奈纹摩尔 VIP
码龄7年 南京邮电大学
2 原创 | 40万+ 周排名 | 191万+ 总排名 | 2543 访问 | 等级
51 积分 | 1 粉丝 | 3 获赞 | 0 评论 | 10 收藏
私信 关注

搜博主文章

热门文章
Enigma加解密算法实现C++ 2224
用Virtualbox搭建固件分析平台 318

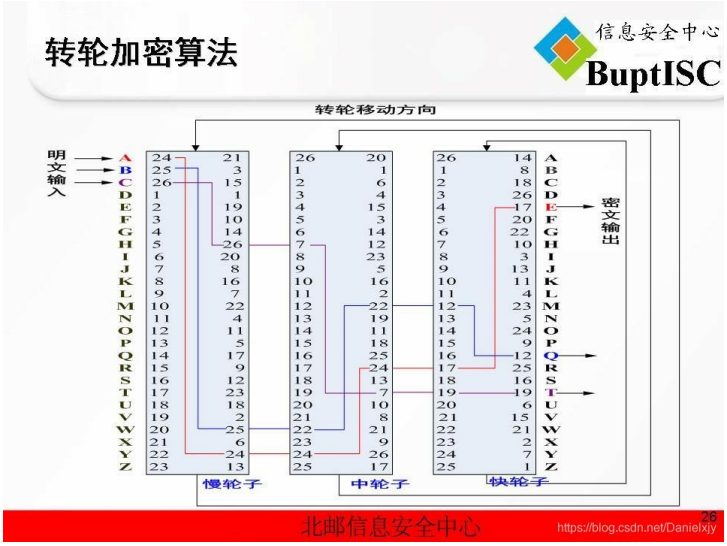
您愿意向朋友推荐“博客详情页”吗？
强烈不推荐 不推荐 一般般 推荐 强烈推荐

Enigma加解密算法实现C++

原创 奈纹摩尔 于 2020-03-06 10:13:07 发布 2225 收藏 7 版权
文章标签： c++ 算法 密码学 加密解密

Enigma加解密算法实现C++

刚刚接触密码学，就打算用C++实现一下Enigma的加解密。加密的思想就是非常经典的多表代换。
德国二战时期的密码系统：亚瑟·谢尔比乌斯
为了战胜enigma，英国在布莱榭丽公园的小木屋里建起了密码学校，这里聚集着各种不同寻常的怪才数学家、军事家、心理学家、语言学家、象棋高手、填字游戏专家，有些人专门负责处理细节，有些人则通过不合常理的思维跳跃来寻找灵感，到二战结束时，这里已经聚集了7000人。
二战成立密码史上的黄金时代。军事科学家估计，盟军对密码的成功破译使得二战至少提前一年结束。
二战结束后，英国并没有透露ENIGMA被破解的秘密，知道二十世纪70年代，各国转向计算机加密的研究，人们才知道布莱榭丽公园的故事。但那时，很多无名英雄已经长眠地下。这其中，天才密码学家图灵的命运最为不幸，他不但没有因为破译受到嘉奖，反而因为“同性恋”而被政府以有伤风化罪起诉。1954年，身心疲惫的图灵服毒自杀，时年42岁。今天，信息学领域最重要的奖项被命名为“图灵奖”，也许是对他的补偿吧。



慢轮子、中轮子和快轮子分别用三个循环链表实现。下面给出慢轮子的定义。

```cpp
#include "Node.h"
class Swheel{
private:
  Node *first;
    int _key;
public:
    Swheel();
    ~Swheel();
    void run();
    int key();
    int encrypt(int);
    int decode(int);
};
Swheel::Swheel(){
    // 定义表头
  first = new Node(23, 13, nullptr);
    // 定义指针
  Node *p = first;
    // 定义轮
    p->link = new Node(22, 24, nullptr);p = p->link;
        p->link = new Node(21, 6, nullptr);p = p->link;
        p->link = new Node(20, 25, nullptr);p = p->link;
        p->link = new Node(19, 2, nullptr);p = p->link;
        p->link = new Node(18, 18, nullptr);p = p->link;
        p->link = new Node(17, 23, nullptr);p = p->link;
        p->link = new Node(16, 12, nullptr);p = p->link;
        p->link = new Node(15, 9, nullptr);p = p->link;
        p->link = new Node(14, 17, nullptr);p = p->link;
        p->link = new Node(13, 5, nullptr);p = p->link;
        p->link = new Node(12, 11, nullptr);p = p->link;
        p->link = new Node(11, 4, nullptr);p = p->link;
        p->link = new Node(10, 22, nullptr);p = p->link;
        p->link = new Node(9, 7, nullptr);p = p->link;
        p->link = new Node(8, 16, nullptr);p = p->link;
        p->link = new Node(7, 8, nullptr);p = p->link;
        p->link = new Node(6, 20, nullptr);p = p->link;
        p->link = new Node(5, 26, nullptr);p = p->link;
        p->link = new Node(4, 14, nullptr);p = p->link;
        p->link = new Node(3, 10, nullptr);p = p->link;
        p->link = new Node(2, 19, nullptr);p = p->link;
        p->link = new Node(1, 1, nullptr);p = p->link;
        p->link = new Node(26, 15, nullptr);p = p->link;
        p->link = new Node(25, 3, nullptr);p = p->link;
        p->link = new Node(24, 21, first);p->link->link = first;
        // 定义密钥
      _key = 0;
    }
Swheel::~Swheel(){
  Node *p = first;
    for (int i = 0; i < 26; i++){
      p   = first->link;
        delete first;
    first = p;
    }
}
void Swheel::run(){
  first = first->link;
  _key = (_key + 1) % 26;
}
int Swheel::key(){
    return _key;
}
int Swheel::encrypt(int lIndex){
    int rIndex = 0;
  Node *p = first;
  Node *q = first;
    // 输入异常处理
    if (lIndex < 0 || lIndex >= 26)
        return -1;
    // 寻址
    for (int i = 0; i < lIndex; i++)
      p   = p->link;
    while (q->rNum != p->lNum){
      q   = q->link;
    rIndex++;
    }
```


