1.Basic Static Malware Analysis~Temel Statik Kötü Amaçlı Yazılım Analizi

Öncellikle bu yazı serimde malware datascience kitabından yararlanıp onların sağlamış olduğu kodların üzerinden anlatmaya çalışacağım.Umarım faydalı olur.Teşekkürler..

Statik analiz: Bir programın dosyasının disassembled kodunun , grafik görüntülerinin , yazdırlabilir dizelerin strings ve diğer disk üzerindeki kaynakların analiz edilmesiyle gerçekleşir. Programı çalıştırmadan recerse engineering tersine mühendisliği ifade eder. Statik analizin eksikleri olsa da bize bir zararlıyı anlamızda yardımcı bilgiler verir. Statik analiz tekniklerini açıklayıp bunlara örnek verelim.

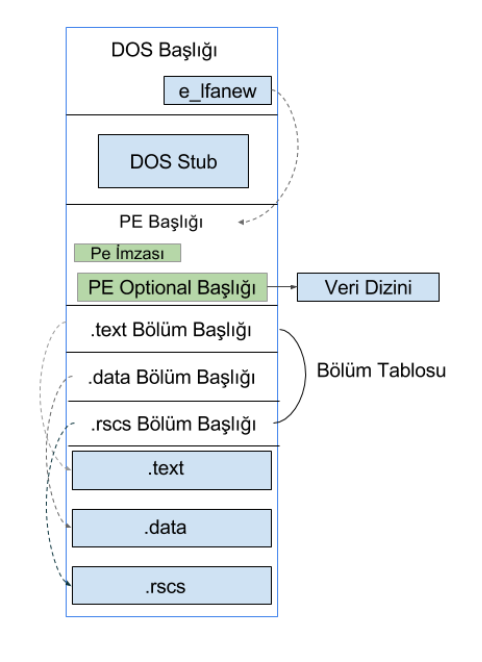
İlk olarak çoğu windows programı tarafından kullanılan portable executable(PE) dosya  formatını açıklayacak ve sonra pythonda olan pefile kütüphanesi ile  nasıl inceleyebileceğimize bakalım. Sonra içe aktarma analizi describe techniques import analysis, grafik görüntü analizi , graphical image analysis ve dizi analizi strings analysis tekniklerine bakacak.Son olarak bölüm sonunda kötü amaçlı yazılımların kötü amaçlı yazılım analistleri için zorlaştırıcı yllların ne olduğu bunları azaltabileceğini gösterecek. Bu bölümdeki kodları kitabın bize sağlamış olduğu kod dizinlerinden ch1/ dizninin altından bulabilirsiniz. Analiz için bize kitabın sağlamış olduğu Internet Relay Chat (IRC) botu olan ircbot.exe 'yi kullanıyor. Bu program hedef bilgisyara yerleşip hedefi ele geçirdikten sonra saldırganların irc ile hedef sistemi kontrol eder. Statik analiz ile bunu incelemeye başlıyalım.

\*The Microsoft Windows Portable Executable Format: Mikrosoft Windows Taşınabilir Yürütülebilir Biçim:

Statik malware analizi için , .exe,.dllve .sys dosyaları gibi modern Windows program dosyalarının yapısını tanımlayan ve verileri sakalam yöntemlerini tanımlayan Windows PE biçimine bakalım.PE dosyaları x86 talimatlarını , görüntüler ve metin gibi verileri ve bir programın çalışması için ihtiyaç duyduğu meta verileri içerir.

Pe dosya biçimi windows!a bir programın belleğe nasıl yüklenmesi gerektiğini ve nereye yükleneceğini açıklar. Ayrıca windows'un program kodunun neresinde bir programın yürütülmesini , başlatmaısnı ve hangi dinamik olarak bağlantılı kod kitaplıklarının belleğe yüklenmesi gerektiğini de söyler. ÇAlışan bir programın yürütülmesi sırasında kullanabileceği kaynaklarıü gui iletişim kutularını veya konsol çıktısında bulunanlar gibi karakter dizilerinin yanı sıra görüntüleri veya videoları içerebilir.

Windows kodun güvenilir bir jaynaktan geldiğinden emin olmak için dijital kod imzaları gibi güvenlik verilerini kullanır.



kaynak:https://ring0.info/posts/pe-dosya-formatina-dalis

8-> .reloc section (memory translations )

7-> .rsrc section (strings, images, ..)

6->  .idata section (imported libraries)

5-> .text section (program code)

4-> Section headers Bölüm başlıkları

3-> Optional headers İsteğe bağlı başlıklar

2-> PE header

1-> DOS header

8'e doğru offaset değeri artıyor.

