“PDF Dosyası” ZARARLI YAZILIM ANALİZİ

Giriş

Siber saldırılar her geçen gün artıyor ve saldırganlar solucanlar ve kötü amaçlı yazılımlar yayarak hedeflerine zarar vermek için yeni teknikler buluyor. Her gün ortaya çıkan yenilikler ve yeni teknolojiler dünyasında, bir sisteme saldırma ve sistemdeki mevcut güvenlik açıklarından yararlanma imkanı yaratır. Kötü amaçlı yazılımın yayılması için kullanılan yöntemlerden biri, Taşınabilir Belge Biçimi (PDF) dosyalarıdır. Bu dosyaların esnek yapısı nedeniyle, saldırganlar için kötü amaçlı yazılımı PDF dosyalarına kolayca yerleştirmek için tatlı bir nokta haline geliyor. Bu raporda, PDF geliştirmenin sağladığı bu esnekliği ve kötü niyetli kişilerin virüsleri veya kötü amaçlı yazılımları PDF dosyalarına yerleştirmeyi neden bu kadar kolay bulduğunu anlayacağız. Ardından, kötü amaçlı dosyaları belirlemek ve ağınızdaki sistemlere zarar vermesini engellemek için python betiğini kullanarak nasıl yöntem ve teknikler geliştirebileceğimize bakacağız.

Bölüm 1

Son yıllarda internet üzerinden verilen hizmetlerde büyük bir artış olmuştur. Akıllı telefonların ve güçlü bilgisayarların kullanımıyla, herhangi bir görevi yapma gücü kolayca elde edilebilir. E-posta kullanıma sunulduğunda, insanlar dünyanın herhangi bir yerine mesaj göndermeye başlamakla kalmadı, aynı zamanda e-posta üzerinden dosya paylaşımı gibi hizmetleri de etkinleştirdi. Fiziksel cihazlardaki bellek ihtiyacı karşılayamadığında, internet erişimi olan her yerde, herkese sınırsız depolama olanağı sağlayan bulut depolama kullanılmaya başlandı.

Ancak bu teknolojik gelişmeler beraberinde ciddi dezavantajları da getirdi. İnternette mevcut olan veri miktarı, bir kişinin kişisel bilgilerini sızdırabilecek ve ona ciddi zarar verebilecek bir siber saldırının olası bir hedefi olabilir. Siber suç artıyor ve karaborsadaki kötü aktörler için

karlı bir faaliyet haline geldi. Örneğin, son zamanlarda hükümetlere, özel sektöre ve sağlık sektörüne zarar veren çok sayıda kötü amaçlı yazılım saldırısı gördük. Bu saldırganlar, web üzerinden sürdürülen bu kötü amaçlı yazılımların gelişmesine ve evrimine yol açan ekonomik olarak motive olurlar.

Taşınabilir Belge Biçimi (PDF) dosyaları, kötü amaçlı yazılımların internet üzerinden yayılmasını durdurmak için ek bir zorluk haline geldi. Gömülü JavaScript ve ActionScript kodlarıyla farklı içerik türlerini dahil etme yeteneği, onu kötü amaçlı yazılım saldırıları için kolay bir hedef haline getirir. 1990'ların başında PDF belge türünü ortaya atan Adobe, bu dosyaları derleyen acrobat okuyucu yazılımındaki güvenlik açıkları ile PDF belgelerindeki bazı kritik güvenlik açıklarını hala kamuoyuna açıklamakta ve araştırmada belirtildiği gibi kullanıcıları için ciddi potansiyel tehditler bırakmaktadır. çalışma [1], [2].

PDF formatının sağladığı esneklik nedeniyle, kötü niyetli PDF dosyalarını tespit etmek zor olmuştur ve çoğu zaman bu dosyaları bulmak ve aramak için adli analistleri sıkıntıya sokar. Makine öğrenimi, adli soruşturma ve siber güvenlikte bu tür karmaşık görevlerin üstesinden gelmek için yeni bir araç olarak görülmüştür [3]. Ancak verimli bir makine öğrenimi modeli geliştirmek için, dosyaları kötü amaçlı veya zararsız olarak sınıflandırmaya yardımcı olacak özelliklerin belirlenmesi her şeyden önce önemlidir. Makine öğrenimi başlangıçta bu tür araştırmalar için geliştirilmemiştir ve bu nedenle, saldırganların eylemleri için kullanılabilecek bazı güvenlik açıkları bırakır. Bununla birlikte, makine öğrenimi, adli analiste fayda sağlayabilir ve algılama sürecini büyük ölçüde etkileyebilir ve bu saldırıların engellenmesine yardımcı olabilir.

Bu raporda, PDF dosyasının ne olduğunu göreceğiz. Kötü amaçlı yazılımları bir PDF belgesine yerleştirmek neden kolaydır? Nasıl bu kadar etkili? Ve PDF belgeleri kullanılarak yapılan saldırıları nasıl durdurabiliriz?

Bu raporda, Bölüm II, PDF belgesinin yapısını araştırır ve PDF belgesinin nasıl kodlandığını anlama konusunda fikir verir. Bölüm III'te, PDF belgelerinde görülen kötü amaçlı yazılımları ve bunların PDF okuyucularındaki güvenlik açıklarından yararlanarak sistemleri nasıl kullandıklarını keşfedeceğiz. Bölüm IV, kötü amaçlı PDF belgelerini bulmak için kullanılabilecek çeşitli teknikler ve yöntemlerden bahseder. Şekil 1, raporun organizasyonunu ve kapsayacağı konuları göstermektedir.

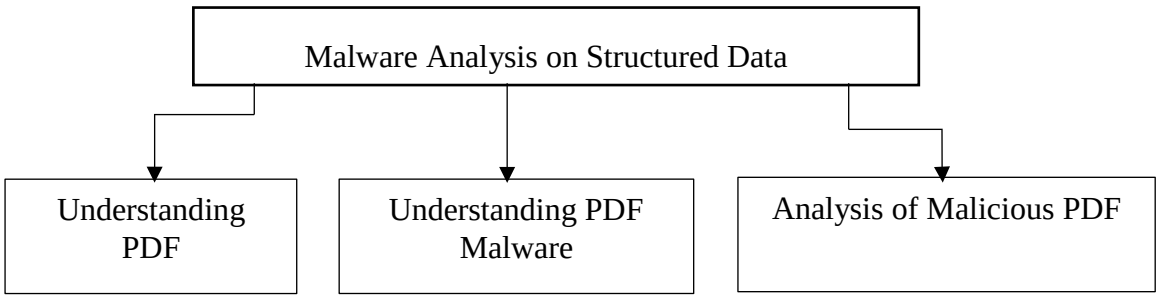


Fig 1. Organization of the report

Bölüm 2

PDF'yi Anlamak

Bu bölümde, PDF dosyası hakkında konuşacağız ve bir kötü amaçlı yazılımın dosyalara nasıl gömüldüğünü veya verilen PDF dosya yapısındaki anormallikleri nasıl tanımlayabileceğimizi anlamak için bir adım taşı olan yapısını anlayacağız.

PDF, 1993 yılında Adobe Systems tarafından icat edildi. Metinleri, resimleri, köprü metni bağlantılarını, yer imlerini ve küçük resimleri destekleyerek bunları orijinal biçimlerinde ve görünümlerinde koruyordu. PDF'nin 1994 yılındaki ilk revizyonu, belgelere şifre eklemeyi ve paylaşılabilen veri türündeki esnekliği ile güvenlik özelliği eklemek için şifrelemeyi destekledi. Yıllar geçtikçe, PDF biçimlerinin kullanımında büyük bir artış gördük. Bugün, PDF'de oluşturulabilen 3B resimleri, düzenlenebilir formları ve animasyonları desteklemektedir. Daha iyi veri güvenliği ve bütünlüğü için AES şifrelemeleri de tanıtıldı. Ayrıca metni görselleştirmek için JavaScript ve ActionScript gibi teknolojileri de destekler.

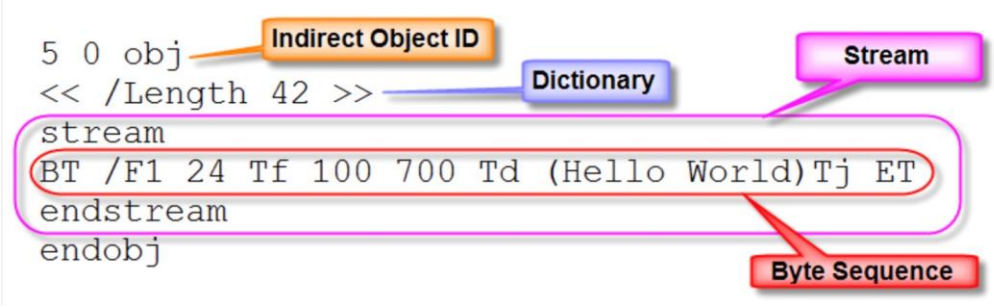
Şimdi bir PDF belgesinin sözdizimini anlayalım.

• Nesneler: Belge, farklı türde veri nesnelerinin bir koleksiyonundan oluşturulan bir veri yapısıdır.

Bu nesneleri yazmak için kullanılan iki tür kural vardır:

- Sözcük Kuralları: Sözdizimsel öğeleri ve nesneleri yazmak için kullanılan karakter kümesi.
- Akış Nesneleri: Karmaşık veri türü olan akış nesnelerinin yazılmasını açıklar. Bir sözlükten ve içinde bayt dizisinin bulunduğu **akış** ve **son akış anahtar sözcüklerinden oluşan bir bayt dizisidir.** Akış nesneleri dolaylı nesnelerdir. Figür

Aşağıdaki 2, bir PDF akış nesnesinin nasıl temsil edildiğine ve içerdiği çeşitli bileşenlere ilişkin bir örneği gösterir.

