

# Practical Malware Analysis & Triage

## Malware Analysis Report

### Process Injection Malware

Nov 2022 | NoobUltraProMax | v1.0



## Table of Contents

<b>Executive Summary.....</b>	<b>3</b>
<b>High-Level Technical Summary.....</b>	<b>4</b>
<b>Malware Composition.....</b>	<b>5</b>
malware.exe.....	5
werflt.exe:.....	5
<b>Basic Static Analysis.....</b>	<b>6</b>
<b>Basic Dynamic Analysis.....</b>	<b>8</b>
<b>Aadvanced Static Analysis.....</b>	<b>10</b>
<b>Advanced Dynamic Analysis.....</b>	<b>12</b>
<b>Indicators of Compromise.....</b>	<b>13</b>
Network Indicators.....	13
Host-based Indicators.....	14
<b>Rules &amp; Signatures.....</b>	<b>15</b>
<b>Appendices.....</b>	<b>16</b>
A. Yara Rules.....	16
B. Callback URLs.....	16
C. Hex ShellCode Snippets.....	17



## Executive Summary

SHA256 hash	fca62097b364b2f0338c5e4c5bac86134cedffa4f8ddf27ee9901734128952e3
MD5 hash	6d8895c63a77ebe5e49b656bdefdb822

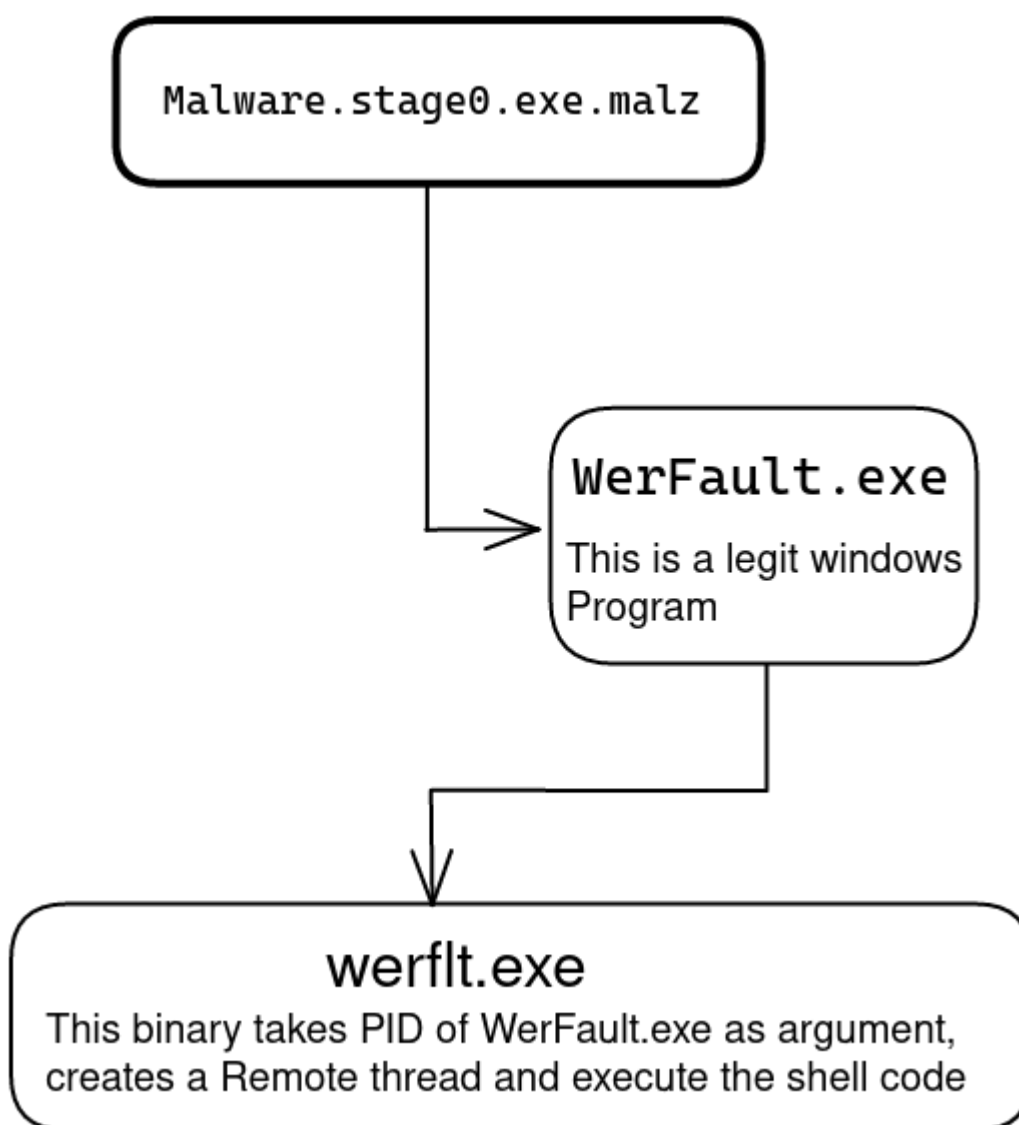
MalwareStage0.exe is a ProcessInjector-dropper malware sample first identified on May 14<sup>th</sup>, 2021. It is a Nim-compiled dropper that runs on the x86 Windows operating system. It consists of one unpacked payloads that is executed following a successful spearphishing attempt. Symptoms of infection includes a new process in system werflt.exe which could be confused with a real microsoft software named WerFault.exe.