PE biçimi , işletim sistemine programın belleğe nasıl yükleneceğini anlatan bir dizi başlık içerir.  Ayrıca, gerçek program verilerini içeren bir dizi bölüm içerir. Windows bölümleri belleğe, bellek uzaklıkları diskte göründükleri yere karşılık gelecek şekilde yükler.1980'lerden kalma Microsoft DOS işletim sisteminin bir kalıntısı olan ve yalnızca uyumluluk nedenleriyle sunulan DOS başlığıyla ilgili tartışmayı atlayacağız.

\*The PE Header:

Bir programın binary kodu, görüntüleri, sıkıştırılmış  veriler ve diğer program öznitelikleri gibi genel özniteliklerini tanımlayan PE başlığıdır. Ayrıca bize bir programın 32 bitlik mi 64 bitlik sistemler için mi tasarlandığını söyler. Anlayacağınız PE bize temel ama yararlı bilgiler verir. Örneğin üstbilgi header, malwarein ne zaman dosyayı derlediğini içeren zaman damgası alanı içerir. Bunu tabi sahte bir değerle de değiştirebilir yazan kişi.🙂

\*The Optional Header:

Günümüzün PE çalıştırılabilir programlarında aslında her yerde bulunur. Programın yüklendikten sonra çalıştırdığı ilk talimatı ifade eden PE dosyasındaki programın giriş noktasının konumunu tanımlar.ayrıca PE dosyasını windows alt sistemini , program hedefleirni (Windows GUI veya Windows komut satırı gibi) ve programla ilgili diğer üst düzey ayrıntıları yüklerken Windows'un vellğe yüklediği verileirn boyutunu da tanımlar. Bu başlıktaki bilgiler tersine mühendislik için programın başlangıç noktasını verdiğinden dolayı değerlidir.

\*Section Header:

Bir PE dosyası içinde bulunan veri bölümlerini açıkalr. Programın belleğe nasıl yüklenmesi gerektiği ile ilgili talimatları içeren veri yığınıdır. Bu kısım ayrıca windowsa bölümlere hangi izinleri vermesi gerektiğini örneğin program yürütülürken program tarafından okunabilir ,yazılabilir veya çalıştırılabilir olması gerektiğini söyler.

\*The .text Section:

her pe programı bölüm başlığında yürütülebilir olarak işaretlenmiş en az bir x86 kodu bölümü içerir. bu bölümler hemen hemen her zaman .text olarak adlandırılır.

\*the .idata Section:

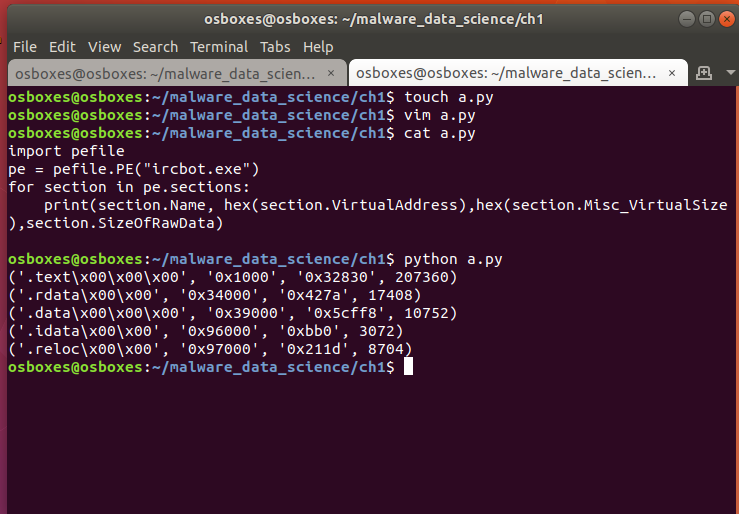
içe aktarma import olarak adlandırılan idata bölümü dinamik olarak bağlantılı kitaplıkları ve işlevlerini listeleyen içe aktarma tablosu (IAT) içerir. IAT , analiz için bir PE ikili programına ilk bakılırken incelenecek en önemli PE yapıları arasındadır. Çünkü bir programın yaptığı kitaplık çağrılarını ortaya çıkarır.

\*The Data Sections:

PE dosyasındaki data bölümleri , fare ilecinin görüntüleri , ses ve bir program tarafından kullanılan diğer ortamlar gibi öğeleri depolayan .rsrc , .data ve .rdata gibi bölümleri içerebilir. .rsrc kısmı bir programın texti string olarak işlemek için kullandığı yazdırılabilir strings içerir. Buradan yazdırılabilir karakter dizelerini ,grafik görüntüleri ve diğer varlıkları inceleyerek dosyanın işlevselliği hakkında önemli iğuçları elde edilebilir. malwarelerin kaynaklar bölümlerinden graik görüntülerini icotuils ile çıkarırız.  .reloc kısmı : bir PE ikili kodu konumundan bağımsız değildir yani amaçlanan bellek konumundan yeni bir bellek konumuna taşınırsa doğru şekilde çalışmayacaktır. reloc kosun kırılmadan taşınmasına izin vererek bunun üstesinden gelir. windows işletim sistemine kod taşınmışsa kodun hala düzgün çalışması için PE doasyaısnın kodundaki bellek adreslerini çevirmesini söyler.

\*Şimdi pefile kullanarak PE den bilgiler nasıl çıkarırız ona bakalım:

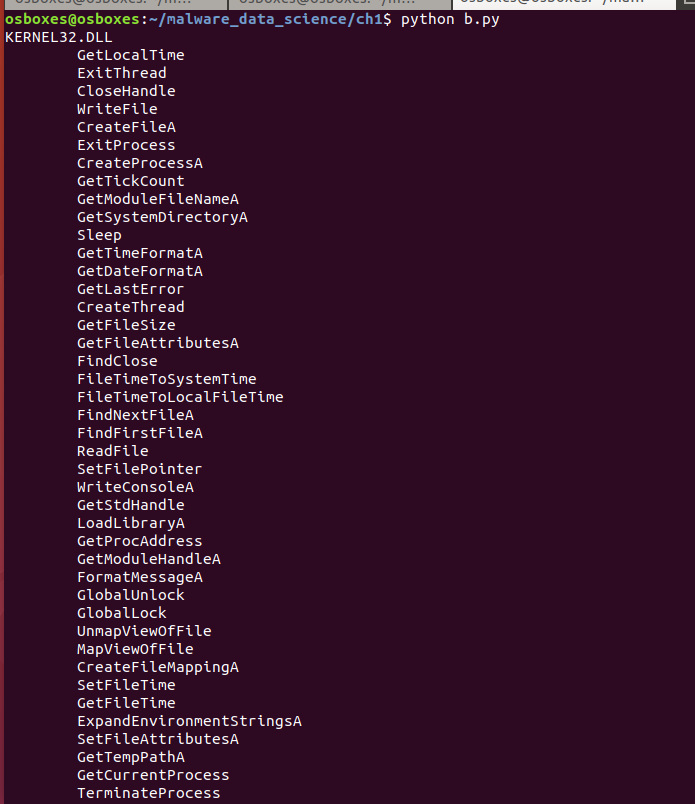
ircbot.exe dosyasını inceleyeceğiz. Bu yazıyı çevirdiğim kitabın sağlamış olduğu sanal makinenin ~/malware\_data\_science/ch1/data dizininde bulabilirsiniz. İşletim sisteminde pefile kurulu ama  kendiniz "pip install pefile" ile indirebilirsiniz.

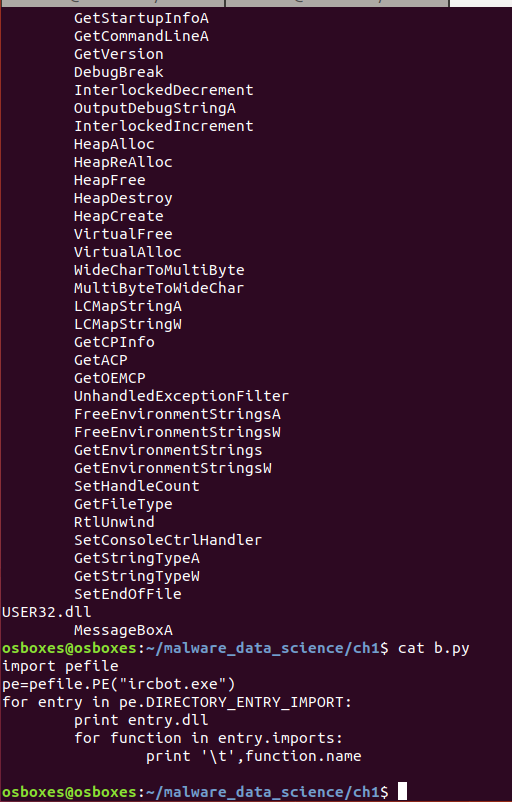


PE dosyalarını ,niteliklerini inceleyebilmemiz için ayrıştırır. PE yi çağırıığ örnek uygulamamızı yükler ve ayrıştırır.PE doasyasının 5 farklı bölümünden veri aldık : .text , ..rdata, .data, .idata ve .reloc. Çıktı 5 tane tuple verir. Her satırda ilk giriş PE bölümünü tanımlar. kalan alanlar bize her bölümün belleğe yüklendikten sonra bellek kullanımının ne olacağını ve yüklendikten sonra bellekte nerede bulunacağını söyler.

