

# Blast from the past: stowaway Virut delivered with Chinese DDoS bot

[blog.malwarebytes.com/threat-analysis/2018/03/blast-from-the-past-stowaway-virut-delivered-with-chinese-ddos-bot/](http://blog.malwarebytes.com/threat-analysis/2018/03/blast-from-the-past-stowaway-virut-delivered-with-chinese-ddos-bot/)

hasherezade

March 1, 2018



Recently, we described an [unusual Chinese drive-by attack](#) that was delivering a variant of the [Avzhan DDoS bot](#). The attack also contained multiple components that were not-so-new. Among the [exploits](#), the newest was from 2016. Avzhan is also not a recent malware—the compilation timestamp of the [unpacked payload](#) was from August 2015. But there was one more unusual thing that triggered our attention. The outer layer of Avzhan matched the signatures of Virut, a malware that’s been dead in the water since 2013.

At first, it was hard to believe this detection. Who would want to distribute such an old piece of malware that is no longer developed, and whose CnC servers were sinkholed long ago by Polish CERT? Maybe it was the author of the packer by which the DDoS bot was wrapped incorporating some Virut-like obfuscation?

After further research, it turned out the detection was not wrong. The Avzhan bot carried along with it a legitimate Virut. But it is unlikely that the distributors added it intentionally. Rather, the server from where the attack was deployed happened to be infected with Virut. The [virus](#) attached as a parasite to the distributed DDoS malware, and was dropped with the

drive-by attack into new places. Interestingly, in 2016 Virut's code was also found in Chinese cameras. Similarly, the computers of developers were infected with Virut, and by this way its code got passed further.

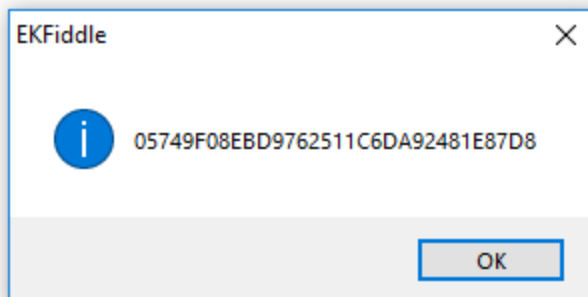
Since Virut has made this unexpected reappearance, we will have a look at how it works in this post.

## Analyzed sample

---

05749f08ebd9762511c6da92481e87d8 – the main sample, dropped by the exploit

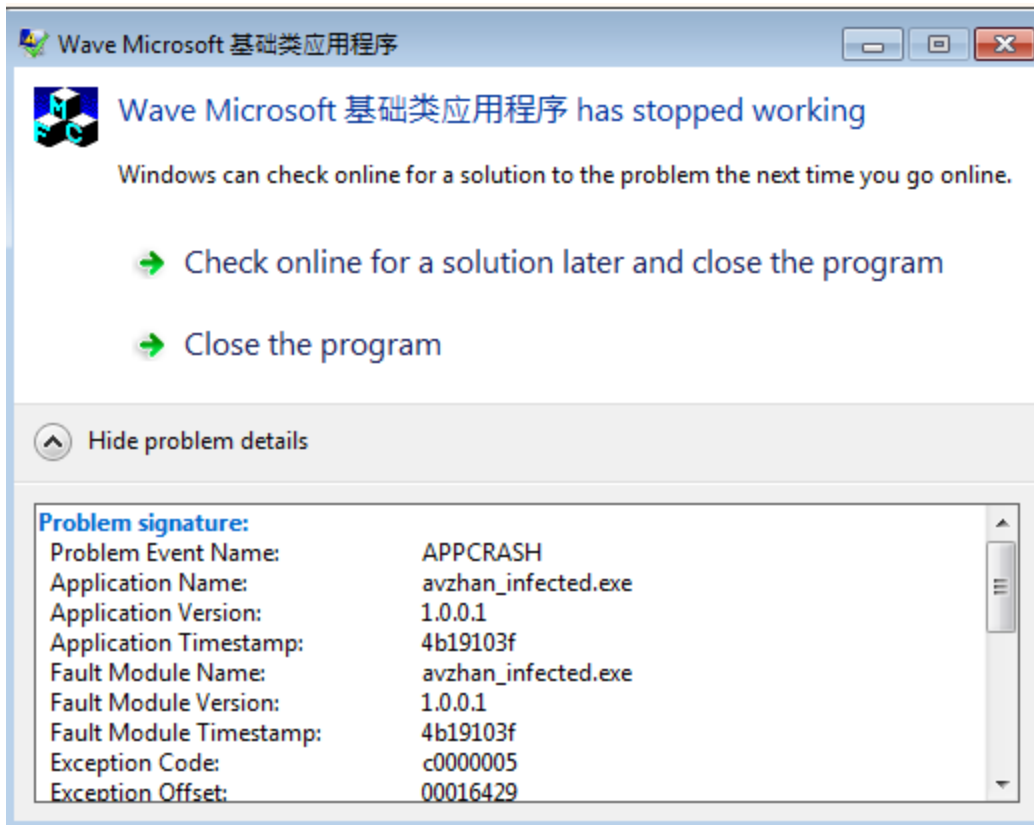
Host	URL	Body	Comments
wm.shiquanxian.cn	/	274	Compromised site
wm.shiquanxian.cn	/ie.html	760	Call for ActiveX
wm.shiquanxian.cn	/ms.html	34,080	CVE-2016-0189
wm.shiquanxian.cn	/DownloaderActiveX.cab	82,625	Malicious ActiveX downloader
wm.shiquanxian.cn	/yuyan.swf	45,037	CVE-2015-5119
wm.shiquanxian.cn	//2.exe	118,784	Payload