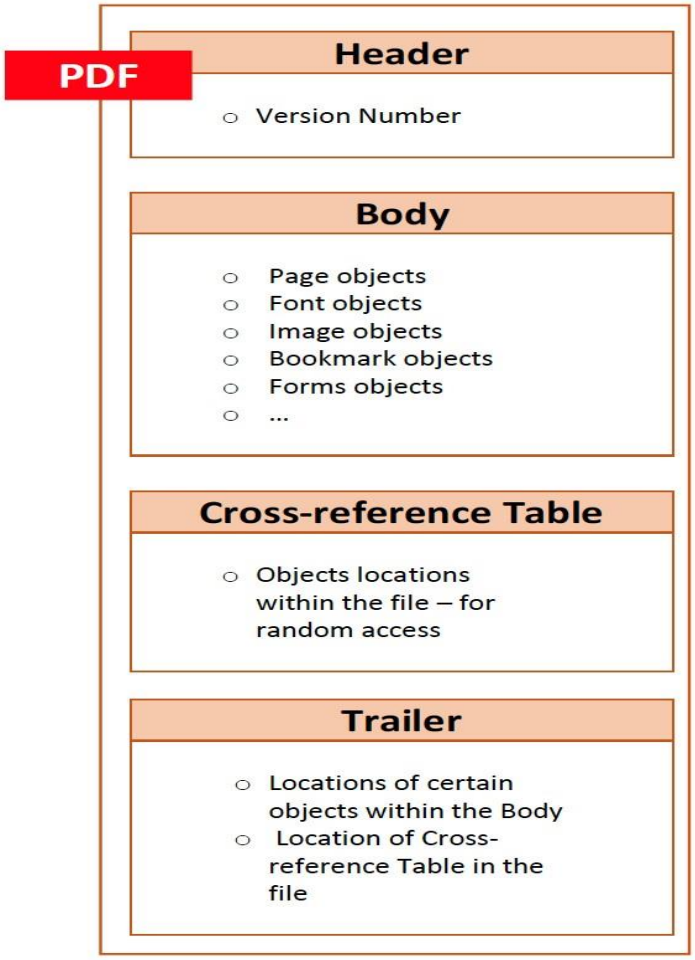


Şekil 2. PDF Akış Nesnesi Örneği [27]

- Dosya Yapısı: PDF belgesindeki nesnelerin nasıl saklandığını, erişildiğini ve güncellendiğini belirlemeye yardımcı olur.
    - Bir PDF belgesi, ilişkili yapısal bilgilerle birlikte tek bir bağımsız bayt dizisini içeren bir nesneler topluluğudur. [8]'de Samir G. Sayed ve Mohamed Shawkey bir PDF belgesinin yapısı hakkında konuşuyorlar. Şekil 3. bir PDF belgesinin tipik yapısını göstermektedir. Bir PDF dosyası, başlık, nesneler, çapraz referans (xref) ve fragman olmak üzere dört bileşenden oluşur.
    - Başlık, PDF dilinin sürümü hakkında bilgi sağlar. Başlık, bir PDF dosyasının başına eklenir. Bir PDF oluşturucu, başlığı eksikse dosyayı yok sayar.
    - Bir veya daha fazla nesneden oluşan gövde. Dizeler, sayılar, sözlükler, boole ve akışlar gibi çeşitli nesne türleri vardır. Nesneler, yazı tiplerini, grafikleri, sayfaları ve JavaScript ve Acrobat formları gibi gömülü kodları içeren bilgilere sahiptir. Sözlük nesnesi, adlar, diziler ve sayılar gibi daha basit nesne türlerini içerir. Akışlar, resimler, multimedya, yazı tipleri ve JavaScript gibi büyük miktarda gömülü veriyi depolamak için kullanılır. İyi huylu PDF dosyalarındaki akışlar normalde metin, yazı tipleri ve resimler içerir. Ancak, kötü amaçlı PDF dosyalarındaki akışlar genellikle JavaScript

içerir. Saldırganlar , kötü amaçlı JavaScript kodlarını gizlemek için sıkıştırılmış akışlar kullanır, aksi takdirde bu kötü amaçlı kodlar düz metin olur ve kolayca görünür olur.

- Çapraz referans (xref) tablosu, PDF dosyasındaki tüm nesnelerin ve bu nesnelerin başlangıcına kadar olan uzaklıkların listesini içerir. PDF dosyasının fragmanı, çapraz referans tablosunun ve nesnelerin konumuna göre ofset hakkında bilgi sağlar.



Şekil 3. PDF dosya yapısı [28]

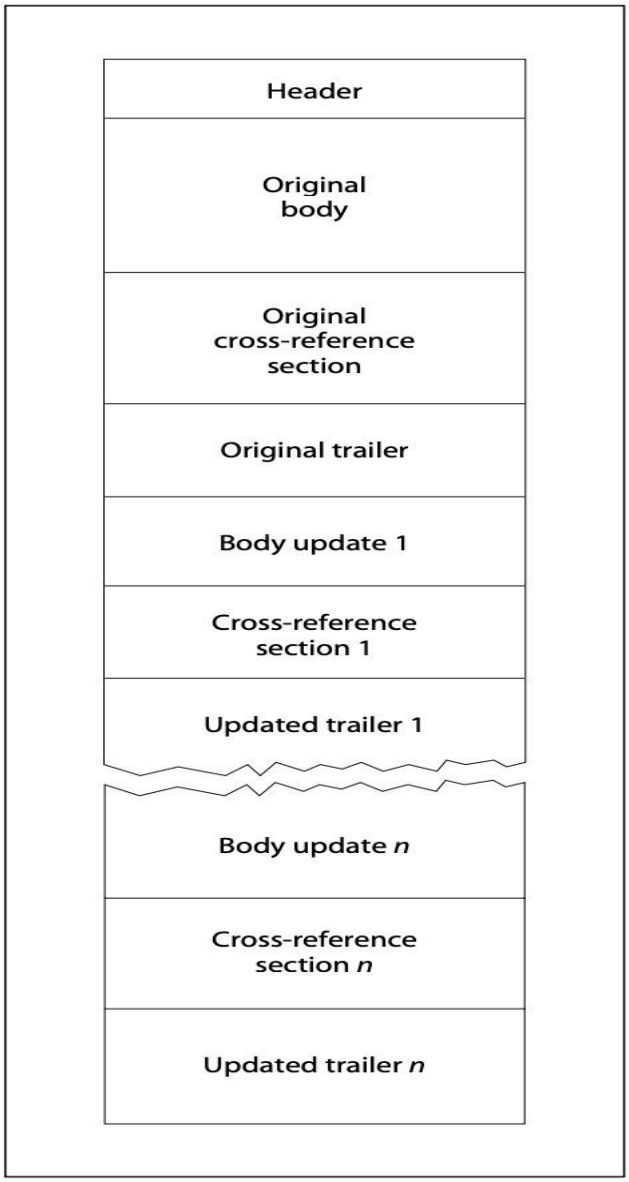
- */Root'u* bularak başlar ve ardından dolaylı nesneyi oluşturur ve verileri açmaya devam eder.

Şekil 4. oluşturulan bir PDF belgesinin nasıl göründüğünün tipik bir örneğini göstermektedir.

```
%PDF-1.1 Header
1 0 obj Top-level of page tree: has two children—page one and an intermediate page tree node
<< /Kids [2 0 R 3 0 R] /Type /Pages /Count 3 >>
endobj
4 0 obj Contents stream for page one
<< >>
stream
1. 0.000000 0.000000 1. 50. 770. cm BT /F0 36. Tf (Page One) Tj ET
endstream
endobj
2 0 obj Page one
<<
  /Rotate 0
  /Parent 1 0 R
  /Resources
    << /Font << /F0 << /BaseFont /Times-Italic /Subtype /Type1 /Type /Font >> >> >>
  /MediaBox [0.000000 0.000000 595.275590551 841.88976378]
  /Type /Page
  /Contents [4 0 R]
>>
endobj
5 0 obj Document catalog
<< /PageLayout /TwoColumnLeft /Pages 1 0 R /Type /Catalog >>
endobj
10 0 obj Document information dictionary
<<
  /Title (PDF Explained Example)
  /Author (John Whitington)
  /Producer (Manually Created)
  /ModDate (D:20110313002346Z)
  /CreationDate (D:2011)
>>
endobj xref
0 11
trailer Trailer dictionary
<<
  /Info 10 0 R
  /Root 5 0 R
  /Size 11
  /ID [<75ff22189ceac848dfa2afec93deee03> <75ff22189ceac848dfa2afec93deee03>]
>>
startxref
0
%%EOF
```