最新文章
用Virtualbox搭建固件分析平台
2020年 1篇 2019年 1篇

```
    return rIndex;
}
int Swheel::decode(int rIndex){
    int lIndex = 0;
 Node *p = first;
 Node *q = first;
    // 输入异常处理
    if (rIndex < 0 || rIndex >= 26)
        return -1;
    // 寻址
    for (int i = 0; i < rIndex; i++)
      p   = p->link;
    while (q->lNum != p->rNum){
      q   = q->link;
    lIndex++;
    }
    return lIndex;
}
```

节点类

```
#pragma once
class Node{
public:
    int lNum;
    int rNum;
 Node *link;
        Node(int left, int right, Node *l);
        Node();
};
Node::Node(int left, int right, Node *l) : lNum(left), rNum(right), link(l) {}
Node::Node() : lNum(0), rNum(0), link(0) {}
```

对加密解密算法经行封装

```
#include "Swheel.h"
#include "Mwheel.h"
#include "Qwheel.h"
#include <string>
using namespace std;
class Enigma{
private:
 Swheel s;
 Mwheel m;
 Qwheel q;
    void run();
    void loadKey(string);
    int char2index(char);
public:
 string encrypt(string, string);
 string decode(string, string);
};
void Enigma::run(){
    q.run();
    if (q.key() == 0) {
        m.run();
        if (m.key() == 0)
            s.run();
    }
    return;
}
int Enigma::char2index(char data){
    int index = -1;
    if (data >= 'A' && data <= 'Z')
    index = 'Z' - data;
    if (data >= 'a' && data <= 'z')
    index = 'z' - data;
    return index;
}
void Enigma::loadKey(string key){
    // 密钥异常判定
    if (key.size() < 3)
     key = "ZZZ";
    // 慢轮密钥
    int sKey = char2index(key[0]);
    if (sKey == -1)
     sKey = 0;
    while (s.key() != sKey)
        s.run();
    // 中轮密钥
    int msKey = char2index(key[1]);
    if (mKey == -1)
      mKey = 0;
    while (m.key() != mKey)
        m.run();
    // 快轮密钥
    int qKey = char2index(key[2]);
    if (qKey == -1)
      qKey = 0;
    while (q.key() != qKey)
        q.run();
    return;
}
string Enigma::encrypt(string plainText, string key){
    // 密文
 string cipherText = "";
    // 明文序号
    int index = 0;
    // 加载密钥
    loadKey(key);
    // 加密
    for (int i = 0; i < plainText.size(); i++) {
    index = char2index(plainText[i]);
        if (index == -1)
     cipherText += plainText[i];
        else
     cipherText += 'Z' - q.encrypt(m.encrypt(s.encrypt(index)));
        run();
    }
    return cipherText;
}
string Enigma::decode(string cipherText, string key){
    // 明文
 string plainText = "";
    // 密文序号
    int index = 0;
    // 加载密钥
    loadKey(key);
    // 解密
    for (int i = 0; i < cipherText.size(); i++) {
    index = char2index(cipherText[i]);
        if (index == -1)
     plainText += cipherText[i];
        else
     plainText += 'Z' - s.decode(m.decode(q.decode(index)));
        run();
    }
    return plainText;
}
```

加解密主函数，下面仅给出加密主函数

```
#include "Enigma.h"
#include <iostream>
using namespace std;
```

```cpp
int main(){
    // 创建密码机
 Enigma machine;
    // 输入缓冲
    char input = '\0';
    // 明文
 string plainText = "";
    // 密钥
 string key = "";
    // 密文
 string cipherText = "";
    // 输入
 cout << "Please input your encrypt key:" << endl;
    while(true) {
    input=cin.get();
        if(input=='\n')
            break;
    key  += input;
    }
 cout << "Please input that you want to encrypt:" << endl;
    while(true) {
    input=cin.get();
        if(input=='\n')
            break;
   plainText += input;
    }
    // 加密
 cipherText = machine.encrypt(plainText, key);
 cout << "After encrypt:" << endl;
 cout << cipherText << endl;
    system("pause");
    return 0;
}
```

源代码会上传到资源。