

Chapter 0 - 1 (Introduction) (2)

* Malware analizi yapmanın temel amacı zararlı yazılımın sisteme ne gibi zararlar verdiğiini anlamaktır. • Tespit Etme

2 * Kitap bir durum senaryosu ile başlamış. • Zararı Telafi Etme
• Açığı Kapatma

3 Kurumsal bir firmayı network güvenliğinden sorumlusunuz ve patronunuz gecenin bir yarısı arayıp bilgisayarlarda bir malware olduğunu bildiriyor. Şirkete gidiyorsunuz malware programını buldunuz ve onu siliyorsunuz.

İki gün sonra patronunuz tekrardan arıyor ve bilgisayarlarda bir malware olduğunu söylüyor.

* Sistemde bir malware bulunduktan sonra sunları bilmeye ihtiyacımız var:

- Saldırgan sisteme bir rootkit veya trojan bıraktı mı?
- Saldırgan gerçekten gitti mi?
- Saldırgan sistemden ne çaldı?
sistemde neyi değiştirdi?
sisteme neyi ekledi?
- Saldırgan içeriye nasıl girdi? Network, USB --

* Sorunun kaynağının çözümlenmesi gereklidir.

Sistem güvenliğinin geliştirilmesi gereklidir.

5 * 2012 yılında Hacker'lar LinkedIn sisteme giriyorlar ve 6.5 milyondan fazla kullanıcının bilgisini çalıyorlar. Ve bu bilgileri Rusya'daki bir forma paylaşıyorlar.

Olayın duyulmasından sonra LinkedIn bu durumu yalanlıyor. Ondan sonra 1 milyon dolar olayın araştırılması için daha sonra 3 milyon dolar da sistemi daha güvenli hale getirmek için harcama yapıyor.

Bu tür olaylarda zaten şirketler,

- Bir kaç hafta yalanlıyorlar
- Sonra çalınan bilgilerin bir tehdit oluşturmadığı söyleniyor.
- Daha sonraki günlerde şirket sistem güvenliğini daha da iyileştirebilmek adına harcamalar yapıyor.

* Zararlı yazılımın anatomisi sunları anlıyor olmaktan geçiyor:

- Zararlı yazılımı nasıl tespit edeceğiz?
- 6 - Zararlı yazılım nasıl çalışıyor?
- Zararlı yazılımı nasıl etkisiz hale getireceğiz?
- Sistemi eski haline nasıl getireceğiz?

* Sistemde bir malware olduğunu tespit ettiniz, ne yaparsınız?

Su sorulara yanıt veriyor olmanız gereklir:

- Z → Zararlı yazılımın yol açtığı tehdit ortadan nasıl kaldırılır?
- E → Network imzası (kuralları) nasıl daha etkin hale getirilir?
- Sor → Network üzerinde başka bir makine enfekte olmuş mu?
- Yaz → Zararlı yazılımın bütün bileşenlerini sistemden sildiğimizden nasıl emin olabilirim? Zararlı yazılımı gerçelten sildik mi, yoksa hala içerisinde bazı bileşenleri var mı?
- Zararlı yazılım sisteme ne gibi zararlar verdi? Bu zararı telafi edebilir miyiz?

* Malicious Software → Kötü niyetli yazılım
Zararlı yazılım
Malware

Zararlı Yazılım (Malware): Bir kullanıcıya, kullanıcının hesa, bina, bilgisayara, ağa zarar veren veya sistemden bilgi çalan yazılımlara zararlı yazılım diyoruz.

<u>Kitap</u>	Virüs	Truva Atı	Solucan	Rootkit	Korkutucu Yaz.	Casus Yaz.
<u>Yaz</u>	(Virus)	(Trojan Horses)	(Worm)	(Rootkit)	(Scareware)	(Spyware)
<u>Fidye Yaz.</u>						
<u>(Ransomware)</u>						

Zararlı Yazılım Analizi (Malware Analysis): Zararlı yazılımin nasıl çalıştığını, nasıl tanımlanacağını, nasıl yenileceğini veya ortadan kaldırılacağını anlamak için bu tür yazılımları inceleme sanatıdır.

Chapter 0

* Zararlı yazılım analizi iki farklı imza türüne bakarak yapılabilir.

Ana Bilgisayar Tabanlı İmzalar (Host-based Signatures): Bu tür imzalar genellikle kötü amaçlı yazılım tarafından oluşturulan veya değiştirilen dosyaları veya kayıt defterlerinde yaptığı belirli değişiklikleri tanımlar.

Anti-virus programlarının virüsten koruma imzalarından farklı olarak bu tür imzalar zararlı yazılımin özelliklerine değil zararlı yazılımin sisteme ne yaptığına odaklanır.

- Dosya silme
- Dosya yaratma
- Dosya değiştirme
- Sistem ve kullanıcı tercihlerini değiştirme
- Kayıt defteri değerlerini değiştirme

▽ Regedit aç → Bir örnek yapılabilir.

Ağ İmzaları (Network Signatures): Ağ imzaları, ağ trafiğini izleyerek zararlı yazılımları tespit etmek için kullanılır. Bir program ağ üzerinde nasıl davranışlar sergiliyor ise zararlı yazılım diyebiliriz, bunu inceler.