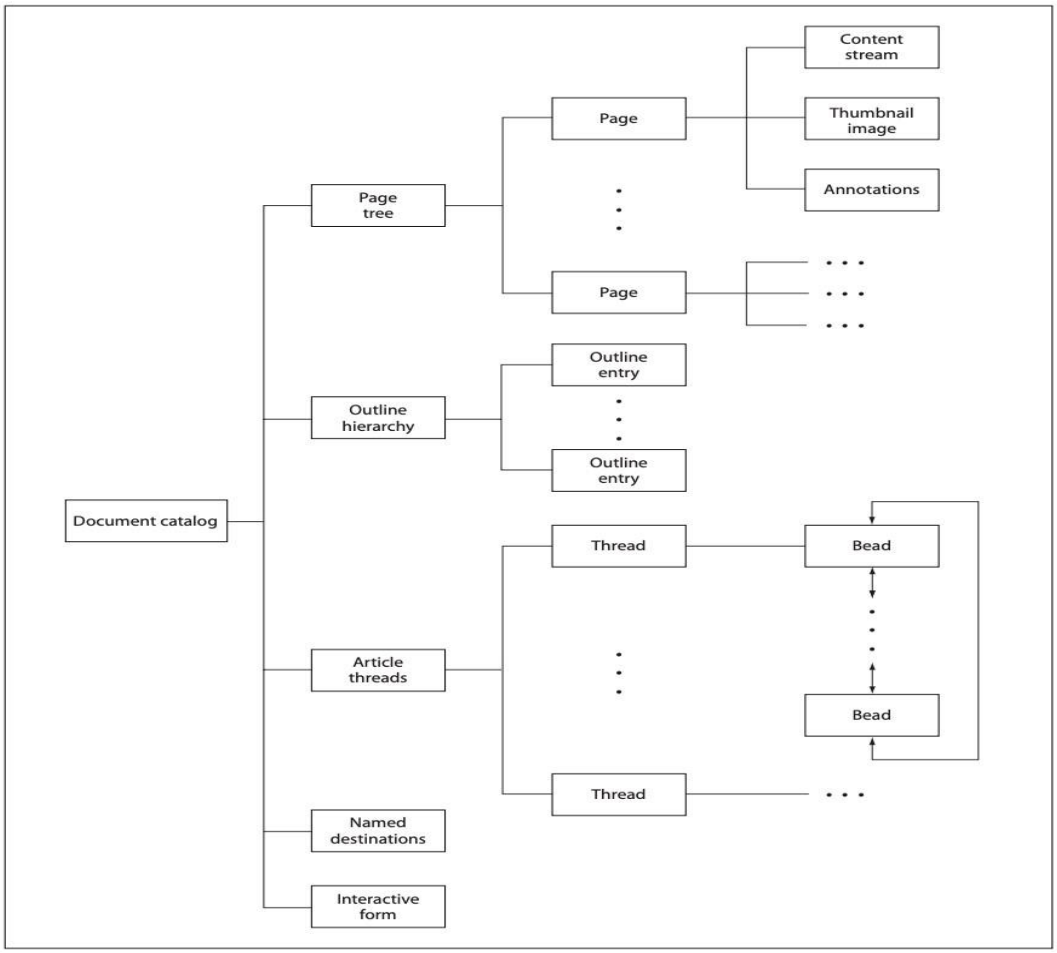
Şekil 4. Bir PDF belgesi örneği

- Artımlı Güncellemeler: Bir PDF dosyası, dosyanın sonuna yeni içerik eklenerek ve orijinal içerik olduğu gibi tutularak güncellenebilir. Bir dosya güncellendiğinde, yalnızca yeni eklenen nesnelerin veya değiştirilen veya silinen nesnelerin bilgilerini içeren bir çapraz referans bölümü de eklenir.



Şekil 5. Güncellenmiş bir PDF dosyasının yapısı. [29]

• Belge Yapısı: Bir PDF belgesinin temel nesne türlerinin ve bileşenlerinin temsilini tanımlar.



Şekil 6. Bir PDF belgesinin Ağaç Yapısı [29]

Ö Belgenin kökü, belgenin fragman bölümünde bulunur. Kök, içerik, makale dizileri, ana hatlar ve adlandırılmış hedefler hakkındaki bilgileri tanımlayan bir katalog sözlüğüdür. Ayrıca, verilerin belgede nasıl görüneceğine ilişkin bilgileri de içerir. Belge, tüm içeriğin korunmasında bir sayfa ağacı yapısını takip eder ve bu da çeşitli uygulamaların 1000'lerce sayfa arasında hızla seyahat etmesine ve içeriği görüntülenmesine olanak tanır. Sayfa ağacı düğümleri olarak adlandırılan ara düğümler ve sayfa nesneleri olarak yaprak düğümler, ağaç yapısındaki iki farklı nesne türüdür. akrobat

Distiller ve PDF Writer programları, dengeli ağaçlar olarak bilinen ağaçları oluşturur.

Bölüm 3

PDF Kötü Amaçlı Yazılımını Anlama

Son bölümde gördüğümüz gibi, bir PDF belgesi, bir kullanıcının ihtiyaç duyduğu bilgi ve içeriği paylaşmanın uygun bir yolunu sağlamak için farklı türde nesnelerin gömülmesine izin verir. Bu seçenekler sayesinde, saldırganların PDF'ye kötü amaçlı kodlar veya kötü amaçlı yazılımlar gömerek belgenin bu özelliklerinden yararlanma olanağı sağlar. Belgenin kendisindeki bu esneklikle, saldırganların bu belgeyi okuyan yazılımdan yararlanmaları daha kolay hale gelir. Davide

Maiorca ve Battista Biggio , bir PDF belgesine yerleştirilmiş kötü amaçlı yazılım örneklerinden bahsediyor. Adobe okuyucu için yıllara dayanan Ortak Güvenlik Açıkları ve Etkilenmeler (CVE) raporu, saldırganların belgedeki kötü amaçlı kodları bulaştırarak yararlanabilecekleri yazılımın güvenlik açığı raporunu gösterir. Şekil 7, Adobe Reader için yıllar boyunca CVE raporunu göstermektedir ve farklı türdeki güvenlik açıklarından yararlanılabilir.

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score
1	CVE-2018-19722	125			2019-01-18	2019-01-23	5.0
Adobe Acrobat and Reader versions 2018.011.20063 and earlier, 2017.011.30102 and earlier, and 2015.006.30452 information disclosure.							
2	CVE-2018-19719	125			2019-01-18	2019-01-22	4.3
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have an out-of-bounds read vulnerability. Successful exploitation could							
3	CVE-2018-19717	125			2019-01-18	2019-01-23	4.3
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have an out-of-bounds read vulnerability. Successful exploitation could							
4	CVE-2018-19716	119		Exec Code Overflow	2019-01-18	2019-01-23	7.5
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have a heap overflow vulnerability. Successful exploitation could lead t							
5	CVE-2018-19715	416		Exec Code	2019-01-18	2019-01-23	10.0
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have a use after free vulnerability. Successful exploitation could lead tc							
6	CVE-2018-19714	125			2019-01-18	2019-01-22	4.3
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have an out-of-bounds read vulnerability. Successful exploitation could							
7	CVE-2018-19713	416		Exec Code	2019-01-18	2019-01-23	9.3
Adobe Acrobat and Reader versions 2019.008.20081 and earlier, 2019.008.20080 and earlier, 2019.008.20081 and and earlier, and 2015.006.30456 and earlier have a use after free vulnerability. Successful exploitation could lead tc							

Şekil 7. PDF Belgesinde Bilinen CVE

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	1		1	1											
2000	1		1	1											
2001	1														
2002	1														
2003	3		2	1											
2004	6		5	4											
2005	9	4	5	3											
2006	7	2	3		1							2			
2007	9	3	3		1		2		1				1		1
2008	17	2	12	4	2							1			
2009	45	15	36	20	13					1		1			3
2010	68	35	60	33	29		2			3		1			4
2011	60	21	48	33	17		3			2		6			1
2012	30	24	30	24	23					1					
2013	66	30	60	49	30					3	1	1			
2014	44	17	35	17	17		1			5	4				
2015	137	29	61	39	24					61	20				
2016	21	11	18	11	11					1		2			
2017	130		82	54	56					6	35				
2018	44		16	4	1					1	1				
Total	700	193	478	298	225		8		1	84	61	14	1		9
% Of All		27.6	68.3	42.6	32.1	0.0	1.1	0.0	0.1	12.0	8.7	2.0	0.1	0.0	

Şekil 8. Adobe okuyucu için CVE raporu

JavaScript kodu, kodlanmış akışlar ve gömülü nesneler, bu güvenlik açıklarından yararlanmak için saldırganların ilk tercihleridir. Belge okuyucu tarafından açılır açılmaz bir uzak kodun yürütülmesine veya uzak ağ bağlantıları oluşturulmasına izin verir. istismar eden örneklerden biri Adobe okuyucu, uzak bir sunucudan kötü amaçlı yazılım indirip yüklemeye izin verdi. Bu, yığının taşmasına ve bir uzak bağlantı kodu yürütmesine izin veren bir BMP/RLE yığın bozulması güvenlik açığıydı (CVE-2013-2729). Adobe Reader'daki bir başka güvenlik açığı örneği, uzaktaki saldırganların (CVE-2010-4091) içinde verilen hedef sistemde rasgele bir kod veya hizmet reddi (DoS) saldırısı yürütmesine izin veren EScript.api ile ilgilidir.

[3]'teki Davide Maiorca ve Battista Biggio ve [10]'daki Soon Heng Tan Mavric ve Chai Kiat Yeo, belgeye gömülü Javascript koduyla ilgili örnekler verdiler. CVE-2010-1240, yalnızca PDF belgesi

açıldığında ikili kod yürüten bir güvenlik açığıdır [3]. Bu, 2010 yılında Contagio tarafından rapor edilmiştir. Gömülü kod örneği aşağıdaki gibidir [3]: <<

/Tür /Eylem

/S /Başlat

/Kazanç

<<

/F (cmd.exe)

/P (/c echo BinaryStream Dim > vbs1.vbs && echo Set BinaryStream =

CreateObject("ADODB.Stream")

>>

....

