南開大學

恶意代码分析与防治技术课程实验报告

实验一：基本静态分析技术实战

学　院　密码与网络空间安全学院

专　业　信息安全

学　号　2411963

姓　名　孟子玉

班　级　0372

摘要

本实验报告围绕恶意代码的静态分析展开。实验原理部分介绍了静态分析的基本思路，判断样本是否加壳的六项判据，本次实验用到的静态分析工具（file、strings、PEiD、PEview、Dependency Walker、Resource Hacker、UPX 和 OllyICE），两个在线分析平台（VirusTotal 与国家计算机病毒协同分析平台），以及批量检测程序的实现原理。实验过程分三部分：第一部分完成教材 Lab01-01 至 Lab01-04 的分析，涉及查壳脱壳、字符串与导入表解读，并从资源段中提取内嵌的可执行文件；第二部分在 Windows 10 虚拟机中分析五个真实恶意代码教学样本，先做本地静态分析，再用 Microsoft Defender、VirusTotal 和国家计算机病毒协同分析平台完成检测；第三部分调用平台接口编写批量检测程序，对 90 个样本和 100 条哈希值完成自动查询、汇总与报告生成。最后总结了实验结论与心得体会。

关键字：恶意代码分析、静态分析、PE 结构、加壳与脱壳、VirusTotal

目录

一、实验目的 3
二、实验原理 3
(1) 判断是否加壳的判据 3
(2) 静态分析工具 4
(3) 在线分析平台 4
(4) 批量检测的原理 5
三、实验过程 5
第一部分：完成教材上 Lab01-01 至 Lab01-04 的题目 5
(1) Lab01-01.exe 与 Lab01-01.dll 5
(2) Lab01-02.exe 13
(3) Lab01-03.exe 18
(4) Lab01-04.exe 21
第二部分：在 Windows 10 虚拟机中分析 5 个真实恶意代码教学样本 28
(1) WannaCrypt0r 28
(2) hrtg 37
(3) BUG32 44
(4) yhzll 52
(5) BossDaMajor 60
第三部分：使用 VirusTotal 提供的 API 开发批量检测程序并形成检测报告 67
(1) 批量检测程序的开发 67
(2) 检测过程 69
(3) 批量检测报告 72
四、实验结论及心得体会 75

一、实验目的

(1) 掌握恶意代码静态分析的基本流程和常用工具。

(2) 学会借助在线分析平台对样本快速定性，读懂检出率、威胁命名、PE 结构信息和行为报告。

(3) 把上述方法应用到两批样本上——教材的 Lab01-01 至 Lab01-04，以及五个在虚拟机中分析的真实恶意代码教学样本，练习从分析结果中提炼可用于排查受感染主机的主机迹证与网络迹证。

(4) 能够调用平台接口编写批量检测程序，对上百个样本自动完成查询、汇总与报告生成。

二、实验原理

恶意代码分析按是否运行样本分为静态分析和动态分析两类。本次实验以前者为主：不执行样本，只从文件本身的字节出发，读取 PE 结构和字符串等静态信息，据此推断样本是否加壳、用什么工具链编译、具备哪些能力，并从中提取可用于排查受感染主机的迹证。动态分析只在静态手段无法求解时才作为补充，例如 FSG 加壳的样本没有可用的自动脱壳工具，只能借助调试器手工脱壳。

(1) 判断是否加壳的判据。加壳会给样本留下若干可观察的特征，本次实验主要看六项。

一看节名。未加壳样本的节名通常是链接器给的默认名，即.text、.rdata、.data、.rsrc；加壳后往往变成 UPX0、UPX1 这类壳自己的命名，或者被破坏成 t、ta、a 这样的残片。

二看节的原始大小与虚拟大小是否相称。壳的解压代码要在运行时把原始内容写回内存，所以会出现某个节在磁盘上几乎不占空间、虚拟大小却很大的情况。

三看节熵。压缩或加密过的数据接近随机分布，熵值一般在 7 以上，正常代码和数据的熵值则明显更低。

四看入口点落在哪个节。入口点落在 UPX1 这类解压节而不是代码段，说明程序是从解压代码开始执行的。

五看导入表。加壳工具会把原始导入表压缩进壳里，只留下 LoadLibraryA 和 GetProcAddress 供解压代码在运行时动态解析真实 API，所以导入表异常”

空” 本身就是加壳的旁证。

六看文件体积和 PE 头的可解析性。加壳后的样本体积常常异常小，有些壳还会破坏 PE 头，使入口点、节数这些字段读不出来。

以上都是间接判据，还要借助专门的查壳工具做交叉验证。PEiD 和 DetectItEasy 都维护着壳与编译器的特征库，签名框给出的是壳名还是编译器名可以直接作为依据；两者都只报编译器、不报 Packer 时，基本可以排除加壳。

(2) 静态分析工具。本次实验用到的工具及其作用如下。

file.exe（GnuWin32）读取文件头判断真实类型，用来确认样本是 PE32 可执行文件。

strings（GnuWin32）提取文件中连续的可打印字符，是获取硬编码的 URL、IP 地址、文件路径、互斥体名和服务名最直接的手段。成品样本的字符串往往集中在文件首尾，中间大片是二进制数据。

PEiD 用特征库匹配壳与编译器签名，是查壳环节的第一道判据。它的特征库较老，报 Nothing found 时不能直接认定加壳，要结合节名和导入表一起判断。

PEview 按 PE 规范逐层展开 IMAGE_FILE_HEADER、DOS 头、节表、导入表和资源树，本次用它读取编译时间戳和各节的大小与权限。

Dependency Walker 解析导入表与导出表，列出样本静态链接了哪些 DLL、各调用了哪些函数，是推断样本能力最可靠的依据之一。

Resource Hacker 用来浏览和提取 PE 文件里的资源。本次用它查看资源树的构成，并从 Lab01-04.exe 和 3-BUG32.exe 中把内嵌的可执行文件按二进制方式导出。

UPX 是 UPX 壳配套的脱壳工具，`upx-d` 一条命令即可完成自动脱壳；脱壳后用节名、PEiD 签名和导入表三项验证是否成功。

OllyICE 是调试器，用于没有自动脱壳工具的壳。这类壳只能从入口点开始单步跟踪解压代码，等控制权跳转到原始入口点后把进程内存 dump 出来并重建导入表。这属于动态调试，超出基本静态分析的范围，本次实验只做说明，没有实际执行。

(3) 在线分析平台。本次实验用到两个平台，作用互补。

VirusTotal 汇集了七十家左右的引擎，按哈希查询或上传样本后一次就能拿到多引擎检出结果。除检出比之外，它还能给出聚合后的威胁命名与家族标签、PE 头与节表、Magic 字段对壳的识别结果、完整的导入表和资源类型统计；由沙箱执行产生的行为报告则提供了互斥体、创建的服务、写入的注册表项和网

络请求，这些正是排查受感染主机时最有用的主机迹证与网络迹证。它给出的 SSDeep 与 TLSH 属于模糊哈希，样本被轻微改动后仍然相似，适合做家族聚类。需要留意的是行为报告来自动态执行，与静态分析的结论要分开看待。

国家计算机病毒协同分析平台是国产的多引擎检测平台，接口与报告结构都和 VirusTotal 不同，两者可以互为对照。它的检出标签偏向中文描述的行为类别，例如勒索行为、挖矿行为、泄密行为；静态信息页提供文件格式、编译器、节表、导入表和资源节信息，其中的”子文件检出威胁”能提示样本内嵌了有问题的子文件。两个平台的检出率差异本身也有参考价值：引擎数量、覆盖的地域和厂商都不同，未检出的引擎名单能看出各家特征库的侧重。

(4) 批量检测的原理。第三部分把前两部分的手工流程自动化。程序用 Python 3 编写，只依赖 `requests` 一个第三方库，通过两个平台的开放接口批量查询样本。

VirusTotal 用官方 v3 接口 `GET https://www.virustotal.com/api/v3/files/{hash}`，鉴权靠请求头 `x-apikey`；协同平台则把 `apikey` 放在请求体里。公开接口都有限流，VirusTotal 限制每分钟 4 次、每天 500 次，所以程序把两次请求的间隔控制在 16 秒，触发 429 时读取响应头里的 `Retry-After` 自动等待，并按指数退避重试。

样本的哈希由程序自己计算：对样本目录递归遍历，分块读取每个文件并算出 SHA256，再按内容去重，改过名的副本因此会自动合并成一条记录。已经下载过报告的哈希会被跳过，中断后重新运行可以接着查没查完的部分，不会重复消耗配额。

三、实验过程

第一部分：完成教材上 Lab01-01 至 Lab01-04 的题目（所有实验均在 win10 虚拟机中进行）

(1) 使用 Lab01-01.exe 和 Lab01-01.dll 文件，使用本章描述的工具和技术来获取关于这些文件的信息。回答以下问题：

1. 将文件上传至 http://www.VirusTotal.com/进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗？

两个文件都被大量引擎检出，匹配到了已有的反病毒软件特征。

Lab01-01.exe 有 56/71 家引擎报恶意。VT 聚合出的流行威胁名是`trojan.ulise/aenjaris`，威胁类别为 trojan / downloader / worm。

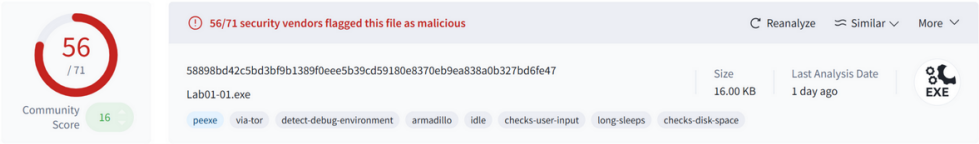


图 1: Lab01-01.exe 的 VirusTotal 页

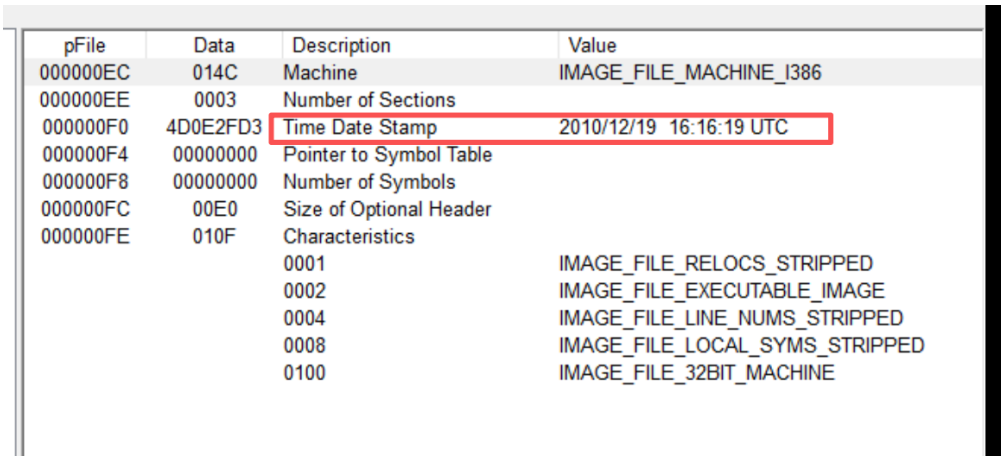
Lab01-01.dll 有 39/71 家引擎报恶意。VT 聚合出的流行威胁名是`trojan.skeeyah/genericrxfo`。

图 2: Lab01-01.dll 的 VirusTotal 页

可以看到 DLL 文件的检出率明显更低。卡巴斯基检出了 exe 却漏掉了 dll，BitDefender、Fortinet、GData、DrWeb、Malwarebytes 也都没有报 dll。

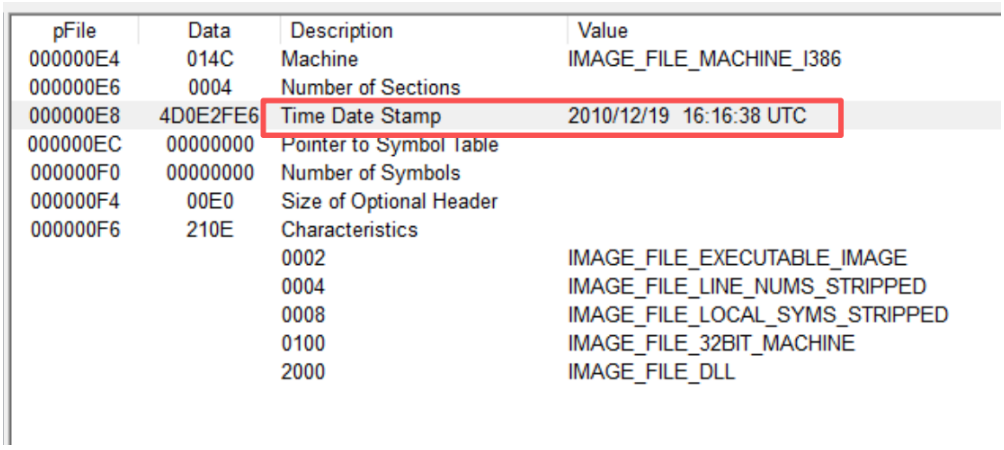
2. 这些文件是什么时候编译的？

在 VT 的 DETAILS 页的 History 段可知两个文件都编译于 2010 年 12 月 19 日，时间相差 19 秒。这个时间来自 PE 头 IMAGE_FILE_HEADER 结构里的 Time Date Stamp 字段。



pFile	Data	Description	Value
000000EC	014C	Machine	IMAGE_FILE_MACHINE_I386
000000EE	0003	Number of Sections	
000000F0	4D0E2FD3	Time Date Stamp	2010/12/19 16:16:19 UTC
000000F4	00000000	Pointer to Symbol Table	
000000F8	00000000	Number of Symbols	
000000FC	00E0	Size of Optional Header	
000000FE	010F	Characteristics	
		0001	IMAGE_FILE_RELOCS_STRIPPED
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0100	IMAGE_FILE_32BIT_MACHINE

图 3: Lab01-01.exe 的 PEview IMAGE_FILE_HEADER 截图



pFile	Data	Description	Value
000000E4	014C	Machine	IMAGE_FILE_MACHINE_I386
000000E6	0004	Number of Sections	
000000E8	4D0E2FE6	Time Date Stamp	2010/12/19 16:16:38 UTC
000000EC	00000000	Pointer to Symbol Table	
000000F0	00000000	Number of Symbols	
000000F4	00E0	Size of Optional Header	
000000F6	210E	Characteristics	
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0100	IMAGE_FILE_32BIT_MACHINE
		2000	IMAGE_FILE_DLL

图 4: Lab01-01.dll 的 PEview IMAGE_FILE_HEADER 截图

3. 这两个文件中是否存在迹象说明它们被加壳或混淆了？如果是，这些迹象在哪里？

这两个文件没有加壳，也没有混淆。

第一，节名是编译器的默认名。两个文件用的是.text、.rdata、.data（DLL 多一个.reloc），都是链接器的默认命名。加壳后的文件节名通常会变成 UPX0、UPX1、.aspack 等，或者干脆是乱码。

第二，用 PEiD 检测没有识别出任何壳。两个文件匹配到的都是编译器签名而不是壳签名：exe 显示为 Microsoft Visual C++，dll 显示为 Microsoft Visual C++ v6.0 DLL。

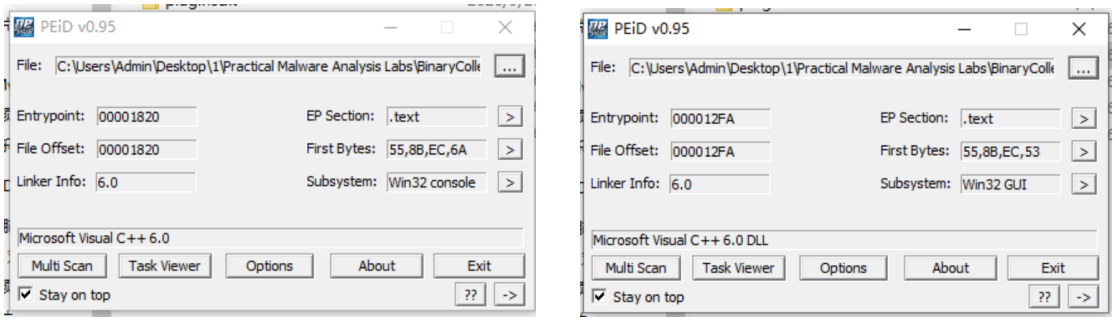


图 5: PEiD 分别加载 Lab01-01.exe 和 Lab01-01.dll 的检测结果

在 VT 上可以对这一结论做交叉验证。VT 的 PEiD 字段给出的签名与上面一致，同一份报告里的 DetectItEasy 也只列出了 Compiler、Linker、Tool 三类条目，没有任何 Packer 或 Protector 条目，说明独立的壳检测工具同样认为这两个文件没有加壳。

Basic properties

MD5	bb7425b82141a1c0f7d60e5106676bb1
SHA-1	9dce39ac1bd36d877fdb0025ee88fdaff0627cdb
SHA-256	58898bd42c5bd3bf9b1389f0eee5b39cd59180e8370eb9ea838a0b327bd6fe47
Vhash	014036151d1bza0f=z
Authentihash	094eed7cfc959fd9ba704d5fe0b965b7bbb6ca09d302870935dc0508d940ba2c
Imphash	2b5f75aa75c57ed7c68f7be490d63605
Rich PE header hash	6a52cc2e068dfb8f2b4715556fd89a66
SSDEEP	96:1t6Y5CuDzp17S5eVlV2cFL+31znx9+NNoyn:v6Y7117S5ercZ+FznxcNNoyn
TLSH	T17C72B44376E51CB1EF2811B6429293FC927DE0604766F2EE78731A46D432893793CADB
File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32 executable (console) Intel 80386, for MS Windows
TrID	Microsoft Visual C++ compiled executable (generic) (33.3%) | Win32 Dynamic Link Library (generic) (13.2%) | Win64 Executable (generic) (13.1%) | W
DetectItEasy	PE32 | Compiler: EP:Microsoft Visual C/C++ (6.0 (1720-9782)) [EXE32] | Compiler: Microsoft Visual C/C++ (12.00.8168) [C++] | Linker: Microsoft Linker
Magika	PEBIN
File size	16.00 KB (16384 bytes)
PEiD packer	Microsoft Visual C++

图 6: VirusTotal 中 Lab01-01.exe 的 PEiD 签名与 DetectItEasy 识别结果

Basic properties

MD5	290934c61de9176ad682ffdd65f0a669
SHA-1	a4b35de71ca20fe776dc72d12fb2886736f43c22
SHA-256	f50e42c8dfaab649bde0398867e930b86c2a599e8db83b8260393082268f2dba
Vhash	115046151d151bz505tza7z
Authentihash	f2186e6806b44e6d2106a3e528010511a5fb450448be8cb026dbc16894e080b5
Imphash	850a8b8b585d7874d0431e8e45d74606
Rich PE header hash	0a70ae41bf95138a8e844adbbcb01dea
SSDEEP	48:aVD3M6gl4PXzLceRoCpbDIa1XBtz2Wuw009WuwazHSz:MrMAzLctgt5z2Wz0sWzSHS
TLSH	T10AF32EB39BE08BFFD5280B37029B49B33437A560039405AB5762C83D2F9562AD56DE1A
File type	Win32 DLL executable windows win32 pe pedll
Magic	PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
TrID	Win32 Dynamic Link Library (generic) (21.4%) | Win64 Executable (generic) (21.2%) | Win16 NE executable (generic) (16.3%) | Win32 Executable (ger
DetectItEasy	PE32 | Compiler: EP:Microsoft Visual C/C++ (6.0 (1720-8966)) [DLL32] | Compiler: Microsoft Visual C/C++ (12.00.8168) [C++] | Linker: Microsoft Linker
Magika	PEBIN
File size	160.00 KB (163840 bytes)
PEiD packer	Microsoft Visual C++ v6.0 DLL

图 7: VirusTotal 中 Lab01-01.dll 的 PEiD 签名与 DetectItEasy 识别结果

第三，节的数量和大小合理。exe 有 3 个节，dll 有 4 个节，各节的 Virtual Size 与 Raw Size 基本相等，没有加壳文件常见的”虚拟大小远大于原始大小”的解压空间。

第四，各节熵值很低。压缩或加密数据的熵一般在 7 以上，这里最高的才 4.45，DLL 的.rdata 熵只有 0.03，说明它整段是零填充而不是加密载荷。下面两张节表截图里的 Entropy 列可以直接核对这些数值。

Portable Executable Info

Compiler Products

[C] VS98 (6.0) build 8168 count=11
[LNK] VS98 (6.0) imp/exp build 8168 count=2
[---] Unmarked objects count=27
[C++] VS98 (6.0) build 8168 count=2
id: 0xc, version: 7291 count=1
id: 0xe, version: 7299 count=1
id: 0x13, version: 8034 count=3

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2010-12-19 16:16:19 UTC
Entry Point	6176
Contained Sections	3

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	2416	4096	4.45	7e39ebe7cdeda4c636d513a0fe140ff4	229395.12
.rdata	8192	690	4096	1.13	2de0f3a50219cb3d0dc891c4fbf6f02a	823067.88
.data	12288	252	4096	0.44	f5e2ba1465f131f57b0629e96bbe107e	963729.62

Imports

+ KERNEL32.dll
+ MSVCRT.dll

图 8: VirusTotal 中 Lab01-01.exe 的节表与熵值

Portable Executable Info

Compiler Products

[C] VS98 (6.0) build 8168 count=4
[---] Unmarked objects count=22
[C++] VS98 (6.0) build 8168 count=2
[LNK] VS98 (6.0) imp/exp build 8168 count=3
id: 0xe, version: 7299 count=1
id: 0x13, version: 8034 count=5

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2010-12-19 16:16:38 UTC
Entry Point	4858
Contained Sections	4

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	926	4096	1.9	65d3ddf9778db8d01e57b5825fbd93ad	678274.25
.rdata	8192	147398	147456	0.03	530532a38a38ea1219e691b8f16d10e9	37481140
.data	155648	108	4096	0.11	0211086333be22ae2620b568fde46fe3	1026641.75
.reloc	159744	516	4096	0.26	a082f3572d17cd40272b3bcfd96b7b2d	997945.62

Imports

+ KERNEL32.dll

图 9: VirusTotal 中 Lab01-01.dll 的节表与熵值

综合以上来看，可得出没有加壳，也没有混淆的结论。

4. 是否有导入函数显示出了这个恶意代码是做什么的？如果是，是哪些导入函数？

Lab01-01.exe 只导入 KERNEL32.dll 和 MSVCRT.dll，其中关键的函数是 FindFirstFileA、FindNextFileA、FindClose（遍历文件系统）、CopyFileA（拷贝文件）、CreateFileA、CreateFileMappingA、MapViewOfFile、UnmapViewOfFile（打开文件并映射进内存，用于改写内容）以及 IsBadReadPtr（解析结构时校验

指针）。把这些函数和字符串里的 C:*、.exe、kerne132.dll、Lab01-01.dll 结合起来看，可以判断它会遍历 C 盘查找可执行文件并改写它们。

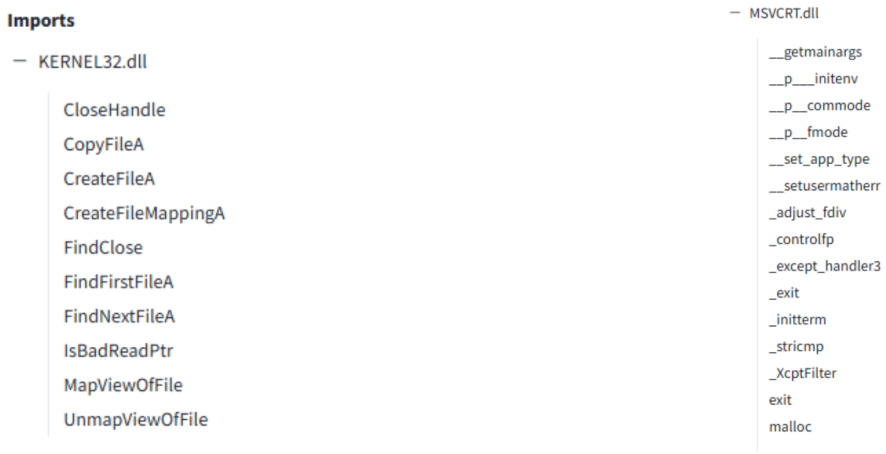


图 10: VirusTotal 中 Lab01-01.exe 的导入函数

Lab01-01.dll 导入 KERNEL32.dll、WS2_32.dll 和 MSVCRT.dll。WS2_32 是 Windows 套接字库，说明它具备网络通信能力，而且这个 DLL 的函数在导入表里没有名字，是按序号导入的。此外 CreateProcessA 说明它可以创建进程执行命令，CreateMutexA 和 OpenMutexA 用于创建或打开互斥体，Sleep 用于延时。

综合来看，DLL 是一个具备联网和起进程能力的后门组件，而 exe 是一个会改写可执行文件的安装或感染组件。

图 11: VirusTotal 中 Lab01-01.dll 的导入函数

5. 是否有任何其他文件或基于主机的迹象，让你可以在受感染系统上查找？

用 strings 可以从两个文件里提取出下列字符串，Lab01-01.exe 中的字符串

是：

```
C:\*
C:\windows\system32\kerne132.dll
kerne132.dll
Lab01-01.dll
C:\Windows\System32\Kernel32.dll
WARNING_THIS_WILL_DESTROY_YOUR_MACHINE
```

