Blackwood APT Group Has a New DLL Loader

blog.sonicwall.com/en-us/2024/01/blackwood-apt-group-has-a-new-dll-loader/

Security News

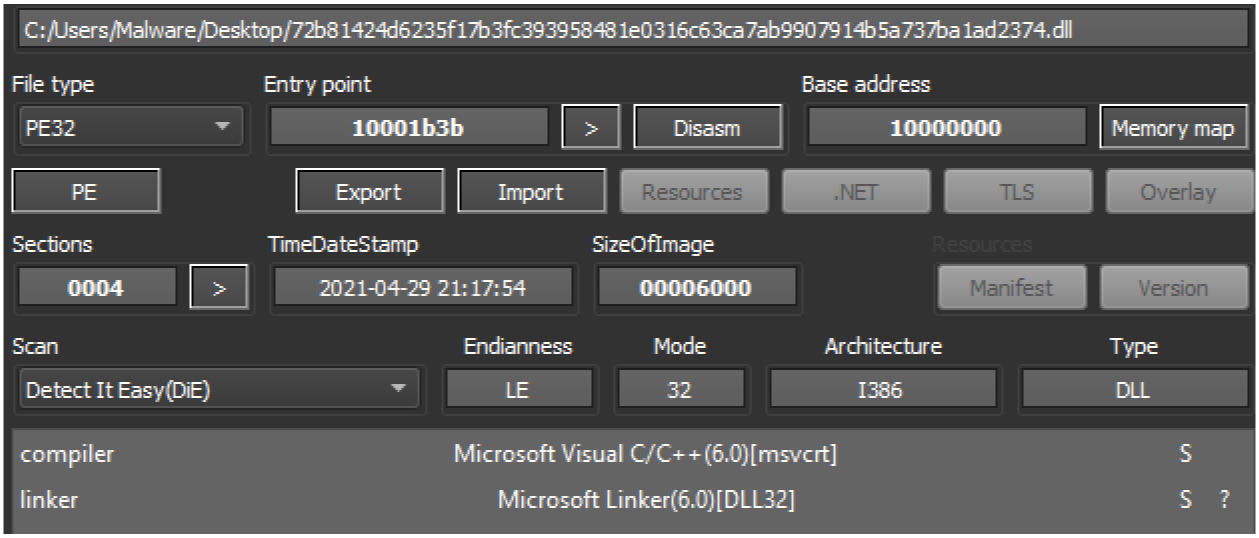
January 29, 2024

Overview

This week, the SonicWall Capture Labs threat research team analyzed a sample tied to the Blackwood APT group. This is a DLL that, when loaded onto a victim's computer, will escalate privileges and attempt to install a backdoor for communications monitoring and diversion. It has evasive capabilities and, as of this writing, is targeting companies and individuals in Japan and China.

Technical Overview

The sample is detected as a 32-bit DLL (Figure 1) with no packer or protector. It has minimal strings and no obvious obfuscation or encryption.