Endobj

Okuyucu belgeyi açtığında, dosya bir Visual Basic betiği olan Windows Kabuğunda bir komut yürütür. Bu güvenlik açığı, genellikle tuş vuruşu günlüğü ile bankacılık kayıtları gibi özel bilgileri çalmak için kullanılan bir Truva atı kötü amaçlı yazılımı olan Zeus Truva Atı'nı yürütmek için kullanıldı.

Yukarıda belirtilen örnekler, yalnızca PDF belge platformu üzerinden yararlanabilen ve bu belgeleri kullanıcı tarafından okunabilirlik amacıyla oluşturmak için kullanılan okuyucuları açığa çıkarabilen bir dizi kötü amaçlı etkinlik ve güvenlik açığı oluşturur. Böylece, bu saldırılar çok karmaşık ve karmaşık hale gelebilir. Bunları her seferinde belirlemek ve bulmak kritik bir iştir.

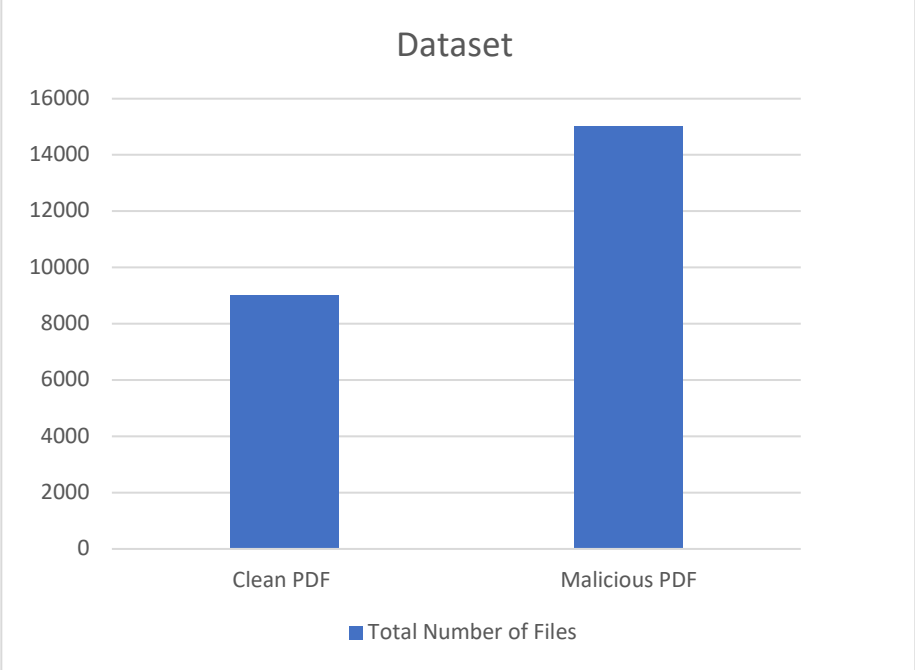
Riskli olabilecek ve hasara neden olabilecek bazı PDF etiketlerine bakalım:

- /OpenAction ve /AA: Bir görevi veya eylemi otomatik olarak yürütür.
- /JavaScript ve /JS, bir komut dosyasını çalıştırabilen veya bir arka kapı açabilen çalıştırılacak komut dosyasını belirtir.
- /GoTo, görünümü belge içindeki belirli bir hedefe değiştirir. Başka bir belgeye de işaret edebilir.
- /Launch, bir belgeyi açmak veya bir programı başlatmak/başlatmak için kullanılabilir.
- /URI bir URL'yi getirecek ve erişecek.
- /SubmitForm ve /GoToR, verileri verilen url'ye gönderir.
- /RichMedia: Flash'ı belgeye gömer.
- /ObjStm: Bir Nesne Akışını gizler.

Deneyler ve Sonuçlar:

Kötü Amaçlı PDF Analizi

PDF belgesine kod gömerek saldırıların nasıl mümkün olabileceğini gördük. Bu, okuyucu ve hedef sistemlerden yararlanmaya yardımcı olur. Bunu önlemek için, kötü amaçlı PDF belgelerini kötü amaçlı olmayanlardan ayırt etmek için kullanılabilecek çeşitli analiz yöntemleri vardır. Bu süreçleri uygulamak ve zaman tüketmek zor olabilir, ancak artan saldırılar ve sonuçları ile bu zamana yatırım yapmak ve saldırganlar tarafından üretilen belgeleri bulmak giderek daha önemli hale geliyor. İnternet üzerinden toplanan 9000 temiz PDF dosyası ve 15000 Kötü Amaçlı PDF dosyasından oluşan 24000'den fazla dosyadan oluşan bir veri kümesi topladık.



Şekil 9. Veri Seti

Pdf analizi için yaygın olarak kullanılan açık kaynak kodlu araçlardan birini inceleyelim.

• peepdf:

Belirli bir PDF dosyasını analiz etmeye yardımcı olan açık kaynaklı bir python aracıdır. Bir güvenlik araştırmacısının değerli bilgiler vermesine ve araştırmacının ihtiyaç duyduğu tüm gerekli nesneleri almasına yardımcı olabilir.

Kullanım:

```
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$ python peepdf.py
Usage: peepdf.py [options] PDF_file

Version: peepdf 0.3 r235

Options:
  -h, --help            show this help message and exit
  -i, --interactive     Sets console mode.
  -s SCRIPTFILE, --load-script=SCRIPTFILE
                        Loads the commands stored in the specified file and
                        execute them.
  -c, --check-vt        Checks the hash of the PDF file on VirusTotal.
  -f, --force-mode      Sets force parsing mode to ignore errors.
  -l, --loose-mode      Sets loose parsing mode to catch malformed objects.
  -m, --manual-analysis
                        Avoids automatic Javascript analysis. Useful with
                        eternal loops like heap spraying.
  -u, --update          Updates peepdf with the latest files from the
                        repository.
  -g, --grinch-mode     Avoids colorized output in the interactive console.
  -v, --version         Shows program's version number.
  -x, --xml             Shows the document information in XML format.
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$
```

Şekil 10. Peepdf aracının kullanımı

Bir araştırmacının derinlere inmesine ve dosyanın yapısını ve içeriğini anlamasına yardımcı olabilecek birden çok seçenek sunar.

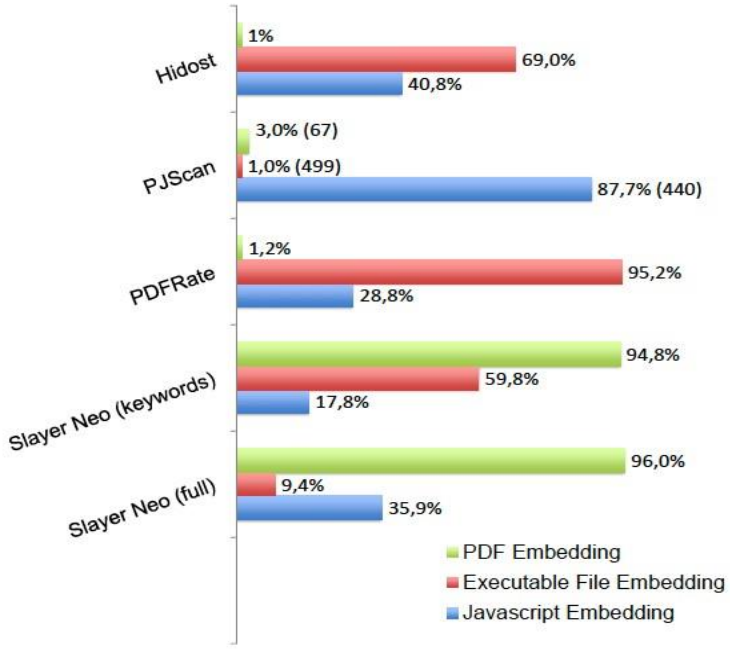
```
Shubhams-MacBook-Pro:peepdf_0.3 pachputeshubham$ python peepdf.py -i
PPDF> help

Documented commands (type help <topic>):
========================================
bytes           exit         js_join           quit          set
changelog       filters      js_unescape       rawobject     show
create          hash         js_vars           rawstream     stream
decode          help         log               references    tree
decrypt         info         malformed_output  replace       vtcheck
embed           js_analyse   metadata          reset         xor
encode          js_beautify  modify            save          xor_search
encode_strings  js_code      object            save_version
encrypt         js_eval      offsets           sctest
errors          js_jjdecode  open              search

PPDF>
```

Şekil 11. Peepdf tarafından sağlanan araçlar

```
PPDF> metadata

Info Object in version 0:

<< /ModDate D:20091004134555
/CreationDate D:20091004134555
/Producer Scribus PDF Library 1.3.5svn
/Creator Scribus 1.3.5svn
/Title
/Trapped /False
/Keywords
/Author  >>

PPDF>
```

