12.Becoming A Data Scientist: Veri Bilimcisi Olmak:

Bu kitabı bitirmek için bir adım geriye gidelim ve bir kötü amaçlı yazılım veri bilimcisi veya genel olarak bir güvenlik veri bilimcisi olarak nasıl bir hayat ve kariyer yapabileceğinizi tartışalım. Bu teknik olmayan bir bölüm olmasına rağmen, daha önemli değilse de bu kitaptaki teknik bölümler kadar önemlidir. Bunun nedeni, başarılı bir güvenlik veri bilimcisi olmanın, konuyu anlamaktan çok daha fazlasını gerektirmesidir.

Bu bölümde, biz yazarlar, profesyonel güvenlik veri bilimcileri olmak için kendi kariyer yollarımızı paylaşıyoruz. Bir güvenlik veri bilimcisi olarak günlük yaşamın neye benzediğine ve etkili bir veri bilimcisi olmanın neye mal olduğuna bir göz atacaksınız. Ayrıca, veri bilimi sorunlarına nasıl yaklaşılacağı ve kaçınılmaz zorluklar karşısında nasıl dayanıklı kalacağımızla ilgili ipuçları da paylaşıyoruz.

\*Paths to Becoming a Security Data Scientist:

c and assembly

şirkette veri sızıdıran kişileri avlama

Kötü amaçlı alan adlarının tespiti

• Kötü amaçlı URL'leri algılama

• Kötü amaçlı e-posta eklerini algılama

• Anormallikleri tespit etmek için ağ trafiğini görselleştirme

• Kimlik avı e-postalarını tespit etmek için e-posta gönderen / alıcı kalıplarını görselleştirme

13.Bölüm:

Chapter 1: Basic Static Malware Analysis:

Bölüm 1'de ircbot.exe adlı kötü amaçlı yazılım ikili dosyasının temel statik analizini ele aldığımızı hatırlayın. Bu kötü amaçlı yazılım bir aşındır, yani kullanıcıların sistemlerinde gizlenir ve bir saldırganın komutlarını bekler, böylece saldırganın kurbanın bilgisayarından özel veriler toplamasına veya kurbanın sabit diskini silmek gibi kötü niyetli amaçlara ulaşmasına izin verir. Bu ikili, bu kitapla birlikte ch1 / ircbot.exe adresindeki verilerde mevcuttur. Ayrıca bu bölümde bir fakepdfmalware.exe örneği de kullanıyoruz (ch1 / fakepdfmalware.exe'de bulunur). Bu, kullanıcıları gerçekten kötü amaçlı programı çalıştırıp sistemlerine bulaştırırken bir PDF belgesi açtıklarını düşünmeleri için kandırmak için bir Adobe Acrobat / PDF masaüstü simgesine sahip bir kötü amaçlı yazılım programıdır.

Chapter2: Beyond Basiv Static Analysis : x86 Disassembly:

Bu bölümde kötü amaçlı yazılım ters mühendisliğinde daha derin bir konuyu keşfediyoruz: x86 demontajını analiz etme. Bu bölümdeki Bölüm 1'deki ircbot.exe örneğini yeniden kullanıyoruz.

Chapter 3: A Brief Introduction to Dynamic Analysis:

Bölüm 3'teki dinamik kötü amaçlı yazılım analizi tartışmamız için, bu kitapla birlikte gelen verilerde ch3 / d676d9dfab6a424 2258362b8ff579cfe6e5e6db3f0cdd e0069ace50f80af1c5 yolunda depolanan bir fidye yazılımı örneğini deniyoruz. Dosya adı, dosyanın SHA256 şifreleme karmasına karşılık gelir. VirusTotal.com’un kötü amaçlı yazılım veritabanını fidye yazılımı örnekleri için arayarak elde ettiğimiz bu fidye yazılımı hakkında özellikle özel bir şey yok

Chapter 4: Identifying Attack Campaigns Using Malware Networks:

Bölüm 4, kötü amaçlı yazılımlara ağ analizi ve görselleştirme uygulamasını tanıtır. Bu teknikleri göstermek için, yüksek profilli saldırılarda kullanılan bir dizi yüksek kaliteli kötü amaçlı yazılım örneği kullanıyoruz ve analizimizi, güvenlik topluluğu tarafından Advanced Persistent Threat 1 (Gelişmiş Kalıcı Tehdit 1) olarak bilinen Çin ordusu içindeki bir grup tarafından üretilen bir grup kötü amaçlı yazılım örneğine odaklıyoruz. veya kısaca APT1) Bu örnekler ve bunları oluşturan APT1 grubu, siber güvenlik firması Mandiant tarafından keşfedildi ve halka açıklandı. "APT1: Çin'in Siber Casus Birimlerinden Birini Açığa Çıkarma" başlıklı raporunda (buradan alıntılanmıştır) (https: // www.fireeye .com / content / dam / fireeye-www / services / pdfs / mandiant-apt1 report.pdf), Mandiant şunları buldu:

• 2006'dan beri Mandiant, APT1'in 20 büyük endüstriye yayılmış 141 şirketten ödün verdiğini gözlemledi.

• APT1, yıllar içinde geliştirilmiş ve büyük hacimli değerli fikri mülkiyeti çalmak üzere tasarlanmış, iyi tanımlanmış bir saldırı metodolojisine sahiptir.

• APT1 erişim sağladıktan sonra, mağdurun ağını birkaç ay veya yıl boyunca periyodik olarak yeniden ziyaret eder ve teknoloji planları, tescilli üretim süreçleri, test sonuçları, iş planları, fiyatlandırma belgeleri, ortaklık anlaşmaları ve e-postalar ve iletişim dahil olmak üzere geniş fikri mülkiyet kategorilerini çalarlar. kurban kuruluşların liderliğinden listeler.

• APT1, e-posta çalmak için tasarlanmış iki yardımcı program dahil olmak üzere, diğer gruplar tarafından henüz kullanılmadığını gözlemlemediğimiz bazı araçları ve teknikleri kullanır: GETMAIL ve MAPIGET.

• APT1 kurban ağlarına ortalama 356 gün boyunca erişimi sürdürdü.

• Bir kurbanın ağına erişimini sürdürdüğü en uzun süre APT1 1.764 gün veya dört yıl on aydı.

• Diğer büyük ölçekli fikri mülkiyet hırsızlıklarının yanı sıra, APT1'in on aylık bir süre içinde tek bir kuruluştan 6,5 TB sıkıştırılmış veri çaldığını gözlemledik.

• 2011'in ilk ayında APT1, 10 farklı sektörde faaliyet gösteren en az 17 yeni kurbanı başarıyla tehlikeye attı.

Raporun bu kısmının da gösterdiği gibi, APT1 örnekleri yüksek miktarlarda, ulus devlet düzeyinde casusluk için kullanıldı. Bu örnekler, bu kitaba eşlik eden verilerde ch4 / data / APT1\_MALWARE\_FAMILIES adresinde mevcuttur.

\*Chapter 5: Shared Code Analysis:

Bölüm 5, Bölüm 4'te kullanılan APT 1 örneklerini yeniden kullanır. Kolaylık olması açısından, bu örnekler ayrıca Bölüm 5 dizininde, ch5 / data / APT1 KÖTÜ AMAÇLI AİLELER'de bulunur.

\*Chapter 6: Understanding Machine Learning–Based Malware Detectors and

Chapter 7: Evaluating Malware Detection Systems

Bu kavramsal bölümler herhangi bir örnek veri gerektirmez.

\*Chapter 8: Building Machine Learning Detectors:

Bölüm 8, makine öğrenimi tabanlı kötü amaçlı yazılım algılayıcıları oluşturmayı araştırır ve kendi makine öğrenimi algılama sisteminizi eğitmek için örnek veri kümesi olarak 1.419 örnek ikili dosya kullanır. Bu ikili dosyalar, iyi huylu örnekler için ch8 / data / benignware'de ve kötü amaçlı yazılım örnekleri için ch8 / data / malware'de bulunur. Veri kümesi 991 iyi huylu yazılım örneği ve 428 kötü amaçlı yazılım örneği içeriyor ve bu verileri VirusTotal.com'dan aldık. Bu örnekler, kötü amaçlı yazılım durumunda, 2017'de internette gözlemlenen kötü amaçlı yazılım türünü ve iyi huylu yazılım durumunda, kullanıcıların 2017'de VirusTotal.com'a yüklediği ikili dosya türlerini temsil ediyor.

\*Chapter 9: Visualizing Malware Trends:

Bölüm 9, veri görselleştirmeyi araştırır ve ch9 / code / malware\_data.csv dosyasındaki örnek verileri kullanır. Dosyadaki 37.511 veri satırından her satır, tek bir kötü amaçlı yazılım dosyasının kaydını, ilk görüldüğünde, onu kaç antivirüs ürününün tespit ettiğini ve ne tür kötü amaçlı yazılım olduğunu (örneğin, Truva atı, fidye yazılımı ve yakında). Bu veriler VirusTotal.com'dan toplanmıştır.

\*Chapter 10: Deep Learning Basics:

Bu bölüm derin sinir ağlarını tanıtır ve herhangi bir örnek veri kullanmaz.

\*Chapter 11: Building a Neural Network Malware Detector with Keras:

Bu bölüm, kötü niyetli ve zararsız HTML dosyalarını algılamak için bir sinir ağı kötü amaçlı yazılım algılayıcısı oluşturmayı anlatır. İyi huylu HTML dosyaları yasal web sayfalarından ve kötü niyetli web sayfaları, web tarayıcıları aracılığıyla kurbanlara bulaşmaya çalışan web sitelerinden gelmektedir. Bu veri kümelerinin ikisini de VirusTotal.com'dan, milyonlarca örnek kötü niyetli ve zararlı HTML sayfalarına erişim sağlayan ücretli bir abonelik kullanarak aldık. Tüm veriler ch11 / data / html kök dizininde saklanır. İyi huylu yazılım ch11 / data / html / benign\_files dizininde ve kötü amaçlı yazılım ise ch11 / data / html / malware\_files konumunda saklanır. Ek olarak, bu dizinlerin her birinde alt dizinler eğitim ve doğrulama bulunur. Eğitim dizinleri, bölümde sinir ağını eğittiğimiz dosyaları içerir ve doğrulama dizinleri, doğruluğunu değerlendirmek için sinir ağını test ettiğimiz dosyaları içerir.

\*Chapter 12: Becoming a Data Scientist:

Bölüm 12, nasıl bir veri bilimcisi olunacağını tartışır ve herhangi bir örnek veri kullanmaz.

Tool Implementation Guide: Araç uygulama Kılavuzu

Bu kitaptaki tüm kodlar, kitaptaki fikirleri göstermeyi amaçlayan ve gerçek dünyada kullanılmayan örnek kod olsa da, sağladığımız kodun bir kısmı kendi kötü amaçlı yazılım analizi çalışmanızda bir araç olarak kullanılabilir. , özellikle kendi amaçlarınız için genişletmeye istekliyseniz

Not:Tam teşekküllü kötü amaçlı yazılım veri bilimi araçları için örnekler ve başlangıç yerleri olması amaçlanan bu araçlar sağlam bir şekilde uygulanmamaktadır. Ubuntu 17 üzerinde test edilmişlerdir ve bu platformda çalışmaları beklenmektedir, ancak doğru gereksinimleri yükleme konusunda biraz çalışarak, araçları macOS ve diğer Linux sürümleri gibi diğer platformlarda oldukça kolay bir şekilde alabilmelisiniz. .

Bu bölümde, bu kitapta sağlanan yeni ortaya çıkan araçları, göründükleri sırayla gözden geçireceğiz.

\*Shared Hostname Network Visualization:Paylaşılan Ana Bilgisayar Adı Ağ Görselleştirme

Paylaşılan bir ana bilgisayar adı ağ görselleştirme aracı Bölüm 4'te verilmiştir ve ch4 / code / list 4-8.py adresinde bulunmaktadır. Bu araç, hedef kötü amaçlı yazılım dosyalarından ana bilgisayar adlarını çıkarır ve ardından dosyalar arasındaki bağlantıları, içlerinde bulunan yaygın ana bilgisayar adlarına göre gösterir.

Araç, girdi olarak bir kötü amaçlı yazılım dizinini alır ve ardından görselleştirebileceğiniz üç GraphViz dosyası çıkarır. Bu aracın gereksinimlerini kurmak için ch4 / code dizinindeki bash install\_requirements.sh komutunu çalıştırın. A-1 listesi, araçtan "yardım" çıktısını gösterir, ardından parametrelerin ne anlama geldiğini tartışırız.

Liste A-1: Bölüm 4'te verilen paylaşılan ana bilgisayar adı ağ görselleştirme aracından yardım çıktısı

Liste A-1'de gösterildiği gibi, paylaşılan ana bilgisayar adı görselleştirme aracı dört komut satırı bağımsız değişkeni gerektirir: target\_path (1), output\_file (2), malware\_projection (3) ve hostname\_projection (4). Target\_path parametresi, analiz etmek istediğiniz kötü amaçlı yazılım örnekleri dizininin yoludur. Output\_file parametresi, programın, kötü amaçlı yazılım örneklerini içerdikleri ana bilgisayar adlarına bağlayan ağı temsil eden bir GraphViz .dot dosyası yazacağı dosyaya giden bir yoldur.

Malware\_projection ve hostname\_projection parametreleri de dosya yollarıdır ve programın bu türetilmiş ağları temsil eden .dot dosyalarını yazacağı konumları belirtir (ağ projeksiyonları hakkında daha fazla bilgi için Bölüm 4'e bakın). Programı çalıştırdıktan sonra, ağları görselleştirmek için Bölüm 4 ve 5'te açıklanan GraphViz paketini kullanabilirsiniz. Örneğin, kendi kötü amaçlı yazılım veri kümelerinizde Şekil A-1'de oluşturulan .png dosyası gibi bir dosya oluşturmak için fdp malware\_projection.dot -Tpng -o malware \_projection.png komutunu kullanabilirsiniz.

Figure A-1: Sample output from the shared hostname visualization tool given in Chapter 4

\*Shared Image Network Visualization:

Bölüm 4'te ch4 / code / list-4 12.py adresinde bulunan paylaşılan bir görüntü ağı görselleştirme aracı sunuyoruz. Bu program, paylaştıkları gömülü görüntülere göre kötü amaçlı yazılım örnekleri arasındaki ağ ilişkilerini gösterir. Araç, girdi olarak bir kötü amaçlı yazılım dizinini alır ve ardından görselleştirebileceğiniz üç GraphViz dosyası çıkarır. Bu aracın gereksinimlerini kurmak için ch4 / code dizinindeki bash install\_requirements.sh komutunu çalıştırın. Şimdi, aracın "yardım" çıktısındaki parametreleri tartışalım (bkz. Liste A-2).

Liste A-2: Bölüm 4'te verilen paylaşılan kaynak ağı görselleştirme aracından yardım çıktısı

Liste A-2'de gösterildiği gibi, paylaşılan görüntü ilişkileri görselleştirme aracı dört komut satırı argümanı gerektirir: target\_path (1), output\_file (2), malware\_projection (3) ve resource\_projection (4). Paylaşılan ana bilgisayar adı programında olduğu gibi, burada hedef\_yol, analiz etmek istediğiniz kötü amaçlı yazılım örneklerinin dizinine giden yoldur ve output\_file, programın bağlanan iki bölümlü grafiği temsil eden bir GraphViz .dot dosyası yazacağı dosyaya giden bir yoldur. içerdikleri görüntülere kötü amaçlı yazılım örnekleri (ikili grafikler Bölüm 4'te tartışılmaktadır). Malware\_projection ve resource\_projection parametreleri de dosya yollarıdır ve programın bu ağları temsil eden .dot dosyalarını yazacağı konumları belirtir. Paylaşılan ana bilgisayar adı programında olduğu gibi, programı çalıştırdıktan sonra, ağları görselleştirmek için GraphViz paketini kullanabilirsiniz. Örneğin, kendi kötü amaçlı yazılım veri kümelerinizde fdp resource\_projection.dot -Tpng -o resource projection.png komutunu kullanarak Şekil 4-12, sayfa 55 içinde gösterilen .png dosyası gibi bir dosya oluşturabilirsiniz.

\*Malware Similarity Visualization:

Bölüm 5'te, kötü amaçlı yazılım benzerliğini ve paylaşılan kod analizi ve görselleştirmesini tartışıyoruz. Sağladığımız ilk örnek araç, ch5 / code / list\_5\_1 .py içinde verilmiştir. Bu araç, girdi olarak kötü amaçlı yazılım içeren bir dizini alır ve ardından dizindeki kötü amaçlı yazılım örnekleri arasındaki paylaşılan kod ilişkilerini görselleştirir. Bu aracın gereksinimlerini kurmak için ch5 / code dizinindeki bash install\_requirements.sh komutunu çalıştırın. A-3 listesi, araç için yardım çıktısını gösterir.

Liste A-3: Bölüm 5'te verilen kötü amaçlı yazılım benzerlik görselleştirme aracından yardım çıktısı

Bu paylaşılan kod analiz aracını komut satırından çalıştırdığınızda, iki komut satırı argümanını iletmeniz gerekir: target\_directory (1) ve output\_ dot\_file (2). Bu örnekler arasında bir kenar oluşturup oluşturmayacağını belirlemek için programın iki örnek arasındaki Jaccard indeksi benzerliğiyle kullandığı eşiği ayarlamak için isteğe bağlı bağımsız değişken olan jaccard\_index\_threshold (3) 'ü kullanabilirsiniz. Jaccard indeksi Bölüm 5'te ayrıntılı olarak tartışılmıştır. Şekil A-2, output\_dot\_file dosyasını fdp output\_dot\_file.dot -Tpng -o benzerity\_network.png komutuyla oluşturduğunuzda bu araçtan örnek çıktıyı göstermektedir. Bu, az önce açıkladığımız APT1 kötü amaçlı yazılım örnekleri için aracın çıkardığı paylaşılan kod ağıdır.

Şekil A-2: Bölüm 5'te verilen kötü amaçlı yazılım benzerlik analiz aracından örnek çıktı

\*Malware Similarity Search System:Kötü Amaçlı Yazılım Benzerlik Arama Sistemi

Bölüm 5'te sunduğumuz ikinci kod paylaşım tahmin aracı, ch5 / code / list\_5\_2.py'de verilmiştir. Bu araç, bir veritabanındaki binlerce örneği dizine eklemenize ve ardından bunlar üzerinde kötü amaçlı bir sorgu örneğiyle benzerlik araması yapmanıza olanak tanır; bu, bu örnekle muhtemelen kod paylaşan kötü amaçlı yazılım örneklerini bulmanıza olanak tanır. Bu aracın gereksinimlerini kurmak için ch5 / code dizinindeki bash install\_requirements.sh komutunu çalıştırın. A-4 listesi, araç için yardım çıktısını gösterir.

Liste A-4: Bölüm 5'te verilen kötü amaçlı yazılım benzerliği arama sisteminden yardım çıktısı

Bu aracın çalıştırılabileceği dört mod vardır. İlk mod, LOAD (1), kötü amaçlı yazılımları benzerlik arama veritabanına yükler ve parametresi olarak, içinde kötü amaçlı yazılım bulunan bir dizine işaret etmesi gereken bir yol alır. LOAD'u birden çok kez çalıştırabilir ve her seferinde veritabanına yeni kötü amaçlı yazılım ekleyebilirsiniz.

İkinci mod, SEARCH (2), tek bir kötü amaçlı yazılım dosyasının yolunu parametresi olarak alır ve ardından veritabanında benzer örnekleri arar. Üçüncü mod, COMMENT (3), argüman olarak bir kötü amaçlı yazılım örnek yolunu alır ve ardından bu örnek hakkında kısa bir metinsel yorum girmenizi ister. COMMENT özelliğini kullanmanın avantajı, kötü amaçlı bir sorgu örneğine benzer örnekler aradığınızda, benzer örneğe karşılık gelen yorumları görmeniz ve böylece sorgu örneği hakkındaki bilginizi zenginleştirmenizdir.

Dördüncü mod olan wipe (4), farklı bir kötü amaçlı yazılım veri kümesini yeniden başlatmak ve dizine eklemek istemeniz durumunda benzerlik arama veritabanındaki tüm verileri siler. A-5 Listesi, bir ARAMA sorgusundan alınan bazı örnek çıktıları gösterir ve size bu aracın çıktısının nasıl göründüğüne dair bir lezzet verir. Burada, daha önce LOAD komutunu kullanarak açıklanan APT1 örneklerini indeksledik ve daha sonra APT1 örneklerinden birine benzer örnekler için veritabanında arama yaptık.

Liste A-5: Bölüm 5'te verilen kötü amaçlı yazılım benzerliği arama sistemi için örnek çıktı

\*Machine Learning Malware Detection System:

Kendi kötü amaçlı yazılım analizi çalışmanızda kullanabileceğiniz son araç, Bölüm 8'de kullanılan ve ch8 / code / complete\_detector.py adresinde bulunan makine öğrenimi kötü amaçlı yazılım algılayıcısıdır. Bu araç, bir kötü amaçlı yazılım tespit sistemini kötü amaçlı yazılım ve zararlı yazılımlar konusunda eğitmenize ve ardından bu sistemi yeni bir örneğin kötü niyetli mi yoksa zararlı mı olduğunu tespit etmek için kullanmanıza olanak tanır. Bu aracın gereksinimlerini ch8 / code dizininde bash install.sh komutunu çalıştırarak kurabilirsiniz. A-6 Listesi, bu araç için yardım çıktısını gösterir.

Liste A-6: Bölüm 8'de verilen makine öğrenimi kötü amaçlı yazılım algılama aracı için yardım çıktısı

Bu aracın çalıştırılabileceği üç mod vardır. Değerlendirme modu (4), sistemi eğitmek ve değerlendirmek için seçtiğiniz veriler üzerinde sistemin doğruluğunu test eder. Bu modu, python tam dedektör.py -malware\_paths <içinde kötü amaçlı yazılım bulunan dizine giden yol> --benignware \_paths <içinde iyi huylu yazılım bulunan dizine giden yol> çalıştırarak çalıştırabilirsiniz. Bu komut, dedektörünüzün ROC eğrisini gösteren bir matplotlib penceresi başlatacaktır (ROC eğrileri Bölüm 7'de tartışılmıştır). Şekil A-3, değerlendirme modundan bazı örnek çıktıları gösterir.

Şekil A-3: Bölüm 8'de sağlanan kötü amaçlı yazılım algılama aracından örnek çıktı, değerlendirme modunda çalıştırın

Eğitim modu, bir kötü amaçlı yazılım algılama modelini eğitir ve onu diske kaydeder. Bu modu python complete\_detector.py çalıştırarak çalıştırabilirsiniz.

  malware \_paths (1) <içinde kötü amaçlı yazılım bulunan dizine giden yol> - benignware\_paths (2) <içinde iyi huylu yazılım bulunan dizine giden yol>. Bu komut çağrısı ile değerlendirme modunun çağrılması arasındaki tek farkın --evaluate bayrağını bırakmış olmamız olduğunu unutmayın. Bu komutun sonucu, kaydedilen\_detector.pkl adlı bir dosyaya kaydettiği ve mevcut çalışma dizininize kaydedilen bir model oluşturmasıdır.

Üçüncü mod, scan (3), Saved\_detector.pkl'yi yükler ve ardından kötü amaçlı olup olmadığını tahmin ederek bir hedef dosyayı tarar. Tarama yapmadan önce egzersiz modunu çalıştırdığınızdan emin olun. Sistemi eğittiğiniz dizinde python complete\_detector.py -scan\_file\_path <PE EXE dosyası> çalıştırarak bir tarama çalıştırabilirsiniz. Çıktı, hedef dosyanın kötü amaçlı olma olasılığı olacaktır.

<https://nostarch.com/androidsecurity>

BUNA BİR DAHA BAK !!!!

BİRDE SONUNU KİTABIN BAK!!,