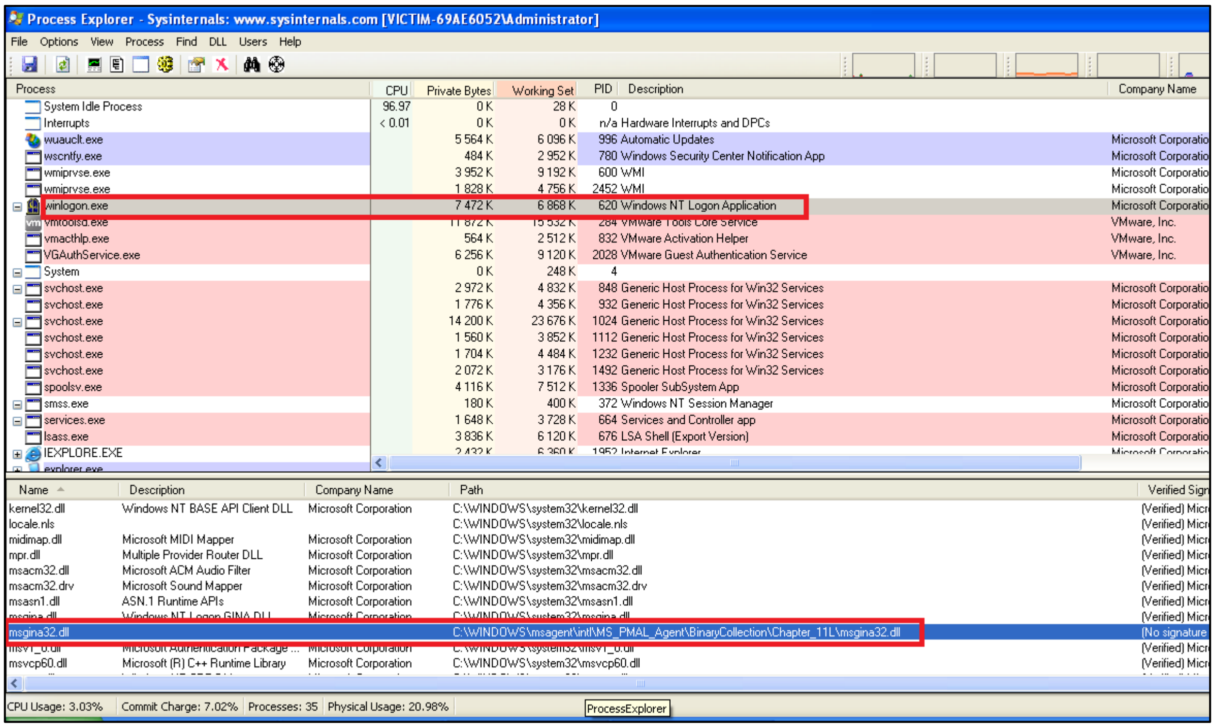
#Malware 2

Start looking at the Process Explorer, look for unusual processes.



Process Explorer - Sysinternals: www.sysinternals.com [VICTIM-69AE6052\Administrator]

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
System Idle Process	96.97	0 K	28 K	0		
Interrupts	< 0.01	0 K	0 K	n/a	Hardware Interrupts and DPCs	
wuauclt.exe		5 564 K	6 096 K	996	Automatic Updates	Microsoft Corporatio
wscntfy.exe		484 K	2 952 K	780	Windows Security Center Notification App	Microsoft Corporatio
wmiprvse.exe		3 952 K	9 192 K	600	WMI	Microsoft Corporatio
wmiprvse.exe		1 828 K	4 756 K	2452	WMI	Microsoft Corporatio
winlogon.exe		7 472 K	6 868 K	620	Windows NT Logon Application	Microsoft Corporatio
vmtoolsd.exe		11 872 K	15 532 K	284	VMware Tools Core Service	VMware, Inc.
vmacthlp.exe		564 K	2 512 K	832	VMware Activation Helper	VMware, Inc.
VGAuthService.exe		6 256 K	9 120 K	2028	VMware Guest Authentication Service	VMware, Inc.
System		0 K	248 K	4		
svchost.exe		2 972 K	4 832 K	848	Generic Host Process for Win32 Services	Microsoft Corporatio
svchost.exe		1 776 K	4 356 K	932	Generic Host Process for Win32 Services	Microsoft Corporatio
svchost.exe		14 200 K	23 676 K	1024	Generic Host Process for Win32 Services	Microsoft Corporatio
svchost.exe		1 560 K	3 852 K	1112	Generic Host Process for Win32 Services	Microsoft Corporatio
svchost.exe		1 704 K	4 484 K	1232	Generic Host Process for Win32 Services	Microsoft Corporatio
svchost.exe		2 072 K	3 176 K	1492	Generic Host Process for Win32 Services	Microsoft Corporatio
spoolsv.exe		4 116 K	7 512 K	1336	Spooler SubSystem App	Microsoft Corporatio
smss.exe		180 K	400 K	372	Windows NT Session Manager	Microsoft Corporatio
services.exe		1 648 K	3 728 K	664	Services and Controller app	Microsoft Corporatio
lsass.exe		3 836 K	6 120 K	676	LSA Shell (Export Version)	Microsoft Corporatio
IEXPLORE.EXE		2 432 K	6 360 K	1952	Internet Explorer	Microsoft Corporatio
explorer.exe						

Name	Description	Company Name	Path	Verified Sign
kernel32.dll	Windows NT BASE API Client DLL	Microsoft Corporation	C:\WINDOWS\system32\kernel32.dll	(Verified) Micr
locale.nls			C:\WINDOWS\system32\locale.nls	(Verified) Micr
midimap.dll	Microsoft MIDI Mapper	Microsoft Corporation	C:\WINDOWS\system32\midimap.dll	(Verified) Micr
mpr.dll	Multiple Provider Router DLL	Microsoft Corporation	C:\WINDOWS\system32\mpr.dll	(Verified) Micr
msacm32.dll	Microsoft ACM Audio Filter	Microsoft Corporation	C:\WINDOWS\system32\msacm32.dll	(Verified) Micr
msacm32.drv	Microsoft Sound Mapper	Microsoft Corporation	C:\WINDOWS\system32\msacm32.drv	(Verified) Micr
msasn1.dll	ASN.1 Runtime APIs	Microsoft Corporation	C:\WINDOWS\system32\msasn1.dll	(Verified) Micr
msgina.dll	Windows NT Logon GINA DLL	Microsoft Corporation	C:\WINDOWS\system32\msgina.dll	(Verified) Micr
msgina32.dll			C:\WINDOWS\msagent\intl\MS_PMAL_Agent\BinaryCollection\Chapter_11L\msgina32.dll	(No signature
msv1_0.dll	Microsoft Authentication Package ...	Microsoft Corporation	C:\WINDOWS\system32\msv1_0.dll	(Verified) Micr
msvcp60.dll	Microsoft (R) C++ Runtime Library	Microsoft Corporation	C:\WINDOWS\system32\msvcp60.dll	(Verified) Micr

CPU Usage: 3.03% | Commit Charge: 7.02% | Processes: 35 | Physical Usage: 20.98%

This DLL is not usual. All the DLLs are stored in C:\WINDOWS\system32\, and msgina32.dll is not in this folder.

This behaviour is typical from a Hook.

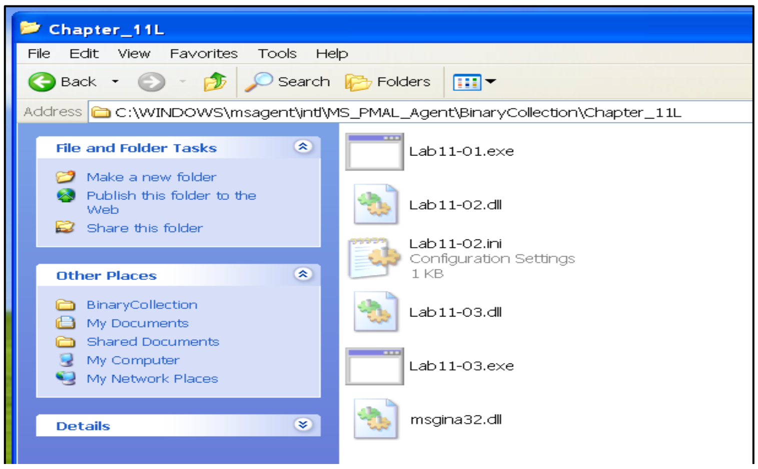
What is a hook?

A hook intercepts the function calls or events between software components. For example, the hook will write in a file all the events of a software and send this file to the hacker (via Internet).

Where is the Malware located?

Lab11-01.exe

C:\WINDOWS\msagent\intl\MS_PMAL_Agent\BinaryCollection



Chapter_11L

Address C:\WINDOWS\msagent\intl\MS_PMAL_Agent\BinaryCollection\Chapter_11L

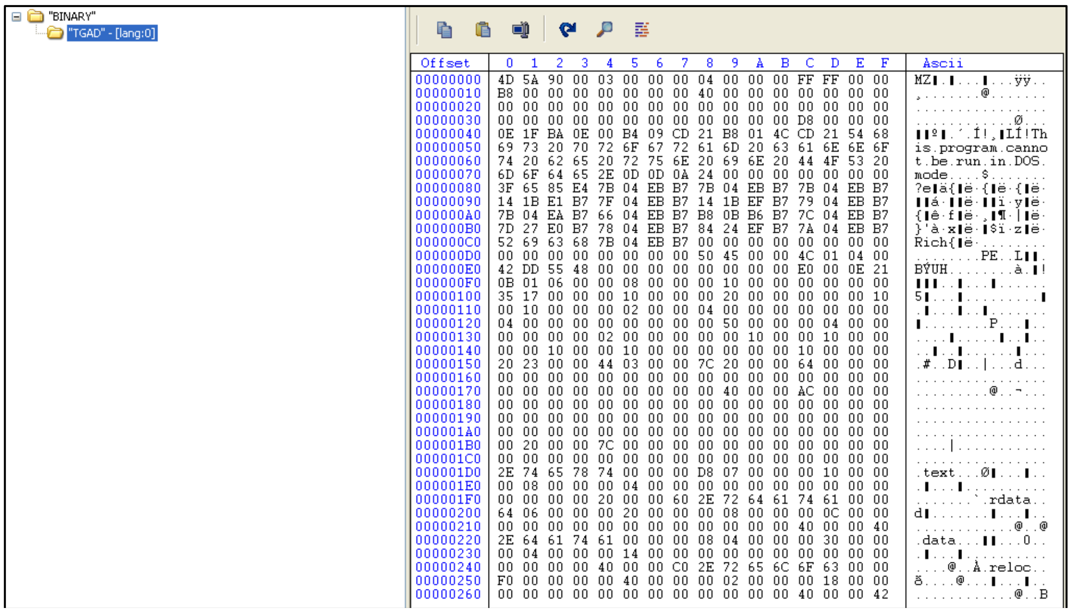
File and Folder Tasks
- Make a new folder
- Publish this folder to the Web
- Share this folder

Other Places
- BinaryCollection
- My Documents
- Shared Documents
- My Computer
- My Network Places

Details

- Lab11-01.exe
- Lab11-02.dll
- Lab11-02.ini Configuration Settings 1 KB
- Lab11-03.dll
- Lab11-03.exe
- msgina32.dll

Next, with use of Resource Hacker, we identified a Binary file (which is a PE – cf MZ header). We will analyse this file too. And the result of the analyse is in the resource section (below).



"BINARY"
"TGAD" - [lang:0]

```
Offset    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   Ascii
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ▌.▌...▌...ÿÿ..
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸.......@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 D8 00 00 00  ............Ø...
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ▌▌º▌.´.Í!¸▌LÍ!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is.program.canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t.be.run.in.DOS.
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  3F 65 85 E4 7B 04 EB B7 7B 04 EB B7 7B 04 EB B7  ?e▌ä{▌ë·{▌ë·{▌ë·
00000090  14 1B E1 B7 7F 04 EB B7 14 1B EF B7 79 04 EB B7  ▌▌á·▌▌ë·▌▌ï·y▌ë·
000000A0  7B 04 EA B7 66 04 EB B7 B8 0B B6 B7 7C 04 EB B7  {▌ê·f▌ë·¸▌¶·|▌ë·
000000B0  7D 27 E0 B7 78 04 EB B7 84 24 EF B7 7A 04 EB B7  }'à·x▌ë·▌$ï·z▌ë·
000000C0  52 69 63 68 7B 04 EB B7 00 00 00 00 00 00 00 00  Rich{▌ë·........
000000D0  00 00 00 00 00 00 00 00 50 45 00 00 4C 01 04 00  ........PE..L▌▌.
000000E0  42 DD 55 48 00 00 00 00 00 00 00 00 E0 00 0E 21  BÝUH........à.▌!
000000F0  0B 01 06 00 00 08 00 00 00 10 00 00 00 00 00 00  ▌▌▌..▌...▌......
00000100  35 17 00 00 00 10 00 00 00 20 00 00 00 00 00 10  5▌...▌.........▌
00000110  00 10 00 00 00 02 00 00 04 00 00 00 00 00 00 00  .▌...▌..▌.......
00000120  04 00 00 00 00 00 00 00 00 50 00 00 00 04 00 00  ▌........P...▌..
00000130  00 00 00 00 02 00 00 00 00 00 10 00 00 10 00 00  ....▌.....▌..▌..
00000140  00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00  ..▌..▌......▌...
00000150  20 23 00 00 44 03 00 00 7C 20 00 00 64 00 00 00  .#..D▌..|...d...
00000160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000170  00 00 00 00 00 00 00 00 00 40 00 00 AC 00 00 00  .........@..¬...
00000180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000190  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000001A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000001B0  00 20 00 00 7C 00 00 00 00 00 00 00 00 00 00 00  ....|...........
000001C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000001D0  2E 74 65 78 74 00 00 00 D8 07 00 00 00 10 00 00  .text...Ø▌...▌..
000001E0  00 08 00 00 00 04 00 00 00 00 00 00 00 00 00 00  .▌...▌..........
000001F0  00 00 00 00 20 00 00 60 2E 72 64 61 74 61 00 00  .......`.rdata..
00000200  64 06 00 00 00 20 00 00 00 08 00 00 00 0C 00 00  d▌.......▌...▌..
00000210  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 40  ............@..@
00000220  2E 64 61 74 61 00 00 00 08 04 00 00 00 30 00 00  .data...▌▌...0..
00000230  00 04 00 00 00 14 00 00 00 00 00 00 00 00 00 00  .▌...▌..........
00000240  00 00 00 00 40 00 00 C0 2E 72 65 6C 6F 63 00 00  ....@..À.reloc..
00000250  F0 00 00 00 00 40 00 00 00 02 00 00 00 18 00 00  ð....@...▌...▌..
00000260  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 42  ............@..B
```

The malware extracts and drops the file *msgina32.dll* onto disk from a resource section named TGAD.

Next, performed dynamic analysis and monitor the malware with procmon by setting a filter for *Lab11-01.exe*. When launch the malware, it creates a file named *msgina32.dll* on disk in the same directory from which the malware was launched. The malware inserts the path to *msgina32.dll* into the registry key HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\ GinaDLL, so that the DLL will be loaded by Winlogon when the system reboots.

Extracting the TGAD resource section from *Lab11-01.exe* (using Resource Hacker) and comparing it to *msgina32.dll*, found that the two are identical.

What is gina.dll?

GINA stands for Graphical Identification and Authentication. This DLL is loaded by **Winlogon.exe**. The GINA implements the authentication policy of the interactive logon model and is expected to perform all identification and authentication user interactions.

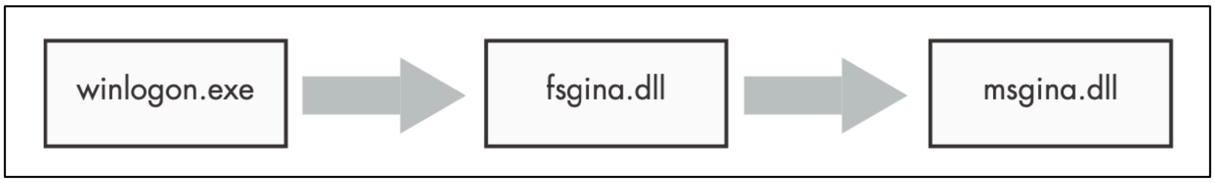
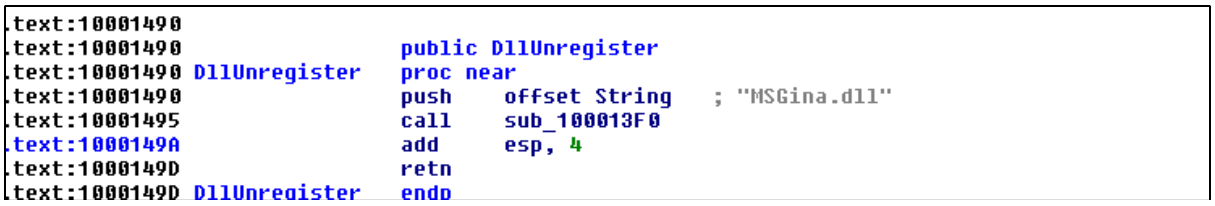


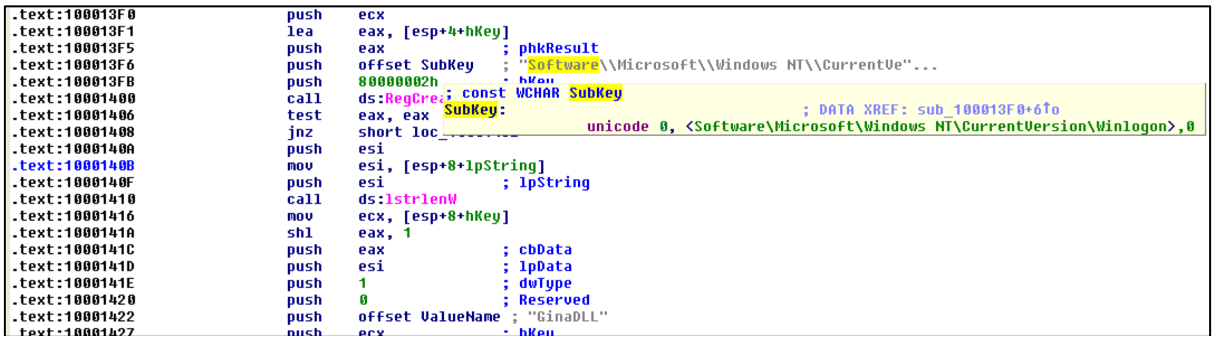
Figure above shows an example of the way that logon credentials flow through a system with a malicious file between Winlogon and *msgina.dll*. The malware (*fsgina.dll*) is able to capture all user credentials submitted to the system for authentication. It can log that information to disk or pass it over the network. Because *fsgina.dll* intercepts the communication between Winlogon and *msgina.dll*, it must pass the credential information on to *msgina.dll* so that the system will continue to operate normally. In order to do so, the malware must contain all DLL exports required by GINA; specifically, it must export more than 15 functions, most of which are prepended with *Wlx*. Clearly, if you find that you are analyzing a DLL with many export functions that begin with the string Wlx, you have a good indicator that you are examining a GINA interceptor.

What does it do?



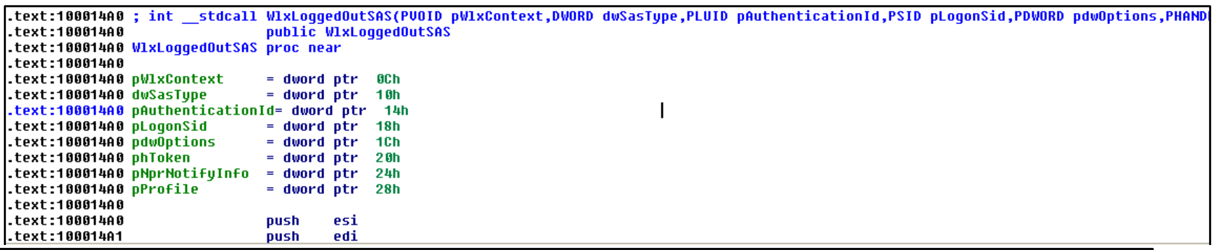
```
.text:10001490
.text:10001490                  public DllUnregister
.text:10001490 DllUnregister    proc near
.text:10001490                  push    offset String   ; "MSGina.dll"
.text:10001495                  call    sub_100013F0
.text:1000149A                  add     esp, 4
.text:1000149D                  retn
.text:1000149D DllUnregister    endp
```

Lab11-01.exe unregister the msgina.dll

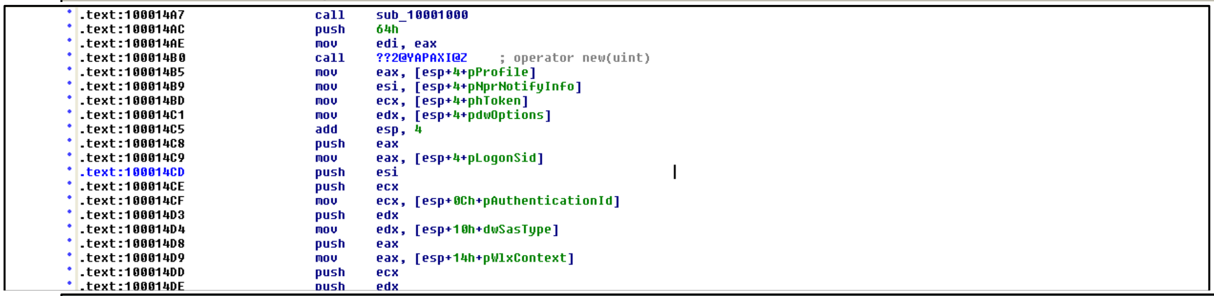


```
.text:100013F0                  push    ecx
.text:100013F1                  lea     eax, [esp+4+hKey]
.text:100013F5                  push    eax             ; phkResult
.text:100013F6                  push    offset SubKey   ; "Software\\Microsoft\\Windows NT\\CurrentVe"...
.text:100013FB                  push    80000002h       ; hKey
.text:10001400                  call    ds:RegCre; const WCHAR SubKey
.text:10001406                  test    eax, eax SubKey:                 ; DATA XREF: sub_100013F0+6↑o
.text:10001408                  jnz     short loc_          unicode 0, <Software\Microsoft\Windows NT\CurrentVersion\Winlogon>,0
.text:1000140A                  push    esi
.text:1000140B                  mov     esi, [esp+8+lpString]
.text:1000140F                  push    esi             ; lpString
.text:10001410                  call    ds:lstrlenW
.text:10001416                  mov     ecx, [esp+8+hKey]
.text:1000141A                  shl     eax, 1
.text:1000141C                  push    eax             ; cbData
.text:1000141D                  push    esi             ; lpData
.text:1000141E                  push    1               ; dwType
.text:10001420                  push    0               ; Reserved
.text:10001422                  push    offset ValueName ; "GinaDLL"
.text:10001427                  push    ecx             ; hKey
```

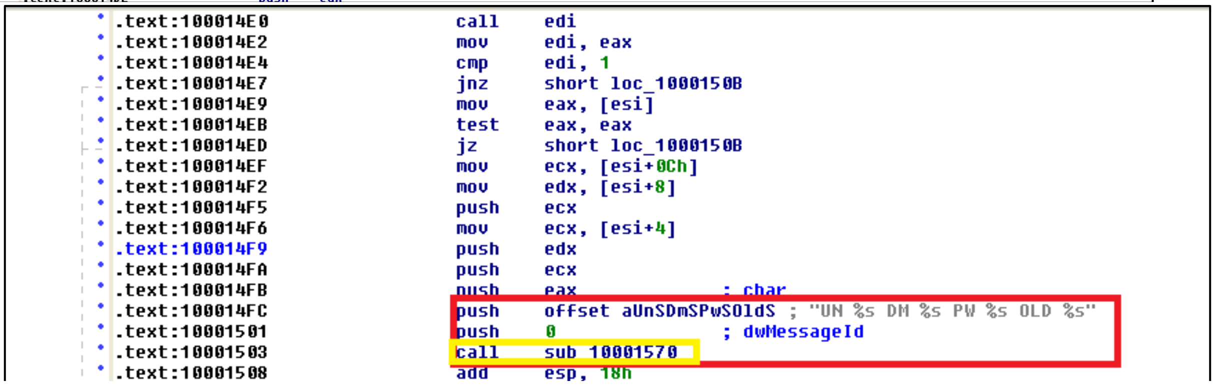
Then, it creates a registry key in Winlogon : **HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon** and names GinaDLL



```
.text:100014A0 ; int __stdcall WlxLoggedOutSAS(PVOID pWlxContext,DWORD dwSasType,PLUID pAuthenticationId,PSID pLogonSid,PDWORD pdwOptions,PHAND
.text:100014A0                 public WlxLoggedOutSAS
.text:100014A0 WlxLoggedOutSAS proc near
.text:100014A0
.text:100014A0 pWlxContext     = dword ptr  0Ch
.text:100014A0 dwSasType       = dword ptr  10h
.text:100014A0 pAuthenticationId= dword ptr  14h
.text:100014A0 pLogonSid       = dword ptr  18h
.text:100014A0 pdwOptions      = dword ptr  1Ch
.text:100014A0 phToken         = dword ptr  20h
.text:100014A0 pNprNotifyInfo  = dword ptr  24h
.text:100014A0 pProfile        = dword ptr  28h
.text:100014A0
.text:100014A0                 push    esi
.text:100014A1                 push    edi
```



```
.text:100014A7                 call    sub_10001000
.text:100014AC                 push    64h
.text:100014AE                 mov     edi, eax
.text:100014B0                 call    ??2@YAPAXI@Z    ; operator new(uint)
.text:100014B5                 mov     eax, [esp+4+pProfile]
.text:100014B9                 mov     esi, [esp+4+pNprNotifyInfo]
.text:100014BD                 mov     ecx, [esp+4+phToken]
.text:100014C1                 mov     edx, [esp+4+pdwOptions]
.text:100014C5                 add     esp, 4
.text:100014C8                 push    eax
.text:100014C9                 mov     eax, [esp+4+pLogonSid]
.text:100014CD                 push    esi
.text:100014CE                 push    ecx
.text:100014CF                 mov     ecx, [esp+0Ch+pAuthenticationId]
.text:100014D3                 push    edx
.text:100014D4                 mov     edx, [esp+10h+dwSasType]
.text:100014D8                 push    eax
.text:100014D9                 mov     eax, [esp+14h+pWlxContext]
.text:100014DD                 push    ecx
.text:100014DE                 push    edx
```



```
.text:100014E0                 call    edi
.text:100014E2                 mov     edi, eax
.text:100014E4                 cmp     edi, 1
.text:100014E7                 jnz     short loc_1000150B
.text:100014E9                 mov     eax, [esi]
.text:100014EB                 test    eax, eax
.text:100014ED                 jz      short loc_1000150B
.text:100014EF                 mov     ecx, [esi+0Ch]
.text:100014F2                 mov     edx, [esi+8]
.text:100014F5                 push    ecx
.text:100014F6                 mov     ecx, [esi+4]
.text:100014F9                 push    edx
.text:100014FA                 push    ecx
.text:100014FB                 push    eax             ; char
.text:100014FC                 push    offset aUnSDmSPwSOldS ; "UN %s DM %s PW %s OLD %s"
.text:10001501                 push    0               ; dwMessageId
.text:10001503                 call    sub_10001570
.text:10001508                 add     esp, 18h
```

3 figures above shows the majority of the exported functions are redirections to the original functions of msginall.dll except **WlxLoggedOutSAS** which is overridden.

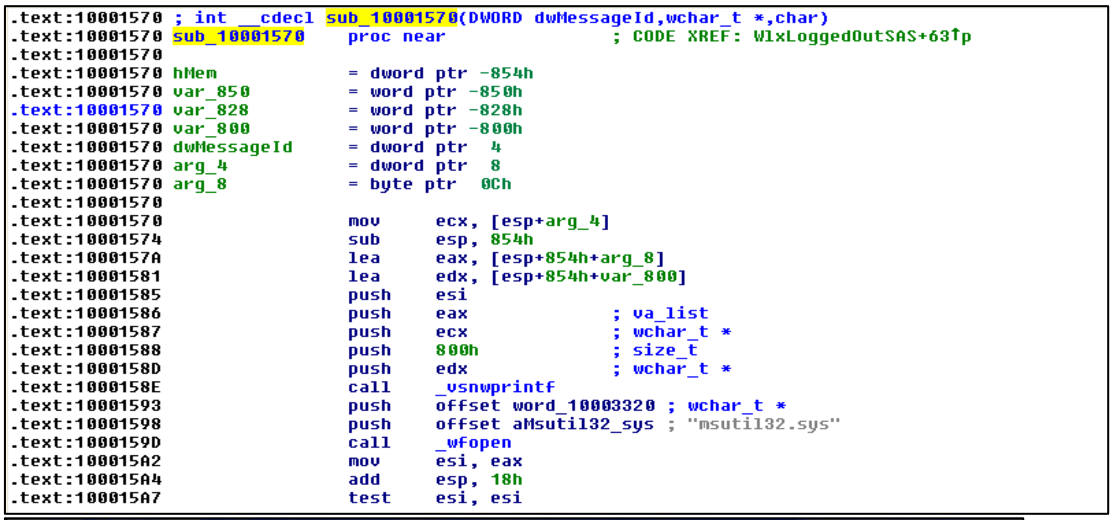
Reference: https://msdn.microsoft.com/en-us/library/windows/desktop/aa380571(v=vs.85).aspx

WlxLoggedOutSAS is invoked when the user is logged out.

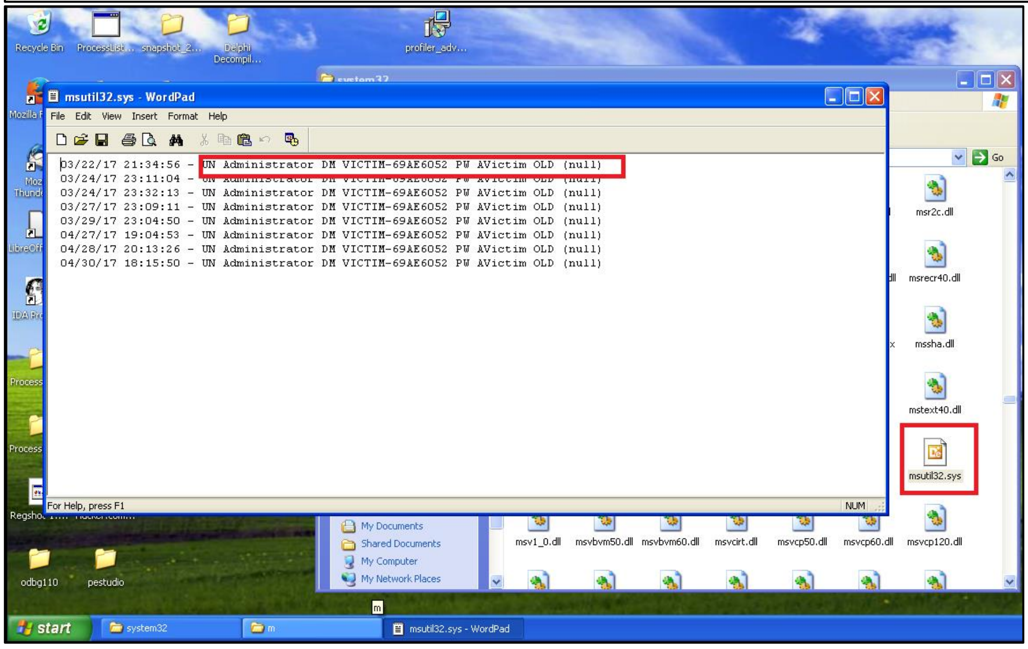
The WlxLoggedOutSAS function:

It defines an offset with the string "*UN %s DM %s PW %s OLD %s*" where %s represents variable containing specific values. We can suppose it will retrieve the Windows credentials.

Then, it calls the **sub_10001570** routine: it requires as parameters *DWORD MessageId* and a string, which is the offset previously defined



```
.text:10001570 ; int __cdecl sub_10001570(DWORD dwMessageId,wchar_t *,char)
.text:10001570 sub_10001570    proc near               ; CODE XREF: WlxLoggedOutSAS+63↑p
.text:10001570
.text:10001570 hMem            = dword ptr -854h
.text:10001570 var_850         = word ptr -850h
.text:10001570 var_828         = word ptr -828h
.text:10001570 var_800         = word ptr -800h
.text:10001570 dwMessageId     = dword ptr  4
.text:10001570 arg_4           = dword ptr  8
.text:10001570 arg_8           = byte ptr  0Ch
.text:10001570
.text:10001570                 mov     ecx, [esp+arg_4]
.text:10001574                 sub     esp, 854h
.text:1000157A                 lea     eax, [esp+854h+arg_8]
.text:10001581                 lea     edx, [esp+854h+var_800]
.text:10001585                 push    esi
.text:10001586                 push    eax             ; va_list
.text:10001587                 push    ecx             ; wchar_t *
.text:10001588                 push    800h            ; size_t
.text:1000158D                 push    edx             ; wchar_t *
.text:1000158E                 call    _vsnwprintf
.text:10001593                 push    offset word_10003320 ; wchar_t *
.text:10001598                 push    offset aMsutil32_sys ; "msutil32.sys"
.text:1000159D                 call    _wfopen
.text:100015A2                 mov     esi, eax
.text:100015A4                 add     esp, 18h
.text:100015A7                 test    esi, esi
```



```
03/22/17 21:34:56 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
03/24/17 23:11:04 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
03/24/17 23:32:13 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
03/27/17 23:09:11 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
03/29/17 23:04:50 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
04/27/17 19:04:53 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
04/28/17 20:13:26 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
04/30/17 18:15:50 - UN Administrator DM VICTIM-69AE6052 PW AVictim OLD (null)
```

Notice the file "**msutil32.sys**" at this location: c:\windows\system32. Further, force the malware to log credentials by running *Lab11-01.exe*, rebooting the machine, and then logging in and out of the system. The figure above is an example of the data contained in a log file created by this malware: the credential detail username and password.

Does the malware use the network?

This malware does not communicate with the network

Potential manual removal techniques

To remove this malware, we must delete the Registry Key and download the true Gina.dll

Summary

This malware uses the persistence mechanism via the replacement of Gina.dll in the Registry. The malware drops a DLL on the system and installs it to steal user credentials, beginning after system reboot. Then, it overrides the function named *WlxLoggedOutSAS* to collect the Windows credential at the user session logout. Once the GINA interceptor DLL is installed and running, it logs credentials to *msutil32.sys* when a user logs out of the system.