YARA signature rules are attached in Appendix A. Malware sample and hashes have been submitted to VirusTotal for further examination.



## High-Level Technical Summary

Malware.stage0.exe consists of two parts: an unpacking stage where werfault.exe is executed. Werfault.exe is the Windows Error Reporting process of Windows 10. Second stage creates a file in C drive with slightly different name werflt.exe, this file is a Remote Process Injector. This malware is executed with parameter as PID of the original error checking program, now the shellcode is executed with the privileges of the Werfault.exe. After execution of the second stage it opens a port on localhost (hxxps://localhost:8443)



*Illustration 1: Flow Diagram of Malware*



## Malware Composition

Malware consists of the following components:

File Name	SHA256 Hash
<b>Malware.Stage0.exe</b>	fca62097b364b2f0338c5e4c5bac86134cedffa4f8ddf27ee9901734128952e3
<b>werflt.exe</b>	0516009622b951c6c08fd8d81a856eaab70c02e6bc58d066bbdfafe8c6edabea

### [malware.exe](#)

The initial executable that runs after a successful spearphish. It acts as a trojan horse carrying a process injector malware, which when executed unpacks the payload, werflt.exe.

### [werflt.exe:](#)

This binary is unpacked when Malware.exe is detonated. This binary is a Remote Thread Process Injector Malware.



# Basic Static Analysis

- **Virustotal Information :**

- Uploading hash/Binary
- Its a known malware, 51 security vendors and 3 sandboxes flagged this file as malicious
- <https://www.virustotal.com/gui/file/fca62097b364b2f0338c5e4c5bac86134cedffa4f8ddf27ee9901734128952e3>

- **Strings/Floss output Analysis :**

- "\$ floss -n 9 Malware.exe.malz"
- "\$ cat floss.out | grep nim | wc -l"
- which gives output of "37"
- We can conclude that the binary is written in "nim" Programming Language.
- Some other interesting strings -

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

- **PEView/PE-bear :**

- Compression Analysis
  - Section Headers (.text)
    - Marginal difference between raw size and virtual size
    - It's an un-packed binary
- Compiled Time/Date
  - Image\_file\_headers : "Thu 7.10.2021 17:43:04 UTC"
- Suspicious Imports
  - kernel32.dll
  - mscvrt.dll
  - user32.dll



- **PEStudio**

- Indicators
  - Suspicious Strings : 18
  - Suspicious Imports : 4
- Flagged Imports/API calls
  - Enumeration
    - GetCurrentProcessId
    - GetCurrentThreadId
  - Injection
    - VirtualProtect
  - Helper
    - TerminateProcess
- Flagged Strings
  - Apart from Import Strings
  - "CreateProcess"
  - "OpenProcess"
  - "Suspend Thread"
  - "WriteProcessMemory"

pestudio 9.44 - Malware Initial Assessment - www.winitor.com [c:\users\bunny\desktop\malware.stage0.exe]

file settings about

c:\users\bunny\desktop\malware.stage0.exe

indicators (43)

