Министерство образования и науки Российской Федерации

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Клиент-серверное приложение для обмена сообщениями

Пояснительная записка

RU.02068048.502900.01.01

На 23 страницах

|  |  |  |
| --- | --- | --- |
| Руководитель |  | к.т.н. доцент кафедры ИЗИ Ю.М. Монахов |
|  |
| Исполнитель |  | студент гр. ИБ-116 А.В.Шахина |

Владимир 2018

# СОДЕРЖАНИЕ

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

2

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

[АННОТАЦИЯ 3](#_Toc483750422)

[СТРУКТУРА ПРОГРАММНОГО ПРОДУКТА 4](#_Toc483750423)

[ПОЯСНЕНИЯ ПО РАБОТЕ ОСНОВНЫХ ФУНКЦИЙ СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ 5](#_Toc483750424)

[ПОЯСНЕНИЯ ПО РАБОТЕ ОСНОВНЫХ ФУНКЦИЙ КЛИЕНТСКОЙ ЧАСТИ ПРИЛОЖЕНИЯ 9](#_Toc483750425)

[ТРЕОВАНИЯ К ОПЕРАЦИОННОЙ СИСТЕМЕ 13](#_Toc483750426)

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

[ПЕРЕЧЕНЬ ВЫХОДНЫХ ДАННЫХ 14](#_Toc483750427)

[АУДИТ БЕЗОПАСНОСТИ](#_Toc483750428) 15

ПРИЛОЖЕНИЕ А 22

# АННОТАЦИЯ

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лит

Лист

Листов

3

33

Авиационная корпорация "Рубин"

ШАБЛОН для создания отчетов по ГОСТу

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

NIO280P300

Разраб.

Пров.

Т. контр.

Н. контр.

Утв.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лит

Лист

Листов

3

33

Авиационная корпорация "Рубин"

ШАБЛОН для создания отчетов по ГОСТу

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

NIO280P300

Разраб.

Пров.

Т. контр.

Н. контр.

Утв.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

3

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

В данном программном документе приведено описание функций программы, предназначенной для обмена сообщениями в локальной сети в защищенном режиме, а также аудит безопасности. Исходным языком данной разработки является С++. Текст программы приведён в Приложении А данного программного документа.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Основной функцией клиент-серверного приложения является обеспечение обмена текстовыми сообщениями между пользователями. Для запуска программы необходимо знать пароль как для сервера, так и для клиента.

# СТРУКТУРА ПРОГРАММНОГО продукта

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

4

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Система состоит из двух частей: приложение клиента, которое устанавливается непосредственно на машине пользователя, и приложение сервера, находящееся на специально отведенной машине. С помощью клиентской части пользователи подключаются к серверу для дальнейшего обмена сообщениями с остальными уже подключенными пользователями. Программа защищена от статического и динамического анализа лицензионным ключом и защитой от отладчика соответственно. Для запуска приложения необходимо знать лицензионный ключ от клиентской части и от серверной.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

# ПОЯСНЕНИЯ ПО РАБОТЕ ОСНОВНЫХ ФУНКЦИй серверной части ПРИЛОЖЕНИЯ

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

5

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

После запуска программы сервера вызывается функция, отслеживающая присутствие отладчика, осуществляющая защиту от динамического анализа. Если отладчик запущен, то программа закрывается:

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

if (IsDebuggerPresent()) ExitProcess(0);

CheckRemoteDebuggerPresent(GetCurrentProcess(), &f);

if (f==true) ExitProcess(0);

Далее программа запрашивает ввод пароля от пользователя:

scanf("%s1", &s1);

Затем программа для каждого символа пароля получает его ASCII-код, производит хэширование этого символа и с помощью обратной функции преобразует ASCII-код в символ, который хранится в переменной char w1.

unsigned int Hash37(char str) // функция получения ASCII-кода

{

int hash = (int)(str);

return hash;

}

unsigned int toHash37(int str) //обратная функция Hash37

{

char hash = (char)(str);

return hash;

}

unsigned int function(int p) // хэш-функция

{

int o=(sin(p)/cos(p)+4\*p\*p)/1000+32-(sin(p)\*cos(p)+4\*cos(p)\*5)\*0;

return o;

}

В коде программы хранится «эталонная» строка, с которой и происходит сравнение хэша пароля. Это строка разделена на 4 части и хранится в разных местах кода программы: char str3[]="MW>5"; char str2[]=")@1\*"; char str4[]=":SY,"; char str1[]="+H<K". Полученный хэш введенного пароля посимвольно сравнивается с эталонной строкой, при сравнении идет накапливание в переменную rez.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

for (int i=0; i<=sizeof(s1); i++)