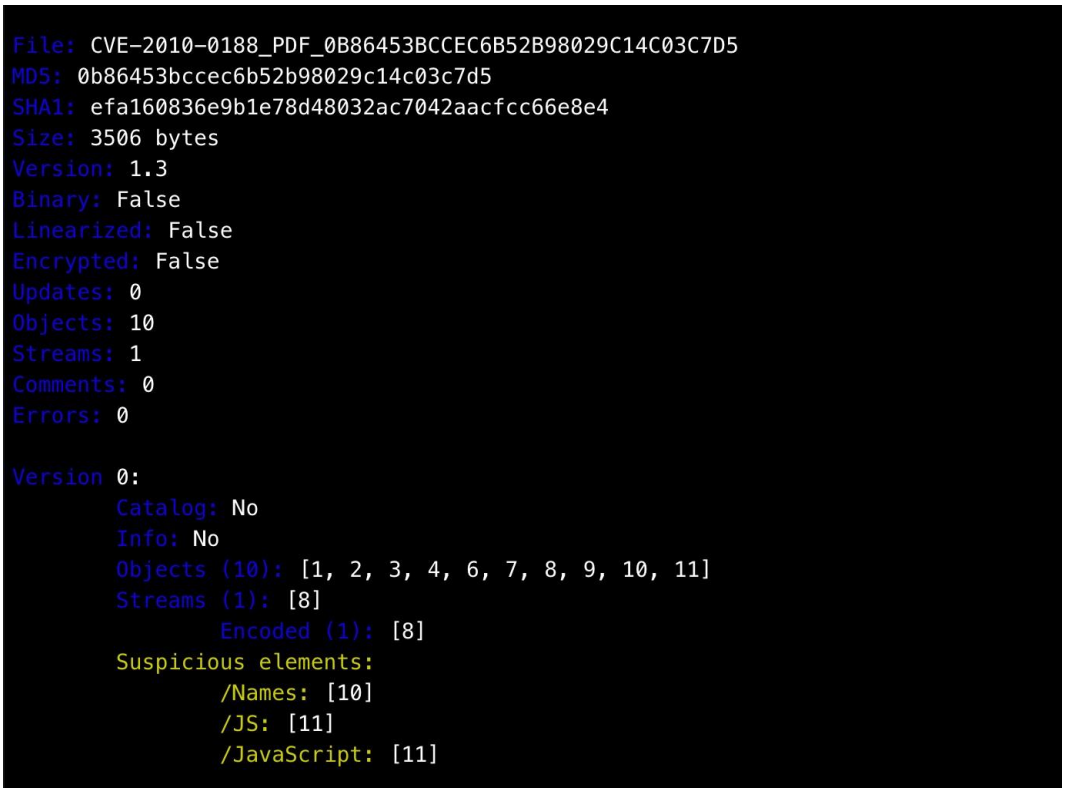
Şekil 12. PDF Belgesinin Meta Verileri

Bu kötü amaçlı belgeleri keşfetme süreci, adli analiz veya öğrenmeye dayalı algılama olabilir.



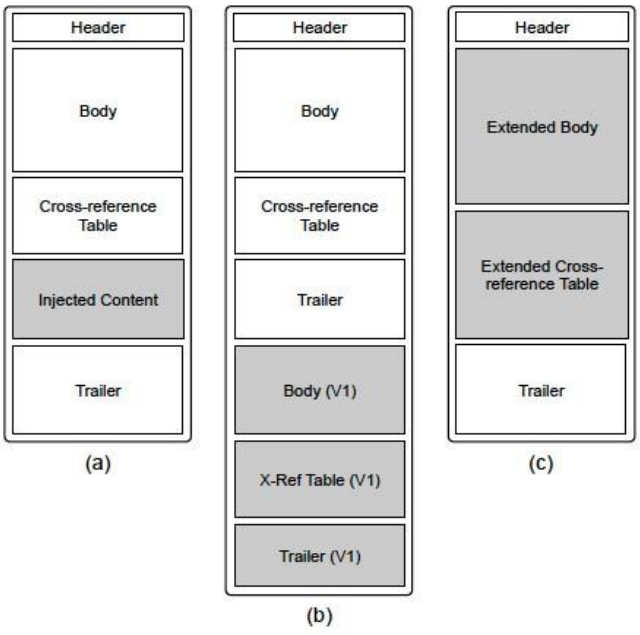
Şekil 13. PDF kötü amaçlı yazılım algılama oranı

S. Pai [5]'te adli analizden, bir belgeyi kötü amaçlı olarak sınıflandırabilecek önemli yönleri belirlemek için veya belgenin kaynağına odaklanarak makine öğrenimi tekniklerini kullanabileceğimiz bir yöntem olarak bahseder. Kaynak bir spam e-posta veya kimlik avı web sitesiyse, içerik analiz edilebilir. Daha fazla analiz, kötü amaçlı kod tarafından gerçekleştirilen eylemleri ve bunların sistem veya veriler üzerindeki sonuçlarını keşfedecektir.



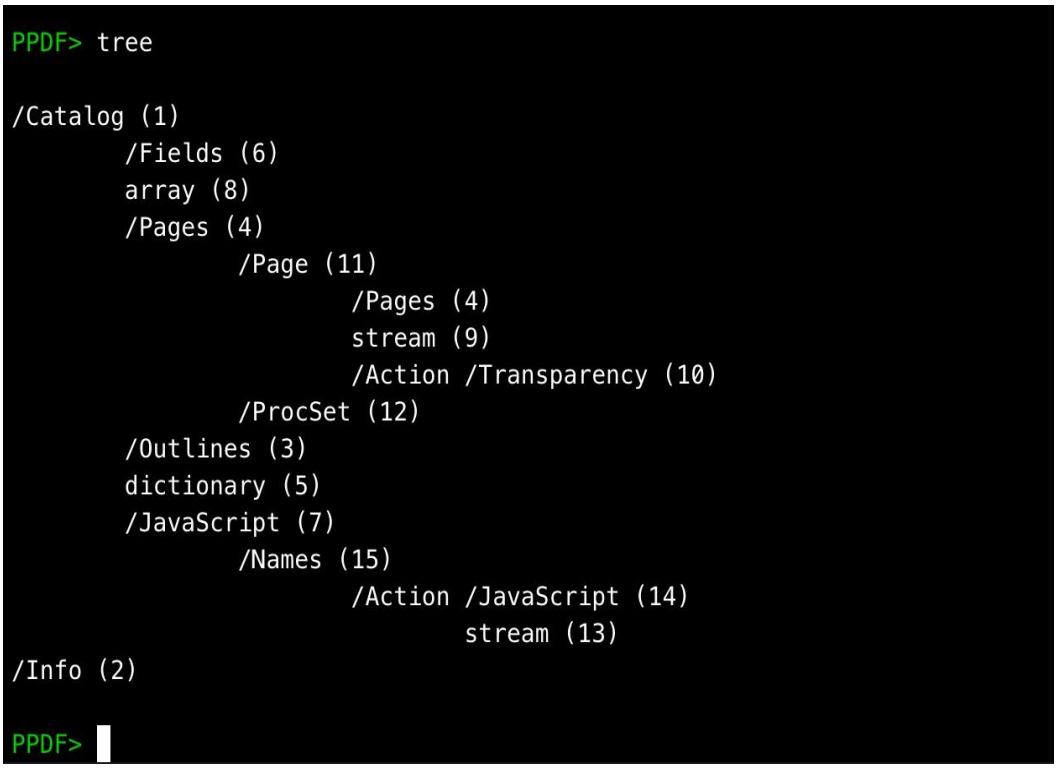
Şekil 14. PDF belgesinin analizi

A. Damodaran [6]'da anahtar kelime tabanlı analizden bahsediyor. Bu, belgedeki anahtar kelime nesnelerinin */JavaScript* veya */JS olarak bulunmasına bağlıdır.* Anahtar kelime bulunursa, analist akış nesnelerini daha fazla inceleyebilir ve gerçek işlevlerini anlamak için onları açabilir. Anahtar kelime bulunamazsa, belgenin kullanıcının erişmesi için güvenli olduğu kabul edilebilir.



Şekil 15. PDF dosyalarına içerik enjeksiyonu

Diğer bir yöntem, PDF belgesinin nesneleri arasındaki bağlantıları gösteren bir ağacın oluşturulduğu Ağaç tabanlı analizdir. Sistem , ağacı oluşturmaya başladığı yerden */Root anahtar sözcüklerini* bulmak için PDF belgesinin römork nesnesiyle başlar [6]. Sistem daha sonra her nesneyi inceler. Kötü amaçlı PDF dosyalarının çoğu, şüpheli eylemler içeren nesnelerle sona erer.



Şekil 16. PDF'nin Ağaç Yapısı

Kod tabanlı Analiz, gömülü kodu incelemeye, yani belgedeki /JavaScript kodunu bulmaya ve hangi güvenlik açığından yararlanabileceğini (bilinen veya bilinmeyen) belirlemeye dayanır. Amaç, dosyaların iç derinliğine girmeden dosyalara gömülü olan komut dosyası kodunu bulmaktır. Bu, kod satırlarının bilinen güvenlik açıklarıyla eşleşip eşleşmediğini bulmaya ve böylece dosyanın kötü niyetli olup olmadığını belirlemeye yardımcı olur.

```
PPDF> rawobject 7

<< /JavaScript 15 0 R >>

PPDF> rawobject 5

<< >>

PPDF> rawobject 3

<< /Type /Outlines
/Count 0 >>

PPDF> rawobject 1

<< /AcroForm 6 0 R
/Threads 8 0 R
/ViewerPreferences << /PageDirection /L2R >>
/OpenAction << /S /JavaScript
/JS (this.STARTSCRIPT\(\)) >>
/Pages 4 0 R
/Outlines 3 0 R
/Type /Catalog
/PageLayout /SinglePage
/Dests 5 0 R
/Names 7 0 R >>
```

Şekil 17. PDF Belgesindeki Nesneler

25000'den fazla dosyalık bir veri setine sahip PDF dosyalarının analizi sırasında bulunan özellikler kullanılarak, PDF dosyasını okuyarak özelliklerin tanımlanması ve kötü amaçlı olup olmadığının belirlenmesi görevinin gerçekleştirildiğini kontrol etmek için bir araç oluşturuldu.