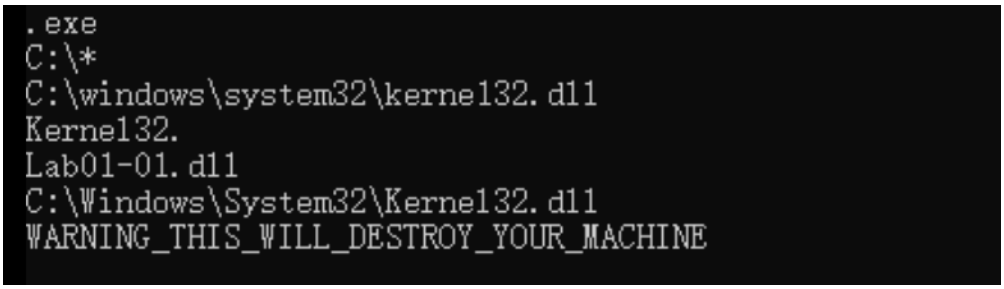


图 12: 对 Lab01-01.exe 执行 strings 的命令行输出

Lab01-01.dll 中的字符串是：

```
exec
sleep
hello
127.26.152.13
SADFHUHF
```

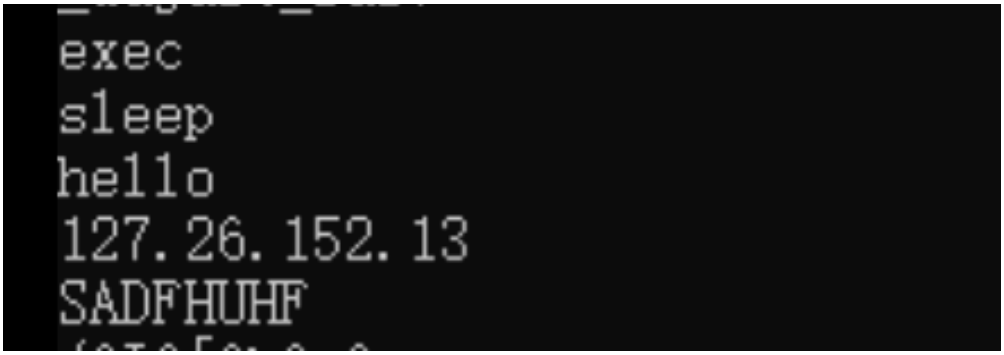


图 13: 对 Lab01-01.dll 执行 strings 的命令行输出

据此可以在受感染系统上查找两条主机迹证。第一条是文件%WINDIR%\System32\kerne132.dll，注意拼写里用的是数字 1 而不是字母 l，它是伪装成系统库 kernel32.dll 的恶意 DLL。第二条是互斥体 SADFHUHF，DLL 会创建这个

互斥体，可以用 Process Explorer 的 Find Handle 功能搜索它，定位到持有它的进程。

6. 是否有基于网络的迹象，可以用来发现受感染机器上的这个恶意代码？

有，在 Lab01-01.dll 里存在硬编码的 IP 地址 127.26.152.13。它是 DLL .data 段里的明文字符串，与 DLL 导入的 WS2_32 库相互印证，可以判断这是样本回连使用的 C2 地址。VT 的 Behavior 页在 DLL 的 IP Traffic 里也列出了同一个地址，与静态分析结果一致。

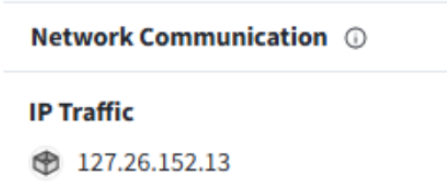


图 14: VirusTotal 中 Lab01-01.dll 的 Behavior 页 IP Traffic

需要指出的是，Lab01-01.exe 里没有任何网络相关的字符串或导入函数，网络能力完全集中在 DLL 上，说明两个文件的分工是明确的。

7. 你猜这些文件的目的是什么？

这两个文件构成一套组合：一个文件感染器加一个后门。

Lab01-01.exe 负责投放和持久化。它把同目录下的 Lab01-01.dll 复制成 C:\Windows\System32\kerne132.dll，用数字 1 冒充字母 l 伪装成系统库，然后递归遍历 C:\下的所有.exe 文件，改写它们的 PE 导入表，把对 kernel32.dll 的引用替换成 kerne132.dll。这样就实现了隐蔽的持久化——系统里几乎所有程序启动时都会加载这个恶意 DLL，而文件名与系统库相似，不容易引起用户怀疑。exe 的 FindFirstFileA、FindNextFileA、CopyFileA 对应遍历与拷贝这一步，CreateFileMappingA 和 MapViewOfFile 对应改写文件这一步，字符串里的 kerne132.dll 和 Lab01-01.dll 则印证了复制改名的动作。

Lab01-01.dll 是真正的载荷，一个后门或远控客户端。它通过 WS2_32 回连硬编码的 127.26.152.13，通过 CreateProcessA 执行命令，用互斥体 SADFHUHF 做标记或单实例控制。字符串 exec、sleep、hello 推测是它与 C2 通信时的命令关键字，分别对应用户执行命令、休眠和上线握手。

综合来看，exe 的作用是把后门铺到系统各处并保证它被反复加载，dll 的作用是接受远程控制，属于“感染器投放后门”这一类型的组合。

(2) 分析 Lab01-02.exe 文件，回答以下问题：

1. 将 Lab01-02.exe 文件上传至 http://www.VirusTotal.com/进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗？

匹配到了已有的反病毒软件特征。Lab01-02.exe 被 58/71 家引擎报恶意，VT 聚合出的流行威胁名是`trojan.ulise/trojanclicker`，威胁类别为 trojan/downloader。

检出名称中普遍带有 Clicker、StartPage 字样，说明多数厂商把它归为点击器类而非单纯的下载器；未检出的引擎里包括卡巴斯基、Panda 和 QuickHeal。

图 15: VirusTotal 对 Lab01-02.exe 的检出结果，58/71 家引擎报恶意

2. 是否有这个文件被加壳或混淆的任何迹象？如果是这样，这些迹象是什么？如果该文件被加壳，请进行脱壳，如果可能的话。

有，这个文件被 UPX 加壳了。判断迹象有六条。

第一，节名异常。三个节分别叫 UPX0、UPX1、UPX2，这是 UPX 的固定命名。

第二，UPX0 节的 Raw Size 为 0。它在磁盘上不占空间，虚拟大小却有 16384 字节，说明这段空间要留到运行时由解压代码写入，是加壳的典型特征。

第三，UPX1 节的熵值为 7.07。接近随机分布，说明里面装的是压缩数据加解压代码，而不是正常代码。

第四，入口点为 21520，正好落在 UPX1 的虚拟地址区间（20480 到 24576）内。这说明程序是从解压代码开始执行的，而不是从原始代码段。

第五，文件体积只有 3072 字节（3KB），对这类程序来说异常小。

第六，多个检测器都识别出了壳。VT 的 Magic 字段写的是 UPX compressed，

DetectItEasy 报 Packer: UPX (3.04) [NRV,best]，PEiD 报 UPX v0.89.6 - v1.02 / v1.05 -v1.24，F-PROT packer 直接报 UPX。

Portable Executable Info

Compiler Products

[C] VS98 (6.0) build 8168 count=11
[LNK] VS98 (6.0) imp/exp build 8168 count=2
[---] Unmarked objects (old) count=4
[---] Unmarked objects count=27
[C++] VS98 (6.0) build 8168 count=2
id: 0xe, version: 7299 count=1
id: 0x13, version: 8034 count=5

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2011-01-19 16:10:41 UTC
Entry Point	21520
Contained Sections	3

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
UPX0	4096	16384	0	0	d41d8cd98f00b204e9800998ecf8427e	-1
UPX1	20480	4096	1536	7.07	ad0f236c2b34f1031486c8cc4803a908	5848.3
UPX2	24576	4096	512	2.8	f998d25f473e69cc89bf43af3102beea	53922

图 16: VirusTotal 中 Lab01-02.exe 的 PE 头与节表，三个节名为 UPX0、UPX1、UPX2

File type	Win32 EXE　executable　windows　win32　pe　peexe
Magic	PE32 executable (console) Intel 80386, for MS Windows, UPX compressed
TrID	UPX compressed Win32 Executable (34.7%)　Win32 EXE Yoda's Crypter (34.1%)　Win32
DetectItEasy	PE32　Packer: UPX (3.04) [NRV,best]　Compiler: Microsoft Visual C/C++ (12.00.8168) [C
Magika	PEBIN
File size	3.00 KB (3072 bytes)
PEiD packer	UPX v0.89.6 - v1.02 / v1.05 -v1.24 -> Markus & Laszlo [overlay]
F-PROT packer	UPX

History

Creation Time	2011-01-19 16:10:41 UTC

图 17: VirusTotal 中 Lab01-02.exe 的加壳识别结果，多个检测器均报 UPX

脱壳使用工具箱里的 upx，命令如下：

```
upx -d Lab01-02.exe -o Lab01-02-unpacked.exe
```

脱壳是否成功可以看三点：节名不再是 UPX0/UPX1/UPX2，PEiD 不再识别出壳，导入表从只剩 LoadLibraryA 和 GetProcAddress 变成完整的函数列表。

把脱壳后的文件再上传 VT 验证，三点都得到确认。文件大小从 3072 字节增加到 16384 字节，节名恢复成.text、.rdata、.data，入口点从 21520 变成 4496 并落在.text 区间内，各节熵值降到 1.38、1.17 和 0.24；同时 Magic 字段不再写 UPX compressed，PEiD packer 字段从 UPX 变成 Microsoft Visual C++，页面标签里的 upx 也消失了。

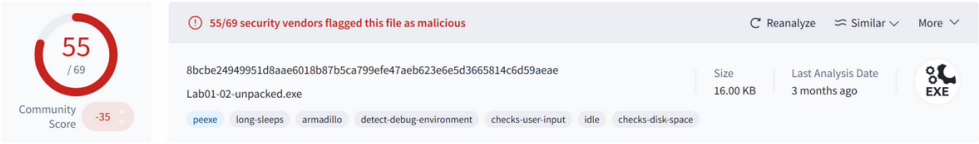
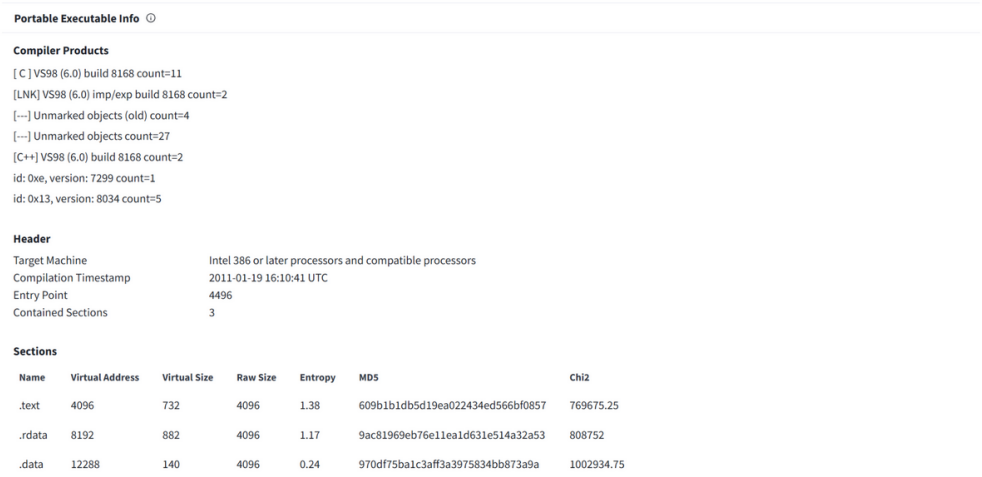


图 18: 脱壳后 Lab01-02.exe 的 VirusTotal 页，体积变为 16KB 且标签里不再有 upx



Portable Executable Info

Compiler Products

[C] VS98 (6.0) build 8168 count=11
[LNK] VS98 (6.0) imp/exp build 8168 count=2
[---] Unmarked objects (old) count=4
[---] Unmarked objects count=27
[C++] VS98 (6.0) build 8168 count=2
id: 0xe, version: 7299 count=1
id: 0x13, version: 8034 count=5

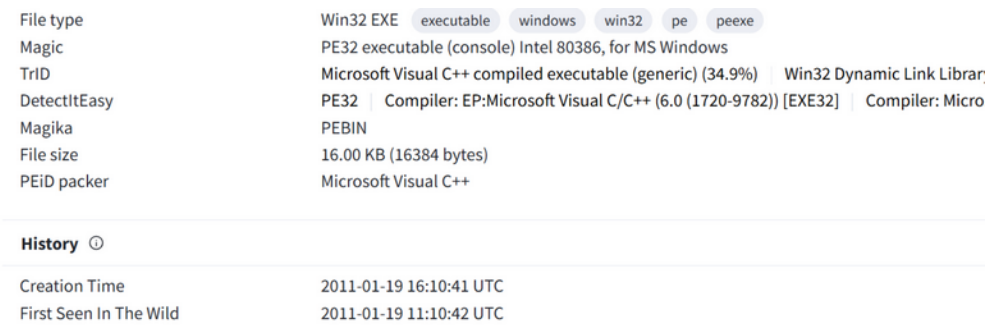
Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2011-01-19 16:10:41 UTC
Entry Point	4496
Contained Sections	3

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	732	4096	1.38	609b1b1db5d19ea022434ed566bf0857	769675.25
.rdata	8192	882	4096	1.17	9ac81969eb76e11ea1d631e514a32a53	808752
.data	12288	140	4096	0.24	970df75ba1c3aff3a3975834bb873a9a	1002934.75

图 19: 脱壳后的 PE 头与节表，节名恢复成.text、.rdata、.data



File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32 executable (console) Intel 80386, for MS Windows
TrID	Microsoft Visual C++ compiled executable (generic) (34.9%) Win32 Dynamic Link Librar
DetectItEasy	PE32 Compiler: EP:Microsoft Visual C/C++ (6.0 (1720-9782)) [EXE32] Compiler: Micro
Magika	PEBIN
File size	16.00 KB (16384 bytes)
PEiD packer	Microsoft Visual C++

History

Creation Time	2011-01-19 16:10:41 UTC
First Seen In The Wild	2011-01-19 11:10:42 UTC

图 20: 脱壳后的加壳识别结果，各检测器均不再报壳

编译时间为 2011-01-19 16:10:41 UTC，加壳版与脱壳版完全一致，说明这个字段没有被加壳过程破坏。

3. 有没有任何导入函数能够暗示出这个程序的功能？如果是，是哪些导入函数，它们会告诉你什么？

有，导入函数清楚地表明了程序的功能。

脱壳后的文件导入四个 DLL：KERNEL32.DLL、ADVAPI32.dll、MSVCRT.dll 和 WININET.dll，关键函数按 DLL 分组如下。

ADVAPI32.dll 负责服务控制。程序调用 OpenSCManagerA 打开服务控制管理器，用 CreateServiceA 注册名为 Malservice 的系统服务，再调用 StartServiceCtrlDispatcherA 把自身接入服务控制管理器，说明它是一个结构完整的 Windows 服务程序。

WININET.dll 负责网络访问。程序调用 InternetOpenA 建立 Internet 会话，再用 InternetOpenUrlA 向 http://www.malwareanalysisbook.com 发起 HTTP 请求。

KERNEL32.dll 负责进程与同步控制。CreateMutexA 和 OpenMutexA 用于保证单实例运行；GetModuleFileNameA 用于取得自身文件路径，配合 CreateServiceA 把注册的服务指向自己；CreateWaitableTimerA 和 SetWaitableTimer 创建并设置一个可等待定时器，SystemTimeToFileTime 把日期时间转换成 FILETIME，用于指定定时器的绝对触发时刻，再由 WaitForSingleObject 等待它触发；CreateThread 用于创建执行线程；ExitProcess 用于结束进程。

MSVCRT.dll 是 C 运行时库，导入的 __getmainargs、__p____initenv、__set_app_type、_controlfp 等都是程序启动时的初始化函数，只能说明这个程序由 Microsoft Visual C++ 编译，与恶意行为无关。

把这些函数连起来看，程序的行为是：用 GetModuleFileNameA 取得自身路径，注册一个指向自己的 Malservice 服务来实现开机自启，创建互斥体 HGL345 保证单实例运行，再用 SystemTimeToFileTime 和 SetWaitableTimer 设置一个绝对时刻的定时器，等定时器触发后通过 CreateThread 创建多个线程，反复向固定网址发起 HTTP 请求。VT 对脱壳版的分析进一步指出，定时器的触发时间被设在 2100 年，触发后会开出 20 个线程持续请求该网址。

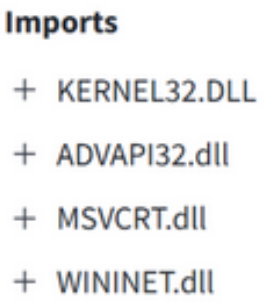


图 21: VirusTotal 中 Lab01-02.exe 的导入表

图 22: 脱壳后 Lab01-02.exe 的导入函数，左为 KERNEL32.DLL 与 ADVAPI32.dll，右为 MSVCRT.dll 与 WININET.dll

上面这份完整的函数清单来自脱壳后的版本。脱壳前的原始文件导入表通常只剩 LoadLibraryA 和 GetProcAddress 两项，UPX 的解压代码靠它们在运行时动态解析真实 API，完整的函数列表藏在压缩数据里。

4. 哪些基于主机或基于网络的迹象可以被用来确定被这个恶意代码所感染的机器？

有，主机层面能找到两条迹证，网络层面一条。

主机层面的第一条是服务。程序通过 CreateServiceA 在系统里注册了一个名为 Malservice 的服务，VT 的 Behavior 页在 Services Created 里记录了这个名字，并在 Registry keys set 里记录了对应的注册表项：

```
HKLM\System\CurrentControlSet\Services\Malservice\ImagePath
HKLM\System\CurrentControlSet\Services\Malservice\Start
```

其中 Start 的值为 DWORD 0x00000002，即 SERVICE_AUTO_START，表示开机自动启动，ImagePath 指向样本自身路径。在受感染系统上可以用 sc query Malservice、services.msc 或 Autoruns 查到它。

主机层面的第二条是互斥体。Behavior 页的 Mutexes Created 里记录了 HGL345，程序用它判断自己是否已经在运行。可以用 Process Explorer 的 Find Handle 功能搜索它。

图 23: VirusTotal 行为报告中 Lab01-02.exe 创建的服务 Malservice

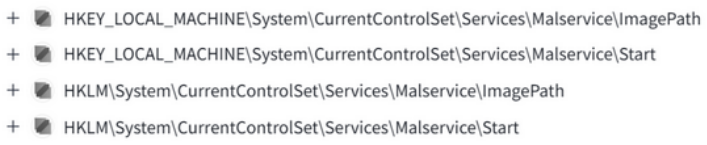


图 24: VirusTotal 行为报告中 Lab01-02.exe 写入的 Malservice 注册表项

图 25: VirusTotal 行为报告中 Lab01-02.exe 创建的互斥体 HGL345

网络层面的迹证是程序通过 InternetOpenUrlA 访问的地址 www.malwareanalysisbook.com。它硬编码在程序内部，正是《恶意代码分析实战》这本书的官网域名，实验样本拿它当靶子，实际不会造成危害。

(3) 分析 Lab01-03.exe 文件，回答以下问题：

1. 将 Lab01-03.exe 文件上传至 http://www.VirusTotal.com/进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗？

匹配到了已有的反病毒软件特征。Lab01-03.exe 被 61/71 家引擎报恶意，VT 聚合出的流行威胁名是 trojan，家族标签为 graftor、genome 和 tnega。

检出名称里普遍带 Tnega、Graftor 字样，其中 Alibaba 和 AliCloud 直接用了 TrojanClicker 这个类别。

页面标签也值得留意，除了常见项，还有 fsg 和 overlay：前者指向加壳工具，后者指向文件尾部在最后一个节之外追加的数据。

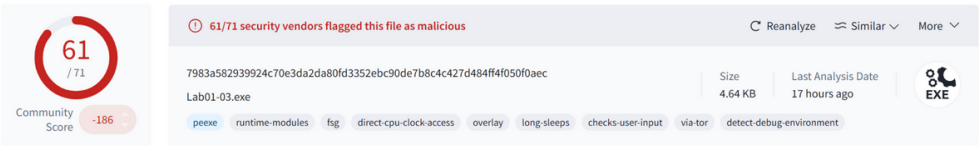


图 26: VirusTotal 对 Lab01-03.exe 的检出结果，61/71 家引擎报恶意

2. 是否有这个文件被加壳或混淆的任何迹象？如果是这样，这些迹象是什么？如果该文件被加壳，请进行脱壳，如果可能的话。

有，这个文件被 FSG 1.0 加壳了。判断迹象有以下六条。

第一，节名异常且不完整。三个节分别叫 t、ta、a，看不出任何含义，既不是.text、.rdata、.data 这类链接器默认名，也不像 UPX 那样有规律，属于节名被破坏后留下的残片。

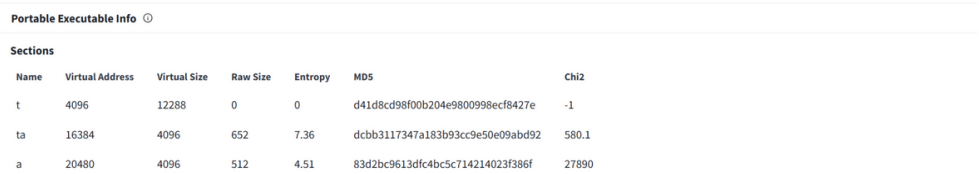
第二，第一个节 t 的 Raw Size 为 0，Virtual Size 却有 12288 字节。它在磁盘上不占空间，说明这段空间要留到运行时由解压代码填充。

第三，第二个节 ta 的熵值为 7.36。接近随机分布，说明里面装的是压缩数据加解压代码。

第四，文件体积只有 4752 字节（4.64KB），对这类程序来说异常小。

第五，PE 头里的 Header 信息无法读出。VT 的 Portable Executable Info 区域只有节表和导入表，没有 Entry Point 和 Contained Sections 这两项，而脱壳后的 Lab01-02 在同一区域是有的，说明加壳破坏了 PE 头的可解析性。

第六，多个检测器一致报壳。DetectItEasy 报 Packer: FSG (1.0)，PEiD 报 FSG v1.00 (Eng) -> dulek/xt，F-PROT packer 报 FSG，Command packer 也报 FSG；TrID 则给出一堆互相矛盾的结论（Win16 NE 可执行文件、Win32 可执行文件、OS/2 可执行文件等），说明它无法识别。

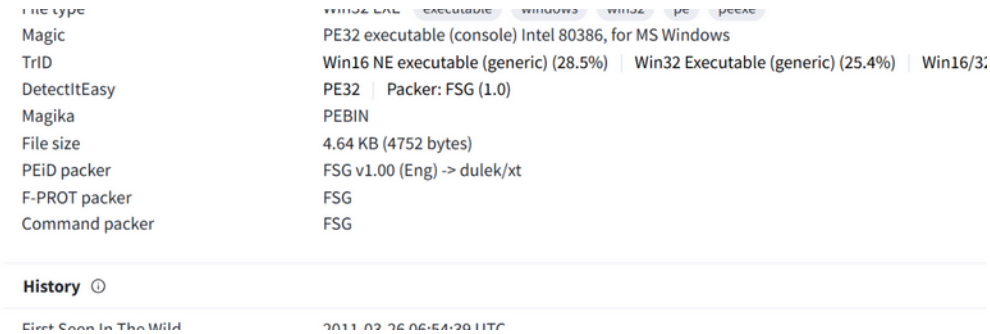


Portable Executable Info

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
t	4096	12288	0	0	d41d8cd98f00b204e9800998ecf8427e	-1
ta	16384	4096	652	7.36	dcbb3117347a183b93cc9e50e09abd92	580.1
a	20480	4096	512	4.51	83d2bc9613dfc4bc5c714214023f386f	27890

图 27: VirusTotal 中 Lab01-03.exe 的节表，三个节名为 t、ta、a



Magic	PE32 executable (console) Intel 80386, for MS Windows
TrID	Win16 NE executable (generic) (28.5%) | Win32 Executable (generic) (25.4%) | Win16/3
DetectItEasy	PE32 | Packer: FSG (1.0)
Magika	PEBIN
File size	4.64 KB (4752 bytes)
PEiD packer	FSG v1.00 (Eng) -> dulek/xt
F-PROT packer	FSG
Command packer	FSG

History

图 28: VirusTotal 中 Lab01-03.exe 的加壳识别结果，多个检测器均报 FSG

由于 FSG 是早期的压缩壳，没有可以像 UPX 那样直接调用的自动脱壳工具，工具箱里的 upx 只认 UPX 格式，对 FSG 文件无效。可行的做法是用 OllyICE 手工脱壳：从入口点开始单步跟踪解压代码，直到控制权跳转到原始入口点（OEP），再把进程内存 dump 出来重建导入表。这属于动态调试，超出基本静态分析的范围。

3. 有没有任何导入函数能够暗示出这个程序的功能？如果是，是哪些导入函数，它们会告诉你什么？

导入函数提供不了任何关于功能的线索，原因是文件被加壳了。

这个文件的导入表里只有 KERNEL32.dll 一个 DLL，下面只有两个函数：LoadLibraryA 和 GetProcAddress。

这两个函数恰好是壳的解压代码自己需要的：LoadLibraryA 用来在运行时加载 DLL，GetProcAddress 用来按名字取得函数地址。程序真正要用到的 API——比如创建窗口、调用浏览器组件、发起网络请求——在加壳时被压缩进壳里，运行时才由这两个函数动态解析出来，所以在静态的导入表里看不到。

可以和 Lab01-02 做一个对照：未加壳的 Lab01-02 导入四个 DLL、十几个函数，能直接看出服务控制、网络访问和同步控制三类能力；而 Lab01-03 只有一个 DLL、两个函数，什么都推不出来。导入表的”空”本身就是加壳的旁证。

所以这一题的结论是：导入函数无法暗示功能，要判断功能必须脱壳或者改用动态分析。VT 的行为报告显示它会通过 COM 组件打开浏览器并跳转到 http://www.malwareanalysisbook.com/ad.html，但这是动态分析得出的结论，不是从导入表读出来的。

