课程编号： B080209030

信息安全工程实践 3
总结报告

姓　　名	薛旗	学　　号	20155362
班　　级	软信-1503	指 导 教 师	徐剑
实 验 名 称	信息安全工程实践 3		
开 设 学 期	2017-2018 第 一 学 期		
开 设 时 间	第 1 周 — — 第 3 周		
报 告 日 期	2017 年 9 月 15 日		
评 定 成 绩		评　　定　　人	
		评　　定　　日　　期	

东北大学软件学院

一、 实践目的与意义

目的：

1）加深对密码学基本原理和加解密理论知识的理解。

2）提高在计算机上实现不同类型加密、解密操作运算功能的能力，实现基本的密钥分配方法。

3）具备简单网络程序开发能力，能够在两个联网终端进行消息加密解密操作。

4）提高分析设计能力、团队合作能力、组织管理能力和语言表达能力。

意义：

通过实践，不仅能够验证理论知识，加深对理论知识的理解，还能通过实践加强实验手段和实践技能，培养分析问题、解决问题、应用知识的能力和创新精神，提高综合素质。

二、 实践环境

操作系统：Windows 10 专业版

开发工具：Qt Creator

编译器：MinGW

调试软件：GNU gdb for MinGW

版本号：4.3.1

上机地点：信息学馆 B405

三、 项目组分工

承担任务：程序界面框架设计，单机界面设计，代码移植，古典加密，DES，MD5，DH 密钥交换

四、 系统设计

算法设计：

（一）古典密码（Vigenere Cipher，Column Permutation Cipher）

（1）Vigenere Cipher

原理：首先构造一个或多个密文字母表，然后用密文字母表中的字母或字母组来代替明文字母或字母组，各字母或字母组的相对位置不变，但其本身改变了。

Vigenere Cipher 数学表达式：

设 m 是一个正整数，定义$\mathrm{P}=\mathrm{C}=\mathrm{K}=(Z_{26})^{m}$ 对任意的密钥

$\mathrm{K}=(k_1,k_2,\ldots,k_m)$,定义

$e_k(x_1,x_2,\ldots,x_m)=(x_1+k_1,x_2+k_2,\ldots,x_m+k_m)$和

$d_K(y_1, y_2, \ldots, y_m) = (y_1 - k_1, y_2 - k_2, \ldots, y_m - k_m)$,以上所有运算都是在$Z_{26}$上进行。

Vigenere Cipher 加密/解密原理：在一个具有密钥字长度为 m 的 Vigenere Cipher 中，一个字母可以被映射为 m 个字母中的某一个(假定密钥字包含 m 个不同的字母)。Vigenere Cipher 使用一个词组作为密钥，密钥中的每一个字母确定一个代替表，每一个密钥字母用来加密一个明文字母。等所有密钥字母都使用玩后，密钥再循环使用。Vigenere Cipher 的替代规则是用明文字母在 Vigenere 方阵中的列和密钥字母在 Vigenere 方阵中的行的交点处的字母来代替该明文字母。其解密就是利用 Vigenere Table 进行反替代。

（2）Column Permutation Cipher

原理：把明文中的字母重新排列，字母本身不变，但其位置改变了，这样编成的密码称为置换密码。置换密码又称为移位密码，因为对照明密文来看，字母的位置被移动了。Column Permutation Cipher 属于置换密码的一种，其加密过程大致如下：把明文按某一顺序排列成一个矩阵，其中不足部分用符号Φ填充。然后按另一顺序选出矩阵中的字母以形成密文，最后截成固定长度的字母组成密文。由加密过程可知，改变矩阵的大小和选出顺序可以得到不同形式的密码。

（二）DES

DES 算法中数据以 64 比特分组进行加密，有效密钥长度为 56 位。它的加密算法与解密算法相同，只是解密子密钥与加密子密钥的使用顺序刚好相反。

（三）MD5

MD5 算法以任意长度的消息文件为输入，产生一个 128 比特的摘要作为输出，在算法内部，消息是按照 512 比特进行分组处理的。

（四）DH 密钥交换

DH 密钥交换协商机制通过两个回合的传输来建立通信双方之间的共享密钥。协议提供了一种联合的密钥控制，使参与协议的双方谁也不能单独确定生成的密钥。DH 协议的安全性源于在有限域中计算离散对数比计算指数更为困难。首先参与协议的双方 A 和 B 协商一个大的质数 p 和 g，g 是模 p 的原根。这两个整数不必是秘密的，所以 A 和 B 可以通过即使是不安全的途径协商。

功能设计：

（一）古典密码（Vigenere Cipher，Column Permutation Cipher）

（见源码：classical.cpp）

Vigenere Cipher：

要实现对 Vigenere Cipher,关键是找到对应关系。因此，其中最重要的一项工作是构造 Vigenere Table,之后便可根据密钥与明文一一对应的关系在表中找对应的字符。假定密钥确定的是行替代表，那么密钥所确定的行和明文所确定的列的交叉位置处的字符，即为一个明文字符加密得到的密文字符。对于解密过程则与加密过程相反，由密钥先确定一个替代表（假定密钥确定的行替代表），然后在该表中找到密文字符，密文字符所在列对应的字符即为明文。这里还需要注意的是密钥是循环使用的，密钥长度与明文一致。

Column Permutation Cipher

要实现 Column Permutation Cipher 的加密/解密，关键是构造明文/密文矩阵并能够得到一个合适的选出规则。在编程中通过输入一串英文字符来完成以上操作，具体步骤为：首先选用一串英文字符作为密钥，构造的矩阵的列数与字符串长度保持一致，明文/密文的长度不足以构造矩阵的则用Φ填充。按照字母的字典顺序给密钥字母编号，对于重复的字母，谁的位置靠前，谁的编号就靠前。于是我们就得到一组与密钥词语对应的数字序列。最后据此数字序列中的数字顺序按列选出明文/密文。