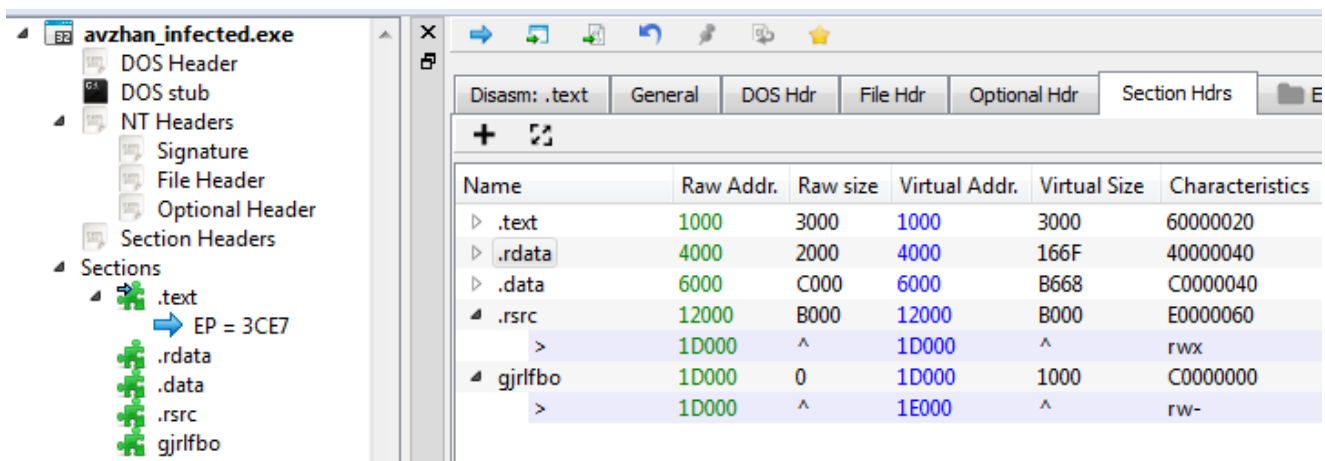
## Behavioral analysis

---

Virut behaves like a typical, old-fashioned infectious virus. As we observed, samples infected by Virut always crashed on 64-bit systems.



However, when deployed in a 32-bit environment, Virut spread like fire, trying to infect all executables it could reach by attaching its own code. The code of Virut is polymorphic and designed with great care, so the infection patterns are not easy to grasp. Often (if there is enough space), Virut adds a new, empty section with a random name, for example:



If there is no space for a new header in the input file, this step is omitted. So, the absence of the added section does not guarantee that the file is clean. Another suspicious indicator may be that the last section is set to RWX (Read-Write-eXecute).

Virut changes sizes of the sections and the entry point of the application in order to redirect to its own code. After the malicious code is deployed, the original entry point is executed. So, from the user's point of view, the infected application works as before.

In addition to infecting files on the disk, Virut attacks running processes as well. So, even if the first infected process was killed, the malicious code keeps running in the memory.

The malware uses some hardcoded CnC addresses, as well as a DGA (Domain Generation Algorithm). Looking at the network traffic, we can see the queries to the domains follow the pattern of using six letters before the dot com: `6{a-z}.com`

Destination	Protocol	Length	Info
100.78.239.97	NTP	90	NTP Version 4, server
185.89.185.1	DNS	70	Standard query 0x0ace A bobyku.com
100.78.239.97	DNS	143	Standard query response 0x0ace No such name A bobyku.com SOA a.gtld-servers.net
255.255.255.255	NBNS	92	Name query NB BOBYKU.COM<00>
255.255.255.255	NBNS	92	Name query NB BOBYKU.COM<00>
255.255.255.255	NBNS	92	Name query NB BOBYKU.COM<00>
185.89.185.1	DNS	70	Standard query 0x37a1 A zocarg.com
100.78.239.97	DNS	143	Standard query response 0x37a1 No such name A zocarg.com SOA a.gtld-servers.net
255.255.255.255	NBNS	92	Name query NB ZOCARG.COM<00>
255.255.255.255	NBNS	92	Name query NB ZOCARG.COM<00>
255.255.255.255	NBNS	92	Name query NB ZOCARG.COM<00>
185.89.185.1	DNS	70	Standard query 0x5ff0 A dwrnpe.com
100.78.239.97	DNS	143	Standard query response 0x5ff0 No such name A dwrnpe.com SOA a.gtld-servers.net
255.255.255.255	NBNS	92	Name query NB DWRNPE.COM<00>
255.255.255.255	NBNS	92	Name query NB DWRNPE.COM<00>
255.255.255.255	NBNS	92	Name query NB DWRNPE.COM<00>
185.89.185.1	DNS	70	Standard query 0xae0 A kkipad.com
100.78.239.97	DNS	143	Standard query response 0xae0 No such name A kkipad.com SOA a.gtld-servers.net
255.255.255.255	NBNS	92	Name query NB KKIPAD.COM<00>

Due to the fact that the full infrastructure of Virut was sinkholed, none of its CnC servers are active.

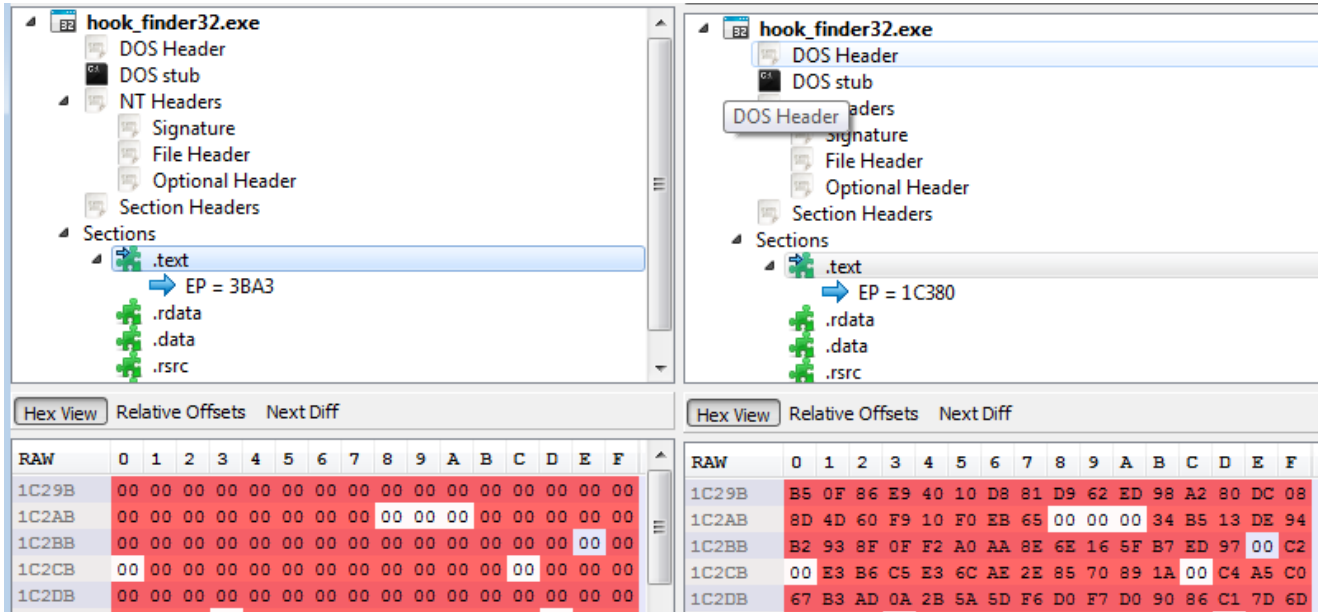
## Inside

### Infection patterns

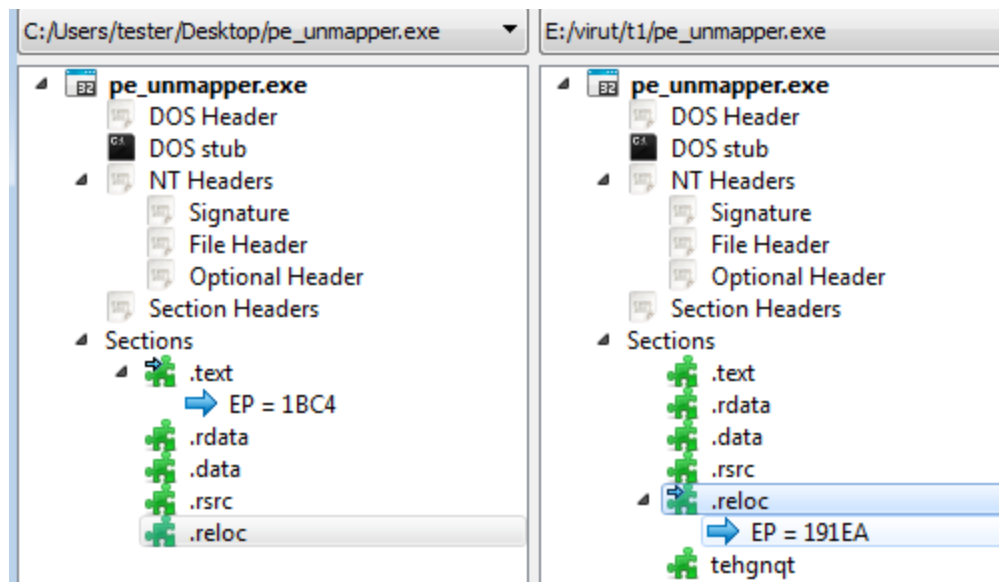
As mentioned before, Virut's code can mutate—each infection looks different. Some of the chosen patterns depend on the features of the input.

In PE files, each section must be aligned to the minimal unit that is indicated by a file alignment field in the PE header. This is why sometimes there is an empty space between one PE section and the other, filled only with padding. This empty space is called the cave. Old infectors often used this space to implant their own code. This is what Virut also tries to do.

In the example below, a cave after the .text section has been filled with malicious code:



Depending on the input, there may not be sufficient caves between sections. Then, Virut adds its code just at the end of the last section:



But this is not the only thing that impacts the features of the infection. The code generated by Virut is polymorphic, so the same file will not be infected twice in the same way. Below is a comparison of code from the same application, infected by Virut in two different runs:

Hex	Disasm	Hex	Disasm		
1CF08	FEC0	INC AL	3CE7	693A6B0000	PUSH DWORD 0X6B3A
1CF0A	19C8	SBB EAX, ECX	3CEC	F8	CLC
1CF0C	8AC2	MOV AL, DL	3CED	59	POP ECK
1CF0E	91	XCHG ECX, EAX	3CEE	86E6	XCHG DH, AH
1CF0F	20C1	AND CL, AL	3CF0	E9A5000000	JMP 0X00403D9A
1CF11	40	INC EAX	3CF5	41	INC ECK
1CF12	F5	CMC	3CF6	03A16E00C480	ADD ESP, [ECX+0X80C4006E]
1CF13	68526B0000	PUSH DWORD 0X6B52	3CFC	6D	INSD
1CF18	F8	CLC	3CFD	1A5372	SBB DL, [EBX+0X72]
1CF19	5A	POP EDX	3D00	EAEAD4D4000F9E	JMP DWORD 0X9E0F.0XD4D4EA
1CF1A	86E9	XCHG CL, CH	3D07	00FE	ADD DH, BH
1CF1C	86C8	XCHG AL, CL	3D09	81CCF1000005	OR ESP, 0X50000F1

Virut's shellcode

The code appended to the infected files makes an initial stub that unpacks in the memory of Virut's shellcode. That is a heart of the malware. This is how the unpacked shellcode looks:

The same code is also injected into other processes. It is implanted in a new page in the memory. Example:

771E0000	00001000	msvcrt.dll					
771E1000	0009F000	".text"	Executable	IMG	-R---	ERWC-	
77280000	00007000	".data"	Initial	IMG	-RW--	ERWC-	
77287000	00001000	".rsrc"	Resource	IMG	-R---	ERWC-	
77288000	00004000	".reloc"	Base rel	IMG	-R---	ERWC-	
772A0000	00001000			IMG	-R---	ERWC-	
7F6F0000	00005000			MAP	-R---	-R---	
7F6F5000	000FB000	Reserved (7F6F0000)		MAP	-R---	-R---	
7FFA0000	0000C000			MAP	ERW-G	ERW--	
7FFB0000	00023000			MAP	-R---	-R---	
7FFDD000	00001000	Thread 8BC TEB		PRV	-RW--	-RW--	

The shellcode contains the functionality of a userland rootkit. It hooks NTDLL within every infected process so that each time the specific function is called, the execution is redirected first to Virut's implant. There are seven functions that are hooked:

1. NtCreateFile
2. NtCreateProcess
3. NtCreateProcessEx
4. NtCreateUserProcess
5. NtDeviceIoControlFile
6. NtOpenFile
7. NtQueryInformationProcess

Below you can see an example of the hooked function `NtCreateFile`. As you can see, the first instruction is a call to the malicious memory page:

	Hex	Disasm	
455C8	★ E8C40FF008	CALL 0X7FFA6591	patch_0
455CD	BA0003FE7F	MOV EDX, 0X7FFE0300	
455D2	FF12	CALL DWORD NEAR [EDX]	
455D4	C22C00	RET 0X2C	
455D7	90	NOP	

And this is how the code looks that is being called:

7FFA6591	B8 42000000	MOV EAX, 0x42
7FFA6596	60	PUSHAD
7FFA6597	91	XCHG EAX, ECX
7FFA6598	8B4424 30	MOV EAX, DWORD PTR SS:[ESP+0x30]
7FFA659C	8B50 08	MOV EDX, DWORD PTR DS:[EAX+0x8]
7FFA659F	66:813A 0602	CMPL WORD PTR DS:[EDX], 0x206
7FFA65A4	73 44	JNB SHORT 7FFA65EA
7FFA65A6	E8 96FFFFFF	CALL 7FFA6541
7FFA65AB	75 3F	JNZ SHORT 7FFA65EC
7FFA65AD	64:FF35 24000000	PUSH DWORD PTR FS:[0x24]
7FFA65B4	8F85 027F270D	POP DWORD PTR SS:[EBP+0xD277F02]
7FFA65BA	8DB5 A27C270D	LEA ESI, DWORD PTR SS:[EBP+0xD277CA2]
7FFA65C0	56	PUSH ESI
7FFA65C1	68 0000FF00	PUSH 0xFF0000
7FFA65C6	8BC4	MOV EAX, ESP
7FFA65C8	6A 00	PUSH 0x0
7FFA65CA	52	PUSH EDX
7FFA65CB	50	PUSH EAX
7FFA65CC	FF95 0224270D	CALL DWORD PTR SS:[EBP+0xD272402]
7FFA65D2	83C4 08	ADD ESP, 0x8
7FFA65D5	813E 5C3F3F5C	CMPL DWORD PTR DS:[ESI], 0x5C3F3F5C
7FFA65DB	75 03	JNZ SHORT 7FFA65E0
7FFA65DD	83C6 04	ADD ESI, 0x4
7FFA65E0	E8 27E6FFFF	CALL 7FFA4C0C
7FFA65E5	E8 71FFFFFF	CALL 7FFA655B
7FFA65EA	61	POPAD
7FFA65EB	C3	RETN