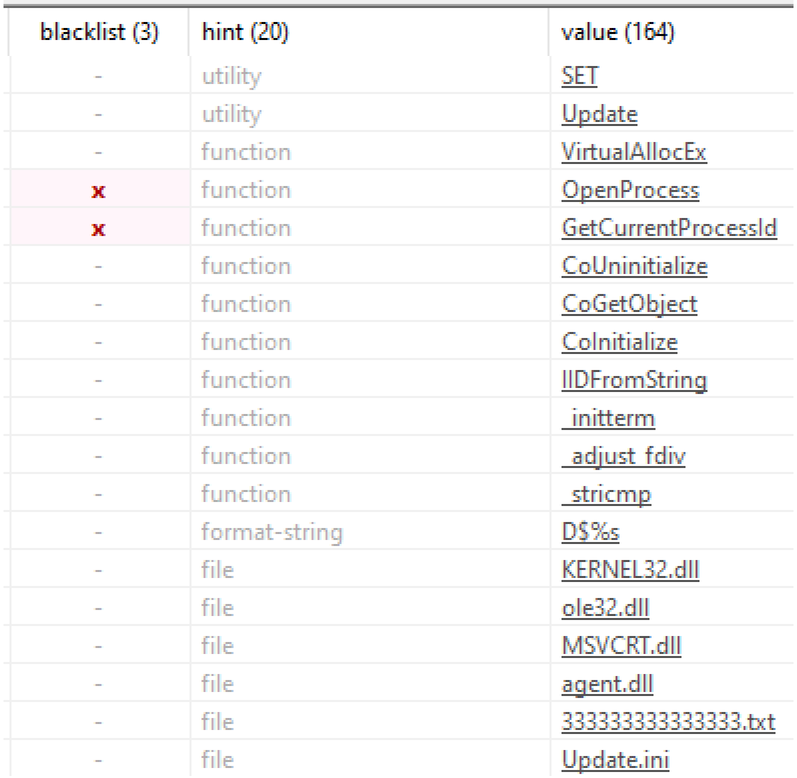
C:/Users/Malware/Desktop/72b81424d6235f17b3fc393958481e0316c63ca7ab9907914b5a737ba1ad2374.dll

File type: PE32
Entry point: 10001b3b
Base address: 10000000
Sections: 0004
TimeDateStamp: 2021-04-29 21:17:54
SizeOfImage: 00006000
Scan: Detect It Easy(DiE)
Endianness: LE
Mode: 32
Architecture: I386
Type: DLL

compiler	Microsoft Visual C/C++(6.0)[msvcrt]	S
linker	Microsoft Linker(6.0)[DLL32]	S ?

Figure 1: Sample detection

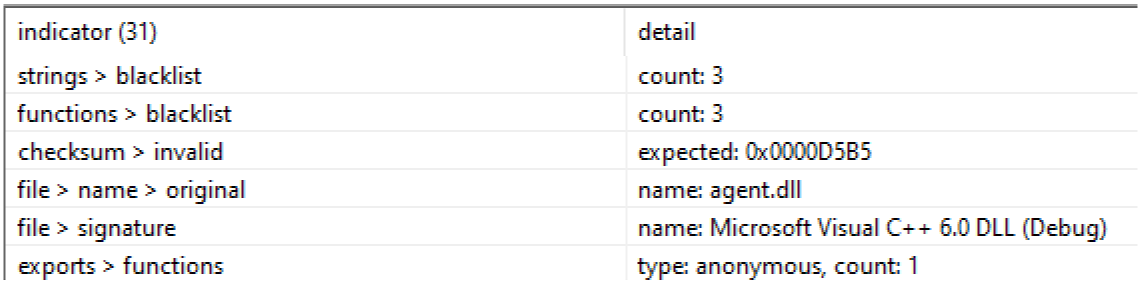
Strings show several API calls of concern, including GetCurrentProcessID, OpenProcess and VirtualAlloc – all of which are used to load malicious DLLs into memory. There are also two files listed: '333333333333333.txt' and 'Update.ini', as shown in Figure 2.



blacklist (3)	hint (20)	value (164)
-	utility	SET
-	utility	Update
-	function	VirtualAllocEx
x	function	OpenProcess
x	function	GetCurrentProcessId
-	function	CoUninitialize
-	function	CoGetObject
-	function	CoInitialize
-	function	IIDFromString
-	function	_initterm
-	function	_adjust_fdiv
-	function	_stricmp
-	format-string	D$%s
-	file	KERNEL32.dll
-	file	ole32.dll
-	file	MSVCRT.dll
-	file	agent.dll
-	file	333333333333333.txt
-	file	Update.ini

Figure 2: Static string detection

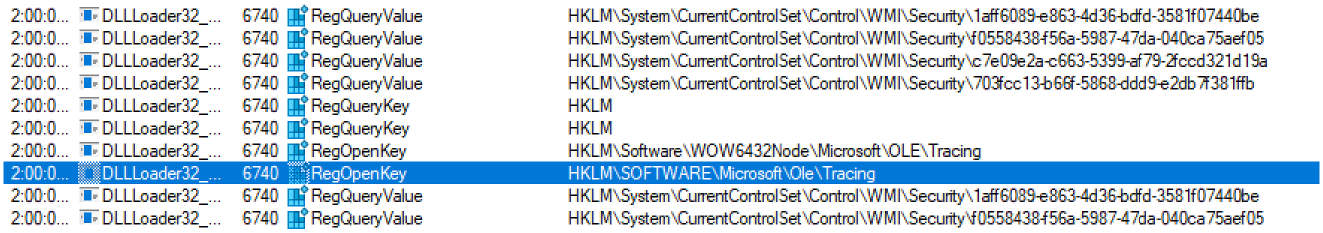
The name of the file is shown as 'agent.dll' (Figure 3) and there is one anonymous export that is only shown as an ordinal value when looking at the file with multiple tools.



indicator (31)	detail
strings > blacklist	count: 3
functions > blacklist	count: 3
checksum > invalid	expected: 0x0000D5B5
file > name > original	name: agent.dll
file > signature	name: Microsoft Visual C++ 6.0 DLL (Debug)
exports > functions	type: anonymous, count: 1

Figure 3: Original name and anonymous export

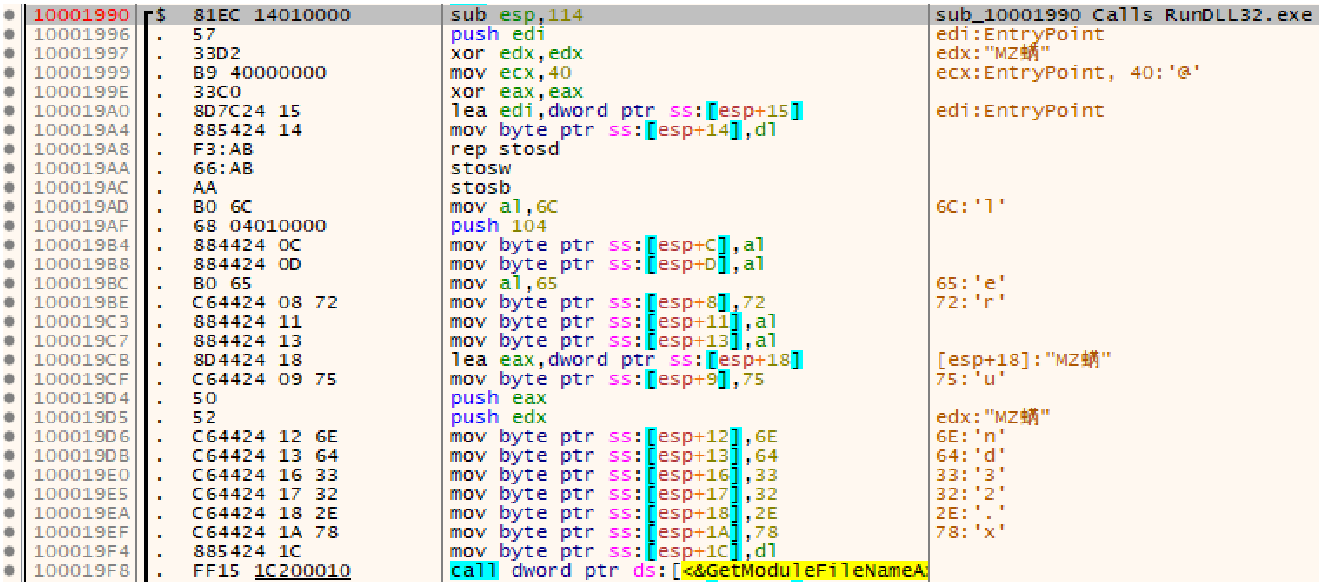
When dynamically analyzing the sample, it has multiple anti-analysis capabilities that prevent most of its function from being observed. It will look for debuggers, processor features and security settings in the registry (Figure 3). There are also locale checks that, when failed, will kill the process.



```
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\1aff6089-e863-4d36-bdfd-3581f07440be
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\f0558438-f56a-5987-47da-040ca75aef05
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\c7e09e2a-c663-5399-af79-2fccd321d19a
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\703fcc13-b66f-5868-ddd9-e2db7f381ffb
2:00:0...  DLLLoader32_...  6740  RegQueryKey    HKLM
2:00:0...  DLLLoader32_...  6740  RegQueryKey    HKLM
2:00:0...  DLLLoader32_...  6740  RegOpenKey     HKLM\Software\WOW6432Node\Microsoft\OLE\Tracing
2:00:0...  DLLLoader32_...  6740  RegOpenKey     HKLM\SOFTWARE\Microsoft\Ole\Tracing
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\1aff6089-e863-4d36-bdfd-3581f07440be
2:00:0...  DLLLoader32_...  6740  RegQueryValue  HKLM\System\CurrentControlSet\Control\WMI\Security\f0558438-f56a-5987-47da-040ca75aef05
```