（二）DES（见源码：DES.cpp）

实验最关键的地方在于对明文的分块处理、二进制转化、盒子操作、以及子密钥的产生和十六次加密迭代使用。因此，在进行实验时目标明确，先处理密钥，生成 16 个 48 位的子密钥，再处理明文，接着实现盒子操作与加密函数 f，然后迭代 16 次，产生 64 位数据组，之后对数据组进行处理后，经过逆初始置换，得到 64 位明文，至此一组加密过程全部结束。在本实验中，定义以下规则：(1)将初始明文/密文分组并依次转化为二进制比特流之后，若分组内比特流长度不足 64 位，则补 0。(2)对密钥的操作处理：输入 7 位英文字符，得到其对应的 ASCII 码，将 ASCII 码用二进制表示，最终得到 56 位二进制比特流，将此作为 DES 的 56 位密钥。之后在第 8，16，

24，32，40，48，56，64 位处插入奇偶校验位，合成 64 位密钥。

（三）MD5（见源码：MD5.cpp）

MD5 算法包括以下几个步骤：

（1）附加填充比特。对输入消息进行填充，使消息长度与 448 模 512 同余。需要注意的是，如果消息是 448 比特，那么需要填充 512 比特以形成 960 比特的比特串。填充的比特最高位是 1，其余都是 0.

（2）附加长度值。用 64 比特的数字表示填充前消息的长度，将这 64 比特附加在前面填充后的比特串后面，这样，比特串的长度将是 512 比特的整数倍。经过此扩展的比特串以 512 比特的分组形成序列$Y_1,Y_2,\cdots,Y_L$，整个比特串的长度为 512L 比特。

（3）初始化链接变量缓冲区。一个 128 位的链接变量缓冲区用于保存链接变量和最终哈希函数输出。链接变量缓冲区可以表示为 4 个 32 比特的寄存器 A、B、C 和 D，这些寄存器初始化为以下十六进制值：A=67452301，B=EFCDAB89，C=98BADCFE，D=10325476,这些值在寄存器中以低位在前的方式存储，即字的低位字节放在低地址字节上，因此，寄存器的初始值是这样存储的:A=01 23 45 67，B=89 AB CD EF，C=FE DC BA 98，D=76 54 32 10

（4）HMD5 依次处理每一个 512 比特的消息分组，如果消息比特串分组树为 L，则需要执行 L 次 HMD5 以处理每一个分组。

(5)所有 L 个 512 比特的分组处理完成后，第 L 个阶段产生的输出 CV_L 便是最后输出的 128 比特的摘要。

（四）DH 密钥交换（见源码：big_int.h， table.h， DH.cpp）

DH 密钥交换实现步骤如下：

（1）A 选取一个大的随机整数 x,并且发送给 B：$X = g^x mod\ p$；

（2）B 选取一个大的随机整数 y,并且发送给 A:$Y = g^y mod\ p$；

（3）A 计算$k_1 = Y^x mod\ p$；

（4）B 计算$k_2 = X^y mod\ p$。

可以得到$k_1 = k_2$。

界面设计：

概要：界面采用 Qt 集成开发环境 Qt Creator，将加密解密算法代码整合到界面程序中。服务端和客户端采用相同的模板框架，主界面分为两块：登录界面和操作界面。

详细设计：

（1）编写代码，设计程序框架布局、样式风格及皮肤属性，对消息对话框弹窗风格进行重定义，对默认字体及文本框样式进行自定义。实现代码：frmmain.h，frmmessagebox.h，iconhelper.h，frmmain.cpp， frmmessagebox.cpp，iconhelper.cpp。

（2）登陆界面：显示校徽及学校名称，展示课程名称、班级及小组成员，另有“进入系统”和“退出程序”按钮。如果按下“进入系统”，则进入加密解密主界面，如按下“退出程序”，则退出。此界面实现了基本信息展示以及从登陆对话框到主界面的功能。（见源码：mydialog.ui）

(3)主界面：主界面采用多界面切换，添加 tabWidget 控件，通过 QTabWidget 类实现。每一个界面实现不同的功能，包含古典密码、RC4、DES、RSA、MD5、数字签名、密钥交换、双机通信、文件传输、安全通信等标签。（见源码：frmmain.h， frmmain.cpp，frmmain.ui）

古典密码界面采用抽屉式风格，将不同类型的古典密码归类到不同的抽屉中，实现了分类管理，更加高效。此部分通过添加 toolBox 控件，通过 QToolBox 类实现。

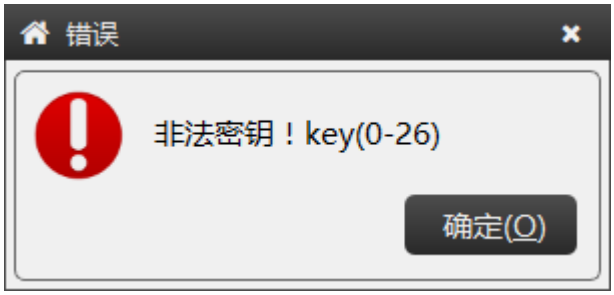
DES 界面新增二进制明文展示及二进制密文展示

DH 密钥交换通过布局设计，可动态展示从生成参数到交换密钥，再到计算共享密钥的全过程。

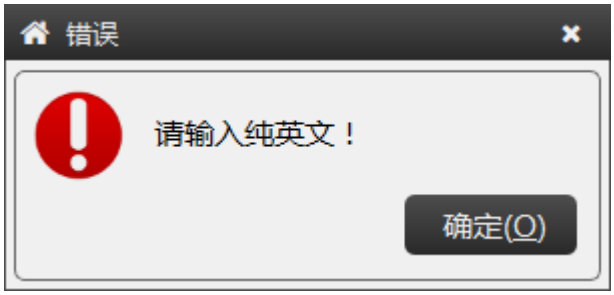
安全通信界面则可实战模拟通信双方从交换密钥到加密通信的全过程。

加解密代码与界面的融合，主要采用信号和槽机制实现，通过给 Push Button 添加 clicked()信号，并转到对应的槽函数，编写程序。通过字符间的转换，将加解密结果转换成 QString 类型，调用函数将加解密结果显示到界面上，完成界面制作，实现功能。

