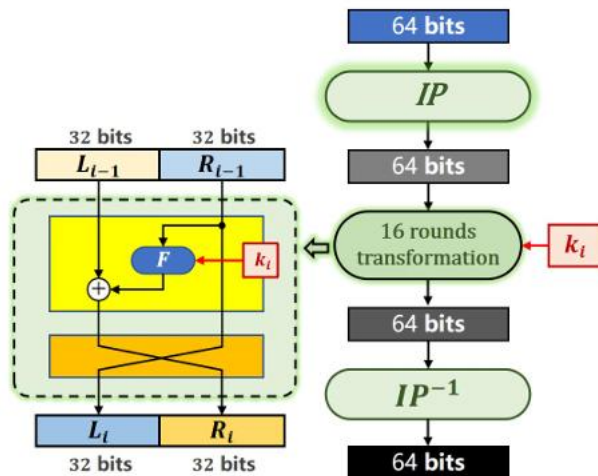
2、代码演示：

该函数接受加解密过程中拆分出的右半部分以及轮密钥输入，首先将右半部分（right-half）进行 E P-Box 的扩展置换，扩展为 10-bits，再与轮密钥（10-bits）进行亦 或操作。进而对所得结果通过 S-boxs 进行压缩替换，得到 8-bits 结果。最后，进行 SP-box 的直接置换。 所用函数如下所示：

- `F(vector<int> input, int turn)`: 核心的轮函数，接受输入和当前轮次的参数。它通过扩展置换、异或操作、S-Box 替换和 P-Box 置换来处理数据。
- `splitedP(vector<int> input, int turn, bool flag)`: 将输入分为左右部分，并根据当前轮次进行处理，调用轮函数 F 来更新左半部分。

### 4.2.3 加密算法

#### 1、原理：



#### 2、代码演示：

该加密函数接受输入框中明文和密钥的输入，首先调用密钥生成函数生成密钥  $K_1$ ， $K_2$ 。然后，利用 IP 进行初始置换操作。再进入多轮变换部分，此时需要把现有 8-bits 经过转换的明文分为左右两部分（left-half 和 right-half）。执行第一轮加密，先用右半部分和轮密钥  $key_1$  执行轮函数操作。再将结果与左半部分进行亦或操作。然后按照同样的过程执行第二轮加密。最终将经过一次交换（第二轮不会发生左右部分的交换）的内容合并，利用  $IP^{-1}$  进行最终置换操作。所用函数如下所示：

- `char2Binary(char input)`: 将字符转换为二进制数组并补齐。
- `int2Binary(int input)`: 将整数转换为二进制数组并补齐。
- `forceInt2Binary(int input)`: 将整数转换为 10 位的二进制数组，主要用于暴力破解。
- `binary2Cha(vector<int> input)`: 将二进制数组转换为字符。
- `strToBinary(string plainText)`: 将字符串转换为二进制数组。
- `IP(vector<int> input)`: 初始置换。
- `IPReverse(vector<int> input)`: 逆置换。
- `EPBox(vector<int> input)`: 扩展置换。

- SPBox(vector<int> input): S 盒置换。

注：解密算法的解密过程与加密过程基本一致，只有轮密钥的顺序有变化。

#### 4.2.4 暴力加密和解密函数

1、原理：已知明密文对，寻找可能的密钥空间

2、代码演示：

```
string forceEncryptionAPI(string s, vector<int> key) { //暴力加密的接口
    string res = "";
    globalKey = key;
    vector<vector<int>> plainText = strToBinary(s);
    int len = plainText.size();
    //cout << len << endl;
    for (int i = 0; i < len; i++) {
        vector<int> temp = IP(plainText[i]);
        temp = splitedP(temp, 0, true);
        temp = splitedP(temp, 1, true);
        temp = IPReverse(temp);
        for (int i = 0; i < 8; i++) {
            res.push_back(temp[i] + '0');
        }
        //char word = binary2Cha(temp);
    }
    return res;
}
```

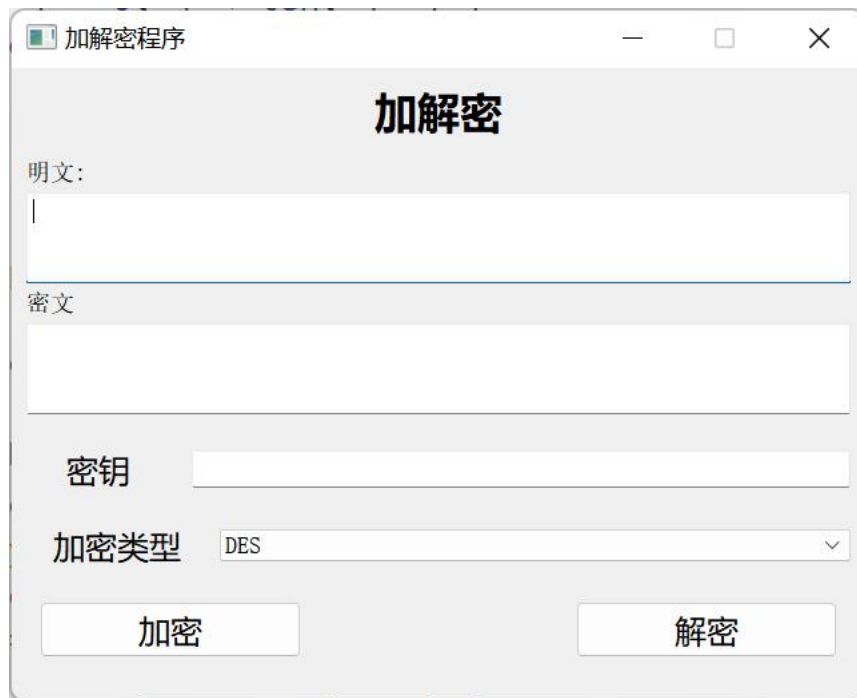
暴力破解函数通过生成所有可能的 10 位密钥，并将其与给定密文进行比较，从而找到正确的密钥，确保用户可以在已知明密文的情况下恢复明文。

```
string forceDecryptionAPI(string s, vector<int> key) { //暴力解密接口
    string res = "";
    globalKey = key;
    vector<vector<int>> plainText = strSplit(s);
    int len = plainText.size();
    for (int i = 0; i < len; i++) {
        vector<int> temp = IP(plainText[i]);
        temp = splitedP(temp, 1, false);
        temp = splitedP(temp, 0, false);
        temp = IPReverse(temp);
        char word = binary2Cha(temp);
        res.push_back(word);
    }
    return res;
}
```

## 5. 基础功能及界面

### 5.1 加解密页面

该页面提供了二进制和 ASCII 码加解密功能。



加解密程序

**加解密**

明文:  
|

密文  
|

密钥 |

加密类型 DES

加密 解密

- 1、用户可以在明文文本框和密钥文本框分别输入明文和密钥，点击“加密”按钮，执行加密操作。
- 2、用户可以在密文文本框和密钥文本框输入密文和密钥，点击“解密”按钮，执行解密操作。