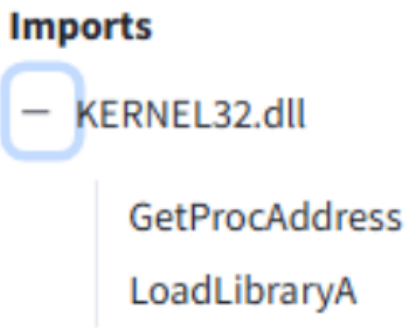


图 29: VirusTotal 中 Lab01-03.exe 的导入表，只有 KERNEL32.dll 一个 DLL

4. 有哪些基于主机或基于网络的迹象，可以被用来确定被这个恶意代码所感染

的机器？

网络层面有一条明确可用的迹证：程序会向 http://www.malwareanalysisbook.com/ad.html 发起 GET 请求。VT 的行为报告记录了这条请求，DNS 解析记录里也有 www.malwareanalysisbook.com。这个地址硬编码在程序内部，是广告点击器的典型做法——用浏览器打开指定页面。

主机层面没有可用的迹证。所以判断机器是否被感染主要靠两条：一是排查网络层是否有对该域名的请求记录，二是检查浏览器是否在用户没有操作的情况下自行打开过该页面。

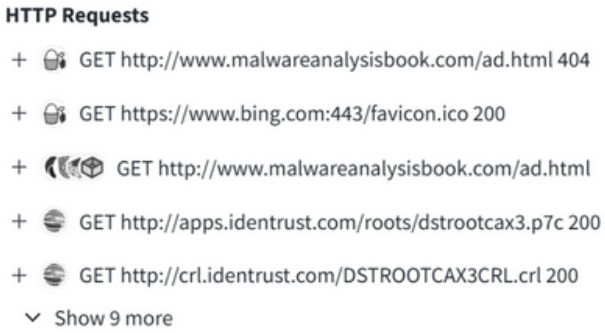


图 30: VirusTotal 行为报告中 Lab01-03.exe 发起的 HTTP 请求，第一条为该样本自身的请求

(4) 分析 Lab01-04.exe 文件，回答以下问题：

1. 将 Lab01-04.exe 文件上传至 http://www.VirusTotal.com/进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗？

匹配到了已有的反病毒软件特征。Lab01-04.exe 被 62/70 家引擎报恶意，VT 聚合出的流行威胁名是 trojan，威胁类别为 trojan/downloader/dropper，家族标签为 cerbu、gofot 和 downldr。检出名称里 Downloader、Dropper、DownLoader 这些字样反复出现，直接点出了它的类型——下载器和释放器。

图 31: VirusTotal 对 Lab01-04.exe 的检出结果，62/70 家引擎报恶意

2. 是否有这个文件被加壳或混淆的任何迹象？如果是这样，这些迹象是什么？如果该文件被加壳，请进行脱壳，如果可能的话。

没有加壳，也没有混淆，因此不需要脱壳。判断依据有四条。

第一，节名是编译器的默认名。四个节分别叫.text、.rdata、.data 和.rsrc，前三个是链接器默认命名，.rsrc 是资源节，都是正常可执行文件的组成。

第二，用 PEiD 检测没有识别出任何壳，签名字段显示为 Microsoft Visual C++；DetectItEasy 的检测结果里只有 Compiler、Linker、Tool 三类条目，没有任何 Packer 或 Protector 条目。

第三，各节熵值很低。.text 为 3.12、.rdata 为 1.59、.data 为 0.51，压缩或加密数据一般在 7 以上，这里最高的才 3.12。

第四，文件体积 36KB，与它的代码量相称，没有加壳文件那种异常的体积压缩。

VT 的加壳识别里 Cyren packer 和 Varist packer 两个字段显示的是 rsrc——这不是壳的名字，而是资源节的节名，说明这两个引擎只是检测到文件里有资源节，并非真的识别出壳。

Portable Executable Info

Compiler Products

[C] VS98 (6.0) build 8168 count=11
[LNK] VS98 (6.0) imp/exp build 8168 count=2
[---] Unmarked objects count=36
[RES] VS98 (6.0) cvtres build 1720 count=1
id: 0xc, version: 7291 count=1
id: 0xe, version: 7299 count=2
id: 0x13, version: 8034 count=5
id: 0x16, version: 8168 count=2

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2019-08-30 22:26:59 UTC
Entry Point	5583
Contained Sections	4

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	1824	4096	3.12	77df9f7ebc4a2bc4bdf2b454d7635aee	419793.88
.rdata	8192	978	4096	1.59	d630e1eb49ed821e38202aefef911a39	729061.12
.data	12288	332	4096	0.51	d9a3822a7733a76776d8b6e64e364b9d	946649.25
.rsrc	16384	16480	20480	0.71	398569177d4d82090d3e1747be560f9a	4618603

图 32: VirusTotal 中 Lab01-04.exe 的 PE 头与节表，四个节名为.text、.rdata、.data、.rsrc

File type	Win32 EXE　executable　windows　win32　pe　peexe
Magic	PE32 executable (GUI) Intel 80386, for MS Windows
TrID	Microsoft Visual C++ compiled executable (generic) (33.3%)　Win32 Dynamic Link Librar
DetectItEasy	PE32　Compiler: EP:Microsoft Visual C/C++ (6.0 (1720-9782)) [EXE32]　Compiler: Micro
Magika	PEBIN
File size	36.00 KB (36864 bytes)
PEiD packer	Microsoft Visual C++
Cyren packer	rsrc
Varist packer	rsrc

History

图 33: VirusTotal 中 Lab01-04.exe 的加壳识别结果，PEiD 与 DetectItEasy 均未报壳

3. 这个文件是什么时候被编译的？

这个文件编译于 2019-08-30 22:26:59 UTC。这个时间来自 PE 头 IMAGE_FILE_HEADER 结构里的 Time Date Stamp 字段，用 PEview 打开就能读到。

pFile	Data	Description	Value
000000EC	014C	Machine	IMAGE_FILE_MACHINE_I386
000000EE	0004	Number of Sections	
000000F0	5D69A2B3	Time Date Stamp	2019/08/30 22:26:59 UTC
000000F4	00000000	Pointer to Symbol Table	
000000F8	00000000	Number of Symbols	
000000FC	00E0	Size of Optional Header	
000000FE	010F	Characteristics	
		0001	IMAGE_FILE_RELOCS_STRIPPED
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0100	IMAGE_FILE_32BIT_MACHINE

图 34: 用 PEview 查看 Lab01-04.exe 的 IMAGE_FILE_HEADER，Time Date Stamp 为 2019/08/30 22:26:59 UTC

有一点值得注意：VT 记录的最早野外发现时间是 2011-07-05，比编译时间早了八年多。两者矛盾，说明 PE 头里的这个字段不能全信——要么它被工具改写过，要么这份样本在 2019 年被重新编译过。做时间线判断时不能只依赖这个字段。

4. 有没有任何导入函数能够暗示出这个程序的功能？如果是，是哪些导入函数，它们会告诉你什么？

这个文件导入三个 DLL：KERNEL32.dll、ADVAPI32.dll 和 MSVCRT.dll。

KERNEL32.dll 里的函数可以分成几组。FindResourceA、LoadResource、SizeofResource 和 GetModuleHandleA 一组，作用是从自身镜像里定位并读取一个内嵌资源。GetTempPathA、CreateFileA、WriteFile 和 MoveFileA 一组，作用是把读出来的资源写到临时目录再移动到别的位置。GetWindowsDirectoryA 说明目标路径涉及 Windows 目录。WinExec 用于执行程序。OpenProcess 和 CreateRemoteThread 是最关键的一对，含义是打开另一个进程并在其中创建线程，也就是进程注入。CloseHandle、LoadLibraryA、GetProcAddress 和 GetCurrentProcess 则是配套的句柄管理与动态 API 解析。

ADVAPI32.dll 里的三个函数连起来就是一次权限提升：OpenProcessToken 取得自身进程的访问令牌，LookupPrivilegeValueA 查出某个特权对应的 LUID，AdjustTokenPrivileges 把该特权启用。VT 的行为报告直接给出了答案——申请的特权是 SE_DEBUG_PRIVILEGE，即调试特权，有了它才能打开像 winlogon.exe 这样的系统进程。

MSVCRT.dll 里是 ___getmainargs、___p____initenv、___set_app_type、_controlfp 这类 C 运行时初始化函数，只能说明程序由 Microsoft Visual C++ 编译。

把 KERNEL32 和 ADVAPI32 的导入合起来看，程序的行为链条是：读取自身的内嵌资源并释放成文件，启用 SeDebugPrivilege 特权，打开一个高权限进程并把代码注入进去，最后执行释放出来的程序。

图 35: Lab01-04.exe 的导入函数，左为 KERNEL32.dll 与 ADVAPI32.dll，右为 MSVCRT.dll

5. 有哪些基于主机或基于网络的迹象，可以被用来确定被这个恶意代码所感染

的机器？

网络层面有一条明确的迹证：http://www.practicalmalwareanalysis.com/updater.exe。VT 的行为报告记录到对该域名的 DNS 解析，以及多次针对 updater.exe 的 HTTP 请求。这是硬编码在样本里的下载地址，也正是《恶意代码分析实战》这本书的域名。

主机层面有三条。第一条是权限提升。行为报告在 Permissions Requested 里记录了 SE_DEBUG_PRIVILEGE，说明样本执行时会申请调试特权，这条痕迹可以通过监控进程令牌权限的变更来发现。

第二条是进程注入的目标。行为报告的进程列表里出现了 C:\WINDOWS\system32\winlogon.exe，与导入表里的 OpenProcess 加 CreateRemoteThread 对应，说明样本会把自己注入到 winlogon.exe 中，受感染机器上会出现异常的 winlogon.exe 线程或子进程。

第三条是释放的文件。行为报告显示样本所在目录被写入了 winup.exe 并被运行，VT 的分析把释放路径标为 C:\Windows\System32\wupdmgr.exe，这个名字模仿的是 Windows 更新管理器。VT 的 Contained Resources By Type 显示文件里有一个 BIN 类型的资源，正是 Resource Hacker 要抽取的对象。

注意：真正发起下载的 URLDownloadToFileA 并不在 Lab01-04.exe 自己的导入表里，它属于被释放或注入的那个模块，所以排查时不能只看 Lab01-04.exe 本身。

图 36: VirusTotal 行为报告中 Lab01-04.exe 申请的 SE_DEBUG_PRIVILEGE 与创建的进程

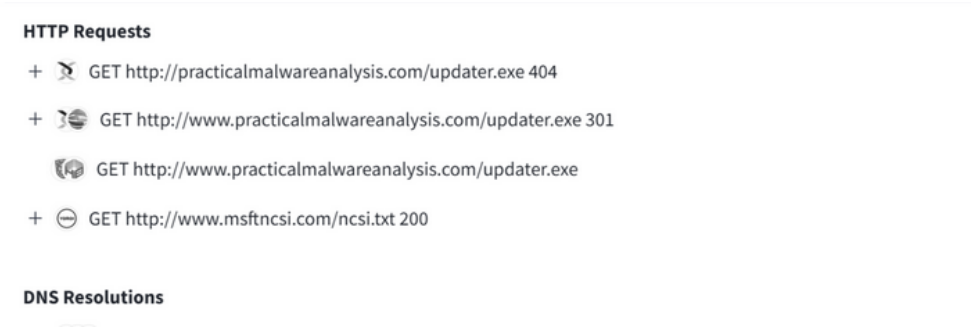


图 37: VirusTotal 行为报告中 Lab01-04.exe 对 updater.exe 发起的 HTTP 请求

6. 这个文件在资源段中包含一个资源。使用 Resource Hacker 工具来检查资源，然后抽取资源。从资源中你能发现什么吗？

资源里藏着一个完整的可执行文件。

VT 的资源统计显示这个文件里只有一个资源，类型为 BIN、语言为 ENGLISH US。

图 38: VirusTotal 中 Lab01-04.exe 的资源统计，含一个 BIN 类型资源

用 Resource Hacker 打开 Lab01-04.exe，左侧树里同样只有一个资源，类型 BIN，编号 101:1033，大小 16384 字节。把右侧切到 Binary View 看十六进制，开头两个字节是 4D 5A，也就是 MZ 头，说明这不是图标或字符串一类的普通资源，而是一个 PE 文件；再往下能看到 PE 头、Rich 标记以及.text、.rdata、.data 三个节名。

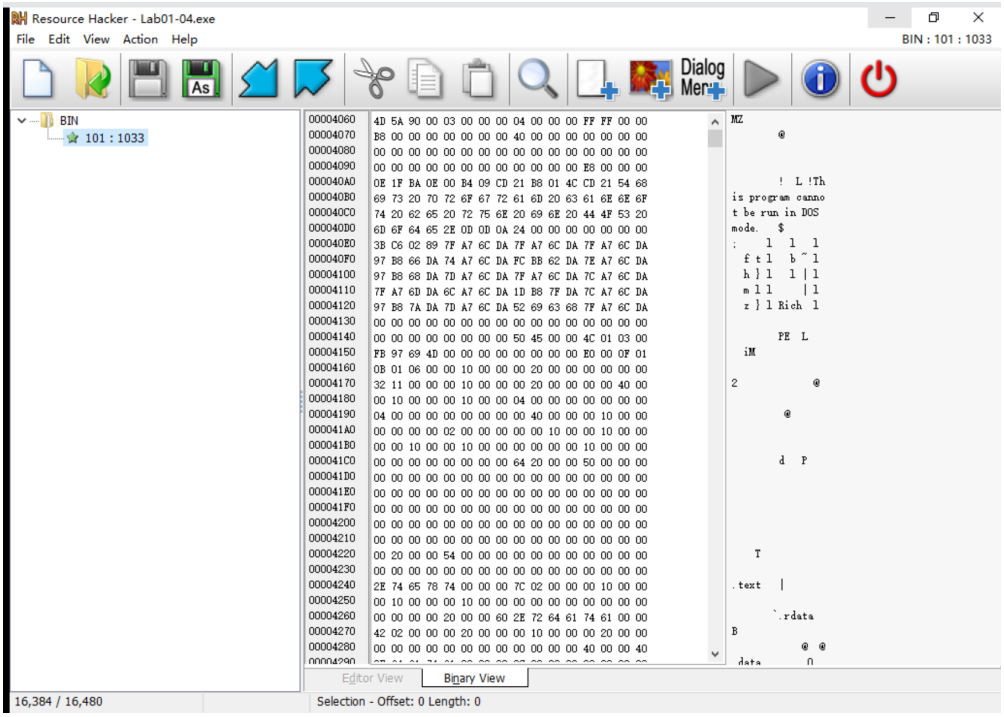
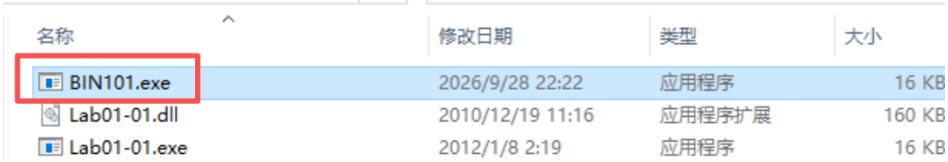


图 39: 用 Resource Hacker 查看 Lab01-04.exe 的资源，二进制开头为 MZ，可见 PE 头和三个节名

按题目要求把资源提取出来保存为 BIN101.exe，得到一个 16KB 的可执行文件。



名称	修改日期	类型	大小
BIN101.exe	2026/9/28 22:22	应用程序	16 KB
Lab01-01.dll	2010/12/19 11:16	应用程序扩展	160 KB
Lab01-01.exe	2012/1/8 2:19	应用程序	16 KB

图 40: 从资源中提取出的 BIN101.exe，大小 16KB

对提取出来的文件做静态分析，它同样没有加壳，Magic 显示为 PE32 executable (GUI)，PEiD 报 Microsoft Visual C++，编译时间是 2011-02-27 00:16:59 UTC。VT 对它的检出率是 61/71。

关键在于它的导入表：除了 KERNEL32.dll 和 MSVCRT.dll，它还多导入了一个 urlmon.dll。urlmon 是 Windows 处理 URL 的库，把任意文件下载到本地磁盘的 URLDownloadToFileA 就在这个库里。

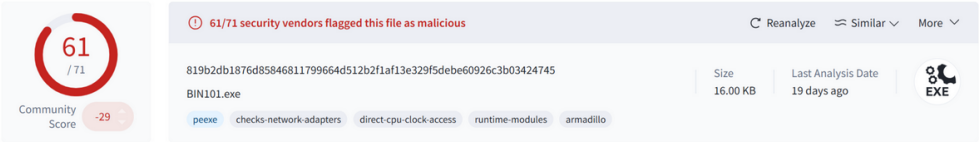


图 41: VirusTotal 对提取出的 BIN101.exe 的检出结果，61/71 家引擎报恶意

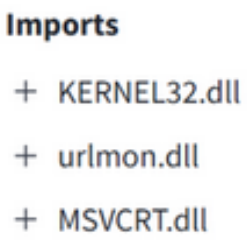


图 42: BIN101.exe 的导入表，除 KERNEL32.dll 和 MSVCRT.dll 外还导入了 urlmon.dll

VT 对它的行为分析印证了这一点：它先取得临时目录和 Windows 目录的路径，尝试执行临时目录里的 winup.exe，然后用 URLDownloadToFileA 从 http://www.practicalmalwareanalysis.com/updater.exe 下载第二阶段的载荷，保存到 System32 目录并重命名为 wupdmgrd.exe，最后用 WinExec 执行。

因此下载动作由这个被释放出来的 URLDownloadToFileA 组件完成，Lab01-04.exe 只负责把资源取出来、释放成文件、再设法让它跑起来。两者分工明确。

与此同时，这个资源的编译时间是 2011-02-27，而 Lab01-04.exe 自己的时间戳是 2019-08-30；同一套恶意代码的两个组件相差八年，而 VT 记录的最早野外发现时间 2011-07-05 与资源的 2011 年吻合、与外壳的 2019 年不吻合。说明 Lab01-04.exe 头部的编译时间被改写过，真实时间应在 2011 年前后。

综上所述，完整的链条是：Lab01-04.exe 运行后读取自身.rsrc 节里的 BIN 资源并释放成文件，启用 SeDebugPrivilege，把代码注入 winlogon.exe，被释放的组件再从 practicalmalwareanalysis.com 下载 updater.exe 到 System32 并改名 wupdmgrd.exe 执行。这是典型的下载器套下载器的两段式结构。

第二部分：在 windows10 虚拟机中分析 5 个真实恶意代码教学样本文件

(1) WannaCrypt0r

1. 使用本章描述的工具和技术做基本静态分析

通过 GnuWin32 的 file.exe 报出类型为 PE32 executable (GUI) Intel 80386 32-bit，PEiD 判定未加壳、未混淆，编译器是 Microsoft Visual C++ 6.0，四个节全是链接器的默认命名。综合来看，这是一个自解压式的勒索软件投放器。

查壳。PEiD 的签名框给出的是编译器名而不是壳名；EP Section 是.text，说明入口点落在代码段而不是解压代码；First Bytes 为 55,8B,EC,6A，即 push ebp、mov ebp, esp 的标准函数序言。三条依据一致，样本未加壳、未混淆，不需要脱壳。

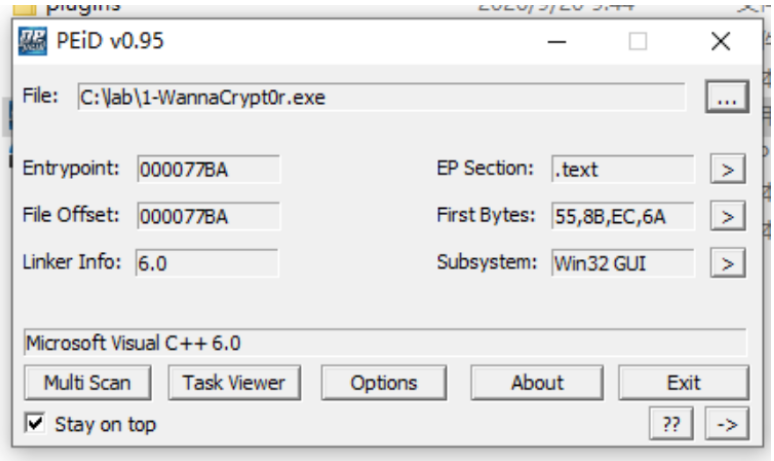


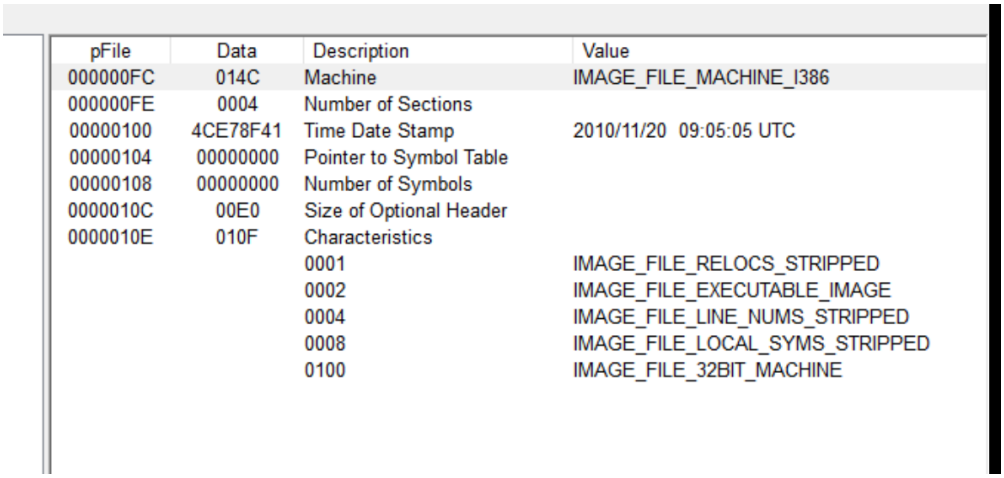
图 43: PEiD 加载 1-WannaCrypt0r.exe 的检测结果，签名为编译器而非壳

字符串。用 strings 提取两种编码的可打印字符串共 11035 行，其中大部分是乱码，有效串集中在文件的首尾两端，最有特征价值的几条如下。

```
WanaCrypt0r
WANACRY!
WNcry@2ol7
tasksche.exe
Global\MsWinZonesCacheCounterMutexA
```

家族名、加密标记和压缩包口令都在其中，tasksche.exe 是它释放的组件，互斥体名可用于主机排查。需要注意的是整份输出里没有任何 URL 或域名，说明联网能力不在这层，由它释放出去的模块完成。

PE 头与分节。



pFile	Data	Description	Value
000000FC	014C	Machine	IMAGE_FILE_MACHINE_I386
000000FE	0004	Number of Sections	
00000100	4CE78F41	Time Date Stamp	2010/11/20 09:05:05 UTC
00000104	00000000	Pointer to Symbol Table	
00000108	00000000	Number of Symbols	
0000010C	00E0	Size of Optional Header	
0000010E	010F	Characteristics	
		0001	IMAGE_FILE_RELOCS_STRIPPED
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0100	IMAGE_FILE_32BIT_MACHINE

图 44: PEview 中 1-WannaCrypt0r.exe 的 IMAGE_FILE_HEADER

节数为 4，Subsystem 为 GUI，入口点 0x77BA 落在.text 区间内；四个节名都是默认名，虚拟大小与原始大小相称，没有加壳文件那种解压空间，.text 是唯一可执行的节。有两处异常值得指出：编译时间戳是 2010-11-20，比该家族的已知活动时间早好几年，这个字段被改过；.rsrc 节的原始大小是 3,449,856 字节，而代码段.text 只有 27,056 字节，资源是代码的 127 倍，占整个文件体积的 98%。

导入函数与资源节。Dependency Walker 显示静态导入只有 KERNEL32、USER32、ADVAPI32 和 MSVCRT 四个 DLL；字符串里还出现过 SHELL32、OLEAUT32、WS2_32、MSVCP60 四个库名却不在导入表内，说明它们是运行时用 LoadLibrary 动态加载的，网络能力就藏在 WS2_32 这一层。

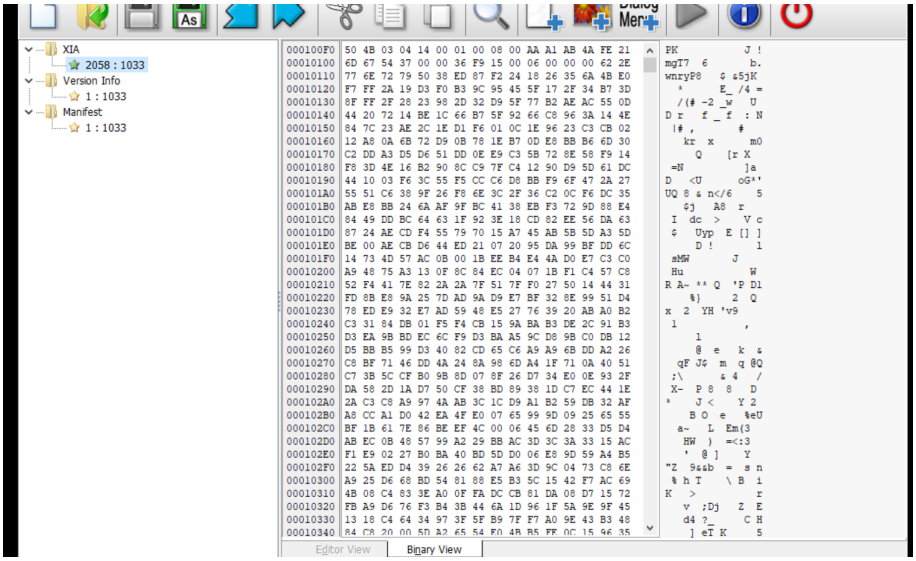


图 45: 用 Resource Hacker 展开.rsrc，左侧为资源树，右侧为 XIA 资源的 Binary View

资源树只有三个条目：自定义类型名 XIA 的 2058:1033、Version Info 和 Manifest，没有图标、对话框和字符串表。XIA 的开头字节是 50 4B 03 04，即 ZIP 的本地文件头签名，标志位 bit 0 置位说明这个 ZIP 是加密的，正好对应字符串里的口令 WNcry@2ol7；版本资源的内容则被伪造成微软的 diskpart.exe。样本把真正的载荷加密打包藏在资源段里，运行时再取出释放，这是自解压投放器最直接的证据。