We also find the lists of AV products, that Virut uses in order to check if it is running in the controlled environment:



7FFA666E	MOV EAX,0xEA	patch_6
7FFA6673	CMP DWORD PTR SS:[ESP+0xC],0x22	
7FFA6678	JNZ SHORT 7FFA66A6	
7FFA667A	CMP DWORD PTR SS:[ESP+0x8],-0x1	
7FFA667F	JNZ SHORT 7FFA66A6	
7FFA6681	CALL 7FFA6686	
7FFA6686	POP EDX	ntdll.770A604D
7FFA6687	CMP DWORD PTR DS:[EDX+0xFFFFA203],0x0	
7FFA668E	JLE SHORT 7FFA66A6	
7FFA6690	LEA EDX,DWORD PTR SS:[ESP+0x8]	
7FFA6694	INT 0x2E	
7FFA6696	CMP EAX,0x0	
7FFA6699	JL SHORT 7FFA66A2	
7FFA669B	MOV EDX,DWORD PTR SS:[ESP+0x10]	
7FFA669F	OR DWORD PTR DS:[EDX],0x30	
7FFA66A2	POP EDX	ntdll.770A604D
7FFA66A3	RETN 0x14	
7FFA66A6	RETN	
7FFA66A7	AND AL,0x65	
7FFA66A9	JNB SHORT 7FFA6710	
7FFA66AB	JE SHORT 7FFA66A0	

EAX=005C2F18

Address	Hex dump	ASCII
7FFA666E	B8 EA 00 00 00 83 7C 24 0C 22 75 2C 83 7C 24 08	...
7FFA667E	FF 75 25 E8 00 00 00 5A 83 8A 03 A2 FF FF 00	uZR...Zall#6 .
7FFA668E	7E 16 8D 54 24 08 CD 2E 83 F8 00 7C 07 8B 54 24	"_2T\$=,ã°.!;òT\$
7FFA669E	10 83 0A 30 5A C2 14 00 C3 24 65 73 65 74 00 23	ã.0ZT\$.!\$eset.#
7FFA66AE	61 76 67 00 6D 69 63 72 6F 73 6F 66 74 00 77 69	avg.microsoft.wi
7FFA66BE	6E 64 6F 77 73 75 70 64 61 74 65 00 77 69 6C 64	ndowupdate.wild
7FFA66CE	65 72 73 73 65 63 75 72 69 74 79 00 74 68 72 65	ersecurity.thre
7FFA66DE	61 74 65 78 70 65 72 74 00 63 61 73 74 6C 65 63	atexpert.castlec
7FFA66EE	6F 70 73 00 73 70 61 60 68 61 75 73 00 63 70 73	ops.spamhaus.cps
7FFA66FE	65 63 75 72 65 00 61 72 63 61 62 69 74 00 65 6D	ecure.arcabit.em
7FFA670E	73 69 73 6F 66 74 00 73 75 6E 62 65 6C 74 00 73	sisoft.sunbelt.s
7FFA671E	65 63 75 72 65 63 6F 6D 70 75 74 69 6E 67 00 72	ecurecomputing.r
7FFA672E	69 73 69 6E 67 00 70 72 65 76 78 00 70 63 74 6F	ising.prevx.pcto
7FFA673E	6F 6C 73 00 6E 6F 72 6D 61 6E 00 68 37 63 6F 6D	ols.norman.k7com
7FFA674E	70 75 74 69 6E 67 00 69 68 61 72 75 73 00 68 61	puting.ikarus.ha
7FFA675E	75 72 69 00 68 61 63 68 73 6F 66 74 00 67 64 61	uri.hacksoft.gda
7FFA676E	74 61 00 66 6F 72 74 69 6E 65 74 00 65 77 69 64	ta.fortinet.ewid
7FFA677E	6F 00 63 6C 61 6D 61 76 00 63 6F 6D 6F 64 6F 00	o.clamav.comodo.
7FFA678E	71 75 69 63 68 68 65 61 6C 00 61 76 69 72 61 00	quickheal.avira.
7FFA679E	61 76 61 73 74 00 65 73 61 66 65 00 61 68 6E 6C	avast.esafe.ahn1
7FFA67AE	61 62 00 63 65 6E 74 72 61 6C 63 6F 6D 6D 61 6E	ab.centralcomman
7FFA67BE	64 00 64 72 77 65 62 00 67 72 69 73 6F 66 74 00	d.drweb.grisoft.
7FFA67CE	6E 6F 64 33 32 00 66 2D 70 72 6F 74 00 6A 6F 74	nod32.f-prot.jot
7FFA67DE	74 69 00 68 61 73 70 65 72 73 68 79 00 66 2D 73	ti.kaspersky.f-s
7FFA67EE	65 63 75 72 65 00 63 6F 6D 70 75 74 65 72 61 73	ecure.computeras
7FFA67FE	73 6F 63 69 61 74 65 73 00 6E 65 74 77 6F 72 68	sociates.network
7FFA680E	61 73 73 6F 63 69 61 74 65 73 00 65 74 72 75 73	associates.etrus
7FFA681E	74 00 70 61 6E 64 61 00 73 6F 70 68 6F 73 00 74	t.panda.sophos.t
7FFA682E	72 65 6E 64 6D 69 63 72 6F 00 6D 63 61 66 65 65	rendmicro.mcafee
7FFA683E	00 6E 6F 72 74 6F 6E 00 73 79 6D 61 6E 74 65 63	.norton.symantec
7FFA684E	00 64 65 66 65 6E 64 65 72 00 72 6F 6F 74 68 69	.defender.rootkl
7FFA685E	74 00 6D 61 6C 77 61 72 65 00 73 70 79 77 61 72	t.malware.spywar
7FFA686E	65 00 76 69 72 75 73 00 00 8D BD A7 76 27 0D 83	e.virus..22ãv'.ã
7FFA687E	FB 04 72 2E 57 FF 95 BE 22 27 0D 38 C3 91 76 04	ûr.w lã''.; Lvø
7FFA688E	03 F9 EB 14 56 5F 01 0C 24 AC 0C 20 3A 07 75 04	*"0UW0.\$C. :uø
7FFA689E	47 49 75 F5 5F 5E 74 0A 47 80 3F 00 75 D6 46 4B	GIuS^t.GC?.uifK
7FFA68AE	EB C7 C3 81 7C 24 1C 23 2D 01 00 75 73 83 7C 24	UãW!\$L# 0.usã!\$
7FFA68BE	24 38 75 6C 8B 44 24 2D 3D 00 00 10 00 72 61 85	\$8ulöD\$ ..ã.raã