Figure 4: WMI registry keys being queried for security checks

The anonymous export at address 0x10001A70 is the file calling 'Rundll32.exe' for process injection, as shown in Figure 5.

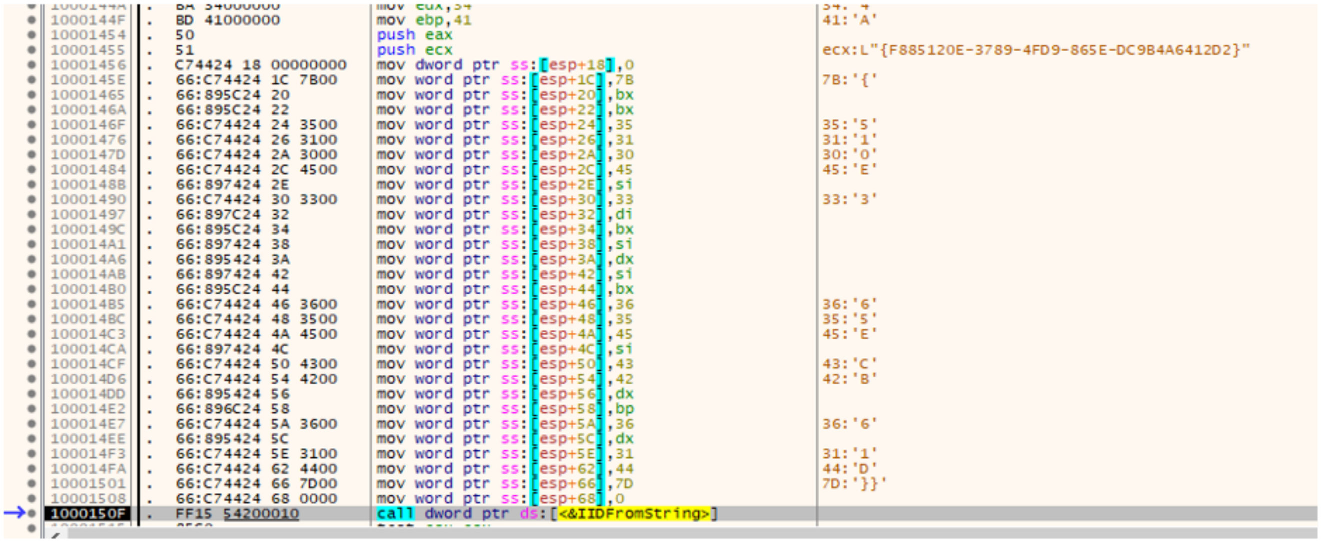


```
10001990  $  81EC 14010000   sub esp,114                              sub_10001990 Calls RunDLL32.exe
10001996  .  57              push edi                                 edi:EntryPoint
10001997  .  33D2            xor edx,edx                              edx:"MZ蛃"
10001999  .  B9 40000000     mov ecx,40                               ecx:EntryPoint, 40:'@'
1000199E  .  33C0            xor eax,eax
100019A0  .  8D7C24 15       lea edi,dword ptr ss:[esp+15]            edi:EntryPoint
100019A4  .  885424 14       mov byte ptr ss:[esp+14],dl
100019A8  .  F3:AB           rep stosd
100019AA  .  66:AB           stosw
100019AC  .  AA              stosb
100019AD  .  B0 6C           mov al,6C                                6C:'l'
100019AF  .  68 04010000     push 104
100019B4  .  884424 0C       mov byte ptr ss:[esp+C],al
100019B8  .  884424 0D       mov byte ptr ss:[esp+D],al
100019BC  .  B0 65           mov al,65                                65:'e'
100019BE  .  C64424 08 72    mov byte ptr ss:[esp+8],72               72:'r'
100019C3  .  884424 11       mov byte ptr ss:[esp+11],al
100019C7  .  884424 13       mov byte ptr ss:[esp+13],al
100019CB  .  8D4424 18       lea eax,dword ptr ss:[esp+18]            [esp+18]:"MZ蛃"
100019CF  .  C64424 09 75    mov byte ptr ss:[esp+9],75               75:'u'
100019D4  .  50              push eax
100019D5  .  52              push edx                                 edx:"MZ蛃"
100019D6  .  C64424 12 6E    mov byte ptr ss:[esp+12],6E              6E:'n'
100019DB  .  C64424 13 64    mov byte ptr ss:[esp+13],64              64:'d'
100019E0  .  C64424 16 33    mov byte ptr ss:[esp+16],33              33:'3'
100019E5  .  C64424 17 32    mov byte ptr ss:[esp+17],32              32:'2'
100019EA  .  C64424 18 2E    mov byte ptr ss:[esp+18],2E              2E:'.'
100019EF  .  C64424 1A 78    mov byte ptr ss:[esp+1A],78              78:'x'
100019F4  .  885424 1C       mov byte ptr ss:[esp+1C],dl
100019F8  .  FF15 1C200010   call dword ptr ds:[<&GetModuleFileNameA:
```

