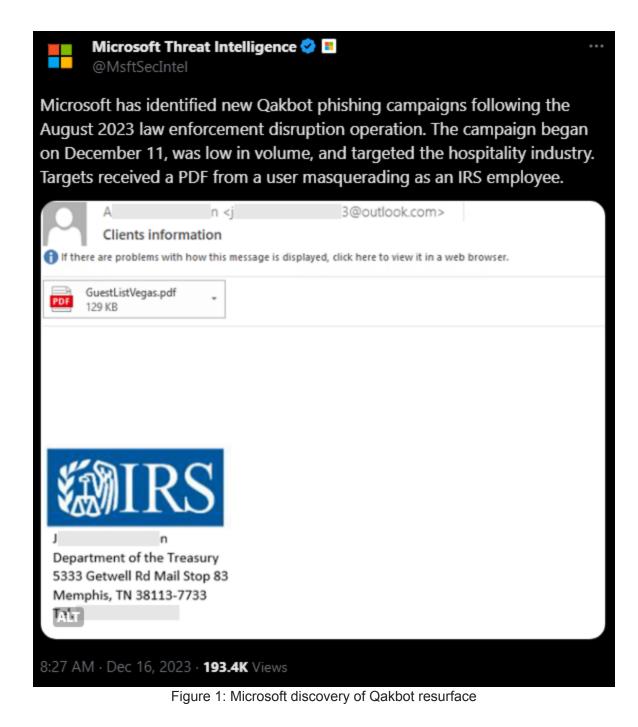
Qakbot Returns - K7 Labs

V labs.k7computing.com/index.php/qakbot-returns/

By Saikumaravel

January 4, 2024

The Qakbot malware has reappeared just four months after law enforcement disrupted its distribution in the "Duck Hunt" operation. Lately, various security companies have noticed the malware spreading through phishing emails. Microsoft, which discovered <u>this</u>, described it as a small-scale campaign starting on December 11, 2023, specifically targeting the hospitality industry. Although the number of these emails is currently low, given Qakbot's past persistence, it's anticipated that the volume will rise soon. We got our hands on one such sample by this <u>tweet</u>.



Binary Analysis

As per Microsoft's tweet, in the recent campaign, an MSI file is being downloaded to the user's machine from the malicious PDFs which were spread through phishing mails.

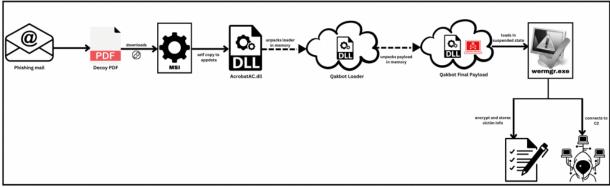


Figure 2: Execution Flow

On analysing the MSI file, we found that the suspicious DLL compressed inside was a patched IDM (Internet Download Manager) DLL with Qakbot inside.

🏂 🕕 C:\Users\	sktop\e88610db056	i36a1476435ec1f39d	13651b080c8a6b875	6452d421d7a822a	2e115.msi\6.1234_1.cab\
Name	Size	Modified	Attributes	Method	Block
l.dll	919 552	2023-12-19 17:20	А	MSZip	0

Qakbot inside IDM DLL

Name		000c DWORD	000a6394	4	Hex	idmcchandler7_64.dll
Base		0010 DWORD	0000000	1		
Show valid						s
Ordinal	RVA	Name				
0003	00039800	000a64b9	hostFile			
0004	0003be20	000a6404	EditOwnerInfo			

Figure 3: Qakbot inside IDM DLL

We found that this DLL was packed with a custom packer. Usually unpacking the Qakbot DLL is quite simple. It uses **VirtualAlloc()** to allocate memory to unpacked code and **VirtualProtect()** to change the protection on a memory region. We set breakpoints on both of those APIs to unpack. We first got the dump of the PE file without the MZ header. Later, we found that it was the Qakbot second stage loader by manually adding the MZ header. The threat actor employs this method to avoid detection by EDR, as it scans memory regions for MZ headers to identify potential process injection methods.