indicator (42)	detail	level
strings > flag	18	1
imports > name > flag	4	1
section > virtualized	.bss	2
overlay > file-ratio	15.10%	2
overlay > entropy	4.665	2
overlay > size	59187 bytes	2
file > embedded	signature: executable, location: .rdata, offset: 0x0000BE28, size: 9060	2
file > embedded	signature: unknown, location: overlay, offset: 0x00051400, size: 59187	2
overlay > signature > name	unknown	2
entry-point > location	0x000014A0	3
file > image-base	0x00400000	3
strings > count	10824	3
tls-callback > count	2	3
libraries > count	3	3
file > size	391987 bytes	3
section > alignment	4096 bytes	3
file > alignment	512 bytes	3
dos-stub > size	64 bytes	3
imports > count	71	3
file > subsystem	GUI	3
file > score > error	The server name or address could not be resolved	3
file > os > target	Windows NT 4.0	3
security > protection	address-space-layout-randomization (ASLR) > OFF	3
security > protection	code-integrity (CI) > OFF	3

sha256: FCA62097B364B2F0338C5E4C5BAC86134CEDFFA4F8DF27EE9901734128952E3    cpu: 32-bit    file-type: executable    subsystem: GUI    entry-point: 0x000014A0

*Illustration 2: PESTudio - Indicators*



## Basic Dynamic Analysis

- **Host Based Indicators :**
  - When Malware is detonated a new file is created C:\Users\Public\werflt.exe

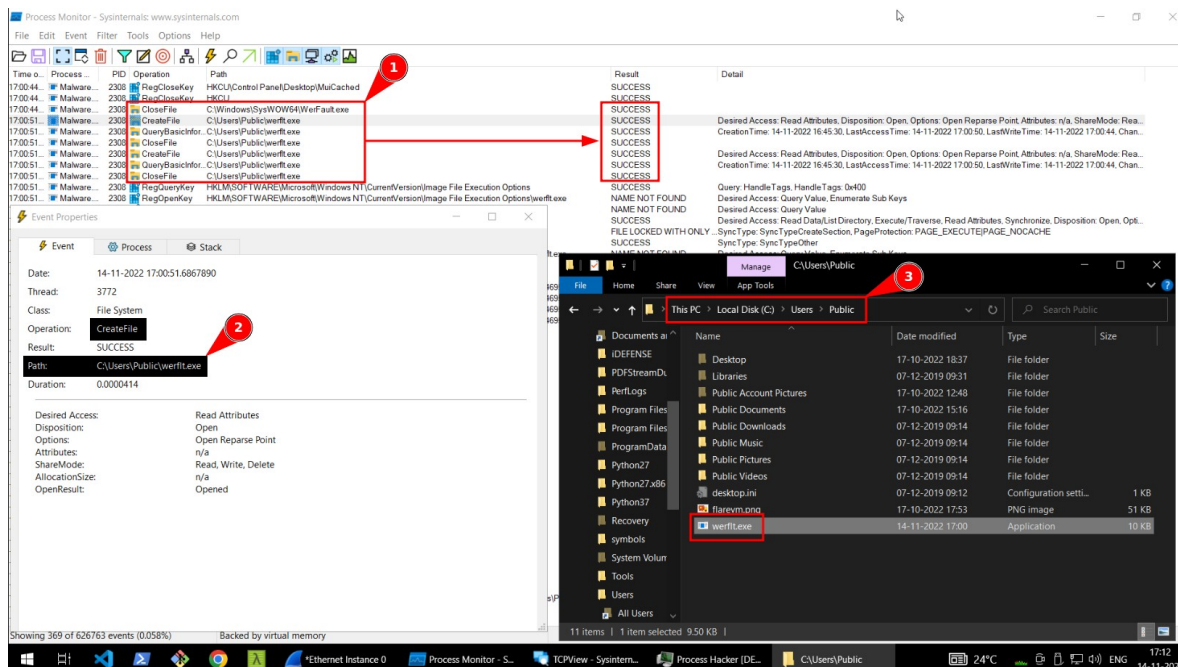


Illustration 3: ProcMon catches a new file creation- C:\Users\Public\werflt.exe

- Process Tree in Procmon shows werflt.exe
  - werflt.exe is executed with parameter PID of WerFault.exe