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

5

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

{

int p=Hash37(s1[i]);

int o=function(p);

char w1=toHash37(o);

printf("%c", w1);

if (i<=3)

{

int j=i%3;

if (str3[j]==w1 || str[3]==w1)

rez=rez+1;

}

if(i>3 && i<=7)

{

int j=i%4;;

if (str4[j]==w1 || str4[3]==w1)

rez=rez+1;

}

if(i>7 && i<=11)

{

int j=i%8;

if (str2[j]==w1 || str2[3]==w1)

rez=rez+1;

}

if(i>11 && i<=15 || str1[3]==w1)

{

int j=i%12;

if (str1[j]==w1)

rez=rez+1;

}

}

Далее сравнивается значение переменной rez с установленным значением, которое соответствует длине эталонной строки. Если переменная lkt не соответствует установленному значению, программа прекращает свою работу и закрывается:

if (rez!=16)

ExitProcess(0);

Также в серверной части приложения реализован таймер. На ввод лицензионного ключа отводится 30 секунд и если в установленное время пользователь не успевает ввести пароль, программа закрывается.

Если введенный пароль удовлетворяет необходимым условиям, то далее происходит создание самого сервера.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Первым этапом является инициализация библиотеки для создания сокетов:

int res = WSAStartup(version, &data);

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

7

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

if (res!=0)

{

return 0;

}

Если инициализация библиотеки прошла успешно, тогда происходит создание сокета и присвоение сокету локальный адрес и вызов функции для ожидания подключений к сокету.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

SOCKET Connect;

SOCKET \* Connections;

SOCKET Listen;

struct addrinfo hints;

struct addrinfo \* result;

Connections = (SOCKET\*)calloc(64,sizeof(SOCKET));

ZeroMemory(&hints,sizeof(hints));

hints.ai\_family=AF\_INET;

hints.ai\_flags=AI\_PASSIVE;

hints.ai\_socktype=SOCK\_STREAM;

hints.ai\_protocol=IPPROTO\_TCP;

getaddrinfo(NULL,"5896",&hints,&result);

Listen=socket(result->ai\_family,result->ai\_socktype,result->ai\_protocol);

bind(Listen,result->ai\_addr, result->ai\_addrlen);

listen(Listen,SOMAXCONN);

freeaddrinfo(result);

После создается сокет для клиента и поток для его обслуживания.

printf("Сервер начал работу....\n");

for (;;Sleep(75))

{

if (Connect=accept(Listen,NULL,NULL))

{

printf("Клиент подключен...\n");

Connections[ClientCount]=Connect;

ClientCount++;

CreateThread(NULL,NULL,(LPTHREAD\_START\_ROUTINE)SendMessageToClient,(LPVOID)(ClientCount-1),NULL,NULL);

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

8

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

}

}

return 1;

Функция SendMessageToClient предназначена для обслуживания очередного подключившегося клиента, отправки ему сообщений, полученных от других пользователей:

void SendMessageToClient(int ID)

{

char\*buffer=new char[1024];

for(;;Sleep(75))

{

memset(buffer,0,sizeof(buffer));

if (recv(Connections[ID],buffer,1024,NULL))

{

printf(buffer);

printf("\n");

for (int i=0; i<=ClientCount; i++)

{

send(Connections[i],buffer,strlen(buffer),NULL);

}

}

}

delete buffer;

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

}

# ПОЯСНЕНИЯ ПО РАБОТЕ ОСНОВНЫХ ФУНКЦИй клиентской части ПРИЛОЖЕНИЯ

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

9

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Принцип работы клиента в целом схож с работой сервера. Проверка лицензионного ключа отличается только самой эталонной строкой для сравнения char str3[]="NSJ="; char str4[]="1,KU"; char str2[]=":PTG"; char str1[]=")<\*7"; , также присутствует защита от отладчика.

if (IsDebuggerPresent()) ExitProcess(0);

BOOL f=false;

CheckRemoteDebuggerPresent(GetCurrentProcess(), &f);

if (f==true) ExitProcess(0);

Полученный хэш введенного пароля посимвольно сравнивается с эталонной строкой, при сравнении идет накапливание в переменную rez. Далее сравнивается значение переменной rez с установленным значением, которое соответствует длине эталонной строки, и если условие не выполняется, то программа клиента закрывается:

if (rez!=16)

{

MyForm::Close();

ExitProcess(0);

}

Также, реализована защита от пустых полей:

if (textBox3->Text=="")