Apart from the rootkit, it contains the code responsible for communication with the CnC. For example, among the embedded strings we found IRC commands that suggest that IRC was part of Virut's communication:

7FFA3210	27 0D C6 85 BC 87 27 00 01 E8 C7 09 00 00 FF 95	'..ãã" c'.0Rã... l
7FFA3220	1A 23 27 0D 05 B8 88 00 00 8B 8D 79 18 27 0D 89	+*'.#St..ö2y+'.ë
7FFA3230	85 C0 87 27 0D 6A 00 6B C9 0D 51 E8 0B 00 00 00	d'c'.j.kf.0Rø...
7FFA3240	4A 4F 49 4E 20 23 2E 25 64 0A 00 57 FF 95 4C 7A	JOIN #.%d..w lLz
7FFA3250	27 0D 83 C4 0C 50 57 53 E8 FB 00 00 00 85 C0 0F	'..ã-.PWSRü...ã*
7FFA3260	8F E2 00 00 00 E9 EC 00 00 00 8D 85 94 85 27 0D	c0...0y...2ãã'
7FFA3270	50 FF 95 0A 23 27 0D 0F B7 95 9A 85 27 0D 0F B7	P l.#'.*Eüã'.*E
7FFA3280	8D 96 85 27 0D 0F B7 85 94 85 27 0D 52 51 50 E8	2lã'.*Eãã'.RQPR
7FFA3290	12 00 00 00 44 53 54 41 4D 50 20 25 64 25 30 32	*...DSTAMP %d%02
7FFA32A0	64 25 30 32 64 0A 57 FF 95 4C 7A 27 0D 83 C4 14	d%02d..w lLz'.ã-ü

List of command patterns:

```

PING
NICK nrmbhoz
PRIV
JOIN #.%d
DSTAMP %s%02d%02d

```



There are also hardcoded addresses of the CnCs. Two servers are static and always occur in Virut samples (both of them are sinkholed by Polish CERT):

```
ilo.brenz.pl  
ant.trenz.pl
```

But, we can also see the domains generated by the Virut's DGA:

```
7108: 00 00 00 00 00 00 00 00 |.....  
7110: 00 00 00 00 00 00 00 00 |.....  
7118: 00 00 00 00 00 00 00 00 |.....  
7120: 7D C8 92 68 75 78 77 75 |}Ć'huxwu  
7128: 68 65 2E 63 6F 6D 00 69 |he.com.i  
7130: 69 78 65 6C 69 2E 63 6F |ixeli.co  
7138: 6D 00 6E 6B 61 77 6B 69 |m.nkawki  
7140: 2E 63 6F 6D 00 78 72 62 |.com.xrb  
7148: 75 75 73 2E 63 6F 6D 00 |uus.com.  
7150: 6D 61 76 65 71 69 2E 63 |maveqi.c  
7158: 6F 6D 00 6F 75 74 78 70 |om.outxp  
7160: 68 2E 63 6F 6D 00 75 67 |h.com.ug  
7168: 6A 6F 7A 62 2E 63 6F 6D |jozb.com  
7170: 00 65 6E 66 61 64 77 2E |.enfadw.  
7178: 63 6F 6D 00 77 69 75 6D |com.wium  
7180: 62 73 2E 63 6F 6D 00 6B |bs.com.k  
7188: 75 61 61 6F 64 2E 63 6F |uaaod.co  
7190: 6D 00 75 67 6F 67 6D 6A |m.ugogmj  
7198: 2E 63 6F 6D 00 6E 65 61 |.com.nea  
71A0: 75 71 67 2E 63 6F 6D 00 |uqg.com.  
71A8: 75 79 6F 64 6D 61 2E 63 |uyodma.c  
71B0: 6F 6D 00 64 74 6C 65 69 |om.dtlei  
71B8: 63 2E 63 6F 6D 00 76 6C |c.com.vl  
71C0: 75 65 6E 76 2E 63 6F 6D |uenv.com  
71C8: 00 71 6B 6B 65 6E 75 2E |.qkkenu.  
71D0: 63 6F 6D 00 65 69 6C 64 |com.eild  
71D8: 61 61 2E 63 6F 6D 00 64 |aa.com.d  
71E0: 66 6A 62 74 69 2E 63 6F |fjbtj.co  
71E8: 6D 00 6B 63 61 78 66 68 |m.kcaxfh  
71F0: 2E 63 6F 6D 00 65 62 65 |.com.ebe  
71F8: 71 75 67 2E 63 6F 6D 00 |qug.com.  
7200: 6A 61 71 76 72 6D 2E 63 |jaqvrn.c  
7208: 6F 6D 00 65 63 6F 6A 72 |om.ecojr  
7210: 6B 2E 63 6F 6D 00 77 79 |k.com.wy  
7218: 73 74 6F 64 2E 63 6F 6D |stod.com  
7220: 00 75 66 79 6F 77 73 2E |.ufyows.  
7228: 63 6F 6D 00 71 61 72 69 |com.qari  
7230: 70 65 2E 63 6F 6D 00 76 |pe.com.v  
7238: 74 74 67 69 72 2E 63 6F |ttgir.co  
7240: 6D 00 6F 65 66 73 62 72 |m.oefsbr  
7248: 2E 63 6F 6D 00 6D 73 79 |.com.msy  
7250: 65 65 63 2E 63 6F 6D 00 |eec.com.  
7258: 6F 75 6A 74 6E 72 2E 63 |oujtnr.c  
7260: 6F 6D 00 76 7A 6F 62 6C |om.vzobl  
7268: 7A 2E 63 6F 6D 00 6B 69 |z.com.ki  
7270: 71 77 61 65 2E 63 6F 6D |qwae.com  
7278: 00 78 61 75 74 61 75 2E |.xautau.  
7280: 63 6F 6D 00 69 79 66 73 |com.iyfs
```

While the code infecting the file mutates, the injected shellcode has a pretty consistent structure. If we compare dumps from two different processes, we find that most of the code is the same.

```

0000: 83 3C 24 FE 77 FE 8D 64 |<šřwtřĎ
0008: 24 CC 60 83 EC DC E8 D8 |šĚ`ēÜčŘ
0010: 00 00 00 4B 66 4B 0F 85 |...KfK...
0018: F8 FF FF FF FF 73 3C 59 |ž''''s<Y
0020: 81 E9 FD FF FF 7F 0F 83 |éý''Ď.
0028: E7 FF FF FF 81 D9 E6 13 |č''''Ůć.
0030: 00 00 0F 81 DB FF FF FF |...Ů''''
0038: FF B4 19 E4 13 00 80 83 |'''.ä..Ě
0040: C4 04 66 81 44 24 FC B0 |Ä.fDšü°
0048: BA 0F 85 C4 FF FF FF 68 |š...Ä''''h
0050: 37 DD 43 1C E8 AC 00 00 |7ŸC.č~..
0058: 00 89 74 24 44 E8 2D 01 |.žřDč~.
0060: 00 00 89 44 24 34 83 E8 |...žDš4č
0068: 04 0F 82 45 00 00 00 64 |...E...d
0070: A1 18 00 00 00 85 C0 0F |'.....Ř.
0078: 88 08 00 00 00 8B 40 34 |....<@4
0080: E9 0F 00 00 00 68 D2 56 |é....hŇV
0088: DF 1E E8 76 00 00 00 E8 |š.čv...č
0090: 6D 00 00 00 38 D8 0F 85 |m...8Ř...
0098: 05 00 00 00 E8 BD 01 00 |....č''..
00A0: 00 68 D6 DD A1 62 E8 5A |.hÖŸ'bčZ
00A8: 00 00 00 FF 74 24 34 E8 |...tš4č
00B0: 4D 00 00 00 5F 5E 5D 8B |M..._<]
00B8: 5C 24 04 83 C4 08 5A 59 |\$.Ä.ZY
00C0: 58 C2 30 00 2B C0 48 47 |XÄ0.+ŘHG
00C8: B9 B5 EF 00 00 0F AF CA |apd...žĚ
00D0: 30 4F FF 47 28 77 FE 87 |00'G(wřř
00D8: D1 4F 83 E8 F8 86 D6 3D |Ň+ÖÖčř=
00E0: 97 36 03 00 0F 86 DD FF |-6...+Ÿ'
00E8: FF FF C3 8B 2C 24 8D 64 |''Ä<,šřĎ
00F0: 24 E0 FF 74 24 54 81 44 |šř'tšřĎ
00F8: 24 24 FA CE FC FF 5B FF |ššůřř'['
0100: E5 FF E6 16 96 8D 4B 09 |í'ć.-řK.
0108: 6A FF FF 71 33 5A 03 4C |j''q3Z.L
0110: 1A 78 5A 8B 79 17 42 8B |.xZ<y.<B<
0118: 04 1F 83 C7 04 8D 74 18 |..Č.řt.
0120: FD 3B 51 0F 0F 83 64 00 |ý;Q...d.
0128: 00 00 53 51 33 C0 6B C0 |...SQ3ŘkŘ
0130: 0F 0F B6 4E 03 2B C1 83 |...ŇN.+Á
0138: EE EE 8D 76 EF 80 7E 03 |íířřvdč~.
0140: 0A 0F 83 E7 FF FF FF 33 |...č''''3
0148: 44 24 0C 59 5B 35 06 8E |D$.Y[5.ž
0150: A5 CF 0F 85 BE FF FF FF |Ađ...I''''
0158: 83 E9 EF 83 C4 08 FF 71 |édÄ.'q
0160: 02 5F 03 FB 03 D2 8B 71 |...Ů.Ň<q
0168: 0A 03 D3 66 8B 04 32 C1 |..Óř<.2Ä
0170: E0 10 C1 E8 0E FF 34 38 |ř.Äč.'48
0178: 01 1C 24 8B 44 24 FC 5E |...š<Dšřü^

```

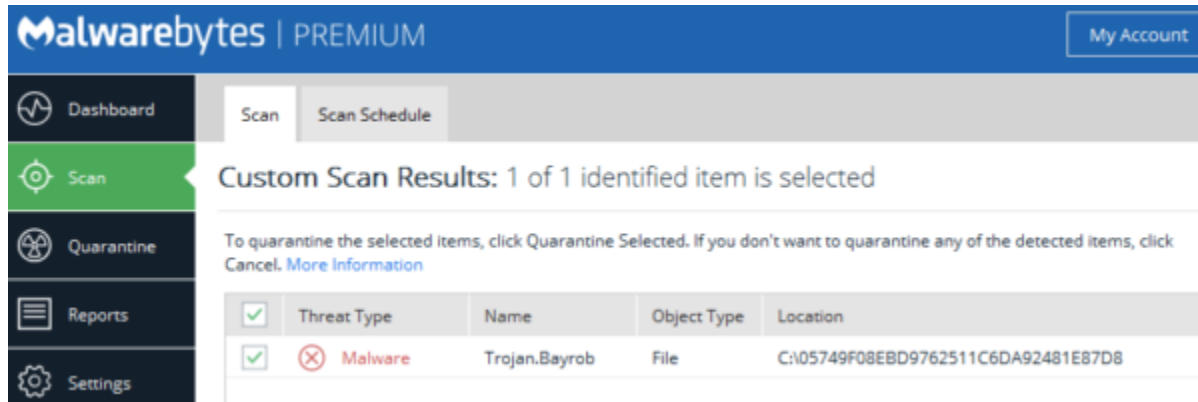
## Conclusion

Nowadays, such old viruses are mostly forgotten, but it doesn't mean that we are fully safe from them. Fortunately, most AV products can detect viruses like Virut by their signatures – but the people who decided not to use AV may still become their victims.

Even their command-and-control infrastructure is dead, the old infectors can roam around. There are old servers in the world that are left infected with old viruses, such as Virut or MyDoom. On our honeypots, we regularly get spam that is being sent from such abandoned bots.

Yet, it is unusual to encounter an old virus in wild sent by a modern-style drive-by attack. We never know how an old threat can get blended with a new one. This time we were lucky and the attack was simple, with a small reach.

Malwarebytes detects this DDoS bot binary as Trojan.Bayrob.



The screenshot shows the Malwarebytes Premium interface. The top navigation bar includes the Malwarebytes logo, 'PREMIUM' status, and a 'My Account' button. A left sidebar contains navigation options: Dashboard, Scan (highlighted), Quarantine, Reports, and Settings. The main content area is titled 'Custom Scan Results: 1 of 1 identified item is selected'. Below this, there is a text instruction: 'To quarantine the selected items, click Quarantine Selected. If you don't want to quarantine any of the detected items, click Cancel. [More information](#)'. A table displays the scan results:

<input checked="" type="checkbox"/>	Threat Type	Name	Object Type	Location
<input checked="" type="checkbox"/>	Malware	Trojan.Bayrob	File	C:\05749F08EBD9762511C6DA92481E87D8