Bir PDF belgesi ayrıca JavaScript ve URL'ler gibi öğeler içerir. Zararlı URL sorununu çözmek için mantıksal regresyon kullanılarak bir ML modeli kullanıldı. Lojistik regresyon, verileri tanımlayan ve bir veya daha fazla nominal, sıra, aralık veya oran düzeyinde bağımsız özniteliğe bağlı olan ikili öznitelikler için çıkarım ilişkisi analizi çizen bir tahmine dayalı analiz modelidir. Bu algoritmanın uygulanması, çok değişkenli bağımlı sınıfları işleme özelliği nedeniyle yapıldı.

Modelin performansı, geri çağırma ve ROC altındaki alan gibi çeşitli değerlendirme parametrelerini test etmek için karışıklık matrisi kullanılarak değerlendirilir. Elde edilen sonuçların sonuçları ve görsel temsilleri aşağıdadır.

Sonuçlar:

```
Gerçek Olumlu: 10780

Gerçek Negatif: 10835

Yanlış Pozitif: 1

Yanlış Negatif: 664

Toplam: 22280


Doğruluk: 97.0153

F-Skoru: 97.0078741

Geri Çağırma: 94.1978

Hassasiyet: 99.9907
```

Performans sonuçları aşağıdaki formüller kullanılarak hesaplanır:

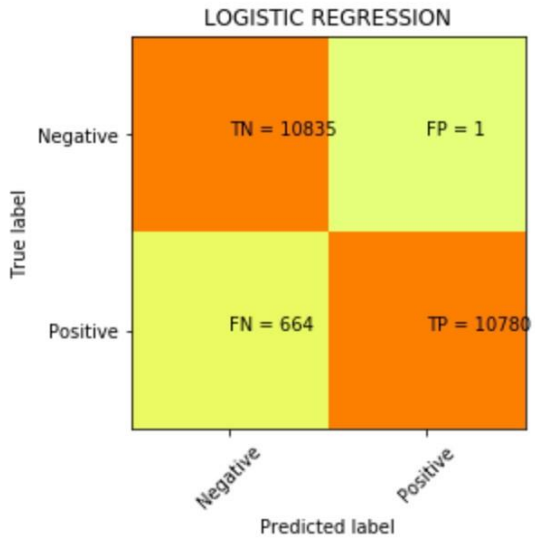
Doğruluk = (TP + TN) / TP + FN + FP + TN

F-Skoru = 2 ∗(P ∗R) / (P + R)

Geri Çağırma = TP / (TP + FN)

Hassasiyet = TP / (TP + FP)

Karışıklık matrisi tablosundan elde edilen Değerlendirme Metrik sonuçları, Doğruluk ve F-ölçüsünün sınıflandırıcı performansının değerlendirilmesinde kullanılan metrikler olduğunu göstermektedir. F ölçüsü, Geri Çağırma (R) ve Kesinlik (P) açısından tanımlanır. Değerlendirme metrikleri daha yüksek değere sahipse, sınıflandırıcı veri seti için en uygun olanıdır. Karışıklık matrisi tarafından etkin bir şekilde açıklanan değerlendirme metrikleri sonuçları şunlardır:



Şekil 18. Karışıklık Matrisi

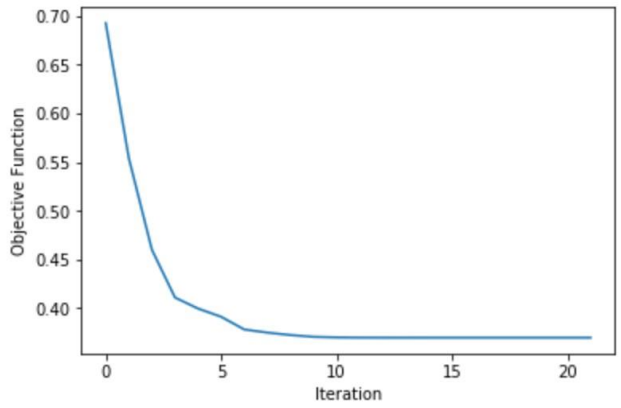
Karışıklık matrisinin Şekil 18'deki girişleri, bağlamımızda aşağıdaki anlamlara sahiptir:

Gerçek Negatif (TN): Kötü Amaçlı URL örnekleri Kötü Amaçlı olarak etiketlenir.

Yanlış Pozitif (FP): Kötü amaçlı URL örnekleri, Meşru(kötü amaçlı olmayan) olarak etiketlenir.

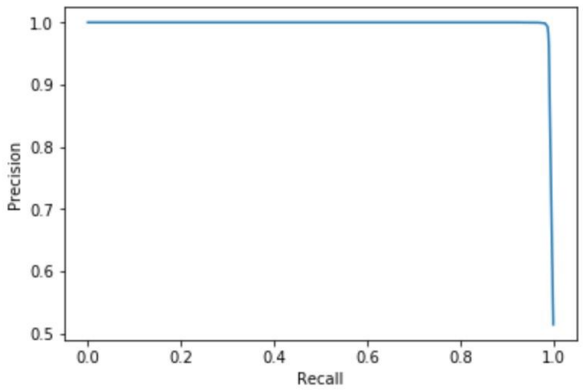
Yanlış Negatif (FN): Meşru(kötü amaçlı olmayan) URL örnekleri Kötü Amaçlı olarak etiketlenir.

True Positive (TP): Meşru(kötü amaçlı olmayan) URL örnekleri, Meşru(zararlı olmayan) olarak etiketlenir.



Şekil 19. Yakınsama Eğrisi

Şekil 19 , URL veri kümesine uygulanan lojistik regresyon yöntemi için yineleme sayısına karşı çizilen amaç fonksiyonunun Yakınsama eğrisini vermektedir.



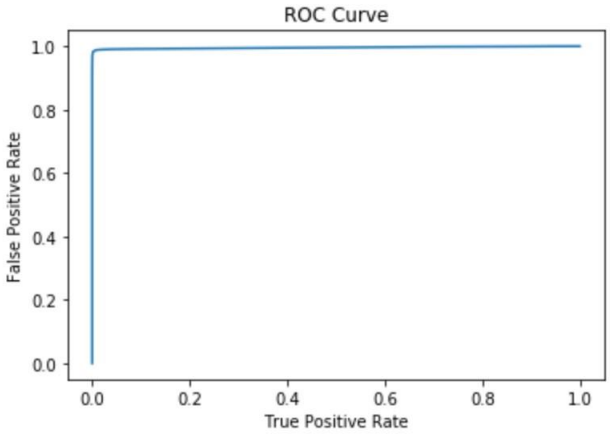
Şekil 20. Hassas Geri Çağırma Eğrisi

Şekil 20. Her olası kesme için kesinlik (pozitif tahmin değeri) ve geri çağırma (duyarlılık) arasındaki ilişkiyi gösteren kesinlik-hatırlama eğrisini verir. Grafik şu şekilde hesaplanır:

X ekseni: Geri Çağırma = hassasiyet = TP / (TP + FN)

Y ekseni: Hassasiyet = pozitif tahmin edilen değer = TP / (TP + FP)

```
areaUnderROC: 0.9956972387097107
```

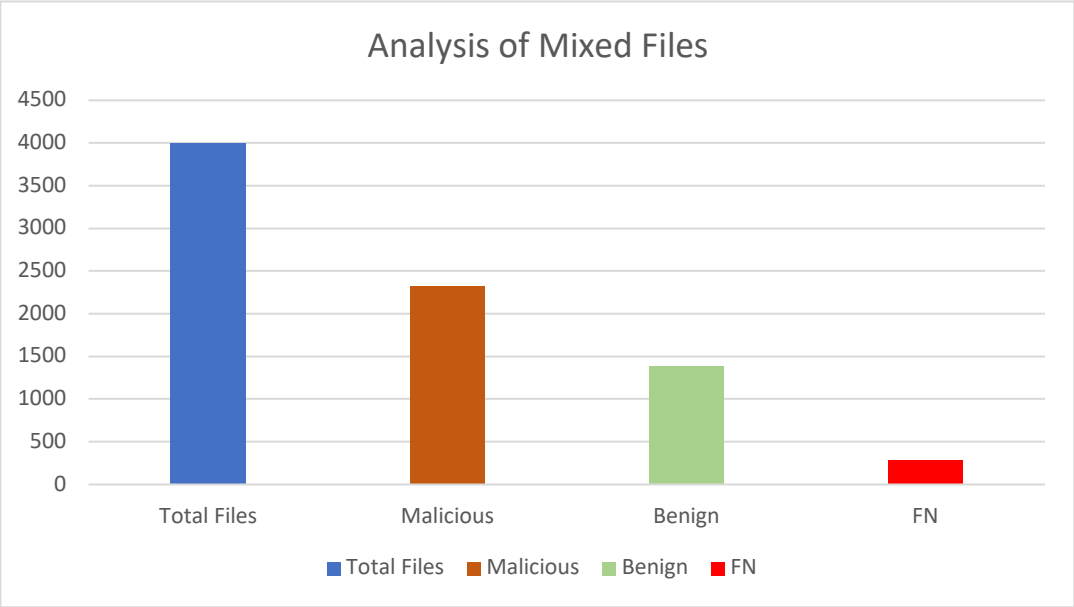


Şekil 21. AUC-ROC

Şekil 21, AUC-ROC'yi gösterir (eğri altındaki alan - alıcı çalışma özelliği Eğri) değişken ayrım eşiğine sahip lojistik regresyon modelinin tanısal yeteneğini grafiksel olarak gösterir. ROC eğrisi, düşme olarak bilinen veya 1-özgüllük olarak hesaplanan yanlış pozitif oranın (FPR), duyarlılık veya geri çağırma olarak bilinen gerçek pozitif orana (TPR) karşı çeşitli eşik ayarlarında çizilmesiyle oluşturulur.

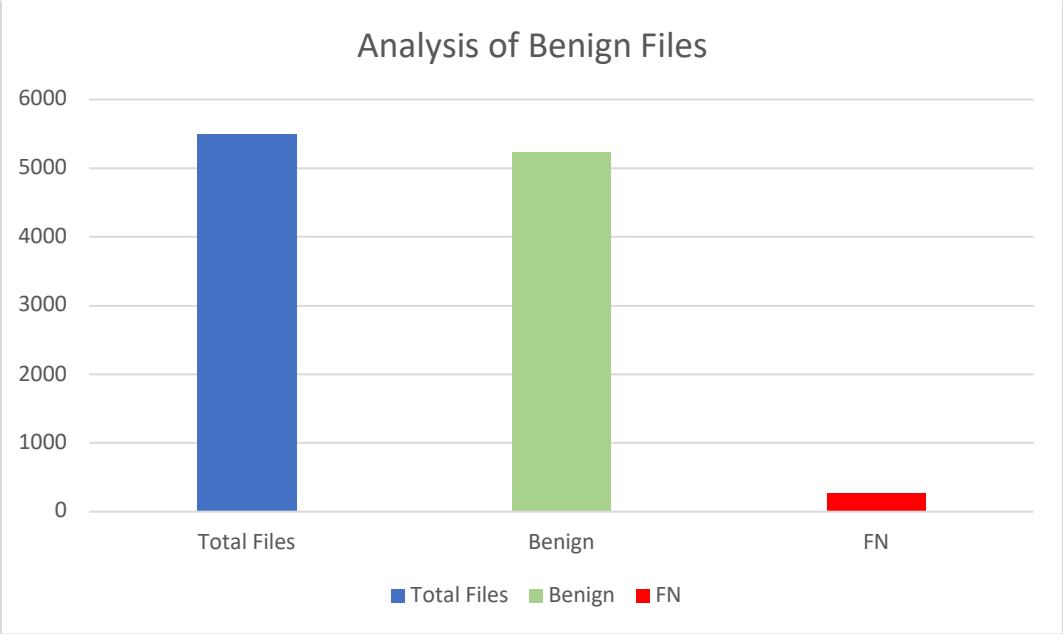
Bir dosyanın kötü amaçlı olup olmadığını belirlemek için bir dosyayı analiz eden Python aracı, belirli bir girdi hakkında karar vermek için uygulanır. Bu komut dosyası, belirli bir PDF dosyasının içeriğini okur ve şüpheli nesneleri tanımlar ve buna göre karar verir.

Bir dizi karışık, iyi huylu ve kötü amaçlı dosyayı taramanın sonuçları aşağıda verilmiştir:



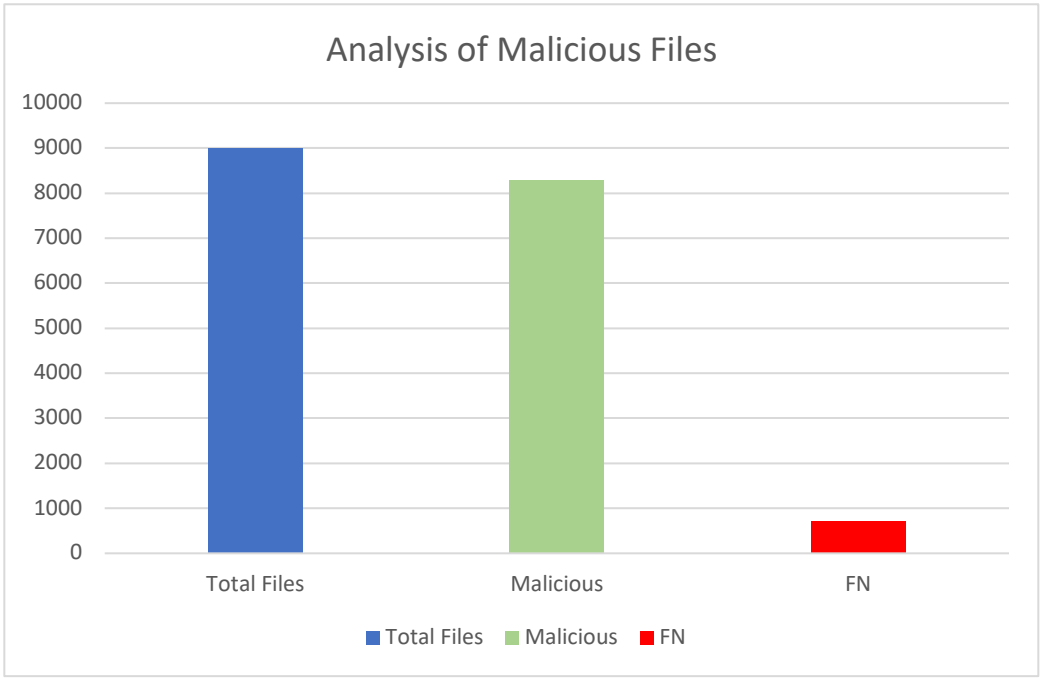
Şekil 22. Karışık Dosyaların Analizi

Böylece, tüm girdi kümesi hem kötü amaçlı hem de iyi huylu dosyaların bir karışımı olduğunda, aracın kötü amaçlı dosyaları tanımlamada %92.92 doğrulukla ve iyi huylu dosyaları tanımlamada %92.46 doğrulukla çalıştığını görebiliriz.



Şekil 23. İyi huylu Dosyaların Analizi

Böylece, tüm girdi seti iyi huylu dosyalardan oluştuğunda, aracın iyi huylu dosyaları tanımlamada %95.07 doğrulukla çalıştığını görebiliriz.



Şekil 24. Kötü Amaçlı Dosyaların Analizi

Böylece, aracın tanımlamada %92,07 doğrulukla çalıştığını görebiliriz.

tüm giriş kümesi kötü amaçlı dosyalardan oluştuğunda kötü amaçlı dosyalar.

Bölüm 5

Çözüm

Saldırganlar, saldırı gerçekleştirmek için daha güçlü ve ekonomik olarak yetenekli hale geldi. Bu saldırılar giderek daha karmaşık hale geliyor ve tespit edilmesi zorlaşıyor. Saldırganlar sistemdeki güvenlik açıklarını öğrenerek bu tür saldırıları analiz eden sistemdeki boşlukları tespit etmekte ve bu nedenle yaygın olarak kullanılan PDF belgelerine yönelik kötü amaçlı yazılım saldırılarını durduracak yöntem ve tekniklerin geliştirilmesi giderek daha önemli hale gelmektedir. Bir PDF belgesi, içine kötü amaçlı yazılımları yerleştirmek için giderek daha esnek hale geliyor. Kötü amaçlı yazılımın PDF'ye yerleştirilmesine yardımcı olabilecek çeşitli işlevleri inceledik. Kötü amaçlı URL'ler, JavaScript ve ActionScript kodu, kötü amaçlı yazılımı yerleştirmek için PDF belgesinin en çok yararlanılan özellikleridir.

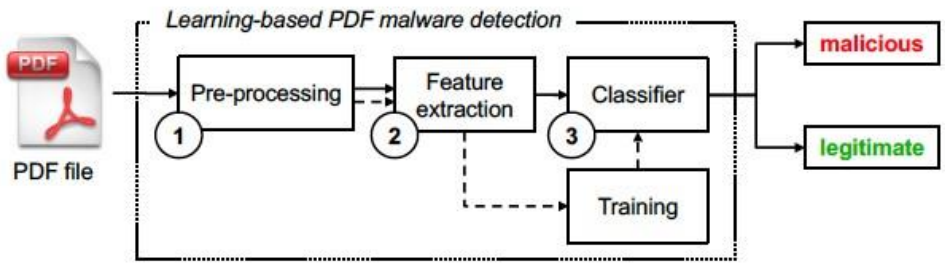
Bu raporda, okuyucu sistemde bulunan güvenlik açıklarından yararlanmak için kötü amaçlı yazılımın belgeye nasıl yerleştirildiğine dair bazı örneklere de baktık. Bunu bize daha iyi kavrayış sağlayan belgenin oluşumu açısından da gördük. Adli analiz ve makine öğrenimi modellerini eğitmek için özellikler geliştirme gibi bu tür saldırıları analiz etmek ve durdurmak için kullanılabilecek bazı yöntemler gördük.

Ancak, saldırganlar bu tekniklerde boşluklar bulur ve bu tür analiz bulgularından kaçmak için gerçek gömülü kodu gizlemede başarılıdır. Saldırıları gerçekleştirmek için ekonomik kazanımlar ve rafine yöntemlerin kullanılmasıyla birlikte, PDF belgelerinin kullanımı için güvenlik garantileri sağlamamız ve bu nedenle bu saldırıları durdurmaya ve önlemeye yardımcı olabilecek

algoritmaların geliştirilmesine ihtiyacımız var. Saldırganların er ya da geç yararlanacağı makine öğrenimi tekniklerinde birkaç güvenlik açığı vardır . Bu nedenle, daha iyi makine öğrenimi tekniklerinde de ilerlemelere ihtiyacımız var.

Bölüm 6

Öğrenmeye dayalı algılama yöntemleri, makine öğrenmesi tekniklerinin kullanımına bağlıdır. Yıllar geçtikçe, makine öğrenimi tekniklerinin kullanımı arttı. Kötü amaçlı yazılımı belirleme ve saldırıları durdurma gibi karmaşık görevleri basitleştirmeye yardımcı olur. Makine öğrenme teknikleri, ön işleme ve özellik çıkarımına dayanmaktadır [7]. Ön işlemede, dosya okuyucuda yürütülmeden belgenin içeriği çıkarılır. Bu içerikler analiz edilir ve şüpheli dosyalar daha fazla analiz için gönderilir. Şüpheli dosyalar, davranışlarını izlemek için sanal makinelerde veya sanal alanlarda yürütülür. Bu yöntem, ilk analizi atlamak için uygulanabilecek herhangi bir yöntemin atlanmasına yardımcı olur. Özellik çıkarmada [7] [8], /JavaScript gibi nesnelere daha fazla odaklanılır ve bu içerikler inceleme için belgeden çıkarılır. Kullanılan nesnelerin adları gibi belgenin yapısal özellikleri de, bulunan şüpheli bilgilerin analizi ve çıkarılması için dikkate alınır. Anusha Damodaran ve diğerleri, [6]'da öğrenme temelli modellerin uygulanabileceği çeşitli yolları tarif etmektedir. [8]'deki Samir G. Sayed ve Mohamed Shawkey, öğrenmeye dayalı algılama yöntemlerine dayanan modellerinin bir örneğini verirler. Modellerini eğitmek ve kötü amaçlı PDF belgelerini bulmak için bir özellik çıkarma, özellik seçimi ve sınıflandırıcı modeli kullanırlar.



Şekil 25. Öğrenmeye dayalı algılama mimarisi

Böylece, bu rapordan ve araçlardan teknikler öğrenilerek, bir modeli eğitmek ve bundan elde ettiğimiz sonuçlar üzerinde doğaçlama yapmak için özellikler olarak yeniden yapılandırılabilir.

REFERANSLAR

[1]. C.Curtis, X.Hu, H.Yin,AVBhaskarandM.Zhang,“Extract Me If You Can: Abusing PDF Parsers in Malware Detectors,” *23th Annual Network & Distributed System Security Symp.,* San Diego, California *, USA* , 2016 .

[2]. W Xu, Y. Qi ve D. Evans, “Otomatik Olarak Sınıflandırıcılardan Kaçınmak: PDF Üzerine Bir Vaka Çalışması

Kötü Amaçlı Yazılım Sınıflandırıcıları”, *23th Annual Network & Distributed System Security Symp., San Diego, California, ABD* , 2016.

[3]. Maiorca, Davide ve Battista Biggio. "PDF Dosyalarının Dijital Olarak İncelenmesi: Gömülü Kötü Amaçlı Yazılımın İzlerini Ortaya Çıkarmak." [astro-ph/0005112] Hubble Sabitinin Belirlenmesi

Sefe Uzaklıkları ve Yerel Tuhaf Hız Alanının Bir Modeli. 17 Temmuz 2017. Erişim tarihi: 10 Kasım 2018. https://arxiv.org/abs/1707.05102.

[4]. FD Troia, CA Visaggio, TH Austin ve M. Stamp, "Advanced transkriptaz for

JavaScript malware," *2016 11. Uluslararası Kötü Amaçlı ve İstenmeyen Yazılımlar Konferansı (MALWARE)* , Fajardo, 2016, s. 1-8. doi: 10.1109/MALWARE.2016.78888737

[5]. S. Pai, F. Di Troia, CA Visaggio, TH Austin ve M. Stamp, "Kötü amaçlı yazılım sınıflandırması için kümeleme", J. Comput. Virol. Hacking Tech., cilt. 13, hayır. 2, s. 95--107, 2017.

[6]. A. Damodaran, F. Di Troia, CA Visaggio, TH Austin, M. Stamp, "Kötü amaçlı yazılım tespiti için statik dinamik ve hibrit analiz karşılaştırması", *JCVHT* , s. 1-12, 2015.

[7]. Isabelle Guyon, André Elisseeff, Değişken ve özellik seçimine giriş, *The Journal of Machine Learning Research* , 3, 3/1/2003

[8]. Daniele Ucci, Leonardo Aniello, Roberto Baldoni, "Machine Learning tekniklerinin Kötü Amaçlı Yazılım Analizi için Kullanımına İlişkin Anket", Web'de ACM İşlemleri, Ekim 2017.

[9]. Sayed, Samir G. ve Mohmed Shawkey. "Kötü Amaçlıları Tespit Etmek İçin Veri Madenciliğine Dayalı Strateji

PDF Dosyaları." *2018 17. IEEE Uluslararası Güven, Güvenlik ve Gizlilik Konferansı*

Bilgi İşlem ve İletişim/ 12. IEEE Uluslararası Büyük Veri Bilimi ve Mühendisliği Konferansı (TrustCom/BigDataSE) (2018): 661-667 .

[10]. SHT Mavric ve CK Yeo, "Pdf belgeleri için çevrimiçi ikili görselleştirme", *2018 Uluslararası Tüketici Teknolojileri Sempozyumu (ISCT)* , St. Petersburg, 2018, s. 18-21.

doi: 10.1109/ISCE.2018.8408906

[11].

https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the_rise_of_pdf_malware.pdf

[12]. Adobe, "PDF Referansı ve PDF spesifikasyonuna Adobe Uzantıları"

https://www.adobe.com/devnet/pdf/ pdf referans.html/, erişim tarihi: 2018-03.

[13]. Hyrum S. Anderson, Anant Kharkar, Bobby Filar ve Phil Roth, “Machine Learning Kötü Amaçlı Yazılım Algılamadan Kaçınma”, Black Hat USA 2017, 22-27 Temmuz 2017, Las Vegas, NV, ABD

[14]. Himanshu Pareek, PRL Eswari ve N. Sarat Chandra Babu, “ Özellik Çıkarma ve Entropiye Dayalı Kötü Amaçlı PDF Belge Algılama”, *International Journal of Security, Privacy and Trust Management (IJSPTM),* Cilt 2, Sayı 5, Ekim 2013

[15]. Maiorca, D., Giacinto, G., Corona, I.: Kötü amaçlı pdf dosyalarının tespiti için bir kalıp tanıma sistemi. İçinde: MLDM, s. 510–524 (2012)

[16]. N. Srndic ve P. Laskov, “Hiyerarşiye Dayalı Kötü Amaçlı PDF Dosyalarının Tespiti Document Structure”, Proceedings of the 20th Annual Network & Distributed System Security Symposium, San Diego, CA USA, 2013

[17]. B. Cuan, A. Damien, C. Delaplace ve M. Valois, “Makine Öğrenimi Kullanarak PDF Dosyalarında Kötü Amaçlı Yazılım Tespiti”, REDOCS, Tech. Temsilci Raport LAAS No. 18030, Şubat 2018.

[18]. JS Cross ve MA Munson, "Gömülü kötü amaçlı yazılımları algılamaya yönelik özellikleri çıkarmak için derin pdf ayrıştırma", Sandia National Laboratories, Albuquerque, New Mexico 87185 ve Livermore, CA

94550, Tekn. SAND2011-7982, Eylül 2011.

[19]. T. Mitchell, Makine Öğrenimi. McGraw Tepesi, 1997.

[20]. J. Zhang ve J. Rabaiotti, “The PDF Exploit: Same Crime, Different Face,” https://www.symantec.com/connect/blogs/pdf-exploit-same-crime- Different-face/, erişim tarihi: 2018 -03

[21]. J. Zhang, "Görünmez"i Görünür Yap - Vaka Çalışmaları PDF Kötü Amaçlı Yazılım", Bildiriler Kitabı'nda

Hacktivite 2015, Budapeşte, Macaristan, 2015.
[22]. Choi, YH ve diğerleri: Statik ve dinamik analiz kullanarak sınıflandırma için kötü amaçlı yazılım özelliklerini çıkarmaya doğru. Bilgisayar Ağı Teknoloji. (ICCNT) (2012)

[23]. D. Maiorca, D. Ariu, I. Corona ve G. Giacinto, 1st Int'l Conf. Bilgi Sistemleri Güvenliği ve Gizliliği, 2015, s. 27–36.

[24]. M. Cova, C. Kruegel ve G. Vigna, " Drive-by-download saldırılarının ve kötü niyetli Javascript kodunun tespiti ve analizi", 19. Uluslararası Konf. Dünya Çapında Ağ, 2010.

[25]. D. Maiorca, I. Corona ve G. Giacinto, 8. ACM SIGSAC Symp'de "Çantaya bakmak bombayı bulmak için yeterli değil: Kötü amaçlı PDF dosyalarının tespiti için yapısal yöntemlerden kaçınma". Bilgi, Bilgisayar ve İletişim Güvenliği Üzerine, 2013.

[26]. C. Curtis, X. Hu, H. Yin, AV Bhaskar ve M. Zhang, "Yapabiliyorsanız Beni Çıkarın: Kötüye Kullanmak

Kötü Amaçlı Yazılım Dedektörlerinde PDF Ayrıştırıcıları", 23. Yıllık Ağ ve Dağıtılmış Sistem Güvenliğinde

Symp., San Diego, California, ABD, 2016.

[27]. https://blog.didierstevens.com/2008/05/19/pdf-stream-objects/ [05/05/2019]

[28]. http://www.simpopdf.com/resource/pdf-file-structure.html [05/05/2019]

[29]. https://resources.infosecinstitute.com/pdf-file-format-basic-structure/#gref [05/05/2019]