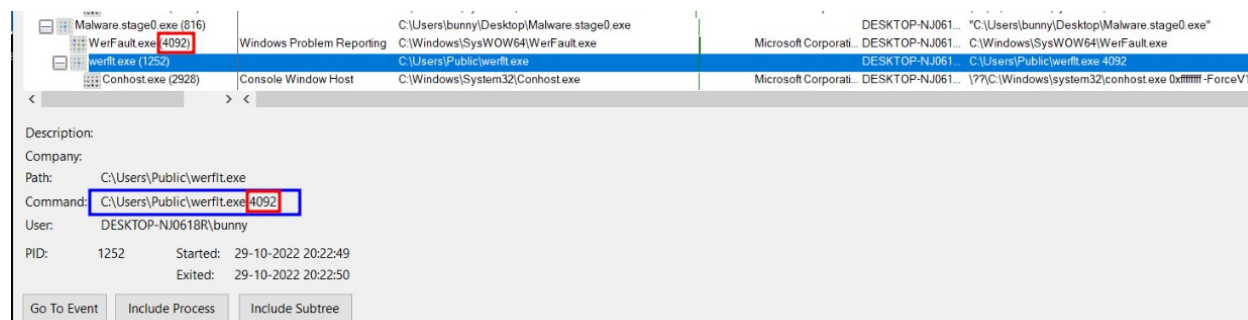


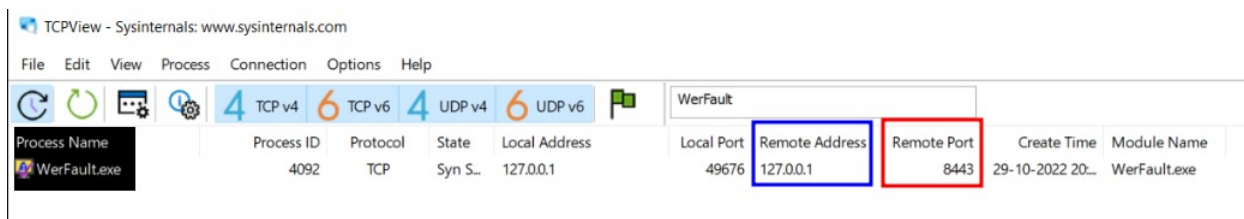
Illustration 4: ProcessTree - Malware runs werfault.exe, which runs werflt.exe





- **Network Based Indicators**

- TcpView catches a process “WerFault.exe” which opens port 8443 on localhost.



*Illustration 5: TCPView - Process WerFault opens port on localhost*

- We can try and connect to this port
  - Starting a listener on machine `ncat.exe -lvnp 8443`

```
C:\Users\bunny
λ ncat.exe -lvnp 8443
Ncat: Version 5.59BETA1 ( http://nmap.org/ncat )
Ncat: Listening on 0.0.0.0:8443
Ncat: Connection from 127.0.0.1:49675.
Microsoft Windows [Version 10.0.19043.2130]
(c) Microsoft Corporation. All rights reserved.

FLARE 16-11-2022 16:02:25.04
C:\Users\bunny\Desktop>whoami
whoami
desktop-nj0618r\bunny
```

*Illustration 6: netcat set up to listen on port 8443*



## Advanced Static Analysis

- Cutter – Disassembled code if werflt.exe

```
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    0xffffffff                      ; DWORD dwDesiredAccess
call    dword [OpenProcess]             ; 0x402004 ; HANDLE OpenProcess(DWORD dwDesiredAccess, BOOL bI...
push    0x40                            ; 'Q' ; 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
```

*Illustration 7: Decompiled Binary in ASM*



- Analysis of De-compiled x86 Binary :
  - Main Function takes in “arg\_ch” parameter
  - variable is defined in main “IpBuffer”
  - “arg\_ch is moved to “eax”
    - This eax will be used in future API call
  - 1<sup>st</sup> API call : OpenProcess
    - It takes 3 parameters
      - Desired Access Level (0x1fffff)
      - Boolean InheritHandle (0)
      - ProcessId (eax)
  - “eax” is moved to “edi”
  - 2<sup>nd</sup> API call : VirtualAlloc
    - Allocates memory in the opened process
    - It takes 5 parameters
      - hProcess (edi)
      - IpAddress (0)
      - Size (0x145)
      - AllocationType (0x3000)
      - Protect (0x40)
  - 3<sup>rd</sup> API call : WriteProcessMemory
    - Writes shellcode in the allocated memory
    - Takes 5 parameters
      - hProcess (edi)
      - IpBaseAddress (esi)
      - IpBuffer (eax) (IPBuffer)
      - Size\_T nSize (0x145)
      - IpNumberOfBytesWritten (0)
  - 4<sup>th</sup> API call : CreateRemoteThread
    - 7 parameters
      - First is hProcess (edi)
      - Fourth is IpStartAddress (esi)
      - Every other parameter is 0
  - 5<sup>th</sup> API call : Closehandle
    - 1 parameter
      - Handle Hobject (edi)
  - Fourth call creates Remote thread on local machine.