örneğin 0x1000 bölümün yükleneceği temel sanal bellek adresi. 0x32830 yüklendikten sonra bölüm için gerekn bellek miktarını verir. üçüncü alan 207360 bu bellek parçası içinde alacağı veri miktarını verir.

Bir programın bölümlerini ayrıştırmak için pefile kullanmaya ek olarak, bir ikilinin yükleyeceği DLL'leri ve bu DLL'ler içinde isteyeceği işlev çağrılarını listelemek için de kullanabiliriz. Bunu, bir PE dosyasının IAT'sini boşaltarak yapabiliriz. Şimdi ircbot.exe nin IAT'yi dökmek için pefile'ın nasıl kullanıcağına bakalım:



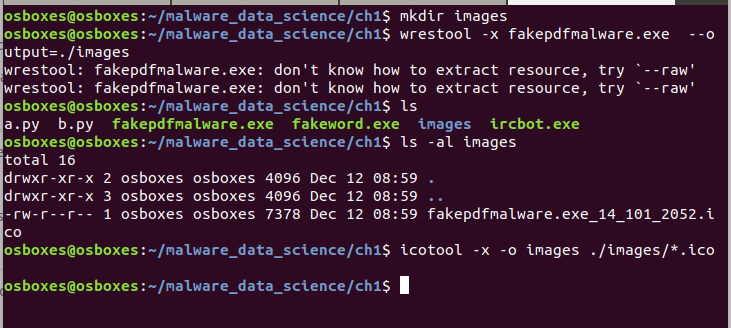


Bu, kötü amaçlı yazılım hakkında oldukça temel bilgiler olsa da, kötü amaçlı yazılımın davranışını daha ayrıntılı olarak anlamak için bir başlangıçtır.

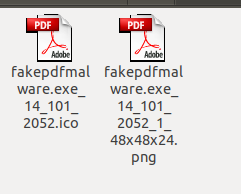
\*Examining Malware Images:Kötü Amaçlı Yazılım Görüntüleri İnceleme:

Bir zararlıdan resimlerini çekip onun işlevi ile ilgili veya yapan grup hakkında bilgi gibi çeştili bilgileri elde edebilmek adına önemlidir.

Şimdi örnek olarak kitabın sağlamış olduğu fakepdfmalware.exe ikili dosyasından linuz komut satırı aracı olan wrestool ile görüntüleri çıkaralım.

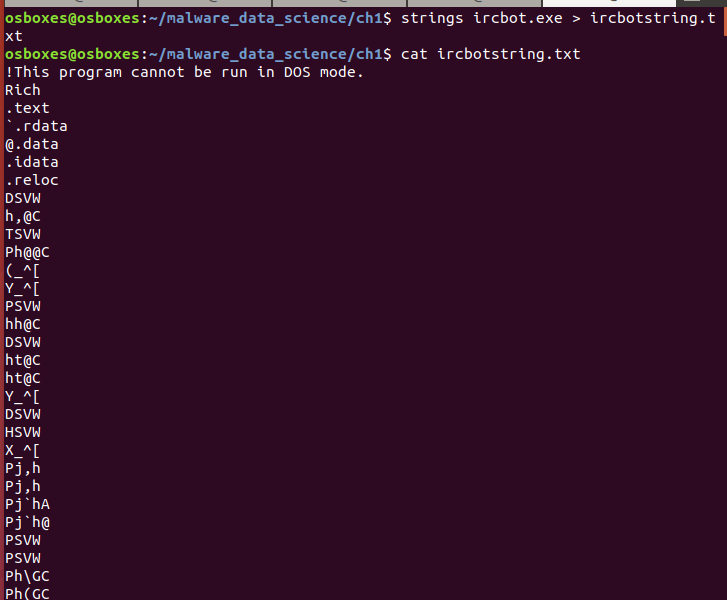


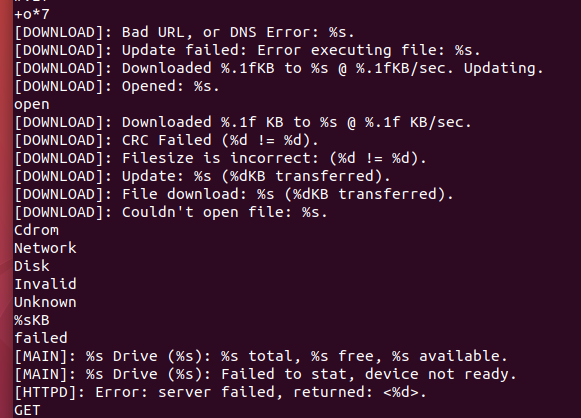
.ico uzantılı dosayı .png ye icotool ile çevirelim.



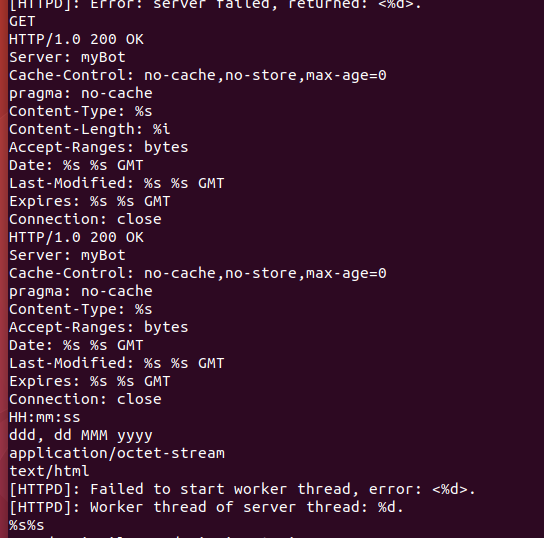
\*Şimdi malware hakkında bilgi çıkarma işlemlerinden bir diğeri olan string inceleme işlemlerine bakalım.Dizeler Strings , bir program ikili dosyası içindeki yazdırılabilir karakter dizileridir.Bu dizeler genellikle web sayfalarını ve dosyaları, IP adreslerini ve kötü amaçlı yazılımın hangi adreslere bağlandığını size söyleyen ana bilgisayar adlarını ve benzerlerini indiren HTTP ve FTP komutları gibi şeyler içerir.Bazen stringlerde ki kullanılan  dil bile bize ülke hakkında bilgi verir. . Dizeler ayrıca bir ikili dosya hakkında daha fazla teknik bilgi ortaya çıkarabilir. Örneğin, onu oluşturmak için kullanılan derleyici, ikilinin yazıldığı programlama dili, gömülü komut dosyaları veya HTML vb. Hakkında bilgi bulabilirsiniz. . Kötü amaçlı yazılım yazarları tüm bu izleri gizleyebilse, şifreleyebilse ve sıkıştırabilse de, gelişmiş kötü amaçlı yazılım yazarları bile çoğu zaman en azından bazı izleri açığa çıkararak kötü amaçlı yazılımları analiz ederken dize dökümlerini incelemeyi özellikle önemli hale getirir.

strings komutu minimum 4 bayt uzunluğa sahip tüm yazdırılabilir dizeleri bulur, ancak command page sayfasında listelendiği gibi farklı bir minimum uzunluk ayarlayabilir ve çeşitli diğer parametreleri değiştirebilirsiniz.





Bu satırlar, ircbot.exe'nin bir saldırgan tarafından belirtilen dosyaları hedef makineye indirmeyi deneyeceğini gösterir.  Başka birini analiz etmeyi deneyelim. Liste 1-8'de gösterilen dize dökümü, ircbot.exe'nin saldırgandan gelen bağlantılar için hedef makinede dinleyen bir web sunucusu görevi görebileceğini belirtir.



\*Summary : özet

Bu bölümde, bir kötü amaçlı yazılım programını çalıştırmadan incelemeyi içeren statik kötü amaçlı yazılım analizine ilişkin üst düzey bir genel bakış elde edeceksiniz. Windows .exe ve .dll dosyalarını tanımlayan PE dosya biçimini öğrendiniz ve gerçek bir kötü amaçlı yazılım olan ircbot.exe ikili dosyasını incelemek için Python kitaplık dosyasını nasıl kullanacağınızı öğrendiniz. Kötü amaçlı yazılım örneklerinden daha fazla bilgi elde etmek için görüntü analizi ve dizi analizi gibi statik analiz tekniklerini de kullandınız. Bölüm 2, kötü amaçlı yazılımdan kurtarılabilen derleme kodunu analiz etmeye odaklanarak statik kötü amaçlı yazılım analizi tartışmamıza devam ediyor.