2. 用 Windows 系统自带的防病毒软件 MicroSoft Defender AntiVirus 扫描样本，给出检测结果截图

用 Windows 安全中心里的 Microsoft Defender Antivirus 对样本扫描。做静态分析时为避免样本被自动清除，工作目录曾加进过排除项，这一步先把 C:\lab 从排除列表中移除，再用”扫描选项”发起一次自定义扫描。

检出结果是一次性给出的，五个样本全部被检出。

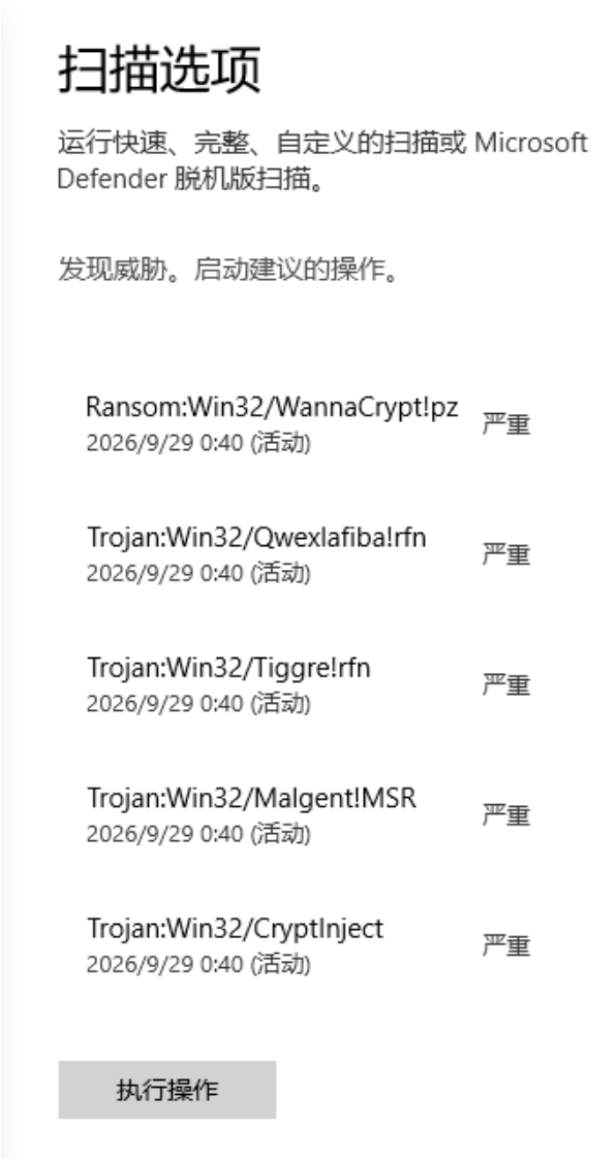


图 46: Microsoft Defender Antivirus 自定义扫描的检出结果，五个样本全部报毒

第一条对应本次分析的 1-WannaCrypt0r.exe，其余四条分别对应另外四个样本。

样本自身的检出详情如下。

图 47: Microsoft Defender Antivirus 对 1-WannaCrypt0r.exe 的威胁详情，威胁名为 Ransom:Win32/WannaCrypt!pz

3. 通过 VirusTotal、国家计算机病毒协同分析平台扫描样本，给出检测结果，包括但不限于：SSDeep、TLSH 等哈希值、PE 文件头的信息、PE 文件的函数信息

一、VirusTotal

检出结果是 67/71，71 个引擎里有 67 个报恶意。

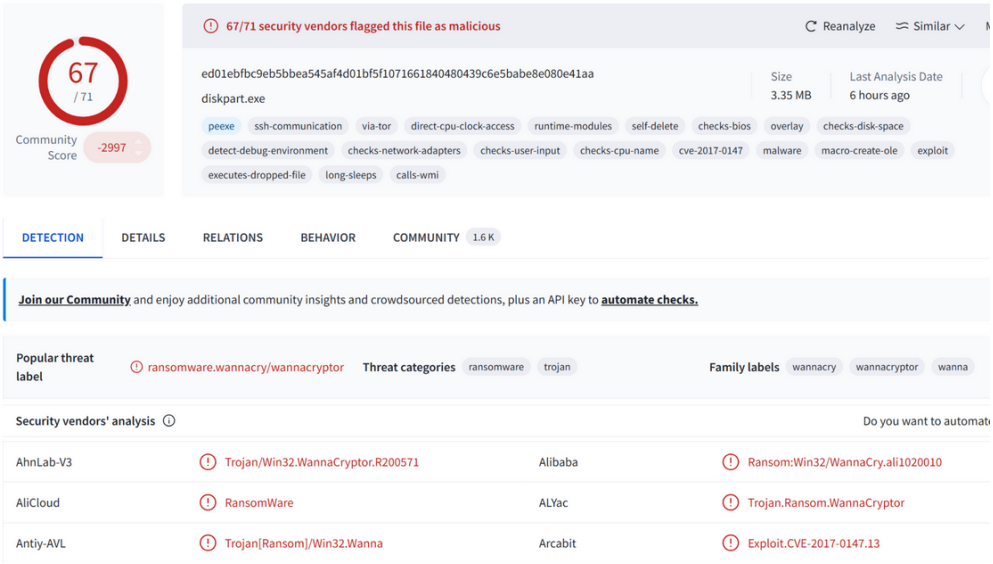


图 48: VirusTotal 对 1-WannaCrypt0r.exe 的检出结果，67/71 家引擎报恶意

哈希值。VT 给出的完整哈希信息如下，其中红框内两项是模糊哈希。

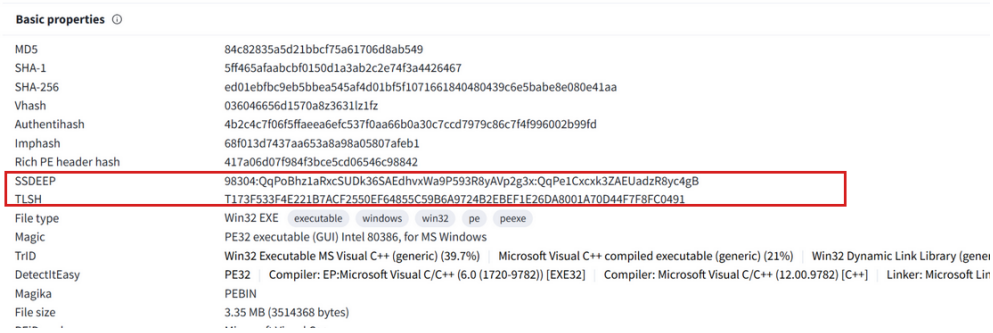


图 49: VirusTotal 给出的哈希值，红框内为 SSDeep 与 TLSH

PE 文件头信息。

Portable Executable Info

Compiler Products

[IMP] Windows Server 2003 SP1 DDK build 4035 count=13
[---] Unmarked objects count=163
[C++] VS98 (6.0) SP6 build 8804 count=7
[RES] VS98 (6.0) SP6 cvtres build 1736 count=1
id: 0xc, version: 7291 count=2
id: 0xb, version: 8047 count=1
id: 0xe, version: 7299 count=4
id: 0xa, version: 8047 count=11
id: 0x4, version: 8047 count=4

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2010-11-20 09:05:05 UTC
Entry Point	30650
Contained Sections	4

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	27056	28672	6.4	920e964050a1a5dd60dd00083fd541a2	205225.92
.rdata	32768	24432	24576	6.66	2c42611802d585e6eed68595876d1a15	421271.69
.data	57344	6488	8192	4.46	83506e37bd8b50cacabd480f8eb3849b	502756.31
.rsrc	65536	3448736	3448832	8	f99ce7dc94308f0a149a19e022e4c316	647.82

图 50: VirusTotal 中 1-WannaCrypt0r.exe 的 PE 头与节表

PE 文件的函数信息。

Imports

+ KERNEL32.dll
+ USER32.dll
+ ADVAPI32.dll
+ MSVCRT.dll

Contained Resources By Type

RT_MANIFEST	1
RT_VERSION	1
XIA	1

图 51: VirusTotal 中 1-WannaCrypt0r.exe 的导入表与资源类型

导入表里只有四个 DLL：KERNEL32.dll、USER32.dll、ADVAPI32.dll 和 MSVCRT.dll。

同一段的 Contained Resources By Type 显示资源类型为 RT_MANIFEST、RT_VERSION 和 XIA 各一个。其中 XIA 是作者自定义的类型名。

二、国家计算机病毒协同分析平台

平台的检出结果是 27/29。

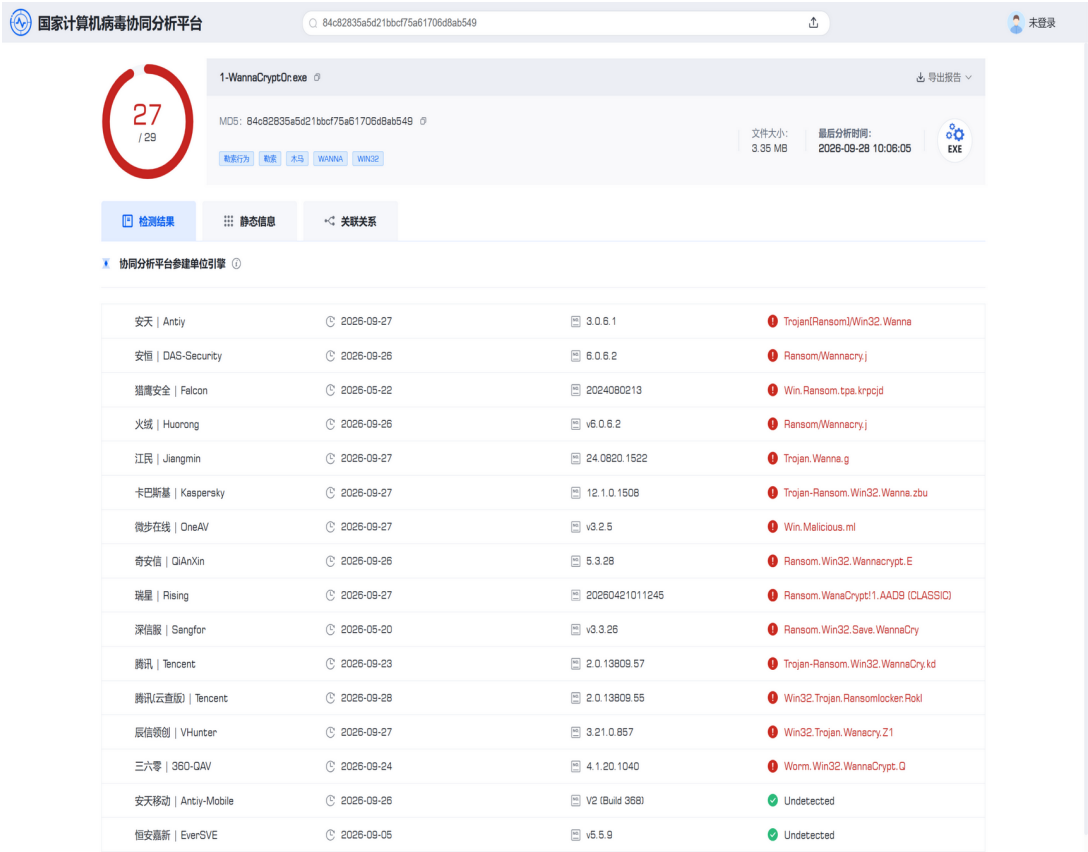


图 52: 国家计算机病毒协同分析平台对 1-WannaCrypt0r.exe 的检出结果，27/29

平台的检出标签是勒索行为、勒索、木马、WANNA、WIN32，与 VT 的家族标注一致。创新试验引擎里南开大学报 Virus.ByS.ed01ebfb，CVERC 的 Yara 规则报 Ransomware.Win.WannaCry。29 个引擎中只有安天移动和恒安嘉新两个未检出。

哈希值。

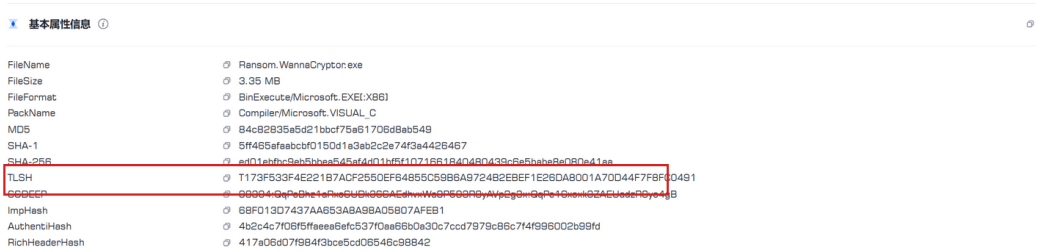
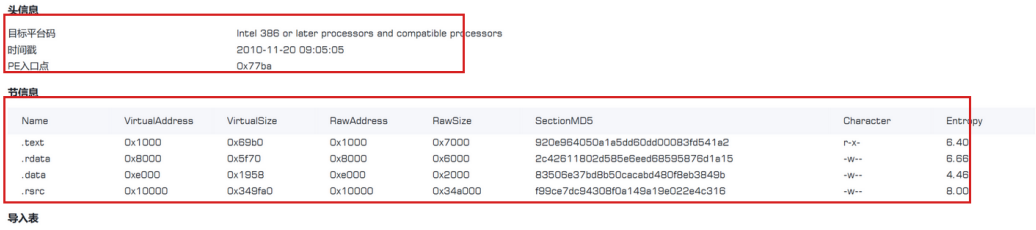


图 53: 国家计算机病毒协同分析平台给出的哈希值，红框内为 TLSH 与 SSDEEP

平台的 FileFormat 是 BinExecute/Microsoft.EXE[:X86]，PackName 是 Compiler/Microsoft.VISUAL_C。平台的原始文件名栏显示 Ransom.WannaCryptor.exe，历史文件名里保留了我们提交时的

1-WannaCrypt0r.exe。

PE 格式信息。平台的静态信息页给出头信息、节信息、导入表和资源节信息。



头信息

目标平台码	Intel 386 or later processors and compatible processors
时间戳	2010-11-20 09:05:05
PE入口点	0x77ba

节信息

Name	VirtualAddress	VirtualSize	RawAddress	RawSize	SectionMD5	Character	Entropy
.text	0x1000	0x69b0	0x1000	0x7000	920e964050e1e5dd60dd00083fd541e2	r-x-	6.40
.rdata	0x8000	0x5f70	0x8000	0x6000	2c42611802d585e6eed68595876d1e15	-w--	6.66
.data	0xe000	0x1958	0xe000	0x2000	83506e37bd8b50cacabd480f8eb3849b	-w--	4.46
.rsrc	0x10000	0x349fe0	0x10000	0x34e000	f99ce7dc94308f0e149e19e022e4c316	-w--	8.00

导入表

图 54: 国家计算机病毒协同分析平台的 PE 头信息与节信息，红框分别标出头部字段和节表

导入表

- KERNEL32.dll
- USER32.dll
- ADVAPI32.dll
- MSVCRT.dll

资源类型

XIA	1
RT_VERSION	1

资源节信息

RsrcOffset	RsrcSize	RsrcSHA256	RsrcRootName	Entropy
0x000100f0	0x00349635	5873c1b5b246c80ab88172d3294140a83d711cd64520e0c7dd7837f028146b80	XIA	8.00
0x0035972f	0x000003f8	f8cbc0ddb17a85f2ba099416961efef915f8eba926681df7cd2c1fa69f3c2b6a	RT_VERSION	3.53
0x00359ab0	0x000004ef	590b5bae6a9c329da6d5b836e3ec9baeb9607b8ea88e7015a01e021fc416707f		5.04

图 55: 国家计算机病毒协同分析平台的导入表、资源类型与资源节信息

(2) hrtg

1. 使用本章描述的工具和技术做基本静态分析

通过 GnuWin32 的 file.exe 报出类型为 PE32 executable (GUI) Intel 80386 32-bit。这只样本体积 41.8 MB，PEiD 没有匹配到任何签名，但四个节都是链接器默认命名、导入表完整、文件末尾没有追加数据，综合判定未加壳、未混淆。它的结构很特别：40 MB 集中在一个只读数据节里。

查壳。PEiD 的签名框显示为 Nothing found，既不是壳名也不是编译器名，这是它那份特征库较老的缘故，不能据此判定加壳。真正的判据来自另外三处：EP Section 是.text，入口点落在代码段而不是解压代码；四个节名叫.text、.rdata、.data、.rsrc，全是链接器默认命名；Dependency Walker 解析出了 12 个 DLL 的完整导入函数表。加壳文件的导入表会被压缩到只剩 LoadLibraryA 和 GetProcAddress 两项，这里不存在这种情况，所以判为未加壳、未混淆。

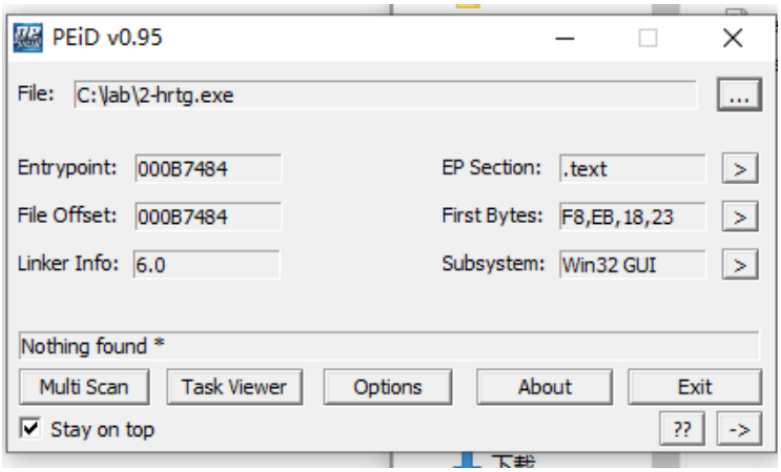


图 56: PEiD 加载 2-hrtg.exe 的检测结果，未匹配到签名

字符串。用 strings 以最小长度 8 提取出 27616 行，其中最有特征价值的是这几条。

```
taskkill /f /im explorer.exe
http://google.co.ck/search?q=dank+memz
http://google.co.ck/search?q=how+to+remove+memz+trojan+virus
C:\Windows\System32\dead.bat
C:\Windows\System32\mmc.exe
EDriver32.sys
```

这几条把它的底细交代得比较清楚：结束 explorer 进程、往 System32 里释放 dead.bat、调用 mmc.exe，还有一批 google.co.ck 上的搜索串，其中反复出现 MEMZ——那是一只在网上公开的恶搞程序，本次样本是它的中文改版。字符串里还出现了 InstallWinIoDriver、RemoveWinIoDriver 和设备名 EDriver，说明它会安装一个改名成 EDriver32.sys 的 WinIo 内核驱动，用来取得硬件级访问权限。

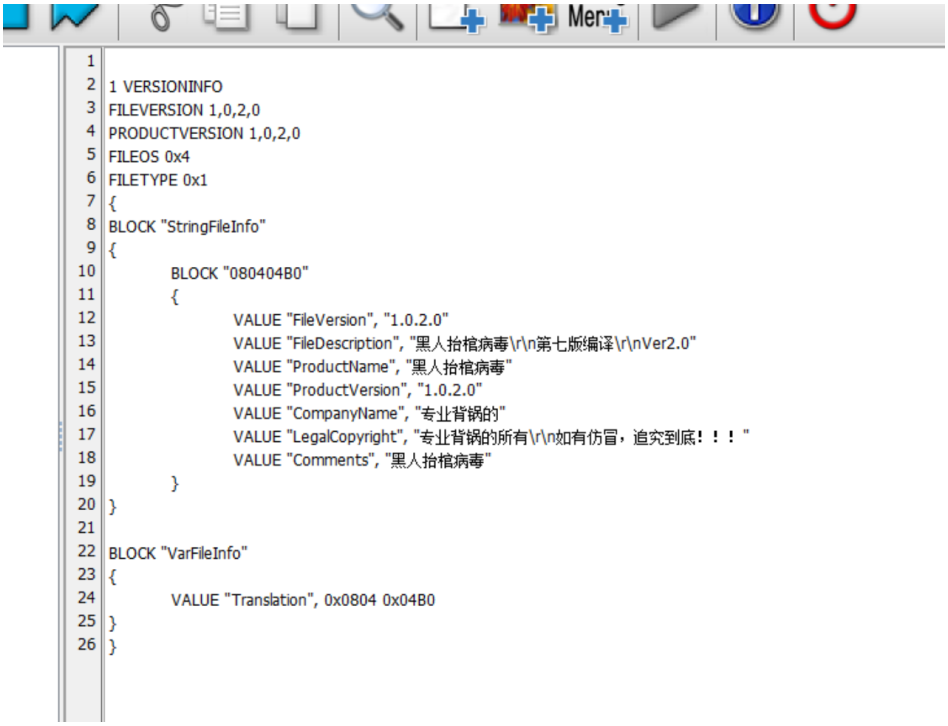
PE 头与分节。

pFile	Data	Description	Value
00000240	2E 72 64 61	Name	.rdata
00000244	74 61 00 00		
00000248	026FC0A8	Virtual Size	
0000024C	000B8000	RVA	
00000250	026FD000	Size of Raw Data	
00000254	000B8000	Pointer to Raw Data	
00000258	00000000	Pointer to Relocations	
0000025C	00000000	Pointer to Line Numbers	
00000260	0000	Number of Relocations	
00000262	0000	Number of Line Numbers	
00000264	40000040	Characteristics	
		00000040	IMAGE_SCN_CNT_INITIALIZED_DATA
		40000000	IMAGE_SCN_MEM_READ

图 57: PEview 中.rdata 节的 IMAGE_SECTION_HEADER，虚拟大小 0x026FC0A8

节数为 4，Subsystem 为 GUI。值得注意的是体积分布：.text 只有 747 KB，.data 450 KB，.rsrc 25 KB，而.rdata 一个节就有 40,878,248 字节，占整个文件的 97.9%，也就是说这只程序本质上是”一小段代码加一大块静态数据”。四个节的虚拟大小与原始大小都相称，没有留出解压空间；最后一节.rsrc 的原始偏移加原始大小恰好等于文件大小 41,771,008 字节，说明文件末尾没有追加数据。编译时间戳是 2020/11/27 04:43:13 UTC，与链接时间、文件修改时间三者一致，没有被改过。有一处异常是.text 的权限是 E0000020，除可执行和可读之外还带写权限，而正常编译出来的代码段是只读可执行的。

导入函数与资源节。Dependency Walker 显示静态导入 12 个 DLL：KERNEL32、ADVAPI32、COMCTL32、COMDLG32、GDI32、OLE32、OLEAUT32、SHELL32、USER32、WINMM、WINSPOOL.DRV 和 WS2_32。WS2_32 直接出现在导入表里，说明它的网络能力在静态层面就是可见的，不需要运行时动态加载；COMCTL32 那一组、加上 WINMM 和 WINSPOOL.DRV，再配上完整的一套菜单、对话框、图标和字符串表资源，外观上就是一个正常的 Windows 桌面程序。资源里最说明问题的是作者自己留下的版本信息。



```
1 VERSIONINFO
FILEVERSION 1,0,2,0
PRODUCTVERSION 1,0,2,0
FILEOS 0x4
FILETYPE 0x1
{
BLOCK "StringFileInfo"
{
        BLOCK "080404B0"
        {
                VALUE "FileVersion", "1.0.2.0"
                VALUE "FileDescription", "黑人抬棺病毒\r\n第七版编译\r\nVer2.0"
                VALUE "ProductName", "黑人抬棺病毒"
                VALUE "ProductVersion", "1.0.2.0"
                VALUE "CompanyName", "专业背锅的"
                VALUE "LegalCopyright", "专业背锅的所有\r\n如有仿冒，追究到底！！！"
                VALUE "Comments", "黑人抬棺病毒"
        }
}

BLOCK "VarFileInfo"
{
        VALUE "Translation", 0x0804 0x04B0
}
}
```

图 58: Resource Hacker 中 2-hrtg.exe 的 Version Info

FileDescription 写着”黑人抬棺病毒第七版编译 Ver2.0”，ProductName 是”黑人抬棺病毒”，CompanyName 是”专业背锅的”，版权声明是”专业背锅的所有，如有仿冒，追究到底!!! ”。样本名 hrtg 正是”黑人抬棺”的拼音首字母。作者在版本资源里就把自己的作品标注成了恶搞程序。

2. 用 Windows 系统自带的防病毒软件 MicroSoft Defender AntiVirus 扫描样本，给出检测结果截图

用 Windows 安全中心里的 Microsoft Defender Antivirus 对样本做自定义扫描得到结果。

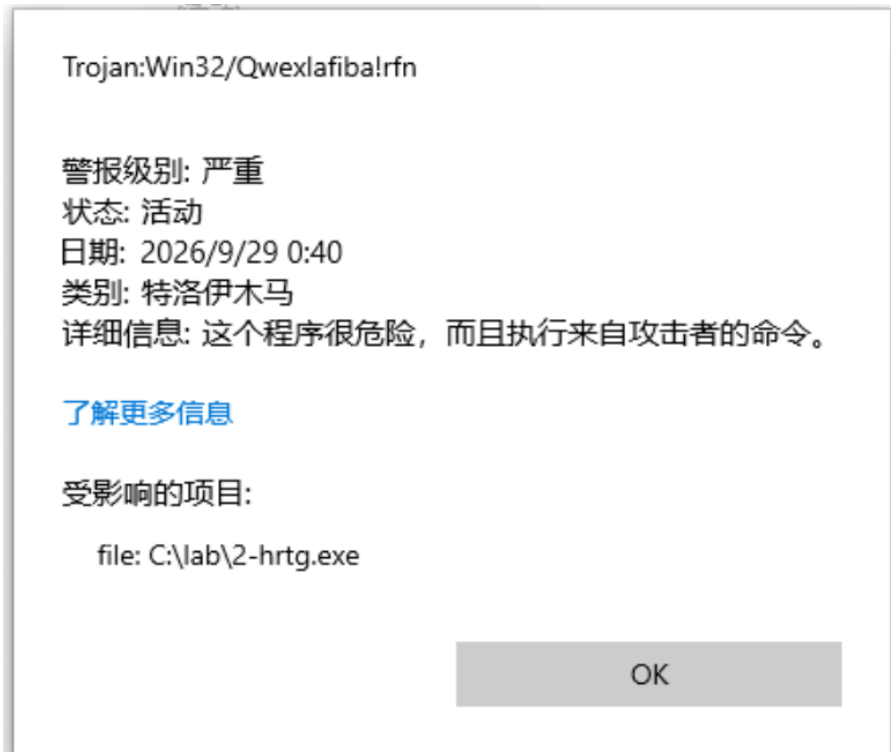


图 59: Microsoft Defender Antivirus 对 2-hrtg.exe 的威胁详情，威胁名为 Trojan:Win32/Qwexlafiba!rfn

3. 通过 VirusTotal、国家计算机病毒协同分析平台扫描样本，给出检测结果，包括但不限于：SSDeep、TLSH 等哈希值、PE 文件头的信息、PE 文件的函数信息

一、VirusTotal

检出结果是 43/71。

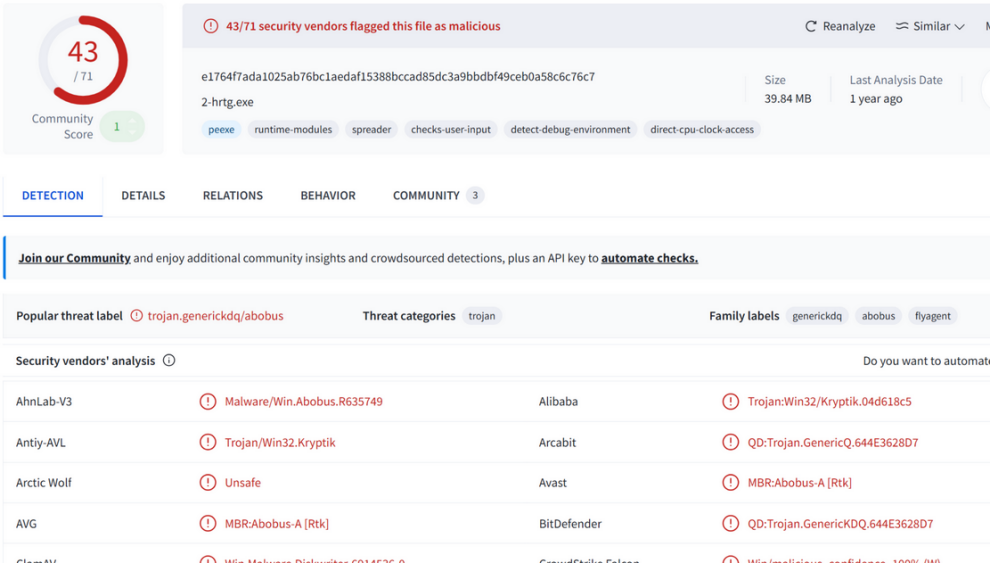


图 60: VirusTotal 对 2-hrtg.exe 的检出结果，43/71 家引擎报恶意

VT 聚合出的流行威胁名是 trojan.generickdq/abobus，Avast 和 AVG 报出的威胁名是 MBR:Abobus-A [Rtk]，即修改主引导记录的风险工具。

哈希值。VT 给出的完整哈希信息如下。

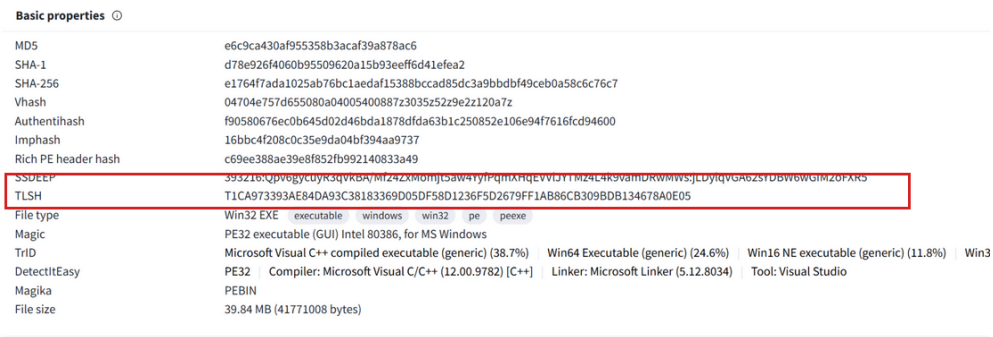


图 61: VirusTotal 给出的哈希值，红框内为 SSDeep 与 TLSH

PE 文件头信息。

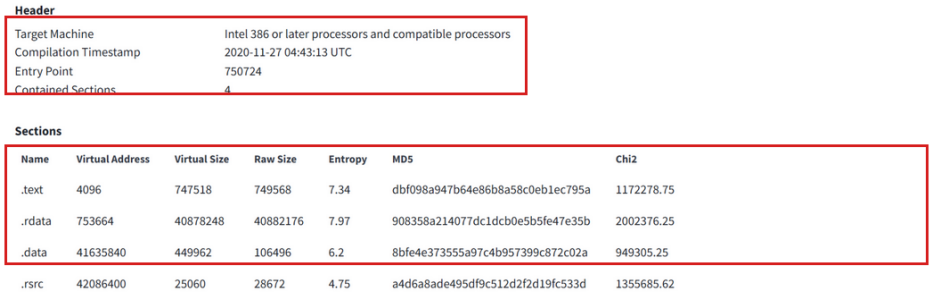


图 62: VirusTotal 中 2-hrtg.exe 的 PE 头与节表

PE 文件的函数信息。

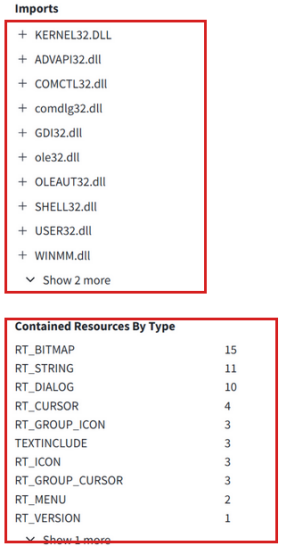


图 63: VirusTotal 中 2-hrtg.exe 的导入表与资源类型

导入表里共 12 个 DLL。同一段的 Contained Resources By Type 显示资源以 RT_BITMAP、RT_STRING 和 RT_DIALOG 为主，语言是简体中文。

二、国家计算机病毒协同分析平台

平台的检出结果是 16/29。

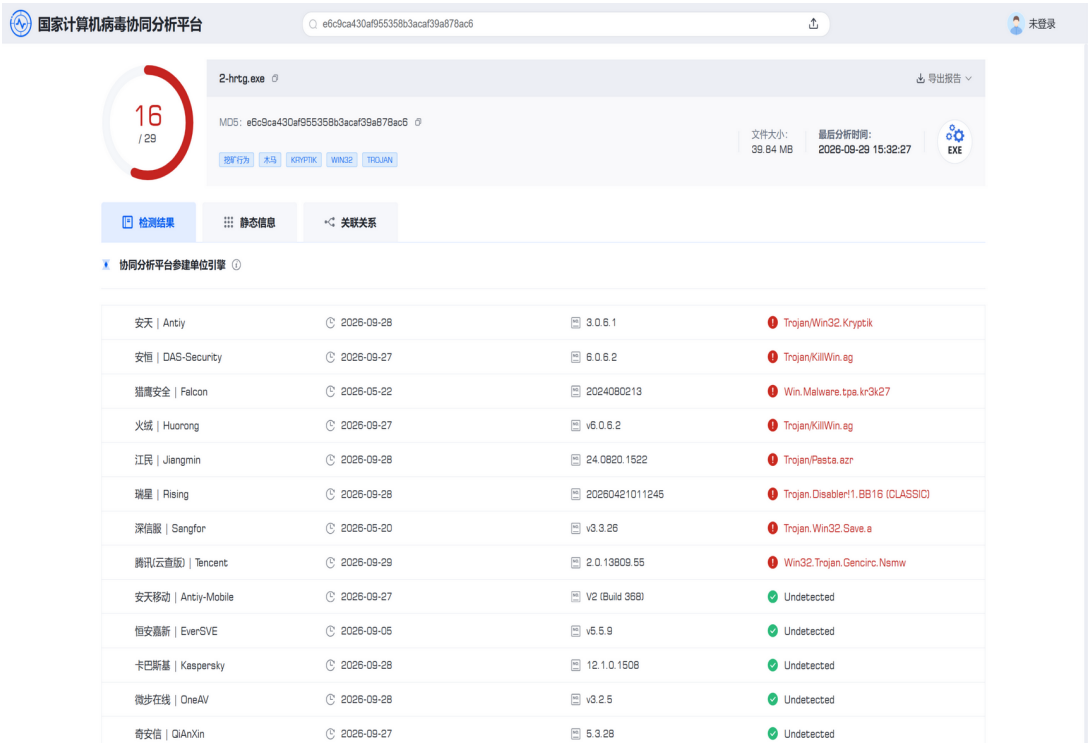


图 64: 国家计算机病毒协同分析平台对 2-hrtg.exe 的检出结果，16/29

平台的检出标签是挖矿行为、木马、KRYPTIK、WIN32、TROJAN。29 个引擎中未检出的包括卡巴斯基、奇安信、腾讯、微步在线和 360。

哈希值。

基本属性信息

FileName	2-hrtg.exe
FileSize	39.84 MB
FileFormat	BinExecute/Microsoft.EXE[:X86]
MD5	e6c9ca430af955358b3acaf39e878ac6
SHA-1	d78e926f4060b95509620a15b93eeff6d41efea2
SHA-256	e1764f7ada1025ab76bc1aedaf15388bccad85dc3a9bbdbf49cebOa58c6c76c7
TLSH	T1CA973393AEB4DA93C38183369D05DF58D1236F5D2679FF1AB86CB309BDB13467BA0B05
SSDEEP	393216:Qpv6gycuyR3qVkBAMf24ZxMomjt5awvXyfRqmXHqFVVLlXTMx4L4k9uamDRwMWe:jLDuhVGA62sYDBW6wGIM2oFXR5
ImpHash	16BBC4F208C0C35E9DA04BF394AA9737
AuthentiHash	f90580676ec0b645d02d46bda1878dfda63b1c250852e106e94f7616fcd94600
RichHeaderHash	c69ee388ae39e8f852fb992140833a49

图 65: 国家计算机病毒协同分析平台给出的哈希值，红框内为 TLSH 与 SSDEEP

PE 格式信息。

头信息

目标平台码	Intel 386 or later processors and compatible processors
时间戳	2020-11-27 04:43:13
PE入口点	0xb7484

节信息

Name	VirtualAddress	VirtualSize	RawAddress	RawSize	SectionMD5	Character	Entropy
.text	0x1000	0xb67fe	0x1000	0xb7000	dbf098a947b64e86b8a58c0eb1ec795a	r-x-	7.34
.rdata	0xb8000	0x26fc0a8	0xb8000	0x26fd000	908358a214077dc1dcb0e5b5fe47e35b	-w--	7.97
.data	0x27b5000	0x6ddaa	0x27b5000	0x1a000	8bfe4e373555a97c4b957399c872c02a	-w--	6.20
.rsrc	0x2823000	0x61e4	0x27cf000	0x7000	a4d6a8ade495df9c512d2f2d19fc533d	-w--	4.75

导入表

图 66: 国家计算机病毒协同分析平台的 PE 头信息与节信息，红框分别标出头部字段和节表

导入表

- KERNEL32.DLL
- ADVAPI32.dll
- COMCTL32.dll
- comdlg32.dll
- GDI32.dll
- ole32.dll
- OLEAUT32.dll
- SHELL32.dll
- USER32.dll
- WINMM.dll
- WINSPOOL.DRV
- WS2_32.dll

资源类型

TEXTINCLUDE	3
RT_CURSOR	4
RT_BITMAP	15
RT_ICON	3
RT_MENU	2
RT_DIALOG	10
RT_STRING	11
RT_GROUP_CURSOR	3
RT_GROUP_ICON	3
RT_VERSION	1

资源节信息

RsrcOffset	RsrcSize	RsrcSHA256	RsrcRootName	Entropy
0x027cfbf4	0x0000000b	7791d56b6792eb1836f2e46ea02d208ffb1343ae922d6c696ad04de2e2c09441	TEXTINCLUDE	3.10
0x027cfc00	0x00000016	4b345f7ee4c62582d946797d37a7dd8498af15f4afda496999d1ac54c108714a	TEXTINCLUDE	4.28
0x027cfc18	0x00000151	6b993c9dc325863302d0ea34e4287822286f7fdf8e023320b9a520b5c9dcfa47	TEXTINCLUDE	5.25
0x027cfd6c	0x00000134	d27dd42f6f0db9536d3d59a42c61f4049f73c3373c00afa0554e3e7f85f9a92b	RT_CURSOR	3.17
0x027cfea0	0x00000134	ba8a8cf5ecd16690d37ba7f41f107973656c61748Ba880538e1e33821929d3a3	RT_CURSOR	2.61

图 67: 国家计算机病毒协同分析平台的导入表、资源类型与资源节信息

导入表同样列出 12 个 DLL。

(3) BUG32

1. 使用本章描述的工具和技术做基本静态分析

通过 GnuWin32 的 file.exe 报出类型为 PE32 executable (GUI) Intel 80386 32-bit。这只样本 3,192,320 字节，PEiD 没有匹配到任何签名，但导入表完整、资源段占了全文件的 98%，综合判定未加壳、未混淆。它是一只恐吓型的恶搞程序，把整套恶搞工具和提示音都打包在资源里。

查壳。PEiD 的签名框显示 Nothing found，本身说明不了问题。判据来自另外三处：EP Section 是.code、First Bytes 是 68,D4,00,00；Dependency Walker 解析出了 9 个 DLL 的完整导入函数表，分别是 MSVCRT、KERNEL32、USER32、GDI32、COMCTL32、OLE32、SHELL32、WINMM 和 SHLWAPI；五个节里.rsrc 的原始大小是 3,133,440 字节。加壳文件的导入表会被压缩到只剩 LoadLibraryA 和 GetProcAddress 两项，这里不存在这种情况，所以判为未加壳、未混淆。另外 Linker Info 是 2.50，比常见 MSVC 程序低得多，配合代码段名为.code，这两个特征指向 Delphi 或 Borland 系的工具链。

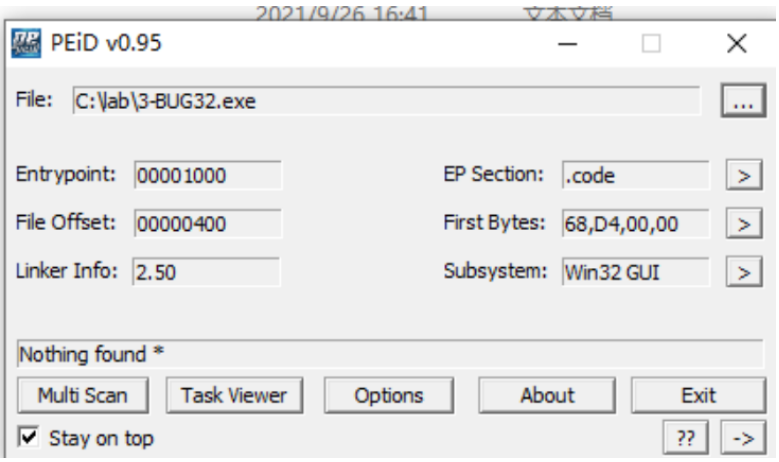


图 68: PEiD 加载 3-BUG32.exe 的检测结果，未匹配到签名

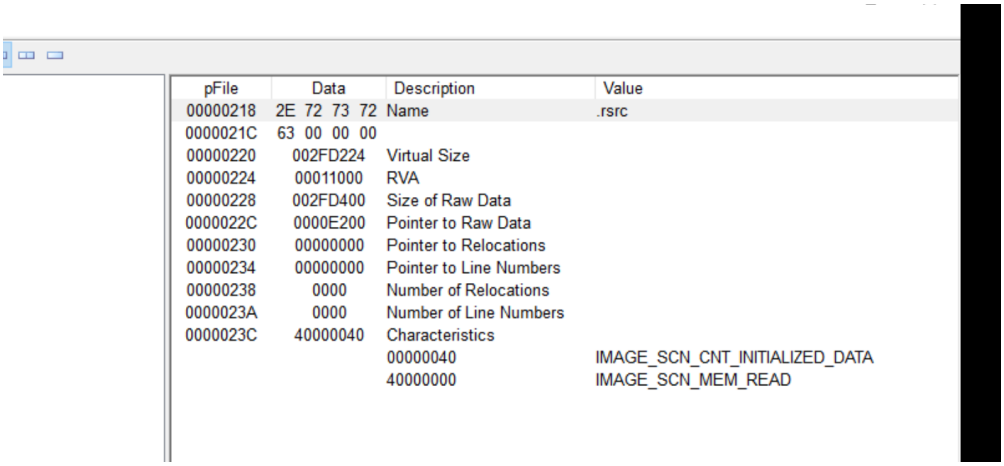
字符串。用 strings 以最小长度 6 提取出 14488 行，最有特征价值的是这几条。

```
msgbox"Umm.. Well. Your Computer Has Been Infected by Bugscare.exe
    ",1+16,"aferim"
buhu="C:\bug32\jaq.vbs"
C:\BUG32\Kill.bat
```

```
x.RegWrite"HKCU\...\Policies\System\DisableRegistrytools","1","
    REG_DWORD"
taskkill /f /im eset.exe
C:\BUG32\scrm.wav
```

这几条把它的行为交代得比较清楚：弹一个”你的电脑已被 Bugscare.exe 感染”的提示框；往 C:\BUG32\释放 jaq.vbs、Kill.bat、jsc.exe 和几个提示音；用 wscript 以管理员身份运行脚本；通过写入 HKCU 的策略键禁用 Windows 键、禁止更换壁纸、并禁用注册表编辑器；还用 taskkill 结束一批进程，除了浏览器、记事本和画图，还包括 eset.exe 杀毒软件。整份输出里筛 http、www、ftp 一条都没有，导入表里也没有 WS2_32，说明它没有任何联网行为。

PE 头与分节。



pFile	Data	Description	Value
00000218	2E 72 73 72	Name	.rsrc
0000021C	63 00 00 00		
00000220	002FD224	Virtual Size	
00000224	00011000	RVA	
00000228	002FD400	Size of Raw Data	
0000022C	0000E200	Pointer to Raw Data	
00000230	00000000	Pointer to Relocations	
00000234	00000000	Pointer to Line Numbers	
00000238	0000	Number of Relocations	
0000023A	0000	Number of Line Numbers	
0000023C	40000040	Characteristics	
		00000040	IMAGE_SCN_CNT_INITIALIZED_DATA
		40000000	IMAGE_SCN_MEM_READ

图 69: PEview 中.rsrc 节的 IMAGE_SECTION_HEADER，原始大小 0x2FD400

节数为 5，Subsystem 为 GUI。体积分布是这只样本的关键：第一个节.code 只有 8,704 字节，.text 是 40,448 字节，.rdata 和.data 分别只有 2,560 和 5,120 字节，而最后的.rsrc 原始大小是 3,133,440 字节，占整个文件的 98%，也就是说程序的实体全在资源段里。编译时间戳是 2016/05/04 07:30:05 UTC，与 Dependency Walker 读到的链接时间 2016/05/04 15:30 一致，没有被改过。它也没有版本资源，Dependency Walker 的 File Ver 和 Product Ver 都显示 N/A。

导入函数与资源节。

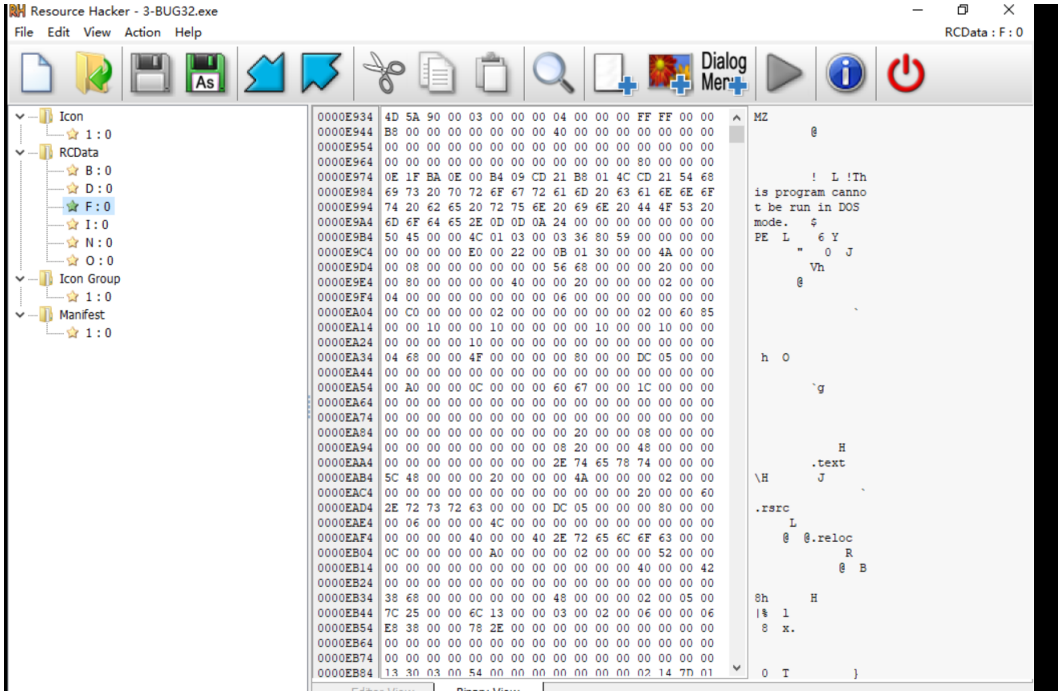


图 70: 用 Resource Hacker 展开 RCDATA 资源，十六进制开头为 4D 5A

资源树里只有 Icon、Icon Group、Manifest 和 RCDATA 四类，其中 RCDATA 下面有六项，名字是单个字母 B、D、F、I、N、O。展开后开头两个字节是 4D 5A，即 MZ 头，再往下能看到 50 45 00 00 和.text、.rsrc、.reloc 三个节名，说明这六个资源里装的是可执行文件。字符串里的线索进一步交代了它们的来历：出现了.NET Framework 4.5.2、mscoree.dll 和 System.Drawing，以及两条开发者机器上的调试路径。

```
C:\Users\Guest\documents\visual studio 2015\Projects\Noways\Noways\
    obj\Debug\Noways.pdb
c:\users\guest\documents\visual studio 2015\Projects\msaccessdenied\
    msaccessdenied\obj\Debug\msaccessdenied.pdb
```

内嵌的是用 Visual Studio 2015 编译的.NET 4.5.2 程序，其中至少包含 Noways.exe 和 msaccessdenied.exe 两个恶搞组件。

2. 用 Windows 系统自带的防病毒软件 MicroSoft Defender AntiVirus 扫描样本，给出检测结果截图

用 Windows 安全中心里的 Microsoft Defender Antivirus 对样本做自定义扫描得到结果。

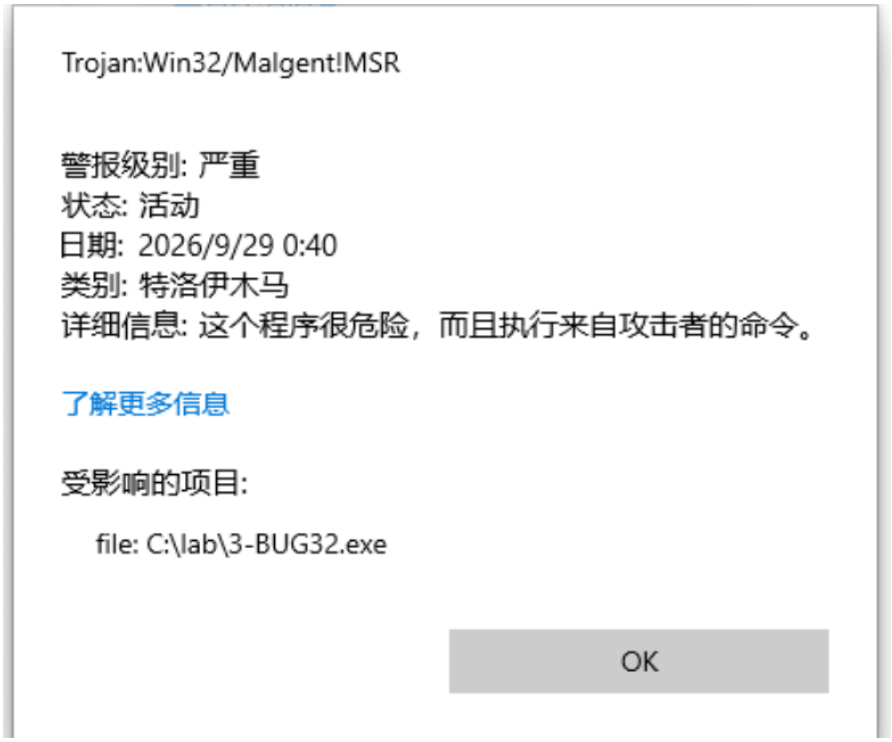


图 71: Microsoft Defender Antivirus 对 3-BUG32.exe 的威胁详情，威胁名为 Trojan:Win32/Malgent!MSR

3. 通过 VirusTotal、国家计算机病毒协同分析平台扫描样本，给出检测结果，包括但不限于：SSDeep、TLSH 等哈希值、PE 文件头的信息、PE 文件的函数信息

一、VirusTotal

检出结果是 64/71。

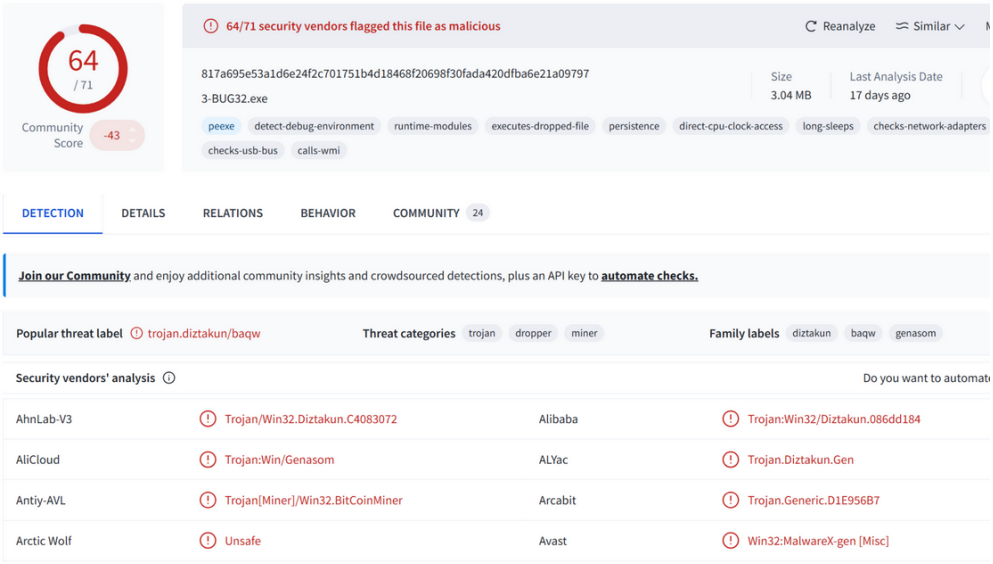


图 72: VirusTotal 对 3-BUG32.exe 的检出结果，64/71 家引擎报恶意

VT 聚合出的流行威胁名是 trojan.diztakun/baqw，威胁类别有 trojan、dropper 和 miner 三类。

哈希值。VT 给出的完整哈希信息如下。

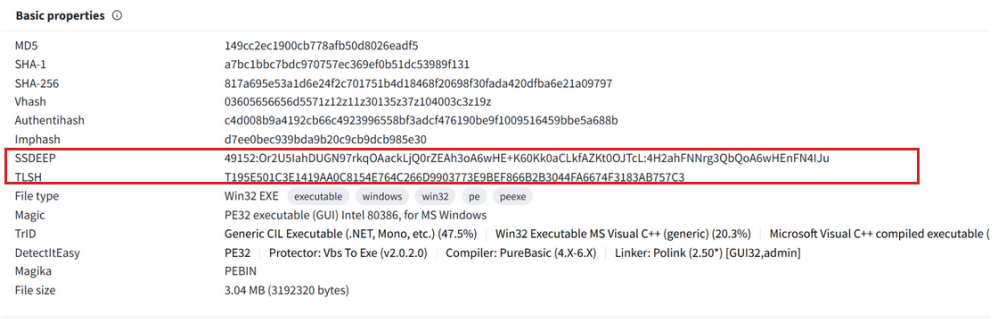


图 73: VirusTotal 给出的哈希值，红框内为 SSDeep 与 TLSH

DetectItEasy 报出的 Protector 是 Vbs To Exe (v2.0.2.0)，Compiler 是 PureBasic (4.X-6.X)，Linker 是 Polink (2.50*)，说明这只文件是用 VBS 转 EXE 的工具封装出来的。

PE 文件头信息。

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2016-05-04 07:30:05 UTC
Entry Point	4096
Contained Sections	5

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.code	4096	8391	8704	5.44	494ff811ab595efaec1a575a2c43cc17	168241.84
.text	16384	39962	40448	6.57	0c28d3ebfbb61b593d041a5b3bd2e932	198264.03
.rdata	57344	2430	2560	6.61	facc1a97edd0b9fbad36add1494b300f	21032.22
.data	61440	6020	5120	4.8	1dd4e42d4fa3e61136779d0126dddb62	163019.33
.rsrc	69632	3133988	3134464	7.25	2a7f249ab3c2e7f88a627e8d2ee9da0a	9379698

图 74: VirusTotal 中 3-BUG32.exe 的 PE 头与节表

PE 文件的函数信息。

Imports

+ MSVCRT.dll
+ KERNEL32.dll
+ USER32.DLL
+ GDI32.DLL
+ COMCTL32.DLL
+ OLE32.DLL
+ SHELL32.DLL
+ WINMM.DLL
+ SHLWAPI.DLL

Contained Resources By Type

RT_RCDATA	6
RT_GROUP_ICON	1
RT_MANIFEST	1
RT_ICON	1

Contained Resources By Language

NEUTRAL	9

Contained Resources

图 75: VirusTotal 中 3-BUG32.exe 的导入表与资源类型

导入表里共 9 个 DLL。

二、国家计算机病毒协同分析平台

平台的检出结果是 15/18。

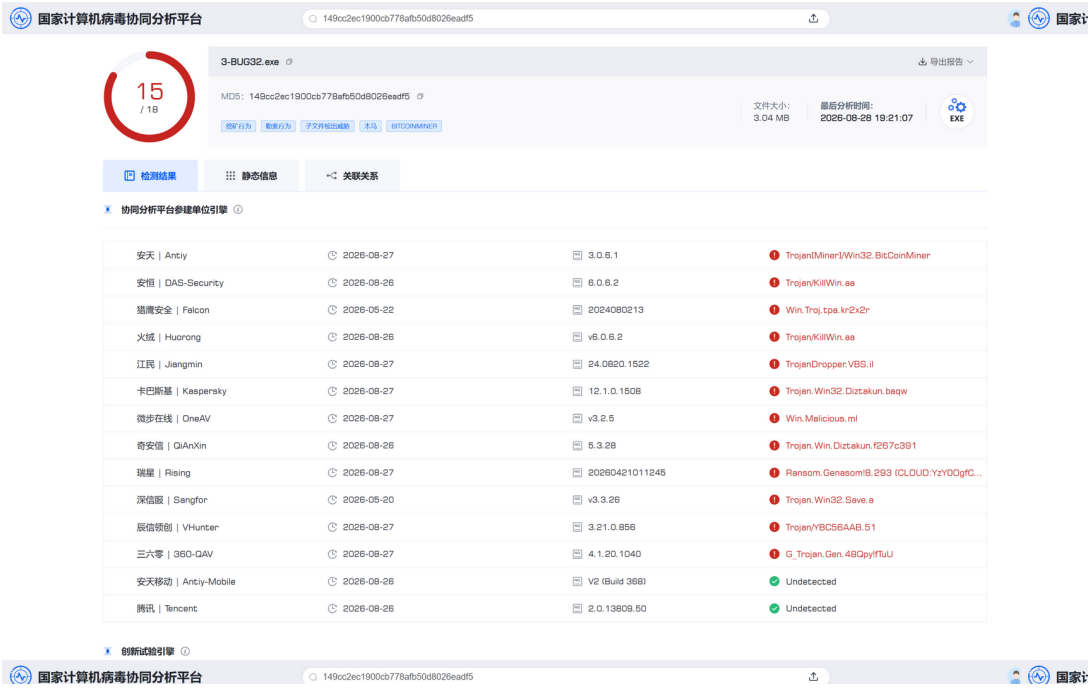


图 76: 国家计算机病毒协同分析平台对 3-BUG32.exe 的检出结果，15/18

平台的检出标签是挖矿行为、勒索行为、子文件检出威胁、木马、BITCOIN-MINER，其中”子文件检出威胁”说明平台也认定内嵌的子文件有问题。未检出的引擎包括安天移动和腾讯。

哈希值。

图 77: 国家计算机病毒协同分析平台给出的哈希值，红框内为 TLSH 与 SSDEEP

PE 格式信息。

头信息

目标平台码	Intel 386 or later processors and compatible processors
时间戳	2016-05-04 07:30:05
PE入口点	0x1000

节信息

Name	VirtualAddress	VirtualSize	RawAddress	RawSize	SectionMD5	Character	Entropy
.code	0x1000	0x20c7	0x400	0x2200	494ff811ab595efaec1a575a2c43cc17	r-x-	5.44
.text	0x4000	0x9c1a	0x2600	0x9e00	0c28d3ebfbb61b593d041a5b3bd2e932	r-x-	6.57
.rdata	0xe000	0x97e	0xc400	0xa00	facc1a97eddOb9fbad36add1494b300f	-w--	6.61
.data	0xf000	0x1784	0xce00	0x1400	1dd4e42d4fa3e61136779d0126dddb62	-w--	4.80
.rsrc	0x11000	0x2fd224	0xe200	0x2fd400	2a7f249ab3c2e7f88a627e8d2ee9da0a	-w--	7.25

导入表

[redacted]

图 78: 国家计算机病毒协同分析平台的 PE 头信息与节信息，红框分别标出头部字段和节表

平台给出完整的五个节。

导入表

- MSVCRT.dll
- KERNEL32.dll
- USER32.DLL
- GDI32.DLL
- COMCTL32.DLL
- OLE32.DLL
- SHELL32.DLL
- WINMM.DLL
- SHLWAPI.DLL

资源类型

RT_ICON	1
RT_RCDATA	6
RT_GROUP_ICON	1

资源节信息

RsrcOffset	RsrcSize	RsrcSHA256	RsrcRootName	Entropy
0x0000e438	0x000002e8	124c17c752fede6a53874a52a6eafcbd4ec346b470eb99be62990a35fdfde17a	RT_ICON	3.37
0x0000e720	0x00000201	91fb5cad72f9c4259ad14cb5ddcb0a575bad021207b8dfbf1980c5e74754927e	RT_RCDATA	5.28
0x0000e924	0x0000000e	c2d31570188dd35cd7dc1ee6c9e79a313165f8b079edf13d4b2e3471cd20cd16	RT_RCDATA	3.47
0x0000e934	0x002fc6f1	057ac25bc581ce429fdffc0a160fba5ca2baaad3901121c576cbb286cdc5bccd	RT_RCDATA	7.25
0x0030b028	0x0000013c	bc966c60d2498e1cd83edc9e7c4fddf81c257bb76d9c7a1e63823b9d2781502e	RT_RCDATA	4.85

图 79: 国家计算机病毒协同分析平台的导入表、资源类型与资源节信息

导入表同样列出 9 个 DLL。资源节信息里体积最大的一项位于偏移 0x0000e934，类型是 RT_RCDATA，正是 VT 识别为 DOS EXE 的那一个内嵌文件。

(4) yhzll

1. 使用本章描述的工具和技术做基本静态分析

通过 GnuWin32 的 file.exe 报出类型为 PE32 executable (GUI) Intel 80386 32-bit。这只样本 2,527,232 字节，PEiD 报出的是编译器签名而不是壳名，导入表完整、四个节的原始大小加起来正好等于文件大小，综合判定未加壳、未混淆。它是一个用易语言编写的离线多媒体恶搞程序。

查壳。PEiD 的签名框给出 Microsoft Visual C++ 6.0，判为未加壳、未混淆。需要说明的是签名框报的 Microsoft Visual C++ 6.0 并不代表作者真的用 VC6

写程序——版本资源里写着”本程序使用易语言编写”，易语言的运行库本身就是用 VC6 编译的，编译器签名来自运行库。

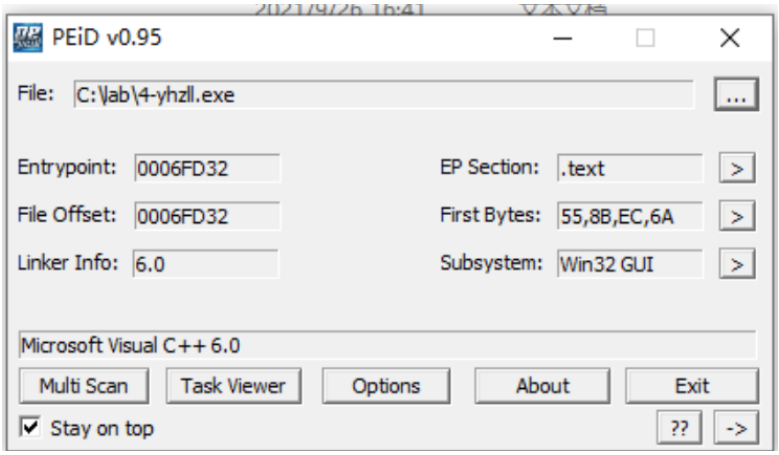


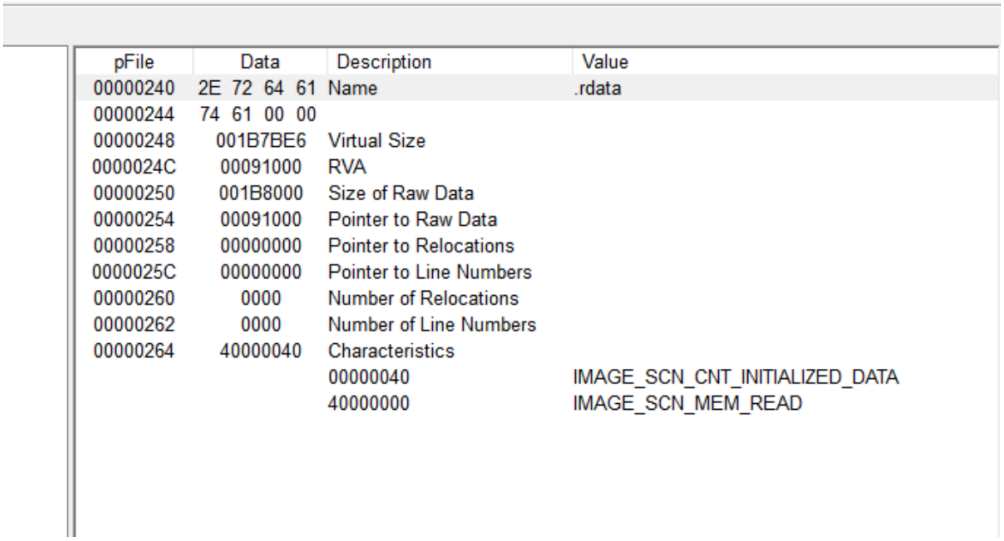
图 80: PEiD 加载 4-yhzll.exe 的检测结果，签名为编译器而非壳

字符串。用 strings 以最小长度 6 提取出 9992 行，关键线索如下。

```
(http://www.eyuyan.com)
(*.WAV)|*.WAV|MIDI
Application built with libpng-
FindWindowA
SetCursorPos
SystemParametersInfoA
```

整个文件里唯一的 URL 就是易语言的官网域名，来自版本资源的注释字段；第三行说明它静态链接了图像处理库；文件对话框的过滤器只列了 WAV 和 MIDI，说明音频是它的主要功能之一；最后一组是窗口查找、鼠标位置和系统参数相关的 API，配合资源里的 WAVE 音频资源，指向一只会摆弄窗口和鼠标的恶搞程序。字符串里既没有注册表自启动项，也没有命令执行痕迹。

PE 头与分节。



pFile	Data	Description	Value
00000240	2E 72 64 61	Name	.rdata
00000244	74 61 00 00		
00000248	001B7BE6	Virtual Size	
0000024C	00091000	RVA	
00000250	001B8000	Size of Raw Data	
00000254	00091000	Pointer to Raw Data	
00000258	00000000	Pointer to Relocations	
0000025C	00000000	Pointer to Line Numbers	
00000260	0000	Number of Relocations	
00000262	0000	Number of Line Numbers	
00000264	40000040	Characteristics	
		00000040	IMAGE_SCN_CNT_INITIALIZED_DATA
		40000000	IMAGE_SCN_MEM_READ

图 81: PEview 中.rdata 节的 IMAGE_SECTION_HEADER，原始大小 0x1B8000

节数为 4，Subsystem 为 GUI。体积的 71% 集中在.rdata 节，这与易语言程序把运行库和数据都放在只读段的特征相符。编译时间戳是 2017/09/23 04:50:27 UTC，与链接时间和文件修改时间三者一致，没有被改过。

导入函数与资源节。

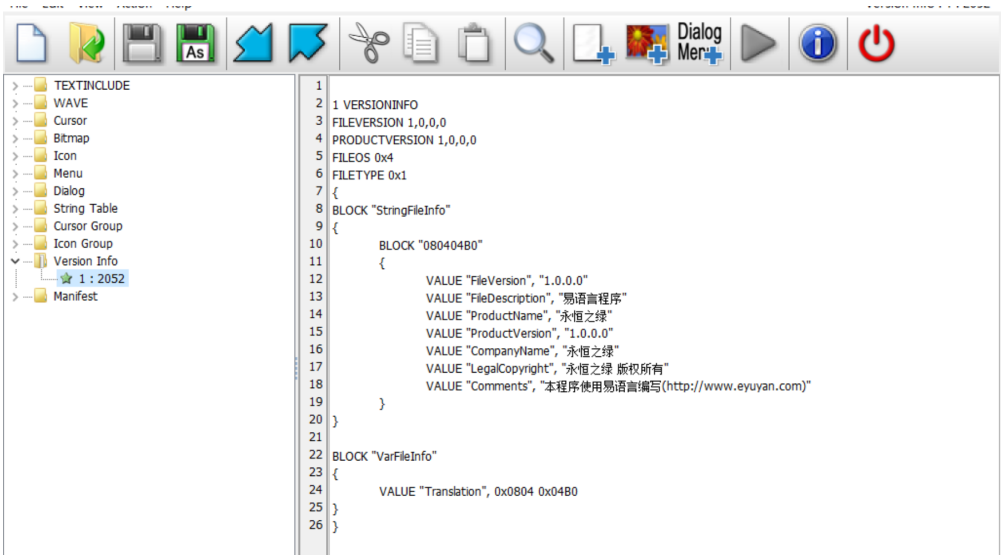


图 82: Resource Hacker 中 4-yhzll.exe 的 Version Info

Dependency Walker 显示静态导入 14 个 DLL、共 431 个函数，样本自身没有导出函数。关键的是三组：WINMM 的 18 个函数全是音频，包括 PlaySoundA 和整套 midiStream、midiOut、waveOut 调用；AVIFIL32 的 AVIStreamGetFrame、AVIStreamInfoA 配合 MSVFW32 的 DrawDibDraw 负责解码和绘制 AVI 视频

帧；KERNEL32 里既有 CreateProcessA 和 WinExec，也有 OpenProcess、CreateToolhelp32Snapshot 和 Process32First/Next，能枚举进程并启动程序；USER32 里则有 FindWindowA、FindWindowExA、SetCursorPos、SetWindowPos 和 SystemParametersInfoA，负责找窗口、移动鼠标和改系统参数。这几组能力合起来，正是一只多媒体恶搞程序需要的全部动作。

其中 WS2_32 的 9 个函数全部按序号导入，在导入表里看不到函数名。结合整个文件里没有任何网络地址，这 9 个函数应当来自易语言运行库的网络支持，而不是程序自己要通信。

资源树是完整的一套图形界面程序资源，其中 WAVE 就是音频资源。版本信息自报家门：FileDescription 写着”易语言程序”，ProductName 和 CompanyName 都是”永恒之绿”，版权声明是”永恒之绿版权所有”，注释写着”本程序使用易语言编写”。样本名 yhzll 正是”永恒之绿”的拼音首字母，而”永恒之绿”显然是拿”永恒之蓝”做的谐音梗。

2. 用 Windows 系统自带的防病毒软件 MicroSoft Defender AntiVirus 扫描样本，给出检测结果截图

用 Windows 安全中心里的 Microsoft Defender Antivirus 对样本做自定义扫描得到结果。

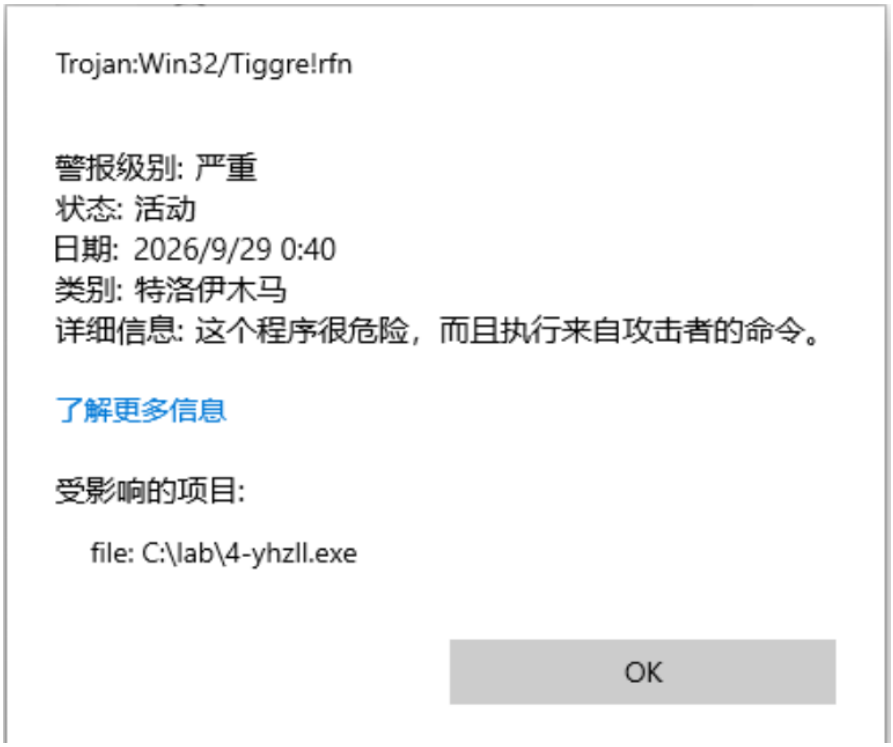


图 83: Microsoft Defender Antivirus 对 4-yhzll.exe 的威胁详情，威胁名为 Trojan:Win32/Tiggre!rfn

3. 通过 VirusTotal、国家计算机病毒协同分析平台扫描样本，给出检测结果，包括但不限于：SSDeep、TLSH 等哈希值、PE 文件头的信息、PE 文件的函数信息

一、VirusTotal

检出结果是 60/72。

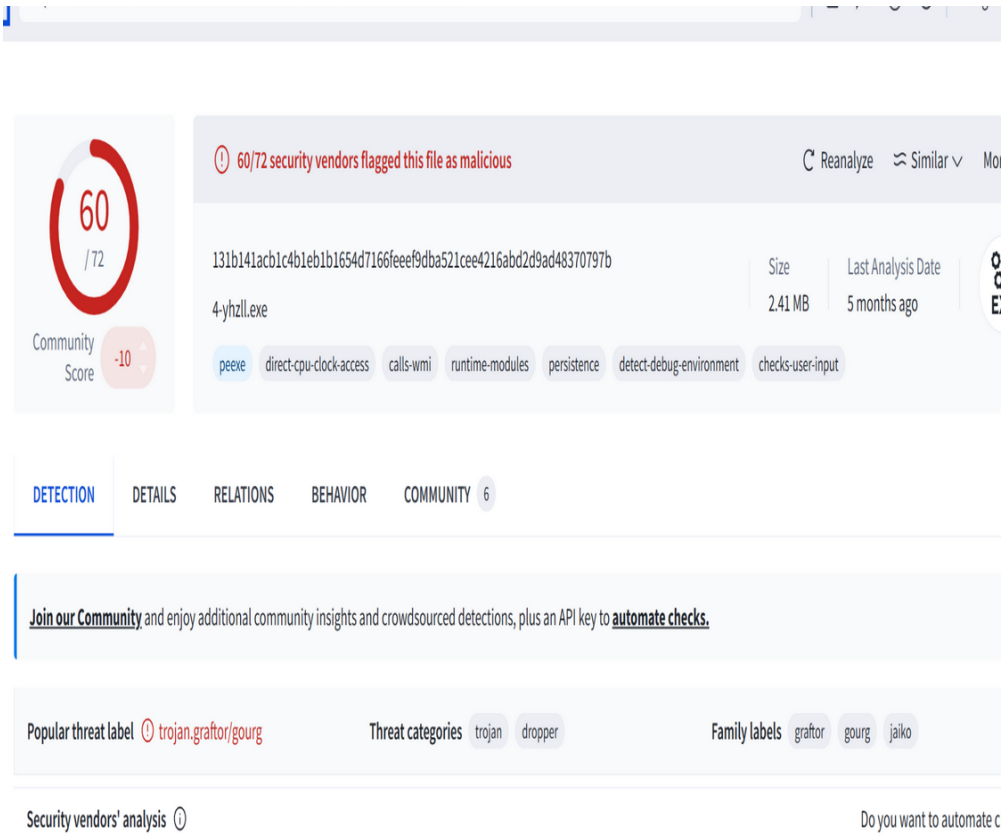


图 84: VirusTotal 对 4-yhzll.exe 的检出结果，60/72 家引擎报恶意

VT 聚合出的流行威胁名是 trojan.graftor/gourg，威胁类别有 trojan 和 dropper 两类，家族标签有 graftor、gourg 和 jaiko。厂商命名里值得注意的是火绒报出的 Joke/StartleBox.a，属于恶搞类程序。

哈希值。VT 给出的完整哈希信息如下。

图 85: VirusTotal 给出的哈希值，红框内为 SSDeep 与 TLSH

PE 文件头信息。

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2017-09-23 04:50:27 UTC
Entry Point	458034
Contained Sections	4

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.text	4096	585822	589824	6.54	5de6c4c4092f0b8627df0dd8ce63fdeb	3796023.75
.rdata	593920	1801190	1802240	7.32	2b514fb1795a6ca4a3d2aa9365cbf921	8362053
.data	2396160	310026	98304	5.36	d83edc01dae5a15c66d68ca014dc72d0	3918125.25
.rsrc	2707456	31088	32768	5.5	1a46158ecb63da8f463b0d0a326a8339	888010

图 86: VirusTotal 中 4-yhzll.exe 的 PE 头与节表

Show 6 more

Signature info

Signature Verification

File is not signed

File Version Information

图 87: VirusTotal 中 4-yhzll.exe 的版本信息与数字签名

VT 读出的版本信息与资源段里的完全一致，另外它判定该文件未签名。

二、国家计算机病毒协同分析平台

平台的检出结果是 12/28。

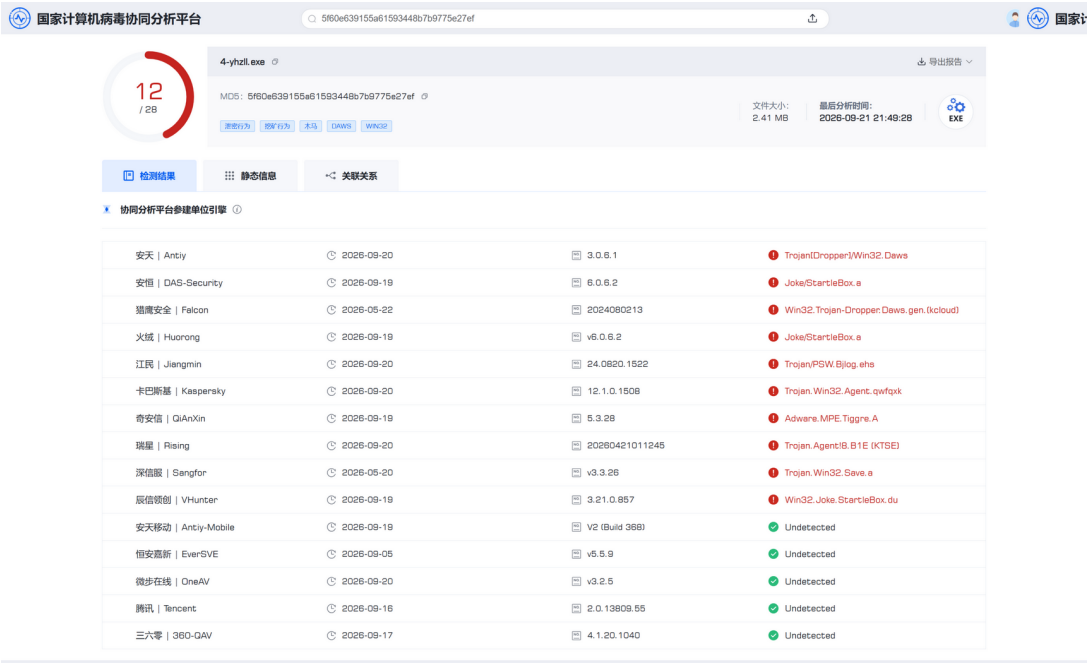


图 88: 国家计算机病毒协同分析平台对 4-yhzll.exe 的检出结果，12/28

平台的检出标签是泄密行为、挖矿行为、木马、DAWS、WIN32。安恒和火绒都报 Joke/StartleBox.a，辰信领创报 Win32.Joke.StartleBox.du，三家把它归入恶搞类；奇安信报 Adware.MPE.Tiggre.A，与 Defender 的 Trojan:Win32/Tiggre!rfn 同族。

哈希值。

图 89: 国家计算机病毒协同分析平台给出的哈希值，红框内为 TLSH 与 SSDEEP

PE 格式信息。

头信息

目标平台码	Intel 386 or later processors and compatible processors
时间戳	2017-09-23 04:50:27
PE入口点	0x8fd5E

节信息

Name	VirtualAddress	VirtualSize	RawAddress	RawSize	SectionMD5	Character	Entropy
.text	0x1000	0x8f05e	0x1000	0x90000	5de6c4c4092f0b8627df0dd8ce63fdeb	r-x-	6.54
.rdata	0x91000	0x1b7be6	0x91000	0x1b8000	2b514fb1795a6ca4a3d2aa9365cbf921	-w--	7.32
.data	0x249000	0x4bb0a	0x249000	0x18000	d83edc01dae5a15c66d68ca014dc72d0	-w--	5.36
.rsrc	0x295000	0x7970	0x261000	0x8000	1a46158ecb63da8f463b0d0a326a8339	-w--	5.50

导入表

图 90: 国家计算机病毒协同分析平台的 PE 头信息与节信息，红框分别标出头部字段和节表

导入表

- MSVFW32.dll
- AVIFIL32.dll
- WINMM.dll
- WS2_32.dll
- KERNEL32.dll
- USER32.dll
- GDI32.dll
- WINSPOOL.DRV
- comdlg32.dll
- ADVAPI32.dll
- SHELL32.dll
- ole32.dll
- OLEAUT32.dll
- COMCTL32.dll

资源类型

TEXTINCLUDE	3
WAVE	1
RT_CURSOR	6
RT_BITMAP	15
RT_ICON	3
RT_MENU	2
RT_DIALOG	10
RT_STRING	11
RT_GROUP_CURSOR	5
RT_GROUP_ICON	3
RT_VERSION	1

资源节信息

RsrcOffset	RsrcSize	RsrcSHA256	RsrcRootName	Entropy
0x00261d9c	0x0000000b	7791d56b6792eb1836f2e46ea02d208ffb1343ae922d6c696ad04de2e2c09441	TEXTINCLUDE	3.10
0x00261da8	0x00000016	4b345f7ee4c62582d946797d37a7dd8498af15f4afda496999d1ac54c108714a	TEXTINCLUDE	4.28
0x00261dc0	0x00000151	6b993c9dc325863302d0ea34e4287822286f7fdf8e023320b9a520b5c9dcfa47	TEXTINCLUDE	5.25
0x00261f14	0x00001448	47595617a47452a1438f15e053321e5d40e13b8c7a70b34bd3dc7814dd5233f0	WAVE	6.35
0x0026336c	0x00000184	d27dd4f6f6fdb9665d6d53e4f6f1f4646f76c6679e66efe6554c6e7f65f6e6f6	RT_CURSOR	3.17

图 91: 国家计算机病毒协同分析平台的导入表、资源类型与资源节信息

导入表列出全部 14 个 DLL。资源类型里有 1 项 WAVE，资源节信息显示它位于偏移 0x0028f114，正是那个音频资源。

(5) BossDaMajor

1. 使用本章描述的工具和技术做基本静态分析

通过 GnuWin32 的 file.exe 报出类型为 PE32 executable (GUI) Intel 80386 32-bit。PEiD 没有匹配到签名，但导入表完整、资源段里是明文脚本，综合判定未加壳、未混淆。它是一只用 VBS 脚本封装起来的恶搞程序，会把脚本和组件释放到 C:\Program Files\mrsmajor\，然后替换系统里常见文件类型的图标。

查壳。PEiD 的签名框显示 Nothing found。加壳文件的导入表会被压缩到只剩 LoadLibraryA 和 GetProcAddress 两项，这里不存在这种情况，所以判为未

加壳、未混淆。Linker Info 是 2.50，代码段名为.code，这两个都不是 MSVC 的常见取值，说明作者用的不是 Visual Studio 那套工具链。

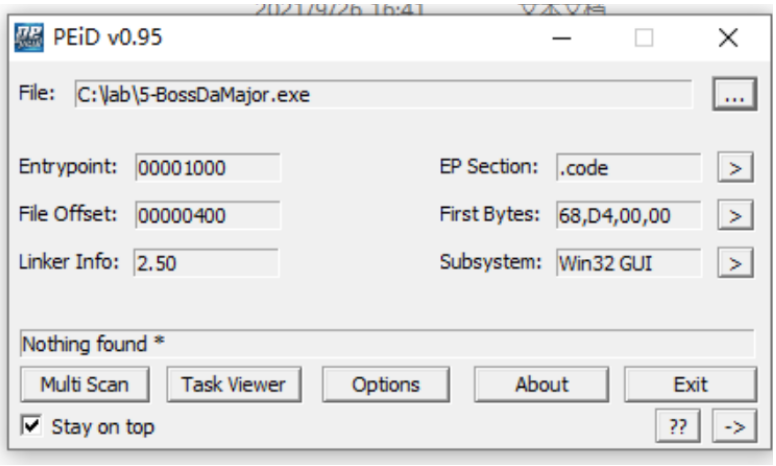


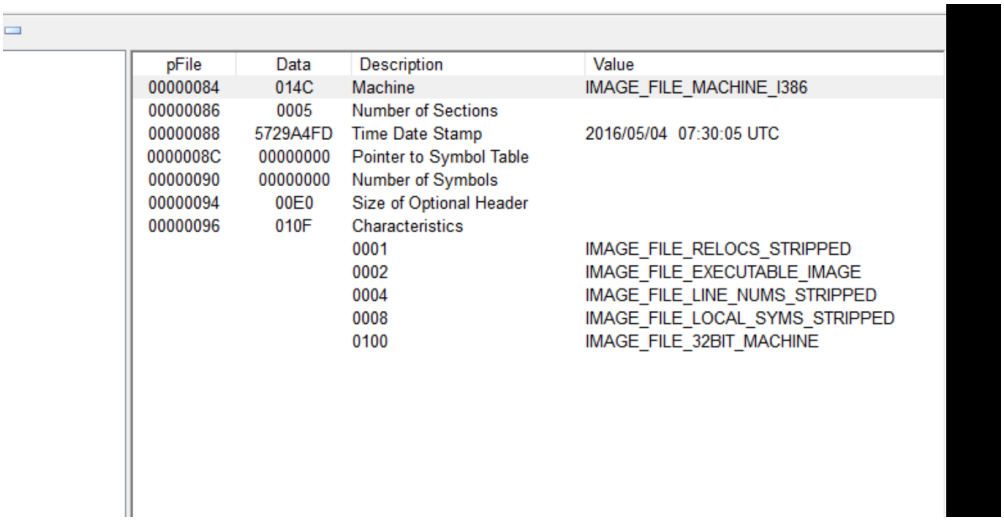
图 92: PEiD 加载 5-BossDaMajor.exe 的检测结果，未匹配到签名

字符串。用 strings 以最小长度 6 提取出 6501 行，最有特征价值的是这几条。

```
buhu="C:\Program files\mrsmajor\mrsmajorlauncher.vbs"
ObjShell.ShellExecute "wscript.exe", """" & buhu & """
    RunAsAdministrator", , "runas", 1
msgbox"MrsMajor Wants TO MEET YOU!",1+48,"Cute Doll!"
x.RegWrite"HKCR\exefile\DefaultIcon\",""&ke&"\Icon_resource\SkullIco.
    ico","REG_SZ"
c.CopyFile""&ke&"\def_resource\@Tile@@.jpg", ""&tmp&"\"
taskkill /f /im dllhost.exe
```

这几条把它的行为交代得比较清楚：把 VBS 脚本释放到 C:\Program Files\mrsmajor\，再用 wscript 以管理员身份运行；弹一个”MrsMajor Wants TO MEET YOU!” 的恶搞提示框；改写 HKCR 下 exefile、inifile、mp3file、mp4file、txtfile 这几个类型的 DefaultIcon，把它们统统换成自己携带的 SkullIco.ico，也就是一个骷髅图标；把一张 @Tile@@.jpg 复制到临时目录；还用 taskkill 结束一批进程，除了浏览器和画图，还包括 dllhost.exe。整份输出里没有任何 URL，也没有注册表自启动项。

PE 头与分节。



pFile	Data	Description	Value
00000084	014C	Machine	IMAGE_FILE_MACHINE_I386
00000086	0005	Number of Sections	
00000088	5729A4FD	Time Date Stamp	2016/05/04 07:30:05 UTC
0000008C	00000000	Pointer to Symbol Table	
00000090	00000000	Number of Symbols	
00000094	00E0	Size of Optional Header	
00000096	010F	Characteristics	
		0001	IMAGE_FILE_RELOCS_STRIPPED
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0004	IMAGE_FILE_LINE_NUMS_STRIPPED
		0008	IMAGE_FILE_LOCAL_SYMS_STRIPPED
		0100	IMAGE_FILE_32BIT_MACHINE

图 93: PEview 中 5-BossDaMajor.exe 的 IMAGE_FILE_HEADER

节数为 5，Subsystem 为 GUI，编译时间戳是 2016/05/04 07:30:05 UTC，与 Dependency Walker 读到的链接时间一致，没有被改过。Characteristics 是 010F，其中包含 RELOCS_STRIPPED，说明重定位表被剥掉了。

导入函数与资源节。Dependency Walker 显示静态导入 9 个 DLL：MSVCRT、KERNEL32、USER32、GDI32、COMCTL32、OLE32、SHELL32、WINMM 和 SHLWAPI，样本自身没有导出函数。

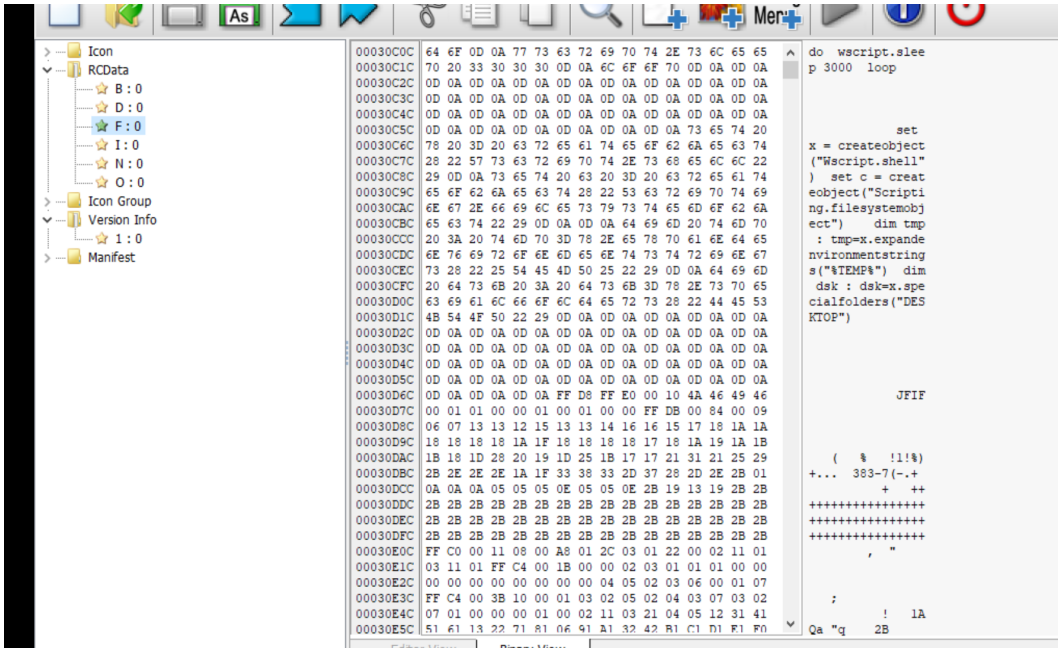


图 94: 用 Resource Hacker 展开 RCDATA 资源，十六进制里能看到 VBS 源码

资源树只有 Icon、RCDATA、Icon Group、Version Info 和 Manifest 五类，其中 RCDATA 下面六项的名字是单个字母 B、D、F、I、N、O。展开 F 这一

项，右侧十六进制里能直接读出 VBS 源码，能够说明这段资源里同时打包了脚本和一张 JPEG 图片。脚本和图形资源都以明文躺在资源段里，是"VBS 转 EXE" 这类封装工具的典型产物。版本信息里 ProductName 写着 Mrsmajor.exe，InternalName 是 mrsmajoruserinterface.exe。

2. 用 Windows 系统自带的防病毒软件 MicroSoft Defender AntiVirus 扫描样本，给出检测结果截图

用 Windows 安全中心里的 Microsoft Defender Antivirus 对样本做自定义扫描得到结果。

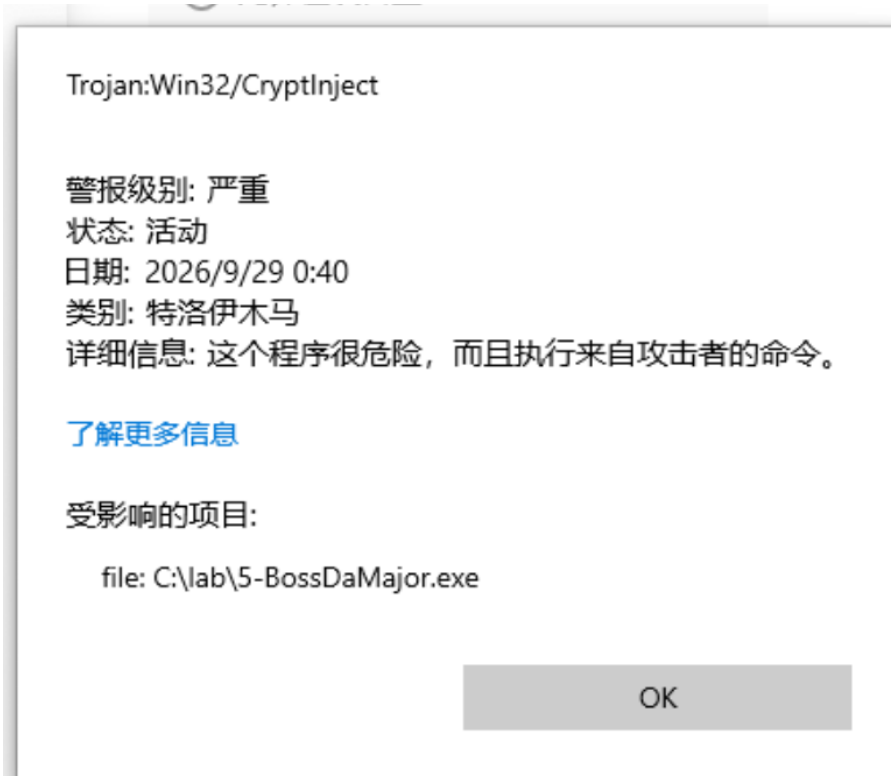


图 95: Microsoft Defender Antivirus 对 5-BossDaMajor.exe 的威胁详情，威胁名为 Trojan:Win32/CryptInject

3. 通过 VirusTotal、国家计算机病毒协同分析平台扫描样本，给出检测结果，包括但不限于：SSDeep、TLSH 等哈希值、PE 文件头的信息、PE 文件的函数信息

一、VirusTotal

检出结果是 62/71。

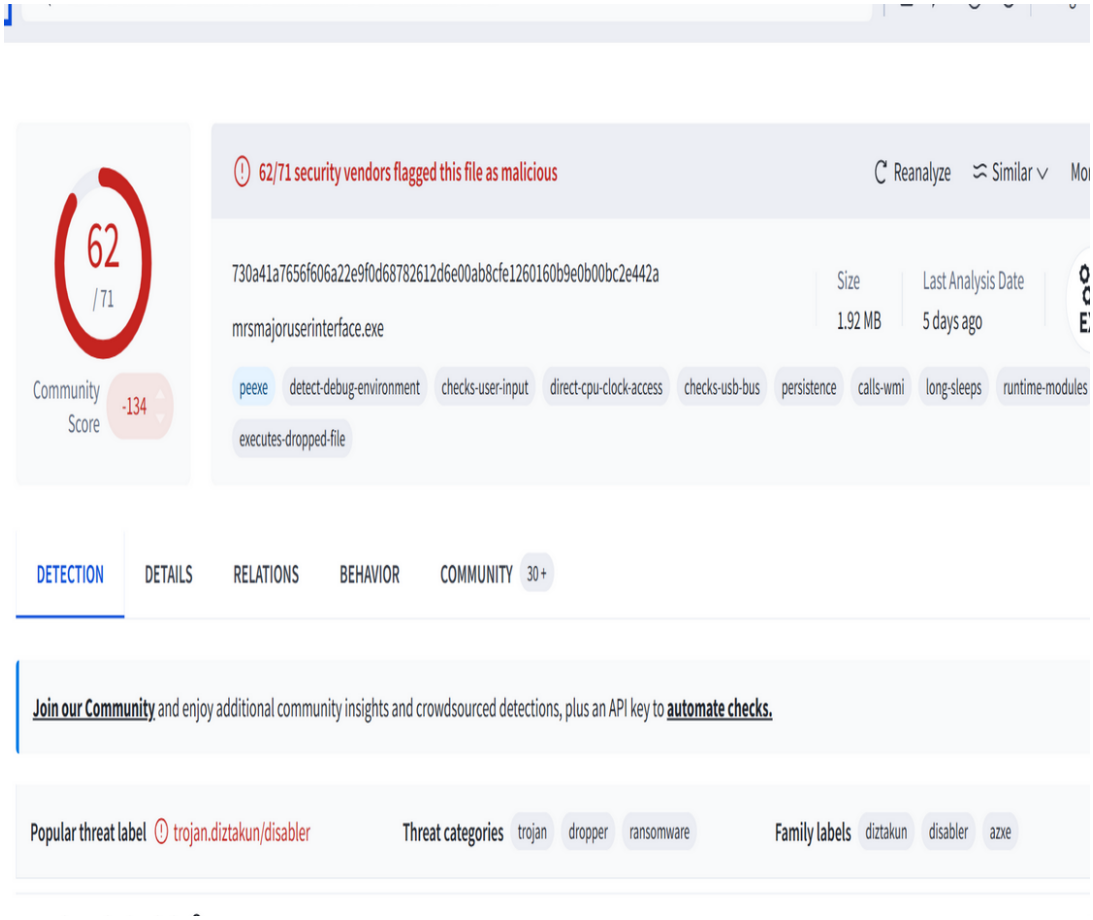


图 96: VirusTotal 对 5-BossDaMajor.exe 的检出结果，62/71 家引擎报恶意

VT 聚合出的流行威胁名是 trojan.diztakun/disabler，威胁类别有 trojan、dropper 和 ransomware 三类。

哈希值。VT 给出的完整哈希信息如下。

Basic properties

MD5	38ff71c1dee2a9add67f1edb1a30ff8c
SHA-1	10f0defd98d4e5096fbeb321b28d6559e44d66db
SHA-256	730a41a7656f606a22e9f0d68782612d6e00ab8cfe1260160b9e0b00bc2e442a
Vhash	02605656656d5571z12z11z30135z37z104003c3z19z
Authentihash	2514c3dd2ab1bb1e0550005631e476cd0fe3de50682265542676833d1fbf1e8a2
Imphash	d7ee0bec939bda9b20c9cb9dcb985e30
SSDEEP	49152:veG3J7FtM9SbJakTiTBMGSARaspyyx979PSxgKFdGlYU:2GZxSoJrTiTBMGtRa8t7EFddU
TLSH	T1E0952336F7697951D14202B148A26BA0AA304C36E7205CDB7AE57B0FFC722FD563326C
File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32 executable (GUI) Intel 80386, for MS Windows
TrID	Generic CIL Executable (.NET, Mono, etc.) (47.5%) Win32 Executable MS Visual C++ (generic) (20.3%) Microsoft Visual C++ compiled executable (g
DetectItEasy	PE32 Protector: Vbs To Exe (v2.0.2.0) Compiler: PureBasic (4.X-6.X) Linker: Polink (2.50*) [GUI32,admin]

图 97: VirusTotal 给出的哈希值，红框内为 SSDeep 与 TLSH

PE 文件头信息。

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2016-05-04 07:30:05 UTC
Entry Point	4096
Contained Sections	5

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Chi2
.code	4096	8391	8704	5.44	494ff811ab595efaec1a575a2c43cc17	168241.84
.text	16384	39962	40448	6.57	0c28d3ebfbb61b593d041a5b3bd2e932	198264.03
.rdata	57344	2430	2560	6.61	facc1a97edd0b9fbad36add1494b300f	21032.22
.data	61440	6020	5120	4.8	1dd4e42d4fa3e61136779d0126dddb62	163019.33
.rsrc	69632	1955932	1956352	7.82	e93155fa57047f3c916a54216df9c201	1636212.5

图 98: VirusTotal 中 5-BossDaMajor.exe 的 PE 头与节表

Signature Verification

⚠ File is not signed

File Version Information

Product	Mrsmajor.exe
Internal Name	mrsmajoruserinterface.exe
File Version	1.0.0.0

图 99: VirusTotal 中 5-BossDaMajor.exe 的版本信息与数字签名

VT 读出的版本信息与资源段里的一致，另外它判定该文件未签名。另外，这个样本的 Imphash 与 3-BUG32.exe 完全相同，前四个节的虚拟大小、原始大小和熵值也逐项一致，编译时间戳同为 2016-05-04 07:30:05 UTC，可以判定两者出自同一个封装工具链。

二、国家计算机病毒协同分析平台

平台的检出结果是 20/28。

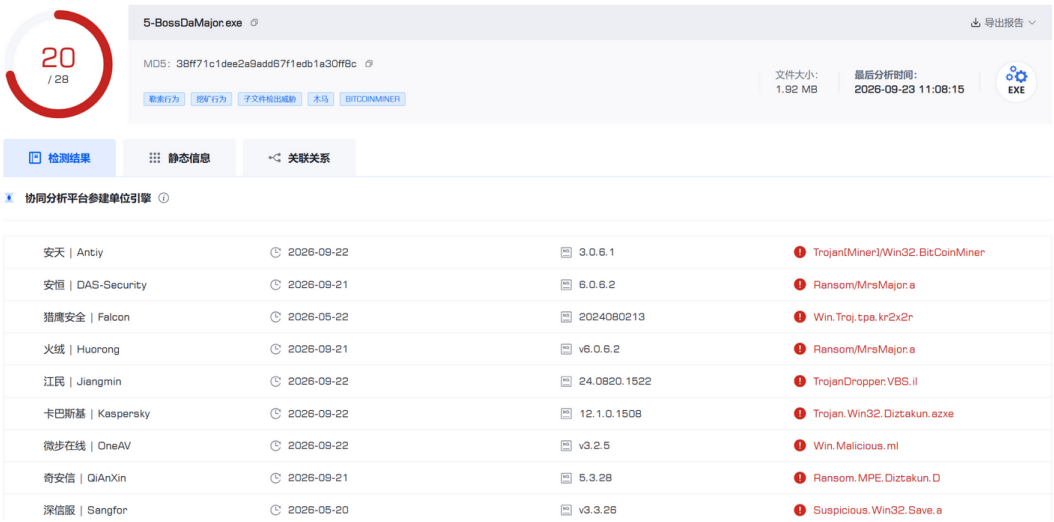


图 100: 国家计算机病毒协同分析平台对 5-BossDaMajor.exe 的检出结果，20/28

平台的检出标签是勒索行为、挖矿行为、子文件检出威胁、木马、BITCOIN-MINER。

哈希值。

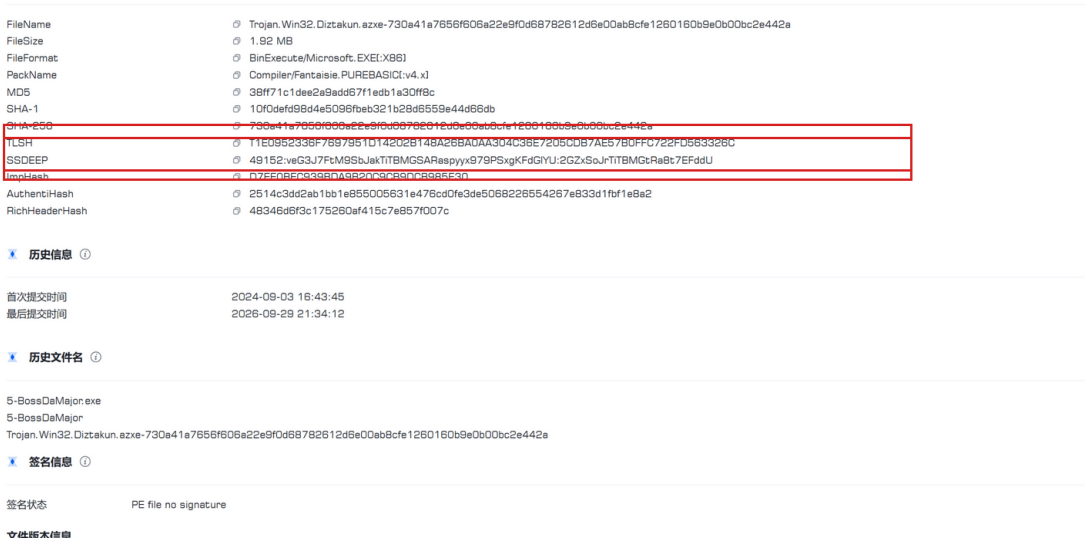


图 101: 国家计算机病毒协同分析平台给出的哈希值，红框内为 TLSH 与 SSDEEP

PE 格式信息。

PE格式信息

调试信息

资源目录大小	0x1dd85c
资源目录偏移	0xe200

头信息

目标平台码	Intel 386 or later processors and compatible processors
时间戳	2016-05-04 07:30:05
PE入口点	0x1000

节信息

Name	VirtualAddress	VirtualSize	RawAddress	RawSize	SectionMD5	Character	Entropy
.code	0x1000	0x20c7	0x400	0x2200	494ff811ab595efaec1a575a2c43cc17	r-x-	5.44
.text	0x4000	0x9c1a	0x2600	0x9e00	0c28d3ebfbb61b593d041a5b3bd2e932	r-x-	6.57
.rdata	0xe000	0x97e	0xc400	0xe00	facc1a97edd0b9fbad36add1494b300f	-w--	6.61
.data	0xf000	0x1784	0xce00	0x1400	1dd4e42d4fa3e61136779d0126dddb62	-w--	4.80
.rsrc	0x11000	0x1dd85c	0xe200	0x1dda00	e93155fa57047f3c916a54216df9c201	-w--	7.82

图 102: 国家计算机病毒协同分析平台的 PE 头信息与节信息，红框分别标出头部字段和节表

导入表

- MSVCRT.dll
- KERNEL32.dll
- USER32.DLL
- GDI32.DLL
- COMCTL32.DLL
- OLE32.DLL
- SHELL32.DLL
- WINMM.DLL
- SHLWAPI.DLL

资源类型

RT_ICON	12
RT_RCDATA	6
RT_GROUP_ICON	1
RT_VERSION	1

资源节信息

RsrcOffset	RsrcSize	RsrcSHA256	RsrcRootName	Entropy
0x0000e690	0x00004b61	b50f6a1b6a81039b6ff6dd951c7c63286d299903ac8e8064f5b4d76f94cdff94	RT_ICON	7.93
0x000131f4	0x00000668	ee50a3da009696284d6b32a32db7ca47e7eb54103e924b3c0e2c6e5bb5681b47	RT_ICON	3.47
0x0001385c	0x000002e8	055d7b0a25c6634a81fc610c774177b585578f965e6302f76ba453a977f20a4e	RT_ICON	3.70
0x00013b44	0x00000128	b61df98ee2c1f612be03d99ad92f116d38008fab14f10f0d0f83305d019c42f4	RT_ICON	3.19
0x00013c6c	0x0000acd6	95ccb2ab51b7a0faaee5e85da8adcfd8e0b1ac45bbe6276cc675d698b60c4827	RT_ICON	7.98

图 103: 国家计算机病毒协同分析平台的导入表、资源类型与资源节信息

导入表列出全部 9 个 DLL。

第三部分：使用 VirusTotal 提供的 API，针对给出的恶意代码样本及 SHA256 哈希值开发批量检测程序并形成检测报告

检测对象 1：90 个真实恶意代码样本。

检测对象 2：每位同学分配 100 条恶意代码的 SHA256 哈希值。

(1) 批量检测程序的开发

批量检测程序是一个独立的 Python 3 脚本（`query_reports.py`），借助大模型辅助编写。分工上，大模型负责代码实现和两个平台接口的分析，我负责确定需求、准备输入、实际运行，并逐项核对输出结果。程序只依赖`requests` 一个第三方库，计算 SHA256 用的是 Python 标准库，因此在联网的宿主机和断网的虚拟机里都能直接运行。程序主要实现以下功能：

输入

程序支持两种输入方式。一是哈希清单（`--hashes`），一行一个 SHA256，行尾可以带备注，`#` 开头的行是注释，空行和重复行自动忽略。二是样本目录（`--samples`），程序自己递归遍历目录、对每个文件分块读取计算 SHA256，并按内容去重，改过名的副本自动合并成一条。本次作业的两个检测对象正好各对应一种：检测对象 1 用样本目录，检测对象 2 用分配到的哈希清单。程序不关心样本数量，清单里有几行就查几个。

查询

程序对接两个平台的 API。VirusTotal 用的是官方 v3 接口 `GET https://www.virustotal.com/api/v3/files/{hash}`，鉴权靠请求头 `x-apikey`，取回的是与网页版内容一致的完整 JSON 报告。协同平台用的是它的开放接口，`apikey` 放在请求体里。API key 可以写在配置文件里，也可以用命令行参数或环境变量传入。由于 VirusTotal 公开 API 限制每分钟 4 次、每天 500 次，程序默认把两次请求的间隔控制在 16 秒（约 3.8 次每分钟），触发 429 限流时会读取响应头里的 `Retry-After` 自动等待，并按指数退避重试。程序支持断点续传：已经下载过报告的哈希直接跳过，中途按 Ctrl+C、断网或请求超时都不会丢进度，重新运行会接着查没查完的部分，而且中断时会先把已有报告整理成汇总表和检测报告再退出。

输出

每个样本的原始 JSON 报告按平台分目录存放；一张 `summary.csv` 汇总全部字段，包括检出比、家族标签、MD5/SHA1、SSDeep 与 TLSH 模糊哈希、首次提交时间等；此外还有一份可直接阅读的 Markdown 检测报告 `detection_report.md`。报告只保留概览性的短表——总体情况、检出率分档、恶意家族、文件类型、检出率最高的样本、未收录样本，控制在两页以内，完整明细放在 `summary.csv`。

异常与边界

程序专门处理了几类工程问题。平台对没有权限的接口会返回「当前账户未开通此 api 权限」，程序识别后直接禁用该接口，不会在后面每个样本上反复重试；协同平台对未收录的样本返回的是 `code=0` 加上一个空的 `data`，程序据此判断为「未收录」，而不会误报成「已收录」；不管成功还是失败，接口的原始响应

都会存盘，事后可以打开看平台到底返回了什么。另有一个可选开关 `--upload`，能把平台库里还没有的样本提交给平台去分析，默认关闭——提交样本是不可逆的对外披露行为，必须显式开启。

程序的主要参数如下。

```
usage: query_reports.py [-h] [--hashes HASHES] [--samples SAMPLES]
    [--list-only] [--with-name] [--upload]
    [--hash ONE_HASH] [--out OUT] [--only {vt,cverc,both}]
    [--vt-interval VT_INTERVAL]
    [--cverc-endpoints CVERC_ENDPOINTS] [--limit LIMIT]
    [--force] [--summary-only] [--label LABEL]
    [--note NOTE] [--report-detail]
```

(2) 检测过程

两个检测对象分两条路径完成。检测对象 1 的样本是课程发放的压缩包，按课程要求只在隔离虚拟机里解压、取哈希。检测对象 2 是老师直接分配的 100 条哈希值，本身不含可执行代码，直接在宿主机上查询即可。

检测对象 1：在虚拟机里取哈希

在虚拟机里把样本压缩包解压到工作目录，然后让程序只算哈希：

```
python query_reports.py --samples <dir> --list-only --with-name
```

程序递归遍历目录、对每个文件分块读取计算 SHA256，并按内容去重，最后把结果写成一份哈希清单，共 106 条。

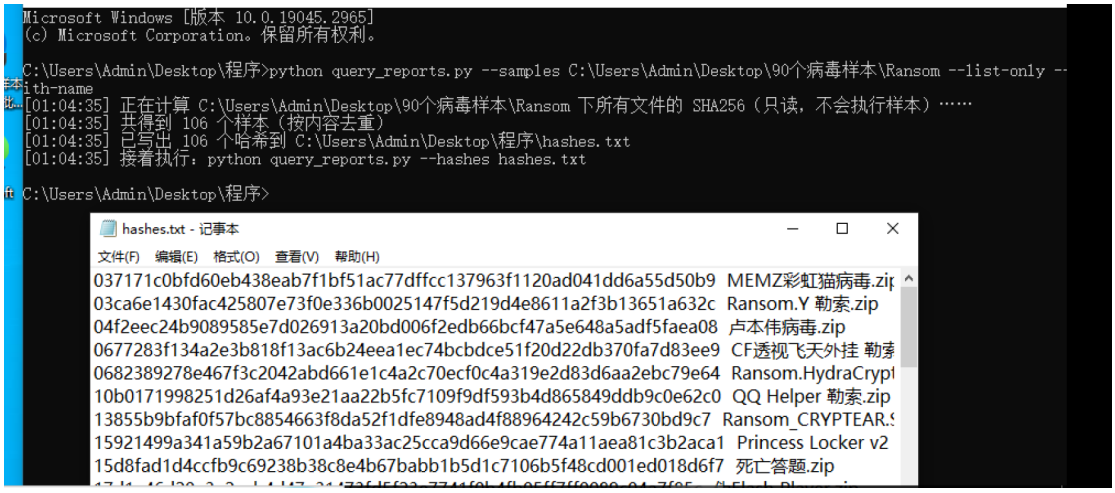


图 104: 在隔离虚拟机内计算样本 SHA256 并生成哈希清单（上为运行输出，下为生成的清单）

检测对象 1：在宿主机上查询

把清单带出虚拟机后，在宿主机上执行查询。协同平台对这批压缩包几乎都没有收录，所以这里只查 VirusTotal：

```
python query_reports.py --hashes hashes-90.txt --only vt
```

程序按 16 秒的间隔逐条查询，106 个样本耗时约 28 分钟；中途按 Ctrl+C 或断网都不会丢进度，重新运行会跳过已下载的报告接着查。跑完后输出目录里得到三类结果：每个样本的原始 JSON 报告、一张汇总全部字段的 `summary.csv`，以及检测报告 `detection_report.md`。

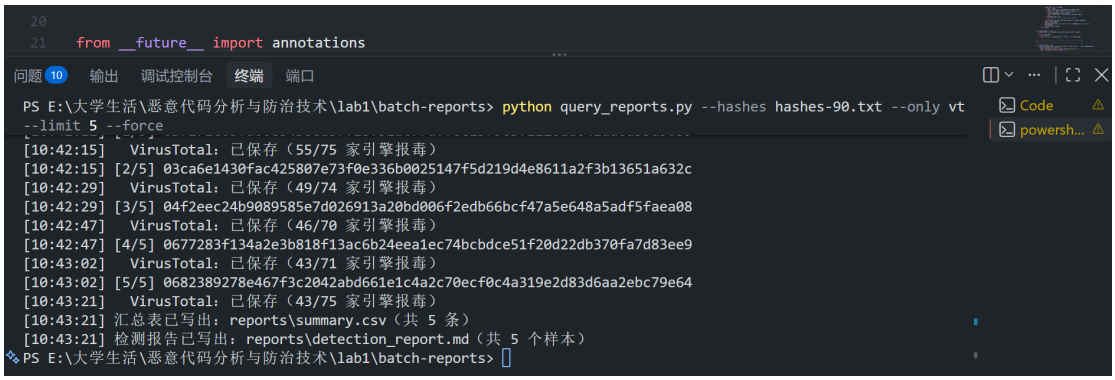


图 105: 宿主机上批量查询的运行过程（为便于截图只跑了前 5 条，实际查询 106 条）

检测对象 2：直接查询分配的哈希值

100 条哈希单独跑一遍，输出到另一个目录，避免两个检测对象的结果混在同一份汇总表和报告里：

```
python query_reports.py --hashes hashes-100.txt --out <dir>
```

过程中遇到的困难

一是三个样本在虚拟机里读不出来。第一轮计算哈希时，有三个文件报出 [Errno 22] Invalid argument，只有 103 个成功。原因既不是程序逻辑也不是路径写法，而是 Windows Defender 的实时保护判定这三个文件为病毒并介入处理，程序读取时被系统直接拒绝。把样本所在目录加入 Defender 的排除项之后重新计算，106 个文件全部成功。

```
C:\Users\Admin\Desktop\程序>python query_reports.py --samples C:\Users\Admin\Desktop\90个病毒样本\Ransom --list-only --with-n
ame
[01:01:43] 正在计算 C:\Users\Admin\Desktop\90个病毒样本\Ransom 下所有文件的 SHA256（只读，不会执行样本）……
[01:01:46]   跳过 Ransom.SyncCrypt 勒索.zip：[Errno 22] Invalid argument: 'C:\\Users\\Admin\\Desktop\\90个病毒样本\\Ransom\\R
ansom.SyncCrypt 勒索.zip'
[01:01:47]   跳过 Unlock92.Ransomware 勒索.zip：[Errno 22] Invalid argument: 'C:\\Users\\Admin\\Desktop\\90个病毒样本\\Ransom
\\Unlock92.Ransomware 勒索.zip'
[01:01:48]   跳过 强制改电脑密码 勒索.zip：[Errno 22] Invalid argument: 'C:\\Users\\Admin\\Desktop\\90个病毒样本\\Ransom\\强
制改电脑密码 勒索.zip'
[01:01:48] 共得到 106 个样本（按内容去重）
[01:01:48] 已写出 106 个哈希到 C:\Users\Admin\Desktop\程序\hashes.txt
[01:01:48] 接着执行：python query_reports.py --hashes hashes.txt

C:\Users\Admin\Desktop\程序>
```

图 106: 三个文件因 Defender 介入而读取失败时的运行输出

二是查询过程中连接被平台重置。表现为协同平台一侧反复报 `ConnectionResetError`，即远程主机强迫关闭了一个现有的连接。排查发现是本机开着代理软件，关掉之后同样的请求立刻恢复正常，与代码无关。

三是协同平台的权限与额度限制。平台对普通用户账号只开放了”文件信誉报告”和”文件多引擎检测报告”两个接口，静态信息、动态分析、完整检测报告三个接口返回”当前账户未开通此 api 权限”；而且每个接口有总计 100 次的调用配额，两个检测对象合计的请求量把配额用完之后，平台开始返回”用户总可用额度已用尽”。程序会自动识别这两类响应并跳过相应接口，不会对后面每个样本反复重试。

图 107: 协同分析平台个人中心的接口权限与配额

(3) 批量检测报告

两个检测对象的检测报告如下，完整文件见文末的仓库地址。报告由程序直接从下载好的原始 JSON 报告汇总生成，每个样本的哈希、大小、类型、检出比等明细都在报告中。

恶意代码批量检测报告

- 检测对象：90 个真实恶意代码样本
- 数据来源：VirusTotal API v3；国家计算机病毒协同分析平台开放接口
- 样本数：106
- 生成时间：2026-09-30 03:15:30
- 说明：该目录实际含 106 个文件，均已检测

一、检测结果概览

指标	数量	占比
样本总数	106	100.0%
VirusTotal 已收录	106	100.0%
VirusTotal 未收录	0	0.0%
至少一家引擎报毒	105	99.1%
平均检出率（报毒引擎占比）	62.5%	—

二、检出率分布

检出的引擎数	样本数	占比
0（无引擎报毒）	1	0.9%
1–5 家	0	0.0%
6–20 家	0	0.0%
21 家及以上	105	99.1%

三、主要恶意家族（Top 10）

家族	样本数
trojan.cerber/zerber	1
trojan.encoder/expelcod	1
trojan.gandcrab/chapak	1
trojan.xorist/razr	1

> VirusTotal 只为 106 个样本中的 4 个给出了聚合家族标签，其余样本没有家族标注——这是平台侧的数据限制，不是漏查；未标注样本的检出命名可查 summary.csv。

四、文件类型分布（Top 10）

文件类型	样本数
ZIP	106

五、检出率最高的样本（Top 10）

#	SHA256	文件名	检出比	家族	协同平台检出
1	`d4a2a8ff4c30e42c6e0df15ab0e4ab4ca9aef10ecc06a84e8845fcfe5ae66483`	锁机.zip	57/75	-	-
2	`696a971f31a94211807e35560fcd34bde565e13404aac4c16dee3a5840123e06`	Ransom.crowti 勒索.zip	56/74	-	-
3	`037171c0bfd60eb438eab7f1bf51ac77dffcc137963f1120ad041dd6a55d50b9`	memz____.zip	55/75	-	-
4	`ad886c73ae4ff4e6ab2ea50be01e90cf12d93cef9014a61a0a5c01dc0399905c`	Ransom_HPCERBER.SMONT2 勒索.zip	55/75	-	-
5	`54f7edcb849c72c7c8ea43b91250e5d70867cf217bd011b93055afa120d15a6f`	Ransom.Mole 勒索.zip	54/75	-	-
6	`5df11d20ab7bd6f2535067e5b7a21cb6fc4b7c7bf0cb1cfc6b451367ccae5da7`	Ransom.Petya (DLL)勒索.zip	54/75	-	-
7	`7580532be2f1215f83574f14d14a939988a54516f9617c7b2f1cc6bcde1fea72`	Sage.Ransom 勒索.zip	54/75	-	10/17
8	`f636b1a16aab9ae14cfdde5b0963e36d3d1fbce1d70262e833402d7fa722f67c`	Petya 勒索.zip	54/75	-	-
9	`fa491f5ff14e079bcc8f01667e1b42133a46e47efb8dd472fe1f2b47917ad5a2`	wcry2.0 勒索.zip	54/75	-	-
10	`26c186803930e1ff4ac8ee741a53ad77bc6d7bae92aa225ca4b4c502699de648`	Ransom.Cerber 2 勒索.zip	53/75	-	-

> 「协同平台检出」列中的「-」表示国家计算机病毒协同分析平台没有收录该样本；本批 106 个样本中该平台有检出数据的是 4 个。

六、VirusTotal 未收录的样本

（全部样本均已在 VirusTotal 收录）

> 每个样本的完整字段（哈希、大小、类型、检出比、家族、提交时间等）见同目录的 `summary.csv`。

图 108: 检测对象 1（106 个样本）的批量检测报告

恶意代码批量检测报告

- 检测对象：100 条 SHA256 哈希值
- 数据来源：VirusTotal API v3；国家计算机病毒协同分析平台开放接口
- 样本数：100
- 生成时间：2026-09-30 03:09:25

一、检测结果概览

指标	数量	占比
样本总数	100	100.0%
VirusTotal 已收录	99	99.0%
VirusTotal 未收录	1	1.0%
至少一家引擎报毒	99	99.0%
平均检出率（报毒引擎占比）	75.9%	—

二、检出率分布

检出的引擎数	样本数	占比
0（无引擎报毒）	0	0.0%
1–5 家	0	0.0%
6–20 家	1	1.0%
21 家及以上	98	98.0%
平台未收录（无检出数据）	1	1.0%

三、主要恶意家族（Top 10）

家族	样本数
trojan.upatre/jqby	2
ransomware.gandcrab/encoder	1
ransomware.gandcrab/gandcrypt	1
ransomware.lockbit/blackmatter	1
trojan.convagent/injuke	1
trojan.cryptolocker/upatre	1
trojan.dpeq/presenoker	1
trojan.emotetu/hes7fooi	1
trojan.guloader/nsis	1
trojan.msil/taskun	1

> VirusTotal 只为 100 个样本中的 15 个给出了聚合家族标签，其余样本没有家族标注——这是平台侧的数据限制，不是漏查；未标注样本的检出命名可查 summary.csv。

四、文件类型分布（Top 10）

文件类型	样本数
Win32 EXE	99

五、检出率最高的样本（Top 10）

#	SHA256	文件名	检出比	家族	协同平台检出
1	`e4340671519e67e8bf5f79c24b9ddeb2393e79f6f9467d89167eb81dbb4c0056`	misid.exe	68/76	-	14/14
2	`e4c4038f4aafd810ad506d2dcc1ce5bd4de17dac221a895c4aaefde150f3fb4e`	lossy.exe	67/75	trojan.upatre/jqby	14/14
3	`e4e024d389669e5d1ded99f025d7d87675eb4e6cbb49e9a59775d47425748f42`	-	67/76	-	13/14
4	`e4e7cc527184a1724559b6c2887514d7e45ef0c990b861b4667799553c199ea8`	{12A061A6-DB1D-4433-B7D9-66ABC23E26DF}.exe	66/76	-	14/14
5	`e4f49129ec71efbff45599bfe2d1c8d50bed72e218a82a9e7c962413dd5e9813`	WinWord.exe	66/76	-	13/14
6	`e50d2e5f050a76a9e063e6361b29d83c55960291668f2178fb7ea45e89ff1539`	-	66/75	trojan.upatre/jqby	14/14
7	`e594e1a1f984b5b3982571d5519c39ba132eb63a4fdd3cbdcc60f7c4775fb2e3`	asih.exe	66/75	-	14/14
8	`e407076bd7576e05d6d9b4a37fca380d9b8561c308f9eb1f1ed9c42861234a48`	Multimedia.exe	65/76	-	14/14
9	`e44e5833e97e572af6bef5e53cee0cd328a6a14767ea64c4ff1c88b1f47e479e`	WinWord.exe	65/75	-	13/14
10	`e48f56198e301531ce5bff14c3abdfab0059d61b0bc47d9da5a3618342364fb7`	WinWord.exe	65/75	-	13/14

> 「协同平台检出」列中的「-」表示国家计算机病毒协同分析平台没有收录该样本；本批 100 个样本中该平台有检出数据的是 93 个。

六、VirusTotal 未收录的样本

共 1 个样本未收录，其中：

`e53b1a1d8bcdcedd187302b4e9c5b063ce579cd2be4b181f0e7005eae6e8be18`

> 每个样本的完整字段（哈希、大小、类型、检出比、家族、提交时间等）见同目录的 `summary.csv`。

图 109: 检测对象 2（100 条哈希值）的批量检测报告

完整文件与仓库地址

批量检测程序、两份哈希清单、每个样本的 VirusTotal 原始 JSON 报告、汇总明细 `summary.csv` 以及上述两份检测报告的完整版本，都已放在 GitHub 仓库中：

`https://github.com/Bilnd-me-with-your-light/nankai_learn/tree/main/恶意代码分析与防治技术/lab1/code`

四、实验结论及心得体会

本次实验把恶意代码静态分析的工具链完整走了一遍。我知道了判断样本性质靠的不是单一证据，而是细节的相互印证；在线平台的检出用起来方便，但也有局限，不能把它当成最终结论，而是自己也加入到平台的建设中；

关于 LaTeX。这份报告用 XeLaTeX 编译，花在排版上的时间远超预期。最麻烦的是模板由 Word 转换而来，像封面五行下划线等长这种小版式，LaTeX 里得先量出最宽的一栏再统一撑开；而”宋体加粗”这种看似简单的需求也不直接成立——SimSun 没有粗体字重，\textbf 会静默回退成常规字重，不报错也不变粗，最后靠编译日志里一条字体警告才定位到根因。此外，中文夹西文的长串容易顶出右边距，报错信息也常常指向错误的下游。

关于大模型的帮助。这份报告的模板转换、宏定义和第三部分的 `query_reports.py` 都借助了大模型，它是帮助最大的一环。但它能给出看起来完整的方案，却不会替你验证是否真的成立——粗体失效就是它一开始也没意识到的问题。所以更靠谱的用法是把它当成实现层的手：由我确定分析目标、准备样本和哈希清单、实际运行程序，再逐条比对输出与平台网页上的结果，写进报告的结论都是核对过才算数的。