MessageBoxA(NULL,"Введите пароль!","Ошибка",MB\_OK);

if (textBox1->Text =="")

MessageBoxA(NULL,"Введите имя!","Ошибка",MB\_OK);

if ((richTextBox2->Text) == "")

MessageBoxA(NULL,"Введите сообщение!","Ошибка",MB\_OK);

Далее начинается непосредственное создание клиента, создание самого сокета:

WSADATA wsaData;

SOCKET ConnectSocket = INVALID\_SOCKET;

struct addrinfo \*result = NULL, \*ptr = NULL, hints;

int iResult;

Далее заполняем структуру сокета и устанавливаем соединение:

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

String^ ip ="";

ip=textBox2->Text;

char ipchar[16];

for (int i=0; i<sizeof(ipchar); i++)

ipchar[i]=0;

for (int i = 0; i < ip->Length; i++)

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

ipchar[i] = ip[i];

iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);

ZeroMemory(&hints, sizeof(hints));

hints.ai\_family = AF\_UNSPEC;

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

10

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

hints.ai\_socktype = SOCK\_STREAM;

hints.ai\_protocol = IPPROTO\_TCP;

iResult = getaddrinfo(ipchar,port, &hints, &result);

for (ptr = result; ptr != NULL; ptr = ptr->ai\_next) {

ConnectSocket = socket(ptr->ai\_family, ptr->ai\_socktype, ptr->ai\_protocol);

iResult = connect(ConnectSocket, ptr->ai\_addr, (int)ptr->ai\_addrlen);

if (iResult == SOCKET\_ERROR) {

MessageBoxA(NULL,"Ошибка установления соединения! Проверьте ip адрес сервера","Ошибка",MB\_OK);

closesocket(ConnectSocket);

ConnectSocket = INVALID\_SOCKET;

continue;

}

break;

}

freeaddrinfo(result);

richTextBox1->AppendText("Соединение установлено"+"\n");

Для каждого клиента создается свой отдельный поток для приема сообщений:

ThreadStart^ threadDelegate = gcnew ThreadStart(this, &MyForm::RecvMessage);

Thread^ newThread = gcnew Thread(threadDelegate);

newThread->Start();

Сама функция принятия сообщений:

System::Void RecvMessage()

{

for (;; Sleep(300)) {

char buffer[1024];

for (int i = 0; i < sizeof(buffer); i++)

buffer[i] = 0;

recv(ConnectSocket, buffer, sizeof(buffer), 0);

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

11

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

String^ message = "";

for (int i = 0; i < sizeof(buffer); i++)

message += Convert::ToChar(buffer[i]);

richTextBox1->Invoke(gcnew Action<String^>(this, &MyForm::recvmes), message);

}

}

System::Void recvmes(String^ recvm)

{

richTextBox1->AppendText(recvm);

richTextBox1->ScrollToCaret();

}

Отправка сообщений осуществляется при нажатии на кнопку, пустое сообщение отправить нельзя:

String^ message = "";

if ((richTextBox2->Text) != "")

{

message="";

message +=textBox1->Text+": "+richTextBox2->Text;

message += "\n";

char buffer[1024];

for (int i = 0; i < 1024; i++)

buffer[i] = 0;

for (int i = 0; i < message->Length; i++)

buffer[i] = message[i];

send(ConnectSocket, buffer,sizeof(buffer), 0);

richTextBox2->Clear();

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

}

else MessageBoxA(NULL,"Введите сообщение!","Ошибка",MB\_OK);

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Графический дизайн интерфейса клиента реализован с помощью WindowsForm:

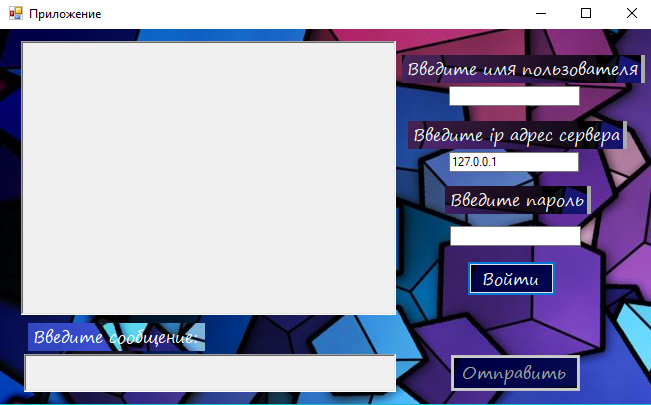


Рисунок 1 – реализация графической части клиента

# Требования к ОПЕРАЦИОННой СИСТЕМе

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

15

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Для выполнения программы необходима операционная система Windows 7 x64/86 или новее.

# Перечень выходных данных

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

16

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Выходными данными для клиентской части приложения являются сообщения, принятые с сервера.

Выходными данными для серверной части приложения являются сообщения от клиентов, отправляемые каждому пользователю.

# АУДИТ БЕЗОПАСНОСТИ

В качестве программы для проведения аудита была взята работа студента группы ИБ-116 Зеленцовой А.А.

Были получены два файла: ConsoleApplication4.exe и ChatCpp.exe.

Программный продукт дизассемблировался, отлаживался и редактировался с помощью программы IDA Pro 7.0.

После запуска сервера мы видим, что он защищён паролем. В начале рассмотрим список функций, используемых в программе. При просмотре списка вызываемых функций (Рисунок 1) можно заметить функцию защиты от отладчика: IsDebuggerPresent, т.е. при попытке отладить программу, она закрывается и выдает ошибку. Таким образом, можно сделать очевидный вывод, что реализована защита от динамического анализа.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

17

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

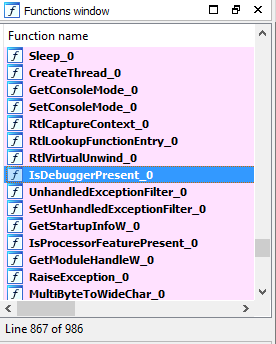


Рисунок 2 – Список функций

Далее я начала проводить статический анализ. Открыв программу в строковом виде, я заметила, что присутствует строка «Enter Password». Логично предположить, что пароль хранится где-то рядом.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Также я обратила внимание на строки «The time» и «Время истекло», из чего можно сделать вывод, что на ввод пароля отводится определённое количество времени.

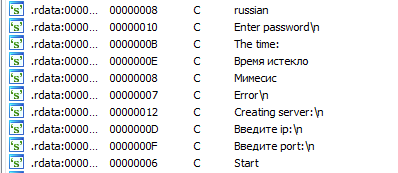


Рисунок 3 – Программа в строковом виде



Рисунок 4 – Строка Enter Password

Я заметила переменную строкового типа со значением «Мимесис» и предположила, что в коде программы находится какая-то строка, с которой в последствии сравнивается введенный пользователем пароль.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

18

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата



Рисунок 5 – Строка Enter Password

Очевидно, что здесь реализована функция, которая хеширует пароль.

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

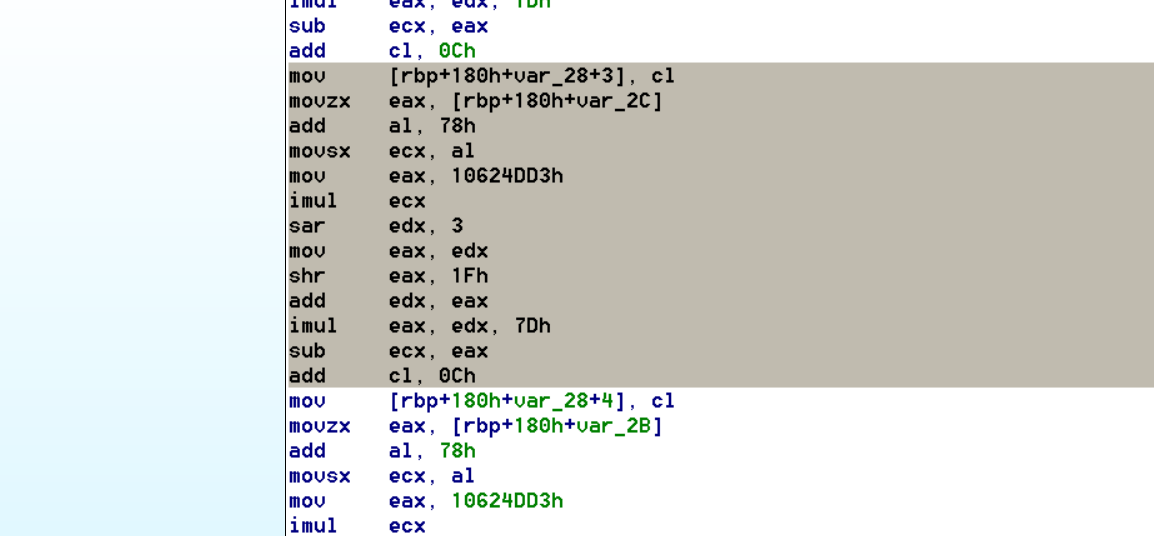


Рисунок 6 – Функция, хэширующая пароль

Анализируя эту функцию, я смогла восстановить формулу, по которой изменяется каждый символ строки.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

17

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

y=(x+120)%125+12

Таким образом, пароль в клиент-серверном приложении реализован следующим образом: пользователь вводит пароль, от каждого символа пароля вычисляется его ASCII-код, далее этот код изменяется с помощью формулы, затем этот ASCII-код преобразуется в новый символ. Именно так происходит изменение строки и превращение ее в зашифрованную, которая впоследствии и сравнивается с эталонной.