（4）消息对话框：对输入的错误信息进行提示，对输入字符进行限制。通过 QMessageBox 类提供的一个模态对话框来通知错误信息。（见源码：frmmessagebox.h， frmmessagebox.cpp， frmmessagebox.ui）



（密钥限制）



（明文限制）

五、 系统实现

环境配置：

构建套件(Kit)：Desktop Qt 5.9.1 MinGW 32bit

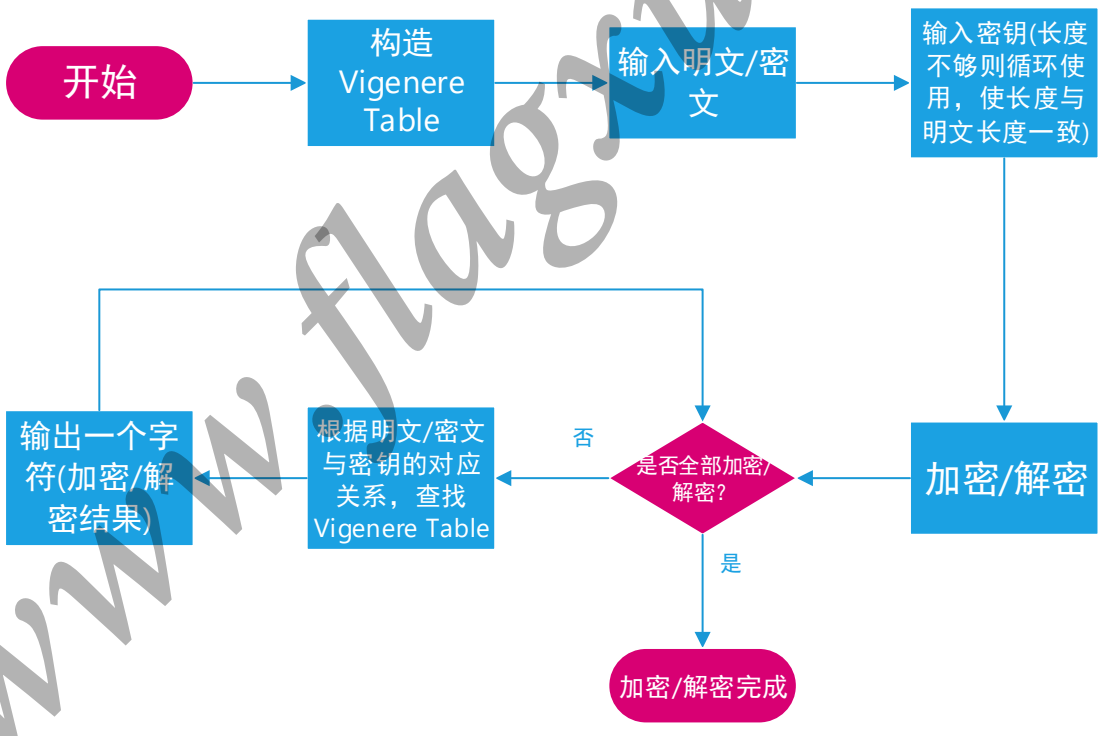
编译器：MinGW 5.3.0 32bit for C/C++

构建:Release

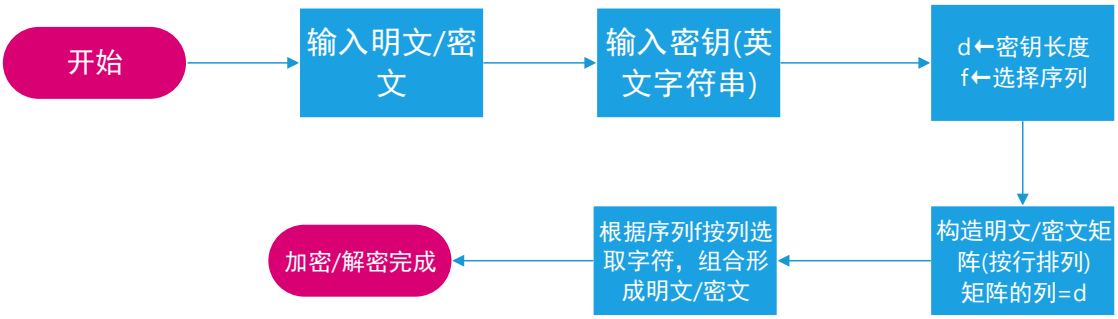
调试器：GNU gdb 7.10.1 for MinGW 5.3.0 32bit