Figure 5: Export address calls sub_10001990, which creates 'rundll32.exe'

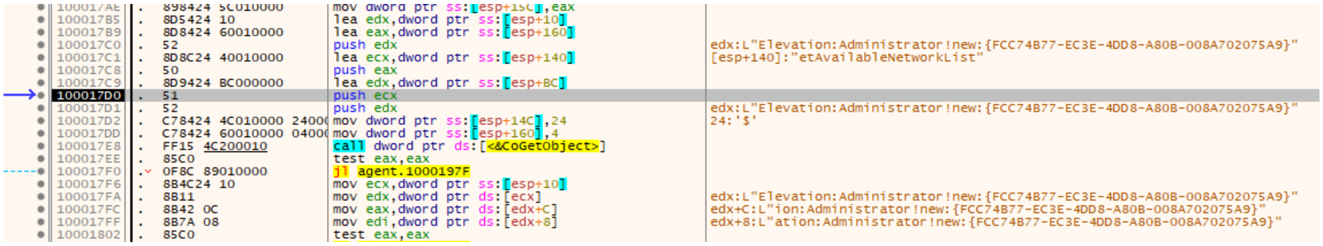
Controlling the program's execution allows the check for a UAC bypass to be generated. The DLL will attempt to escalate privileges via CMSTPLUA interface[1]. The following strings are created, as shown in Figures 5 and 6:

