Alex Smetana

Lab03

1) **<u>What malware is this? Provide all the elements of the CARO naming scheme - Type, platform, family, major variant, minor variant (if present),modifiers</u>**

    a. Type – Trojan

    b. Platform – Win32

    c. Family – Vigorf

    d. Variant – A

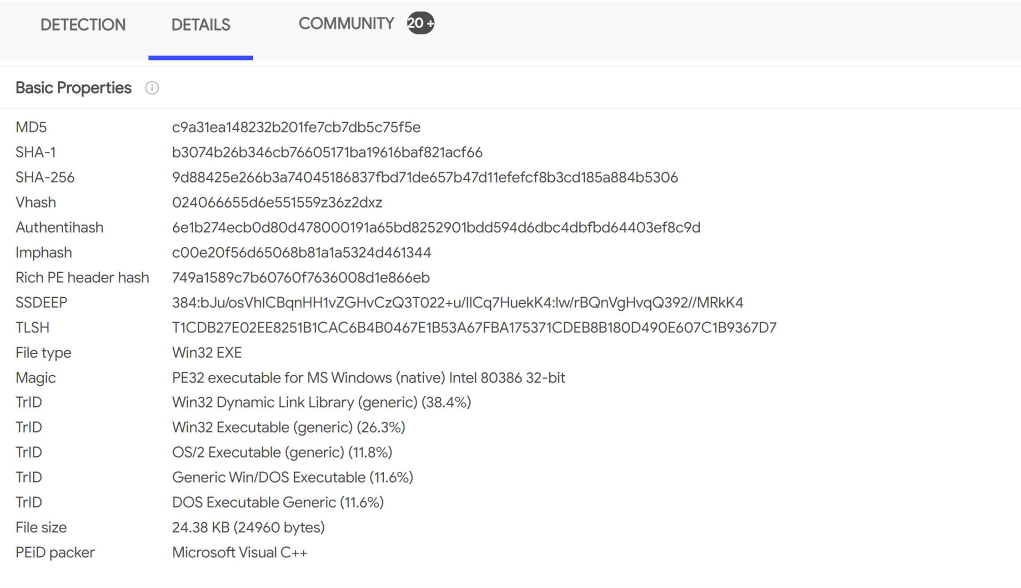
2) **<u>Provide a brief explanation of what the malware does, according to threat analysts. (Feel free to consult blogs, signature encyclopedias, or whatever.)</u>**

    a. This virus is known as a Trojan. It's name is derived from the "Trojan Horse", and works in a similar fashion. The Trojan designed to fool individuals to downloading it by being disguised as something else. Once an individual clicks on the link or download, the Trojan downloads files onto the victim's computer, which can be very detrimental.

    b. "This Trojan arrives on a system as a file dropped by other malware or as a file downloaded unknowingly by users when visiting malicious sites. This Trojan arrives on a system as a file dropped by other malware or as a file downloaded unknowingly by users when visiting malicious sites." – Trend Micro

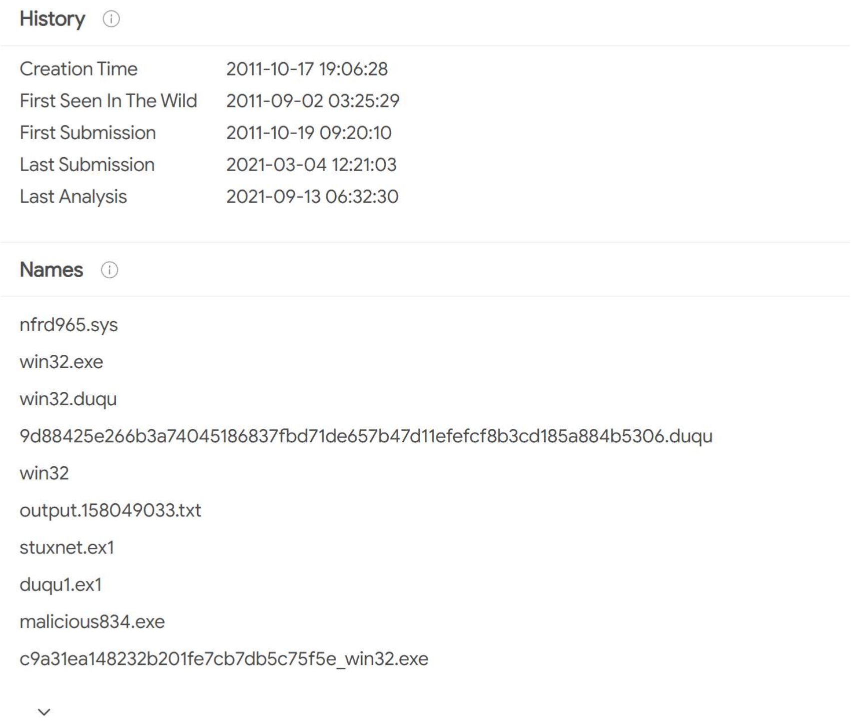
3) **<u>Perform a static analysis of the malware. Be very careful to not execute the binary. You may need to create an exception to the working directory for your AV to keep it from being cleaned as you work. If so, don't forget to strike the exception when you are done. Write a report that contains your analysis. be sure to include some analysis of disassembly, but you can essentially choose any random function to interpret. You just need to show you can do it.</u>**

