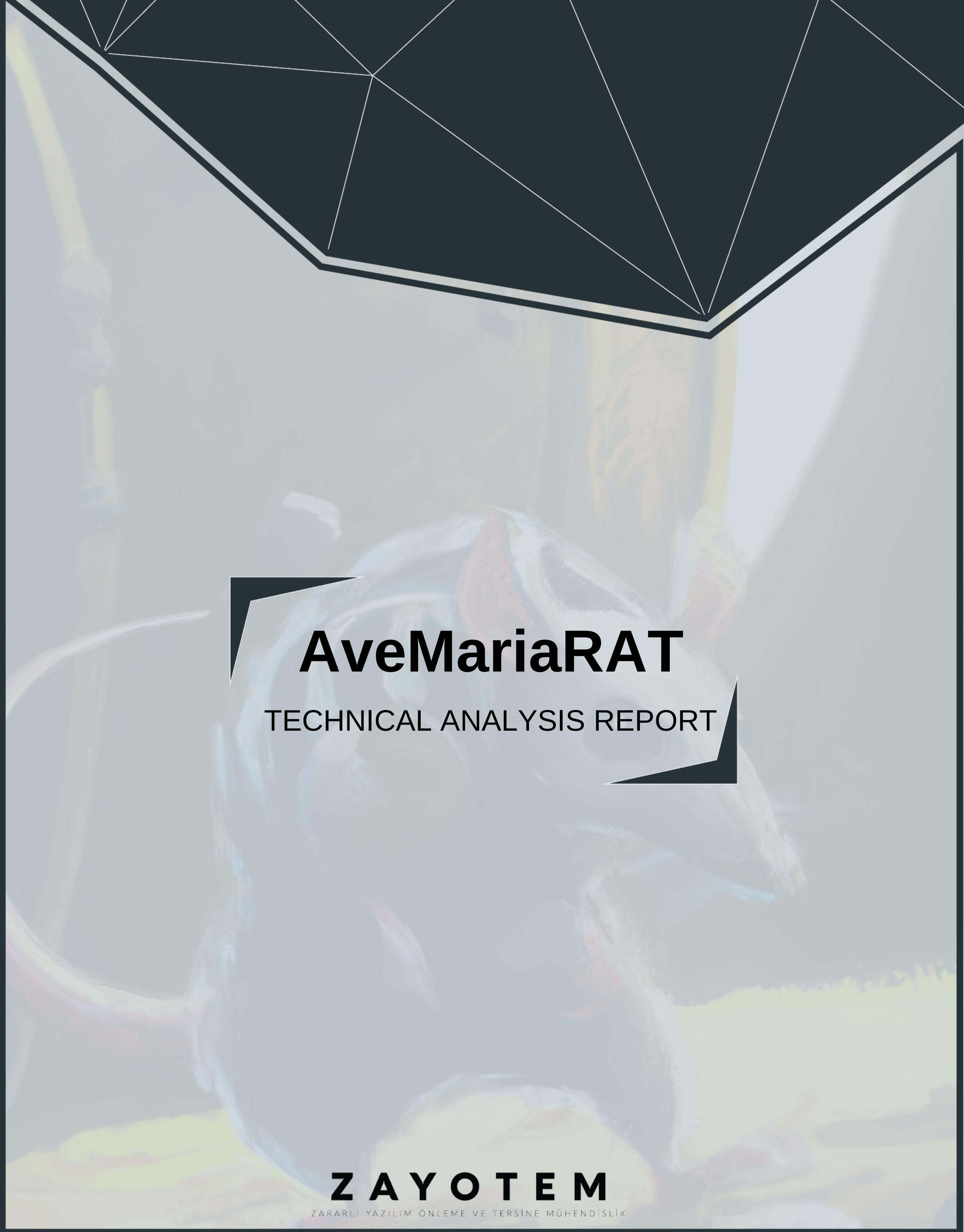


AveMariaRAT

TECHNICAL ANALYSIS REPORT

ZAYOTEM

ZARARLI YAZILIM ÖNLEME VE TERSİNE MÜHENDİSLİK

Contents

Overview

AveMariaRAT is a type of malicious software, also known as Warzone RAT. It is typically used to gain remote access capabilities by infecting systems. This Trojan was first spread through malicious phishing campaigns in 2018 and has since become more visible. Methods such as social engineering, email attachments, and malicious websites are used to infect users. Remote Access Tools (RATs) like AveMariaRAT can pose a serious risk to users' computer systems by being used by cybercriminals for espionage, data theft, and other malicious activities. This malicious software, once it infects a computer, exhibits behaviors such as:

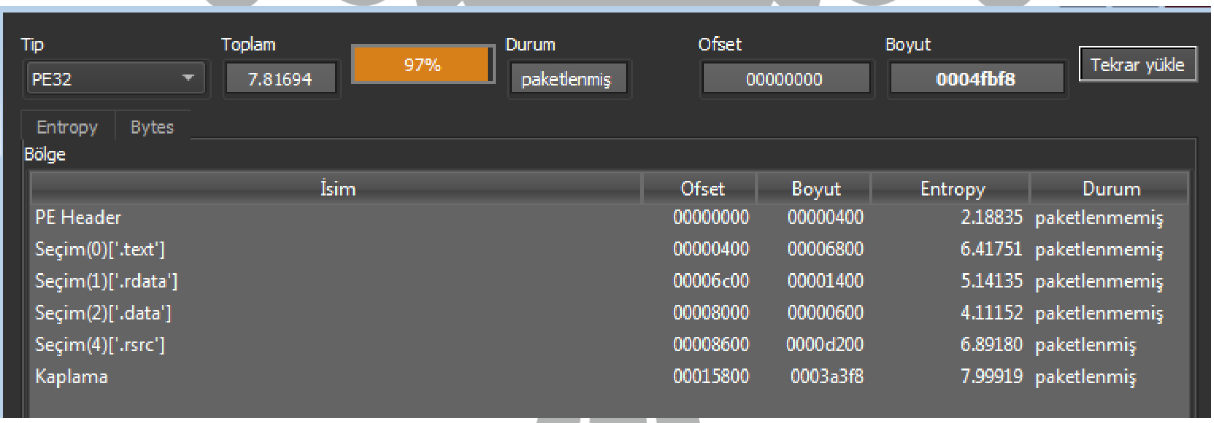
- Remote control access,
- Downloading and deleting files,
- Recording keystrokes,
- Monitoring system information,
- Gaining access to data on browsers.

AveMariaRAT.exe Analysis

Name	AveMariaRAT.exe
MD5	d802bc50f7321efb13358d27280910ca
SHA256	45c59e6d1a36e978efffba98230fe70262b68748ff190562d2f2b8ccad7c43c7
File Type	Portable Executable 32 (x86)

The MD5, SHA256, and other such information about the malware are listed in the table above. The original name of the malware is “45c59e6d1a36e978efffba98230fe70262b68748ff190562d2f2b8ccad7c43c7.exe”, but for ease of analysis, it has been renamed to **“AveMariaRAT.exe”.**

Static Analysis



Tip: PE32
Toplam: 7.81694
97%
Durum: paketlenmiş
Ofset: 00000000
Boyut: 0004fbf8

İsim	Ofset	Boyut	Entropy	Durum
PE Header	00000000	00000400	2.18835	paketlenmemiş
Seçim(0)['.text']	00000400	00006800	6.41751	paketlenmemiş
Seçim(1)['.rdata']	00006c00	00001400	5.14135	paketlenmemiş
Seçim(2)['.data']	00008000	00000600	4.11152	paketlenmemiş
Seçim(4)['.rsrc']	00008600	0000d200	6.89180	paketlenmiş
Kaplama	00015800	0003a3f8	7.99919	paketlenmiş

Figure 1 Examination of the Malware in the DIE Tool

When the AveMariaRAT.exe malware is examined in the DIE tool, it appears to **be packed.**

Advapi32.dll	Shell32.dll	Ole32.dll
Comctl32.dll	User32.dll	Gdi32.dll
Kernel32.dll		

Table 1 Some DLLs Used by the Malware

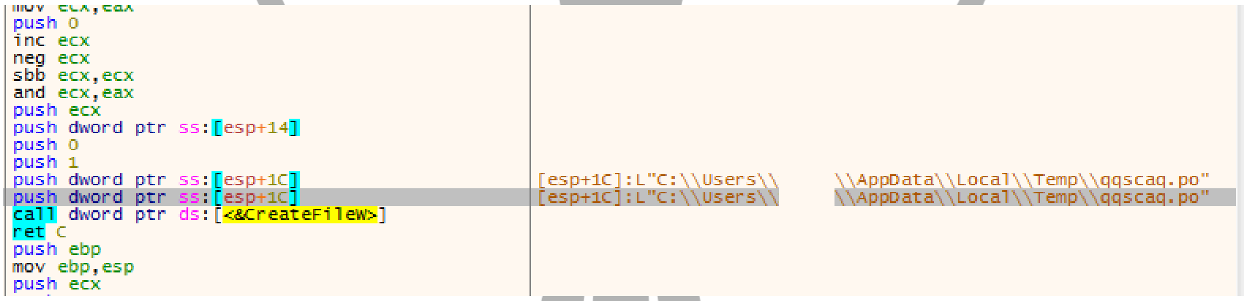
Table 1 shows some of the DLLs used by the malware.

Dynamic Analysis

Uxtheme.dll	Userenv.dll	Setupapi.dll
Apphelp.dll	Propsys.dll	Dwmapi.dll
Cryptbase.dll	Oleacc.dll	Clbcatq.dll
Ntmarta.dll		

Table 2 Dynamically Extracted DLLs

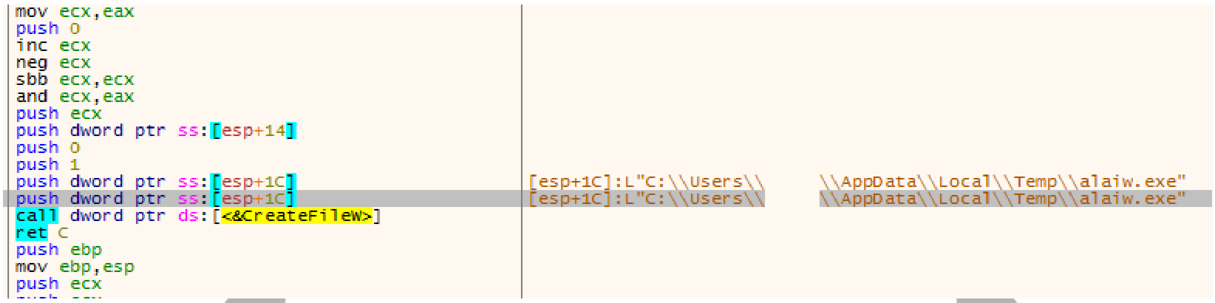
Some of the dynamically loaded DLLs are shown in Table 2.



```
mov ecx,eax
push 0
inc ecx
neg ecx
sbb ecx,ecx
and ecx,eax
push ecx
push dword ptr ss:[esp+14]
push 0
push 1
push dword ptr ss:[esp+1C]               [esp+1C]:L"C:\\Users\\      \\AppData\\Local\\Temp\\qqscaq.po"
push dword ptr ss:[esp+1C]               [esp+1C]:L"C:\\Users\\      \\AppData\\Local\\Temp\\qqscaq.po"
call dword ptr ds:[<&CreateFileW>]
ret C
push ebp
mov ebp,esp
push ecx
```

Figure 2 Creation of the “qqscaq.po” File Using the CreateFileW API

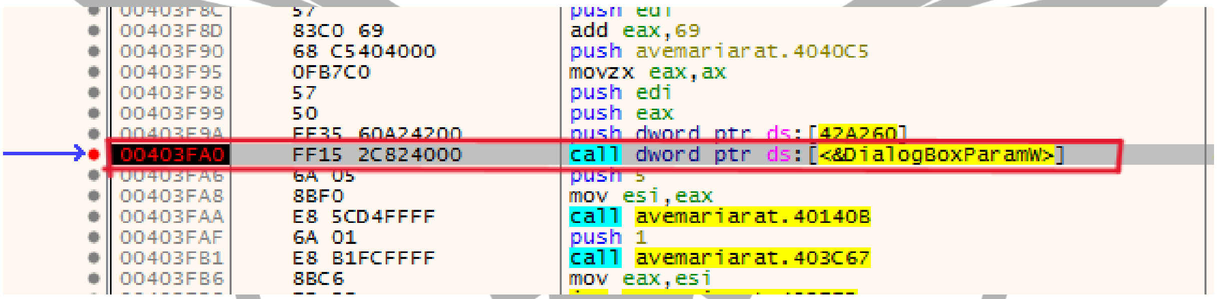
The malware creates a file named **“qqscaq.po”** in the **“C:\Users\%username%\AppData\Local\Temp”** location using the **CreateFileW** API.



```
mov ecx,eax
push 0
inc ecx
neg ecx
sbb ecx,ecx
and ecx,eax
push ecx
push dword ptr ss:[esp+14]
push 0
push 1
push dword ptr ss:[esp+1C]        [esp+1C]:L"C:\\Users\\      \\AppData\\Local\\Temp\\alaiw.exe"
push dword ptr ss:[esp+1C]        [esp+1C]:L"C:\\Users\\      \\AppData\\Local\\Temp\\alaiw.exe"
call dword ptr ds:[<&CreateFileW>]
ret C
push ebp
mov ebp,esp
push ecx
```

Figure 3 Creation of the "alaiw.exe" File Using the CreateFileW API

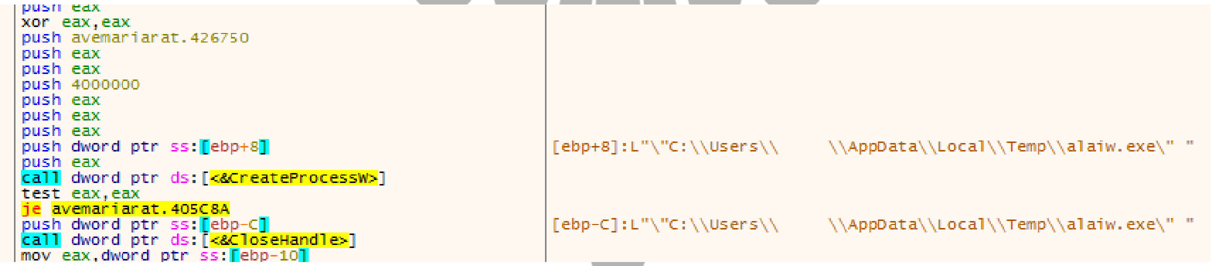
After that, the malware uses the CreateFileW API again to create an executable file (PE) named **"alaiw.exe"** in the **"C:\Users\%username%\AppData\Local\Temp"** location.



```
00403F8C   57              push edi
00403F8D   83C0 69         add eax,69
00403F90   68 C5404000     push avemariarat.4040C5
00403F95   0FB7C0          movzx eax,ax
00403F98   57              push edi
00403F99   50              push eax
00403F9A   FF35 60A24200   push dword ptr ds:[42A260]
00403FA0   FF15 2C824000   call dword ptr ds:[<&DialogBoxParamW>]
00403FA6   6A 05           push 5
00403FA8   8BF0            mov esi,eax
00403FAA   E8 5CD4FFFF     call avemariarat.40140B
00403FAF   6A 01           push 1
00403FB1   E8 B1FCFFFF     call avemariarat.403C67
00403FB6   8BC6            mov eax,esi
```

Figure 4 Execution of Shellcode Using the DialogBoxParamW API

The malware executes the **shellcode** stored in memory using the **DialogBoxParamW** WinAPI call, as shown in Figure 4.



```
push eax
xor eax,eax
push avemariarat.426750
push eax
push eax
push 4000000
push eax
push eax
push eax
push dword ptr ss:[ebp+8]         [ebp+8]:L"\"C:\\Users\\      \\AppData\\Local\\Temp\\alaiw.exe\" "
push eax
call dword ptr ds:[<&CreateProcessW>]
test eax,eax
je avemariarat.405C8A
push dword ptr ss:[ebp-C]         [ebp-C]:L"\"C:\\Users\\      \\AppData\\Local\\Temp\\alaiw.exe\" "
call dword ptr ds:[<&CloseHandle>]
mov eax,dword ptr ss:[ebp-10]
```

Figure 5 Execution of "alaiw.exe" Using the CreateProcessW API

Then, it executes the previously created **"alaiw.exe"** using the **CreateProcessW** API.

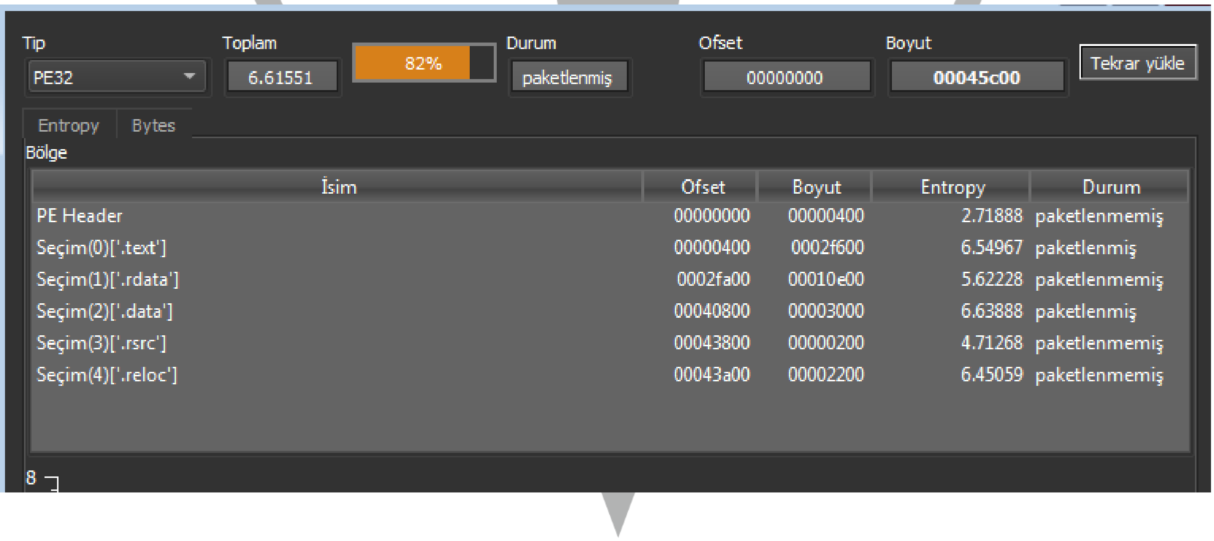
alaiw.exe Analysis

Name	alaiw.exe
MD5	fa0be3eb24b13d060a0ae4e25c22ef1c
SHA256	54152ed7b7386c7a7bef26fafcc72fe3d51ddfbb677292bd9d1261b2c6199ebd
File Type	Portable Executable 32 (x86)

The MD5, SHA256, and other such information of the alaiw.exe file created in the **“C:\Users\%username%\AppData\Local\Temp”** directory within AveMariaRAT.exe are listed in the table above.

Static Analysis

When examined with the DIE tool, alaiw.exe appears to **be packed.**



Tip: PE32
Toplam: 6.61551
82%
Durum: paketlenmiş
Ofset: 00000000
Boyut: 00045c00

İsim	Ofset	Boyut	Entropy	Durum
PE Header	00000000	00000400	2.71888	paketlenmemiş
Seçim(0)['.text']	00000400	0002f600	6.54967	paketlenmiş
Seçim(1)['.rdata']	0002fa00	00010e00	5.62228	paketlenmemiş
Seçim(2)['.data']	00040800	00003000	6.63888	paketlenmiş
Seçim(3)['.rsrc']	00043800	00000200	4.71268	paketlenmemiş
Seçim(4)['.reloc']	00043a00	00002200	6.45059	paketlenmemiş

Figure 6 Examination of alaiw.exe in the DIE Tool

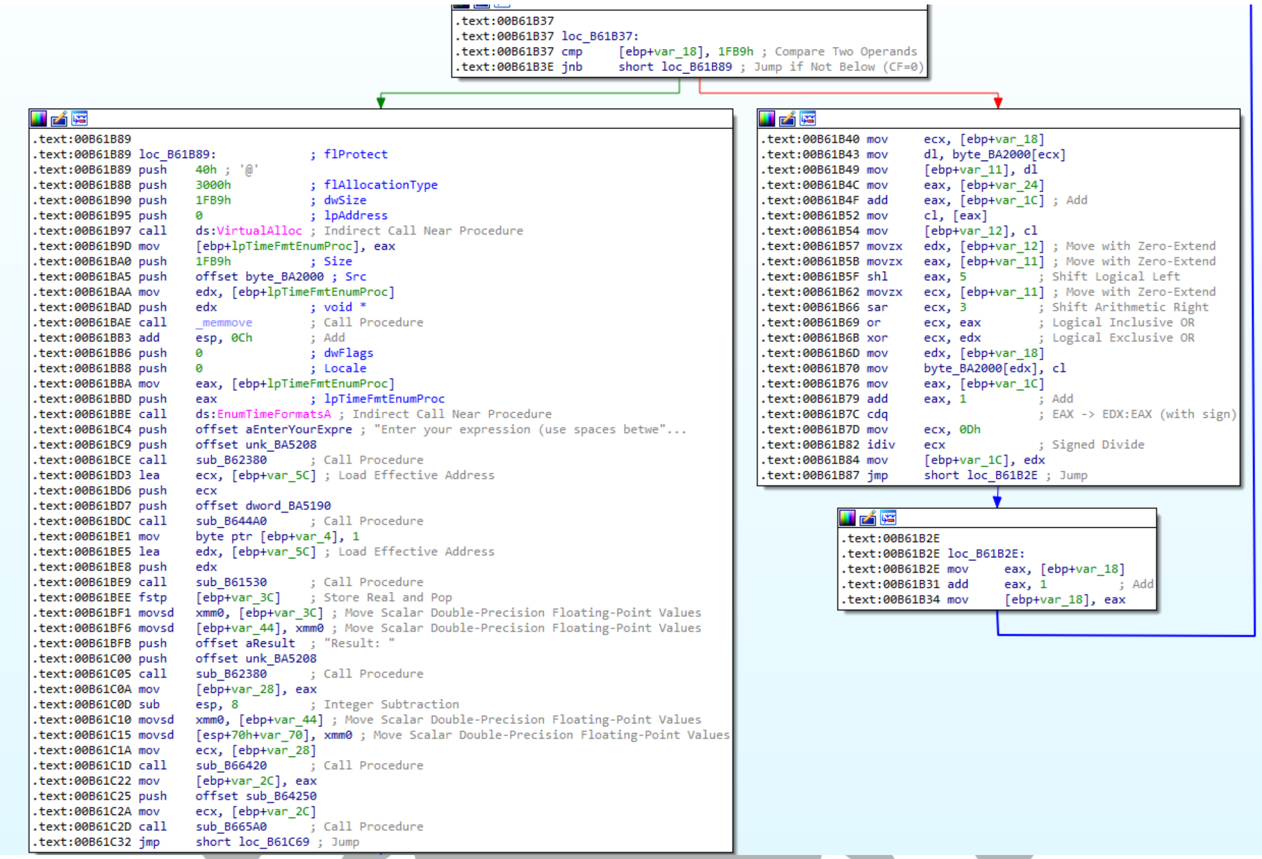
Kernel32.dll	Winspool.drv	Crypt32.dll
Loadperf.dll	Wininet.dll	Rtutils.dll
User32.dll		

Table 3 Some DLLs Used

Table 3 shows some of the DLLs used by the malware.

Dynamic Analysis

The malware decrypts the previously created encrypted **“qqscaq.po”** file using the **“vtwkwntewuzvb”** key phrase. In this way, it creates the shellcode. This code is then copied into memory allocated by the **VirtualAlloc** API using the memmove function. Then, it executes the shellcode using the **EnumTimeFormatsA** API.



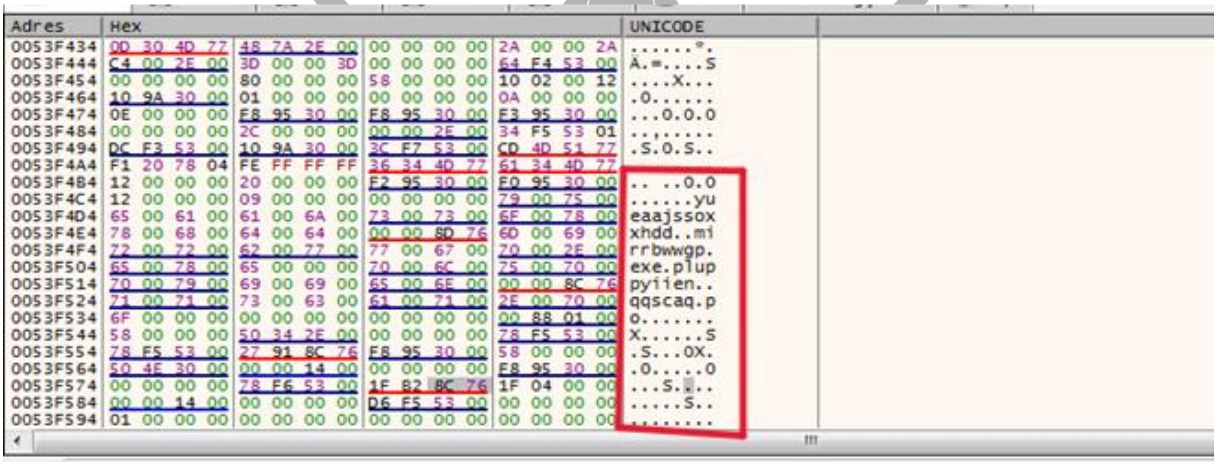
```
.text:00B61B37
.text:00B61B37 loc_B61B37:
.text:00B61B37 cmp     [ebp+var_18], 1FB9h ; Compare Two Operands
.text:00B61B3E jnb     short loc_B61B89 ; Jump if Not Below (CF=0)
```

```
.text:00B61B89
.text:00B61B89 loc_B61B89:             ; flProtect
.text:00B61B89 push    40h ; '@'
.text:00B61B8B push    3000h           ; flAllocationType
.text:00B61B90 push    1FB9h           ; dwSize
.text:00B61B95 push    0               ; lpAddress
.text:00B61B97 call    ds:VirtualAlloc ; Indirect Call Near Procedure
.text:00B61B9D mov     [ebp+lpTimeFmtEnumProc], eax
.text:00B61BA0 push    1FB9h           ; Size
.text:00B61BA5 push    offset byte_BA2000 ; Src
.text:00B61BAA mov     edx, [ebp+lpTimeFmtEnumProc]
.text:00B61BAD push    edx             ; void *
.text:00B61BAE call    _memmove        ; Call Procedure
.text:00B61BB3 add     esp, 0Ch        ; Add
.text:00B61BB6 push    0               ; dwFlags
.text:00B61BB8 push    0               ; Locale
.text:00B61BBA mov     eax, [ebp+lpTimeFmtEnumProc]
.text:00B61BBD push    eax             ; lpTimeFmtEnumProc
.text:00B61BBE call    ds:EnumTimeFormatsA ; Indirect Call Near Procedure
.text:00B61BC4 push    offset aEnterYourExpre ; "Enter your expression (use spaces betwe"...
.text:00B61BC9 push    offset unk_BA5208
.text:00B61BCE call    sub_B62380      ; Call Procedure
.text:00B61BD3 lea     ecx, [ebp+var_5C] ; Load Effective Address
.text:00B61BD6 push    ecx
.text:00B61BD7 push    offset dword_BA5190
.text:00B61BDC call    sub_B644A0      ; Call Procedure
.text:00B61BE1 mov     byte ptr [ebp+var_4], 1
.text:00B61BE5 lea     edx, [ebp+var_5C] ; Load Effective Address
.text:00B61BE8 push    edx
.text:00B61BE9 call    sub_B61530      ; Call Procedure
.text:00B61BEE fstp    [ebp+var_3C]    ; Store Real and Pop
.text:00B61BF1 movsd   xmm0, [ebp+var_3C] ; Move Scalar Double-Precision Floating-Point Values
.text:00B61BF6 movsd   [ebp+var_44], xmm0 ; Move Scalar Double-Precision Floating-Point Values
.text:00B61BFB push    offset aResult  ; "Result: "
.text:00B61C00 push    offset unk_BA5208
.text:00B61C05 call    sub_B62380      ; Call Procedure
.text:00B61C0A mov     [ebp+var_28], eax
.text:00B61C0D sub     esp, 8          ; Integer Subtraction
.text:00B61C10 movsd   xmm0, [ebp+var_44] ; Move Scalar Double-Precision Floating-Point Values
.text:00B61C15 movsd   [esp+70h+var_70], xmm0 ; Move Scalar Double-Precision Floating-Point Values
.text:00B61C1A mov     ecx, [ebp+var_28]
.text:00B61C1D call    sub_B66420      ; Call Procedure
.text:00B61C22 mov     [ebp+var_2C], eax
.text:00B61C25 push    offset sub_B64250
.text:00B61C2A mov     ecx, [ebp+var_2C]
.text:00B61C2D call    sub_B665A0      ; Call Procedure
.text:00B61C32 jmp     short loc_B61C69 ; Jump
```

```
.text:00B61B40 mov     ecx, [ebp+var_18]
.text:00B61B43 mov     dl, byte_BA2000[ecx]
.text:00B61B49 mov     [ebp+var_11], dl
.text:00B61B4C mov     eax, [ebp+var_24]
.text:00B61B4F add     eax, [ebp+var_1C] ; Add
.text:00B61B52 mov     cl, [eax]
.text:00B61B54 mov     [ebp+var_12], cl
.text:00B61B57 movzx   edx, [ebp+var_12] ; Move with Zero-Extend
.text:00B61B5B movzx   eax, [ebp+var_11] ; Move with Zero-Extend
.text:00B61B5F shl     eax, 5          ; Shift Logical Left
.text:00B61B62 movzx   ecx, [ebp+var_11] ; Move with Zero-Extend
.text:00B61B66 sar     ecx, 3          ; Shift Arithmetic Right
.text:00B61B69 or      ecx, eax        ; Logical Inclusive OR
.text:00B61B6B xor     ecx, edx        ; Logical Exclusive OR
.text:00B61B6D mov     edx, [ebp+var_18]
.text:00B61B70 mov     byte_BA2000[edx], cl
.text:00B61B76 mov     eax, [ebp+var_1C]
.text:00B61B79 add     eax, 1          ; Add
.text:00B61B7C cdq                     ; EAX -> EDX:EAX (with sign)
.text:00B61B7D mov     ecx, 0Dh
.text:00B61B82 idiv    ecx             ; Signed Divide
.text:00B61B84 mov     [ebp+var_1C], edx
.text:00B61B87 jmp     short loc_B61B2E ; Jump
```

```
.text:00B61B2E
.text:00B61B2E loc_B61B2E:
.text:00B61B2E mov     eax, [ebp+var_18]
.text:00B61B31 add     eax, 1          ; Add
.text:00B61B34 mov     [ebp+var_18], eax
```

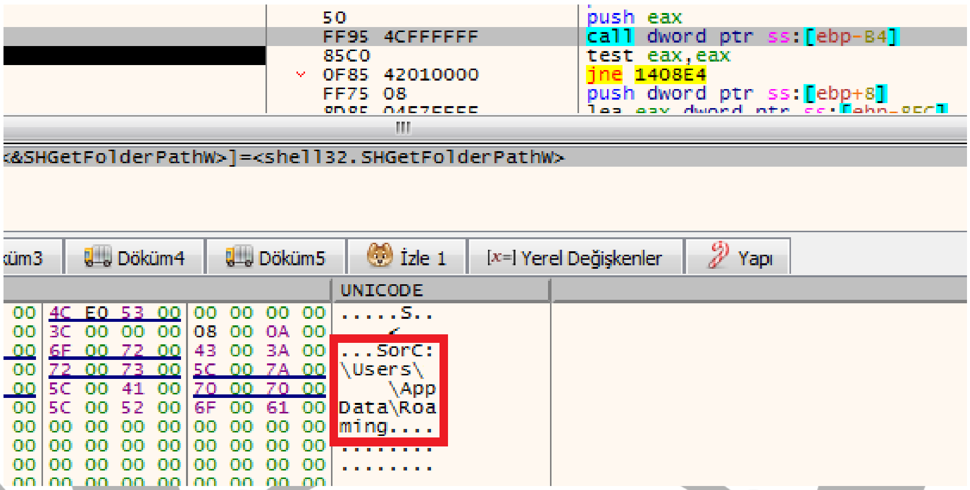
Figure 7 Decryption and Execution of Shellcode



```
Adres     Hex                                                  UNICODE
0053F434  0D 30 4D 77  48 7A 2E 00  00 00 00 00  2A 00 00 2A   ......*.
0053F444  C4 00 2E 00  3D 00 00 3D  00 00 00 00  64 F4 53 00   Ä.=....S
0053F454  00 00 00 00  80 00 00 00  58 00 00 00  10 02 00 12   ....X...
0053F464  10 9A 30 00  01 00 00 00  00 00 00 00  0A 00 00 00   .0......
0053F474  0E 00 00 00  F8 95 30 00  F8 95 30 00  F3 95 30 00   ...0.0.0
0053F484  00 00 00 00  2C 00 00 00  00 00 2E 00  34 F5 53 01   ..,.....
0053F494  DC F3 53 00  10 9A 30 00  3C F7 53 00  CD 4D 51 77   .S.0.S..
0053F4A4  F1 20 78 04  FE FF FF FF  36 34 4D 77  61 34 4D 77
0053F4B4  12 00 00 00  20 00 00 00  F2 95 30 00  F0 95 30 00   .. ..0.0
0053F4C4  12 00 00 00  09 00 00 00  00 00 00 00  79 00 75 00   ......yu
0053F4D4  65 00 61 00  61 00 6A 00  73 00 73 00  6F 00 78 00   eaajssox
0053F4E4  78 00 68 00  64 00 64 00  00 00 8D 76  6D 00 69 00   xhdd..mi
0053F4F4  72 00 72 00  62 00 77 00  77 00 67 00  70 00 2E 00   rrbwwgp.
0053F504  65 00 78 00  65 00 00 00  70 00 6C 00  75 00 70 00   exe.plup
0053F514  70 00 79 00  69 00 69 00  65 00 6E 00  00 00 8C 76   pyiien..
0053F524  71 00 71 00  73 00 63 00  61 00 71 00  2E 00 70 00   qqscaq.p
0053F534  6F 00 00 00  00 00 00 00  00 00 00 00  00 88 01 00   o.......
0053F544  58 00 00 00  50 34 2E 00  00 00 00 00  78 F5 53 00   X......S
0053F554  78 F5 53 00  27 91 8C 76  F8 95 30 00  58 00 00 00   .S...0X.
0053F564  50 4E 30 00  00 00 14 00  00 00 00 00  F8 95 30 00   .0.....0
0053F574  00 00 00 00  78 F6 53 00  1F B2 8C 76  1F 04 00 00   ...S....
0053F584  00 00 14 00  00 00 00 00  D6 F5 53 00  00 00 00 00   .....S..
0053F594  01 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ........
```

Figure 8 Decrypted Strings within the Shellcode

The malware then places the strings **yueaajssoxxhdd**, **mirrbwwgp.exe**, **pluppyiien**, and **qqscaq.po** in memory using **deobfuscation.**



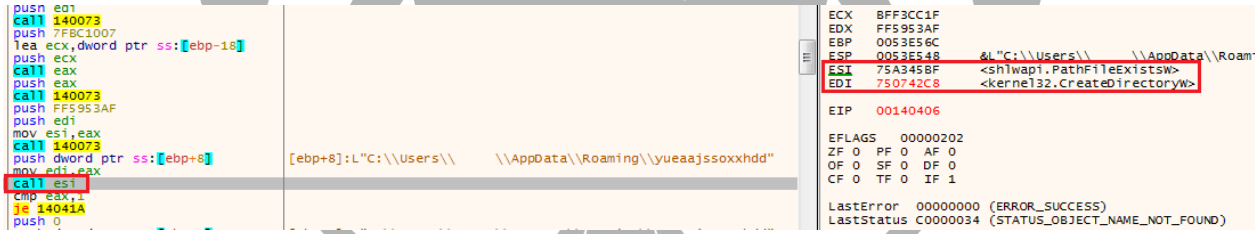
```
[redacted]
    50                      push eax
    FF95 4CFFFFFF           call dword ptr ss:[ebp-B4]
    85C0                    test eax,eax
    0F85 42010000           jne 1408E4
    FF75 08                 push dword ptr ss:[ebp+8]
```

```
<&SHGetFolderPathW>]=<shell32.SHGetFolderPathW>
```

```
                                      UNICODE
00 4C E0 53 00 00 00 00 00   .....S..
00 3C 00 00 00 08 00 0A 00
00 6F 00 72 00 43 00 3A 00   ...SorC:
00 72 00 73 00 5C 00 7A 00   \Users\
00 5C 00 41 00 70 00 70 00       \App
00 5C 00 52 00 6F 00 61 00   Data\Roa
00 00 00 00 00 00 00 00 00   ming....
00 00 00 00 00 00 00 00 00   ........
00 00 00 00 00 00 00 00 00   ........
```

Figure 9 Accessing File Path Using the SHGetFolderPathW API

The malware accesses the file path **“C:\Users\%username%\AppData\Roaming”** using the **SHGetFolderPathW** API.

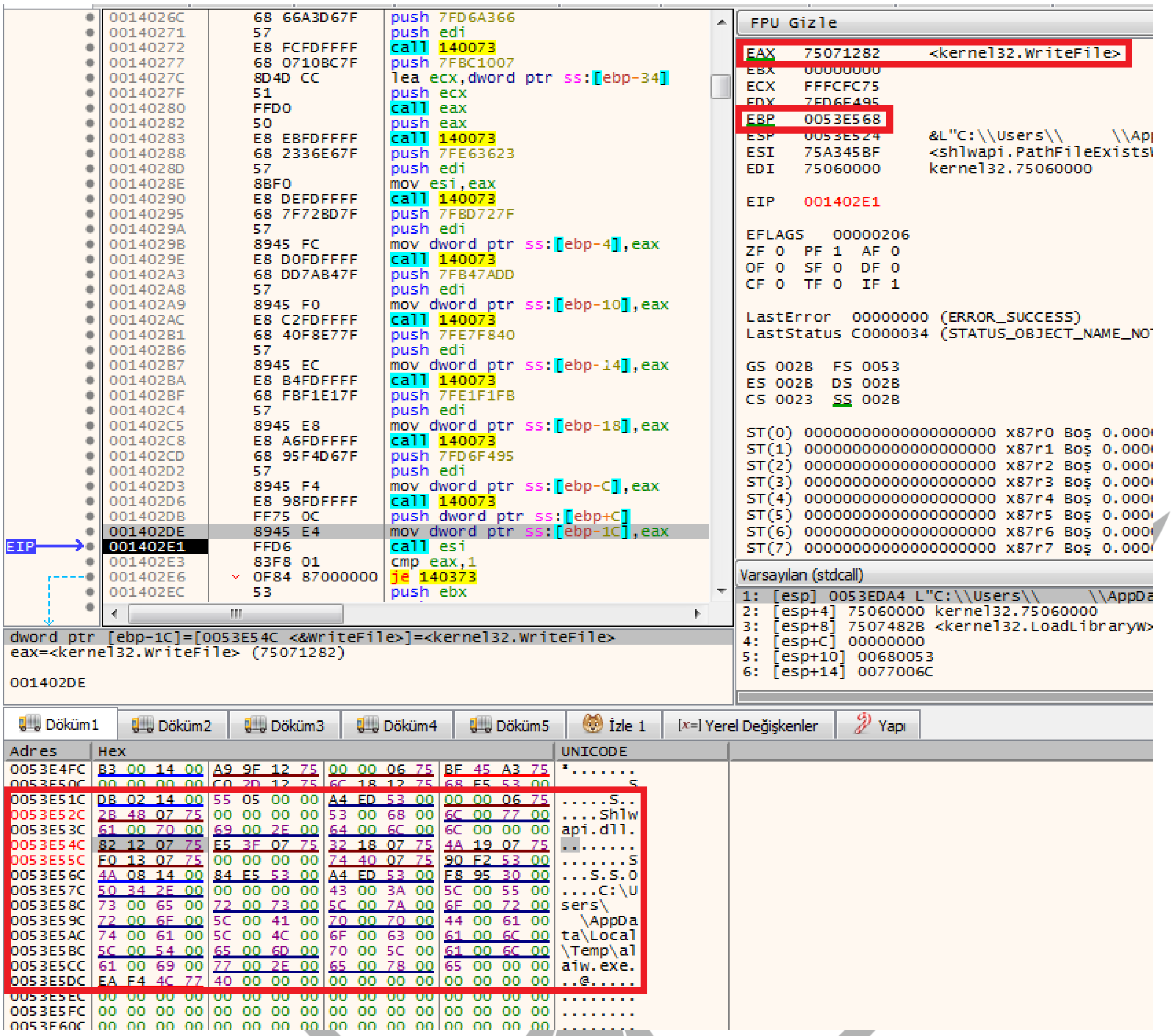


```
push edi
call 140073
push 7FBC1007
lea ecx,dword ptr ss:[ebp-18]
push ecx
call eax
push eax
call 140073
push FF5953AF
push edi
mov esi,eax
call 140073
push dword ptr ss:[ebp+8]          [ebp+8]:L"C:\\Users\\      \\AppData\\Roaming\\yueaajssoxxhdd"
mov edi,eax
call esi
cmp eax,1
je 14041A
push 0
```

```
ECX   BFF3CC1F
EDX   FF5953AF
EBP   0053E56C
ESP   0053E548      &L"C:\\Users\\      \\AppData\\Roam
ESI   75A345BF      <shlwapi.PathFileExistsW>
EDI   750742C8      <kernel32.CreateDirectoryW>

EIP   00140406

EFLAGS    00000202
ZF 0  PF 0  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C0000034 (STATUS_OBJECT_NAME_NOT_FOUND)
```

Figure 10 "PathFileExistW" and "CreateDirectoryW" APIs

It checks for the existence of the directory **“C:\Users\%username%\AppData\Roaming\yueaajssoxxhdd”** using the **PathFileExistW** API, and if it doesn’t exist, creates it using the **CreateDirectoryW** API.



```
0014026C   68 66A3D67F       push 7FD6A366
00140271   57                push edi
00140272   E8 FCFDFFFF       call 140073
00140277   68 0710BC7F       push 7FBC1007
0014027C   8D4D CC           lea ecx,dword ptr ss:[ebp-34]
0014027F   51                push ecx
00140280   FFD0              call eax
00140282   50                push eax
00140283   E8 EBFDFFFF       call 140073
00140288   68 2336E67F       push 7FE63623
0014028D   57                push edi
0014028E   8BF0              mov esi,eax
00140290   E8 DEFDFFFF       call 140073
00140295   68 7F72BD7F       push 7FBD727F
0014029A   57                push edi
0014029B   8945 FC           mov dword ptr ss:[ebp-4],eax
0014029E   E8 D0FDFFFF       call 140073
001402A3   68 DD7AB47F       push 7FB47ADD
001402A8   57                push edi
001402A9   8945 F0           mov dword ptr ss:[ebp-10],eax
001402AC   E8 C2FDFFFF       call 140073
001402B1   68 40F8E77F       push 7FE7F840
001402B6   57                push edi
001402B7   8945 EC           mov dword ptr ss:[ebp-14],eax
001402BA   E8 B4FDFFFF       call 140073
001402BF   68 FBF1E17F       push 7FE1F1FB
001402C4   57                push edi
001402C5   8945 E8           mov dword ptr ss:[ebp-18],eax
001402C8   E8 A6FDFFFF       call 140073
001402CD   68 95F4D67F       push 7FD6F495
001402D2   57                push edi
001402D3   8945 F4           mov dword ptr ss:[ebp-C],eax
001402D6   E8 98FDFFFF       call 140073
001402DB   FF75 0C           push dword ptr ss:[ebp+C]
001402DE   8945 E4           mov dword ptr ss:[ebp-1C],eax
001402E1   FFD6              call esi
001402E3   83F8 01           cmp eax,1
001402E6   0F84 87000000     je 140373
001402EC   53                push ebx

dword ptr [ebp-1C]=[0053E54C <&WriteFile>]=<kernel32.WriteFile>
eax=<kernel32.WriteFile> (75071282)

001402DE
```

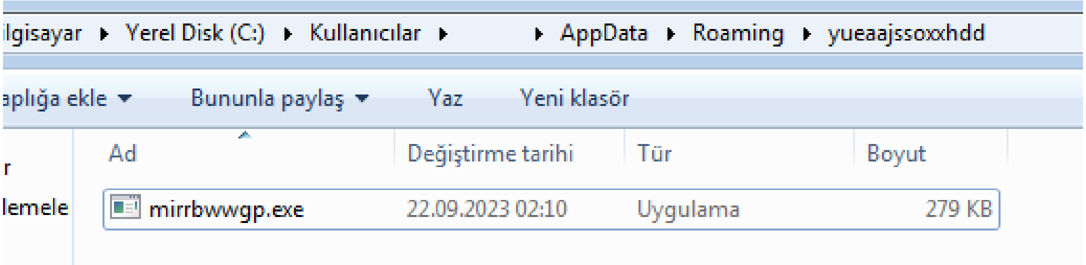
```
EAX   75071282     <kernel32.WriteFile>
EBX   00000000
ECX   FFFCFC75
EDX   7FD6F495
EBP   0053E568
ESP   0053E524     &L"C:\\Users\\      \\Ap
ESI   75A345BF     <shlwapi.PathFileExists
EDI   75060000     kernel32.75060000

EIP   001402E1

EFLAGS   00000206
ZF 0  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C0000034 (STATUS_OBJECT_NAME_NO

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Boş 0.000
ST(1) 00000000000000000000 x87r1 Boş 0.000
ST(2) 00000000000000000000 x87r2 Boş 0.000
ST(3) 00000000000000000000 x87r3 Boş 0.000
ST(4) 00000000000000000000 x87r4 Boş 0.000
ST(5) 00000000000000000000 x87r5 Boş 0.000
ST(6) 00000000000000000000 x87r6 Boş 0.000
ST(7) 00000000000000000000 x87r7 Boş 0.000
```

```
1: [esp] 0053EDA4 L"C:\\Users\\      \\AppDa
2: [esp+4] 75060000 kernel32.75060000
3: [esp+8] 7507482B <kernel32.LoadLibraryW>
4: [esp+C] 00000000
5: [esp+10] 00680053
6: [esp+14] 0077006C
```

```
Adres     Hex                                                UNICODE
0053E4FC  B3 00 14 00  A9 9F 12 75  00 00 06 75  BF 45 A3 75  *.......
0053E50C  00 00 00 00  C0 2D 12 75  6C 18 12 75  68 E5 53 00        S
0053E51C  DB 02 14 00  55 05 00 00  A4 ED 53 00  00 00 06 75  .....S..
0053E52C  2B 48 07 75  00 00 00 00  53 00 68 00  6C 00 77 00  ....Shlw
0053E53C  61 00 70 00  69 00 2E 00  64 00 6C 00  6C 00 00 00  api.dll.
0053E54C  82 12 07 75  E5 3F 07 75  32 18 07 75  4A 19 07 75  ........
0053E55C  F0 13 07 75  00 00 00 00  74 40 07 75  90 F2 53 00  .......S
0053E56C  4A 08 14 00  84 E5 53 00  A4 ED 53 00  F8 95 30 00  ...S.S.0
0053E57C  50 34 2E 00  00 00 00 00  43 00 3A 00  5C 00 55 00  ....C:\U
0053E58C  73 00 65 00  72 00 73 00  5C 00 7A 00  6F 00 72 00  sers\
0053E59C  72 00 6F 00  5C 00 41 00  70 00 70 00  44 00 61 00    \AppDa
0053E5AC  74 00 61 00  5C 00 4C 00  6F 00 63 00  61 00 6C 00  ta\Local
0053E5BC  5C 00 54 00  65 00 6D 00  70 00 5C 00  61 00 6C 00  \Temp\al
0053E5CC  61 00 69 00  77 00 2E 00  65 00 78 00  65 00 00 00  aiw.exe.
0053E5DC  EA F4 4C 77  40 00 00 00  00 00 00 00  00 00 00 00  ..@.....
0053E5EC  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ........
0053E5FC  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ........
0053E60C  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ........
```

Figure 11 Dynamic API Resolution

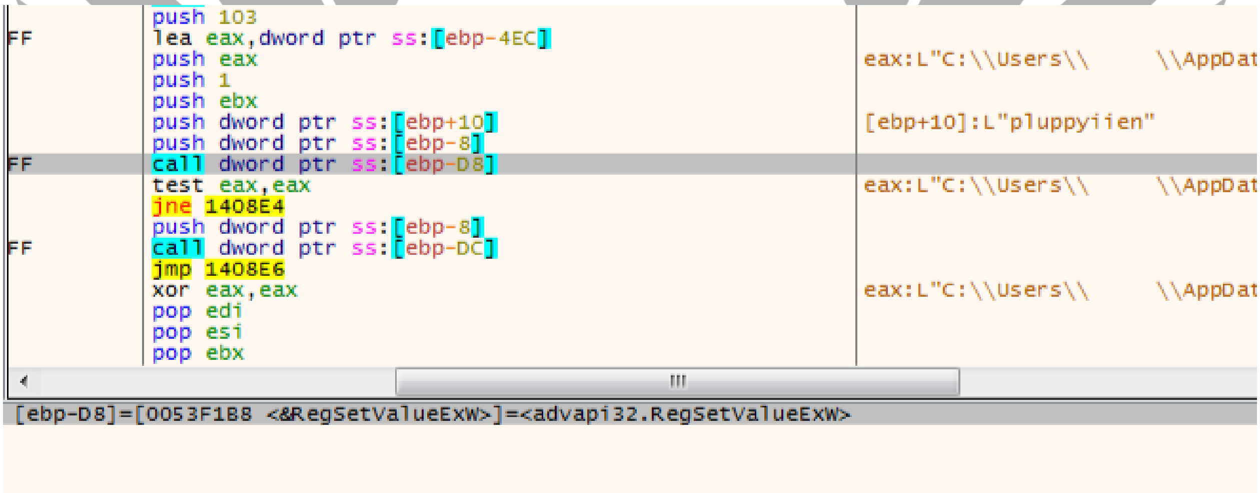
The call instructions shown in Figure 11, respectively, resolve the **LoadLibrary**, **PathFileExistsW**, **CreateFileW**, **GetFileSize**, **VirtualAlloc**, **ReadFile**, **CloseHandle**, and **WriteFile** APIs and store their addresses in memory.


lgisayar ▸ Yerel Disk (C:) ▸ Kullanıcılar ▸ ▸ AppData ▸ Roaming ▸ yueaajssoxxhdd
apliğa ekle ▾
Bununla paylaş ▾
Yaz
Yeni klasör
r
lemele

Ad	Değiştirme tarihi	Tür	Boyut
mirrbwwgp.exe	22.09.2023 02:10	Uygulama	279 KB

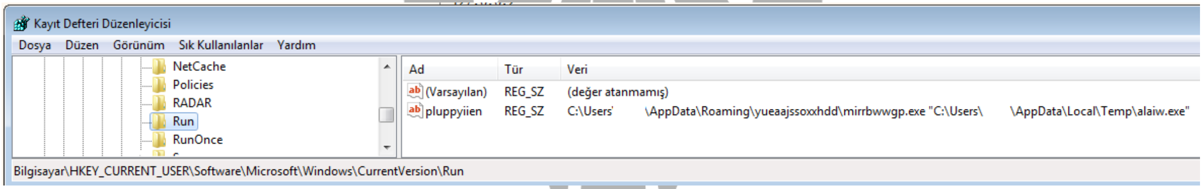
Figure 12 alaiw.exe Copying Itself as mirrbwwgp.exe

Using these APIs, it copies itself as **"mirrbwwgp.exe"** into the directory **“C:\Users\%username%\AppData\Roaming\yueaajssoxxhdd”**.



```
           push 103
FF         lea eax,dword ptr ss:[ebp-4EC]
           push eax                          eax:L"C:\\Users\\      \\AppDat
           push 1
           push ebx
           push dword ptr ss:[ebp+10]        [ebp+10]:L"pluppyiien"
           push dword ptr ss:[ebp-8]
FF         call dword ptr ss:[ebp-D8]
           test eax,eax                      eax:L"C:\\Users\\      \\AppDat
           jne 1408E4
           push dword ptr ss:[ebp-8]
FF         call dword ptr ss:[ebp-DC]
           jmp 1408E6
           xor eax,eax                       eax:L"C:\\Users\\      \\AppDat
           pop edi
           pop esi
           pop ebx
[ebp-D8]=[0053F1B8 <&RegSetValueExW>]=<advapi32.RegSetValueExW>
```

Figure 13 Ensuring Persistence with RegSetValueExW

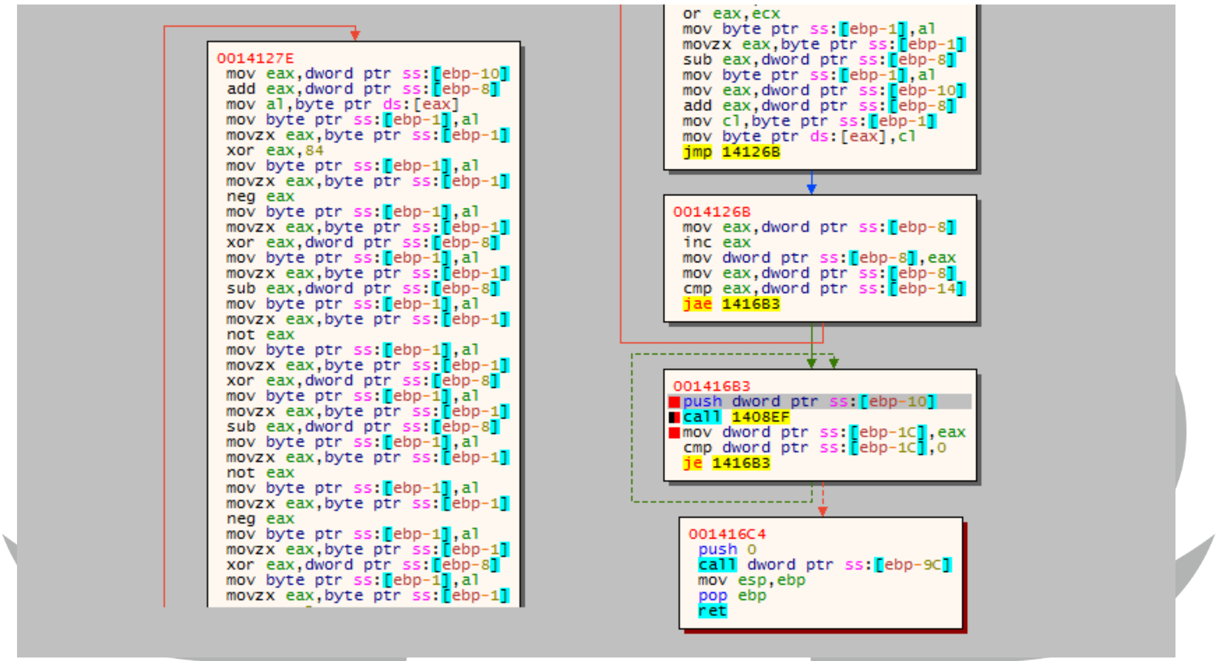


Ad	Tür	Veri
(Varsayılan)	REG_SZ	(değer atanmamış)
pluppyiien	REG_SZ	C:\Users' \AppData\Roaming\yueaajssoxxhdd\mirrbwwgp.exe "C:\Users\ \AppData\Local\Temp\alaiw.exe"

Bilgisayar\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

Figure 14 Creation of the "pluppyiien" Key

To ensure its persistence, it accesses the registry and creates the **“pluppyiien”** key at the location **HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run.** This way, the malware executes itself every time the system starts.

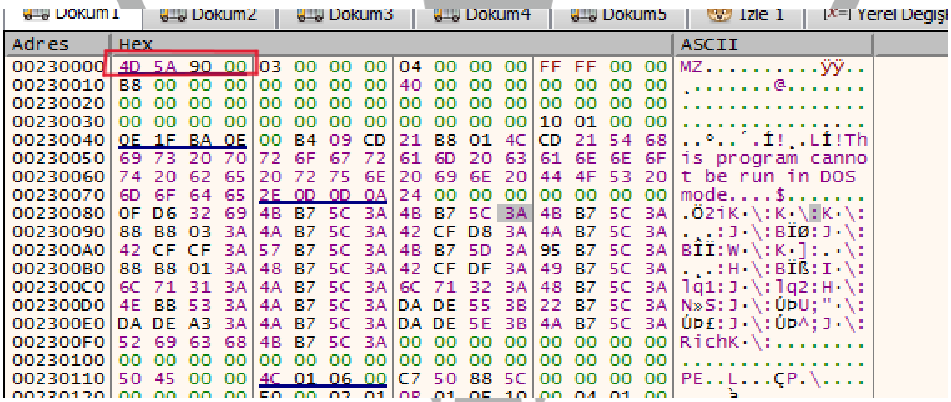


```
0014127E
 mov eax,dword ptr ss:[ebp-10]
 add eax,dword ptr ss:[ebp-8]
 mov al,byte ptr ds:[eax]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 xor eax,84
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 neg eax
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 xor eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 sub eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 not eax
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 xor eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 sub eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 not eax
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 neg eax
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 xor eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
```

```
 or eax,ecx
 mov byte ptr ss:[ebp-1],al
 movzx eax,byte ptr ss:[ebp-1]
 sub eax,dword ptr ss:[ebp-8]
 mov byte ptr ss:[ebp-1],al
 mov eax,dword ptr ss:[ebp-10]
 add eax,dword ptr ss:[ebp-8]
 mov cl,byte ptr ss:[ebp-1]
 mov byte ptr ds:[eax],cl
 jmp 14126B

0014126B
 mov eax,dword ptr ss:[ebp-8]
 inc eax
 mov dword ptr ss:[ebp-8],eax
 mov eax,dword ptr ss:[ebp-8]
 cmp eax,dword ptr ss:[ebp-14]
 jae 1416B3

001416B3
 push dword ptr ss:[ebp-10]
 call 1408EF
 mov dword ptr ss:[ebp-1C],eax
 cmp dword ptr ss:[ebp-1C],0
 je 1416B3

001416C4
 push 0
 call dword ptr ss:[ebp-9C]
 mov esp,ebp
 pop ebp
 ret
```

Figure 15 Algorithm Used to Decrypt Data

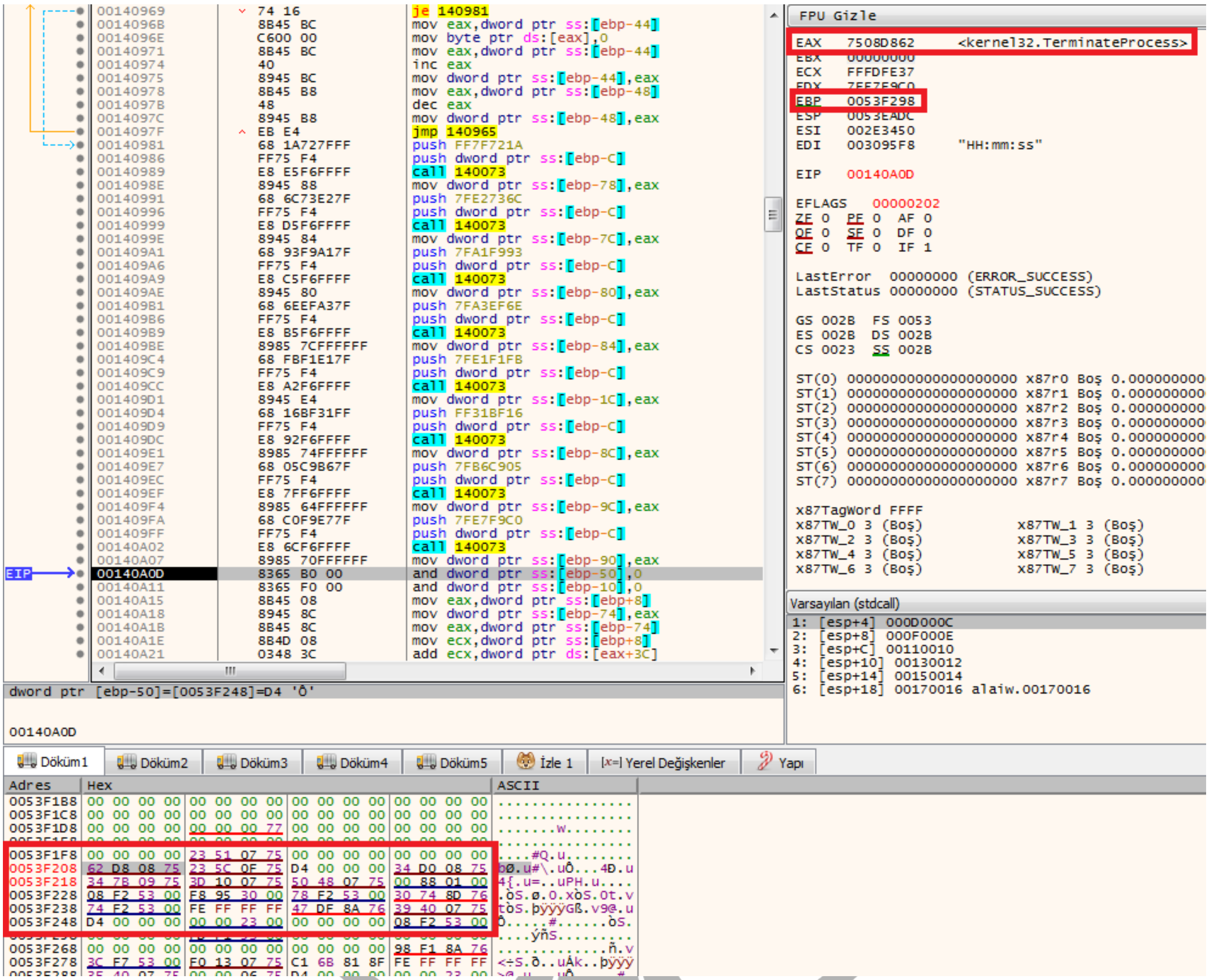
The malware then uses a unique algorithm to decrypt **encrypted data**.



```
Adres     Hex                                              ASCII
00230000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
00230010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸.......@.......
00230020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00230030  00 00 00 00 00 00 00 00 00 00 00 00 10 01 00 00  ................
00230040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
00230050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00230060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS 
00230070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00230080  0F D6 32 69 4B B7 5C 3A 4B B7 5C 3A 4B B7 5C 3A  .Ö2iK·\:K·\:K·\:
00230090  88 B8 03 3A 4A B7 5C 3A 42 CF D8 3A 4A B7 5C 3A  .¸.:J·\:BÏØ:J·\:
002300A0  42 CF CF 3A 57 B7 5C 3A 4B B7 5D 3A 95 B7 5C 3A  BÏÏ:W·\:K·]:.·\:
002300B0  88 B8 01 3A 48 B7 5C 3A 42 CF DF 3A 49 B7 5C 3A  .¸.:H·\:BÏß:I·\:
002300C0  6C 71 31 3A 4A B7 5C 3A 6C 71 32 3A 48 B7 5C 3A  lq1:J·\:lq2:H·\:
002300D0  4E BB 53 3A 4A B7 5C 3A DA DE 55 3B 22 B7 5C 3A  N»S:J·\:ÚÞU;"·\:
002300E0  DA DE A3 3A 4A B7 5C 3A DA DE 5E 3B 4A B7 5C 3A  ÚÞ£:J·\:ÚÞ^;J·\:
002300F0  52 69 63 68 4B B7 5C 3A 00 00 00 00 00 00 00 00  RichK·\:........
00230100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00230110  50 45 00 00 4C 01 06 00 C7 50 88 5C 00 00 00 00  PE..L...ÇP.\....
```

Figure 16 The Memory Region Containing the Executable File

When this data is examined in the memory region, it appears to be an **executable file**.



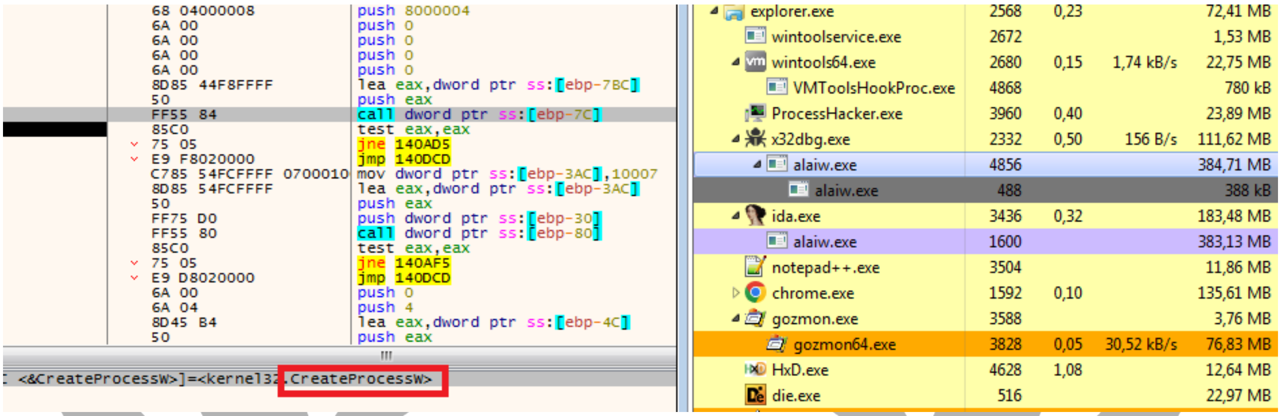
```
00140969   74 16          je 140981
0014096B   8B45 BC        mov eax,dword ptr ss:[ebp-44]
0014096E   C600 00        mov byte ptr ds:[eax],0
00140971   8B45 BC        mov eax,dword ptr ss:[ebp-44]
00140974   40             inc eax
00140975   8945 BC        mov dword ptr ss:[ebp-44],eax
00140978   8B45 B8        mov eax,dword ptr ss:[ebp-48]
0014097B   48             dec eax
0014097C   8945 B8        mov dword ptr ss:[ebp-48],eax
0014097F   EB E4          jmp 140965
00140981   68 1A727FFF    push FF7F721A
00140986   FF75 F4        push dword ptr ss:[ebp-C]
00140989   E8 E5F6FFFF    call 140073
0014098E   8945 88        mov dword ptr ss:[ebp-78],eax
00140991   68 6C73E27F    push 7FE2736C
00140996   FF75 F4        push dword ptr ss:[ebp-C]
00140999   E8 D5F6FFFF    call 140073
0014099E   8945 84        mov dword ptr ss:[ebp-7C],eax
001409A1   68 93F9A17F    push 7FA1F993
001409A6   FF75 F4        push dword ptr ss:[ebp-C]
001409A9   E8 C5F6FFFF    call 140073
001409AE   8945 80        mov dword ptr ss:[ebp-80],eax
001409B1   68 6EEFA37F    push 7FA3EF6E
001409B6   FF75 F4        push dword ptr ss:[ebp-C]
001409B9   E8 B5F6FFFF    call 140073
001409BE   8985 7CFFFFFF  mov dword ptr ss:[ebp-84],eax
001409C4   68 FBF1E17F    push 7FE1F1FB
001409C9   FF75 F4        push dword ptr ss:[ebp-C]
001409CC   E8 A2F6FFFF    call 140073
001409D1   8945 E4        mov dword ptr ss:[ebp-1C],eax
001409D4   68 16BF31FF    push FF31BF16
001409D9   FF75 F4        push dword ptr ss:[ebp-C]
001409DC   E8 92F6FFFF    call 140073
001409E1   8985 74FFFFFF  mov dword ptr ss:[ebp-8C],eax
001409E7   68 05C9B67F    push 7FB6C905
001409EC   FF75 F4        push dword ptr ss:[ebp-C]
001409EF   E8 7FF6FFFF    call 140073
001409F4   8985 64FFFFFF  mov dword ptr ss:[ebp-9C],eax
001409FA   68 C0F9E77F    push 7FE7F9C0
001409FF   FF75 F4        push dword ptr ss:[ebp-C]
00140A02   E8 6CF6FFFF    call 140073
00140A07   8985 70FFFFFF  mov dword ptr ss:[ebp-90],eax
EIP → 00140A0D   8365 B0 00     and dword ptr ss:[ebp-50],0
00140A11   8365 F0 00     and dword ptr ss:[ebp-10],0
00140A15   8B45 08        mov eax,dword ptr ss:[ebp+8]
00140A18   8945 8C        mov dword ptr ss:[ebp-74],eax
00140A1B   8B45 8C        mov eax,dword ptr ss:[ebp-74]
00140A1E   8B4D 08        mov ecx,dword ptr ss:[ebp+8]
00140A21   0348 3C        add ecx,dword ptr ds:[eax+3C]

dword ptr [ebp-50]=[0053F248]=D4 'Ô'

00140A0D
```

```
FPU Gizle

EAX   7508D862     <kernel32.TerminateProcess>
EBX   00000000
ECX   FFFDFE37
EDX   7FE7F9C0
EBP   0053F298
ESP   0053EADC
ESI   002E3450
EDI   003095F8     "HH:mm:ss"

EIP   00140A0D

EFLAGS   00000202
ZF 0  PF 0  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0023  SS 002B

ST(0) 00000000000000000000 x87r0 Boş 0.000000000
ST(1) 00000000000000000000 x87r1 Boş 0.000000000
ST(2) 00000000000000000000 x87r2 Boş 0.000000000
ST(3) 00000000000000000000 x87r3 Boş 0.000000000
ST(4) 00000000000000000000 x87r4 Boş 0.000000000
ST(5) 00000000000000000000 x87r5 Boş 0.000000000
ST(6) 00000000000000000000 x87r6 Boş 0.000000000
ST(7) 00000000000000000000 x87r7 Boş 0.000000000

x87TagWord FFFF
x87TW_0 3 (Boş)          x87TW_1 3 (Boş)
x87TW_2 3 (Boş)          x87TW_3 3 (Boş)
x87TW_4 3 (Boş)          x87TW_5 3 (Boş)
x87TW_6 3 (Boş)          x87TW_7 3 (Boş)

Varsayılan (stdcall)
1: [esp+4] 000D000C
2: [esp+8] 000F000E
3: [esp+C] 00110010
4: [esp+10] 00130012
5: [esp+14] 00150014
6: [esp+18] 00170016 alaiw.00170016
```

```
Adres     Hex                                               ASCII
0053F1B8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0053F1C8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0053F1D8  00 00 00 00 00 00 00 77 00 00 00 00 00 00 00 00  .......w........
0053F1E8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0053F1F8  00 00 00 00 23 51 07 75 00 00 00 00 00 00 00 00  ....#Q.u........
0053F208  62 D8 08 75 23 5C 0F 75 D4 00 00 00 34 D0 08 75  bØ.u#\.uÔ...4Ð.u
0053F218  34 7B 09 75 3D 10 07 75 50 48 07 75 00 88 01 00  4{.u=..uPH.u....
0053F228  08 F2 53 00 F8 95 30 00 78 F2 53 00 30 74 8D 76  .òS.ø.0.xòS.0t.v
0053F238  74 F2 53 00 FE FF FF FF 47 DF 8A 76 39 40 07 75  tòS.þÿÿÿGß.v9@.u
0053F248  D4 00 00 00 00 00 23 00 00 00 00 00 08 F2 53 00  Ô.....#......òS.
0053F258                                                    ....ýñS.........
0053F268  00 00 00 00 00 00 00 00 00 00 00 00 98 F1 8A 76  .............ñ.v
0053F278  3C F7 53 00 F0 13 07 75 C1 6B 81 8F FE FF FF FF  <÷S.ð..uÁk..þÿÿÿ
```

Figure 17 Dynamic API Resolution

Then, it resolves the **GetModuleFileNameW**, **CreateProcessW**, **GetThreadContext**, **ReadProcessMemory**, **CloseHandle**, **SetThreadContext**, **GetCommandLineW**, and **TerminateProcess** APIs in sequence with the call instructions shown in Figure 17 and stores their addresses in memory.



```
68 04000008                  push 8000004
6A 00                        push 0
6A 00                        push 0
6A 00                        push 0
6A 00                        push 0
8D85 44F8FFFF                lea eax,dword ptr ss:[ebp-7BC]
50                           push eax
FF55 84                      call dword ptr ss:[ebp-7C]
85C0                         test eax,eax
75 05                        jne 140AD5
E9 F8020000                  jmp 140DCD
C785 54FCFFFF 0700010        mov dword ptr ss:[ebp-3AC],10007
8D85 54FCFFFF                lea eax,dword ptr ss:[ebp-3AC]
50                           push eax
FF75 D0                      push dword ptr ss:[ebp-30]
FF55 80                      call dword ptr ss:[ebp-80]
85C0                         test eax,eax
75 05                        jne 140AF5
E9 D8020000                  jmp 140DCD
6A 00                        push 0
6A 04                        push 4
8D45 B4                      lea eax,dword ptr ss:[ebp-4C]
50                           push eax
[ <&CreateProcessW>]=<kernel32.CreateProcessW>
```

```
explorer.exe              2568   0,23                72,41 MB
  wintoolservice.exe      2672                        1,53 MB
  wintools64.exe          2680   0,15   1,74 kB/s    22,75 MB
    VMToolsHookProc.exe   4868                         780 kB
  ProcessHacker.exe       3960   0,40                23,89 MB
  x32dbg.exe              2332   0,50    156 B/s    111,62 MB
    alaiw.exe             4856                      384,71 MB
      alaiw.exe            488                         388 kB
  ida.exe                 3436   0,32               183,48 MB
    alaiw.exe             1600                      383,13 MB
  notepad++.exe           3504                       11,86 MB
  chrome.exe              1592   0,10               135,61 MB
  gozmon.exe              3588                        3,76 MB
    gozmon64.exe          3828   0,05  30,52 kB/s    76,83 MB
  HxD.exe                 4628   1,08                12,64 MB
  die.exe                  516                       22,97 MB
```

Figure 18 File Running in Suspend Mode

Using the **GetModuleFileNameW** API, it gets the path to the file it is located in. Then, it starts alaiw.exe in **suspend mode** as a child process by giving this file path as a parameter to the **CreateProcessW** API.

It writes the data decrypted with the algorithm in Figure 15 into alaiw.exe running in **suspend mode** using the GetThreadContext, ReadProcessMemory, and SetThreadContext APIs. While alaiw.exe is running as a **child process**, it continues to run as the **parent process** after this process.

The memory region where the codes were added to alaiw.exe (Figure 16) was **dumped** and continued to be examined.

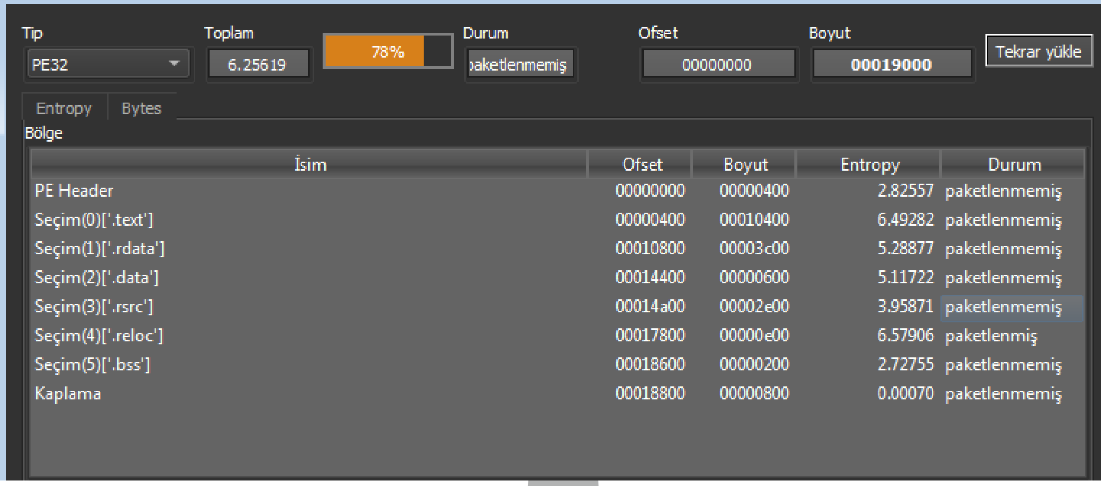
warzone160.exe Analysis

Name	warzone160.exe
MD5	bfa56fb7698757d5316e3cd458008541
SHA256	a4470593f2ebc45b1be6f2d432c90f1a5120dab98427bb6aed8319235b52d4cb
File Type	Portable Executable 32 (x86)

The **dumped** file has been named **"warzone160"**. Information such as its MD5 and SHA256 is listed in the table above.

Static Analysis

When warzone160.exe is examined in the DIE tool, it appears to **be packed**.



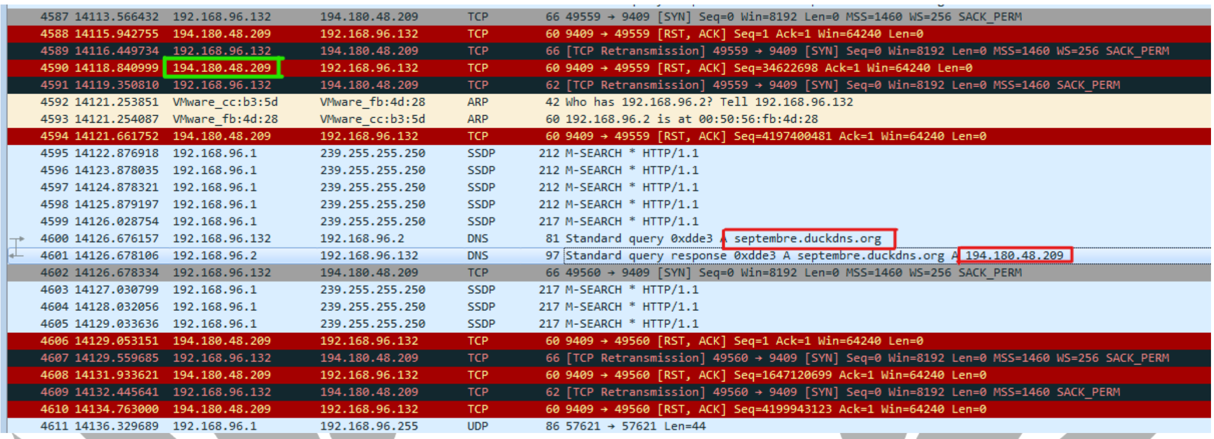
Tip: PE32
Toplam: 6.25619
Durum: ıaketlenmemiş
Ofset: 00000000
Boyut: 00019000

Bölge

İsim	Ofset	Boyut	Entropy	Durum
PE Header	00000000	00000400	2.82557	paketlenmemiş
Seçim(0)['.text']	00000400	00010400	6.49282	paketlenmemiş
Seçim(1)['.rdata']	00010800	00003c00	5.28877	paketlenmemiş
Seçim(2)['.data']	00014400	00000600	5.11722	paketlenmemiş
Seçim(3)['.rsrc']	00014a00	00002e00	3.95871	paketlenmemiş
Seçim(4)['.reloc']	00017800	00000e00	6.57906	paketlenmiş
Seçim(5)['.bss']	00018600	00000200	2.72755	paketlenmemiş
Kaplama	00018800	00000800	0.00070	paketlenmemiş

Figure 19 Examination of warzone160.exe in the DIE Tool

Dynamic Analysis

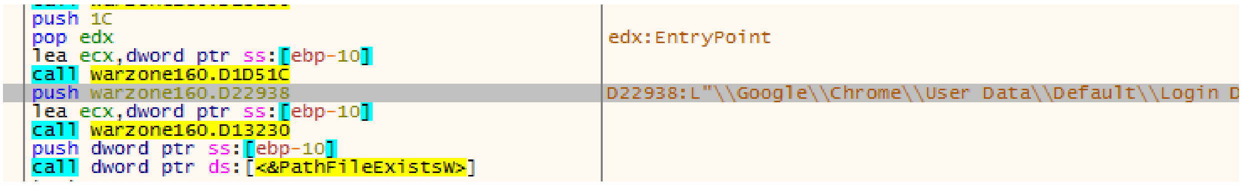


```
4587 14113.566432  192.168.96.132    194.180.48.209    TCP     66 49559 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
4588 14115.942755  194.180.48.209    192.168.96.132    TCP     60 9409 → 49559 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
4589 14116.449734  192.168.96.132    194.180.48.209    TCP     66 [TCP Retransmission] 49559 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
4590 14118.840999  194.180.48.209    192.168.96.132    TCP     60 9409 → 49559 [RST, ACK] Seq=34622698 Ack=1 Win=64240 Len=0
4591 14119.350810  192.168.96.132    194.180.48.209    TCP     62 [TCP Retransmission] 49559 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM
4592 14121.253851  VMware_cc:b3:5d   VMware_fb:4d:28   ARP     42 Who has 192.168.96.2? Tell 192.168.96.132
4593 14121.254087  VMware_fb:4d:28   VMware_cc:b3:5d   ARP     60 192.168.96.2 is at 00:50:56:fb:4d:28
4594 14121.661752  194.180.48.209    192.168.96.132    TCP     60 9409 → 49559 [RST, ACK] Seq=4197400481 Ack=1 Win=64240 Len=0
4595 14122.876918  192.168.96.1      239.255.255.250   SSDP   212 M-SEARCH * HTTP/1.1
4596 14123.878035  192.168.96.1      239.255.255.250   SSDP   212 M-SEARCH * HTTP/1.1
4597 14124.878321  192.168.96.1      239.255.255.250   SSDP   212 M-SEARCH * HTTP/1.1
4598 14125.879197  192.168.96.1      239.255.255.250   SSDP   212 M-SEARCH * HTTP/1.1
4599 14126.028754  192.168.96.1      239.255.255.250   SSDP   217 M-SEARCH * HTTP/1.1
4600 14126.676157  192.168.96.132    192.168.96.2      DNS     81 Standard query 0xdde3 A septembre.duckdns.org
4601 14126.678106  192.168.96.2      192.168.96.132    DNS     97 Standard query response 0xdde3 A septembre.duckdns.org A 194.180.48.209
4602 14126.678334  192.168.96.132    194.180.48.209    TCP     66 49560 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
4603 14127.030799  192.168.96.1      239.255.255.250   SSDP   217 M-SEARCH * HTTP/1.1
4604 14128.032056  192.168.96.1      239.255.255.250   SSDP   217 M-SEARCH * HTTP/1.1
4605 14129.033636  192.168.96.1      239.255.255.250   SSDP   217 M-SEARCH * HTTP/1.1
4606 14129.053151  194.180.48.209    192.168.96.132    TCP     60 9409 → 49560 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
4607 14129.559685  192.168.96.132    194.180.48.209    TCP     66 [TCP Retransmission] 49560 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
4608 14131.933621  194.180.48.209    192.168.96.132    TCP     60 9409 → 49560 [RST, ACK] Seq=1647120699 Ack=1 Win=64240 Len=0
4609 14132.445641  192.168.96.132    194.180.48.209    TCP     62 [TCP Retransmission] 49560 → 9409 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM
4610 14134.763000  194.180.48.209    192.168.96.132    TCP     60 9409 → 49560 [RST, ACK] Seq=4199943123 Ack=1 Win=64240 Len=0
4611 14136.329689  192.168.96.1      192.168.96.255    UDP     86 57621 → 57621 Len=44
```

Figure 20 Data Obtained in Wireshark

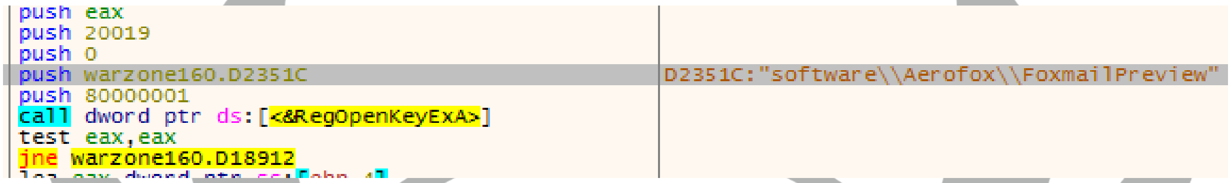
The malware is continuously attempting to communicate with the command-and-control server by trying to connect to the socket **194[.]180[.]48[.]209[:]9409**. This process is repeated constantly because the **connection cannot be established**.

When the malware is running, it has been observed that it attempts to connect to the domain **"septembre[.]duckdns[.]org"**. However, the server is not active, so the connection cannot be established. Therefore, the analysis has been continued **statically**.



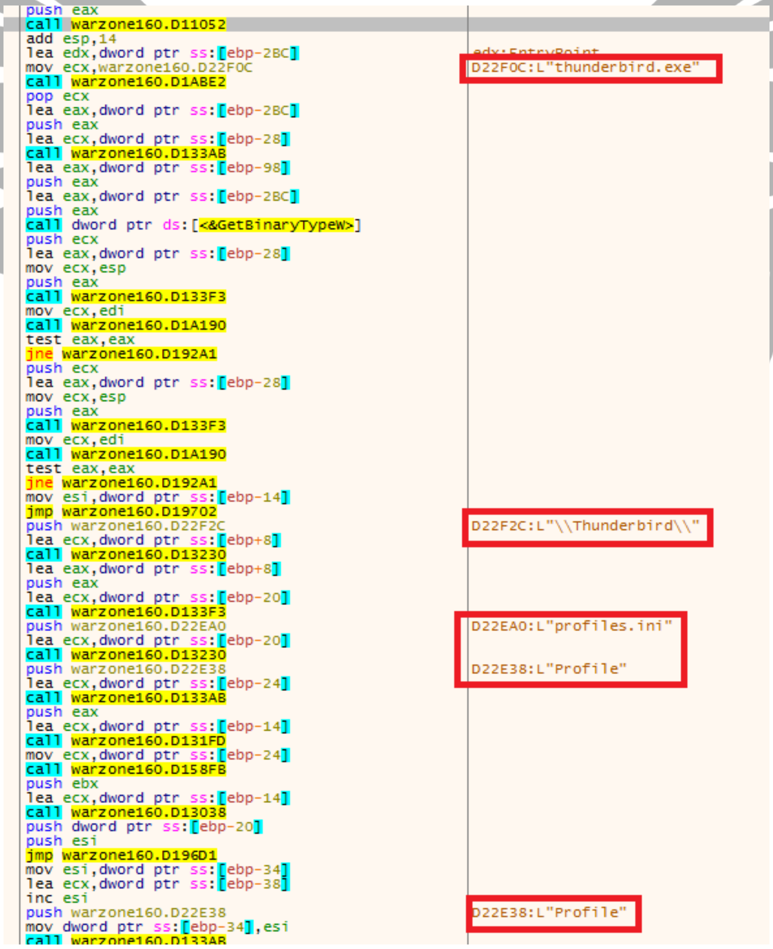
```
push 1C
pop edx                                      edx:EntryPoint
lea ecx,dword ptr ss:[ebp-10]
call warzone160.D1D51C
push warzone160.D22938                       D22938:L"\\Google\\Chrome\\User Data\\Default\\Login D
lea ecx,dword ptr ss:[ebp-10]
call warzone160.D13230
push dword ptr ss:[ebp-10]
call dword ptr ds:[<&PathFileExistsW>]
```

Figure 21 Google Chrome Browser



```
push eax
push 20019
push 0
push warzone160.D2351C                       D2351C:"software\\Aerofox\\FoxmailPreview"
push 80000001
call dword ptr ds:[<&RegOpenKeyExA>]
test eax,eax
jne warzone160.D18912
```

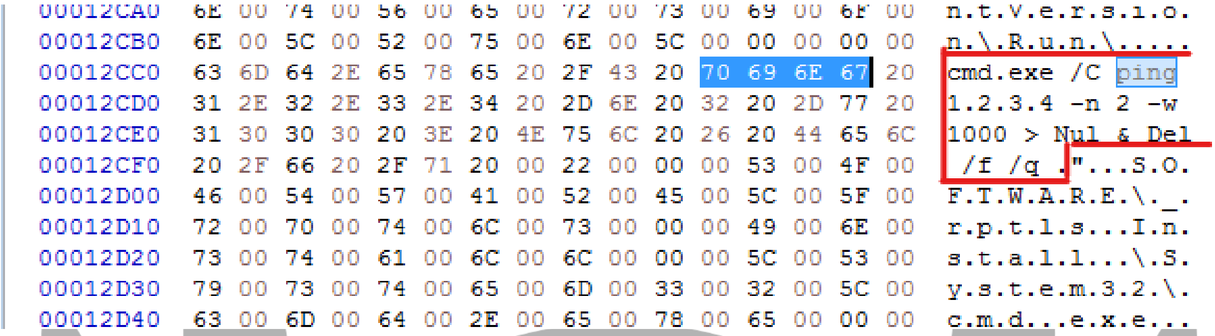
Figure 22 Foxmail Email Service



```
push eax
call warzone160.D11052
add esp,14
lea edx,dword ptr ss:[ebp-2BC]               edx:EntryPoint
mov ecx,warzone160.D22F0C                    D22F0C:L"thunderbird.exe"
call warzone160.D1ABE2
pop ecx
lea eax,dword ptr ss:[ebp-2BC]
push eax
lea ecx,dword ptr ss:[ebp-28]
call warzone160.D133AB
lea eax,dword ptr ss:[ebp-98]
push eax
lea eax,dword ptr ss:[ebp-2BC]
push eax
call dword ptr ds:[<&GetBinaryTypeW>]
push ecx
lea eax,dword ptr ss:[ebp-28]
mov ecx,esp
push eax
call warzone160.D133F3
mov ecx,edi
call warzone160.D1A190
test eax,eax
jne warzone160.D192A1
push ecx
lea eax,dword ptr ss:[ebp-28]
mov ecx,esp
push eax
call warzone160.D133F3
mov ecx,edi
call warzone160.D1A190
test eax,eax
jne warzone160.D192A1
mov esi,dword ptr ss:[ebp-14]
jmp warzone160.D19702
push warzone160.D22F2C                       D22F2C:L"\\Thunderbird\\"
lea ecx,dword ptr ss:[ebp+8]
call warzone160.D13230
lea eax,dword ptr ss:[ebp+8]
push eax
lea ecx,dword ptr ss:[ebp-20]
call warzone160.D133F3
push warzone160.D22EA0                       D22EA0:L"profiles.ini"
lea ecx,dword ptr ss:[ebp-20]
call warzone160.D13230
push warzone160.D22E38                       D22E38:L"Profile"
lea ecx,dword ptr ss:[ebp-24]
call warzone160.D133AB
push eax
lea ecx,dword ptr ss:[ebp-14]
call warzone160.D131FD
mov ecx,dword ptr ss:[ebp-24]
call warzone160.D158FB
push ebx
lea ecx,dword ptr ss:[ebp-14]
call warzone160.D13038
push dword ptr ss:[ebp-20]
push esi
jmp warzone160.D196D1
mov esi,dword ptr ss:[ebp-34]
lea ecx,dword ptr ss:[ebp-38]
inc esi
push warzone160.D22E38                       D22E38:L"Profile"
mov dword ptr ss:[ebp-34],esi
call warzone160.D133AB
```

Figure 23 Thunderbird Email Service

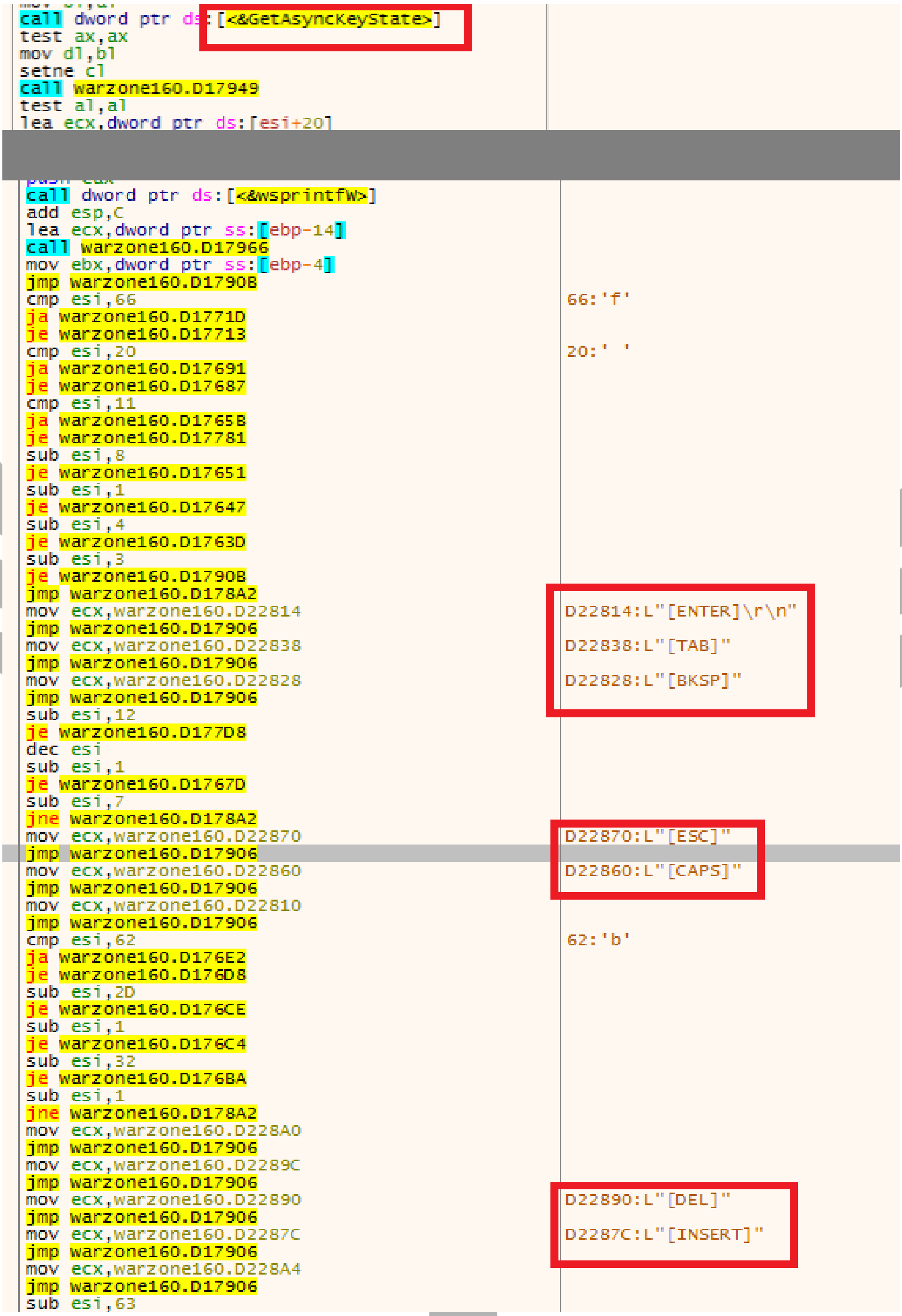
When static analysis and string scanning were performed, as seen in Figures 21, 22, and 23, it was understood that the malware was targeting data such as passwords and **client settings** from some browsers and **email** services.



```
00012CA0  6E 00 74 00 56 00 65 00 72 00 73 00 69 00 6F 00  n.t.V.e.r.s.i.o.
00012CB0  6E 00 5C 00 52 00 75 00 6E 00 5C 00 00 00 00 00  n.\.R.u.n.\.....
00012CC0  63 6D 64 2E 65 78 65 20 2F 43 20 70 69 6E 67 20  cmd.exe /C ping 
00012CD0  31 2E 32 2E 33 2E 34 20 2D 6E 20 32 20 2D 77 20  1.2.3.4 -n 2 -w 
00012CE0  31 30 30 30 20 3E 20 4E 75 6C 20 26 20 44 65 6C  1000 > Nul & Del
00012CF0  20 2F 66 20 2F 71 20 00 22 00 00 00 53 00 4F 00   /f /q ."...S.O.
00012D00  46 00 54 00 57 00 41 00 52 00 45 00 5C 00 5F 00  F.T.W.A.R.E.\._.
00012D10  72 00 70 00 74 00 6C 00 73 00 00 00 49 00 6E 00  r.p.t.l.s...I.n.
00012D20  73 00 74 00 61 00 6C 00 6C 00 00 00 5C 00 53 00  s.t.a.l.l...\.S.
00012D30  79 00 73 00 74 00 65 00 6D 00 33 00 32 00 5C 00  y.s.t.e.m.3.2.\.
00012D40  63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 00  c.m.d...e.x.e...
```

Figure 24 CMD Command

The **Cmd** command shown in Figure 24 aims to complicate the detection of the attack through ping requests sent to an **invalid** IP address. Additionally, the **"Del /f /q"** command is intended to delete the malware without permission in case it is detected.

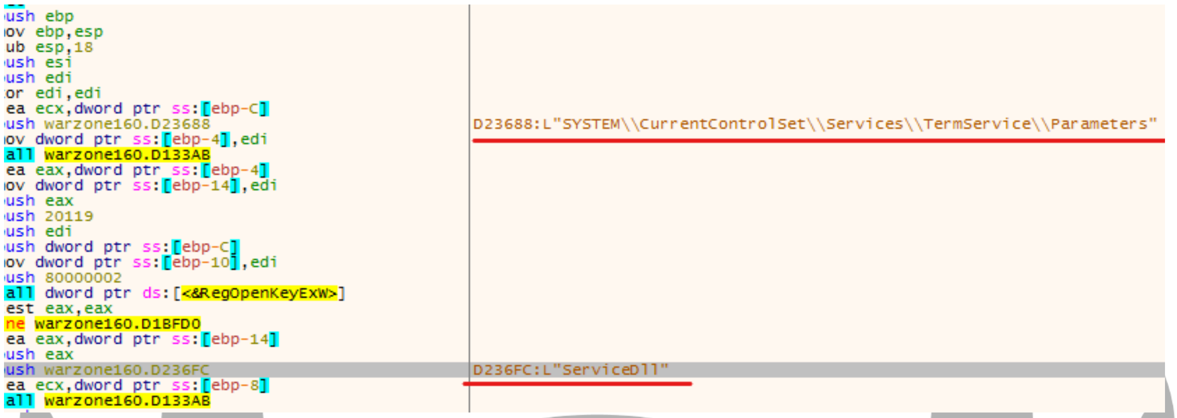


```
call dword ptr ds:[<&GetAsyncKeyState>]
test ax,ax
mov dl,bl
setne cl
call warzone160.D17949
test al,al
lea ecx,dword ptr ds:[esi+20]
```

```
call dword ptr ds:[<&wsprintfW>]
add esp,C
lea ecx,dword ptr ss:[ebp-14]
call warzone160.D17966
mov ebx,dword ptr ss:[ebp-4]
jmp warzone160.D1790B
cmp esi,66                                   66:'f'
ja warzone160.D1771D
je warzone160.D17713
cmp esi,20                                   20:' '
ja warzone160.D17691
je warzone160.D17687
cmp esi,11
ja warzone160.D1765B
je warzone160.D17781
sub esi,8
je warzone160.D17651
sub esi,1
je warzone160.D17647
sub esi,4
je warzone160.D1763D
sub esi,3
je warzone160.D1790B
jmp warzone160.D178A2
mov ecx,warzone160.D22814                    D22814:L"[ENTER]\r\n"
jmp warzone160.D17906
mov ecx,warzone160.D22838                    D22838:L"[TAB]"
jmp warzone160.D17906
mov ecx,warzone160.D22828                    D22828:L"[BKSP]"
jmp warzone160.D17906
sub esi,12
je warzone160.D177D8
dec esi
sub esi,1
je warzone160.D1767D
sub esi,7
jne warzone160.D178A2
mov ecx,warzone160.D22870                    D22870:L"[ESC]"
jmp warzone160.D17906
mov ecx,warzone160.D22860                    D22860:L"[CAPS]"
jmp warzone160.D17906
mov ecx,warzone160.D22810
jmp warzone160.D17906
cmp esi,62                                   62:'b'
ja warzone160.D176E2
je warzone160.D176D8
sub esi,2D
je warzone160.D176CE
sub esi,1
je warzone160.D176C4
sub esi,32
je warzone160.D176BA
sub esi,1
jne warzone160.D178A2
mov ecx,warzone160.D228A0
jmp warzone160.D17906
mov ecx,warzone160.D2289C
jmp warzone160.D17906
mov ecx,warzone160.D22890                    D22890:L"[DEL]"
jmp warzone160.D17906
mov ecx,warzone160.D2287C                    D2287C:L"[INSERT]"
jmp warzone160.D17906
mov ecx,warzone160.D228A4
jmp warzone160.D17906
sub esi,63
```

Figure 25 Keylogger

Figure 25 shows that the malware records special **keystrokes** like ENTER, TAB, BKSP, ESC, CAPS, DEL, and INSERT using the **GetAsyncKeyState** API.



```
ush ebp
ov ebp,esp
ub esp,18
ush esi
ush edi
or edi,edi
ea ecx,dword ptr ss:[ebp-C]
ush warzone160.D23688                  D23688:L"SYSTEM\\CurrentControlSet\\Services\\TermService\\Parameters"
ov dword ptr ss:[ebp-4],edi
all warzone160.D133AB
ea eax,dword ptr ss:[ebp-4]
ov dword ptr ss:[ebp-14],edi
ush eax
ush 20119
ush edi
ush dword ptr ss:[ebp-C]
ov dword ptr ss:[ebp-10],edi
ush 80000002
all dword ptr ds:[<&RegOpenKeyExW>]
est eax,eax
ne warzone160.D1BFD0
ea eax,dword ptr ss:[ebp-14]
ush eax
ush warzone160.D236FC                  D236FC:L"ServiceDll"
ea ecx,dword ptr ss:[ebp-8]
all warzone160.D133AB
```

Figure 26 Remote Access

The malware accesses the ServiceDll record in the **“SYSTEM\CurrentControlSet\Services\TermService\Parameters”** path. This action enables remote access to the device through the **Remote Desktop Protocol (RDP)** and makes it possible to control the device.

YARA Rules

```
import "hash"

rule avemariarat {

        meta:

                author = "Team-5"

   strings:

                $hex_1 = { 55 58 54 48 45 4D 45 00 55 53 45 52 45 4E 56 00 53 45 54
55 50 41 50 49 00 41 50 50 48 45 4C 50 00 50 52 4F 50 53 59 53 00 44 57 4D 41 50
49 00 43 52 59 50 54 42 41 53 45 00 4F 4C 45 41 43 43 00 43 4C 42 43 41 54 51 00
4E 54 4D 41 52 54 41 }

                $hex_2 = { 50 33 C0 68 50 67 42 00 50 50 68 00 00 00 04 50 50 50 FF
75 08 50 FF 15 ?? ?? ?? ?? }

                $str1 = "http://nsis.sf.net/NSIS_Error" wide

                $str2 = "\\Microsoft\\Internet Explorer\\Quick Launch" wide

                $api1 = "DialogBoxParamW" ascii

                $api2 = "RegSetValueExW" ascii

                $api3 = "CreateProcessW" ascii

                $api4 = "ExitProcess" ascii

                $api5 = "WriteFile" ascii

                $api6 = "FindNextFileW" ascii

   condition:

      hash.md5 (0, filesize) == "d802bc50f7321efb13358d27280910ca" or (all of ($str*)
and (5 of ($api*))) or (all of ($hex_*)) }
```

```
import "hash"

rule alaiw_d {

        meta:

                author = "Team-5"

                description = "AveMariaRAT"

                weight = “10”

   strings:

        $algorithm1 = { C1 E0 05 0F B6 4D EF C1 F9 03 0B C8 33 CA 8B 55 E8 88 8A ?? ?? ?? ?? } //Shellcode decryption algorithm

        $str1 = "vtwkwntewuzvb"

        $str2 = "find.exe"

        $str3 = "-w %ws -d C -f %s"

        $str4 = "\\System32\\cmd.exe"

        $str5 = "SELECT * FROM logins"

        $str6 = "Accounts\\Account.rec0"

        $str7 = "cmd.exe /C ping 1.2.3.4 -n 2 -w 1000 > Nul & Del /f /q"

        $str8 = "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\SpecialAccounts\\UserList"
        $w1 = "http://5.206.225.104/dll/msvcp140.dll" wide

        $w2 = "http://5.206.225.104/dll/softokn3.dll" wide

        $w3 = "http://5.206.225.104/dll/mozglue.dll" wide

        $w4 = "http://5.206.225.104/dll/vcruntime140.dll" wide

        $w5 = "http://5.206.225.104/dll/freebl3.dll" wide

        $w6 = "http://5.206.225.104/dll/nss3.dll" wide

        $w7                                                                  =
"C:\\Users\\louis\\Documents\\workspace\\MortyCrypter\\MsgBox.exe"
wide

        $w8 = "\\Google\\Chrome\\User Data\\Default\\Login Data" wide

        $w9 = "profiles.ini" wide

   condition:

     hash.md5(0,filesize)  ==  "fa0be3eb24b13d060a0ae4e25c22ef1c"  or
(((5 of $str*) or (7 of $w*)) or ($algorithm1 and (2 of $w*)))

}
```

```
rule warzone {
        meta:
                author = "Team-5"
   strings:
        $str1 = "find.exe"
        $str2 = "-w %ws -d C -f %s"
        $str3 = "\\System32\\cmd.exe"
        $str4 = "SELECT * FROM logins"
        $str5 = "Accounts\\Account.rec0"
        $str6 = "cmd.exe /C ping 1.2.3.4 -n 2 -w 1000 > Nul & Del /f /q"
        $str7 = "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\\SpecialAccounts\\UserList"

        $w1 = "http://5.206.225.104/dll/msvcp140.dll" wide
        $w2 = "http://5.206.225.104/dll/softokn3.dll" wide
        $w3 = "http://5.206.225.104/dll/mozglue.dll" wide
        $w4 = "http://5.206.225.104/dll/vcruntime140.dll" wide
        $w5 = "http://5.206.225.104/dll/freebl3.dll" wide
        $w6 = "http://5.206.225.104/dll/nss3.dll" wide

        $w7                                                                    =
"C:\\Users\\louis\\Documents\\workspace\\MortyCrypter\\MsgBox.exe"
wide

        $w8 = "\\Google\\Chrome\\User Data\\Default\\Login Data" wide

        $w9 = "profiles.ini" wide

        $e1 = "hostname"

        $e2 = "encryptedUsername"

        $e3 = "encryptedPassword"

        $v1 = "vaultcli.dll"

        $v2 = "VaultOpenVault"

        $v3 = "VaultCloseVault"

        $v4 = "VaultEnumerateItems"

        $v5 = "VaultGetItem"

        $v6 = "VaultFree"

   condition:

      ((5 of ($str*)) or (4 of ($w*))) or ((all of ($e*)) and (all of ($v*)))

}
```

MITRE ATTACK TABLE

Reconnaissance	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	C&C	Exfiltration
Gather Victim Network Information (T1590)	Command and Scripting Interpreter (T1059)	Create Account (T1136)	Process Injection (T1055)	Deobfuscate/Decode Files or Information (T1140)	OS Credential Dumping (T1003.008)	Application Layer Protocol (T1071.004)	Exfiltration Over C2 Channel (T1041)
Gather Victim Host Information (T1592)	Shared Modules (T1129)	Boot or Logon Autostart Execution (T1547)	Create or Modify System Process (T1543.003)	Masquerading (T1036)			
	Native API (T1106)						

Recommendations

1. Do not download files from unknown sources.
2. Do not click on links from unknown sources.
3. Be cautious when using unknown external devices.
4. Be technology literate.
5. Keep the operating system updated.
6. Do not open untrusted emails.

PREPARED BY

Barış TURAL	https://www.linkedin.com/in/baristural/
Betül ŞAHİN	https://www.linkedin.com/in/betulsahinn/
Zeynep ÖZDEMİR	https://www.linkedin.com/in/zeynep-ozdemir/