- Elevation:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}
- Elevation:Administrator!new:{F885120E-3789-4FD9-865E-DC9B4A6412D2}



```
1000144F  .  BD 41000000             mov ebp,41                        41:'A'
10001454  .  50                      push eax
10001455  .  51                      push ecx                          ecx:L"{F885120E-3789-4FD9-865E-DC9B4A6412D2}"
10001456  .  C74424 18 00000000      mov dword ptr ss:[esp+18],0
1000145E  .  66:C74424 1C 7B00       mov word ptr ss:[esp+1C],7B       7B:'{'
10001465  .  66:895C24 20            mov word ptr ss:[esp+20],bx
1000146A  .  66:895C24 22            mov word ptr ss:[esp+22],bx
1000146F  .  66:C74424 24 3500       mov word ptr ss:[esp+24],35       35:'5'
10001476  .  66:C74424 26 3100       mov word ptr ss:[esp+26],31       31:'1'
1000147D  .  66:C74424 2A 3000       mov word ptr ss:[esp+2A],30       30:'0'
10001484  .  66:C74424 2C 4500       mov word ptr ss:[esp+2C],45       45:'E'
1000148B  .  66:897424 2E            mov word ptr ss:[esp+2E],si
10001490  .  66:C74424 30 3300       mov word ptr ss:[esp+30],33       33:'3'
10001497  .  66:897C24 32            mov word ptr ss:[esp+32],di
1000149C  .  66:895C24 34            mov word ptr ss:[esp+34],bx
100014A1  .  66:897424 38            mov word ptr ss:[esp+38],si
100014A6  .  66:895424 3A            mov word ptr ss:[esp+3A],dx
100014AB  .  66:897424 42            mov word ptr ss:[esp+42],si
100014B0  .  66:895C24 44            mov word ptr ss:[esp+44],bx
100014B5  .  66:C74424 46 3600       mov word ptr ss:[esp+46],36       36:'6'
100014BC  .  66:C74424 48 3500       mov word ptr ss:[esp+48],35       35:'5'
100014C3  .  66:C74424 4A 4500       mov word ptr ss:[esp+4A],45       45:'E'
100014CA  .  66:897424 4C            mov word ptr ss:[esp+4C],si
100014CF  .  66:C74424 50 4300       mov word ptr ss:[esp+50],43       43:'C'
100014D6  .  66:C74424 54 4200       mov word ptr ss:[esp+54],42       42:'B'
100014DD  .  66:895424 56            mov word ptr ss:[esp+56],dx
100014E2  .  66:896C24 58            mov word ptr ss:[esp+58],bp
100014E7  .  66:C74424 5A 3600       mov word ptr ss:[esp+5A],36       36:'6'
100014EE  .  66:895424 5C            mov word ptr ss:[esp+5C],dx
100014F3  .  66:C74424 5E 3100       mov word ptr ss:[esp+5E],31       31:'1'
100014FA  .  66:C74424 62 4400       mov word ptr ss:[esp+62],44       44:'D'
10001501  .  66:C74424 66 7D00       mov word ptr ss:[esp+66],7D       7D:'}}'
10001508  .  66:C74424 68 0000       mov word ptr ss:[esp+68],0
1000150F  .  FF15 54200010           call dword ptr ds:[<&IIDFromString>]
```