    a. Firstly, I put the file into Virus Total. It gave me some basic information about the Virus. It confirmed that it was indeed a type A Trojan malware. It is demed as malicious. These are the specs that were found:



Basic Properties

MD5	c9a31ea148232b201fe7cb7db5c75f5e
SHA-1	b3074b26b346cb76605171ba19616baf821acf66
SHA-256	9d88425e266b3a74045186837fbd71de657b47d11efefcf8b3cd185a884b5306
Vhash	024066655d6e551559z36z2dxz
Authentihash	6e1b274ecb0d80d478000191a65bd8252901bdd594d6dbc4dbfbd64403ef8c9d
Imphash	c00e20f56d65068b81a1a5324d461344
Rich PE header hash	749a1589c7b60760f7636008d1e866eb
SSDEEP	384:bJu/osVhICBqnHH1vZGHvCzQ3T022+u/IICq7HuekK4:lw/rBQnVgHvqQ392//MRkK4
TLSH	T1CDB27E02EE8251B1CAC6B4B0467E1B53A67FBA175371CDEB8B180D490E607C1B9367D7
File type	Win32 EXE
Magic	PE32 executable for MS Windows (native) Intel 80386 32-bit
TrID	Win32 Dynamic Link Library (generic) (38.4%)
TrID	Win32 Executable (generic) (26.3%)
TrID	OS/2 Executable (generic) (11.8%)
TrID	Generic Win/DOS Executable (11.6%)
TrID	DOS Executable Generic (11.6%)
File size	24.38 KB (24960 bytes)
PEiD packer	Microsoft Visual C++



History

Creation Time	2011-10-17 19:06:28
First Seen In The Wild	2011-09-02 03:25:29
First Submission	2011-10-19 09:20:10
Last Submission	2021-03-04 12:21:03
Last Analysis	2021-09-13 06:32:30