## Advanced Dynamic Analysis

```
werflt.exe - PID: 1956 - Module: werflt.exe - Thread: Main Thread 3496 - x32dbg [Elevated]
File View Debug Tracing Plugins Favourites Options Help Apr 17 2021 (TitanEngine)

CPU Log Notes Breakpoints Memory Map Call Stack SEH Script Symbols Source References

EIP 00701267 56 push esi
00701268 FF30 push dword ptr ds:[eax]
0070126A E8 91FDFFFF call werflt.701000
0070126F 83C4 0C add esp,c
00701272 8BF0 mov esi,eax
00701274 E8 34060000 call werflt.7018AD
00701279 84C0 test al,al
0070127B 74 68 je werflt.7012E8
0070127D 84D8 test bl,bl
0070127F 75 05 jne werflt.701286
00701281 E8 F2090000 call <JMP.&_cexit>
00701286 6A 00 push 0
00701288 6A 01 push 1
0070128A E8 7B030000 call werflt.70160A
0070128F 59 pop ecx
00701290 59 pop ecx
00701291 C745 FC FEFFFFFF mov dword ptr ss:[ebp-4],FFFFFFFE
00701298 8BC6 mov eax,esi
0070129A EB 35 jmp werflt.7012D1
0070129C 8B4D EC mov ecx,dword ptr ss:[ebp-14]
0070129F 8B01 mov eax,dword ptr ds:[ecx]
007012A1 8B00 mov eax,dword ptr ds:[eax]
007012A3 8945 E0 mov dword ptr ss:[ebp-20],eax
007012A6 51 push ecx
007012A7 50 push eax
007012A8 E8 7D090000 call <JMP.&_seh_filter_exe>
007012AD 59 pop ecx
007012AE 59 pop ecx
007012AF C3 ret
007012B0 8B65 E8 mov esp,dword ptr ss:[ebp-18]
007012B3 E8 F5050000 call werflt.7018AD
007012B8 84C0 test al,al
007012BA 74 32 je werflt.7012EE
007012BC 807D E7 00 cmp byte ptr ss:[ebp-19],0
007012C0 75 05 jne werflt.7012C7
007012C2 E8 B7090000 call <JMP.&_c_exit>
007012C7 C745 FC FEFFFFFF mov dword ptr ss:[ebp-4],FFFFFFFE
007012CE 8B45 E0 mov eax,dword ptr ss:[ebp-20]
007012D1 8B4D F0 mov ecx,dword ptr ss:[ebp-10]
007012D4 64 8900 00000000 mov dword ptr esi,[0] ecx

esi:&"C:\\Users\\bunny\\Desktop\\werflt.exe"
esi:&"C:\\Users\\bunny\\Desktop\\werflt.exe"
esi:&"C:\\Users\\bunny\\Desktop\\werflt.exe"
esi:&"C:\\Users\\bunny\\Desktop\\werflt.exe"
[ebp-10]:&"Púu"
```

Illustration 8: werflt.exe debugging

```
0055F978 0055F998 return to werflt.00701173 from werflt.00701C90
0055F97C 00701173 return to ucrtbase.755CAC47 from ???
0055F980 00000000 return to ucrtbase.755CAC47 from ???
0055F984 755CAC47 return to ucrtbase.755CAC47 from ???
0055F988 0028E000 werflt.EntryPoint
0055F98C 007012F7 werflt.EntryPoint
0055F990 007012F7 werflt.EntryPoint
0055F994 00000002 return to werflt.007015F5 from werflt.00701C06
0055F998 007015F5 return to werflt.007015F5 from werflt.00701C06
0055F99C 00000001 return to werflt.007015F5 from werflt.00701C06
0055F9A0 00599E00 &"C:\\Users\\bunny\\Desktop\\werflt.exe"
0055F9A4 0059E660 &"ALLUSERSPROFILE=C:\\ProgramData"
0055F9A8 878A63D1 werflt.EntryPoint
0055F9AC 007012F7 werflt.EntryPoint
0055F9B0 007012F7 werflt.EntryPoint
0055F9B4 0028E000 return to ucrtbase.755CAC47 from ???
0055F9B8 00000000 return to ucrtbase.755CAC47 from ???
0055F9BC 00000000 return to ucrtbase.755CAC47 from ???
0055F9C0 00000000 return to ucrtbase.755CAC47 from ???
0055F9C4 0055F9A8 return to ucrtbase.755CAC47 from ???
0055F9C8 00000000 return to ucrtbase.755CAC47 from ???
0055F9CC 0055FA38 Pointer to SEH_Record[1]
0055F9D0 00701A06 werflt.00701A06
0055F9D4 87AFBCBD return to ucrtbase.755CAC47 from ???
0055F9D8 00000000 return to ucrtbase.755CAC47 from ???
0055F9DC 0055F9EC &"xúU"
0055F9E0 r74EEFA29 return to kernel32.74EEFA29 from ???
```