程序流程图：

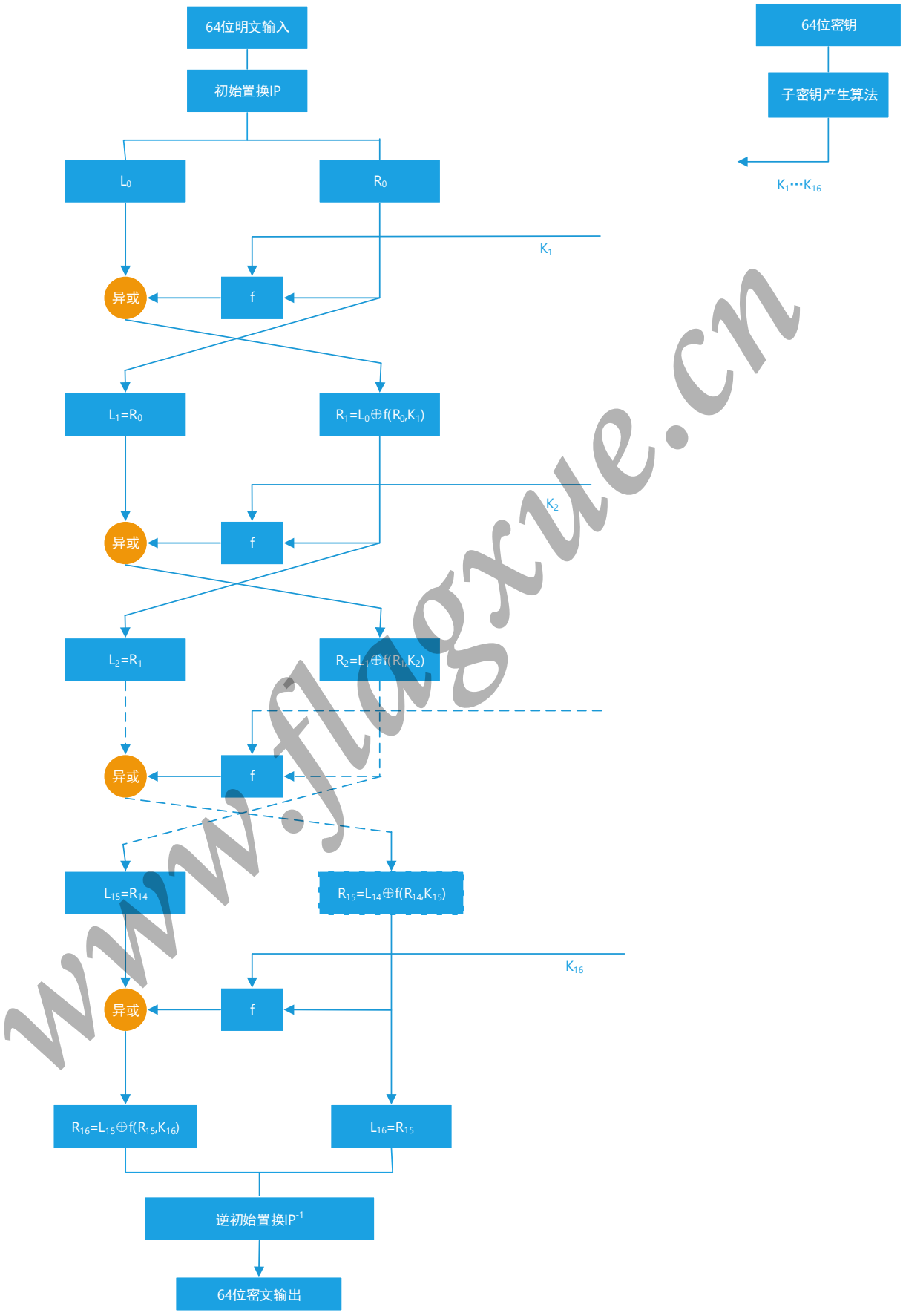
Vigenere Cipher



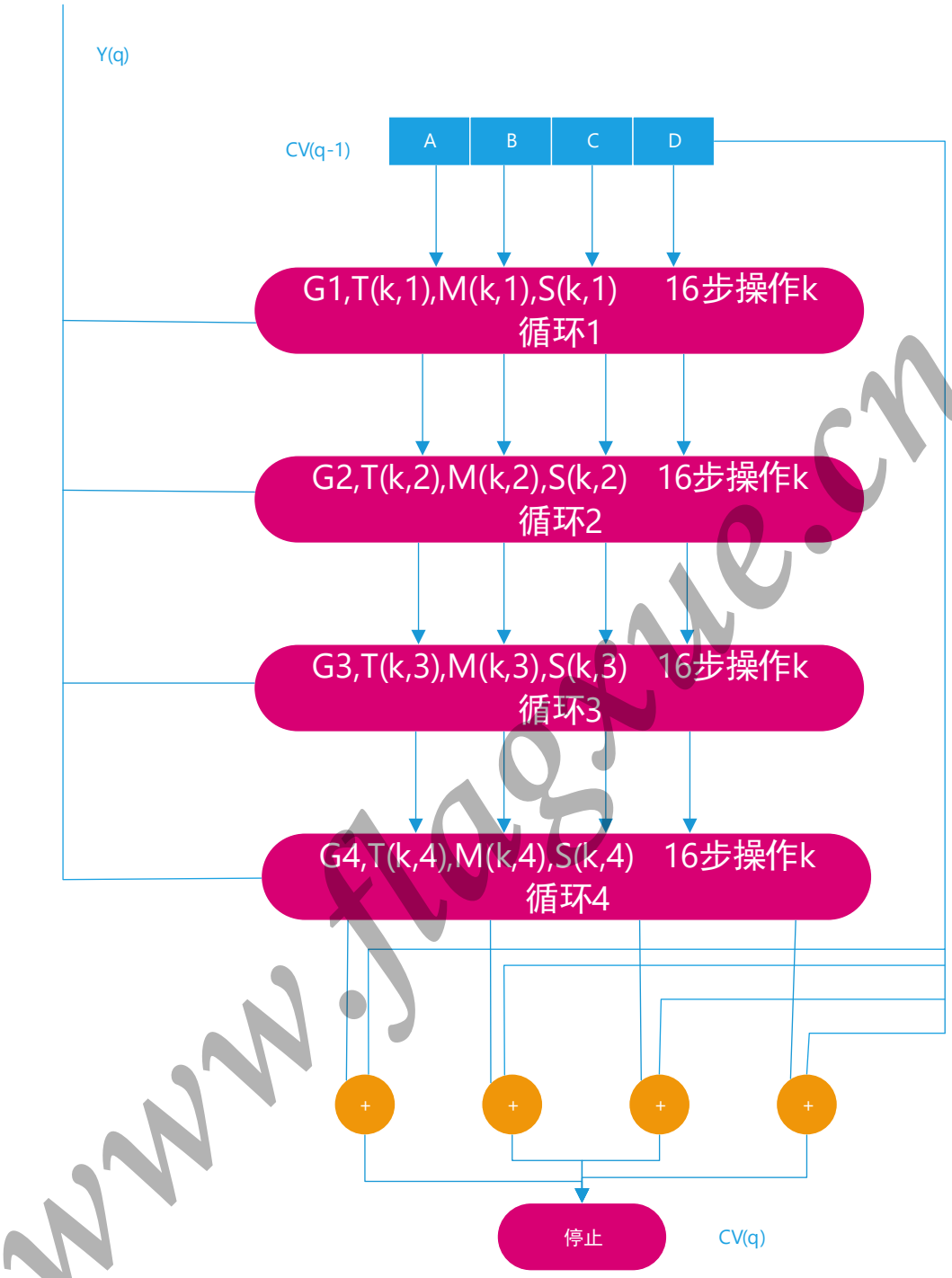
Column Permutation Cipher



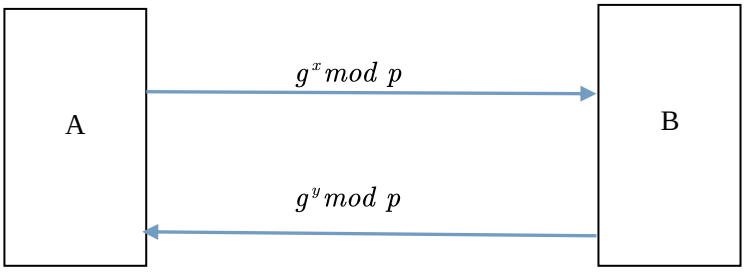
DES



MD5



DH 密钥交换



主要界面：

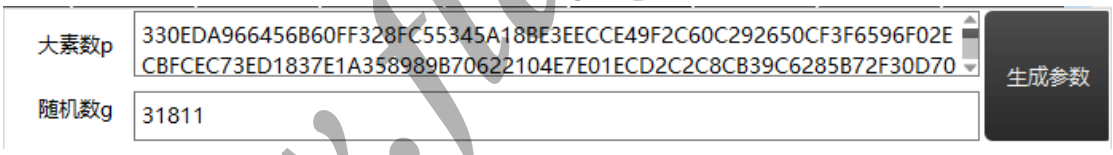
古典密码加解密：

DES

MD5

密钥交换：

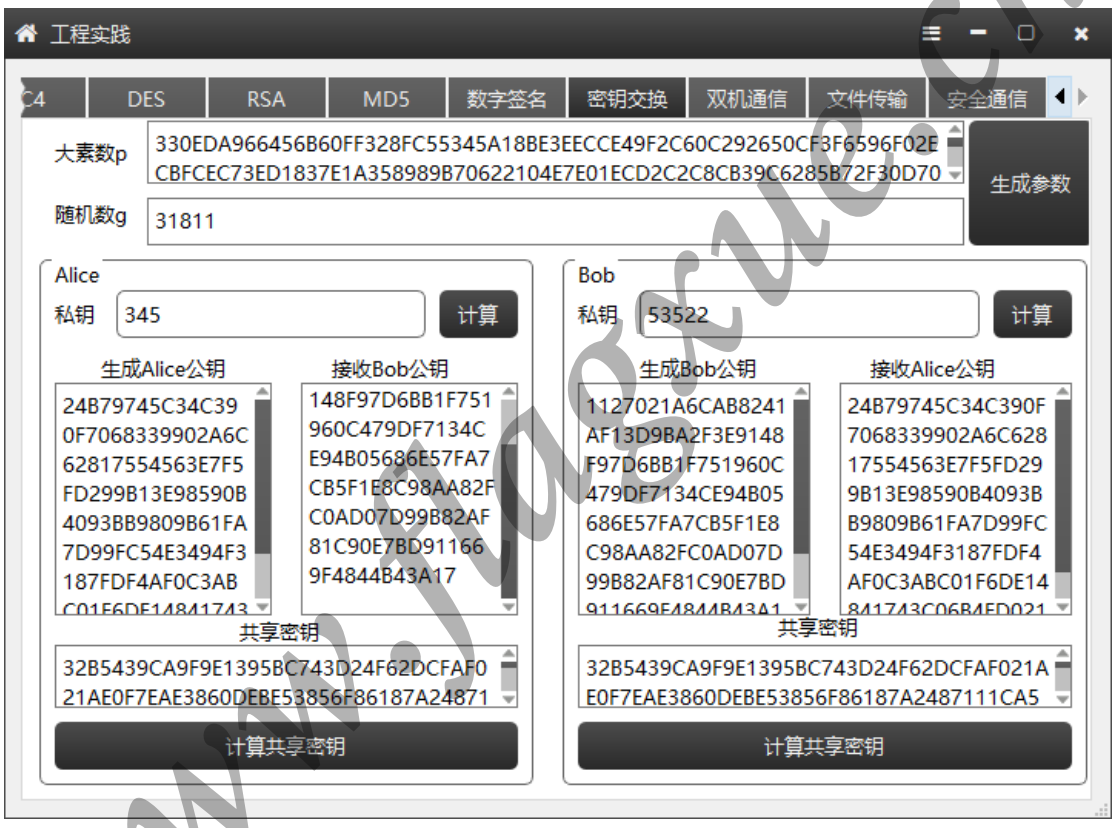
(1)生成参数：



(2)Alice 选取私钥，计算公钥并发送给 Bob

(3)Bob 选取私钥，计算公钥并发送给 Alice

(4)计算共享密钥，二者相等



关键代码：

（古典）构建 Vigenere Table

```
char(*vigenereTable(char(&list)[26][26]))[26]{        //返回二维数组
int i, j, k;
char ch;
for (i = 0; i < 26; i++) {
    for (j = 0; j < 26; j++) {
        k = 'a' + (i + j) % 26;
        ch = k;
        list[i][j] = ch;
    }
```

```
    }
    cout << "  a b c d e f g h i j k l m n o p q r s t u v w x y z" << endl;
    for (i = 0; i < 26; i++) {
        cout << list[0][i] << " ";
        for (j = 0; j < 26; j++) {

            cout << list[i][j] << " ";
        }
        cout << endl;
    }
    cout << endl;
    return list;
 }
```

（古典）计算模乘法逆元

```
int e_gcd(int a, int b, int &x, int &y) {
    if (b == 0) {

        x = 1;
        y = 0;
        return a;
    }
    int ans = e_gcd(b, a%b, x, y);
    int temp = x;
    x = y;
    y = temp - a / b*y;
    return ans;
}

int cal(int a, int m) {
    int x, y;
    int gcd = e_gcd(a, m, x, y);
    if (1 % gcd != 0)
        return -1;
    x *= 1 / gcd;
    m = abs(m);
    int ans = x%m;
    return ans;
 }
```

（DES）明文/密文分块，二进制转换

```
void plainTrans(string st, int block, int(&plaintext)[800][64], int way) { //  实现明文/密文分块和二进制转化
    unsigned int i, k, t, m, n, j;
```

```
    int flag = 0;
    int temp[56] = { 0 };

    //将明文分块，存入二维数组
    n = 0;
    k = st.length() % 8;
    if (k == 0)
        k = 8;
    m = block;
    for (i = 0; i < st.length(); i++) {
        t = (unsigned int)st[(st.length() - 1 - i)];
        if (t > 256) {
            t = t - 4294967040;        //若字符的ASCII超出范围，则减去过余码
        }
        for (j = 0; j < 8; j++) {
            if (n == 64) {
                n = 0;
                m--;
            }
            if (m != block) {
                plaintext[m][(64 - 1 - n)] = t % 2;
            }
            else {
                plaintext[m][(8 * k - 1 - n)] = t % 2;
                if (n == (8 * k - 1)) {    //明文最后一个分块，不足64位补0
                    n = -1;
                    m--;
                }
            }
            t /= 2;
            n++;
        }
    }
    cout << endl;

    //输出分块明文比特流
    if (way == 0) {
        cout << "--------------------------------Binary Plaintext
Block--------------------------------" << endl << endl;
    }
    else {
        cout << "--------------------------------Binary Ciphertext
Block--------------------------------" << endl << endl;
    }
```

```
    for (i = 0; i < block + 1; i++) {
        cout << "Block" << i + 1 << ":";
        for (j = 0; j < 64; j++) {
            cout << plaintext[i][j];
        }
        cout << endl;
    }
    cout << endl;
}
```

（DES）密钥二进制转换输出

```
int *keyTrans(string key, int keytemp[64]) {//密钥二进制转换并输出
    int i, k, t, m, n, q;
    int block;
    int flag = 0;
    int temp[56] = { 0 };
    int *p = keytemp;
    m = 0;
    for (k = 0; k < 7; k++) {
        t = key[6 - k];        //二进制转换
        for (i = 0; i < 8; i++) {

            temp[56 - 1 - m] = t % 2;
            t /= 2;
            m++;
        }
    }
    cout << endl;

    cout << "----------------------------------Key
Information----------------------------------" << endl << endl;
    cout << "Key(56bit): ";        //56位密钥
    for (i = 0; i < 56; i++) {
        cout << temp[i];
        if ((i + 1) % 7 == 0)
            cout << " ";
    }
    cout << endl << endl;

    n = 0;
    m = 0;
    q = 0;
    cout << "Key(64bit)(With odd parity)" << endl;   //   显示有奇偶校验位的64位密钥
```

```
    cout << "Key(64bit): ";
    for (i = 0; i < 56; i++) {
        cout << temp[i];
        keytemp[q] = temp[i];
        q++;
        if (temp[i] == 0) {
            n++;
        }

        if (((m + 1) % 7 == 0) && (n % 2 != 0)) {
            cout << "1 ";
            keytemp[q] = 1;   //将64位密钥存入数组
            q++;
            m++;
            n = 0;
            continue;
        }
        else if (((m + 1) % 7 == 0) && (n % 2 == 0)) {
            cout << "0 ";
            keytemp[q] = 0;
            q++;
            m++;
            n = 0;
            continue;
        }
        m++;
    }
    cout << endl << endl;
    return p;
}
```

（MD5）定义运算法则

```
#define F(x, y, z) ((x & y) | ((~x) & z))
#define G(x, y, z) ((x & z) | (y & (~z)))
#define H(x, y, z) (x ^ y ^ z)
#define I(x, y, z) (y ^ (x | (~z)))
#define FF(a, b, c, d, m, s, t) a = b + ROL((a + F(b, c, d) + m + t), s)
#define GG(a, b, c, d, m, s, t) a = b + ROL((a + G(b, c, d) + m + t), s)
#define HH(a, b, c, d, m, s, t) a = b + ROL((a + H(b, c, d) + m + t), s)
#define II(a, b, c, d, m, s, t) a = b + ROL((a + I(b, c, d) + m + t), s)
#define ROL(x, s) ((x << s) | (x >> (32 - s)))
```

（DH 密钥交换）素性检验

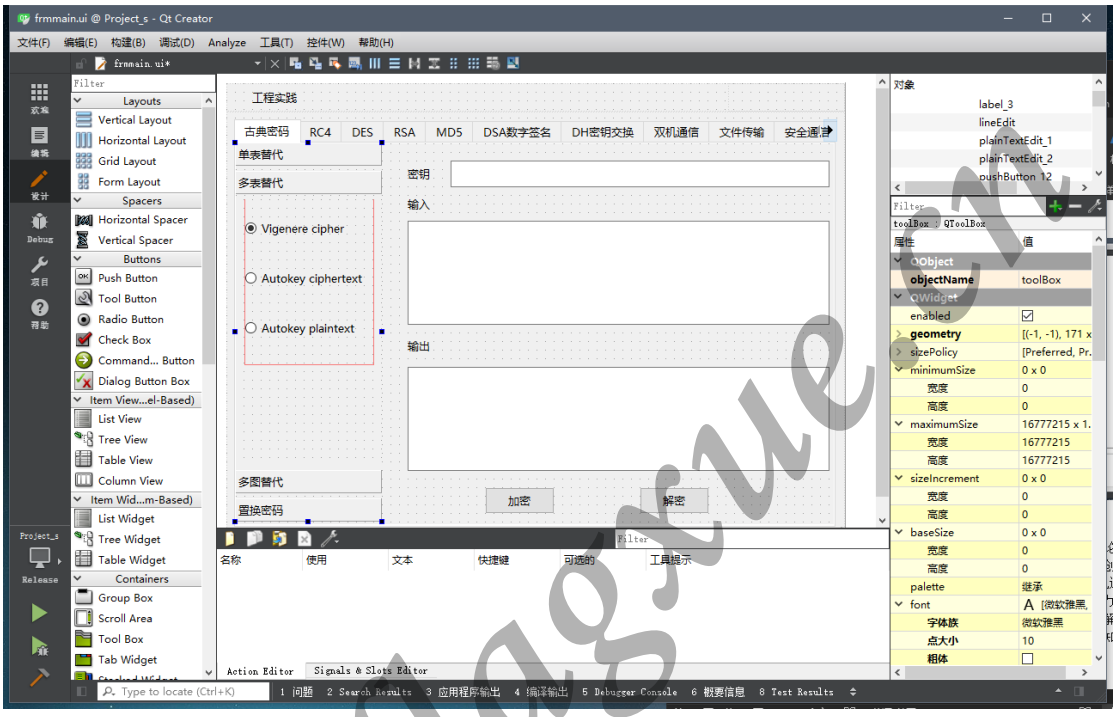
```
long remainder(big_int &a, unsigned long b)
{
    return a%b;
}

int Rabin_miller(big_int &x)
{
    for (int i = 0; i<1000; i++) { if (remainder(x, table[i]) == 0) return 0; }
    big_int s, a, r, z;
    z = x;
    z.value[0]--;
    for (int i = 0; i<5; i++)
    {
        a = rand()*rand();
        s = z / 2;
        r = a.power_mod(s, x);
        if (r != 1 && r != z)
            return 0;
    }
    return 1;
 }
```

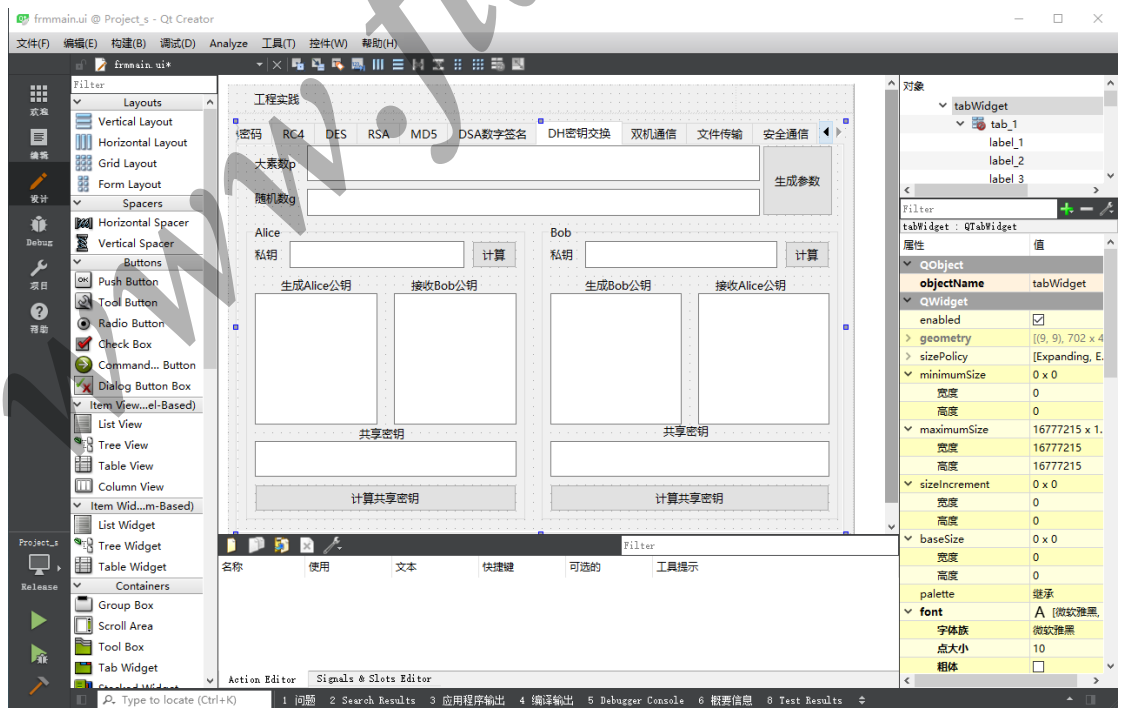
（界面）设置消息对话框

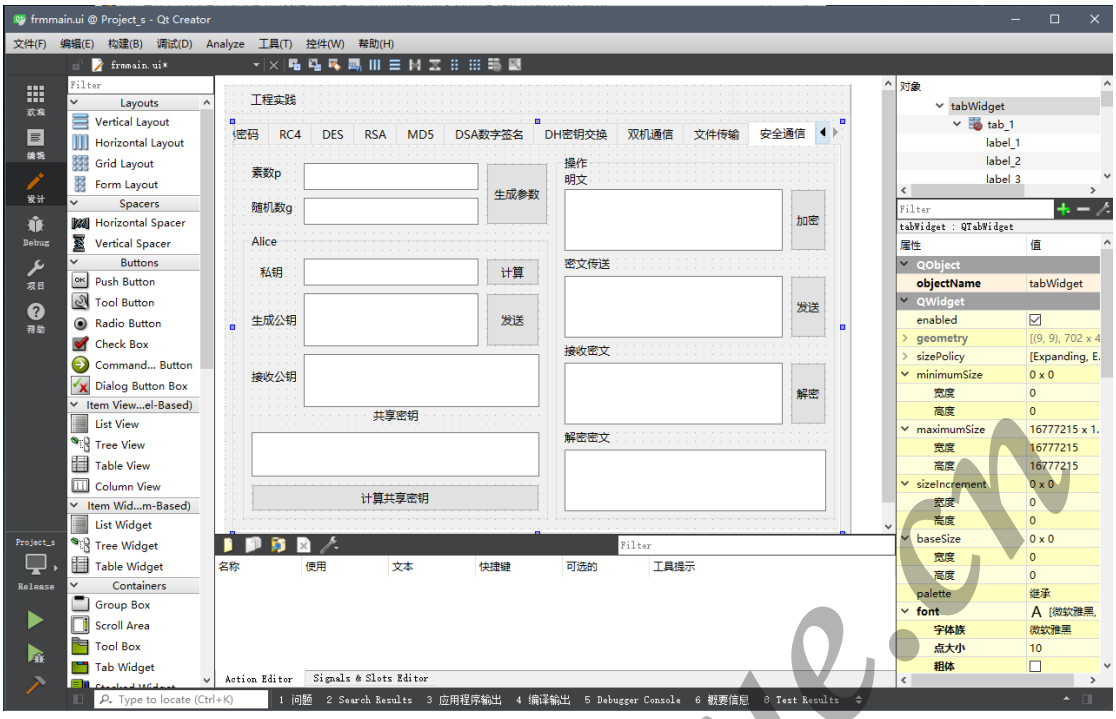
```
frmMessageBox::frmMessageBox(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::frmMessageBox)
{
    ui->setupUi(this);

    this->mousePressed = false;
    //设置窗体标题栏隐藏
    this->setWindowFlags(Qt::FramelessWindowHint);
    //设置窗体关闭时自动释放内存
    this->setAttribute(Qt::WA_DeleteOnClose);
    //设置图形字体
    IconHelper::Instance()->SetIcon(ui->lab_Ico, QChar(0xf015), 12);
    IconHelper::Instance()->SetIcon(ui->btnMenu_Close, QChar(0xf00d),
10);
    //关联关闭按钮
    connect(ui->btnMenu_Close, SIGNAL(clicked()), this,
SLOT(close()));
    connect(ui->btnCancel, SIGNAL(clicked()), this, SLOT(close()));
    //窗体居中显示
```

```
    myHelper::FormInCenter(this);
}
```

附：Qt Creator 设计师界面开发面板







六、 结束语

本次实践，获益匪浅，通过实践，不仅回顾了知识点，够验证理论知识，还加强了实验手段和实践技能，培养分析问题、解决问题、应用知识的能力和创新精神，提高综合素质。本次实践，实现了古典密码、现代密码的加解密，实现了双机通信加解密功能，还自学了 Qt 界面设计，开发了加解密的图形化界面，考验了自学能力，提高了编程水平，对我综合能力的提高有极大的帮助。实践课的开设，对我深入理解密码学的内涵有很大的帮助，理论与实践结合，最能体现课程开设的意义。实践出真知，希望以后可以多多开设实践课，多多实践，提高综合能力。

七、 参考文献

1.Douglas R.Stinson:《Cryptography Theory and Practice》(Third Edition)，电子工业出版社，2008

2.张焕国，唐明著：《密码学引论》(第三版)，武汉大学出版社，2015

3.郑东 李祥学等著：《密码学——密码算法与协议》(第二版),电子工业出版社，2013

4.Michael Welschenbach:《Cryptography in C and C++》(Second Edition),机械工业出版社，2015

5.霍亚飞：《Qt Creator 快速入门》(第三版)，北京航空航天大学出版社，2016

6.霍亚飞，程梁：《Qt5 编程入门》，北京航空航天大学出版社，2015

7.霍亚飞：《Qt 及 Qt Quick 开发实战精解》，北京航空航天大学出版社，2012

8.【韩】金大跴：Qt5 开发实战，人民邮电出版社，2015

附录：

《信息安全程序实践 3》成绩评定表

评价内容	具　体　要　求	分值	得分
报告质量	实验报告格式规范，符合要求；报告内容充实、正确，实验目的归纳合理到位。	10	
平时表现	课程设计过程中，无缺勤现象，态度积极，具有严谨的学习态度和认真、踏实、一丝不苟的科学作风。	10	
提交材料及实验内容	能够按照规范提交课程设计要求的所有材料（要求在以“学号-姓名”命名的文件夹中，包含实验报告电子版和实验源代码等），材料完备，格式内容等符合要求。能够按实验要求合理设计并开发出程序，功能完整性强，原理及实验结果分析准确，归纳总结充分。	50	
答辩情况	答辩认真，清晰，分工明确。	30	
总　　分			