CPU	Log	- PL	lotes	Pro	akpoints		III Memo	ry Man		Call S	tack	SEH	0	Script	🔮 Symbols	<>> Source	P Refer
	ZLUg							i y Map							- Symbols		
RIP	1		000007			EB C	6					.&Virtua	TATIO	oc>		VirtualAl	100
			000007			90 90				nop							
			000007			90				nop							
			000007			90				nop							
			000007			90				nop							
			000007			90				nop							
	_ ,	• 000	000007	68F67A	باب 8	FF25	7A6B0	008		imp		d ptr ds	:[<mark><&</mark> \	/irtual	A110c>1	JMP.&Virt	ualAlloc
		000	000007	6BF67A	E	90				nop	· · ·		-		-		
		• 000	000007	6BF 67 A	F	90				nop							
		• 200								1						1	
		٩ 🗉															
<jmp.&virtua< th=""><th>a1A11c</th><th>)C></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></jmp.&virtua<>	a1A11c)C>															
										_							
.text:000000	DOO76E	SF67A0	kerne	32.dl	1:\$167	AO #:	LSFA0	<virti< th=""><th>JAIAI</th><th>10C></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></virti<>	JAIAI	10C>							
		-	- m	-	-			_	0.0			In the state	6)	7		
Dump 1	😓 Du	ump 2	😓 Du	mp 3	😓 Du	mp 4	9-0 D	ump 5		Watd	n 1	[x=] Locals	1	Struct			
Address		Нех									ASCI	I					
00000000018									00 0								
00000000018																	
00000000018			00 00														
00000000018												/	····				
00000000018									00 0								
000000000000000000000000000000000000000																	
0000000000018																	
0000000000018									00 0								
000000000018																	
000000000018																	
00000000018									00 0								
00000000018	300C0	50 45	00 00	64 86	04 00	8D	BA 81	65 00	00 0	0 00							
00000000018	BOODO	00 00	00 00	F0 00	22 20	OB	02 OE					ð."&					
00000000018																	
00000000018			00 80														
00000000018																	
00000000018											.ð						
0000000018																	
00000000018	30130	00 00	10 00	00 00	00 00	00	10 00	00 00	00 0	0 00							
						- ·		4 1 1		.1.5		alchat					

Figure 4: Unpacking Qakbot

On further unpacking, we got the final Qakbot payload which loads from memory while executing. Some security researchers <u>found</u> that in the new campaign, Qakbot uses AES encryption to encrypt and store victim information but the final payload we got was the usual Qakbot payload with the same RC4 encryption.

Name		000c DWORD	00025ad2	Hex	acledit.dll
Race		0010 DWORD	0000001]	
Show v	alid				
Ordinal	RVA	Name			_
0001	00001000	00025ade Edit	OwnerInfo		

2 Dump 1 🛛 2	Dump 2	2	Dur	mp 3		Z	Dum	np 4		2	Dum	p 5	4	2 v	Vatd	h 1	Discol	s d
Address	Hex															ASCII	Ι	
000000001C0000	0 4D 5	5A 90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZ		.ÿÿ
0000000001C0001	0 B8 0	00 00	00	00	00	00	00	40	00	00	00	00	00	00	00		@	
000000001C0002	0 00 0	00 00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000001C0003	0 00 0	00 00	00	00	00	00	00	00	00	00	00	00	01	00	00			
0000000001C0004	0 0E 1	1F BA	0E	00	Β4	09	CD	21	B8	01	4C	CD	21	54	68	°	.´.Í!	LÍ!Th
000000001C0005	0 69 7	73 20	70	72	6F	67	72	61	6D	20	63	61	6E	6E	6F		rogram	
0000000010000	0 74 2	20 62	65	20	72	75	6E	20	69	6E	20	44	4F	53	20	t be	run in	DOS
0000000001C0007	0 6D 6	6F 64	65	2E	OD	OD	0A	24	00	00	00	00	00	00	00	mode.		
000000001C0008	0 6D E	EB 1A	9F	29	8A	74	CC	29	8A	74	CC	29	8A	74	CC).tl).t	
000000001C0009	0 62 F	F2 72	CD	28	8A	74	CC	62	F2	75	CD	25	8A	74	CC		(.tÌbòu	
0000000001C000A	0 32 1	17 E8	CC	2A	8A	74	CC	29	8A	75	CC	49	8A	74	CC	2.èÌ:	€.tÌ).u	ÌI.tÌ
000000001C000E		F2 70	CD	2A	8A	74	CC	11	0A	70	CD	3B	8A	74	CC	bòpí	°.tÌp	í;.tÌ
000000001C0000		DA 71	CD	2D	8A	74	CC	11	0A	7D	CD	6A	8A		CC		t1}	
000000001C000D	0 11 0	DA 74	CD	28	8A	74	CC	11	0A	76	CD	28	8A	74	CC	tÍ((.t]v	Í(.tÌ
000000001C000E	0 52 6	69 63	68	29	8A	74	CC	00	00	00	00	00	00	00	00	Rich)).tl	
0000000001C000F	0 00 0	00 00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000001C0010		45 00	00	64	86	05	00	37	59	81	65	00	00	00	00	PE0	d7Y.	e
0000000001C0011		00 00	00	FO	00	22	20	OB	02	OE	25	00	F8	01	00	5	ð."s	%.ø
0000000001C0012		48 00	00	00	00	00	00	20	10	00	00	00	10	00	00			
000000001C0013		00 CO	01	00	00	00	00	00	10	00	00	00	02	00	00	A		
0000000001C0014		00 00	00	00	00	00	00	06	00	00	00	00	00	00	00			
0000000001C0015		EO 02	00	00	04	00	00	00	00	00	00	02	00	20	00	.a		
0000000001C0016		00 10	00	00	00	00	00	00	10	00	00	00	00	00	00			
0000000001C0017		00 10	00	00	00	00	00	00	10	00	00	00	00	00	00			
0000000010018		00 00	00	10	00	00	00	A0	5A	02	00	4C	00	00	00		Z.	
000000001C0019		5A 02	00	8C	00	00	00	00	00	00	00	00	00	00	00	ìΖ,		
0000000001C001A		80 02	00	DC	14	00	00	00	00	00	00	00	00	00	00	.°l	J	
0000000001C001E		00 02	00	44	00	00	00	10	45	02	00	38	00	00	00	.ĐD)E.	.8
00000000100010		00 00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000001C001D		00 00	00	00	00	00	00	00	00	00	00	00	00	00	00			
0000000001C001E		00 00	00	00	00	00	00	00	10	02	00	90	02	00	00			
0000000001C001F		00 00	00	00	00	00	00	00	00	00	00	00	00	00	00			
0000000010020		00 00	00	00	00	00	00	2E	74	65	78	74	00	00	00		te	xt
000000001C0021		F6 01	00	00	10	00	00	00	F8	01	00	00	04	00	00	.ö	ø.	
000000000100027	0 00 0	00 00	00	00	00	00	00	00	00	00	00	20	00	00	60			

Figure 5: Final Qakbot payload

On dynamic analysis, the MSI drops an installer temp file which passes the command line to invoke rundll32.exe and hides the window to run in background.

GPU Graph	Threads	TCP/IP	Security	Environmen	t Strings	
Image	Performance	Perf	ormance Grap	h Diska	and Network	
-Image File						
9	File that lau	inches anot	her file			
	Caphyon LT	D				
Version:	21.2.2.0					
Build Time	e: Wed Nov 1	5 19:23:34	2023			
Path:						
C:\Windows\Installer\MSI5895.tmp Explore						
Command line:						
"C:\Windows\Installer\MSI5895.tmp" /HideWindow rundll32						

Figure 6: MSI installer

Since the threat actor uses PDF in their kill chain, the malicious DLL self copies itself in the name of **AcrobatAC.dll** and passes the command line arguments to execute the DLL with Qakbot export function **EditOwnerInfo**.

TCP/IF	,	Security	Environment	St	trings			
Image	Performance	Performance Graph	Disk and Network	GPU Graph	Threads			
Image File	Image File							
Windows host process (Rundll32)								
Microsoft Corporation								
Version:	6.1.7600.1638	5						
Build Time:	Tue Jul 14 05:2	27:20 2009						
Path:								
C:\Windows\System32\rundll32.exe Explore								
Command line:								

Figure 7: Malicious DLL running on background

It showed the dummy Acrobat window and fake error window as a decoy. Further we found that the malicious DLL invokes the *wermgr.exe* – Windows Error Manager in suspended state to pursue its kill chain.

Hobe AC S X				
Adobe Acrobat				×
Cannot load library:	mfc140.dll e	rror 0x00000502		
			ОК	
E 📳 msiexec.exe	0.01	11,220 K	24,784 K	796 Windows®installer
msiexec.exe		2,036 K	7,064 K	400 Windows®installer
msiexec.exe		2,228 K	8,536 K	2648 Windows®installer
MSI5895.tmp		1,696 K	7,008 K	2324 File that launches another file
🖃 📄 rundll32.exe		1,108 K	3,712 K	1500 Windows host process (Rundll32)
🖃 📄 rundll32.exe		2,740 K	5,284 K	2160 Windows host process (Rundll32)
wermgr.exe	Suspended	500 K	252 K	1096 Windows Problem Reporting
	F :	0. Decov and	in alting the	

Figure 8: Decoy and invoking wermgr.exe

We dumped the PE file from *wermgr.exe* which was our previously unpacked final Qakbot payload. The threat actor implied **Process Hollowing** technique to inject malicious code into the suspended process of Windows Error Manager.

4	🛿 wermgr.exe (8032) Pr	operties		_		×
	Handles	GPU	Disk and Network	Co	mment	
	Control Chattation	Defenses The	Talan Madalan	Mamory	En la com	

Figure 9: Process Hollowing wermgr.exe

As mentioned earlier, the *wermgr.exe* creates a registry key with RC4 encrypted data of victim system information, timestamp of installation and C2 information which is a <u>usual</u> Qakbot TTP.

ᅒ Ever	nt Properti	es		_	×
Event	Process	Stack			
Date:		12/25/2023 2:29:20.8482310 PM			
Threa	ad:	2708			
Class	:	Registry			
Opera	ation:	RegCreateKey			
Resul	lt:	SUCCESS			
Path:		HKCU\Software\Classes\jaipdhvkg	ju		
Durat	tion:	0.0000718			
Gran	red Access: ited Access osition:	None 0x0			^

^	Name	Туре	Data
	ab (Default)	REG_SZ	(value not set)
	88 4a7c9f16	REG_BINARY	46 ab 1a 44 63 37 1e ae 25 df 18 9f 1c 00 41 16 50 d2 b4 97
	👪 4bfbc291	REG_BINARY	a4 c2 c8 73 cf 57 24 e7 95 83 c6 e2 28 15 16 e3 06 ca d4 39
	腿 5533843d	REG_BINARY	c4 0e 56 eb fe 61 e5 67 12 4f e3 1b 8c cc 6c be d5 2d 46 c
	👪 86d69f88	REG_BINARY	45 0c 03 4e 4a ca e2 8e 54 a1 2b 41 20 98 3c 02 7f e0 91 1c
	8751c20f	REG_BINARY	67 50 5d 39 2c fd 88 18 b2 0c 6d 27 e6 34 73 b2 21 e0 8d c
	🕫 981ed924	REG_BINARY	e4 aa 74 6c 69 e3 78 f7 89 a1 83 81 4c 8e df 17 5c 3b 80 1b
	👯 999984a3	REG_BINARY	85 18 ce d5 53 1c af f3 f0 c2 6d b2 ac 82 64 fd 01 5c 1c e8
	腿 d0fed740	REG_BINARY	45 83 62 5b 3f 46 1d 9e 05 40 b3 26 da 5a 5d a2 be aa 2b 6
	👪 d1798ac7	REG_BINARY	c4 08 2f fe 37 77 3a 40 6e e1 5a 28 c4 51 e3 b4 d0 54 11 31

Figure 10: Creates registry key

Qakbot tries to make a C2 connection in the background when the victim believes *wermgr.exe* is running. Since the C2 was down at the time of analysis, it was unable to establish a connection for carrying out any further malicious activity.

Wwermgr.exe	8032 📥 TCP Connect
wermgr.exe	8032 📥 TCP Send
wermgr.exe	8032 🔬 TCP Receive
wermgr.exe	8032 🔬 TCP Receive
👰 wermgr.exe	8032 📥 TCP Receive
Figure 11, C2 a	"

Figure 11: C2 connection by wermgr.exe

We at K7 Labs provide detection against latest threats and also for this newer variant of Qakbot. Users are advised to use a reliable security product such as "K7 Total Security" and keep it up-to-date so as to safeguard their devices.

loCs

Hash	Detection Name
723DAE8ED3F157E40635681F028328E6	Backdoor (005af9cf1)
88BBF2A743BAAF81F7A312BE61F90D76	Backdoor (005af9cf1)

References