Illustration 9: werflt.exe Stack



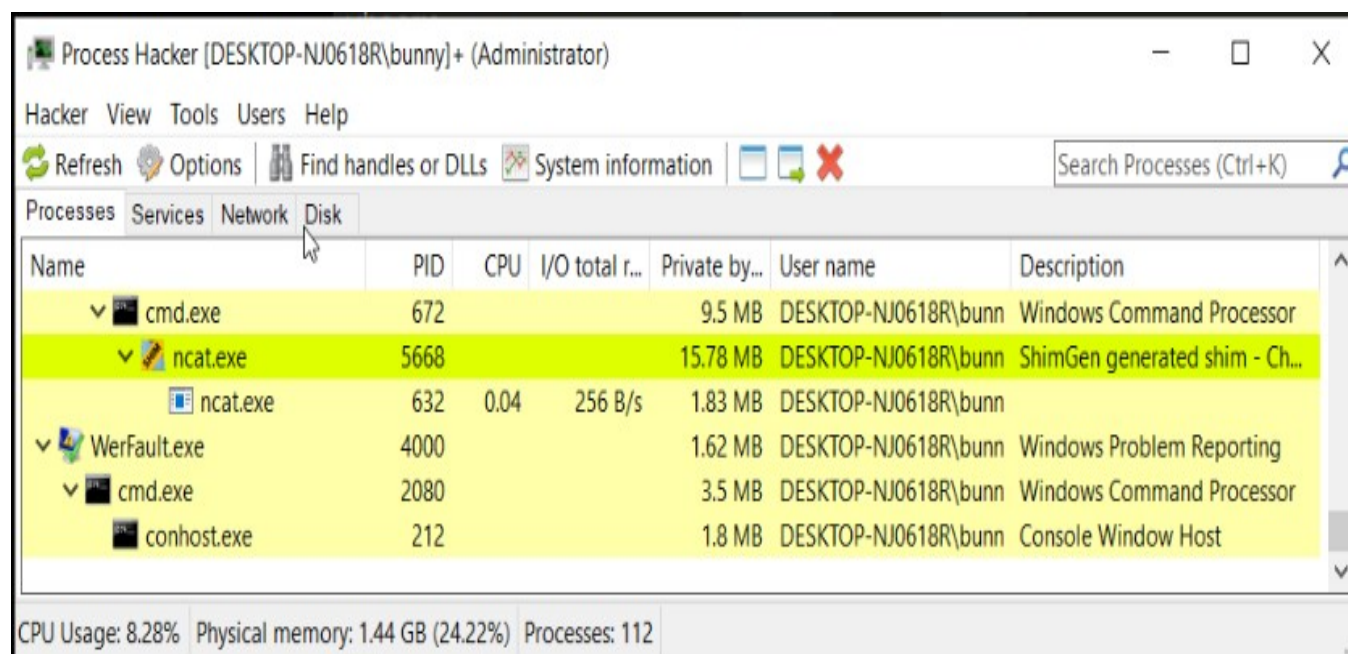


## Indicators of Compromise

The full list of IOCs can be found in the Appendices.

### Network Indicators

Werflt.exe injects shell code in WerFault.exe and executes it with its PID.  
The Process persist when a shell is connected to a remote machine.

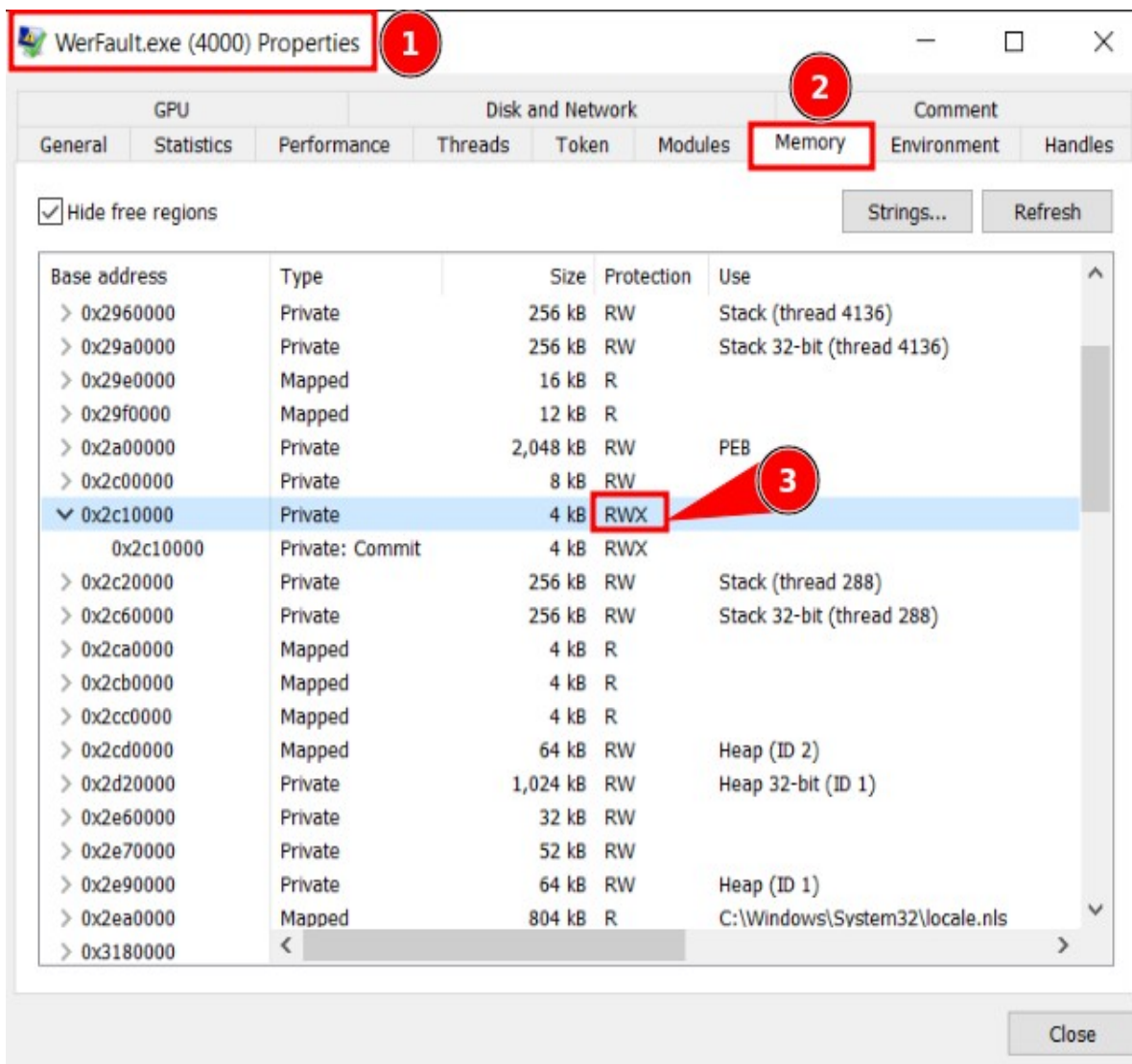


*Illustration 10: werflt.exe runs shellcode as WerFault.exe*



## Host-based Indicators

We can open this process to see memory protections, There's one with Full Read-Write-Execute (RWX) permission, which is very suspicious for a process like WerFault which is a error logging process.



*Illustration 11: Werfault.exe has R-W-X*



## Rules & Signatures

A full set of YARA rules is included in Appendix A.