Теперь проведем статический анализ клиентской части приложения.

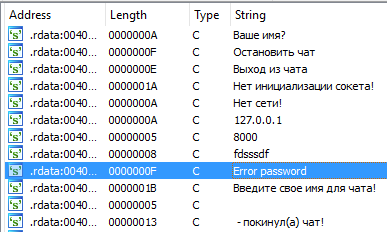
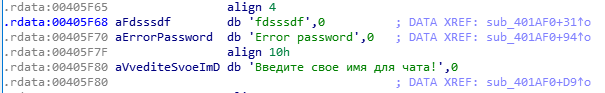


Рисунок 7 – строка Error password

Открыв программу в строковом виде, заметила, что присутствует строка «Enter Password». Перехожу в место кода программы, где упоминается эта строка, замечаю «странную» строку - fdsssdf, состоящую из 7 символов, она очень напоминает эталонную строку в серверной части приложения.



Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

20

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Рисунок 8 – обнаружение эталонной строки

Предполагаю, что динамическая защита в клиенте реализована также, как и на сервере. Теперь необходимо реализовать функцию, обратную хэшированию, чтобы восстановить по найденному хэшу исходный пароль. Данную функцию я реализовала на языке программирования С++.

#include <iostream>

#include <string>

using namespace std;

int main()

{

setlocale(LC\_ALL, "russian");

char hash[] = "fdsssdf";

char pass[8];

char s[3];

for (int i = 0; i < sizeof(hash)-1; i++)

{

int p = char(hash[i]);

int q = 0;

int flag = 0;

while (flag != 1)

{

int m = (char)(q + 120) % 125 + 12;

if (m == p)

{

pass[i] = char(q);

flag = 1;

}

else

q = q + 1;

}

printf("%c",pass[i]);

}

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

system (“pause>nul”);

}

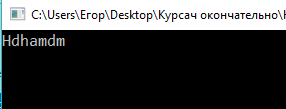


Рисунок 9 – результат выполнения программы подбора пароля для сервера

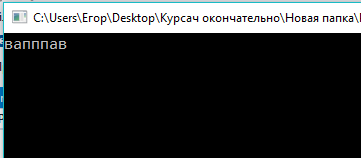


Рисунок 10 – результат выполнения программы подбора пароля для клиента

Теперь попробуем ввести полученные пароли в сервер и клиент. При вводе пароля в сервер, он не отображается, но это никак не повлияло на ввод пароля. Видим, что сервер успешно запустился. Далее проделываем ту же операцию над клиентом и отправляем тестовое сообщение. Результат так же успешен. Аудит безопасности завершен.

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

21

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

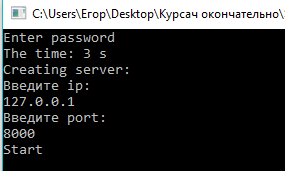


Рисунок 11 – результат работы сервера

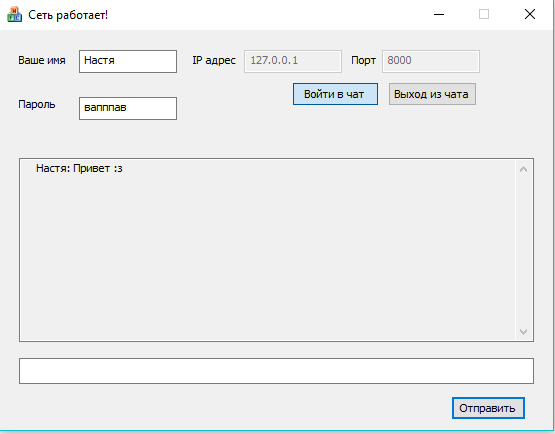


Рисунок 12 – результат работы клиента

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Таблица 1 – Лист регистрации изменений проекта

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

| №п/п | ФИО исполнителя | Примечание | Версия документа | Дата | Подпись |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Инв. № подл.

*33*

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

**ПРИЛОЖЕНИЕ А**

Текст программы на исходном языке.

**Клиентский модуль «Client.exe»:**

**MyForm.h:**

#pragma once

#pragma comment(lib, "Ws2\_32.lib")

#pragma comment(lib, "User32.lib")

#pragma comment(lib, "Mswsock.lib")

#pragma comment(lib, "AdvApi32.lib")

#include <WinSock2.h>

#include <iostream>

#include <WS2tcpip.h>

#include <string>

#include <windows.h>

using namespace System;

using namespace std;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Text;

using namespace System::Collections::Generic;

using namespace System::Drawing;

using namespace System::Threading::Tasks;

using namespace System::Windows::Forms;

using namespace System::Net;

using namespace System::Net::Sockets;

using namespace System::Threading;

namespace Project {

SOCKET ConnectSocket = INVALID\_SOCKET;

char port[] = "5896";

char terp3[]="NSJ=";

int rez=0;

unsigned int Hash37(char str)

{

# 

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

int hash = (int)(str);

return hash;

}

char terp4[]="1,KU";

unsigned int toHash37(int str)

{

char hash = (char)(str);

return hash;

}

unsigned int function(int p)

{

int o=(sin(p)/cos(p)+4\*p\*p)/1000+32-(sin(p)\*cos(p)+4\*cos(p)\*5)\*0;

return o;

}

char terp2[]=":PTG";

public ref class MyForm : public System::Windows::Forms::Form

{

public:

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox3;

public:

private: System::Windows::Forms::Button^ button2;

public:

MyForm(void)

{

InitializeComponent();

}

System::Void RecvMessage() // функция принятия сообщений от клиента

{

for (;; Sleep(300)) {

char buffer[1024];

for (int i = 0; i < sizeof(buffer); i++)

buffer[i] = 0;

recv(ConnectSocket, buffer, sizeof(buffer), 0);

String^ message = "";

for (int i = 0; i < sizeof(buffer); i++)

Лист

26

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

# 

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

message += Convert::ToChar(buffer[i]);

richTextBox1->Invoke(gcnew Action<String^>(this, &MyForm::recvmes), message);

}

}

System::Void recvmes(String^ recvm)

{

richTextBox1->AppendText(recvm);

richTextBox1->ScrollToCaret();

}

protected:

~MyForm()

{

if (components)

{

delete components;

}

}

protected:

private:

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MyForm::typeid));

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->richTextBox2 = (gcnew System::Windows::Forms::RichTextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->button2 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->button3 = (gcnew System::Windows::Forms::Button());

this->label2 = (gcnew System::Windows::Forms::Label());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->SuspendLayout();

//

// richTextBox1

//

this->richTextBox1->Enabled = false;

this->richTextBox1->Location = System::Drawing::Point(21, 12);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->ScrollBars = System::Windows::Forms::RichTextBoxScrollBars::Vertical;

this->richTextBox1->Size = System::Drawing::Size(375, 274);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = L"";

//

// richTextBox2

//

Лист

27

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

this->richTextBox2->Enabled = false;

this->richTextBox2->Location = System::Drawing::Point(24, 325);

this->richTextBox2->Name = L"richTextBox2";

this->richTextBox2->Size = System::Drawing::Size(372, 38);

this->richTextBox2->TabIndex = 2;

this->richTextBox2->Text = L"";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Segoe Print", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->ForeColor = System::Drawing::SystemColors::ButtonHighlight;

this->label1->Image = (cli::safe\_cast<System::Drawing::Image^ >(resources->GetObject(L"label1.Image")));

this->label1->Location = System::Drawing::Point(28, 294);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(177, 28);

this->label1->TabIndex = 3;

this->label1->Text = L"Введите сообщение:";

//

// button2

//

this->button2->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^ >(resources->GetObject(L"button2.BackgroundImage")));

this->button2->Font = (gcnew System::Drawing::Font(L"Segoe Print", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button2->ForeColor = System::Drawing::SystemColors::ButtonFace;

this->button2->Location = System::Drawing::Point(467, 232);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(89, 35);

this->button2->TabIndex = 5;

this->button2->Text = L"Войти";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(449, 57);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(131, 20);

this->textBox1->TabIndex = 6;

//

// button3

//

this->button3->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^ >(resources->GetObject(L"button3.BackgroundImage")));

this->button3->Enabled = false;

this->button3->Font = (gcnew System::Drawing::Font(L"Segoe Print", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

Лист

28

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

this->label4->AutoSize = true;

this->label4->BackColor = System::Drawing::SystemColors::AppWorkspace;

this->label4->Font = (gcnew System::Drawing::Font(L"Segoe Print", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label4->ForeColor = System::Drawing::SystemColors::ButtonFace;

this->label4->Image = (cli::safe\_cast<System::Drawing::Image^ >(resources->GetObject(L"label4.Image")));

this->label4->ImageAlign = System::Drawing::ContentAlignment::MiddleRight;

this->label4->Location = System::Drawing::Point(445, 157);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(146, 28);

this->label4->TabIndex = 11;

this->label4->Text = L"Введите пароль";

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(450, 197);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(131, 20);

this->textBox3->TabIndex = 12;

//

// MyForm

//

this->AcceptButton = this->button3;

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^ >(resources->GetObject(L"$this.BackgroundImage")));

this->ClientSize = System::Drawing::Size(655, 375);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->label2);

this->Controls->Add(this->button3);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button2);

this->Controls->Add(this->label1);

this->Controls->Add(this->richTextBox2);

this->Controls->Add(this->richTextBox1);

this->Name = L"MyForm";

this->Text = L"Приложение";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

char terp1[]=")<\*7";

if (textBox3->Text=="")

setlocale(LC\_ALL, "russian");

Лист

29

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

MessageBoxA(NULL,"Введите пароль!","Ошибка",MB\_OK);

String^ s1=textBox3->Text;

for (int i=0; i<s1->Length; i++)

{

int p=Hash37(s1[i]);

int o=function(p);

char w1=toHash37(o);

if (i<=3)

{

int j=i%3;

if (terp3[j]==w1 || terp3[3]==w1)

rez=rez+1;

}

if(i>3 && i<=7)

{

int j=i%4;;

if (terp4[j]==w1 || terp4[3]==w1)

rez=rez+1;

}

if(i>7 && i<=11)

{

int j=i%8;

if (terp2[j]==w1 || terp2[3]==w1)

rez=rez+1;

}

if(i>11 && i<=15 || terp1[3]==w1)

{

int j=i%12;

if (terp1[j]==w1)

rez=rez+1;

}

}

if (rez!=16)

{

MyForm::Close();

ExitProcess(0);

}

textBox3->Clear();

textBox3->Enabled=false;

WSADATA wsaData;

struct addrinfo \*result = NULL, \*ptr = NULL, hints;

int iResult;

String^ ip ="";

ip=textBox2->Text;

char ipchar[16];

for (int i=0; i<sizeof(ipchar); i++)

ipchar[i]=0;

for (int i = 0; i < ip->Length; i++)

ipchar[i] = ip[i];

Лист

30

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

if (textBox1->Text =="")

{

MessageBoxA(NULL,"Введите имя!","Ошибка",MB\_OK);

}

else

{

iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);

ZeroMemory(&hints, sizeof(hints));

hints.ai\_family = AF\_UNSPEC;

hints.ai\_socktype = SOCK\_STREAM;

hints.ai\_protocol = IPPROTO\_TCP;

loop1:

iResult = getaddrinfo(ipchar,port, &hints, &result);

for (ptr = result; ptr != NULL; ptr = ptr->ai\_next) {

ConnectSocket = socket(ptr->ai\_family, ptr->ai\_socktype, ptr->ai\_protocol);

iResult = connect(ConnectSocket, ptr->ai\_addr, (int)ptr->ai\_addrlen);

if (iResult == SOCKET\_ERROR) {

MessageBoxA(NULL,"Ошибка установления соединения! Проверьте ip адрес сервера","Ошибка",MB\_OK);

goto loop1;

closesocket(ConnectSocket);

ConnectSocket = INVALID\_SOCKET;

continue;

}

break;

}

freeaddrinfo(result);

richTextBox1->AppendText("Соединение установлено"+"\n");

textBox1->Enabled=false;

textBox2->Enabled=false;

button2->Enabled=false;

button3->Enabled=true;

richTextBox2->Enabled=true;

ThreadStart^ threadDelegate = gcnew ThreadStart(this, &MyForm::RecvMessage);

Thread^ newThread = gcnew Thread(threadDelegate);

newThread->Start();

}

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) //кнопка "отправить"

{

String^ message = "";

if ((richTextBox2->Text) != "") {

message="";

message +=textBox1->Text+": "+richTextBox2->Text;

message += "\n";

Лист

31

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

char buffer[1024];

for (int i = 0; i < 1024; i++) buffer[i] = 0;

for (int i = 0; i < message->Length; i++) {buffer[i] = message[i];}

send(ConnectSocket, buffer,sizeof(buffer), 0);

richTextBox2->Clear();

}

else MessageBoxA(NULL,"Введите сообщение!","Ошибка",MB\_OK);

}

};

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**MyForm.cpp:**

#include "MyForm.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThread]

void main (array<String^>^ args)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Project::MyForm form;

Application::Run(%form);

if (IsDebuggerPresent()) ExitProcess(0);

BOOL f=false;

CheckRemoteDebuggerPresent(GetCurrentProcess(), &f);

if (f==true) ExitProcess(0);

}

Лист

32

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

**Серверный модуль «Server.exe»:**

**main.cpp:**

#pragma comment(lib, "Ws2\_32.lib")

#include <WinSock2.h>

#include <WS2tcpip.h>

#include <locale.h>

#include <stdio.h>

#include <iostream>

#include <ostream>

#include <conio.h>

#include <math.h>

#include <windows.h>

#include <cstring>

#include <chrono>

#include <string.h>

SOCKET Connect;

SOCKET \* Connections;

SOCKET Listen;

int ClientCount=0;

char s1[15];

char str3[]="MW>5";

int rez=0;

unsigned int Hash37(char str)

{

int hash = (int)(str);

return hash;

}

char str2[]=")@1\*";

unsigned int toHash37(int str)

{

char hash = (char)(str);

return hash;

}

unsigned int function(int p)

{

int o=(sin(p)/cos(p)+4\*p\*p)/1000+32-(sin(p)\*cos(p)+4\*cos(p)\*5)\*0;

return o;

}

char str4[]=":SY,";

void SendMessageToClient(int ID)

{

char\*buffer=new char[1024];

for(;;Sleep(75))

{

memset(buffer,0,sizeof(buffer));

if (recv(Connections[ID],buffer,1024,NULL))

{

Лист

33

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

printf(buffer);

printf("\n");

for (int i=0; i<=ClientCount; i++)

{

send(Connections[i],buffer,strlen(buffer),NULL);

}

}

}

delete buffer;

}

int main()

{

auto begin = std::chrono::steady\_clock::now(); //начало таймера

char str1[]="+H<K";

setlocale(LC\_ALL, "russian");

printf("Введите пароль: ");

scanf("%s1", &s1);

auto end = std::chrono::steady\_clock::now();

auto elapsed\_ms = std::chrono::duration\_cast<std::chrono::milliseconds>(end - begin);

std::cout << "The time: " << elapsed\_ms.count()/1000 << " s\n";

if (elapsed\_ms.count()/1000 >= 30)

{

std::cout << "Время истекло";

return 0;

}

for (int i=0; i<=sizeof(s1); i++)

{

int p=Hash37(s1[i]);

int o=function(p);

char w1=toHash37(o);

if (i<=3)

{

int j=i%3;

if (str3[j]==w1 || str3[3]==w1)

rez=rez+1;

}

if(i>3 && i<=7)

{

int j=i%4;;

if (str4[j]==w1 || str4[3]==w1)

rez=rez+1;

}

if(i>7 && i<=11)

{

int j=i%8;

if (str2[j]==w1 || str2[3]==w1)

rez=rez+1;

}

if(i>11 && i<=15 || str1[3]==w1)

{

Лист

34

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

int j=i%12;

if (str1[j]==w1)

rez=rez+1;

}

}

if (rez!=16)

ExitProcess(0);

WSAData data;

WORD version=MAKEWORD(2,2);

int res = WSAStartup(version, &data);

if (res!=0)

{

return 0;

}

struct addrinfo hints;

struct addrinfo \* result;

Connections = (SOCKET\*)calloc(64,sizeof(SOCKET));

ZeroMemory(&hints,sizeof(hints));

hints.ai\_family=AF\_INET;

hints.ai\_flags=AI\_PASSIVE;

hints.ai\_socktype=SOCK\_STREAM;

hints.ai\_protocol=IPPROTO\_TCP;

getaddrinfo(NULL,"5896",&hints,&result);

Listen=socket(result->ai\_family,result->ai\_socktype,result->ai\_protocol);

bind(Listen,result->ai\_addr, result->ai\_addrlen);

listen(Listen,SOMAXCONN);

freeaddrinfo(result);

printf("\n");

printf("Сервер начал работу....\n");

for (;;Sleep(75))

{

if (Connect=accept(Listen,NULL,NULL))

{

printf("Клиент подключен...\n");

Connections[ClientCount]=Connect;

ClientCount++;

CreateThread(NULL,NULL,(LPTHREAD\_START\_ROUTINE)SendMessageToClient,(LPVOID)(ClientCount-1),NULL,NULL);

}

}

return 1;

}

Лист

35

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата

Инв. № подп

Подп. и дата

Взам. инв. №

Инв. № дубл.

Подп. и дата

Лист

14

XXX-XXX

Лит

№ докум.

Изм.

Подп.

Дата