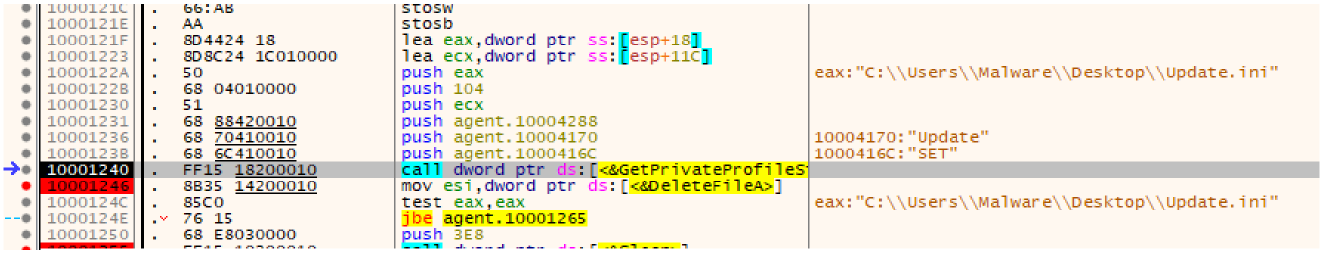
[1] https://gist.github.com/hfiref0x/196af729106b780db1c73428b5a5d68d



```
100017B5  .  8D5424 10               lea edx,dword ptr ss:[esp+10]
100017B9  .  8D8424 60010000         lea eax,dword ptr ss:[esp+160]
100017C0  .  52                      push edx                 edx:L"Elevation:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}"
100017C1  .  8D8C24 40010000         lea ecx,dword ptr ss:[esp+140]   [esp+140]:"etAvailableNetworkList"
100017C8  .  50                      push eax
100017C9  .  8D9424 BC000000         lea edx,dword ptr ss:[esp+BC]
100017D0  .  51                      push ecx
100017D1  .  52                      push edx                 edx:L"Elevation:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}"
100017D2  .  C78424 4C010000 24000   mov dword ptr ss:[esp+14C],24    24:'$'
100017DD  .  C78424 60010000 04000   mov dword ptr ss:[esp+160],4
100017E8  .  FF15 4C200010           call dword ptr ds:[<&CoGetObject>]
100017EE  .  85C0                    test eax,eax
100017F0  .  0F8C 89010000           jl agent.1000197F
100017F6  .  8B4C24 10               mov ecx,dword ptr ss:[esp+10]
100017FA  .  8B11                    mov edx,dword ptr ds:[ecx]       edx:L"Elevation:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}"
100017FC  .  8B42 0C                 mov eax,dword ptr ds:[edx+C]     edx+C:L"ion:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}"
100017FF  .  8B7A 08                 mov edi,dword ptr ds:[edx+8]     edx+8:L"ation:Administrator!new:{FCC74B77-EC3E-4DD8-A80B-008A702075A9}"
10001802  .  85C0                    test eax,eax
```

Figures 6 (top) and 7 (bottom): A function creates GUIDs for privilege escalation

The two files that are listed within the strings are also referenced during runtime (Figure 7), but despite multiple attempts at controlling execution, the files were not observed on test systems.



```
1000121E  .  AA                      stosb
1000121F  .  8D4424 18               lea eax,dword ptr ss:[esp+18]
10001223  .  8D8C24 1C010000         lea ecx,dword ptr ss:[esp+11C]
1000122A  .  50                      push eax                          eax:"C:\\Users\\Malware\\Desktop\\Update.ini"
1000122B  .  68 04010000             push 104
10001230  .  51                      push ecx
10001231  .  68 88420010             push agent.10004288
10001236  .  68 70410010             push agent.10004170               10004170:"Update"
1000123B  .  68 6C410010             push agent.1000416C               1000416C:"SET"
10001240  .  FF15 18200010           call dword ptr ds:[<&GetPrivateProfileS
10001246  .  8B35 14200010           mov esi,dword ptr ds:[<&DeleteFileA>]
1000124C  .  85C0                    test eax,eax                      eax:"C:\\Users\\Malware\\Desktop\\Update.ini"
1000124E  .  76 15                   jbe agent.10001265
10001250  .  68 E8030000             push 3E8
```

Figure 8: Update.ini is referenced but never created

Protection

To ensure SonicWall customers are prepared for any exposure that may occur due to this malware, the following signatures have been released:

MalAgent.Blackwood

IOCs

72B81424D6235F17B3FC393958481E0316C63CA7AB9907914B5A737BA1AD2374

Security News

The SonicWall Capture Labs Threat Research Team gathers, analyzes and vets cross-vector threat information from the SonicWall Capture Threat network, consisting of global devices and resources, including more than 1 million security sensors in nearly 200 countries and territories. The research team identifies, analyzes, and mitigates critical vulnerabilities and malware daily through in-depth research, which drives protection for all SonicWall customers. In addition to safeguarding networks globally, the research team supports the larger threat intelligence community by releasing weekly deep technical analyses of the most critical threats to small businesses, providing critical knowledge that defenders need to protect their networks.