- String1 is set as Nim
- String2 is set as werflt.exe
- Suspicious bit is added as string
- Magic Number is MZ so 32-bit binary
- Callback URL is localhost on port 8443



# Appendices

## A. Yara Rules

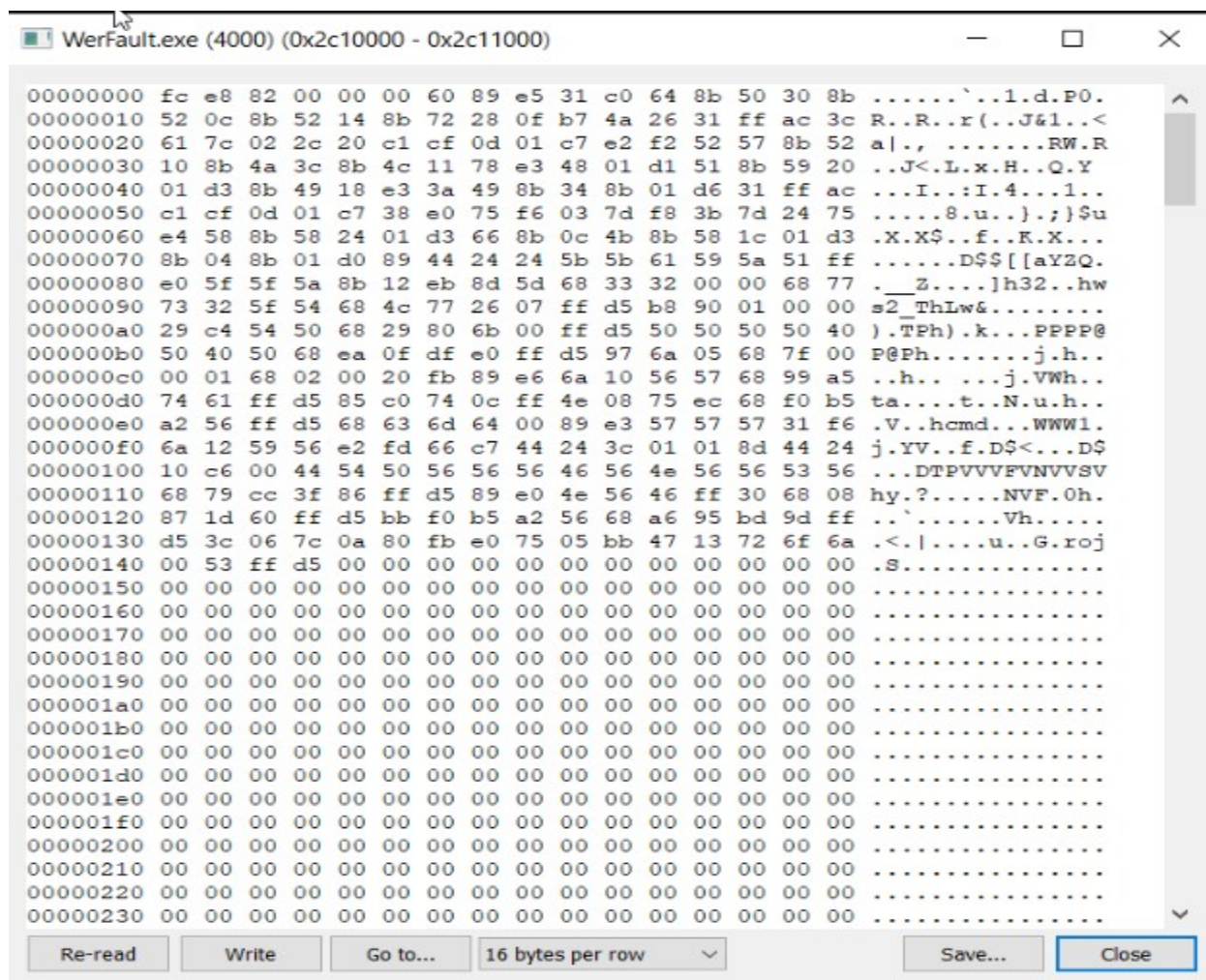
```
rule stage0_malware {  
  
    meta:  
        last_updated = "2022-11-16"  
        author = "NoobUltraProMax"  
        description = "Yara signature of Malware.stage0.exe"  
  
    strings:  
        $string1 = "nim"  
        $string2 = "C:\Users\Public\werflt.exe" ascii  
        $hex_string = { 43 52 54 49 6E 6A 65 63 74 6F 72 43 6F 6E 73 6F }  
        $PE_magic_byte = "MZ"  
  
    condition:  
        $PE_magic_byte at 0 and  
        ($string1 and $string2) or  
  
        $sus_hex_string  
}
```

## B. Callback URLs

Domain	Port
Hxxp://localhost	8443

## C. Hex ShellCode Snippets





*Illustration 12: Process Injection Routine in Cutter*