Names

nfrd965.sys

win32.exe

win32.duqu

9d88425e266b3a74045186837fbd71de657b47d11efefcf8b3cd185a884b5306.duqu

win32

output.158049033.txt

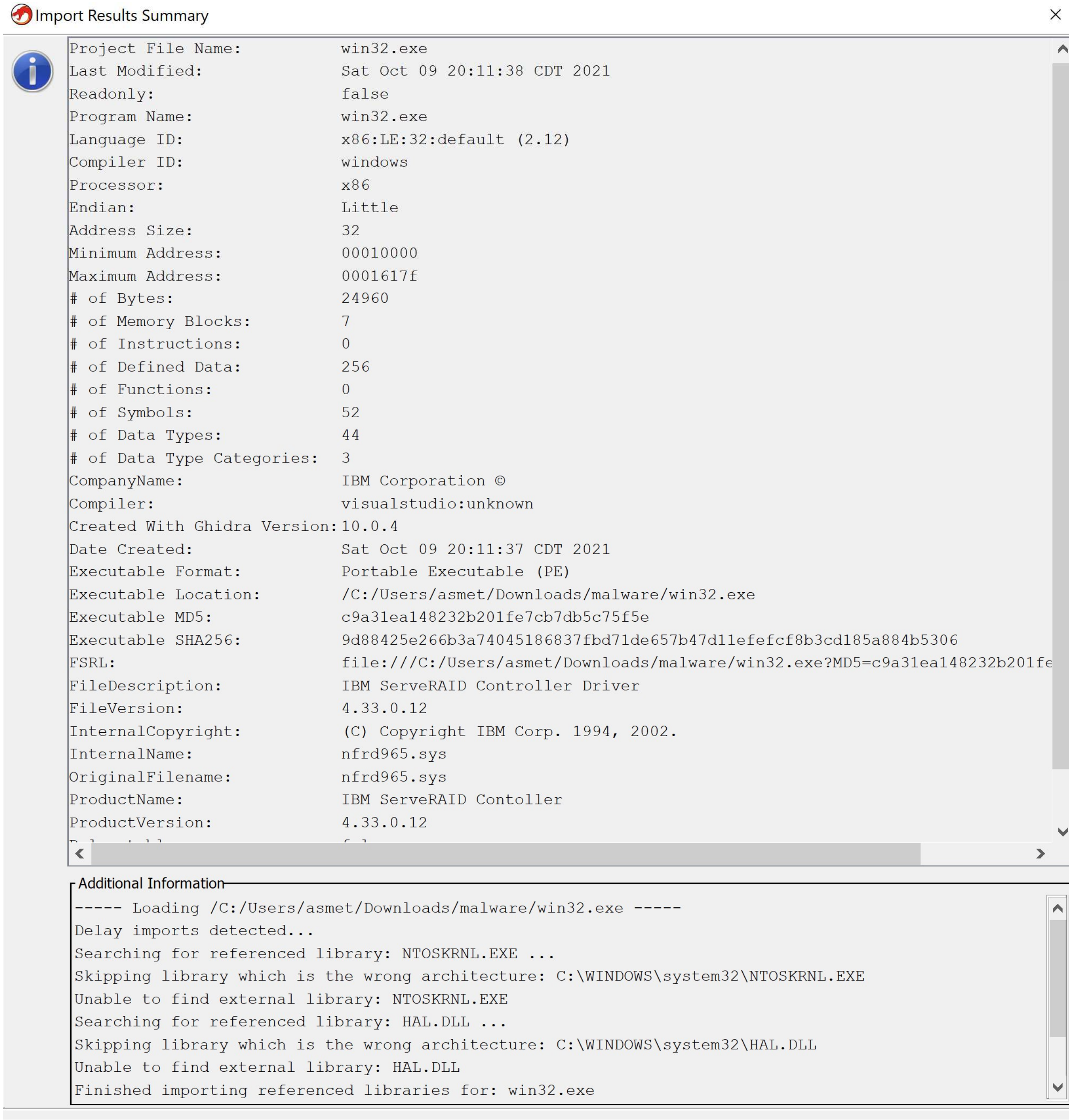
stuxnet.ex1

duqu1.ex1

malicious834.exe

c9a31ea148232b201fe7cb7db5c75f5e_win32.exe

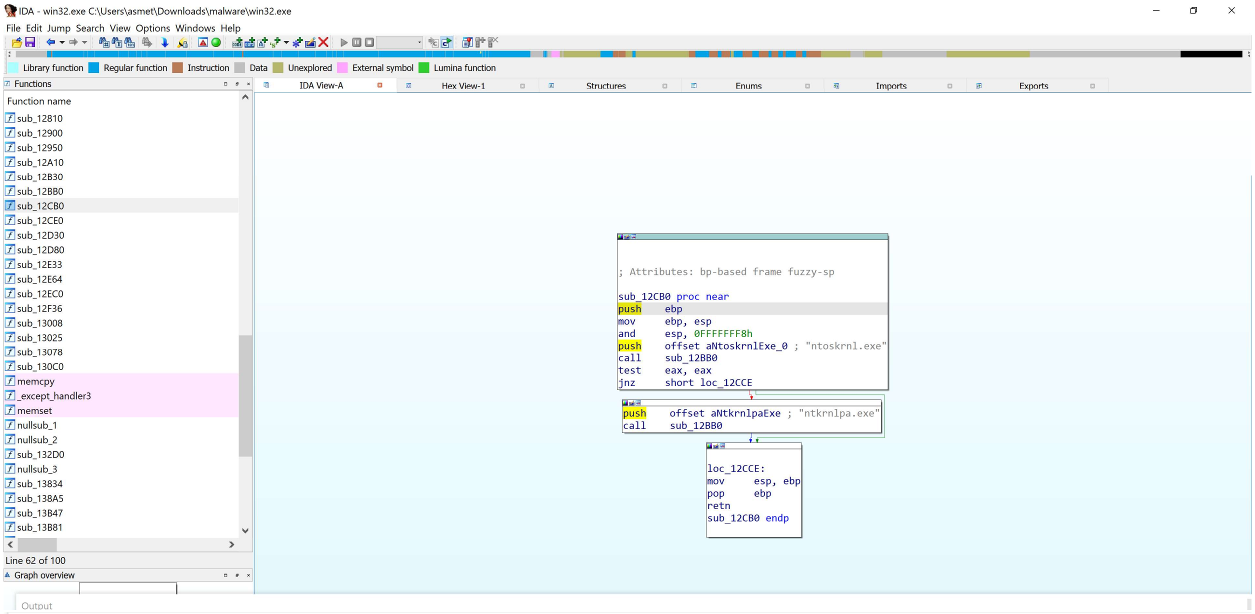
b. Secondly, I used Ida and Ghidra to disassemble the virus. These were the specs found:



```
Project File Name:           win32.exe
Last Modified:               Sat Oct 09 20:11:38 CDT 2021
Readonly:                    false
Program Name:                win32.exe
Language ID:                 x86:LE:32:default (2.12)
Compiler ID:                 windows
Processor:                   x86
Endian:                      Little
Address Size:                32
Minimum Address:             00010000
Maximum Address:             0001617f
# of Bytes:                  24960
# of Memory Blocks:          7
# of Instructions:           0
# of Defined Data:           256
# of Functions:              0
# of Symbols:                52
# of Data Types:             44
# of Data Type Categories:   3
CompanyName:                 IBM Corporation ©
Compiler:                    visualstudio:unknown
Created With Ghidra Version:10.0.4
Date Created:                Sat Oct 09 20:11:37 CDT 2021
Executable Format:           Portable Executable (PE)
Executable Location:         /C:/Users/asmet/Downloads/malware/win32.exe
Executable MD5:              c9a31ea148232b201fe7cb7db5c75f5e
Executable SHA256:           9d88425e266b3a74045186837fbd71de657b47d11efefcf8b3cd185a884b5306
FSRL:                        file:///C:/Users/asmet/Downloads/malware/win32.exe?MD5=c9a31ea148232b201fe
FileDescription:             IBM ServeRAID Controller Driver
FileVersion:                 4.33.0.12
InternalCopyright:           (C) Copyright IBM Corp. 1994, 2002.
InternalName:                nfrd965.sys
OriginalFilename:            nfrd965.sys
ProductName:                 IBM ServeRAID Contoller
ProductVersion:              4.33.0.12
```

```
----- Loading /C:/Users/asmet/Downloads/malware/win32.exe -----
Delay imports detected...
Searching for referenced library: NTOSKRNL.EXE ...
Skipping library which is the wrong architecture: C:\WINDOWS\system32\NTOSKRNL.EXE
Unable to find external library: NTOSKRNL.EXE
Searching for referenced library: HAL.DLL ...
Skipping library which is the wrong architecture: C:\WINDOWS\system32\HAL.DLL
Unable to find external library: HAL.DLL
Finished importing referenced libraries for: win32.exe
```

c. I decided to try to analyze a function in the code. I picked a very simple function 12CB0. This function defines an integer variable. The function sets the variable equal to the integer value it is comparing. If this integer is equal to zero, the function will return itself.



Function name

sub_12810
sub_12900
sub_12950
sub_12A10
sub_12B30
sub_12BB0
sub_12CB0
sub_12CE0
sub_12D30
sub_12D80
sub_12E33
sub_12E64
sub_12EC0
sub_12F36
sub_13008
sub_13025
sub_13078
sub_130C0
memcpy
_except_handler3
memset
nullsub_1
nullsub_2
sub_132D0
nullsub_3
sub_13834
sub_138A5
sub_13B47
sub_13B81

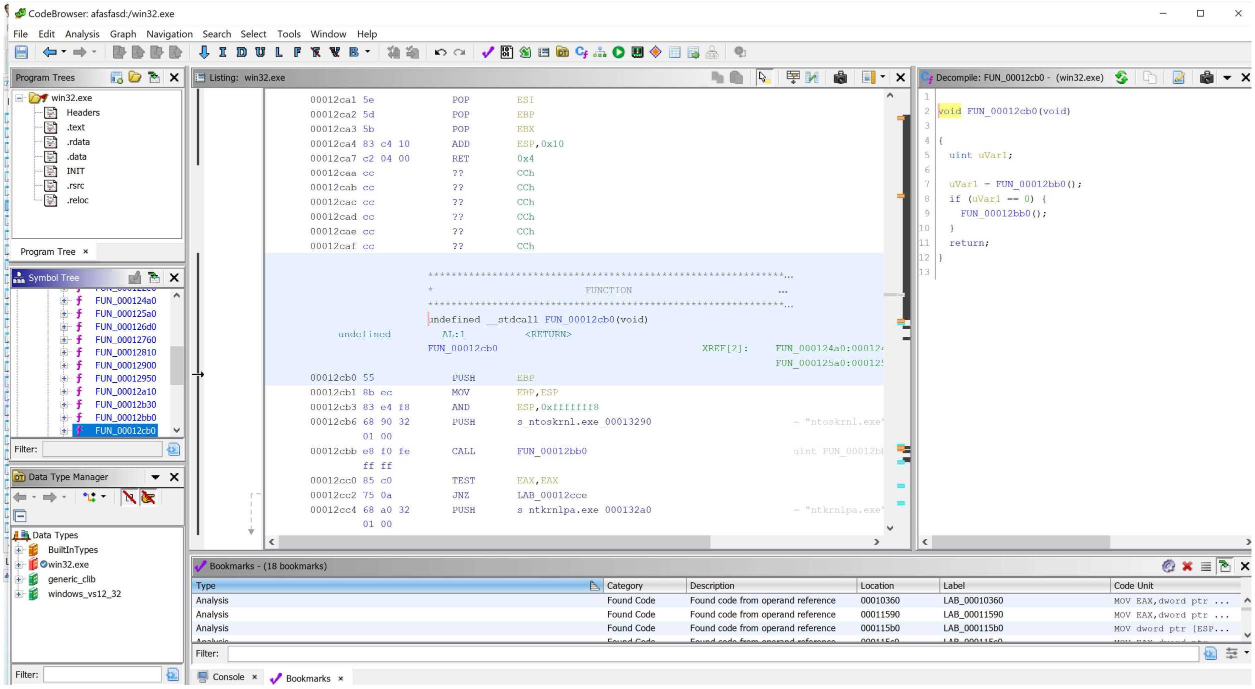
Line 62 of 100

```
; Attributes: bp-based frame fuzzy-sp

sub_12CB0 proc near
push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF8h
push    offset aNtoskrnlExe_0 ; "ntoskrnl.exe"
call    sub_12BB0
test    eax, eax
jnz     short loc_12CCE

push    offset aNtkrnlpaExe ; "ntkrnlpa.exe"
call    sub_12BB0

loc_12CCE:
mov     esp, ebp
pop     ebp
retn
sub_12CB0 endp
```


CodeBrowser: afasfasd:/win32.exe
File Edit Analysis Graph Navigation Search Select Tools Window Help
Program Trees
Program Tree
Symbol Tree
Filter:
Data Type Manager
Listing: win32.exe
Decompile: FUN_00012cb0 - (win32.exe)
Console
Bookmarks

win32.exe
Headers
.text
.rdata
.data
INIT
.rsrc
.reloc

FUN_000124a0
FUN_000125a0
FUN_000126d0
FUN_00012760
FUN_00012810
FUN_00012900
FUN_00012950
FUN_00012a10
FUN_00012b30
FUN_00012bb0
FUN_00012cb0

Data Types
BuiltInTypes
win32.exe
generic_clib
windows_vs12_32

```
00012ca1 5e          POP      ESI
00012ca2 5d          POP      EBP
00012ca3 5b          POP      EBX
00012ca4 83 c4 10    ADD      ESP,0x10
00012ca7 c2 04 00    RET      0x4
00012caa cc          ??       CCh
00012cab cc          ??       CCh
00012cac cc          ??       CCh
00012cad cc          ??       CCh
00012cae cc          ??       CCh
00012caf cc          ??       CCh

                 *************************************************************...
                 *                          FUNCTION                        ...
                 *************************************************************...
                 undefined __stdcall FUN_00012cb0(void)
     undefined       AL:1         <RETURN>
                 FUN_00012cb0                              XREF[2]:    FUN_000124a0:00012
                                                                       FUN_000125a0:00012
00012cb0 55          PUSH     EBP
00012cb1 8b ec       MOV      EBP,ESP
00012cb3 83 e4 f8    AND      ESP,0xfffffff8
00012cb6 68 90 32    PUSH     s_ntoskrnl.exe_00013290                  = "ntoskrnl.exe"
         01 00
00012cbb e8 f0 fe    CALL     FUN_00012bb0                             uint FUN_00012b
         ff ff
00012cc0 85 c0       TEST     EAX,EAX
00012cc2 75 0a       JNZ      LAB_00012cce
00012cc4 68 a0 32    PUSH     s_ntkrnlpa.exe_000132a0                  = "ntkrnlpa.exe"
         01 00
```

```
void FUN_00012cb0(void)

{
  uint uVar1;

  uVar1 = FUN_00012bb0();
  if (uVar1 == 0) {
    FUN_00012bb0();
  }
  return;
}
```

Bookmarks - (18 bookmarks)

Type	Category	Description	Location	Label	Code Unit
Analysis	Found Code	Found code from operand reference	00010360	LAB_00010360	MOV EAX,dword ptr ...
Analysis	Found Code	Found code from operand reference	00011590	LAB_00011590	MOV EAX,dword ptr ...
Analysis	Found Code	Found code from operand reference	000115b0	LAB_000115b0	MOV dword ptr [ESP...

Filter: