信息安全导论第三次作业

使用QT编写C++可视化程序模拟D-H密钥交换过程，主程序部分代码如下：

#include "figure.h"

#include "ui\_figure.h"

#include "mainwindow.h"

figure::figure(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::figure)

{

ui->setupUi(this);

}

figure::~figure()

{

delete ui;

}

bool prime(int n)

{

if(n <= 1)

{

return false;

}

for (int i = 2; i <sqrt(n); i++)

{

if ((n % i) == 0)

{

return false;

}

}

return true;

}

void figure::on\_pushButton\_clicked()

{

QString text1= ui->lineEdit->text();

QString text2= ui->lineEdit\_2->text();

QString text3= ui->lineEdit\_3->text();

QString text4= ui->lineEdit\_4->text();

if(text1.length()==0)

{

QMessageBox::information(NULL, "错误", "你未输入原根g", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(text1.toInt()==0)

{

QMessageBox::information(NULL, "错误", "你输入的素数p不是数字", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(text2.length()==0)

{

QMessageBox::information(NULL, "错误", "你未输入素数p", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(text2.toInt()==0)

{

QMessageBox::information(NULL, "错误", "你输入的素数p不是数字", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(!prime(text2.toInt()))

{

QMessageBox::information(NULL, "错误", "你输入的素数p不是素数", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(text3.toInt()==0)

{

QMessageBox::information(NULL, "错误", "你输入的私钥a不是数字", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

if(text4.toInt()==0)

{

QMessageBox::information(NULL, "错误", "你输入的私钥b不是数字", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

int g=text1.toInt();

int p=text2.toInt();

int a=text3.toInt();

int b=text4.toInt();

int A=fmod(pow(g, a), p);

int B=fmod(pow(g, b), p);

int S1=fmod(pow(B, a), p);

int S2=fmod(pow(A, b), p);

ui->lineEdit\_5->setText(QString::number(A));

ui->lineEdit\_6->setText(QString::number(B));

ui->lineEdit\_7->setText(QString::number(S1));

ui->lineEdit\_8->setText(QString::number(S2));

if(S1==S2)

{

QMessageBox::information(NULL, "验证成功", "客户端和服务端计算的公钥相同", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

else

{

QMessageBox::information(NULL, "验证失败", "客户端和服务端计算的公钥不相同", QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);

}

}

}

}

}

}

}

}

}

void figure::mousePressEvent(QMouseEvent \*e)

{

if(e->button() == Qt::LeftButton)

{

if(e->x()>840&&e->x()<840+110&&e->y()>10&&e->y()<10+60)

{

this->close();

MainWindow \*f=new MainWindow();

f->show();

}

}

}

Mainwindow窗口进入一个欢迎界面，设置自动播放背景音乐

#include "mainwindow.h"

#include <QApplication>

QSound \*sound= new QSound(":/backmusic2.wav");

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

sound->setLoops(100);

sound->play();

w.show();

return a.exec();

}



点击主程序窗口的进入，则跳转到D-H密钥交换窗口

点击退出，则退出程序

进入密钥交换窗口：

日程表

描述已自动生成

首先需要输入双方协定好的原根g和素数p

点击开始计算会逐一检验输入的合法性（没有输入则会显示你未输入某某）

图形用户界面

描述已自动生成

若输入的不是数字也会显示输入不是数字

图形用户界面, 应用程序

描述已自动生成

若输入的素数并非素数也会判断为你输入的不是素数

图形用户界面

描述已自动生成

私钥同理

图形用户界面

描述已自动生成

Step1一开始需要输入双方协定好的原根g和素数p以及双方的私钥a和b

Step2客户端根据私钥a以及双方协定好的原根g和素数p，对a进行幂模运算得到公钥A计算公式如下

Step3客户端将公钥A发送给服务端

Step4客户端根据私钥b以及双方协定好的原根g和素数p，对b进行幂模运算得到公钥B计算公式如下

Step5服务端将公钥B发送给客户端

Step6双方知道A和B的情况下分别计算公钥S1，S2

客户端已知：g、p、A、B、a 

服务端已知：g、p、A、B、b 

若计算出的S1等于S2则验证成功双方计算的公钥相同

计算前：



计算后：

图形用户界面

描述已自动生成

更换不同参数验证：

图形用户界面, 应用程序

描述已自动生成

由于由A逆推a、B逆推b的难度很大很好的完成了公钥交换的任务