Belki de sisteminiz daha önceden enfekte oldu ve su an uzaktan kontrol edilen bir bot, yani bir botnet'in parçası olabilir.

- DDOS saldırısı yapıyor olabilir.
- Belki de sisteminizden her 30 saniyede bir mesaj bir bilgi gönderiliyor olabilir.

▽ Wireshark ile bir ağ trafiği örneğine bakabilirsin.

* False Positive (Yanlış Pozitif)

Aslında yaptığımız iş ikili bir sınıflandırma $\begin{cases} \rightarrow \text{Malware} \\ \rightarrow \text{Normal} \end{cases}$

ikili sınıflandırıcılarında siz bir bilinmeyeni bir sinifa atadığınızda
yani tahmin yaptığınızda kaç farklı durum olusur?

9

Tahmin

		<u>Güçek</u>		Confusion Matrix
		<u>Positive</u>	<u>Negative</u>	
<u>Positive</u>	<u>True Positive</u>	<u>False Positive</u>	Ama bunun oranı da fazla olursa sıkıntı	
	<u>False Negative</u>	<u>True Negative</u>	Amaç Buraya düşmemek	

True Positive \rightarrow Bir yazılıma positive demisiz, yani bir yazılıma zararlidir demisiz ve gerekçetten bu yazılım zararlı.

* Genel hatları ile 2 farklı zararlı yazılım analiz teknigi var:

Statik Analiz: Zararlı yazılımlar çalıştırılmadan incelenir.

11

Araçlar:

- VirusTotal
- strings X more
unix, Linux
Vista, Win10
win7
- Disassemblers \rightarrow IDA Pro

! Eğer zararlı program çalışma zamanında bir DLL'yi uygunsuz kullanıyor ise bu durumu statik analiz ile tespit edemeyiz.

Executable bir dosyadan bir insanın okuyabileceği bazı bilgiler edinmemizi sağlar.

6

Dinamik Analiz: Zararlı yazılım çalıştırılarak analiz edilir.

Bir Virtual Machine içerisinde zararlı yazılımı çalıştırırsınız ve zararlı yazılım arka planda çalışırken bu yazılımı izleyip sistem ile ilgili nelerin değiştigini, network'te neler olduğunu haber veren yardımcı yazılımlar çalıştırırız.

Araçlar: RegShot

Process Monitor

Process Explorer

WireShark

Volatility

Sonuçta zararlı yazılımlar kendilerini sürekli güncelliyor ve geliştiyorlar bu sebeple bu yardımcı araçlara da %100 güvenmemeliyiz.

* Bu analiz teknikleri de kendi içerisinde ikiye ayrılıyor.

Basic Static Analysis: Bir zararlı yazılımı kodlarına bakmadan ve çalıştmadan analiz ederiz.

Hızlı ve basittir fakat spesifik yazılmış önemli davranışları olan zararlı yazılımları tespit edemezler.

12 Bu analiz çeşidi bir dosyanın zararlı olup olmadığı, fonksiyonelliği hakkında bazı bilgileri ve bazen de ağ imzası (network signature) hakkında veriler toplamamıza yardımcı olur.

Bu analiz türü zararlı yazılımlar incelemek için kullanılan ilk adımdır.

Basic Dynamic Analysis: Zararlı yazılım güvenli bir ortamda (Virtual Machine) çalıştırılır ve davranışları izlenir.

Aynı basic static analysis'de olduğu gibi basic dynamic analysis de de derin ve geniş bir programlama bilgisi gerektirmeden analiz yapılabilir ama iyi yazılmış fonksiyonel virüsleri tespit edemezler.



Advanced Static Analysis: Analiz teknikleri içerisinde en zor olanıdır. Tersine mühendislik (reverse-engineering) teknikleri kullanılır. Çalışabilen bir EXE dosyası bir disassembler yardımcı ile assembly koduna dönüştürülür.

13

Assembly kodları zor okunabilen kodlardır ve derin programlama bilgisi gerektirir.

Advanced Dynamic Analysis: Çalıştırılabilir bir EXE dosyası debugger (hata ayıklayıcı) içerisinde çalıştırılır. Debugger'lar programın herhangi bir noktasına breakpoint koyarak o programı belirlenen bu noktada gözlemimize yardımcı olurlar.

Zararlı Yazılım Türleri

1) Ransomware (Fidye Yazılımı) : Fidye ödenene kadar bir hedefin verilerine erişimini devre dışı bırakmak için şifreleme kullanan bir yazılımdır. Saldırıya uğrayan firma ödeme yapana kadar kısmen veya tamamen çalışmaz hale gelir ancak ödeme yapsanız dahi dosyalarınıza ulaşabileceğinizin garantisini yoktur.

Örnek: Örnek olarak 8-10 sene önce Baltimore şehrinde ortaya çıkan Robin Hood ismindeki yazılımı verebiliriz. Bu yazılım vergi tahsilatı, mülk transferleri ve şehrin e-posta ağını ele geçiriyor ve bunları düzeltmesi karşılığında bir ücret talep ediyor. Bu saldırı şehre 18 milyon dolara mal oluyor.

2) Fileless Malware (Dosyasız Zararlı Yazılım) : Dosyasız kötü amaçlı yazılım başlangıçta herhangi bir şey yüklemeyen, bunun yerine PowerShell veya ^{Windows Management} WMI gibi işletim sisteme özgü dosyolarada değişiklik yapar. İşletim sistemi darenlenen dosyaları meşru olarak tanıdığını, dosyasız yapılan bir saldırının antivirus yazılımı tarafından yakalanmadığını. Ve bu saldırılardan on kat kadar daha başarılıdır.

Örnek: Örnek olarak Astaroth yazılımını verebiliriz. Legal Windows dosyaları ile birlikte çalışıyor dolayısı ile fark edilmiyor. Bu saldırıda sadice bellekte yürütülen bilgiler kullanıldı ve kullanıcıların kimlik bilgileri çalındı.

3) Casus Yazılım (Spyware): Casus yazılım bilgileri ve rızası olmadan kullanıcıların etkinlikleri hakkında bilgi toplar. Bu bilgiler parolalar, pimler ödeme bilgileri ve yapılmamış mesajları içerebilir.

Casus yazılım kullanımı masüsstür tarayıcıyla sınırlı değildir, ayrıca kritik bir uygulamada veya bir cep telefonunda da çalışabilir.

Örnek: Örnek olarak DarkHotel isimli yazılımı verebiliriz. Otellerin halka açık (public) WIFI'lerini kullanarak işletme ve hükümet liderlerini hedef alıyorlar ve belirli güclü kişilerin sistemlerine erişim sağlamak için bu zararlı yazılımı kullanıyorlar. Bu erişim elde edildiğinde saldırganlar hedeflerinin şifrelerini ve diğer hassas bilgilerini yakalamak için tuş kaydedicileri (keylogger) kullanıyorlar.

4) Adware (Reklam Yazılımı) : Adware hangi reklamların sunulacağını belirlemek için bir kullanıcının gezinme etkinliğini izler. Reklam yazılımı casus yazılıma benzesede kullanıcının bilgisayarına herhangi bir yazılımı yüklemeyi veya tuş vuruslarını yakalamaz.

Reklam yazılımdaki tehlike bir kullanıcının gizliliğinin ihlal edilmesidir. Reklam yazılımı tarafından yakalanan veriler, kullanıcının internetteki başka bir yerdeki etkinlikleri hakkında açık veya gizli olarak toplanan veriler ile harmanlanır ve bu içiшинin arkadaşlarının kim olduğunu içeren bir profili oluşturmak için kullanılır. Ne satın aldıkları, nereye seyahat ettikleri gibi bilgiler. Bu bilgiler kullanıcının izni olmadan reklamverenlerle paylaşılabilir veya satılabilir.

Örnek: Fireball adlı reklam yazılımı 2017 yılında 250 milyon bilgisayarı ve cihazı etkiledi. Varsayılan arama motorlarını değiştirmek ve web etkinliğini izlemek için tarayıcıları ele geçirdi. Bununla birlikte, bu yazılımların dörtte üçü uzaktan kod cağıstırabiliyor ve kötü amaçlı dosyalar indirebiliyordu.

5) Trojan Horses (Truva Atı): Bir Truva Atı kendisini istenen kod veya yazılım olarak gösterir. Truva atı səphelenmeye kulanıcılar tarafindan indirildikten sonra kötü amaçlarla kurbanların sistemlerinin kontrolünü ele geçirebilir. Turuva atlari günlük uygulamalarımızda, oyunlarda ve hatta yazılım yamalarında gizlenebilir veya kimlik qui e-postalarında bulunan eklerde yerleştirilebilir.

Örnek: Emotet, 2014'ten beri piyasada bulunan spesifik bir bankacılık truva atıdır. İmza tabanlı çalışan güvenlik sistemlerinden kaçmayı başarır dolayısı ile tespit edilmesi zordur. Yayılmasına yardımcı ek modüller içerir.

6) Worm (Solucan): Solucanlar kendilerini ağlara kurmak için işletim sistemlerindeki güvenlik açıklarını hedefler. Birkac yolla sistemimize sizabilirler.

- Yazılımlara yerleştirilmiş arkaparkalar (back door)
- Kasıtsız yazılım açıkları
- USB bellekler

Solucanlar bir kez yerleştirildikten sonra kötü niyetli kişiler tarafından DDOS saldırıları başlatmak, hassas verileri çalmak veya fidye yazılımı saldırıları gerçekleştirmek işin kullanılabilir.

Örnek: Geçtiğimiz bir kaç on yılda bir çok virus bilgisayar ortamında yayıldı. Bunlardan bir tanesi Stuxnet virusudur. İddaya göre Stuxnet virusü ABD ve İsrail istihbarat güçleri tarafından İran'ın nükleer programına zarar vermek amacıyla yazılıyor. Virus bir flash sürücü aracılığı ile İran'ın şevisine tanıtılmıyor. Stuxnet flash'lardan bilgisayarlara, bilgisayarlardan flash'lara --- hızlı bir şekilde yayıldı ancak tek işlevi uranyum zenginleştirme sürecini baltalmak olduğu için normal kullanicılara zarar vermedi.

7) Virus (Virüs): Kendisini bir uygulamaya ekleyen ve uygulama çalıştırıldığında çalışan bir kod parçasıdır. Bir uygulama ile birlikte bir sisteme girdikten sonra hassas verileri çalabilirler enfekte ettiği sisteme veya başka sistemlere zarar verebilirler.

Virüsler ve Truva Atları

Bir virus bulastığı uygulama çalışmadığı sürece yürütülmmez veya çoğaltılamaz. Bir ana bilgisayar uygulamasına olan bu bağılilik, virüsleri kullanıcıların indirmesini gerektiren truva atlarından ve çalışmak için uygulamaları kullanmayan solucanlardan farklılıklar. Birçok kötü amaçlı yazılım örneği birden fazla zararlı yazılım kategorisine girebilir. Örneğin Stuxnet bir solucan, bir virus ve bir rootkit'dir.

8) Rootkit (Kök Seti – Rootkit) : Rootkit kötü niyetli kişilerin kurbanın bilgisayarını tam yönetici ayrıcalığı ile uzaktan kontrol etmesini sağlayan bir yazılımdır. Rootkit'ler uygularımlara, sistem çekirdeğine veya firmware'larla elde edilebilirler. Kimlik avı, kötü amaçlı ekler, kötü amaçlı indirmeler veya güvenliği zayıf olan ortak sürücüler yoluyla yayılırlar.

Örnek: Zaino, Rootkit türü zararlı yazılımlara gerçek dünyadan bir örnek. Kullanıcılar sahte bir VPN uygulaması indirdiğinde sistemlere bulasır. Zaino, yüklenikten sonra, rakip kötü amaçlı yazılımlar için bir güvenlik taraması yapar ve onu kaldırmaya çalışır. Ardından görünmez tarayıcılar açar ve içerik ile bir insan gibi etkileşime girer; Kaydırarak, vurgulayarak, nesnelerin üzerine gelerek veya tıklayarak. Bu aktivite davranış analizi yazılımını kandırma işindir.

Zaino astığı görünmez tarayıcılardaki reklamlara tıklayarak site sahibinin haksız bir kazanç elde etmesini sağlar.

9) Keyloggers (Tuş Kaydedici) : Keylogger'lar kullanıcı aktivitesini inceleyen bir casus yazılım türüdür. Keylogger'ların meşru, legal kullanım alanları vardır: İşletmeler çalışan faaliyetlerini izlemek için bunları kullanabilir ve aileler, çocuklarının çevrimiçi davranışlarını takip etmek için keylogger'ları kullanabilirler. Bununla birlikte kötü amaçlarla yüklediğinde şifre verilerini, bankacılık bilgilerini ve diğer hassas bilgileri

Örnek: Bu alandaki bilindik bir örnek ise Olympic Vision key-logger'. Sistemlere buluştanmak için önce kimlik avı ve sosyal mühendislik tekniklerini kullanıyor. Daha çok iş adamlarını ve onların şahsi bilgilerini çalmayı hedef alan bir yazılım.

10. Bots / Botnets : Bir Bot komut geldiğinde otomatik görevler içeren bir yazılım uygulamasıdır. Arama motorlarını indekslemek için legal amaçlarla da kullanılabılır. Ama kötü amaçlarla kullanıldıklarında merkezi bir sunucuya bağlanıp bu sunucudan komut beklerler ve kendi kendilerine hızla yayılırlar.

Genellikle botlar DDos saldıruları gibi uzaktan kontrol edilen geniş saldırı sellemini başlatmak için kullanılırlar ve bir BotNet'in parçası olurlar. Bot ağları yayılım hızına bağlı olarak oldukça genişleyebilir. Örneğin Mirai IoT botnet'i yaklaşık 2 Milyon civarı bilgisayarı bot haline getirmiştir.

Örnek: Echobot, yaygın olarak bilinen Mirai'nın bir sesididir. 50'den fazla farklı güvenlik açığından yararlanarak çok çeşitli IoT cihazlarına saldırır. Bu yazılım yamalanmamış yanı güvenlik açıklarını kapatılmamış eski sistemleri arar. Echobot kötü niyetli kişiler tarafından DDos saldıruları başlatmak, tedarik zincirlerini kesintiye uğratmak, hassas tedarik zinciri bilgilerini çalmak ve kurumsal sabotaj yapmak için kullanılabilir.

* Zararlı Yazılımları bir hedef gözetip gözetmemesine göre ikiye ayıralım.

22

Mass Malware

- Hedef gözetmez,
- Mمكün olduğunda yayılmayı amaçlar.
- Spesifik değildir, basittir.
- Tespit edilmesi, engellenmesi ve yok edilmesi kolaydır.
- Basit analiz teknikleri ile tespit edilebilir.

Target Malware

- Hedef gözetir.
- Yayılım amacı yoktur.
- Spesifiktir, komplekstir.
- Tespit edilmesi, engellenmesi ve yok edilmesi zordur.
- Tespit etmek için genellikle ileri analiz teknikleri kullanılır.

* Zararlı yazılım analiz etmenin genel kuralları

23

→ Ayrıntıya fazla takılmayın, ve temel özelliklere odaklanın.
→ Zararlı yazılım programlarının çoğu büyük ve karmaşıktır.

24

Muhtemelen her ayrıntıyı anlayamayız.
→ Zor ve anlaşılmayan bölümler ile karşılaşığınızda temel bir fikir edinin, Resmin tamamına bakın.

→ Farklı işler için farklı araçların ve yaklaşımın mevcut olduğunu unutmayın. Tek bir yaklaşım yok. Bir araç ve yöntem ile işin içerisinde çıkmadıysanız yöntem ve aracınızı değiştirin.
Yani farklı bir açıdan bakmaya çalışın.

→ Her durum farklıdır ve her durumu kendi özeline değerlendirin.

→ Zararlı yazılım analizi ilerlemeli bir süreçtir yani birikimli bir süreçtir.

Zararlı yazılım analiz teknikleri gelişikçe, zararlı yazılım yazarları analizi engellemek için yeni teknikler tasarılarlar.

Zararlı yazılım analisti olarak başarılı olmak için, yeni teknikleri tanıyabilmeli, anlayabilmeli ve yenebilmelisiniz.

* Basic Static Analysis neleri içerir?

→ Antivirus taraması

25 → Hashes → Hash'ler bir programız veya dosyanın parmak izidirler.

26 → Bir exe dosyasının kullandığı string'ler, fonksiyonlar, DLL dosyalarına bakılabilir veya bu exe dosyasının header yani başlık bilgisine bakılabilir.

* Anti Virus Taraması

27 Malware'lar imzalarını kolayca değiştirdip güncel olmayan bir antivirus programını atlatabilirler.

28 Bu yüzden Virustotal kullanışlı bir araç, aynı anda onlarca antivirus programının sözkonusu dosyayı incelemesine olanak tanıyor.

Virustotal Örnek Yap - Dosya - Hash - IP, domainname - Link (dosya linki)	→ pdf → normal bir exe → Lab 1-1 → Lab X.X
---	---

* Hashing

Cizim Yap

Hash'ler zararlı yazılımların parmak izi gibidir.
Aslında bütün dosyaların dijital parmak izidir.

29
30
31
Hash'ler aslında veriye daha kolay erişim sağlamak için geliştirilmiş veri yapılarıdır diyebiliriz. Veriyi Hash'leyip kaydederseniz zamanı geldiğinde o veriye daha hızlı ulaşabilirsiniz.

Ancak günümüzde Hash fonksiyonları güvenlik amacı ile de kullanılmaktadır. Sistem güvenliği söz konusu olduğunda bir çok yerde kriptografik hash fonksiyonlarını görmekteyiz.

report collision MD5 128 bit	report collision SHA-1 160 bit	No collisions reported SHA-2 256 bit
---------------------------------------	---	---

Peki Hash fonksiyonlarının güvenlik alanında yıldızlarının paramasını sağlayan özellikleri neler?

Hash Fonksiyonları Özellikleri

- Çeşitli uzunluktaki verileri sabit uzunlukta eşsiz bir değere dönüştürür.
- Aynı Hash fonksiyonu girdi uzayındaki bir D verisini çıktı uzayında her zaman aynı yere map eder.
- Hash fonksiyonları tersinir çalışmazlar. Yani hash fonksiyonları tek yönlü çalışırlar.
- Hash fonksiyonlarının çıktıları tahmin edilebilir değildir.
- Farklı iki input'un output uzayında çakışmasını sağlamaya çalışır. Bunu başarabiliyorsa hash fonksiyonu kalitelidir diyebiliriz. Hatta input'lari çıktı uzayına uniform dağıtıyor olmak hash fonksiyonlarının istenen özelliklerindenidir.

Hash fonksiyonlarının kullanıldığı alanlar

- Dosyaları kolay bulmak adına indekslemek için kullanılır.
 - Network mesajlarında bütünlüğü (integrity) korumak için kullanılır.
 - Kullanıcı şifrelerini veritabanlarına kaydederken kullanılır.
 - Yapısal P2P ağlarda hem doğal bir güvenlik mekanizması sunar, hem de yük dengesini (load balance) sağlar.
 - Zararlı yazılımları etiketlemek için kullanılır.
- Hash Calc Programı Örnek**
- 1 tane normal dosya
1 tane zararlı dosya
MD5 ve SHA1 internetten arastır.

* Eğer dijital imzaları (hash) kullanarak bir dosyanın zararlı olup olmadığını araştırıyorsak;

1. Dosyanın hash'ini alın. En sık kullanılanları MD5 ve SHA-1
2. Bu hash değerini internette aratın. Search sonucu gelen sayfalardan dosya ile alakalı bilgiler edinin.

32

Eğer bir dosyanın zararlı olduğunu tespit etmişseniz;

1. Dosyanın hash'ini alarak onu etiketleyin
2. Dosya ile ilgili edindiğiniz verileri ve dosyanın hash'ini internette kendi sayfanızda veya ilgili formlarda paylaşın.

33

Finding Strings

Biz executable dosyaların içerdiği string ifadelerle bakarak da bu dosyanın zararlı işler yapıp yapmadığını hızlı bir şekilde tespit edebiliriz.

* char olarak tanımladığımız tek karakterden oluşan değişkenlerin dizisine "karakter katarı" yani string ifade ediyoruz.

→ Basılıabilir karakterlerden oluşan diziye string deriz.

34 → String'ler bir null karakteri ile sonlanırlar (0x00)

Dosya okurken de null karakterini görene
kadar döneriz while döngüsü ile.

Hexadecimal
ASCII format

→ ASCII karakterler 8 bit uzunluğundadır.

Unicode karakterler ise 16 bit uzunluğundadır.

* Bir dosyadaki string ifadeleri bulabilmek için "strings" komutu kullanılıyor. Bu komut dosyadaki hata mesajlarını, komutlar, yardım menüsünü, fonksiyon isimlerini, ip adreslerini ve kullanılan kütüphane dosyalarını görmemizi sağlar.

Bu komut Linux çekirdeğinde bulunan bir konut ancak windows'da da mercat.

36 strings → Unix, Linux, WinVista, Win10

37 more → Win7, WinXP, --

38 more < FilePath.exe | findstr ".">

Bu kod tek uzunluklu string ifadeleri getirir.

more < FilePath.exe | findstr "....." minimum 5 karakter uzunluğun-
daki ifadeleri getirir.

more < FilePath.exe | findstr ".dll"

BinText isimli program da aynı işi yapıyor. Bir dosyadaki string'leri buluyor.

* Paketlenmiş ve gizlenmiş zararlı yazılımlar

Çoğu zaman zararlı yazılımlar tespit edilmelerini ve analiz edilmelerini güçlendirmek için paketlenirler veya gizlenirler.

Obfuscated (Gizlenmiş, karmaşık hale getirilmiş): Zararlı kodların ve yazılımların tespit edilmemeleri için gizlenmesi anlamına gelir.

Packed (Paketlenmiş): Bir çeşit gizleme işlemidir. Paketlenen program zip dosyası gibi sıkıştırılır. Bu yüzden analiz etmek zordur. Yani string ifadeleri çok rahat bulamayabiliriz.

* → Paket programlarda kod sıkıştırılmıştır.

→ Bu da program içerisindeki string'leri ve komutları okunamaz yapar.

→ Bu sıkıştırılmış dosyaların başında bir wrapper (sarmalayıcı) programı olur. Siz wrapper programını çalıştırığınızda o da sıkıştırılmış gerçek programı unzip yaparak eski haline getirir ve çalıştırır.

* PEID programı yardımı ile bir yazılımin hangi paketleyici tarafından paketlendiğini bulabiliriz.

41 Daha sonra bu paketi unpack edebilecek bir aracı internetten buluruz ve paketlenmiş yazılımı paketlenmeden önceki hali ile inceleyebiliriz.

- * Burada örnek olarak bir c programı yazılmış ve derlenmiş. Bu programı önce gcc derleyicisi ile derleyip .exe dosyası oluşturulmuş.

42

upx programı yardımı ile de bu yazılımı tekrardan unpack yapıyorlar ve yazılımı incelemeye hazır hale getiriyorlar.

- * Yazdıkları orijinal programın içerisinde 33 817 string var iken paketlenmiş programın içerisinde 23 623 string var ve bu string ifadeler artık okunabilir değil.

- * Portable Executable File Format
- 45 Windows'taki programların tamamı taşınabilir, çalıştırılabilir dosya formatındadır.

- * Windows'taki PE'ler çalıştırılabilir dosyalar, nesne kodları ve Windows'ta çok kullanılan DLL'lerdir.

- 46 Amacımız çalıştırılabilir bir dosyayı formatına bakmadan analiz etmektir ancak bir dosyanın formatı da bize fonksiyonelligi hakkında bilgi verir.

PE file formatı Windows işletim sistemi yükleyicisinin sar- malanmış yürütülebilir kodu yönetmesi için gerekli bilgileri içeren bir veri yapısıdır.

PE dosyaları kod hakkında bazı bilgileri içerir; Uygulamanın cesidi Gerekli kütüphane dos. Gerekli alan

Bu bilgiler malware analistleri tarafından kullanılır.

- * LordPE, PE dosyalarının içeriğinin RAM'in hangi bölümünde olduğunu anlamak için kullanılan bir program. Yani çalışan bir malware programı hakkında bilgi verir.

Bir programın

→ text section
 data section
 resource section
 relocation section
 :

gibi bilgilerinin hafızada hangi adreste olduğu bilgilerini verir.

text kısmı programın kodunun bulunduğu kısımdır.

data kısmı ise kod tarafından kullanılan verilerin tutıldığı alan.

* Linked Libraries and Functions

51 Bu bölümde Windows'taki kütüphaneler yani .dll dosyaları hakkında konuşuyor olacağız.

* Programların kullandıkları fonksiyonlar kendi kodları arasında yazılmış fonksiyonlar olabilir veya daha önce yazılıp kütüphane haline getirilmiş hazır fonksiyonlar olabilir.

Yani siz programınızın içerişine

```
public double Pow ( a, n )
temp ← 1
for i ← 0 to n do
    temp ← temp * a
return temp
```

} kendi yazdığımız fonksiyon.

Math.Pow(a,n)

→ Math kütüphanesinin power fonksiyonu kullanılmış.

Bu dosyalar Linking (bağlama) özelliği ile ana programımıza bağlanır.

52

Çalıştırılabilir bir dosyanın hangi fonksiyonları import ettiğini biliyor olmamız o program hakkında genel bir fikir verebilir.

Bir kütüphane bir programa eklenecek ise şu 3 yoldan biri tercih edilir:

→ Static

→ At Runtime

→ Dynamic



Static Linking

- Windows EXE'lerinde nadiren kullanılır.
- Unix ve Linux tabanlı işletim sistemlerinde yaygındır.
- Kütüphanedeki bütün kodlar EXE dosyasına kopyalanır.
- EXE dosyalarının boyutunun artmasına sebep olur.
- Analizini yapmak su noktada zor hangi fonk. kullanıcının hangisi kütüphanenin?

53



Runtime Linking

- Normal programlar arasında popüler değil, çok sık kullanılmaz.
- Malware'lar sıklıkla kullanır, özellikle gizlenmiş ve paketlenmiş olanları
- Program çalışıyorken kütüphaneye ne zaman ihtiyaç duyulursa
 - zaman bağlanılır. Program yüklenliğinde değil.
- Genellikle Windows'un LoadLibrary ve GetProcAddress komutları ile gerçekleştirilir.

* Dynamic Linking

- 55
- En genel, en çok kullanılan bağlama yöntemi.
 - Program yükleniği zaman işletim sistemi gerekli kütüphaneleri bulur ve program ondan sonra çalışmaya başlar.

* Bu durumun yaptığımız iş açısından farkı ne?

Yani bir programın kütüphaneyi statik, dinamik veya runtime da yükleyip olmasının bizim açısından önemi nedir?

56

PE header (başlık)'ları program ile birlikte yüklenen her bir kütüphaneyi ve fonksiyonu listeler. Dolayısı ile biz bu kütüphane ve fonksiyon isimlerine bakarak programın neler yapmaya çalıştığını anlayabiliriz.

Örneğin URLDownloadToFile dosyasının belirtilen URL'den bir dosya indireceği açıklıdır.

Ancak Runtime Linking'de kullanılacak kütüphane ve fonksiyonları PE header'ında göremeyiz.

* Dependency Walker PE bilgilerini elde eder, → Hangi.dll fonksiyon:

→ Dependency Walker programı anlatılmış.

57 → Gerçek Services.exe ve sahtesi olan Services.ex - programı

58 → arasında kıyaslama yapılmış.

- Gerçek windows servisleri çok fazla sayıda dll import ederler.
- Zararlı yazılımlar az sayıda dll import ederler.

→ Bu program import edilen fonksiyonları gösteriyor.

Sözkonusu dll'in içerisinde hangi fonksiyonlar var onları gösteriyor.

* Dependency Walker (kitapta numaralandırmalar var)

- ① → İncelediğimiz programı gösteriyor.
- ② → Programın kullandığı DLL'leri gösteriyor.
- 60 ③ → Import edilen fonksiyonları gösteriyor. (Önemli)
- 61 ④ → Export edilen fonksiyonları gösteriyor. (Önemsiz) Kullanışlı bir veri değil.
- ⑤-⑥ → DLL'ler hakkında extra bilgi veriyor.

* Common DLLs

Kernel32.dll → Programlara genel bir fonksiyonelit sağlayarak sık kullanılan bir kütüphanedir. Memory'ye, dosyalara, donanıma erişimi ve manipülasyonu sağlar.

Advapi32.dll → Çekirdek Windows bileşenlerine erişmeyi sağlar. Service Manager ve Registry gibi.

User32.dll → Bu DLL bütün kullanıcı arayüzlerinden sorumludur. Buttons, scroll bar, textbox, dropdown menu, vb. Bu bileşenleri kontrol etmekten ve kullanıcı aksiyonlarına cevap vermekten sorumludur.

Gdi32.dll → Bilgisayar grafiklerini göstermeyi ve değiştirmeyi sağlayan fonksiyonları içeren kütüphanedir.

Ntdll.dll → Bu DLL Windows kernel'ini kullanmak için bir arayıldır. EXE dosyaları genellikle bu DLL'i direkt kullanmazlar, dolaylı yoldan Kernel32.dll kütüphanesi üzerinden kullanırlar. ---

Eğer çalıştırılabilir bir dosya bu DLL'ı direkt kullanıyor ise normalde Windows programlarında bulunmayan işlevleri kullanmayı amaçladığı anlamına gelir. Programın fonksiyonellliğini gizleme, çalışan process'leri manipule etme gibi özellikleri olan fonksiyonları içerir.

WSOCK32.dll → Network ile ilgili işlemlerin yapılmasını sağlayan kütüphanedir. Bir program network üzerinde bazı işlemler yapacak ise bu DLL'leri kullanır.

WININET.dll → Yüksek seviye network işlemlerini gerçekleştirmek için tasarlanmış bir kütüphanedir. FTP, HTTP, NTP vb. protokollerini implemente eder.

* DLL dosyaları implemente edilmiş olan hazır fonksiyonları EXE dosyalarının kullanımına sunarak onları Export ederler yani ihraç ederler.

64 EXE dosyaları ise DLL'leri kaynak kodlarına dahil ederek, DLL'ler içersindeki fonksiyonları Import ederler yani ithal ederler.

İşte PE file header (PE dosya başlıklar) çalıştırılabilir bir dosyanın import ve export ettiği fonksiyonların bir listesini sunar. Bu bilgi de bize dosyanın fonksiyonelligi hakkında bilgi verir.

Ders kitabı Appendix A'da malware'ların sıkça kullandıkları fonksiyonların listesi mevcut.

* Notepad programının PE başlık bilgisi görülmekte. Notepad programı hangi DLL ve fonksiyonları import ediyor hepsi listelenmiş.

66

Advance Api → Advapi32.dll 'yi import etmiş.

* Advance Api 32 kütüphanesinin export ettiği fonksiyonların 67 listesine ulaşılabilimekte.

68* iTunesSetup 'ının unicode formatında PE header bilgisi.

* Burada bir senaryo verilmiş. Senaryoya göre sizin elinizde bir program var ve siz bu programın zararlı bir yazılım olup olmadığını süpheleniyorsunuz. Bu yazılımı statik analiz yöntemi ile incelemeye başlıyorsunuz.

69

Tabloda potansiyel bir keylogger programının kullanabileceği DLL'ler ve o DLL içerisinde bulunan tehlikeli fonksiyonlar listelenmiş.

70

Ortalama boyutlara sahip bir program birçok sayıda fonksiyon import etmiş olabilir ama bunlardan çok azı malware'ların sıkılıkla kullandıkları fonksiyonlardır. Eğer bir fonksiyonun ne yaptığı noktasında emin değilseniz MSDN kütüphanesine bakabilirsiniz.

Yeni bir zararlı yazılım analisti ilk analizlerinde fonksiyonları incelerken çok fazla zaman kaybedebilir ancak tecrübe kazandıkça hangi fonksiyonların daha tehlikeli olduğunu daha hızlı anlayacaktır.

Senaryomuza geri dönecek olursak biz elimizdeki programın zararlı bir yazılım olup olmadığını bilmiyoruz. PE başlığının bakarak hangi DLL ve fonksiyonları import etmiş onu inceleyip bir karara varacağız.

Program Kernel32.dll import etmiş dolayısı ile bu elimizdeki program başka programları açabilir ve değiştirebilir. Sistem üzerindeki dosyaları açabilir ve değiştirebilir. Tehlikeli fonksiyonları FindFirstFileW, FindNextFileW, GetCurrentProcess, GetProcessHeap, OpenProcess, ReadFile WriteFile.

Program User32.dll import etmiş. Bu kütüphane çok sayıda GUI fonksiyonu içerir. Dolayısı ile söz konusu bu programın bir GUI'si var. Program o GUI'yi bize göstermek zorunda değil. Visible özelliği False'tur. Tehlikeli fonksiyonları RegisterClassExW, RegisterHotKey, SetWindowTextW, SetWindowsHookExW, ShowWindow

→ SetWindowsHookEx fonksiyonunu genellikle spyware programları kullanırlar ve klavye girişlerine (keyboard inputs) erişirler bu fonksiyon yardımı ile. Amaç - iş uygumuna bakmak gereklidir.

Programın amacı ne yaptığı iş ne? Legal kullanım alanları var Ebeveyn denetim programları gibi.

→ RegisterHotKey fonksiyonu belirli bir tuş kombinasyonuna özel bir görev atamaya yarar. Kullanıcı ne zaman bir atanmış tuş kombinasyonuna bassa olay (event) tetiklenir. Uygulamanın o anki aktif uygulama olup olmamasının önemi yoktur. Uygulama bu olayı yakalar.

GDI32, grafikler ile ilgili işlemler yapmamızı sağlayan fonksiyonları içerir. Programın bir GUI'si olduğunu gösterir. Ancak bu kütüphaneyi legal programlar da sıkça kullandıkları için bu grup altındaki fonksiyonlar sizimize yarayacak bilgi vermeyebilir.

Shell32.dll kütüphanesi bir programa başka programları başlatma yetkisi verir. Ancak birçok legal program tarafından kullanılır.

Advapi32.dll kütüphanesindeki fonksiyonlar register'a ve register anahtarlarına erişimi sağlayacak fonksiyonları içerir. Örneğin register değerlerini değiştirecek bir programın Windows başlangıcında başlatılmasını sağlayabilirsiniz.

İste bu analizden elde edilen bilgiler ışığında bir programın amac-yaptığı iş ilişkisini kullanarak o programın zararlı bir yazılım olup olmadığını karar verebiliriz.

- * Bu listenin kısalığı bize bu programın paketlenmiş ve/veya gizlenmiş olduğunu göstermekte. Çünkü okunabilir string ifadeler yok. Ayrıca bir windows derleyicisi bu kadar az fonksiyon ile bir program derlemez. Hello World programında da hâlâ daha fazla import edilmiş fonksiyon vardır.

Bir programın paketlenmiş olması ne anlama gelir?

Bir programın paketlenmiş olması söz konusu o programın zararlı bir yazılım olduğu anlamına gelmez. Static Analiz Teknikleri ile o programı analiz edemeyeceğimiz anlamına gelir.

(24)

* Windows'taki Exe ve DLL dosyalarının tamamı PE dosyasıdır. PE dosyalarının header yani başlık kısımları bize dosya hakkında sadece import ve export edilen fonksiyonları bilmekten başka bilgiler de verir.

72 PE dosyalarında bir header sekmesi bulunur ve bu header alanında dosyanın metadata'sı bulunur. Yani çalıştırılabilir dosya hakkında bazı ek bilgiler bulunur. Bu ek bilgileri programın zararlı olup olmadığı noktasında yardımcı olabilirler,

73* PE header'ların birçok section var. Bu section'ların açıklamaları
74 şu şekilde: