T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ TABANLI KÖTÜCÜL YAZILIMLARIN TESPİTİ

Arif Metehan YILDIZ

Yüksek Lisans Tezi

ADLİ BİLİŞİM MÜHENDİSLİĞİ ANABİLİM DALI

Adli Bilişim Mühendisliği

HAZİRAN 2020

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Adli Bilişim Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

MAKİNE ÖĞRENMESİ TABANLI KÖTÜCÜL YAZILIMLARIN TESPİTİ

Tez Yazarı
Arif Metehan YILDIZ

Danışman
Doç. Dr. Şengül DOĞAN

HAZİRAN 2020
ELAZIĞ

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Adli Bilişim Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Başlığı: Makine Öğrenmesi Tabanlı Kötücül Yazılımların Tespiti

Yazarı: Arif Metehan YILDIZ

İlk Teslim Tarihi: 10.06.2020

Savunma Tarihi: 30.06.2020

TEZ ONAYI

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına göre hazırlanan bu tez aşağıda imzaları bulunan jüri üyeleri tarafından değerlendirilmiş ve akademik dinleyicilere açık yapılan savunma sonucunda OYBİRLİĞİ ile kabul edilmiştir.

İmza

Danışman: Doç. Dr. Şengül DOĞAN — Onayladım
Fırat Üniversitesi, Teknoloji Fakültesi

Başkan: Doç. Dr. Türker TUNCER — Onayladım
Fırat Üniversitesi, Teknoloji Fakültesi

Üye: Doç. Dr. Şengül DOĞAN — Onayladım
Fırat Üniversitesi, Teknoloji Fakültesi

Üye: Dr. Öğr. Üy. Emrah AYDEMİR — Onayladım
Kırşehir Ahi Evran Üniversitesi, Mühendislik Fakültesi

Bu tez, Enstitü Yönetim Kurulunun ...../...../20..... tarihli toplantısında tescillenmiştir.

İmza

Prof. Dr. Soner ÖZGEN
Enstitü Müdürü

BEYAN

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım "Makine Öğrenmesi Tabanlı Kötücül Yazılımların Tespiti" Başlıklı Yüksek Lisans Tezimin içindeki bütün bilgilerin doğru olduğunu, bilgilerin üretilmesi ve sunulmasında bilimsel etik kurallarına uygun davrandığımı, kullandığım bütün kaynakları atıf yaparak belirttiğimi, maddi ve manevi desteği olan tüm kurum/kuruluş ve kişileri belirttiğimi, burada sunduğum veri ve bilgileri unvan almak amacıyla daha önce hiçbir şekilde kullanmadığımı beyan ederim.

30.06.2020

Arif Metehan YILDIZ

ÖNSÖZ

Kötü amaçlı yazılım büyük bir iştir ve saldırılar herhangi bir kişi ya da şirkete çok pahalıya mal olabilir. Kötü amaçlı yazılım savunmanızı ihlal ettiğinde, mevcut enfeksiyonları iyileştirmek ve gelecektekilerin ortaya çıkmasını önlemek için hızlı hareket edilmesi gerekir. En yeni kötü amaçlı yazılımların bir adım önünde kalmak isteyenler için, kötü amaçlı yazılım analiz işlemlerinin daha da etkili olabilmesi adına birçok özel durumdan bahsedilecektir.

Son saldırılarla "Big Data (Büyük Veri)" güvenlik sorunu haline gelmiştir. Bugünün dünyasında, Kötücül yazılımların büyüme hızı yılda on milyonlarca yeni dosyaya ulaşırken, ağlarımız her geçen gün daha büyük güvenlikle ilgili veri birikimi oluşturmaktadır. Bu gelişmiş saldırılara karşı savunma yapabilmek için güvenlik uzmanlarının bir veri bilimcisi gibi de düşünmeyi bilmeleri gerekmektedir.

Kötü Amaçlı Yazılım Sınıflandırmasında, makine öğrenmesini, istatistiklerini, sosyal ağ analizlerini ve veri görselleştirmesini tanıtarak insanlara bu yöntemleri kötü amaçlı yazılım tespiti ve analizine nasıl uygulayacaklarını göstermeyi planlamaktayız.

Arif Metehan YILDIZ
ELAZIĞ, 2020

İÇİNDEKİLER

Sayfa

ÖZET

Makine Öğrenmesi Tabanlı Kötücül Yazılımların Tespiti

Arif Metehan YILDIZ

Yüksek Lisans Tezi

FIRAT ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

Adli Bilişim Mühendisliği Anabilim Dalı

Haziran 2020, Sayfa: xi + 72

Günümüzde bilgi güvenliği açısından en önemli problemlerden birisi de kötücül yazılımları tespit etmektir. Bilgisayar korsanları çoğunlukla kötücül yazılımları kullanarak sistemleri sömürmeyi hedeflemektedir. Bu sebepten dolayı, kötücül yazılımların tespiti ve bu tür zararlıların engellenmesi bilgi güvenliği için kritik öneme sahiptir. Genel olarak bir kötücül yazılımın davranışları belirlenir ve bu yazılımlar imza tabanlı bir sistem kullanılarak tespit edilir. Ancak, teorik olarak kötücül yazılımlar belirli bir kalıpla ifade edilemeyecek davranışlara sahip olabilirler. Bu durumda kötücül yazılım tespitinin deterministik sistemler kullanılarak yapılmasının zor olduğu ve daha da zorlaşacağı bilinmektedir. Bu problemi çözebilmek için yapay zekâ ve makine öğrenmesi yöntemleri kullanılmaktadır. Son zamanlarda derin öğrenme, makine öğrenmesi ve yapay zekânın fenomeni olarak görülmekte ve kullanım alanı yaygınlaşmaktadır. Derin öğrenme sistemleri görüntü, ses ve metin tanıma gibi alanların yanı sıra bilgi güvenliği alanında da kullanılmaktadır. Özellikle kötücül yazılım tanımada derin öğrenme yöntemleri sıklıkla kullanılan yöntemler arasında sayılmaktadır. Ancak heterojen kötücül yazılım (malware) veri setlerinde de derin öğrenme yöntemleri %85 tanıma oranlarında kalmaktadır.

Bu tez çalışmasında bilgisayar korsanlarının sık sık kullandığı araçlar, kötücül yazılımlar ve anti-kötücül yazılım çözümleri araştırılarak kötücül yazılım tanımadaki etkileri incelenmiş ve böylece bu tür kötücül yazılımları yeni nesil özellik çıkarma ve öğrenme yöntemleri kullanılarak daha yüksek doğrulukta tanımak amaçlanmıştır.

Anahtar Kelimeler: Yapay Sinir Ağları, Yapay Zekâ, Kötücül Yazılım Sınıflandırma, Derin Öğrenme, Makine Öğrenmesi.

ABSTRACT

Machine Learning Based Malwares Detection

Arif Metehan YILDIZ

Master's Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences

Department of Digital Forensic Engineering

June 2020, Pages: xi + 72

Nowadays, one of the most important problems in terms of information security is to detect malware. Hackers often aim to exploit systems using malicious software. For this reason, the detection and prevention of malware is critical to information security. In general, the behavior of a malware is determined and detected using a signature-based system. However, theoretically malicious software can have endless behavior. It is known that detection of malware by using deterministic systems is difficult and will be more difficult. Artificial intelligence and machine learning methods are used to solve this problem. Recently, deep learning has been seen as a phenomenon of machine learning and artificial intelligence and its use is becoming widespread. Deep learning systems are used in information security as well as in areas such as image, voice and text recognition. In particular, deep learning methods are often used in malware detection, but deep learning methods remain at 85% recognition rates in heterogeneous malware data sets.

In this thesis, the tools that hackers frequently, the effects of malware and anti-malware solutions have been investigated and their effects on malware recognition have been investigated and thus, it has been aimed to recognize such malware with higher accuracy by using new generation feature extraction and learning methods.

Keywords: Artificial Neural Networks, Artificial Intelligence, Malware Classification, Deep Learning, Machine Learning.

ŞEKİLLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

Sayfa

SİMGELER VE KISALTMALAR

Simgeler

S3 : Basit Depolama Hizmeti
f : Fonksiyon
G_{min} : Görüntünün minimum piksel değeri
G_{max} : Görüntünün maksimum piksel değeri
G^N : 0 ile 1 arasında normalize edilmiş görüntü

Kısaltmalar

AI : Yapay Zekâ
API : Uygulama Programlama Arayüzü
AUC : Eğrinin Altındaki Alan
AV : Anti-Virüs
BT : Bilgi Teknolojisi
CPU : Merkezi İşlem Birimi
DT : Karar Ağaçları
GPU : Ekran İşlem Birimi
GSYİH : Gayrisafi Yurt İçi Hâsıla
HOG : Yönlendirilmiş Gradyanların Histogramı
HPC : Yüksek Performanslı Bilgi İşlem
IBM : İstatistik Bilgi Merkezi
KNN : K-En yakın Komşu
LBP : Yerel İkili Örüntüler
LTSM : Uzun Kısa Süreli Bellek
NLP : Doğal Dil İşleme
PR : Hassas Geri Çağırma
SMB : Sunucu İleti Bloğu
SVM : Destek Vektör Makineleri
RaaS : Hizmet olarak Fidye Yazılımı
RF : Rastgele Orman
RNN : Tekrarlayan Sinir Ağları
XXE : XML Harici Varlığı
CRLF : Satır Başı ve Yeni Satır
LDA : Doğrusal Ayırım Analizi

1. GİRİŞ

Son zamanlarda, bilgi teknolojisinin hızla gelişmesiyle birlikte, çeşitli ülkelerin hükümetleri ve finansal ağları, genellikle ülkelerin finans, milli savunma ve eğitim endüstrilerini ciddi şekilde tehdit eden kötü amaçlı yazılımlar tarafından sık sık saldırıya uğramıştır. Kaspersky'nin son raporuna göre, 203 ülkede bulunan çevrimiçi kaynaklardan gelen 975.491.360 saldırı engellendi. 273.782.113 eşsiz URL, Web Anti-Virüs bileşenleri tarafından zararlı olarak kabul edildi [1]. Bununla birlikte Kaspersky'nin web ant-virüs yazılımı 24.610.126 benzersiz zararlı nesne tespit etti. Ayrıca 766.728 cihazda çevrimiçi bankacılık aracılığıyla para çalabilen kötü amaçlı yazılım çalıştırma girişimleri engellendi. McAfee ve CISCO tarafından yayınlanan ortak bir rapor (Yeni küresel siber güvenlik, 2018) geçen yıl siber suçun yol açtığı zararların, küresel Gayri Safi Yurt İçi Hâsıla (GSYİH)'nin %0,8'ine yani buna eşdeğer olarak 60 milyar ABD Dolarına ulaştığı belirtildi[2]. Tüm bu rakamlar göz önüne alındığında siber suçlar genellikle saldırı için belirli kötü amaçlı yazılımlar kullanılarak ve hedefe saldırmak için bunların aktif kullanımını gerektirdiğinden, kötü amaçlı yazılımı etkili bir şekilde analiz edebilmek de büyük önem taşır. Günümüzün Anti-Virüs yazılımı (AV) sıklıkla imza tabanlı bir algılama yöntemi kullanır. Ancak, yeni bir kötü amaçlı yazılım çeşidi oluştuğunda, AV zaman içerisinde yeni bir kötü amaçlı yazılım imzası alamaz, bu nedenle bu yöntem yeni kötü amaçlı yazılımı etkili bir şekilde tespit edemez. Araştırma alanında, kötü amaçlı yazılım analizi teknolojisi genellikle statik analiz ve dinamik analiz olmak üzere iki ana kategoriye ayrılır [3]. Statik analiz, kötü amaçlı yazılımları hızlı bir şekilde tanımlayabilir, ancak şifreleme ve paketleme gibi şaşırtma saldırılarına karşı savunmasızdır. Dinamik analiz ise statik analiz teknolojisinin eksikliklerinin etkili bir şekilde üstesinden gelebilir, ancak kötücül yazılım kaçakçılığına karşı savunmasızdır.

AV yazılımının kendisi bir tür yazılım olduğundan, algılama yazılımının kendisinin kusur yoğunluğu normal yazılımınkiyle aynıdır[4]. Bu nedenle, kayda değer miktarda araştırma, donanım özelliklerine dayalı kötü amaçlı yazılımların analizine odaklanılmıştır. Donanım özelliklerinin kullanımı, dinamik analizde kötü amaçlı yazılım sızması sorununu çözmek için yazılım özelliklerinin eksikliğini önleyebilir. Demme ve diğerleri performans sayaçlarının (örneğin IPC, önbellek davranışı, bellek davranışı vb.) kullanılmasının, donanım özelliklerini çıkararak kötü amaçlı yazılımları etkili bir şekilde sınıflandırdığını kanıtlamıştır[5].

Yukarıdaki araştırmaya dayanarak, Tang ve arkadaşları kötü amaçlı yazılımları tespit etmek için denetimsiz yöntemlerle birleştirilmiş donanım performans sayaçlarını (HPC) kullandı. Bununla birlikte, yukarıdaki yöntem, kötü amaçlı yazılımların HPC ile davranışını tam olarak tanımlayamaz ve sınıflandırma doğruluğu yetersiz kalır[4]. Tian ve diğerleri Destek Vektör Makineleri (SVM), Rastgele Orman (RF), Karar Tablosu (DT) ve IB1 + AdaBoost ile birlikte, RF ve DT (Tian) kullanarak %97 kötü amaçlı yazılım sınıflandırma doğruluğunu temel işletim sistemi

olarak Windows XP ile birlikte çağırmıştır [6]. Bu sonuç, 10 kat çapraz doğrulama kullanılarak elde edilmiştir. Firdausi ve diğerleri J48 kullanarak Windows XP tabanlı bir işletim sisteminde % 96,8 doğruluk sağlamak için API çağrıları kullanan K-En Yakın Komşu, J48 Karar Ağaçları, SVM ve Çok Katmanlı Perceptron kullanıldı [7]. 250 adet kötü amaçlı yazılımın küçük bir örneğini kullanmışlar, bu nedenle bu sonuçların çapraz doğrulandığını da varsayılmaktadır. Damodaran ve diğerleri API çağrıları ve bir karma statik / dinamik model ile Saklı Markov Modelleri (HMM) kullanmıştır [8]. Bir başka araştırmada kötücül yazılım sınıflandırması için 0.98 AUC-PR puanı elde etmek için 5 kat çapraz doğrulama kullanmışlardır. Dinamik analizin statik ve hibrit sınıflandırmayı geride bıraktığını göstermişlerdir. Dahası, test sırasında benign sınıfı şişirmişler, daha gerçekçi ve kötü niyetli örneklerin dengesini oluşturmak için, iyi huylu örnekler arttıkça sınıflandırıcı performansının düşmesi ise dinamik yaklaşımın statik veya hibrit yaklaşımlardan daha kararlı durmasını sağlamıştır [9]. Tobiyama ve arkadaşları Uzun Kısa Süreli Bellek (LSTM) modellerini kullanarak 81 kötü amaçlı yazılım günlüğü ve 69 iyi niyetli günlüğünden özellik çıkarımı gerçekleştirmiştir [10]. 5 kat çapraz doğrulama kullanarak 0.96 AUC skoru elde etmişlerdir. Ahmed ve arkadaşları Yaklaşık 500 dosyadan oluşan bir veri seti kullanarak API çağrıları dizisini incelemişlerdir. 10 kat çapraz doğrulama ve Naive Bayes sınıflandırıcısı kullanarak 0.98 AUC performans göstermişlerdir [11]. Dini ve arkadaşları tarafından cihaz etkinliği ölçümleri, daha önce Android gibi mobil platformlarda, kötü amaçlı yazılımları tespit etmek için kullanılmış, SMS ve tuş vuruşlu veriler gibi eklenerek cihaza özgü ölçümler tarafından desteklenmiştir [12]. Daha geleneksel masaüstü bilgisayar ortamlarında Scaife ve arkadaşları bir kullanıcının verilerini şifrelemek için dosya sistemine yoğun bir şekilde okuma / yazma çağrıları kullanan belirli bir kötü amaçlı yazılım türü olan RansomWare'i incelemiş. Makine verilerinden ziyade kullanıcı verilerinin değiştirilmesine ve şüpheli dosya etkinliğine ilişkin kurala dayalı göstergeler kullanarak %100 doğrulukla 492 RansomWare saldırısı örneğiyle eşleşmiştir [13]. Continella ve arkadaşları ayrıca RansomWare'i okumuş ve bu tür saldırıları tespit etmek için dosya sistemi aramalarını deneyerek makine öğrenmesini kullanmıştır. Mevcut literatürlerin çoğundan farklı olarak, sınıflandırıcılarını 305 görünmeyen örnek kullanarak test etmiş ve görünmeyen RansomWare saldırılarının %97,7'sini tespit etmişlerdir [14]. Araştırmalarda, bu sonuçları, görülmeyen verilerin performansta düşüşe neden olacağı hipotezine dair kanıt sağlamak için en iyi performans gösteren sınıflandırma yöntemlerini kullanarak görülmeyen verilerin farklı davranışlar sergilemesi nedeniyle özellikle RF ve DT gibi ayrık tercihli modellerde kanıtlanmasını sağlayacak şekilde üretmeyi denemişlerdir. Pascanu ve diğerleri yeniden düzenlenmiş zamansal kalıpları ele almak için daha fazla araştırma yapılması gerektiğini savunmuştur [15]. Bu nedenle Yankı Durum Ağları(ESN) ve Tekrarlayan Yapay Sinir Ağları (RNN) kullanılarak yürütülen talimatları çıkarmak için bir yaklaşım önermiş. Sonrasında kötü niyetli dosyaları %71.71 gerçek bir pozitif oranla ve yanlış bir şekilde sınıflandırmak için % 0.1'lik pozitif oranla Lojistik Regresyon kullanmıştır.

Buradaki kilit nokta, tekrarlayan modelin üç iyileştirme faktörü gösterdiği, tekrarlayan model ile bir olaylar çantası modeli arasındaki karşılaştırmaydı. Çalışmayı yapay sinir ağları'na genişleten Dahl ve arkadaşları, Lojistik regresyon, sığ ve derin sinir ağları kullanarak modelleri eğitmek ve test etmek için 3.6 milyon dosya kullanmıştır (Microsoft'tan) [16]. API çağrıları kullanarak 9 Sinir Ağları topluluğu ile %0.49 hata oranı ürettiler. En iyi performans gösteren modelin sığ bir Sinir Ağı (1 kat) olduğu dikkat çekici bir durum yaratmıştır.

Bununla birlikte, Huang ve Stokes bu araştırmaya dayanarak Derin Sinir Ağı yetiştirmek ve test etmek için 6 milyonluk bir örnek (Microsoft'tan) kullanmıştır [17]. % 0.36'lık bir hata oranı elde etmek için 114 API olayı ve pozitif parametreler kullanarak iyi bir etki için ReLU ve Release kullanmışlardır. Sinir Ağlarına (NN) daha fazla katman eklemek, performansta önemli kazanımlar elde etmemiş olup Dahl ve arkadaşlarının bulgularını teyit etmiştir. Ayrıca örnek büyüklüğünü de Dahl ve arkadaşlarının neredeyse iki katına çıkarmıştır. Ancak Dropout'u kullanmak, iyileştirmeler göstermiştir; bu durum, Sinir modelinin verimliliğini optimize etmenin, derinliği ve veri kümesi boyutunu artırmaktan daha fazla performansı artırabileceğini öne sürmüştür. Her iki veri seti de halka açık değildir, bu nedenle bu araştırmaya karşı kıyaslama yapmak zordur. Her iki makalenin de bu çalışmaya özgü olmayan, ancak Genel Sinir Ağı modellerinin kilit bir kısıtlaması da model oluşturma süresidir. Dahl ve arkadaşlarının kullandığı modeli eğitmek yaklaşık 3 saat, Huang ve Stokes için ise 7 saat sürmüştür. Her gün ortaya çıkan çok sayıda yeni kötü amaçlı yazılım nedeniyle, bu modellerden herhangi birinin mevcut imza tabanlı virüsten koruma sistemlerinde olduğu gibi "gerçek zamanlı" görünen yeni örnekleri dikkate almak için oluşturulmuş ve dağıtılmış olması mümkün görünmeyecektir.

Yukarıdaki problemler için, bu tez çalışmasında özellikle yeni nesil özellik çıkarma ve öğrenme yöntemleri kullanılarak daha yüksek doğrulukta sınıflandırmak amaçlanmıştır. Bu yöntemler kullanılarak kötücül yazılım özellikleri çıkarılmaya çalışılmıştır. Çıkarılan özellikler konvansiyel sınıflandırıcılar kullanılarak öğrenme modelleri oluşturulmuştur.

2. ARKA PLAN

Bu bölümde literatürde yaygın olarak görülen kötücül yazılımlar ve bunları tespit etmede kullanılan açık kaynaklı araçlar, yöntem ve kavramlar ile ilgili bilgi verilmiştir.

2.1. Genel Kötücül Yazılımlar

Kötücül yazılımlar, bilgisayar korsanları olarak adlandırılan kişi ve gruplar tarafından sık sık kullanılan, bunların kullanımı sonucunda ise kişisel veya kurumsal bilgilerin sızdırılması durumlarına karşı tehdit unsuru oluşturabilecek her türlü kod parçacığıdır.

2.1.1. Virüs

Virüs, meşru programlara eklenmiş küçük kod parçalarından oluşan bir tür kötü amaçlı yazılımdır. Bu program çalıştığında virüs de çalışır.

Yazılım şirketlerine göre ise virüsler, bilgisayar dosyalarına kullanıcı bilgisi olmadan yayılan kötü amaçlı programlardır. Yaygın virüs bulaşmalarının çoğu, açıldığında etkinleşen e-posta mesajı ekleriyle yayılır. Virüsün kısır döngüsü, virüslü e-postaların birden fazla kullanıcıya iletilmesiyle devam eder. Virüsler ayrıca Universal Serial Bus (USB) sürücüleri gibi paylaşılan ortamlara da yayılır.

'Bilgisayar virüsü' terimi ilk olarak 1983 yılında Fred Cohen tarafından resmen tanımlanmıştır. Bilgisayar virüsleri hiçbir zaman doğal olarak oluşmaz. Her zaman insanlar tarafından uyarılırlar. Bununla birlikte, bir kez yaratılıp serbest bırakıldıklarında, bunların yayılması doğrudan insan kontrolünde değildir. Bir virüs, bir bilgisayara girdikten sonra kendisini ana programın yürütülmesiyle virüsün eylemini aynı anda tetikleyecek şekilde başka bir programa bağlar. Kendini çoğaltabilir, kendisini diğer programlara veya dosyalara ekleyerek bu süreç içinde onları etkileyebilir. Tüm bilgisayar virüsleri yıkıcı değildir. Ancak çoğu, veri yok etme gibi teknolojik ortamda zararlı olan eylemleri gerçekleştirir. Bazı virüsler, kod yürütüldüğü anda hasara yol açarken, diğerleri belirli bir olay (programlandığı gibi) başlatılana kadar uykuya yatmakta ve kodlarının bilgisayarda çalışmasına neden olmaktadır. Eklenen yazılım veya belgeler ağ, disk, dosya paylaşım yöntemleri veya virüslü e-posta ekleri aracılığıyla bir bilgisayardan diğerine aktarıldığında virüsler yayılır. Bazı virüsler, virüsten koruma yazılımlarından algılanmalarını önlemek için farklı gizlilik stratejileri kullanır. Örneğin, bazıları dosyaları boyutlarını yükseltmeden etkileyebilir, bazıları ise algılanmadan önce virüsten koruma yazılımıyla ilişkili görevleri bitirerek algılama sisteminden kaçmaya çalışır. Bazı eski virüsler, ana bilgisayar dosyasının "en son değiştirilme" tarihinin dosyaya bulaştıklarında aynı kalmasını sağlar.

Başlangıçta muziplik için oluşturulan virüsler, yaygın ve önemli bilgisayar sistemlerinden ve dosya bozulmalarından sorumludur. Anti-virüs yazılımı yüklemek, önceden yüklenmiş virüsleri önlemeye, engellemeye veya kaldırmaya yardımcı olur.

2.1.2. Truva Atı

Bir bilgisayar virüsünü bir bilgisayarın diğerine bağlantı sağlamaya çalışırken bulaştırdığı zararlı yazılımlar olarak tanımlayabiliriz. Görevi ne olursa olsun uygulamak için kendisini bir programa veya bir dosyaya ekler. Ayrıca, virüslerin bir bilgisayarı rahatsız etmek için insan eylemine ihtiyacı yoktur. Düzenli çalışan virüslerin aksine, Truva atı virüsü kendisini çoğaltamaz ve hatta başka bir dosyaya ekleyemez. Truva atı virüsü, kullanıcıyı bilgisayarına kurması veya indirmesi konusunda kandırmak için meşru bir dosya gibi görünmektedir. Teknik farklılıklara rağmen, her ikisi de bir kullanıcının bilgisayarında ve hatta cihazlarında aynı tepkilere sahiptir.

Truva atı virüsü sahte bir dosyadır diyebiliriz. Bu sizin için yardımcı olması gereken bir dosya veya program gibi görünebilir ama değildir. 2017 yılında, ABD'li işletme sahiplerinin neredeyse % 50'sinin Trojan dâhil birçok farklı siber tehdit mağduru olduklarını bilmediğini belirten bir anket yayınlanmıştı. Bu nedenle, Truva atı virüsünü tüm zamanların en kötü siber saldırılarından biri olduğunu söyleyebiliriz [18].

Bir Truva atı virüsü gerçek bir virüs olarak nitelendirilmez. Ancak, bilgisayarınızda bir tane olmasının tehlike boyutunu bilmek önemlidir. Aslında çok fark edilebilecek bir şey değildir. Sonuçlar, ne tür Truva atı ile karşı karşıya kaldığınıza bağlı olarak değişebilir. Aşağıda bir bilgisayardaki Truva atı virüsünün zararlı faaliyetlerini ele alacağız.

Casus yazılım olarak kullanılan Truva atı: Truva atları bir Casus Yazılım olarak çalışabilir. Çevrimiçi hesaplarınızı kullanana veya kredi kartı bilgilerinizi girinceye kadar bekler. Ardından, şifrelerinizi ve diğer bilgilerinizi Bilgisayar korsanına geri gönderir. Bundan sonra ise bilgisayar korsanı, hedeflerini mağdur etmek için kendi planlarını yapabilir.

Arka kapılar yaratan Truva atı: Truva atlarının bu türleri sistemde giriş kapıları yaratır. Truva atları, kodlarınızı veya güvenlik sisteminizi de değiştirebilir. Bununla, fark edilmeden güvenlik araçlarından daha fazla kötü amaçlı yazılım geçebilir ve bu ise sistemde büyük bir güvenlik açığına neden olabilir.

Bilgisayarları zombilere dönüştüren Truva atı: Bilgisayar korsanı, yalnızca hesapları veya bilgileri çalmaz, ayrıca DDoS saldırıları kullanarak diğer insanları da rahatsız eder. Bunu yapmak için, Truva atlarını bilgisayarlarınıza yerleştirecek ve kendi çıkarları için kullanacaklardır.

Bilgisayarın IP adreslerinden siber suçlar tespit edildiğinden, bu maruz kalan kişileri büyük bir sıkıntıya sokabilecek duruma gelecektir.

Robs telefon faturalarında Truva atı: Truva atlarının tek hedefi bilgisayar değildir. Ayrıca, premium numaralara pahalı SMS mesajları göndermek için akıllı telefonları da kullanabilir. Siber suçlara gönül vermiş bir kişi, nitekim bunlar aynı zamanda siyah şapkalı hacker olarak da nitelendirebildiğimiz kişiler olup bu yolla büyük paralar kazanabilirler.

Truva atı virüsünden korunmak için yapılması gerekenler: İlk olarak yapmanız gereken şey endişelenmemektir. Truva atlarından kaçınılması zor görünse bile, Truva atını bilgisayarınızdan ve aygıtlarınızdan ayırmanın birkaç yolu vardır. Bu basit adımları izlerken asla aldatıcı Truva atıyla uğraşmak zorunda kalmazsınız.

- Virüsten koruma yazılımınızı her zaman mümkün olduğunca sık kullanmanız gerekir.
- Bilinmeyen e-postaların eklerini ve kötü amaçlı bağlantılarını asla tıklamamalısınız.
- Mümkün olduğunca kötü amaçlı web sitelerinden uzak durmalısınız.
- Virüsten koruma yazılımınızın her zaman güncellendiğinden emin olmalısınız.
- Kişisel bilgilerinizi güvende tutmak için her zaman güvenlik duvarları kullanmalısınız.
- Kişisel korunmanız için zor şifreler kullanmanız gereklidir.
- Son olarak bilgisayar korsanları her tür bilgisayar ve akıllı telefon kullanıcısını hedef alırlar ve ceplerinden para alana kadar da durmazlar. Hatta sanal ortamda az zaman geçiren kullanıcıları rahatsız etmeyi bile denerler. Bunlara asla taviz vermeyin.

2.1.3. Reklam(adware)

Adware, web tarayıcılarını reklamlarla dolduran şüpheli bir program grubudur. Adware, reklam veren, açılan pencereleri , pankartları, metin içi bağlantıları, otomatik oynatılan video reklamları ve tarayıcılardaki diğer ticari içerikleri sunan potansiyel olarak istenmeyen bir programdır. Ek olarak, bu reklam destekli uygulama, çeşitli web sitelerine yönlendirmeler başlatabilir ve kullanıcının izni olmadan göz atma geçmişi ile ilgili bilgileri toplayabilir.

Adware programlarının temel amacı, yaratıcıları için kolay kâr getirmektir. Bu amaca ulaşmak için, kötücül yazılım popüler web tarayıcılarına sızar ve daha önce belirtilen promosyon içeriğiyle onları yerleştirir. Bu reklamlar, kullanıcıların dikkatini çekmeye ve onları doğrudan kendilerine bağlı kurulu web sitelerine yönlendirmeye yöneliktir. Böylece katılımları artar ve popülerliklerini veya satışlarını arttırırlar. Bu nedenle ünlülerin imajlarını, tanınmış şirketlerin devasa logolarını veya cazip meblağları içerirler [19].

Aşırı miktarda reklam nedeniyle, reklam yazılımı tarafından baskı altına alınan tarayıcılar çoğu zaman etkin kullanılması imkânsız hale gelir. Ne yazık ki, yavaşlayan veya beklenmeyen

durumlar, mağdurları reklam yazılımlarını temizleme gibi çözümleri aramaya zorlamaktadır. 2019'daki reklam yazılımı ile ilgili duruma bakacak olursak Adware, dünyadaki bilgisayar kullanıcıları için sayısız sorunlara neden olmuştur. Ancak, bu yıl boyunca reklam destekli uygulamalar mobil kullanıcıları ve Mac OS X'e dayanan sistemleri hedeflemeye başladı. Araştırmacılar olarak 2018'de yalnızca kullanıcıların tarayıcılarında sponsorlu içerik göstermeyen, aynı zamanda bunu desteklemeye devam eden adware enfeksiyonlarının sayısını artırdığını ve bunun yanı sıra dolandırıcılığı da desteklediklerini söyleyebiliriz. Ne yazık ki, gelecek yıl boyunca daha baskın adware türleri beklenmektedir.

2018'de Mac OS X kullanıcılarını aktif olarak hedefleyen adware türleri de görülmeye başlandı. Bugün, hangi işletim sisteminin kullanıldığı ve bu küçük hattın 2019'da tamamen kaybolacağı büyük bir fark yok gibi görünmektedir. Ne yazık ki, birçok Mac kullanıcısı hala Mac'lerinin siber tehditlere karşı bağışık olduğuna inanmaktadır. Bu nedenle, virüsten koruma yazılımı yüklemeyi düşünmedikleri için daha da savunmasız hale gelirler.

Bununla birlikte, güvenlik araştırmacıları olarak kullanıcıların adware kaldırılması gerekliliğini önlemelerine yardımcı olan aşağıdaki basit ipuçlarını izleyerek bilgisayarları ve mobil cihazları korumayı önermekteyiz:

• Saygın bir güvenlik yazılımına yatırım yapmak.

• Şüpheli programları veya uygulamaları bilinmeyen kaynaklardan dosya indirmemek veya güvenilmeyen geliştiriciler tarafından dosya oluşturmamak.

• İstenmeyen uygulamalara sızmayı önlemek için Gelişmiş / Özel ayarları kullanarak yazılımı doğru şekilde kurmak.

• Herhangi bir tarayıcı uzantısını, araç çubuğunu veya güvenlik programını reklamlardan, açılır pencerelerden veya yönlendirme web sitelerinden indirmeyi kabul etmemek.

2.1.4. Arka Kapı

Günümüzün iş ortamında, şirketler ağ ihlallerini önlemek için ellerinden geleni yapmaktadırlar. Neredeyse her taraftan gelen saldırılarla, her vektör ve giriş noktasının korunmasını sağlamak bazen zor olabilir. Son zamanlarda, arka kapı saldırılarında bir artış oldu. Burada, bir arka kapı saldırısının ne gerektirdiğini, onları bu kadar tehlikeli bir risk faktörü yapan ve işletmelerin kendilerini nasıl koruyabileceklerinden bahsedilecektir.

Bir arka kapı saldırısının temelleri: Trend Micro'nun " Hedefli Saldırılarda Arka Kapı Kullanımı "[20] adlı raporuna göre, bilgisayarlara uzaktan erişim sağlayan (arka kapılar olarak bilinen uygulamalar) genellikle hedefli saldırılar için kullanılır. Bu tür ihlallerde, bilgisayar korsanları kurbanın ağına erişmek için arka kapı programlarından yararlanır. Bu saldırı yönteminin

faydası, arka kapının kendisinin bilgisayar korsanlarının keşfedilmeden altyapıya girmelerine yardımcı olabileceğidir.

Genellikle başlangıçta hedeflenen saldırı sürecinin ikinci (giriş noktası) veya üçüncü (komut ve kontrol [C&C]) aşamasında kullanılır, arka kapılar, bilgisayar korsanlarının hedef ağların kontrolünü kazanmasını sağlar. Başka bir düşünceye göre ise gerçekçi olacak olursak araştırmalar, hedefli saldırılarda kullanılan arka kapıların çoğunun, herhangi bir izinsiz giriş tespit sistemini (IDS) atlamak için özellikle tasarlandığını ortaya koymaktadır.

Arka kapı saldırılarında saldırı stratejileri Arka kapılar, bilgisayar korsanları için gizli bir giriş noktası sağlamakla kalmaz, aynı zamanda izinsiz giriş için de çeşitli stratejiler sunabilir. Trend Micro'nun raporunu baz alarak şunları belirtebiliriz:

Bağlantı noktası bağlama: Güvenlik duvarları yaygın olarak kullanılmadan önce, bağlantı noktası bağlama, mesajların ağ içinde nerede, nasıl ve ne zaman iletildiğini ortaya çıkarmak için özel bilgi yapılandırmalarını içerir.

Bağlanma: Bilgisayar korsanları, birçok ağda güvenlik duvarları devreye girdiğinde, hedeflenen sistemleri, bilgisayar korsanlarının Komut ve Kontrol (C&C) sunucu sistemlerine bağlamak için arka kapıların güçlendirildiği bağlanma yaklaşımını kullanmaya başladı. Bu durum, güvenlik duvarı koruması altında olmayan bağlantı noktalarından sunucular üzerinde hedef platforma ters bağlantı yapılmasına da izin verir.

Bağlanılabilirlik kullanımına bağlanmak*:* Bu strateji yalnızca ağı ihlal etmek için değil, aynı zamanda uzun süre boyunca tespit edilmeden kalmak için birkaç kötü amaçlı yazılım örneğinin kullanılmasını içerir. Bu, açık kapıları kullanan bilgisayar korsanlarının hedeften hassas verileri çalması gerektiğini belirtir. İlk kötü amaçlı yazılım veya "birincil sıra arka kapı", ikinci bilgiyi indirmek için bir platform görevi görür; bu, gerçek bilginin çalınmasını gerçekleştiren "ikincil sıra arka kapı" sayesinde verileri indirir diyebiliriz.

Meşru platform istismarı: Raporda, meşru platformları kötüye kullanmanın, özellikle bilgisayar korsanlarının artık adım adım güvenlik sistemleri için daha çok çalışması gerektiğinden daha yaygınlaştığı belirtildi. Bu stratejide, bilgisayar korsanları geçerli bir platformu (örneğin bir blog gibi) kötüye kullanıp bunu Komut ve Kontrol (C&C) sunucusu verilerinin depolanması için kullanabilirler. Bunlar arka kapılarla uygulanabilecek birkaç saldırı stratejisidir. Ayrıca diğer yaklaşımların ortak hizmetler protokolü veya dosya başlığının kötüye kullanımı, protokol veya bağlantı noktası dinleme, özel DNS arama kullanımı ve bağlantı noktasının yeniden kullanılmasını içerdiğini de söyleyebiliriz.

Ek olarak, Tripwire, yazılımın arka kapıya sahip olan tek sistem olmadığını belirtti. Kimlik doğrulama belirteçleri, ağ aygıtları, gözetim sistemleri ve bazı iletişim altyapısı aygıtlarını içeren donanım bileşenleri, siber suç girişine izin veren kötü amaçlı arka kapılara da sahip olabilir [20].

Arka kapı saldırılarına karşı korunma önlemleri: Öncelikle ve her şeyden önce, şirketlerin yetkili kullanıcıların dışındaki tüm giriş noktalarını engelleyebilecek güvenlik duvarları olması gerekir. Bir güvenlik duvarı mevcutsa, bir port bağlayıcı ile bile arka kapı saldırısının gerçekleştirilmesi neredeyse imkânsız olduğundan bu özellikle çok önemlidir. Ek olarak, özellikle açık kaynak tabanlı tüm programların güçlü ağ izlemesi teşvik edilmelidir. Bu şekilde, işletmeler kullandıkları açık kaynak uygulamaları konusunda seçici olmalı ve uygulamaların bilinen saygın bir kaynaktan gelmelerini sağlamalıdır.

Ağ izleme aynı zamanda arka kapı saldırılarına karşı koruma söz konusu olduğunda anahtar bir uygulamadır. İzleme, bir komut ve kontrol sunucusu tarafından toplanan bilgiler gibi herhangi bir şüpheli etkinliğin ağ yöneticileriyle işaretlendiğini garanti etmeye yardımcı olabilir. Bunlar düşünülünce BT personeli daha sonra konunun kökenine ulaşmak, saldırıyı durdurmak ve hasarı azaltmak için hızlı bir şekilde tepki verebilir.

Diğer bir koruma önlemi, bir anti-kötücül yazılım çözümünün kullanılmasını içerir. Yazılım şirketleri, bazı arka kapı saldırılarının ağ trafiğine takılmadığını bildirirken ağ etkinliğinin orijinal göründüğünü ve herhangi bir alarm vermediği de belirtildi. Bununla birlikte, iyi bir şekilde düzenlenmiş bir anti-kötücül yazılım sistemi bu tür durumlarda bile arka kapı saldırlarını tespit edebilir.

Arka kapı saldırıları, işletmeler için önemli bir tehdit oluşturmaktadır, ancak bunların nasıl gerçekleştiğini ve nasıl önlenebileceğini anlamak daha iyi bir korunma yöntemi oluşturma aşamasında uzun bir yol kat edilebileceği anlamına gelmektedir.

2.1.5. Casus Yazılım

Bilim-Kurgu filmlerindeki karakterlerin kullanabileceği bir şey gibi görünse de, casus yazılımların hepsi gerçektir. Casus yazılım, kendini bilgisayarınıza yükleyen ve çevrimiçi davranışınızı bilginiz veya izniniz olmadan gizli olarak izlemeye başlayan herhangi bir yazılımdır. Casus yazılım, bir kişi veya kuruluş hakkında gizlice bilgi toplayan ve bu verileri diğer taraflara ileten bir tür kötü amaçlı yazılımdır. Bazı durumlarda, bunlar reklamverenler veya veri pazarlama firmaları olabilir, bu yüzden casus yazılımlara bazen "reklamveren" denir. Örneğin bir sürücü indirmede, meşru bir programa dâhil olan bir Truva atı veya aldatıcı bir açılır pencere gibi yöntemlerle kullanıcı rızası olmadan kurulur [19].

Casus yazılımlar internet bağlantınızı, adınız, adresiniz, tarama alışkanlıklarınız, tercihleriniz, ilgi alanlarınız veya indirilenler gibi kişisel bilgileri iletmek için kullanır. Diğer casus

yazılım türleri, tarayıcınızı başka bir web sitesine yönlendirmek, cihazınızın aramaları yapmasına veya metinleri otomatik olarak göndermesine veya çevrimdışı olduğunuzda bile can sıkıcı reklamlar sunmasına neden olmak için bilgi kaçırır. Kullanıcı adınızı, şifrenizi veya diğer bilgilerinizi çalan casus yazılımlara "keylogger" denir.

Bir casus yazılım bulaşmasının belirtileri istenmeyen davranışları ve sistem performansının düşmesini içerebilir. CPU kapasitesi, disk kullanımı ve ağ trafiğini tüketebilir. Uygulamaların donması, önyükleme başarısızlığı, internete bağlanma zorluğu ve sistem çökmeleri gibi stabilite sorunları da yaygındır.

Casus yazılım ve kullanıcı gizliliği: Hangi verilerin toplandığını ve kiminle paylaşıldığını tam olarak anladığımız sürece, tüm veri toplama programları casus değildir. Kullanıcı bilgilerini izlemek ve raporlamak meşru yazılım satıcılarının ürünlerini geliştirmelerine veya müşterileri daha iyi desteklemelerine yardımcı olabilir. Bu nedenle, pazarlama şirketleri genellikle hizmetlerinin "casus yazılım" olarak adlandırılmasına itiraz ediyorlar. Yasadışı casus yazılımlarla meşru veri toplama arasındaki çizgi, genellikle kişilerin davranışlarında internet davranışları hakkında bilgi depolamak için iyi bilinen bir yöntem olan çerezlere çekilmektedir. Bazı kullanıcılar çerezlere izin verir; diğerleri ise onlardan nefret eder [19]. Casus yazılım tanımlarıyla ilgili farklı tavırlar, haklarını sorgulayan ve tartışan gizlilik uzmanlarının ayrılmaz bir endişesi haline getirir. Casus yazılımlar neredeyse tamamen düzenlenmemiştir.

Bu programlar, bazı durumlarda kullanıcının yasal olduğu halde, hangi bilgilerin toplandığını ve nasıl paylaşıldığını denetleme ve onaylama mekanizmalarını nadiren kullanır. Buna ek olarak, casus yazılımların bant genişliği, işlem gücü ve bellek gibi bilgi işlem kaynaklarını kontrol etmeden yok ettiği gerçeğini de eklemelisiniz. Böylece güvenlik uzmanlarının neden casus yazılımlara karşı önlem almak ve savunmak istediklerini anlamak çok daha kolaydır.

Casus yazılımları önleme ipuçları: Casus yazılımdan korunmak için tarayıcı kullanın. Kötü amaçlı izleme yazılımlarını tespit etmek için bilgisayarınızı tarayacak birçok casus yazılım önleme programı bulunmaktadır. Bir bilgisayardan veya cihazdan casus yazılımları kaldırmak zor olabilir, ancak artık çalışmayacak şekilde karantinaya alınabilir. Çoğu paket, gelen trafiği tarayarak ve olası tehditleri engelleyerek yeni casus yazılımların gerçek zamanlı kurulumuna karşı sürekli casus yazılım koruması sağlar. Herhangi bir virüsten koruma programında olduğu gibi, casus yazılımdan koruma araçlarının da tam etkili olması için düzenli olarak güncelleştirilmesi gerekir.

Tarayıcı güvenlik ayarları düzenlenmelidir. Çoğu tarayıcı, güvenlik düzeylerini "yüksek" ila "düşük" arasında bir ölçek boyunca ayarlamaya olanak tanır. Bazı tarayıcılar, istenmeyen işlemlere karşı güvenlik duvarı gibi işlev görebileceği gibi, istenirse çerez kurulumuna bile izin verebilir.

Pop-up'lara karşı çok dikkatli olunmalıdır. Açılır pencerelerde, özellikle de beklenmedik şekilde görünen reklamlarda gösterilen reklamlar ve teklifler genellikle aldatıcı amaçları gizler. Bazıları bilgisayarınızda virüs bulaştığını iddia eder veya göz atma deneyiminizi geliştirmek için bir eklenti sunar. Bir pencereyi kapatmak için asla "katılıyorum" veya "Tamam" ı tıklamayın; bunun yerine kapatmak için pencerenin köşesindeki kırmızı "x" işaretini tıklayın. Şüpheci bilgi işlem pratiği yapmalısınız (herhangi bir yeni programın güvenli olduğu kanıtlanana kadar potansiyel olarak zararlı olduğunu varsaymak). Anlamadığınız bir mesaja "evet" cevabı vermek, casus yazılımın yüklenmesine izin verebilir.

"Ücretsiz" diye reklamı yapılan uygulamaların asla "ücretsiz" olmadığını anlamalısınız. Ücretsiz uygulamalara sahip birçok durumda, hizmetler için ticari izlenmeyi açıkça kabul etmiş olursunuz. Uygulama için hedefli reklamlar almayı kabul ederek "ödeme" yaparsınız. Bunun adil bir ticaret olduğuna da karar verebilirsiniz. Ancak çoğu şirketin size hangi reklamları göstereceğini belirlemek için çevrimiçi etkinliklerinizi izlemesi gerekmektedir.

Her zaman şartlar ve koşulları okumalısınız. Meşru yazılım satıcıları, kullanıcı şartları ve koşullarında kullanıcı bilgilerini nasıl topladıkları ve kullandıkları hakkındaki bilgileri açıklayacaklardır. Çoğu kullanıcı bunları okumak için bile zahmet etmemektedir. Çevrimiçi gizliliğinizi korumak konusunda özellikle kararlıysanız, tam olarak ne için kayıt yaptırdığınızı bilmek en iyisidir. Gizlilik ilkeleri kullanıcı bilgisi olmadan kötüye kullanılır veya değiştirilirse, bir yazılım satıcısı asıl amacı ne olursa olsun kullanıcının güvenini ciddi şekilde ihlal edebilir [21].

•Normal bir gün boyunca kaç şirketin sizi takip ettiğini görmek için Collusion vb. bir tarayıcı eklentisi kurmalısınız.

•Tarayıcı çerezleri her gün silinmelidir.

•Ghostery, Disconnect veya Peer Block gibi izleyicileri engelleyen eklentiler kurmalısınız.

•İnternet kullanımınıza başka bir katman eklemek için Sanal Özel Ağ (VPN) kullanmalısınız.

•İnternet gizliliği sorunları oylanmaya hazır olduğunda endişelerinizi yerel temsilcilerinize mutlaka iletmelisiniz.

2.1.6. Botnet

İnternet suçlarıyla ilgili haberlerde genellikle "botlar", "zombiler" ve "botnetler" den bahsederler. Bu bağlamdan hareketle bunların bilgisayar ya da ağ güvenliği tehditleri olduğunu anlamak zor değildir. Fakat bunlar tam olarak nedir, nasıl çalışırlar ve nasıl bir hasara neden olabilirler diye de sorgulamak bizim gibi araştırmacılar için kaçınılmazdır.

"Robot" un kısaltması olan bot, komutta otomatik işlem yapan bir tür yazılım uygulaması veya komut dosyasıdır. Kötü botlar, bir saldırganın etkilenen bir bilgisayarı uzaktan kontrol etmesine izin veren kötü amaçlı görevler gerçekleştirir. Sisteme sızma olduktan sonra, bu makinelere ayrıca zombiler de denebilir.

Bir bilgisayarı ele geçirmek kullanışlı olmasına rağmen, bir bilgisayar korsanı için önemli olan değer, çok sayıda zombi bilgisayarı ağ bağlantılarını kullanarak aynı noktaya bağlamak ve onları tek bir yerden kontrol edebilmektir. Bu ağ türü "botnet" olarak bilinir.

Botnetlerin çalışma sistemi: Botnetler, son on yılda yüz milyonlarca bilgisayarı kontrol altına alan en yaygın kötü amaçlı yazılım dağıtım yöntemlerinden biri olmuştur. Botnet'ler evlerde Nesnelerin İnterneti (IoT) cihazları, kamusal alanlar ve güvenli alanlar gibi yeni teknolojileri etkilediğinden, tehlikeye atılan sistemler, şüphesiz kullanıcıların riskini daha da artırabilir. Sistemler üzerinde küçük gibi görünüp büyük işlemler yaparlar.

Çoğu insan, aldıkları spam'ın, tıpkı kendileri gibi binlerce hatta milyonlarca bilgisayardan geldiğini öğrendiğinde şok olur. Bu bilgisayarların gerçek sahipleri hala bunları kullanabilir ve muhtemelen bilgisayarlarının bazen yavaş göründüğü durumlar dışında, bir şeylerin yanlış olduğunu kesinlikle bilmiyorlardır. Çoğu botnet'te son derece küçük bir ayak izi vardır, bu da sisteminizi yavaşlatır veya çok fazla sistem kaynağını kullanır; bu nedenle, makinenizin bir bilgisayar korsanı tarafından kötü amaçlar için kullanıldığını anlamak zor olabilir. Ayrıca, genellikle kendilerini maskeleme kabiliyetine sahiptirler, böylece fark edilmeden büyük çaplı saldırılar da yapabilirler.

Açık kaynaklı ve güvenli olmayan cihazları tehlikeye atarlar. 2016 yılında keşfedilen bir botnet olan Mirai, öncelikle kameralar ve internet yönlendiricileri dâhil olmak üzere IoT cihazlarına saldırdı. Temel olarak, Mirai kötü amaçlı yazılımını bulaştıran cihazlar, IoT cihazlarını bulmak için interneti tarayan botlar haline geldi. Mirai daha sonra bu cihazlara sızmak ve bulaşmak için cihaz üreticileri tarafından ayarlanan varsayılan ortak kullanıcı adlarını ve şifreleri kullanırdı. Çoğunlukla, virüslü cihazlar, dağıtılmış ağ (DDoS) saldırılarında kullanılsalar bile normal şekilde çalışmaya devam edebilirler. Bu sadece korumasız bir sistemde, internet bağlantılı bilgisayar veya başka bir cihaz için çok kısa sürer. Zararlı yazılım bulaşmış bir sistemde her türlü bilgisayar ve akıllı telefon kullanıcısı için kritik ihtiyaçları kullanan bir bot, sürekli güncel olmak amacıyla sistem içindeki kritik noktalarda kullanılan internet güvenlik yazılımı da dâhil tüm bunların ve fabrika ayarlarındaki varsayılan kullanıcı adları ile şifrelerini her zaman değiştirir [22].

Botnet saldırısının bilgisayar korsanları tarafından kullanılması: Botnet saldırıları, bilgisayar korsanları tarafından oluşturulan bir düzenle birbirlerine bağlantılı bilgisayarlar üzerinden yine kendileri tarafından kötü emeller için kullanılan sistemlerdir. Bu bölümde ise bu kullanım amaçlarından bahsedilecektir.

Finansal ve kişisel bilgileri çalmak: Bilgisayar korsanları, spam, kimlik avı veya diğer aldatmacaları göndermek için botnet'leri kullanabilir ve tüketicileri zor kazanılmış paralarından

vazgeçmeleri için kandırabilirler. Ayrıca, bot bulaşmış makinelerden bilgi toplayabilir sonra da kimlik bilgilerini çalmak ve kullanıcı adı altında borç ya da ücret almak için kullanabilirler.

Yasal web servislerine saldırmak: Bilgisayar korsanları, botnetlerini, meşru bir hizmeti veya girişleri sıkışık trafik yoğunluğuyla dolduran DoS ve DDoS saldırıları oluşturmak için kullanabilirler. Bu durum, şirketin hizmetini veya ağının yanıt verme yeteneğini ciddi şekilde yavaşlatabilir veya şirketin hizmetini ya da ağını tamamen sıkıştırıp kapatabilir.

Mağdurlardan para çekmek: DoS saldırılarından elde edilen gelir, gasp (sitenizin ödemesini alması veya alınması) veya bir şirkete ya da ağa zarar vermekle ilgilenen grupların ödemeleri yoluyla gelir. Bu gruplar arasında "bilgisayar korsanları" da dâhil olmak üzere siyasi gündemde olan korsanların yanı sıra yabancı askeri ve istihbarat örgütleri de vardır.

Zombi ve botnet sistemlerinden para kazanmak: Bilgisayar korsanları botnetlerini; spam göndermek, dolandırıcılık yapmak, kimlik avı yapmak, kimlik çalmak ve meşru web siteleri ile ağlara saldırmak isteyen diğer suçlulara da kiralayabilirler.

Botnet saldırısını önlemenin ipuçları: Güvenlik yazılımı yüklemediyseniz eğer açık ve güncel tutulduğundan da emin olduysanız, makinenize her türlü kötü amaçlı yazılım bulaşmış olabilir. Sistemlerinizi botnet sızıntısından korumak için atmanız gereken birkaç adım verilmiştir:

•Virüsten koruma ve casus yazılım önleme programlarınızı otomatik olarak güncelleştirilecek şekilde ayarlamalısınız.

•Tarayıcı ve işletim sistemi güncellemelerini ve yamalarını düzenli olarak kontrol etmelisiniz.

•Sadece kaynağa güveniyorsanız, internet bağlantılarını tıklayıp veya e-postaları açmalısınız.

Bilinmeyen sitelerden veya güncel korumaya sahip olmayan arkadaşlardan içerik indirirken ve virüslü dosyaları istemeden diğer kullanıcılara geçirirken sık karşılaşılan kullanıcı riskleri ortaya çıkar. Başka insanlara ait dosyaları indirdiğinizde, zararlı kod karantinaya alınmış ve sistem bunu kaldırmaya çalışmış olabilir ancak kötü amaçlı yazılım, zayıf güvenlikli kontrol noktalarını atlatmayı da başarabilir. Bilgisayarı korunmayan birinden bilgi veya dosya indirirken daima çok dikkatli olmalısınız.

Kötü amaçlı yazılım geliştiriciler, her zaman güvenlik önlemlerini aşmanın yeni yollarını arar ve sizin veya bilgisayarı veya sistemi kullanan başka bir kişi tarafından gerçekleştirilen eylemler nedeniyle enfeksiyon riski vardır. Yanlışlıkla bir bağlantıya tıklarsanız, bir dosyayı indirirseniz veya makinenize bulaşmasına izin verebilecek diğer işlemleri gerçekleştirseniz bile virüsleri ve

diğer kötü amaçlı yazılımları algılayan ve de onları durdurabilen gelişmiş internet güvenlik yazılımı kullandığınızdan emin olmalısınız.

2.1.7. Solucan

Bir bilgisayar solucanı, birincil işlevi diğer bilgisayarlara bulaşırken virüslü sistemlerde aktif kalmak amacında olan bir tür kötü amaçlı yazılım programıdır. Bir bilgisayar solucanı, virüs bulaşmamış bilgisayarlara yayılması için kendisini kopyalayan, kendi kendini kopyalayarak da yayılan kötü amaçlı bir yazılımdır. Solucanlar, genellikle otomatik ve kullanıcıya görünmeyen bir işletim sisteminin parçalarını kullanır. Solucanların yalnızca kontrolsüz çoğalmaları sistem kaynaklarını kullandığında, diğer işleri yavaşlatırken veya durdurduğunda fark edilmesi daha yaygındır. Bir bilgisayar solucanı WORM ile karıştırılmamalıdır [19].

Bilgisayar solucanlarının tarihi: Her ne kadar 1988'de salınan Morris solucanı, ilk bilgisayar solucanı olarak kabul edilse de, aslında günümüz dünyasında ve daha sonra ortaya çıkan internette yaygın olarak yayılan ilk solucan olarak nitelendirilmektedir.

Morris solucanı, Internet öncü ağı ARPANET'e bağlı tüm sistemleri numaralandırmaya çalışan Cornell lisansüstü öğrencisi Robert Tappan Morris Jr.'ın eseriydi. Birçok farklı Unix programındaki güvenlik açıklarını hedef alan Morris solucanı, bir sistemi bir kereden fazla etkileme yeteneğine sahipti ve virüs bulaşmış ana bilgisayarda hizmet reddi durumu yaratmadan önce tamamen ortadan kaldırılmasını zorlaştırdı. 60.000 sistemin %10'unun ARPANET'e bağlı olduğuna inanılan solucandan etkilendi. Bugüne kadarki en zarar verici bilgisayar solucanlarından biri, ILOVEYOU virüsü, metin dosyaları gibi görünen e-posta ekleri, IM sohbet oturumlarında çalıştırılan komut dosyaları ve virüsün adlarıyla yeniden adlandırılan çalıştırılabilir dosyalardaki kopyaları da içeren ortak sistem dosyalarına sahip kötü amaçlı yazılımlardı. ILOVEYOU virüsü, öncelikle hedeflenen mağdurlar bir e-posta eki açtığında yayıldı ve kötü amaçlı yazılım, kurbanın Microsoft Outlook'taki tüm temaslarına karşı kendisini kızdırdı. Teknik olarak, solucanın bu yönü kullanıcı etkileşimi gerektirse de, genel etki virüsün masaüstü bilgisayarların normal çalışması sırasında ve kurbanların ilk farkındalığı olmadan yayılmasıydı. Kötü amaçlı yazılımın, 4 Mayıs 2000'de 45 milyon kadar kullanıcısını etkilediği ve Ford Motor Company de dâhil olmak üzere bazı işletmelerin e-posta hizmetlerini kapatmaya zorlandığı bildirildi [23].

Bilgisayar solucanların yayılması: Bir bilgisayar solucanı enfeksiyonu kullanıcı etkileşimi olmadan yayılır. Gerekli olan tek şey, bilgisayar solucanının virüslü bir sistemde aktif hale gelmesidir. Ağların yaygın olarak kullanılmasından önce, bilgisayar solucanları, bir sisteme monte edildiklerinde kurban sistemine bağlı diğer depolama aygıtlarına bulaşacak olan disketler gibi virüslü depolama ortamları üzerinden yayılmıştır. USB sürücüler hala bilgisayar solucanlar için

ortak bir vektördür. Bilgisayar solucanları genellikle, çoğalmak için ağ protokollerinin eylemlerine ve kırılganlıklarına güvenir. Örneğin, WannaCry, Ransomware solucanı, Windows işletim sisteminde uygulanan Sunucu İleti Bloğu (SMBv1) kaynak paylaşım protokolünün ilk sürümünde bir güvenlik açığından yararlanmıştır [23].

Solucan tarafından yapılan SMBv1 isteklerine yanıt veren sistemler: Yeni virüs bulaşmış bir bilgisayarda etkin hale geldiğinde, WannaCry kötü amaçlı yazılımı yeni potansiyel mağdurları aramaya başlar. Solucan bu şekilde bir organizasyon içinde yayılmaya devam edebilir. Kendi cihazınızı getirdiğinizde bile solucan virüsü diğer ağlara yayılabilir. E-posta solucanlar, bir kullanıcının kişi listesindeki tüm adreslere giden mesajları oluşturarak ve göndererek yayılır.

Bugüne kadarki en kötü bilinen bilgisayar solucanlarından biri olan Stuxnet, virüs bulaşmış USB cihazlarının paylaşılması yoluyla kötü amaçlı yazılımın yayılması için kullanılan bir solucan bileşeni olarak bilinmektedir. Bunun yanı sıra endüstriyel ortamlarda yaygın olarak kullanılan denetleyici elektrik kontrol hizmetleri, su temini hizmetleri, kanalizasyon tesisleri ve diğer yerler de dâhil olmak üzere veri toplama sistemlerini hedef alan kötü amaçlı yazılım bileşenlerinden oluşur.

Bilgisayar solucanı çeşitleri: Saf bilgisayar solucanı, kendilerini virüslü sistemlerden etkilenmeyen sistemlere yaymaktadır. Bu, bilgisayar solucanlarının zarar görme potansiyelini en aza indirmez. Bazı işlemler, virüslü bir sistem nedeniyle kullanılamayabilir veya güvenilmez duruma gelebilir. Solucanın yayılmasıyla ilişkili bilgisayar ek yükünün hesaplanırken, bilgisayar solucanlarının ayrıca solucan yayılımı ile bağlantılı kötü amaçlı trafiğe sahip ağ bağlantılarının doygunluğu yoluyla ağ kurmayı bozduğu da bilinmektedir.

Bir bot solucanı, bilgisayarlara enfekte olarak ve onları botnet'ler aracılığıyla koordine edilmiş saldırılarda kullanmak amacıyla zombilere veya botlara dönüştürmek için kullanılabilir. Anlık mesajlaşma veya IM solucanlar, anlık mesajlaşma servisleri aracılığıyla yayılır ve mağdur bilgisayarlardaki irtibat listelerine erişimden yararlanır.

E-posta solucanları genellikle sıradan e-posta iletilerinin göründüğüne ekli kötü amaçlı yürütülebilir dosyalar olarak yayılır. E-posta solucanı, virüs bulaşmış bir sistemi kullanıcı iletişim listelerindeki e-posta adreslerine yeniden göndermeye zorlayarak yayılır; solucan, e-posta alıcıları dosyayı açtığında yeni sistemlere bulaşmaktadır. Başarılı e-posta solucanları, genellikle ekli dosyayı açmalarını istemek için sosyal mühendislik yöntemlerini kullanır. Etik bir solucan, bilinen güvenlik açıklarına yönelik düzeltme ekleri sağlama amacıyla ağlar arasında yayılmak üzere tasarlanmış bir bilgisayar solucanıdır.

Etik solucanlar Akademi'de tanımlanmış ve tartışılmış olsa da, günümüz dünyasında gerçek örnekler bulunamamıştır, büyük olasılıkla bu tür bir yazılıma beklenmedik şekilde tepki veren sistemlere beklenmeyen zarar verme potansiyeli, güvenlik açıklarını kaldırma potansiyelinden daha

ağır basmaktadır. Her durumda, sistem sahibinin izni olmadan sistemde değişiklik yapan herhangi bir yazılımın serbest bırakılması, yayıncısını çeşitli cezai ve hukuki suçlamalara açık hale getirir [23].

Solucanlar ve virüsler arasındaki farklar: 1996 yılında Carnegie Mellon Üniversitesi Yazılım Mühendisliği Enstitüsü'nün CERT Bölümü tarafından yayınlanan "İnternet Güvenliği" raporunda tanımlandığı gibi, bilgisayar solucanları "başladıktan sonra insan müdahalesi olmadan yayılan kendi kendini kopyalayan programlardır" [23]. Ancak genel anlamda yanlışlıkla diğer programlara veya sistemlere yayılmak için kullanıcının bazı işlemlerini gerektirir.

Bir bilgisayar solucanı yüklendikten ve yeni virüs bulaşmış bir sistemde çalışmaya başladıktan sonra, genellikle ana direktifi olan "virüslü bir sistemde mümkün olduğu kadar uzun süre aktif kalmak ve mümkün olan en fazla diğer savunmasız sistemlere yayılmak." sürecini izler.

Bilgisayar solucanlarının önlenmesi, tespiti ve uzaklaştırılması: Kullanıcılar, kendilerini bilgisayar solucanı bulaşmasına karşı korumak için iyi siber güvenlik korumasını uygulamalıdır. Bilgisayar solucanı enfeksiyonlarının önlenmesine yardımcı olacak önlemler şunları içerir:

• İşletim sistemleri ve diğer tüm yazılım yamaları ve güncellemeleri hakkında bilgi sahibi olmak, yeni keşfedilen güvenlik açıkları nedeniyle riski azaltmaya yardımcı olur.

• Güvenlik duvarlarının kullanılması, kötü amaçlı yazılımların sistemlere erişimini azaltırken, virüsten koruma yazılımı kullanmak kötü amaçlı yazılımların çalışmasını önlemeye yardımcı olur.

• Sistemleri kötü amaçlı yazılıma maruz bırakabilecek e-posta veya diğer mesajlaşma uygulamalarındaki bağlantılara dikkat edin. Aynı şekilde, bilinmeyen gönderenlerden gelen iletilerin ekleri de genellikle kötü amaçlı yazılımları dağıtmak için vektörler olarak kullanılır.

Her ne kadar bazı solucanlar kendilerini yeni mağdur sistemlerine yaymaktan başka bir şey yapmamak için tasarlanmış olsalar da, çoğu solucanlar virüsler, rootkitler veya diğer zararlı yazılımlarla ilişkilidir. Bir bilgisayar solucanını çıkarmak için ilk adım, zor olabilecek solucanın varlığını tespit etmektir. Bir solucanın varlığını gösterebilecek bazı faktörler şunlardır:

• Düşük sistem performansı, sistemin donması veya beklenmedik şekilde çökmesini de içeren bilgisayar performansı sorunları.

• Kullanıcı etkileşimi olmadan yürütülen veya sonlanan programlar dâhil, olağandışı sistem davranışı; sıra dışı sesler, görüntüler veya mesajlar; yabancı dosya ya da simgelerin ani görünümü, dosya veya simgelerin beklenmedik şekilde kaybolması; işletim sisteminden ya da virüsten koruma yazılımından gelen uyarı mesajları; ve kullanıcı işlemi yapmadan kişilere gönderilen e-posta mesajlarıdır.

Bir bilgisayar solucanını silmek zor olabilir. Aşırı tehlikeli durumlarda, sistemin biçimlendirilmesi ve tüm yazılımların yeniden yüklenmesi gerekebilir. Sisteme bulaşan bilgisayar solucanını belirlemek mümkün ise, enfeksiyonu gidermek için özel talimatlar veya araçlar olabilir. Ancak, bilgisayar solucanını silmeye çalışmadan önce sistem internet ya da kablolu ya da kablosuz olan herhangi bir ağdan kesilmelidir; çıkarılabilir depolama aygıtları da bulaşma risklerine karşı ayrı ayrı çıkarılmalı ve taranmalıdır.

2.1.8. Rootkit

Rootkit, bilgisayar korsanları tarafından bir bilgisayara veya ağa sürekli yönetici seviyesine erişim sağlamak için kullanılan bir yazılımdır. Bir rootkit, çalınan bir parola yoluyla veya kurbanın rızası ya da bilgisi olmadan tipik olarak bir sistemin güvenlik açığından yararlanarak kurulur [24].

Rootkitler öncelikle kullanıcı modu uygulamalarını hedef alır, ancak aynı zamanda bir bilgisayarın hipervizörüne, çekirdeğine ve hatta donanım yazılımına da odaklanırlar. Rootkit'ler, virüslü bir bilgisayarda kurulu olan anti-kötücül yazılım yazılımını tamamen devre dışı bırakabilir veya imha edebilir, böylece bir rootkit saldırısının izlenmesi ve ortadan kaldırılması zorlaşır. İyi yapıldığında, izinsiz giriş gizliliği saklanabilir, böylece sistem yöneticileri bile bu durumdan haberdar olmaz. Hedef sisteme yüklenebilecek birkaç rootkit türü vardır:

Kullanıcı modu veya Rootkit uygulaması: Bunlar paylaşılan bir kütüphaneye kurulur ve uygulama ile API davranışını değiştirebilecekleri uygulama katmanında çalışır. Kullanıcı modu rootkitlerinin antivirüs programları ile aynı katmanda çalıştıkları için fark edilmeleri kolaydır.
Çekirdek modu: Bu rootkitleri, tüm sistem işlemlerini denetleyebilecekleri bir işletim sisteminin çekirdek modülünde uygulanır. Tespit edilmesi zor olmasının yanı sıra, çekirdek modu rootkit'leri hedef sistemin kararlılığını da etkileyebilir.

Bootkits: Bu rootkits, ana önyükleme kaydına (MBR) bulaşarak hedef sistemin kontrolünü ele geçirir. Bootkits, kötü amaçlı bir programın hedef işletim sistemi yüklenmeden önce çalışmasına izin verir.

Üretici (Firmware) rootkitleri: Bu rootkit'ler yönlendiriciler, ağ kartları, sabit sürücüler veya sistem BIOS'u gibi cihazları çalıştıran yazılıma erişim sağlar.

Rootkit hipervizörleri: Bu rootkit'ler bir makinenin kontrolünü kazanmak için donanım sanallaştırma özelliklerinden yararlanır. Bu, çekirdeği atlayarak ve hedef işletim sistemini sanal bir makinede çalıştırarak yapılır. Hipervizörlerin, işletim sisteminden daha yüksek bir seviyede

çalıştıkları ve hedef işletim sistemi tarafından yapılan tüm donanım aramalarını engelleyebileceği için tespit etmek ve temizlemek neredeyse imkânsızdır.

Sistemlerinizi rootkitlerden korumak, mevcut kötü amaçlı yazılımları taramak ve yeni programların kurulmasını engellemek için iki yönlü bir işlem uygulanır ve bu *Anti-Rootkit* önlemleri adı altında gerçekleşir.

Rootkit tarayıcıları: Tarayıcılar, aktif rootkit'leri ayıklamak için bir sistemi ayrıştırmak için tasarlanmış programlardır. Bunlar, uygulama katmanı rootkitlerini algılamaya ve kaldırmaya yardımcı olsa da, genellikle çekirdek, önyükleme veya ürün yazılımı düzeyinde çalışanlara karşı etkisizdir. Çekirdek düzeyinde kötü amaçlı kod arayabilen tarayıcılar yalnızca rootkit etkin değilken çalışabilir. Bu, bir sistemin etkili olması için sistem işlemlerinin durmasıyla güvenli modda başlatılması gerektiği anlamına gelir. Bu sınırlamalar nedeniyle, güvenlik uzmanları, herhangi bir araç ile bir sistemin tamamen temiz olduğunu garanti edemediğinden, birçok tarayıcı ve rootkit engelleyici kullanılmasını önermektedir. Sisteminizi önyükleme, bellenim veya hipervizör düzeyinde çalışan rootkitlerden tamamen korumak için tek çözüm verileri yedeklemektir, ardından cihaz silinmeli ve temiz bir kurulum gerçekleştirilmelidir.

Rootkitleri engelleme: Rootkit önleme, bir rootkit'in sisteminize hem bireysel kullanıcılar hem de web'e bakan varlıklar (yani web siteleri) aracılığıyla iletilebileceği fikrine dayanır.

İlk önleyici tedbir, kuruluşunuzdaki herkes için kullanıcı eğitimidir. Bu, zararlı bağlantıların ve e-posta eklerinin nasıl tespit edileceğine ilişkin talimatların yanı sıra bilinmeyen kaynaklardan dosya indirme veya açma kurallarını da içermelidir.

Kullanıcılar ayrıca, kötü niyetli iletilerin, web sitelerinin veya dosyaların gizlice meşru kaynaklardan geldiği görünen kimlik avı girişimlerini tanımlamak ve önlemek için de eğitilmelidir. Bu, özellikle yönetici ayrıcalıklarına sahip kullanıcılar için oldukça önemlidir.

Rootkitlerden korunmak için alınması gereken ek önlemler aşağıdaki gibi verilmiştir.

• Yazılımların güncel tutulması ve uygulamalarda ve işletim sistemlerinde bilinen güvenlik açıklarının düzeltilmesi.

• Anti-virüs çalıştırma ve zaman zaman hassas makinelerde anti-rootkit araçları çalıştırma.

• API çağrılarının veya CPU kullanımının şüpheli kalıplarını keşfetmek için sistem davranışını analiz eden ve bir rootkiti gösterebilecek davranış tabanlı tespit.

• Uzaktan kontrol merkeziyle iletişim kuran rootkitleri tanımlamak için paket analizörlerinden, güvenlik duvarlarından veya diğer ağ araçlarından ağ günlüklerinin yakından incelenmesi.

2.1.9. Fidye Yazılımları

Fidye yazılımları literatürde ransomware olarak adlandırılmaktadır, fidye yazılımı olarak bilinir. Bir kurbanın bilgisayarındaki verilerin genellikle şifreleme yoluyla kilitlendiği ve verilerin şifresi çözülmeden ve erişim kurbana geri verilmeden önce ödeme(fidye) talep edilmesini gerektiren kötü amaçlı bir yazılım alt kümesidir. Fidye yazılımı saldırıları için sebep her zaman parasaldır ve diğer saldırı türlerinden farklı olarak, mağdurlara genellikle bir istismarın gerçekleştiği ve saldırıdan nasıl kurtulacağı konusunda talimatlar verildiği bildirilir. Ödeme, genellikle siber suçlu kimliğinin bilinmemesi için Bitcoin gibi sanal bir para biriminde talep edilir.

Ransomware, kötü amaçlı e-posta ekleri, virüslü yazılım uygulamaları, virüslü harici depolama aygıtları ve güvenli web siteleri aracılığıyla yayılabilir. Saldırılar ayrıca uzak masaüstü protokolünü ve herhangi bir kullanıcı etkileşimi yöntemine dayanmayan diğer yaklaşımları da kullanmıştır.

Fidye yazılım saldırılarının çalışması: Derin ağdaki Ransomware kitleri, bilgisayar korsanlarının belirli yeteneklere sahip fidye yazılımı oluşturmak için bir yazılım aracı satın almalarını ve kullanmalarını sağlamıştır. Daha sonra bu kötü amaçlı yazılımı kendi dağıtımları için ve bitcoin hesaplarına ödenen fidye ile oluşturabilirler. BT dünyasının geri kalanının çoğunda olduğu gibi, teknik geçmişi az olan veya hiç olmayan kişilerin bir hizmet olarak ucuz fidye yazılımı sipariş etmesi (RaaS) ve minimum çabayla saldırılar başlatması mümkündür. Bir RaaS senaryosunda, sağlayıcı fidye ödemelerini toplar ve hasılatı hizmet kullanıcısına dağıtmadan önce belirli bir yüzdeyi kendisi alır.

Fidye yazılım türleri: Saldırganlar, dijital para birimlerini kurbanlarından zorla almak için birkaç farklı yaklaşımdan birini kullanabilirler.

Örneğin Scareware olarak bilinen Fidye yazılımı, güvenlik yazılımı veya teknik destek olarak dener ve kendini belli eder. Mağdurlar, sistemlerinde kötü amaçlı yazılım bulunduğunu belirten pop-up bildirimler alabilir (ki bu gibi yeni durumlar için bir güncellemeye sahip olmayan güvenlik yazılımı bu bilgilere erişemez). Buna cevap vermemek daha fazla pop-up'a yol açmak dışında hiçbir şey yapmaz [25].

Ekran dolapları olarak bilinen virüs türleri, bir kullanıcıyı bilgisayarından tamamen kilitlemek için tasarlanmış bir fidye yazılımı türüdür. Bilgisayarı başlattıktan sonra bir mağdur, resmi bir hükümet mührü gibi görünen şeyleri görebilir ve mağdurun resmi bir soruşturma konusu olduğuna inanmasına neden olabilir. Lisanssız yazılım veya yasadışı web içeriğinin bilgisayarında bulunduğu bildirildikten sonra, kurbana elektronik para cezasının nasıl ödeneceğine ilişkin talimatlar verilmiştir. Ancak, resmi hükümet kuruluşları bunu yapmaz; bunun yerine uygun yasal kanallardan ve prosedürlerden geçeceklerini bilmeleri gerekir.

Fidye yazılımı veya veri kaçırma saldırılarını şifreleyen saldırgan, mağdurun verilerine erişip şifreleyecek ve dosyaların kilidini açmak için bir ödeme isteyecektir. Bu gerçekleştiğinde, mağdurun pazarlık yapsalar bile verilerine geri erişebileceği garantisi yoktur. Fidye yazılımını şifrelemeye benzer şekilde, saldırgan ayrıca virüslü cihazlardaki dosyaları da şifreleyebilir ve mağdurun dosyaların kilidini açmasına yardımcı olacak ve gelecekteki kötü amaçlı yazılım saldırılarını engellemeye yardımcı olacak bir ürün satarak para kazanacaktır.

Doxware'de, saldırgan bir fidye ödememesi durumunda verilerinizi çevrimiçi yayınlamakla tehdit edebilir. Mobil fidye yazılımı mobil cihazları etkileyen fidye yazılımıdır. Saldırgan, bir telefondan veri çalmak veya kilitlemek için mobil fidye yazılımını kullanabilir ve verileri iade etmek veya cihazın kilidini açmak için bir fidye gerektirebilir. Mağdur ayrıca açılır bir mesaj veya e-posta ile fidye notu alabilir; talep edilen miktar belirli bir tarihe kadar ödenmezse, cihazın kilidini açmak veya dosyaların şifresini çözmek için gereken özel anahtarın imha edileceğini bildiren bir not alabilirsiniz.

Bu saldırıların ilk örneklerini bazen sadece web tarayıcısına veya Windows masaüstüne erişimi "kilitledi" uyarısıyla (bunu oldukça kolay bir şekilde tersine mühendislik ve tekrar açılabilecek şekilde) yaptılar. Bilgisayar korsanları o zamandan beri, bilgisayardaki dosyalara erişimi engellemek için güçlü, açık anahtarlı şifreleme kullanan fidye yazılımı sürümleri oluşturdu.

Fidye yazılımı kaldırma yöntemleri ve gelişimi: Bir mağdurun bir fidye yazılım saldırısını durdurabileceği ve verilerini geri alabileceği konusunda hiçbir garanti yoktur; ancak, bazı durumlarda işe yarayabilecek yöntemler vardır. Örneğin, bir mağdur sistemi güvenli modda durdurabilir ve yeniden başlatabilir, bir anti-kötücül yazılım programı kurabilir, bilgisayarı tarayabilir ve bilgisayarı virüssüz bir duruma geri yükleyebilir. Mağdurlar ayrıca sistemlerini ayrı bir diskte depolanan yedeklerden geri alabilirler. Bulutta varsa, kurbanlar disklerini yeniden biçimlendirebilir ve önceki bir yedekten geri yükleyebilirler.

En bilinen ransomware çeşitleri, CryptoLocker ve WannaCry virüsleridir. Belki de genel anahtar şifrelemesini kullanan yaygın bir saldırı atağının ilk örneği, İnternet üzerinde eylül ayından ertesi yılın mayıs ayına kadar aktif olan bir Truva atı virüsü olan Cryptolocker oldu. Kötü amaçlı yazılım, Bitcoin veya ön ödemeli bir kuponla ödeme yapılmasını istedi ve uzmanlar genel olarak, doğru bir şekilde uygulandığında kullanılan RSA şifrelemesinin esasen aşılamaz olduğuna inanmaktaydılar. Bununla birlikte, Mayıs 2014'te bir güvenlik şirketi saldırı tarafından kullanılan bir komut ve kontrol sunucusuna erişim sağladı ve saldırılarda kullanılan şifreleme anahtarlarını kurtardı. Saldırıyı etkili bir şekilde engellemek için ücretsiz anahtar kurtarmaya izin veren çevrimiçi bir araç kullanıldı.

2017 yılının Mayıs ayında, WannaCry adlı bir saldırı dünya çapında çeyrek milyondan fazla sisteme sızdı ve verileri şifreleyebildi. Kötü amaçlı yazılım asimetrik şifreleme kullanır, böylece

mağdurun fidye dosyalarının şifresini çözmek için gereken (özel ve dağıtılmamış) anahtarı kurtarması beklenemez. Ödemeler Bitcoin üzerinden talep edildi; bu, fidye ödemelerinin alıcısının tespit edilememesi, ancak işlemlerin görünür olması ve dolayısıyla genel fidye ödemelerinin yerine getirilmesi anlamına gelmekteydi. WannaCry'nin en etkin olduğu hafta boyunca, sadece yaklaşık 100.000 dolarlık bitcoin transfer edildi (Ödeme yapıldıktan sonra şifresi çözülmüş hiçbir veri hesabı yok). WannaCry'nin etkisi bazı durumlarda belirgindi. Örneğin, İngiltere'deki Ulusal Sağlık Servisi ağır şekilde etkilenmiş ve saldırı sırasında hizmetlerin etkin bir şekilde çevrimdışı duruma getirilmesi için zorlanmıştır. Yayınlanan raporlar, etkilenen binlerce şirketin neden olduğu zararın 1 milyar doları aşabileceğini ileri sürdü.

Symantec 2017 İnternet Güvenliği Tehdit Raporu'na göre, talep edilen fidye miktarı 2016 yılında önceki iki yıldan yaklaşık üç katına çıkmış ve ortalama talep toplamda 1.077 ABD Doları olmuştur. Genel olarak, bu taleplerin ne sıklıkla karşılandığını söylemek zordur [25]. IBM tarafından yapılan bir araştırma, ankete katılan yöneticilerin %70'inin bir fidye yazılımı talebinde bulunduğunu belirtti, ancak Osterman Research tarafından yapılan bir araştırma ABD merkezli şirketlerin yalnızca %3'ünün ödeme yaptığını buldu (diğer ülkelerdeki oranlar oldukça yüksek olmasına rağmen). Çoğunlukla, ödemenin işe yaramadığı görülmekte, ancak risk olmadan da yapılamazdı. 2016 tarihli bir Kaspersky Güvenlik Bülteni, talep ettiği fidyeyi ödemeyi tercih eden işletmelerin % 20'sinin dosyalarını geri almadığını iddia etmekteydi.

Nesnelerin İnterneti (IoT) için de fidye yazılımı çok geride olmayabilir. İki araştırmacı, Andrew Tierney ve Ken Munro, 2016 Def Con konferansında genellikle mevcut bir akıllı termostat için saldırılarda, sistemi kilitleyen ve bir bitcoin fidye isteyen kötü amaçlı yazılımları gösterdi.

Symantec'in Şubat 2019'da yayımlanan yıllık raporuna göre ise, 2018 yılında Amazon'un neredeyse her şeyi kaydedebilen, veri almak ve depolamak için oluşturulmuş bir nesne depolama hizmeti olan S3 (Simple Storage Service) kovalarından (bucket) çalınan 70 milyondan fazla kayıt ve veri var [25].

Ransomware (fidye yazılımı) ve cryptojacking (tarayıcılar üzerinden bilgisayarlara ait işlemci gücünü kullanıp kripto para madenciliği yapmayı sağlayan yöntemler) ise 2018'de daha az getiri sağladığı için daha az kullanıldı. Fidye yazılımı (cihazları ele geçirip fidye isteme) 2013'den bu yana ilk kez düşüşe geçse de işletmeler için yüzde 12 arttı.

Fidye yazılım saldırılarını önleme: Fidye yazılımı saldırılarına ve diğer siber zorunluluklara karşı korunmak için uzmanlar, kullanıcıları bilgisayar aygıtlarını düzenli olarak yedeklemeleri ve virüsten koruma yazılımı da dâhil olmak üzere yazılımları düzenli olarak güncellemeleri için teşvik eder. Son kullanıcılar yabancılardan gelen e-postalardaki linklere tıklamaktan veya e-posta eklerini açmaktan sakınmalıdırlar. Kurbanlar fidye ödememek için ellerinden geleni yapmalılar.

Fidye yazılımı saldırılarını durdurmak neredeyse imkânsız olsa da, bireylerin ve kuruluşların hasarın minimum ve kurtarma işlemlerinin olabildiğince çabuk olmasını sağlamak için alabilecekleri önemli veri koruma önlemleri vardır. Stratejiler, kimlik doğrulama sistemlerini ve etki alanlarını bölümlere ayırma, anlık güncel depolama görüntülerini birincil depolama havuzunun dışında tutma ve verilere kimin erişebileceği ve ne zaman erişime izin verilebileceği konusunda zorlu sınırlar koymayı içerir.

2.2. Hacker Araçları

Geçmişte, bilgisayar korsanları sıfırdan kötü amaçlı yazılımlar ürettiler ancak bugün çok yetenekli ve aynı zamanda suç aracı olarak da bilinen saldırı kitleri kullanmaktalar. Saldırı kitleriyle, bilgisayar korsanları kendi kötü niyetli amaçlarını gerçekleştirmek için kurbanın bilgisayarına zarar veren saldırıları kolayca ve otomatik olarak başlatabilir.

Bu araç setlerinden bazıları ücretsiz, açık kaynaklıdır ve diğerleri de İnternet'te satılmaktadır. Bu bölümde, hedeflenen kullanıcıları, bilgisayarları veya ağları tehlikeye atmak için yoğun olarak kullanılan mevcut hacker araçlarından kısaca bahsedeceğiz.

2.2.1. John the Ripper

John the Ripper, birkaç farklı crack programını birleştiren ve hem kaba kuvvet hem de sözlük saldırı modlarında çalışan popüler bir açık kaynaklı şifre kırma aracıdır.

John the Ripper, kurumda ağ güvenliğini tehlikeye sokabilecek zayıf şifreleri ve diğer idari amaçları tespit etmek için sıklıkla kullanılır. Yazılım, her işletim sistemindeki çeşitli kullanıcı hesaplarına karşı çok çeşitli şifre kırma teknikleri çalıştırabilir ve yerel olarak veya uzaktan çalıştırmak için komut dosyası yazılabilir.

Başlangıçta Unix'ten türetilen sistemler için geliştirilen John the Ripper, çoğu yaygın platform için kullanılabilir. Serbest ve açık kaynaklı (FOSS) sürüm genellikle kaynak kodu olarak dağıtılır. Ticari bir versiyon olan John the Ripper Pro, belirli bir sistem için yerel kod olarak dağıtılan daha kullanıcı dostu bir versiyondur [26]. Bu şifre kırıcı, hemen hemen her şifrede kullanılan şifreleme türünü otomatik olarak algılayabilir ve şifre testi algoritmasını buna göre değiştirerek onu şimdiye kadarki en akıllı şifre kırma araçlarından biri yapar.

Bu etik korsanlık aracı, aşağıdaki gibi şifreleri ve algoritmaları çözmek için kaba kuvvet (Brute Force) teknolojisi kullanır:

- DES, MD5, Blowfish
- Kerberos AFS
- Karma LM (Lan Yöneticisi), Windows NT / 2000 / XP / 2003'te kullanılan sistemler
- MD4, LDAP, MySQL (üçüncü parti modülleri kullanarak)

Diğer bir avantaj ise, JTR'nin açık kaynaklı, çok platformlu ve Mac, Linux, Windows ve Android için tamamen kullanılabilir olmasıdır.

2.2.2. Nmap

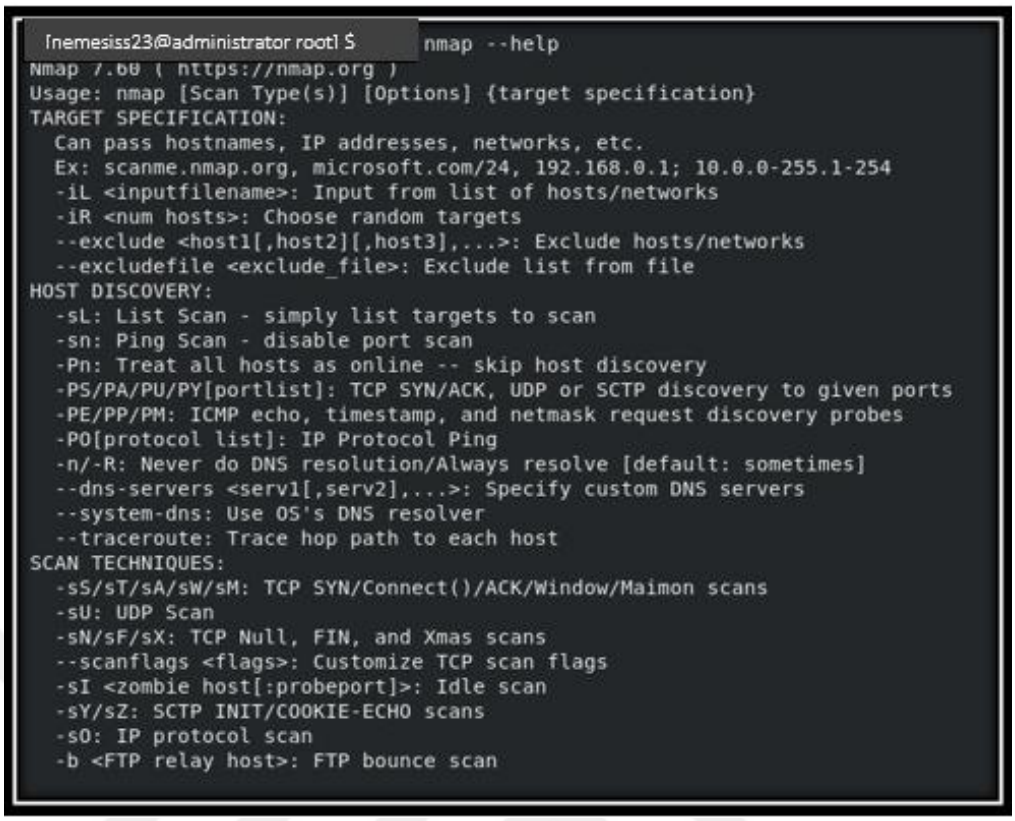
Nmap, bilgisayar korsanları veya BT yöneticileri tarafından sistemdeki kontrolsüz bağlantı noktalarına erişmek için kullanılan bir ağ tarama aracıdır. Hedeflenen bir sisteme başarılı bir şekilde girmek için tüm bilgisayar korsanlarının yapması gereken, Nmap'ı o sistemde çalıştırmak, güvenlik açıklarını aramak ve onları nasıl kullanabileceğini bulmaktır. Ancak, bilgisayar korsanları gibi bu yazılım platformunu kullanan sadece insanlar değildir. BT güvenlik şirketleri genellikle bir sistemin potansiyel olarak karşılaşabileceği saldırı türlerini çoğaltmanın bir yolu olarak bunu kullanırlar [27].

Nmap, ana bilgisayarlar ve hizmetler için bir ağı kontrol ederek çalışır. Hedef bulunduktan sonra, yazılım platformu daha sonra yanıt veren ana bilgisayarlara ve hizmetlere bilgi gönderir. Nmap, geri gelen yanıtı okur ve yorumlar sonrasında bilgiyi ağ haritası oluşturmak için kullanır. Oluşturulan harita, her bir bağlantı noktasının ne yaptığı ve kimlerin (veya ne kullandığı), ana makinelerine nasıl bağlandıkları, güvenlik duvarı üzerinden ne yaptığı ve yapmadığı ve ortaya çıkan tüm güvenlik sorunlarını listeleme hakkında ayrıntılı bilgiler içerir.

Bununla beraber Nmap, ağın her bölümü ile iletişim kuran karmaşık bir komut dosyası sistemi kullanır. Komut dosyaları, ağ bileşenleri ile normal kullanıcıları arasındaki iletişim araçları olarak işlev görür. Nmap'ın kullandığı komut dosyaları güvenlik açığı algılama, arka kapı algılama, güvenlik açığı kullanımı ve ağ bulma yeteneğine sahiptir. Nmap son derece güçlü bir yazılımdır, ancak onu doğru kullanmak için gereken çok sayıda arka plan bilgisine sahip olma eğilimi vardır.

İnternet güvenlik şirketleri Nmap'ı bir sistemi taramak ve bir bilgisayar korsanının potansiyel olarak faydalanabileceği zayıf yönlerin neler olduğunu anlamak için kullanabilir. Şekil 2.1'de gösterildiği gibi program açık kaynaklı ve ücretsiz olduğundan, ağları açık portlar ve diğer zayıf yönler için taramak amacıyla kullanılan en yaygın araçlardan biridir. Desteklenen platformlar şunları içerir:

- Mac OS X
- Linux, OpenBSD ve Solaris
- Microsoft Windows



Şekil 2.1. Nmap kullanımı

2.2.3. Metasploit

Metasploit, bir açık kaynak saldırı uygulaması şekliyle ilk olarak 2003'de HD Moore tarafından geliştirilen test amaçlı sistemlerde hack için kullanılmıştır. Bu, penetrasyon testi yapan, IDS imzasını geliştiren ve araştırmalardan yararlananlar için faydalı bilgiler sağlar . En son Metasploit 3.0 sürümüyle birlikte, proje tümüyle Ruby programlama tabanlarına taşındı [28]. Bu projenin en ünlü sonuçlarından biri, Ruby'de yazılmış ve kolayca istismarları geliştirmenize, test etmenize ve yürütmenize olanak sağlayan Metasploit Framework'dür. Birkaç tür modül vardır. Modül tipi, modülün amacına ve modülün yaptığı işlem tipine bağlıdır. Metasploit Framework'de bulunan modül tipleri şunlardır:

• Exploit: Bir exploit modülü, bir sistemde veya uygulamada bulunan belirli bir güvenlik açığını hedef alan bir dizi komut yürütür. Sömürü modülü olarak bilinen bu modül, hedef sisteme erişim sağlamak için güvenlik açığından yararlanır. Bu modüller arasında tampon taşması, kod enjeksiyonu ve web uygulaması istismarları bulunur.

• Destek: Bir yardımcı modül olarak tanınır herhangi bir yük taşımaz. Doğrudan sömürü ile ilgili olmayabilir keyfi eylemler gerçekleştirmek için kullanılabilir. Yardımcı modüllere örnek olarak tarayıcılar, seslendiriciler ve hizmet reddi saldırıları dâhildir.

• Sömürü Sonrası(Post-exploit) - Bir sömürü sonrası modülü, daha fazla bilgi toplamanıza veya sömürülen bir hedef sisteme daha fazla erişim sağlamanıza olanak tanır. İşletme sonrasındaki modüllere örnek olarak karma çöplükler, uygulama ve servis numaralandırıcılar dâhildir.

• Yük(Payload): Bir payload, bir istismarın başarılı bir şekilde herhangi bir sistemi tehlikeye atmasından sonra çalışan Shell kodudur. Yük, kabuğa nasıl bağlanmak istediğinizi ve kontrol ettikten sonra hedef sisteme ne yapmak istediğinizi tanımlamanızı sağlar. Bir payload bir Meterpreter veya komut kabuğu açabilir. Meterpreter, ihtiyaç duyduğunuzda dinamik olarak yeni özellikler oluşturmak için DLL dosyaları yazmanıza olanak tanıyan gelişmiş bir taşıma yüküdür.

• NOP üreteci: Bir NOP üreteci, standart IDS ve IPS NOP kızak imzalarını atlamak için kullanabileceğiniz rastgele bir dizi bayt üretir. Tamponları doldurmak için NOP jeneratörleri kullanılır.

Aşağıdaki platformlar tarafından desteklenir:

• Mac OS X

• Linux

• Windows

2.2.4. OpenVAS

OpenVAS (eski klasik “Nessus” olarak da bilinir), her türlü sunucu ve ağ cihazında güvenlik sorunlarını tespit edebilen kapsamlı bir güvenlik açığı değerlendirme sistemidir. İnternet altyapınızı kolayca test etmek için OpenVAS yazılımını deneyebilirsiniz.

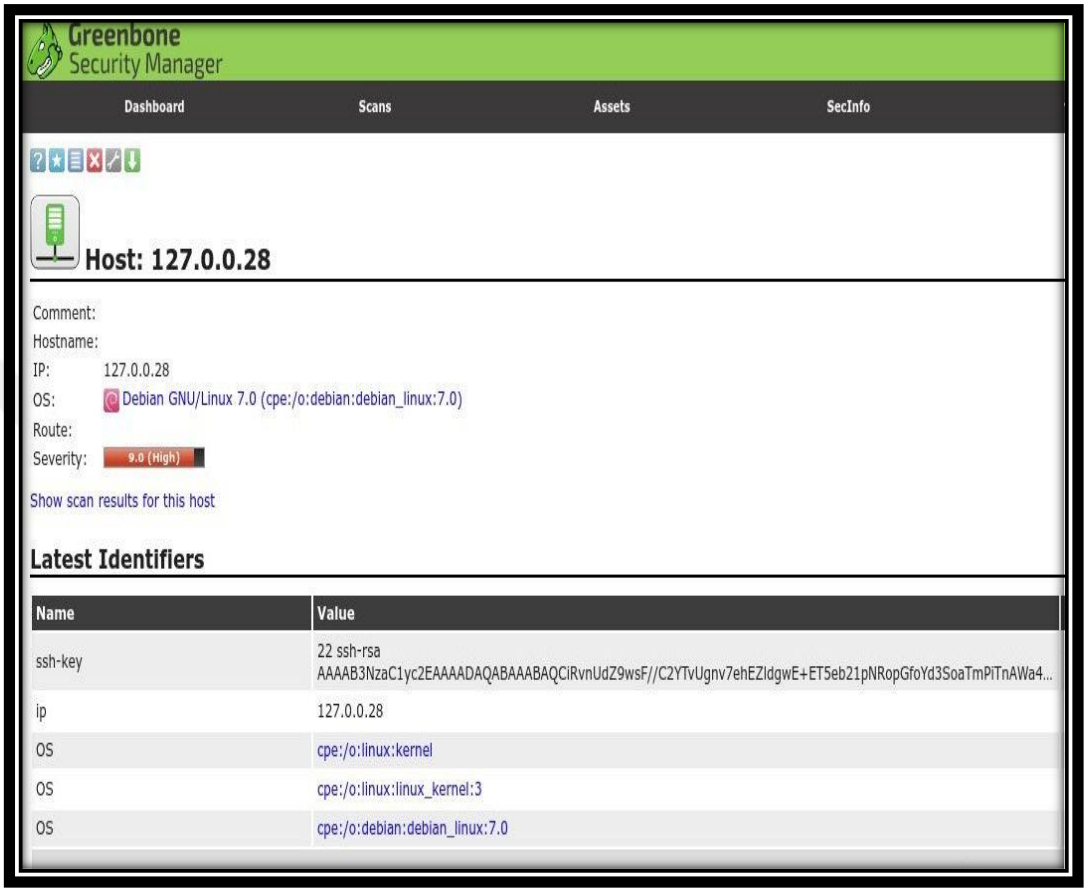
Analiz edilebilmesi için sonuçlar belirtilen e-posta adresine gönderilecektir; sisteminizin dış tehditlerden kaynaklanan risklerini yeniden yönlendirmeye başlamanız açısından kolaylık sağlanır [28].

Ana Özellikler:

Bu tarama türünü kullanmanın birincil nedeni, bir IP adresinin kapsamlı güvenlik testlerini gerçekleştirmektir. Açık hizmetleri bulmak için başlangıçta bir IP adresinin port taraması yapacaktır. Dinleme hizmetleri keşfedildikten sonra, büyük bir veritabanı (53000'den fazla NVT kontrolü) kullanarak bilinen güvenlik açıkları ve yanlış yapılandırma açısından test edilirler. Sonuçlar daha sonra her bir güvenlik açığı ve keşfedilen önemli konularla ilgili ayrıntılı bilgiler içeren bir raporla derlenir.

Testlerin sonuçlarını aldığınızda, her bulgunun alaka düzeyini ve muhtemelen yanlış pozitifleri kontrol etmeniz gerekir. Sisteminizin risk altında olmadığından emin olmak için onaylanmış tüm güvenlik açıklarına yeniden aracılık edilmelidir. Harici olarak barındırılan sunuculardan gerçekleştirilen güvenlik açığı taramaları size saldırganla aynı bakış açısı sunar. Bunun, dışa dönük hizmetlere neyin maruz kaldığını tam olarak anlama avantajı vardır. Web tabanlı arayüzü herhangi bir işletim sisteminden çalıştırılmasına izin verirken, bir CLI de mevcuttur ve

Linux, Unix ve Windows işletim sistemleri için iyi çalışır. Ücretsiz sürümü OpenVAS web sitesinden indirebilirsiniz, ancak Greenbone Security (ana şirket) web sitesinden de temin edilebilen bir ticari işletme lisansı vardır. Şekil 2.2'de gösterildiği gibi bir arayüzü vardır.



Şekil 2.2. OpenVAS kullanımı

2.2.5. Wireshark

Wireshark, kullanıcıların bir bilgisayar ağındaki veri trafiğine etkileşimli olarak göz atmalarını sağlayan ücretsiz ve açık kaynaklı bir ağ protokolü analizörüdür. Geliştirme projesi Ethereal adı altında başlatıldı, ancak 2006'da Wireshark olarak yeniden adlandırıldı.

Dünyanın her yerinden birçok ağ geliştiricisi bu projeye ağ analizi, sorun giderme, yazılım geliştirme ve iletişim protokolleriyle katkıda bulundu. Wireshark birçok eğitim kurumunda ve diğer sanayi sektörlerinde kullanılmaktadır [29].

Ağı tararken sonuçları herhangi bir insan tarafından okunabilirliği yüksek bir biçimde yakalayabilir ve okuyabilirsiniz; bu da olası sorunları (düşük gecikme süresi gibi), tehditleri ve açıkları tespit etmeyi kolaylaştırır.

Ana Özellikler:

• Veriler, ağ bağlantısı üzerinden kablo üzerinden veya zaten veri paketlerini yakalamış veri dosyalarından analiz edilir.

• Çok çeşitli ağlar için (Ethernet, IEEE 802.11, noktadan noktaya Protokol (PPP) ve geri dönüşüm dahil) canlı veri okuma ve analizini destekler.

• GUI veya diğer sürümlerin yardımıyla, kullanıcılar yakalanan veri ağlarına göz atabilirler.

• Ekran filtreleri, veri ekranını filtrelemek ve düzenlemek için kullanılır.

• Yeni protokoller eklentiler oluşturularak incelenebilir.

• Yakalanan trafik, İnternet üzerinden Ses (VoIP) aramalarını ağ üzerinden de izleyebilir.

• Linux kullanırken, ham USB trafiğini yakalamak da mümkündür.

Wireshark, Unix, Linux ve Microsoft Windows işletim sistemlerinde çalışır ve paket yakalama için GTK + widget araç seti ve pcap'i kullanır. Wireshark ve Tshark gibi diğer terminal tabanlı ücretsiz yazılım sürümleri GNU Genel Kamu Lisansı altında yayınlandı.

Tcpdump ile birçok özelliği paylaşıyor gibi görünse de aradaki fark, grafiksel bir kullanıcı arayüzünü (GUI) desteklemesi ve bilgi filtreleme özelliklerine sahip olmasıdır. Ayrıca, Wireshark, kullanıcının ağ üzerinden geçirilen tüm trafiği görmesine izin verir.

2.2.6. Reaver

Reaver, WPA / WPA2 geçiş anahtarlarını kurtarmak için WPS'ye kaba kuvvet saldırısı gerçekleştirmek için kullanılan açık kaynaklı bir araçtır. Bu araç Google Kodunda barındırılıyor ve geliştirici başka bir platforma geçmediyse yakında kaybolabilir. En son 4 yıl önce güncellendi. Diğer araçlara benzer şekilde, bu araç aynı saldırı yöntemini kullanan listedeki diğer araçlara iyi bir alternatif olabilir [29].

Yakın zamana kadar orijinal Reaver sürümü Google Cloud'da barındırılmaktaydı. 1.6 sürümünün sürümünden sonra, Github'da çatallı bir topluluk baskısı başlatıldı. Reaver kullanmanın eskisi kadar bir popülerliği kalmayacak gibi duruyor diyebiliriz.

2.2.7. AirCrack-ng

Aircrack-ng, WiFi ağ güvenliğini değerlendirmek için eksiksiz bir araç paketidir.

Tüm araçları, yoğun komut dosyası çalıştırmaya izin veren komut satırı ile kullanılır [29].

Öncelikle Linux, aynı zamanda Windows, OS X, FreeBSD, OpenBSD, NetBSD, ayrıca Solaris ve hatta eComStation 2'de çalışır. Sürücüsü ham izleme modunu destekleyen ve 802.11a, 802.11b ve 802.11g trafiğini koklayabilen herhangi bir kablosuz ağ arabirim denetleyicisiyle çalışır. WiFi güvenliğinin farklı alanlarına odaklanmaktadır: Şekil 2.3'de gösterildiği gibi kullanılır.

İzleme (Monitoring): Üçüncü parti araçlarla daha fazla işlenmek üzere paket yakalama ve verilerin metin dosyalarına aktarılması.

Saldırı (Attacking): Paket enjeksiyon yoluyla tekrar saldırıları, kimlik doğrulaması, sahte erişim noktaları ve diğerleri.

Deneme (Testing): WiFi kartlarını ve sürücü yeteneklerini kontrol etme (yakalama ve enjeksiyon).

Kırma (Cracking): WEP ve WPA PSK (WPA 1 ve 2).

Önceden okullarda bulunan güvenlik uzmanları için AirCrack-ng, birkaç daha ilginç özelliğin yanı sıra eski sistemlere göre şık bir terminal tabanlı arayüz içerir.

Ana Özellikler:

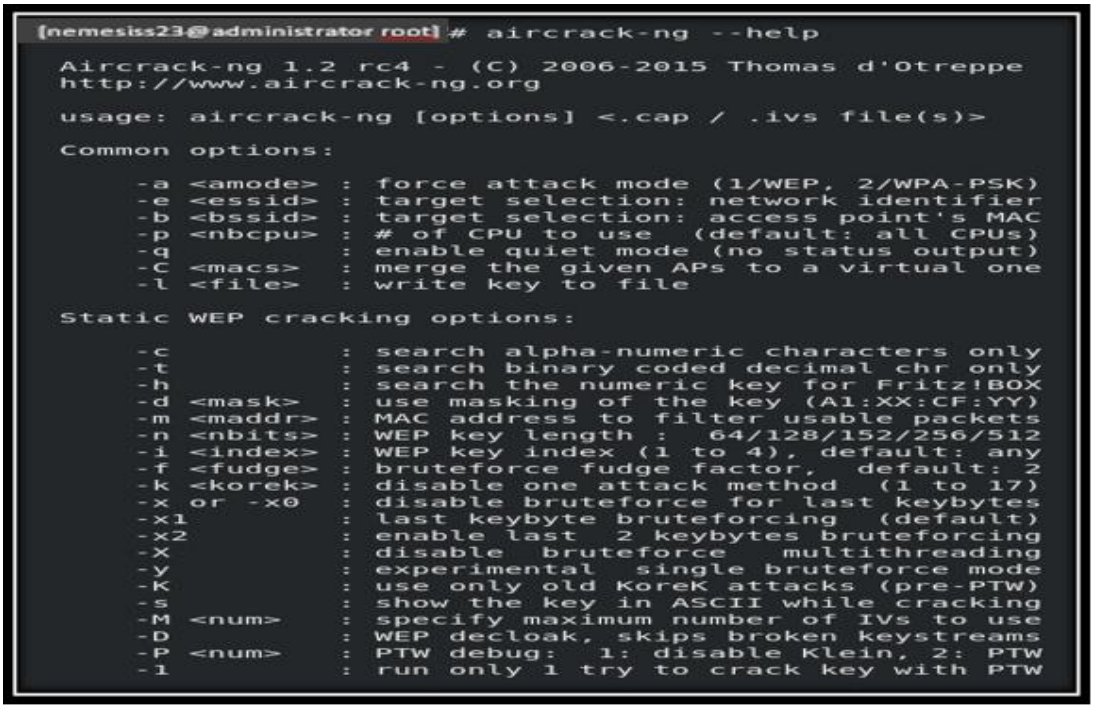
Daha fazla kart / sürücü desteklenir, daha fazla işletim sistemi ve platform desteklenir, yeni WEP saldırısı: PTW, WEP sözlük saldırısı, Fragmentasyon saldırısı, WPA Migration modu, Geliştirilmiş çatlama hızı, birden fazla kartla çekim, optimizasyonlar, diğer iyileştirmeler ve hata düzeltme ve yeni araçlar barındırır.

Ek olarak, aşağıdakiler de dâhil olmak üzere birçok Wifi denetim aracı ile birlikte gelir:

• aireplay-ng: Seçilen bir bilgisayar ya da telefonun ağ ile bağlantısını bir süre kesip otomatik bağlanmasını sağlamak için kullanılır.

• Airmon-ng: Wireless kartının hangi monitörde çalıştığını görmek için kullanılır.

• airodump-ng: Çevredeki ağları listelemek için kullanılır.

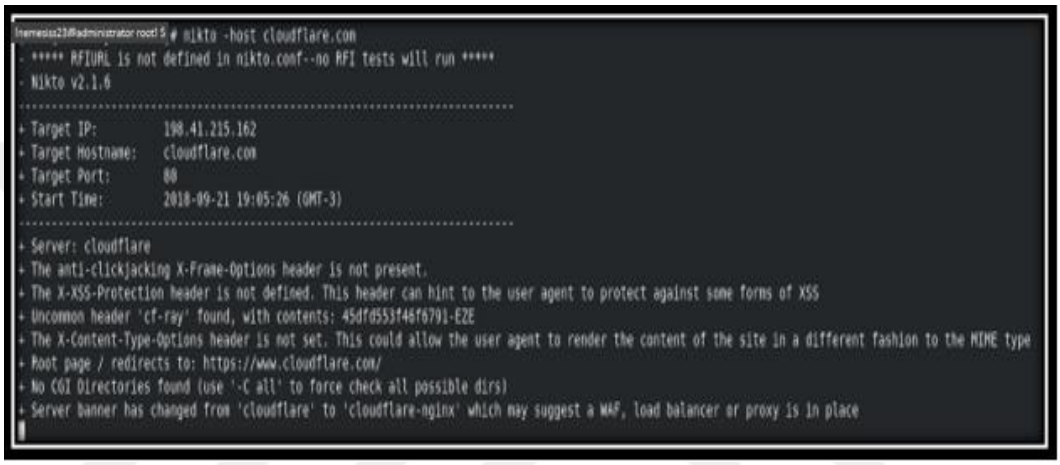


Şekil 2.3. Aircrack-Ng kullanımı

Kablosuz ağ saldırılarında bu komutları kullanabilmek oldukça pratiklik sağlamaktadır.

2.2.8. Nikto

Nikto, 6700'den fazla potansiyel olarak tehlikeli dosya / program dâhil olmak üzere birden fazla öğe için web sunucularına karşı kapsamlı testler yapan, 1250'den fazla sunucunun eski sürümlerini kontrol eden ve 270'den fazla sunucudaki sürüme özgü sorunları içeren bir Açık Kaynak (GPL) web sunucusu tarayıcısıdır. Kullanımı Şekil 2.4'de gösterilmiştir



Şekil 2.4. Nikto kullanımı

Ayrıca, birden fazla dizin dosyasının varlığı, HTTP sunucusu seçenekleri gibi sunucu yapılandırma öğelerini de denetler ve yüklü web sunucularını ve yazılımlarını belirlemeye çalışır. Tarama öğeleri ve eklentileri sık sık güncellenir ve otomatik olarak güncellenebilir. Nikto gizli bir araç olarak tasarlanmamıştır. Bir web sunucusunu mümkün olan en kısa sürede test eder ve günlük dosyalarında veya bir IPS / IDS'de açıktır. Ancak, denemek istemeniz durumunda (ya da IDS sisteminizi test etmeniz) durumunda LibWhisker'in anti-IDS yöntemleri için destek var.

Nikto'nun başlıca özellikleri şunlardır:

• SSL Desteği (OpenSSL ile Unix veya ActiveState Perl / NetSSL ile Windows)

• Tam HTTP proxy desteğine sahip olma

• Eski sunucu bileşenlerini denetleme

• Raporları düz metin, XML, HTML, NBE veya CSV'ye kaydedebilme

• Raporları kolayca özelleştirmek için şablon motoru

• Bir sunucudaki birden çok bağlantı noktasını veya giriş dosyası aracılığıyla birden çok sunucuyu tarayın (nmap çıkışı dâhil)

• LibWhisker'ın IDS kodlama teknikleri

• Komut satırı ile kolayca güncellenebilme
• Yüklü yazılımı başlıklar, favoriler ve dosyalar aracılığıyla tanımlama.
• Basic ve NTLM ile ana bilgisayar doğrulaması
• Apache ve cgiwrap kullanıcı adı sayımı
• Web sunucularındaki içeriklerde "fishing" için mutasyon teknikleri
• Yetkilendirme tahmini, yalnızca kök dizini değil, herhangi bir dizini de işler.
• Görülen “olağandışı” başlıklara ait raporlar
• Etkileşimli durum, duraklatma ve ayrıntı ayarlarında gerçekleşen değişiklikler
• Olumlu testler için tam istek / yanıtı kaydetme özelliği
• Hedef başına maksimum yürütme süresi
• Belirtilen bir zamanda otomatik duraklatma
• Yaygın “park yeri” sitelerini kontrol edebilme
• Metasploit Framework ile entegrasyon halinde çalışma

2.2.9. IronWASP

IronWASP (Web uygulaması Gelişmiş Güvenlik testi Platformu), web uygulaması güvenlik açığı testi için açık kaynaklı bir sistemdir. Kullanıcıların kendi özel güvenlik tarayıcılarını kullanarak oluşturabildikleri ölçüde özelleştirilebilir olacak şekilde tasarlanmıştır. Python / Ruby komut dosyası uzmanlığına sahip ileri düzey bir kullanıcı, platformu tam olarak kullanabilse de, aracın özelliklerinin çoğu mutlaka yeni başlayanlar için kullanılacak kadar basittir [30].

IronWASP ile ilgili en çekici şeylerden biri, ana özelliklerini yönetmek için bir uzman olmanıza gerek olmamasıdır. Hepsi GUI tabanlıdır ve yalnızca birkaç tıklamayla tam taramalar gerçekleştirilebilir. Yani, etik hack araçlarıyla yeni başlıyorsanız, bu başlamak için harika bir yoldur.

Ana özelliklerinden bazıları şunlardır:

• Ücretsiz ve Açık kaynaktır.
• GUI tabanlı ve kullanımı çok kolay, güvenlik uzmanlığı gerektirmez.
• Güçlü ve etkili tarama motoru vardır.
• Giriş sırasını kaydetmeyi destekler.
• Hem HTML hem de RTF formatlarında raporlama özelliğ vardır.
• 25 farklı türden tanınmış web güvenlik açığı olup olmadığını kontrol eder.
• Yanlış Pozitif Tespit Desteği vardır.
• Yanlış Negatif Tespiti Desteği
• Python ve Ruby'yi destekleyen endüstri lideri yerleşik komut dosyası altyapısı
• Python, Ruby, C # veya VB.NET'teki eklentiler veya modüller aracılığıyla genişletilebilir.

2.2.10. SQLNinja

Sqlninja, arka uç olarak Microsoft SQL Server kullanan bir web uygulamasında SQL Injection güvenlik açıklarından yararlanmayı hedefleyen bir araçtır. Asıl amacı, savunmasız DB sunucusuna çok düşmanca bir ortamda bile uzaktan erişim sağlamaktır. Bir SQL Injection güvenlik açığı keşfedildiğinde DB Sunucusunu devralma sürecine yardımcı olmak ve otomatikleştirmek için penetrasyon test cihazları tarafından kullanılmalıdır [30].

SQLninja ana özellikleri aşağıdaki gibidir.

• Uzak SQL Server'ın parmak izi (sürüm, sorguları yapan kullanıcı, kullanıcı ayrıcalıkları, xp_cmdshell kullanılabilirliği, DB kimlik doğrulama modu)

• Zamana dayalı veya bir DNS tüneli ile veri çekebilme

• Metasploit3 ile entegrasyon, uzak DB sunucusuna VNC sunucu enjeksiyonu yoluyla grafiksel bir erişim elde etmek veya sadece Meterpreter yüklemek için kullanılabilme

• Yalnızca normal HTTP isteklerini kullanarak çalıştırılabilir dosyaların yüklenmesi (FTP / TFTP gerekmez) vbscript veya debug.exe aracılığıyla

• Hem TCP hem de UDP ile doğrudan ve ters bağlama

• DNS tüneli sözde shell, doğrudan / ters shell için TCP / UDP bağlantı noktası bulunmadığında, ancak DB sunucusu harici ana bilgisayar adlarını çözebilme

• ICMP tüneli shell, doğrudan / ters shell için TCP / UDP bağlantı noktası bulunmadığında ancak DB kutunuzu silebilme

• Orijinali kaldırılmışsa, özel bir xp_cmdshell oluşturulması

• Hedef ağın güvenlik duvarı tarafından izin verilen bir bağlantı noktası bulmak ve onu ters bir shell için kullanmak amacıyla hedef SQL Server'dan saldıran makineye TCP / UDP portscan uygulayabilme

• Birkaç IDS / IPS / WAF'ı karıştırmak için kaçınma tekniklerine sahiptir.

• Churrasco.exe ile entegrasyon, w2k3'teki SYSTEM ayrıcalıklarını token kaçırma yoluyla artırmak.

• Sqlservr.exe'nin SYSTEM'e ayrıcalıklarını artırmak için CVE-2010-0232 desteği

2.2.11. SQLMap

SQLmap, penetrasyon test cihazlarının SQL enjeksiyon hatalarını otomatik olarak algılayıp kullanmalarına yardımcı olan popüler bir açık kaynak kodlu araçtır. Bu Python tabanlı araç, test edenlerin veritabanı sunucularını ele geçirmelerine yardımcı olur [30].

Güçlü bir algılama motoruna ve en iyi penetrasyon testine olanak veren özelliklere sahiptir. İşletim sistemlerinde veritabanı parmak izi, veri alma ve komut çalıştırma gibi birçok anahtar talimatlar bulabilirsiniz.

SQLmap aşağıdakiler için kullanılabilir:

• SQL enjeksiyon güvenlik açığı karşı web uygulamaları tarayabilme

• SQL enjeksiyon güvenlik açığından yararlanın

• Veritabanlarını ve veritabanı kullanıcısını ayrıntılı bir şekilde çıkarma

• Dış müdahale komut dosyalarını kullanarak Web Uygulaması Güvenlik Duvarı'nı (WAF) atlatma

• Temel işletim sistemine sahip olma

Ana Özellikler:

• MySQL, Oracle, PostgreSQL, Microsoft Access, Microsoft SQL Server, IBM DB2, SQLite, Firebird, Sybase ve SAP MaxDB DBMS'leri desteklemek.

• Altı SQL enjeksiyon prosedürünü tam olarak destekler: boolean tabanlı kör, hata bazlı, UNION sorgusu, zamana dayalı kör, yığılmış sorgular ve bant dışı sorgulama yeteneği.

• Sözlük tabanlı bir saldırı kullanarak şifre kırma biçimlerinin kırılmasını destekler

• Kullanıcıların, parola özetlerinin, ayrıcalıkların, rollerin, veritabanlarının, tabloların ve sütunların numaralandırılmasını sağlamak.

• Çoklu veritabanı sunucusu desteği: Oracle, PostgreSQL, MySQL ve MSSQL, MS Access, DB2 veya Informix.

• Otomatik kod enjeksiyon yeteneklerine sahip olma

• Karma şifre tanıma

• Sözlüğe dayalı şifre kırma

• Kullanıcı ayrıcalıklarını ve veritabanlarını görüntüleme

• Veritabanı kullanıcı ayrıcalığı yükseltmesi

• Döküm tablosu bilgisi çekme

• Uzak SQL SELECTS'i yürütme

2.2.12. Wapiti

Wapiti, web uygulamalarını veri tabanı enjeksiyonları, dosya açıklamaları, siteler arası komut dosyası çalıştırma, komut yürütme saldırıları, XXE enjeksiyonu ve CRLF enjeksiyonunu içeren birden fazla güvenlik açığı için tarayan açık kaynaklı bir araçtır. Veri tabanı enjeksiyonu, SQL, XPath, PHP, ASP ve JSP enjeksiyonlarını içerir. Komut yürütme saldırıları eval (), system () ve passtru () açıklarını içerir.

Yukarıda belirtilen güvenlik açıklarını tespit etmenin yanı sıra, Wapiti ayrıca sunucularda potansiyel olarak tehlikeli dosyalar bulma, .httaccess dosyalarında güvenlik ihlallerine neden olabilecek yapılandırma hatalarını bulma ve sunucudaki uygulamaların yedek kopyalarını bulma gibi bazı sızma testi görevlerini de yerine getirir. Bir saldırgan bu dosyaları ele geçirmeyi başarırsa söz konusu web uygulamalarının güvenliğini tehlikeye atar. Toplanan sonuçlar otomatik olarak bir

html dosyasına kaydedilir. Desteklenen diğer dosya formatları arasında .XML, .JSON ve .TXT bulunur [30].

Wapiti, taradığı her bir güvenlik açığı için özel modüllere sahiptir. Hedefi herhangi bir güvenlik açığı için taramadan önce, Wapiti, hedef web uygulamasıyla ilişkili bağlantıları numaralandırır. Bağlantılar bir kez numaralandırıldığında, Wapiti hedef web uygulamasının savunmasız olup olmadığını test etmek için her modülü birer birer çalıştırır. Güvenlik açıkları ile ilgili ayrıntılar, tarama sonunda oluşturulan bir dosyada depolanır. Wapiti, işlemin ortasında iptal edilirse bile tarama işlemine devam etme yeteneğine sahiptir:

Aracın diğer özellikleri arasında, tarama işleminden istenmeyen URL'leri hariç tutma, JavaScript ve Shockwave Flash dosyalarından URL'leri çıkarma ve kimlik doğrulama desteği gibi HTTP ve HTTPS proxy desteği bulunur.

2.2.13. Maltego

Maltego, link analizi için yönlendirilmiş grafikler sunan etkileşimli bir veri madenciliği aracıdır. Bu araç, çevrimiçi araştırmalarda, İnternette bulunan çeşitli kaynaklardan gelen bilgi parçaları arasındaki ilişkileri bulmak için kullanılır. Daha sonra farklı veri kaynaklarını sorgulama sürecini otomatikleştirmek için dönüşümler fikrini kullanır. Bu bilgi daha sonra bağlantı analizi yapmak için uygun olan bir düğüm bazlı grafikte görüntülenir.

Bu durumda, insanlar, isimler, telefon numaraları, e-posta adresleri, şirketler, organizasyonlar ve sosyal ağ profilleri arasındaki ilişkileri ilişkilendirmek ve belirlemek için kullanılabilir. Whois verileri, DNS kayıtları, sosyal ağlar, arama motorları, coğrafi konum servisleri ve çevrimiçi API servisleri gibi çevrimiçi kaynakların yanı sıra, aşağıdakileri içeren internet tabanlı altyapılar arasındaki ilişkiyi araştırmak için de kullanılabilir:

Maltego, bütün bilgilerin güzel bir şekilde anlaşılması kolay bir varlık-ilişki modelini temsil eder. Çeşitli dönüşümleri ve makineleri kullanarak veri çıkarmanın yanı sıra, diğer kaynaklardan bulunan verileri de alabilir ve daha büyük bir resim oluşturmak için grafiğe eklenebilir. Bu şekilde kullanıcılara kullanışlı bir ara yüz sunar.

Bu uygulama Windows, Linux ve Mac OS için mevcuttur ve yazılım gereksinimi için sadece Java 1.8 veya daha üstü yüklü olmalıdır [31].

2.2.14. Canvas

Canvas, uzak ağları test etmek için 800'den fazla sömürü aracı sunar. Canvas sağladığı seçeneklerle Metasploite alternatif olarak kullanılmaktadır. Bu aracın temel özellikleri aşağıdaki gibidir [32].

• Uzaktan ağ kullanımı

• Farklı tür sistemleri hedefleme yeteneği

• Seçilen coğrafi bölgeleri hedefleme

• Uzak sistemlerin ekran görüntülerini alma

• Şifrelerin indirilebilmesi

• Sistemin içindeki dosyaları değiştirme

• Yönetici erişimi kazanmak için ayrıcalıkları artırma

Bu araç aynı zamanda platformunu yeni istismarlar yazmak için kullanmanıza veya ünlü shellcode jeneratörünü kullanmanıza izin verir. Ayrıca, özellikle orta ve büyük ağlar üzerinden port taraması ve ana bilgisayar keşfi için yararlı olan scanrand adlı tool ile nmap'a bir alternatif de ekler.

Desteklenen platformlar şunları içerir:

• Linux

• MacOSX (PyGTK gerektirir)

• Windows (Python ve PyGTK gerektirir.)

2.2.15. Ettercap

Ettercap, canlı bağlantıların koklanmasını, anında içerik filtrelemeyi ve diğer ilginç püf noktaları içerir. Birçok protokolün aktif ve pasif diseksiyonunu destekler ve ağ ve host analizi için birçok özellik içerir.

Diğer yetenekler arasında ağ ve ana bilgisayar analizi (işletim sistemi parmak izi gibi) ve yerleşik bağlantılar üzerinden ağ manipülasyonu bulunur. Bu durum, bu aracı ortadaki adam saldırılarını test etmek için harika bir örnek kılar [33].

Ettercap, birçok protokolün (şifreli olanlar dahil) aktif ve pasif diseksiyonunu destekler ve ağ ve ana bilgisayar analizi için birçok özellik sunar. Ettercap dört çalışma modu sunar:

IP tabanlı: paketler IP kaynağına ve hedefine göre filtrelenir.

MAC tabanlı: paketler bir ağ geçidi üzerinden bağlantıları koklamak için yararlı olan MAC adresine göre filtrelenir.

ARP tabanlı: iki ana bilgisayar (tam çift yönlü) arasında geçişli bir LAN'da koklamak için ARP zehirlenmesi kullanır.

PublicARP tabanlı: kurban bir konaktan diğer tüm konakçılara (yarı çift yönlü) geçişli bir LAN'da koklamak için ARP zehirlenmesi kullanır.

Ana Özellikler:

Yerleşik bir bağlantıya karakter enjeksiyonu: canlı bağlantıyı koruyarak karakterler bir sunucuya (emülasyon komutları) veya bir müşteriye (cevaplar taklit edilebilir) enjekte edilebilir.

SSH1 desteği: Bir kullanıcı adının ve şifrenin koklanmasının yanı sıra bir SSH1 bağlantısındaki verileri tam çift yönlü koklama yeteneğine sahip ilk yazılımdır

HTTPS desteği: bağlantı güvenli bir proxy aracılığıyla yapılsa bile HTTP SSL güvenli verilerinin aranması.

Bir GRE tüneli üzerinden uzak trafik: uzak bir Cisco yönlendiricisinden bir GRE tüneli üzerinden uzak trafiğin koklanması ve buna “ortadaki adam saldırısı” yapılması.

Eklenti desteği: Ettercap API'sini kullanarak özel eklentiler oluşturmak.

Parola toplayıcılığı: SSH1, ICQ, SMB, BGP, SOCKS 5, IMAP 4, VNC, MySQL, HTTP, NNTP, X11, Napster, IRC, RIP, LDAP, NFS, TELNET, FTP, POP, IMAP, rlogin, SNMP, MSN, YMSG vb.

Paket filtreleme/bırakma: TCP veya UDP yükünde belirli bir dize (veya onaltılık sıra) arayan ve bunu özel bir dize/dizi ile değiştiren veya tüm paketi bırakan bir filtre ayarlamak.

OS parmak izi: Kurban ev sahibinin işletim sistemini ve ağ adaptörünü belirler.

Bir bağlantıyı kesme: bağlantı listesinden tercih edilen bağlantıların öldürülmesi.

LAN'ın pasif taranması: LAN üzerindeki ana bilgisayarlar, açık portları, mevcut servislerin sürüm numaraları, ana bilgisayarın türü (ağ geçidi, yönlendirici veya basit PC) ve atlama sayısında tahmini mesafeler hakkında bilgi alınması, DNS isteklerinin kaçırılması, LAN üzerindeki diğer zehirleyicileri aktif veya pasif olarak bulma yeteneğine de sahip olmak.

3. KÖTÜCÜL YAZILIMLARI SINIFLANDIRMADA KULLANILAN MAKİNE ÖĞRENMESİ VE DERİN ÖĞRENME YÖNTEMLERİ

Bu bölümde, literatür taraması sonuçlarında elde edilen kötücül yazılımları sınıflandırmada kullanılan makine öğrenmesi ve derin öğrenme yöntemleri ele alınmıştır. Öncelikle geçmişten günümüze kadar gelişim gösteren bu yöntemler araştırıldı. Bu bölüm, pratik uygulamayı anlamak için gerekli olan makine öğrenme yöntemleri üzerine teorik bir arka plan sunar. İlk önce, makine öğrenmesi ve derin öğrenme alanına genel bir bakış ve ardından bu çalışmalarla ilgili yöntemlerin açıklaması tartışılmaktadır. Bu yöntemler arasında Naive Bayes, K-En Yakın Komşular, Karar Ağaçları, Rastgele Ormanlar, Destek Vektör Makineleri, LBP, HOG, MPL, GRU, LSTM, AlexNet, GoogLeNet, ResNet-18, ResNet-50, ResNet-101, VGG-16, VGG-19, SqueezeNet, DenseNet-201 sayılabilir.

3.1. Naive Bayes

Naive Bayes sınıflandırıcıları Bayes Teoremine dayanan bir sınıflandırma algoritmaları topluluğudur. Tek bir algoritma değil, hepsinin ortak bir prensibi paylaştığı bir algoritma ailesidir, yani sınıflandırılan her bir özellik çifti birbirinden bağımsızdır. Bu yöntemi tanımlamak için, bir veri kümesini düşünelim.

Golf oynamak için hava koşullarını açıklayan kurgusal bir veri seti düşünün. Hava koşulları göz önüne alındığında, her bir D golf oynamak için koşulları uygun (“Evet”) veya uygun değil (“Hayır”) olarak sınıflandırır [34].

Veri setimizin tablo halinde gösterimi Tablo 3.1’de belirtilmiştir.

Tablo 3.1. Tenis oynayabilme durumunu gösteren veri seti [34]

Gün	Görünüm	Sıcaklık	Nem	Rüzgâr	Tenis Oynama Durumu
D1	Güneşli	Sıcak	Yüksek	Zayıf	Yok hayır
D2	Güneşli	Sıcak	Yüksek	Kuvvetli	Yok hayır
D3	Bulutlu	Sıcak	Yüksek	Zayıf	Evet
D4	Yağmur	Hafif	Yüksek	Zayıf	Evet
D5	Yağmur	Güzel	Normal	Zayıf	Evet
D6	Yağmur	Güzel	Normal	Kuvvetli	Yok hayır

Tablo 3.1. (Devamı)

Gün	Görünüm	Sıcaklık	Nem	Rüzgâr	Tenis Oynama Durumu
D7	Bulutlu	Güzel	Normal	Kuvvetli	Evet
D8	Güneşli	Hafif	Yüksek	Zayıf	Yok hayır
D9	Güneşli	Güzel	Normal	Zayıf	Evet
D10	Yağmur	Hafif	Normal	Zayıf	Evet
D11	Güneşli	Hafif	Normal	Kuvvetli	Evet
D12	Bulutlu	Hafif	Yüksek	Kuvvetli	Evet
D13	Bulutlu	Sıcak	Normal	Zayıf	Evet
D14	Yağmur	Hafif	Yüksek	Kuvvetli	Yok hayır

Veri kümesi, özellik matrisi ve yanıt vektörü olmak üzere iki bölüme ayrılmıştır.

- Özellik matrisi, her vektörün bağımlı özelliklerin değerinden oluştuğu veri kümesinin tüm vektörlerini (satırlarını) içerir. Yukarıdaki veri setinde, özellikler 'Görünüm', 'Sıcaklık', 'Nem' ve 'Rüzgâr' şeklindedir.
- Tepki vektörü, özellik matrisinin her satırı için sınıf değişkeninin değerini (öngörü veya çıktı) içerir. Yukarıdaki veri setinde, sınıf değişkeninin adı 'Tenis Oynama Durumu' şeklindedir.

Temel Naive Bayes varsayımı, her bir özelliğin şunları yaptığıdır:

- Bağımsızlık
- Eşitlik

Ve sonuca katkı olarak bunlar gösterilir.

Veri setiyle ilgili olarak, bu kavram şöyle anlaşılabilir:

Hiçbir özellik çiftinin bağımlı olmadığını varsaymaktayız. Örneğin, "Sıcak" olan sıcaklığın nemle hiçbir ilgisi yoktur veya "Yağmurlu" olmanın rüzgârlar üzerinde bir etkisi yoktur. Dolayısıyla, özelliklerin bağımsız olduğu varsayılmaktadır.

İkincisi, her özelliğe aynı ağırlık (veya önem) verilmiştir. Örneğin, yalnızca sıcaklığı ve nemi yani yalnızca bilineni bilmek sonucun isabet değerini tahmin etmemize yardımcı olmaz. Özelliklerin hiçbiri önemsiz değildir ve sonuca eşit katkıda bulundukları varsayılmaktadır.

Naive Bayes tarafından yapılan varsayımlar, gerçek dünyadaki durumlarda genellikle doğru değildir. Aslında, bağımsızlık varsayımı hiçbir zaman doğru değildir, ancak pratikte sıklıkla işe yarar.

Şimdi, Naive Bayes formülüne geçmeden önce Bayes teoremini bilmek önemlidir. Bayes Teoremi, daha önce gerçekleşmiş olan başka bir olayın olasılığı göz önüne alındığında meydana gelen olay olasılığını bulur. Bayes teoremi matematiksel olarak Denklem 3.1'de gösterildiği gibi ifade edilmektedir:

$$P(A\backslash B) = \frac{P(B\backslash A)P(A)}{P(B)} \tag{3.1}$$

A ve B olayları ve P (B) nerededir?

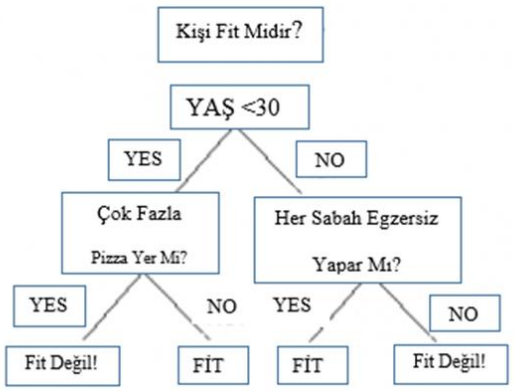
• Temel olarak, B olayı doğruysa, A olayı olasılığını bulmaya çalışmaktayız. B olayı da kanıt olarak adlandırılır.

• P (A), A'nın önceliğidir (önceki olasılık, yani delil görülmeden olayın olasılığı). Kanıt, bilinmeyen bir örneğin nitelik değeridir. (burada, B olayıdır).

• P (A | B), B'nin posteriori olasılığıdır, yani delili gördükten sonraki olay olasılığıdır.

3.2. Karar Ağaçlar

Karar Ağaçları, verilerin belirli parametrelere göre sürekli olarak bölündüğü denetimli bir makine öğrenim yöntemidir. Ağaç iki varlık (karar düğümleri ve yaprakları) ile açıklanabilir. Yapraklar, kararlar veya nihai sonuçlardır. Karar düğümleri verinin bölündüğü yerdir. Bu sistem Şekil 3.1'de şematize edilmiştir.



Şekil 3.1. Karar ağaçları örneği

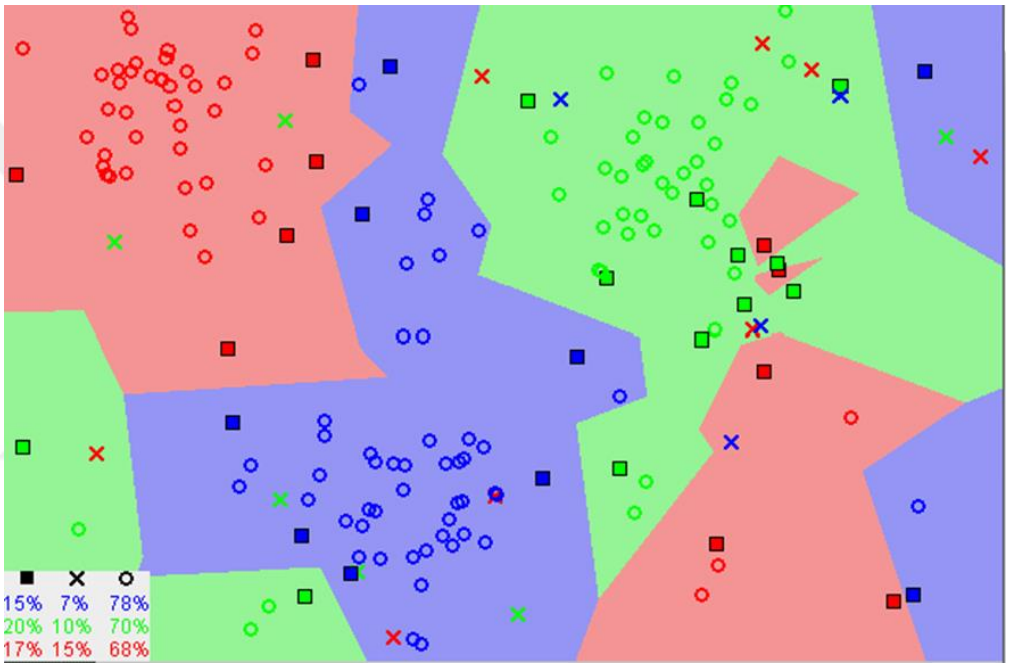
Yukarıdaki ikili ağaç, karar ağaçlarının örneklerini göstermek için kullanılabilir. Bir kişinin yaşı, alışkanlıkları, fiziksel egzersiz vb. Bilgi sağlamak için uygun yöntemi tahmin etmek istediğinizi varsayalım. Buradaki karar düğümleri 'Yaş nedir?', 'Egzersiz yapıyor mu?' çok pizza yer mi? Ve "uygun" veya "uygun olmayan" gibi bir sonuç olan yapraklar. Bu durumda bu ikili bir sınıflandırma problemiydi. (evet/hayır tipi problem) [35].

Yukarıda gördüğümüz, sonucun 'FİT' veya 'Fit Değil' gibi bir değişken olduğu bir sınıflandırma ağacı örneğidir. Burada karar değişkeni kategoriktir.

3.3. K-En yakın Komşular

KNN algoritması mesafe odaklı çalışır, benzer nesnelerin Şekil 3.2'deki gibi yakınlarda bulunduğunu varsayar. Başka bir deyişle, "benzer nesneler birbirine yakındır veya uzaktır." görüşünü esas alır [36].

"Tencere yuvarlanmış kapağını bulmuş."



Şekil 3.2. Benzer Veri Noktalarının Birbirlerine Olan Yakınlık Durumunun Gösterilmesi

Yukarıdaki resimde, çoğu zaman benzer veri noktalarının birbirine yakın olduğuna dikkat etmelisiniz. KNN algoritması, bu varsayıma, algoritmanın faydalı olması için yeterince doğru olduğuna dayanmaktadır. KNN, çocukluğumuzda öğrenmiş olabileceğimiz bazı matematiğe benzerlik fikrini (bazen mesafe, yakınsama veya yakınlık olarak adlandırılır) yakalar. Örneğin bir grafikteki noktalar arasındaki mesafeyi hesaplar.

Bir grafikteki noktalar arasındaki mesafeyi nasıl hesapladığımızı anlamak, devam etmeden önce gereklidir. Bu hesaplamanın nasıl yapıldığını bilmiyorsanız veya bir tazelemeye ihtiyacınız varsa, "2 nokta arasındaki mesafeyi bulma yöntemlerini" öğrenmek gereklidir.

Mesafeyi hesaplamanın başka yolları da vardır ve çözdüğümüz soruna bağlı olarak bir yol tercih edilebilir. Bununla birlikte, en yakın düz çizgi mesafesi (Öklid mesafesi olarak da bilinir) popüler ve tanıdık bir seçimdir.

KNN algoritmasının adımları aşağıdaki gibi verilmiştir.

Adım 1: Verileri yükleyin.

Adım 2: K'yı seçtiğiniz komşu sayısına sıfırlayın.

Adım 3: Verilerdeki her örnek için:

Veriden sorgu örneği ile mevcut örnek arasındaki mesafeyi hesaplayın.

Düzenli bir koleksiyona örneğin mesafesini ve örneğini ekleyin.

Adım 4: Sıralı uzaklık ve indeks koleksiyonunu mesafelere göre en küçüğünden en büyüğüne (artan sırayla) sıralayın.

Adım 5: Sıralanan koleksiyondan ilk K girişini seçin.

Adım 6: Seçilen K girişlerinin etiketlerini alın.

Adım 7: Regresyon varsa, K etiketlerinin ortalamasını döndürün.

Adım 8: Sınıflandırma ise, K etiketlerinin modunu döndürün.

K için doğru değeri seçme

Verilerinize uygun K'yi seçmek için, KNN algoritmasını K'nın farklı değerleriyle birkaç kez çalıştırırız ve algoritmanın sahip olduğu veriler verildiğinde doğru tahminler yapabilme yeteneğini koruyarak karşılaştığımız hata sayısını azaltan K'yi seçeriz.

Akılda tutulması gereken bazı şeyler:

• K değerini 1'e düşürdükçe tahminlerimiz daha az kararlı hale geldi. Sadece bir dakika düşünün, K = 1 imgesi ve birkaç kırmızı ve bir yeşil ile çevrilmiş bir sorgulama noktamız var (yukarıdaki renkli alanın sol üst köşesini düşünüyorum), ancak yeşil en yakın komşudur. Makul bir şekilde, sorgu noktasının büyük olasılıkla kırmızı olduğunu düşünürüz, ancak K = 1 olduğundan, KNN yanlış bir şekilde sorgu noktasının yeşil olduğunu tahmin eder.

• Tersine, K'nin değerini artırdıkça, tahminlerimiz çoğunluk oylaması/ortalamaları nedeniyle daha istikrarlı hale gelir ve bu nedenle daha kesin tahminler yapma (belirli bir noktaya kadar) olasılığı daha yüksektir. Sonunda, artan sayıda hataya tanıklık etmeye başladık. İşte bu noktada K'nin değerini çok fazla ittiğimizi biliyoruz.

• Etiketler arasında çoğunlukla oy kullandığımız durumlarda (örneğin bir sınıflandırma probleminde modu seçme), genellikle K karakterini kullanmak için K tuhaf bir sayı yaparız.

KNN Sınıflandırmasının Avantajları: Algoritması basit ve uygulaması kolaydır. Model oluşturmaya, birkaç parametre ayarlamanıza veya ek varsayımlarda bulunmanıza gerek yoktur. Algoritma çok yönlüdür. Sınıflandırma, regresyon ve arama için kullanılabilir (bir sonraki bölümde göreceğimiz gibi).

KNN Sınıflandırmasının Dezavantajları: Algoritma, örnek sayısı ve / veya öngörücü / bağımsız değişken sayısı arttıkça önemli ölçüde yavaşlar.

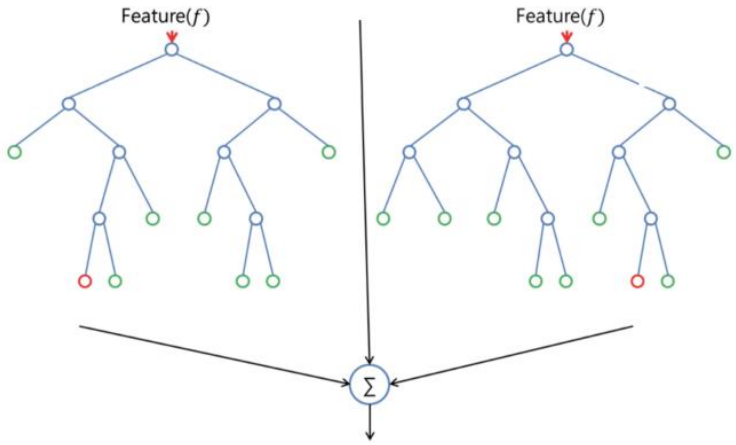
3.3.1. KNN'nin Uygulaması

KNN'nin veri hacmi arttıkça önemli ölçüde yavaşlamasının ana dezavantajı, tahminlerin hızla yapılması gereken ortamlarda pratik olmayan bir seçim olmasını sağlar. Ayrıca, daha doğru sınıflandırma ve regresyon sonuçları üretebilecek daha hızlı algoritmalar vardır. Ancak, tahminlerde bulunmak için kullandığınız verileri hızlı bir şekilde ele almak için yeterli bilgi işlem kaynağınız olması koşuluyla, KNN, benzer nesneleri tanımlamaya dayanan çözümleri olan problemlerin çözümünde faydalı olabilir. Buna bir örnek, KNN araştırmasının bir uygulaması olan, tavsiye veren sistemlerde KNN algoritmasını kullanmaktır.

3.4. Rastgele Orman Sınıflandırıcısı

Rastgele Orman, kontrollü bir öğrenme algoritmasıdır. Adından da görebileceğiniz gibi, bir orman yaratır ve bir şekilde rastgelelik sağlar. Yarattığı "orman" esas olarak "torbalama" yöntemi ile eğitilmiş bir dizi karar ağacıdır(DT). Torbalamanın genel fikri, öğrenme modellerinin kombinasyonunun toplam puanı arttırmasıdır [37].

Kısacası, rastgele bir orman birçok karar ağacı oluşturur ve daha doğru ve istikrarlı bir tahmin elde etmek için bunları birleştirir. Rasgele ormanların en büyük avantajı, mevcut makine öğrenme sistemlerinin çoğunluğu olan hem sınıflandırma hem de regresyon problemleri için kullanılabilmeleridir. Rasgele sınıflandırma, sınıflandırmada listelenecektir. Çünkü sınıflandırma bazen makine öğrenmesinin bir parçası olarak kabul edilir. Şekil 3.3'de rastgele bir ormanın iki ağaçla nasıl görüneceğini görebilirsiniz:



Şekil 3.3. Rastgele ormanlar sınıflandırmasının kullanımı

Rastgele orman, karar ağacı veya torba sınıflandırıcısıyla neredeyse aynı süper parametrelere sahiptir. Neyse ki, karar ağaçlarını torba sınıflandırıcıları ile birleştirmek zorunda değilsiniz ve "rastgele orman" sınıfını kolayca kullanabilirsiniz. Daha önce belirtildiği gibi, rastgele ormanlarda regresyon görevleri gerçekleştirmek için rastgele bir orman regresörü de kullanabilirsiniz. Ağaçlar büyürken, "rastgele ormanlar" modele ekstra bir rastgelelik katar. Düğümleri bölerken en önemli özellikleri aramak yerine, rastgele bir özellik alt kümesinde en iyi özellikleri bulmak daha iyidir. Bu, çeşitli sonuçlar üretir ve genellikle daha iyi bir model üretir.

Bu nedenle, rastgele ormanlarda, yalnızca rastgele özellik alt kümeleri dikkate almak için kullanılır ve algoritma düğümleri ayırmak için kullanılır. Ayrıca, en uygun eşiği (normal bir karar ağacı gibi) bulmak yerine, her özellik için rastgele bir eşik kullanarak ağacın rasgeleliğini artırabilirsiniz. Rasgele orman algoritması için önemli parametreler aşağıdaki gibidir. Rastgele ormandaki hiper parametreler, modelin tahmin gücünü artırmak veya modeli daha hızlı hale getirmek için kullanılır. Burada, rasgele orman fonksiyonunda inşa edilen skearear'ların hiper parametreleri hakkındaki görüşler rapor edilmiştir.

Tahmin gücünün artırılması: İlk olarak, maksimum oy veya tahmin sayısının ortalamasını almadan önce, algoritma tarafından üretilen ağaçların sayısı olan "n_estimators" adında bir hiper parametre vardır. Genel olarak, daha fazla ağaç performansı artırabilir ve tahminleri daha kararlı hale getirebilir, aynı zamanda hesaplama hızını yavaşlatabilir. Bir başka önemli hiperparametre, Rastgele Orman'ın düğümleri bölmeyi düşündüğü maksimum özellik sayısı olan "max_features" dır. Sklearn belgelerinde açıklanan çeşitli seçenekler sunar. Hız ile ilgili olarak tartışmak istediğimiz son önemli parametre "_s min_sample_leaf". Adından da anlaşılacağı gibi, bu, iç düğümleri ayırmak için gereken minimum yaprak sayısını belirler.

Model Hızını Artırma: "N_jobs", kaç işlemci motorunun hiper parametre kullanmasına izin verildiğini söyler. Değer "1" ise, yalnızca bir işlemci kullanabilir. "-1" değeri sınır olmadığı anlamına gelir. "Random_state", modelin çıktısını tekrarlanabilir hale getirir.. Belirli bir random_state değerine sahipse ve aynı hiper parametreler ve aynı eğitim verilerine sahipse, model her zaman aynı sonucu verecektir.

Son olarak, bir rastgele orman çapraz doğrulama yöntemi olan "*oob_score*"(oob örneklemesi olarak da bilinir) vardır. Bu örneklemede, verilerin yaklaşık üçte biri modeli eğitmek için kullanılmaz, ancak performansını değerlendirmek için kullanılabilir. Bu örneklere torbadan numune denir. Dışlama, çapraz onay yöntemine çok benzer, ancak başka hesaplama yükü daha azdır.

Rastgele orman algoritmasının avantaj ve dezavantajları şu şekildedir:

Daha önce de belirtildiği gibi, rastgele ormanın avantajlarından biri, regresyon ve sınıflandırma görevleri için kullanılabilmesidir ve girdi niteliklerine göreceli önemini görmek kolaydır.

Rastgele orman da çok kullanışlı ve kullanımı kolay bir algoritma olarak kabul edilir. Çünkü varsayılan hiper parametreler genellikle iyi tahminler üretir. Hiper parametre sayısı çok yüksek değildir ve anlaşılması kolaydır.

Makine öğrenmesindeki en büyük sorunlardan biri, çok güçlü olmasıdır, ancak rastgele orman sınıflandırıcıları için bu genellikle o kadar kolay değildir. Çünkü ormanda yeterli ağaç varsa, sınıflandırıcı bu model için uygun değildir.

Rastgele ormanların ana sınırlaması, birçok ağacın gerçek zamanlı tahmin için algoritmayı yavaş ve geçersiz hale getirmesidir. Genel olarak, bu algoritmalar hızlı bir şekilde eğitilebilir, ancak bir kez eğitildikten sonra, tahmin oluşturma hızı yavaştır. Daha doğru bir tahmin, modelin yavaşlamasına neden olan daha fazla ağaç gerektirmesidir. Çoğu pratik uygulamada, rastgele orman algoritması yeterince hızlıdır, ancak bazı durumlarda çalışma zamanı performansı önemlidir ve diğer yöntemler tercih edilebilir. Ve elbette, "rastgele orman" bir yorum aracı değil, öngörücü bir modelleme aracıdır. Bu, verilerinizdeki ilişkilerin tanımlarını arıyorsanız, diğer yöntemlerin tercih edildiği anlamına gelir.

Rastgele orman algoritmaları, bankalar, borsalar, tıp ve e-ticaret gibi birçok farklı alanda kullanılmaktadır. Örneğin, bankacılıkta, bankacılık hizmetlerini diğerlerinden daha sık kullanan ve borçları zamanında geri ödeyen müşterileri tanımlamak için kullanılır. Ayrıca, bu alandaki hileli bankaları isteyen hileli müşterileri tanımlamak için de kullanılabilir. Finansmanda, hisse senedinin gelecekteki davranışını belirlemek için kullanılır. Sağlık alanında, ilaçtaki bileşenlerin doğru kombinasyonunu tanımlamak ve hastalığı tanımlamak için hastanın tıbbi geçmişini analiz etmek için kullanılır. Son olarak, müşterilerin ürünü gerçekten sevip sevmediklerini belirlemek için e-ticaretteki rastgele orman algoritması kullanılır.

3.5. Destek Vektör Makinesi

Bir destek vektör makinesi (SVM), sınıflandırma ve regresyon analizi için verileri analiz eden makine öğrenme algoritmasıdır. SVM, verilere bakan ve bunları iki kategoriden birine ayıran denetimli bir öğrenme yöntemidir. Bir SVM, sıralanan verilerin bir haritasını mümkün olduğunca birbirinin arasındaki boşluklarla gösterir. SVM'ler metin kategorizasyonunda, görüntü sınıflandırmasında, el yazısı tanımada ve bilimlerde kullanılır.

Bir destek vektör makinesi aynı zamanda bir destek vektör ağı (SVN) olarak da bilinir. Bir destek vektörü makinesi, verileri iki kategoriye ayıran denetimli bir öğrenme algoritmasıdır. Zaten iki kategoride sınıflandırılmış bir veri dizisi ile eğitilmiştir, modeli başlangıçta eğitildiği gibi yapılandırmaktadır. Bir SVM algoritmasının görevi, yeni bir veri noktasının hangi kategoriye ait olduğunu belirlemektir. Bu, SVM'yi bir tür ikili olmayan doğrusal sınıflandırıcı yapar [38].

Bir SVM algoritması sadece nesneleri kategorilere yerleştirmekle kalmamalı, aynı zamanda aralarındaki marjları mümkün olduğunca geniş bir grafik üzerinde bulundurmalıdır.

SVM'nin bazı uygulamaları şunlardır:

- Metin ve köprü metni sınıflaması
- Görüntü sınıflandırma
- El yazısı karakterlerini tanıma
- Protein sınıflandırması dâhil olmak üzere biyolojik bilimler

3.6. Uzun Kısa Süreli Bellek

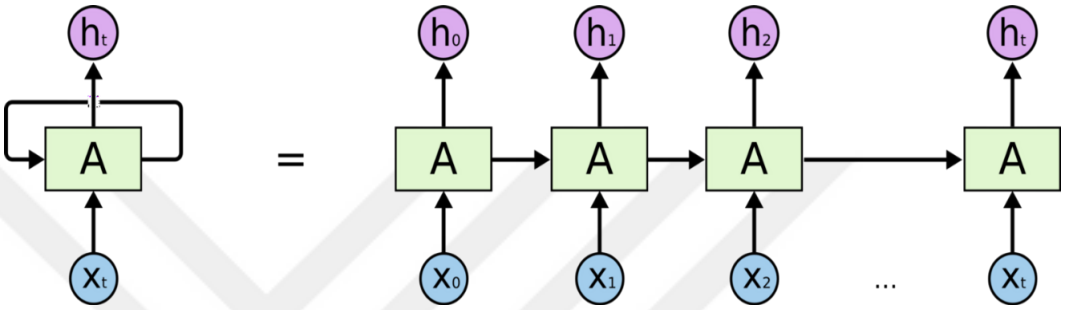
Uzun kısa süreli bellek (LSTM) birimleri veya blokları, tekrarlayan bir sinir ağı yapısının bir parçasıdır. Bu yapay zekâ programlarının insan düşüncesini daha etkin bir şekilde taklit etmesine yardımcı olabilecek belirli yapay bellek işlemlerinden faydalanmak için tekrarlayan sinir ağları yapılır.

Tekrarlayan sinir ağı, programın girdileri alma ve çıktıları oluşturma bağlamı için bağlam sağlamak için uzun kısa süreli hafıza blokları kullanır. Uzun kısa süreli hafıza bloğu, ağırlıklı girdiler, aktivasyon fonksiyonları, önceki bloklardan gelen girdiler ve nihai çıktılar gibi çeşitli bileşenlere sahip karmaşık bir ünitedir.

Üniteye uzun süreli bir hafıza bloğu denir, çünkü program daha uzun süreli hafıza oluşturmak için kısa süreli hafıza işlemlerine dayanan bir yapı kullanmaktadır.

Bu sistemler, örneğin doğal dil işlemede sıklıkla kullanılır. Tekrarlayan sinir ağı, belirli bir kelimeyi ya da fonemi almak için uzun kısa süreli hafıza bloklarını kullanır ve hafızanın bu tip girdileri ayırmada ve kategorize etmede yararlı olabileceği bir dizgede başkalarının bağlamında değerlendirir. Genel olarak, LSTM, tekrarlayan sinir ağlarında öncü olarak kabul edilmiş ve yaygın bir kavramdır. Şekil 3.4'e bakıldığında LSTM sistemi daha net anlaşılabilir.

LSTM ağlarının çalışma sistematiği ise bilgilerin bir adımdan diğerine akmasına izin veren bir döngüyü bütünleştirerek girdilerin bağlamsal bilgilerini tutmayı başarır. Bu döngüler tekrarlayan sinir ağlarını büyülüymüş gibi gösterir. Ancak bir saniye için düşünürsek, bu yazıyı okurken, önceki sözcükleri anlamanıza dayanarak her bir kelimeyi anlamaktasınız. Her şeyi çöpe atmaz ve her kelimede sıfırdan düşünmeye başlamazsınız. Benzer şekilde, LSTM tahminleri her zaman ağın girdilerinin geçmiş deneyimlerine göre düzenlenir.



Şekil 3.4. LSTM Diyagramı

Öte yandan, zaman geçtikçe, bir sonraki çıktının çok eski bir girdiye bağlı olma olasılığı o kadar az olur. Bu zamana bağlılık mesafesinin kendisi de öğrenilecek bağlamsal bir bilgidir [39].

3.7. Geçitli Tekrarlayan Ünite

Geçitli bir tekrarlayan ünite (GRU), örneğin konuşma tanımada hafıza ve kümelemeyle ilişkili makine öğrenme görevlerini gerçekleştirmek için bir dizi düğüm yoluyla bağlantıları kullanmak isteyen belirli bir tekrarlayan sinir ağı modelinin bir parçasıdır. Geçitli tekrarlayan üniteler, tekrarlayan sinir ağlarında sık görülen bir sorun olan kaybolan gradyan problemini çözmek için sinir ağı giriş ağırlıklarını ayarlamaya yardımcı olur. Genel tekrarlayan sinir ağı yapısının bir geliştirmesi olarak, kapılı tekrarlayan birimler, güncelleme kapısı ve sıfırlama kapısı olarak adlandırılan bir şeye sahiptir. Bu iki vektörü kullanarak, model, model boyunca bilgi akışını kontrol ederek çıktıları iyileştirir. Diğer tekrarlayan ağ modellerinde olduğu gibi, kapılı tekrarlayan ünitelere sahip modeller zaman içinde bilgileri tutabilirler - bu yüzden bu tür teknolojileri tanımlamanın en basit yollarından biri, "hafıza merkezli" bir tür sinir ağı olmalarıdır. Buna karşılık, kapılı yinelenen birimler içermeyen diğer tür sinir ağları çoğu zaman bilgi tutma yeteneğine sahip değildir [40].

Konuşma tanımanın yanı sıra, insan genomu, el yazısı analizi ve çok daha fazlası hakkında araştırma yapmak için kapılı yinelenen birimleri kullanan sinir ağı modelleri kullanılabilir. Bu

yenilikçi ağların bazıları borsa analizinde ve devlet çalışmalarında kullanılmaktadır. Birçoğu, makinelerin bilgiyi hatırlama becerilerinin simülasyonundan yararlanır.

3.8. Yerel İkili Örüntüler

Yerel ikili modeller (LBP) bilgisayar vizyonunda sınıflandırma için kullanılan bir tür görsel tanımlayıcıdır. LBP 1990'da önerilen Doku Spektrumu modelinin özel bir halidir. LBP ilk olarak 1994 yılında tanımlanmıştır. O zamandan beri doku sınıflandırması için güçlü bir özellik olduğu bulunmuştur; Ayrıca, LBP'nin yönlendirilmiş gradyanlar (HOG) tanımlayıcısının Histogramı ile birleştirildiğinde bazı veri kümelerinde algılama performansını önemli ölçüde artırdığı tespit edilmiştir. Arkaplan çıkarma alanındaki orijinal LBP'nin bazı iyileştirmelerinin bir karşılaştırması 2015 yılında Silva ve diğerleri tarafından yapılmıştır [41]. LBP'nin farklı versiyonlarının tam bir araştırması Bouwmans ve arkadaşları tarafından yapıldı [42].

LBP özellik vektörü en basit şekliyle aşağıdaki şekilde oluşturulur:

• İncelenen pencereyi hücrelere bölün (örneğin, her hücre için 16x16 piksel).

• Hücredeki her piksel için, pikseli 8 komşusunun her biriyle karşılaştırın (sol üstte, sol ortada, sol altta, sağ üstte vb.). Pikselleri bir daire boyunca, yani saat yönünde veya saat yönünün tersine izleyin.

• Merkez pikselin değerinin komşunun değerinden büyük olması durumunda, "0" yazın. Aksi takdirde, "1" yazın. Bu, 8 basamaklı bir ikili sayı verir (kolaylık sağlamak için genellikle ondalık değerine dönüştürülür).

• Hücrenin üzerindeki histogramı, meydana gelen her "sayının" sıklığının (yani her bir kombinasyonu hangi piksellerin daha küçük ve merkezden daha büyük olduğunu) hesaplamalısınız. Bu histogram 256 boyutlu bir özellik vektörü olarak görülebilir.

• İsteğe bağlı olarak histogramı normalleştirmelisiniz.

• Tüm hücrelerin birleştirilmiş (normalize edilmiş) histogramları elde edilir. Bu, tüm pencere için bir özellik vektörü verir.

Özellik vektörü artık görüntüleri sınıflandırmak için Destek vektör makinesi, aşırı öğrenme makineleri veya diğer bazı makine öğrenme algoritmaları kullanılarak işlenebilir. Bu sınıflandırıcılar yüz tanıma veya doku analizi için kullanılabilir.

Orijinal operatöre faydalı bir uzantı, sözde tek tip kalıptır. Bu özellik vektörünün uzunluğunu azaltmak ve basit bir rotasyon değişmez tanımlayıcısı uygulamak için kullanılabilir. Bu fikir, bazı ikili desenlerin doku görüntülerinde diğerlerinden daha yaygın olarak ortaya çıkmasıyla motive olur. İkili model en fazla iki 0-1 veya 1-0 geçişi içeriyorsa, yerel bir ikili düzen tek tip olarak adlandırılır. Örneğin, 00010000 (2 geçiş) tek tip bir kalıptır, 01010100 (6 geçiş) değildir. LBP histogramının hesaplanmasında histogram, her üniform model için ayrı bir hazneye sahiptir ve tek tip olmayan modellerin tümü tek bir hazneye atanır. Tek tip desenler kullanarak, özellik vektörünün

tek bir hücre için uzunluğu 256'dan 59'a düşer. 58 tek tip ikili desen 0, 1, 2, 3, 4, 6, 7, 8, 12, 14, 15 16, 24, 28, 30, 31, 32, 48, 56, 60, 62, 63, 64, 96, 112, 120, 124, 126, 127 tam sayılarına karşılık gelir.

3.9. Yönlendirilmiş Gradyanların Histogramı

HOG özellik tanımlayıcıları ve uzantıları, derin sinir ağlarının (DNN) son zamanlarındaki başarılarıyla uzaktan rekabet edebilecek nesnelerin tespiti ve yerelleştirilmesi için seçeneklerden biri olmaya devam etmektedir. Uydu görüntüleri için, cep telefonuna veya kişisel kamera görüntülerine kıyasla daha küçük nesne boyutu (piksel cinsinden) ile birlikte neredeyse sabit görüş açısı, birçok bilgisayar görme görevini basitleştirir. Örneğin, otomobilleri tanımak için bir makine öğrenme sistemi kurmayı düşünün. Trafik kameralarından gelen görüntüleri kullanarak yapılan makine öğrenmesi, bir otomobilin sürekli bir açıyla (ön, yan vb.) Tanınması için bir model oluşturmayı gerektirir; sinir ağları, öğrenilmiş özellikler ile bu tür modellerin oluşturulmasında mükemmeldir. Bununla birlikte, uydu görüntülerinde makine öğrenmesi, yalnızca sabit bir nadir bakış açısı varsayarak, bir arabanın üst siluetinin tanınmasını gerektirir. Ayrıca, uydu görüntülerinde ilgilenilen nesneler genellikle yalnızca birkaç piksel boyutundadır ve bir nesneyi tanımlamak için kullanılan tipik olarak birçok özellik bulanık olabilir, bu nedenle yalnızca nesne ana hatlarını farklılaştırılabilir halde bırakır [43]. HOG tanımlayıcıları bu tür anahat bilgileri yakalar ve sinir ağlarına daha basit, daha az güçlü ve daha hızlı (~ 20x) alternatiflerdir. Ek olarak, HOG özellikleri bir dizüstü bilgisayarın veya bilgi işlem kümesinin CPU'ları aracılığıyla çıkarılabilir ve tüm kullanıcılar tarafından kullanılamayacak yüksek performanslı grafiksel işlem birimlerine (GPU) dayanması gerekmez.

Kısaca, bir HOG tanımlayıcısı, gri tonlamalı görüntülerin kontur ve siluet bilgilerini yakalayan görüntü gradyanlarını hesaplayarak hesaplanır. Gradyan bilgisi, 1-D'lik bir yönelme histogramı içinde toplanır, böylece bir 2-D görüntüsünü rastgele ormanlar, destek vektör makineleri veya lojistik regresyon sınıflandırıcıları gibi makine öğrenme algoritmaları için girdi oluşturan daha küçük bir 1-D vektörüne dönüştürür.

3.10. Çok Katmanlı Algılayıcılar

Bu çalışma, kötü amaçlı yazılım dosyalarını gri tonlu görüntülere dönüştüren bellek dökümü verilerine dayanan dinamik bir kötü amaçlı yazılım analiz yöntemi önermekte, görüntünün doku özelliklerini ve görüntüleri sınıflandırmak için Multi-Layer Perceptron (Çok katmanlı algılayıcı, MLP) sınıflandırma yöntemini kullanmaktadır. Kötü amaçlı yazılımların çalıştırılması sırasında, bellekte depolanan veriler, algılama yoluyla zararlı olup olmadığını belirlemek için yeterli bilgiye sahiptir. Bellek verileri bir döküm dosyası oluşturmak için çıkarılır sonrasında gri tonlamalı bir

görüntüye dönüştürülür ve görüntüyü eşit bir boyuta dönüştürmek için görüntü ölçeklendirme algoritması kullanılır. En sonunda ise kötü amaçlı yazılımları sınıflandırmak için bu sınıflandırıcılar kullanılır [44].

Bu çalışma temel olarak 3 noktaya katkıda bulunmaktadır:

1. Belleğe dayalı bir kötücül yazılım çıkarma yöntemi önerilmektedir. Sanal alan, kötü amaçlı yazılım işlem belleğini izlemek için kullanılır ve bellek veri döküm dosyası gri tonlamalı bir görüntüye dönüştürülür ve özellik, bir degrade histogramı (HOG) kullanılarak çıkarılır. Belli olmayan bir dereceye kadar bu yöntem, dinamik analizin kötü amaçlı yazılımdan kaçınma sorununu çözebilir.

2. Bellek dökümü verilerini kullanarak gri tonlamalı görüntüye dönüştürmek için bir yöntem önerilmektedir. Çok katmanlı bir algılayıcı sınıflandırıcı ile birlikte, kötü amaçlı yazılım etkili bir şekilde sınıflandırılabilir.

3. Önerilen yöntemin etkinliği, gerçek sınıflandırma için gerçek arka kapı kötü amaçlı yazılım veri seti kullanılarak ve deneysel değerlendirme yapılarak doğrulanır.

3.11. Lineer Ayırt Edici Analiz(LDA)

Lineer Ayırt Edici Analiz, Makine Öğrenimi ve kalıp sınıflandırma uygulamalarında ön işleme adımı olarak kullanılan boyutsallık azaltma tekniğidir. Boyutsallık azaltma tekniklerinin temel amacı, özellikleri yüksek boyutlu bir alandan daha düşük boyutlu bir alana dönüştürerek, fazladan ve bağımlı özellikleri ortadan kaldırarak boyutları azaltmaktır [45].

Lineer Ayırt Edici Analiz, etiketleri dikkate alan denetimli bir sınıflandırma tekniğidir. Bu boyutsallık azaltma kategorisi, biyometri, biyoinformatik ve kimyada kullanılır.

3.12. Torbalı Ağaçlar(BT)

Önyükleme Toplaması (veya kısaca torbalı ağaçlar), basit ve çok güçlü bir topluluk yöntemidir. Torbalı ağaçlar, Bootstrap prosedürünün tipik olarak karar ağaçları olan yüksek değişkenlikli bir makine öğrenme algoritmasına uygulanmasıdır.

1. Diyelim ki N gözlem ve M özelliği var. Gözlemden bir örnek değiştirme (rastgele) ile rastgele seçilir.

2. Gözlemler ve özelliklerin alt kümesini içeren bir model oluşturmak için bir özellikler alt kümesi seçilmiştir.

3. Eğitim verilerinde en iyi ayrımı veren alt kümeden bir özellik seçilmiştir.

4. Bu, birçok model oluşturmak için tekrarlanır ve her model paralel olarak eğitilir.

5. Tahmin, tüm modellerden tahminlerin toplanmasına dayanarak verilir.

Karar ağaçları ile uğraşırken, eğitim verilerinden daha fazla yararlanan ağaçlardan daha az endişe duymaktayız. Bu nedenle ve verim alabilmek için, bireysel karar ağaçları derin bir şekilde büyür (örneğin, ağacın her yaprak düğümünde az sayıda eğitim örneği) ve ağaçlar budanmaz. Bu ağaçların hem yüksek değişkenliği hem de düşük yanlılığı olacaktır. Bunlar, torbalama kullanarak tahminleri birleştirirken alt modellerin önemli karakteristikleridir. Karar ağaçları torbalanırken tek parametreler örnek sayısı ve dolayısıyla bunları içerecek ağaç sayısıdır. Bu, aktif olduktan sonra çalışan ağaç sayısını artırarak doğrulukta iyileşmeyi göstermeye başlayana kadar seçilebilir [46].

3.13. Evrişimsel Sinir Ağı

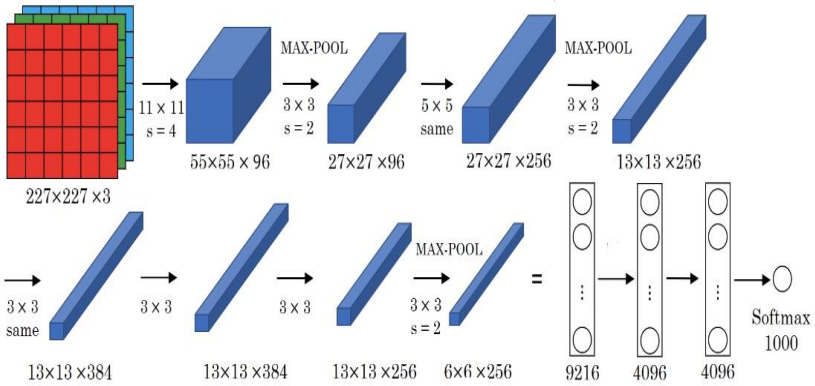
Bir evrişimsel sinir ağı (CNN), verileri analiz etmek, denetimli öğrenme için bir makine öğrenme birimi algoritması olan algılayıcıları kullanan spesifik bir yapay sinir ağı türüdür. CNN'ler görüntü işleme, doğal dil işleme ve diğer bilişsel görevler için geçerlidir. Hem üretici hem de betimleyici görevleri gerçekleştirmek için derin öğrenmeyi kullanan, genellikle tavsiye veren sistemler olmasının yanı sıra doğal dil işleme (NLP) ile birlikte görüntü ve video tanıma özellikleri de içeren makine vizyonu kullanan güçlü görüntü işleme yapabilen bir yapay zekâdır (AI). Bir evrişimsel sinir ağı ayrıca bir ConvNet olarak da bilinir [47].

Diğer yapay sinir ağları gibi, evrişimli bir sinir ağının bir giriş katmanı, bir çıkış katmanı ve çeşitli gizli katmanları vardır. Bu katmanların bazıları, art arda gelen katmanlara sonuçları aktarmak için matematiksel bir model kullanan konvulüsyonlardır. Bu, insanın görsel korteksindeki bazı eylemleri simüle eder.

CNN'ler, daha karmaşık bir modelin, farklı biyolojik insan beyni aktivitesini taklit eden sistemler sunarak yapay zekânın gelişimini zorladığı temel bir derin öğrenme örneğidir. Bu evrişimsel sinir ağı yöntemleri de kendi içinde katman karmaşıklığına göre farklı isimler alabilmektedir. Testlerimizi gerçekleştirirken en çok kullanılan yöntemlerimiz arasında yer almaktadır.

3.14. AlexNet

Bu, 2012'de tekrar popüler hale gelen evrişimli sinir ağı modelleri ve derin öğrenme sağlayan ilk çalışmadır. Alex Krizhevsky, Ilya Sutskever ve Geoffrey Hinton tarafından geliştirildi [48]. Temel olarak, sürekli evrişim ve birleştirme katmanlarına sahip olduğu için LeNet modeline çok benzer. Şekil 3.5'de görüldüğü gibi bir etkinleştirme fonksiyonu olarak, birleştirme katmanında ReLU (doğrultulmuş doğrusal birim) kullanılır ve maksimum birleştirme kullanılır.



Şekil 3.5. AlexNet diyagramı

Evrişimli sinir ağı modelleri ve derin öğrenme sağlayan ilk uygulama 2012'de yine popüler oldu Bu daha büyük ve daha derin ağ modeli, paralel çift GPU'lu (grafik işlem birimi) iki parçalı bir modeldir. Yaklaşık 60 milyon parametre hesaplanmıştır [49]. ImageNet ILSVRC yarışmasında, sınıflandırma doğruluğu aniden %74,3'ten %83,6'ya yükselmiştir, bu da görüntü sınıflandırma sorunları için bir dönüm noktasıdır.

Bu tez çalışması için elimizdeki veri seti AlexNet ile eğitildiğinde testlerdeki en yüksek değer %81 ile Subspace Discriminant (Altuzay ayırıcı) sınıflandırıcısıyla elde edildi.

3.15. GoogleNet

ILSVRC 2014 yarışmasının galibi Google'dan GoogLeNet'ti (aynı zamanda Inception V1). % 6.67'lik ilk 5 hata oranına ulaştı! Bu, mücadeleyi düzenleyenlerin değerlendirmek zorunda kaldıkları insani performansa çok yakındı [49].

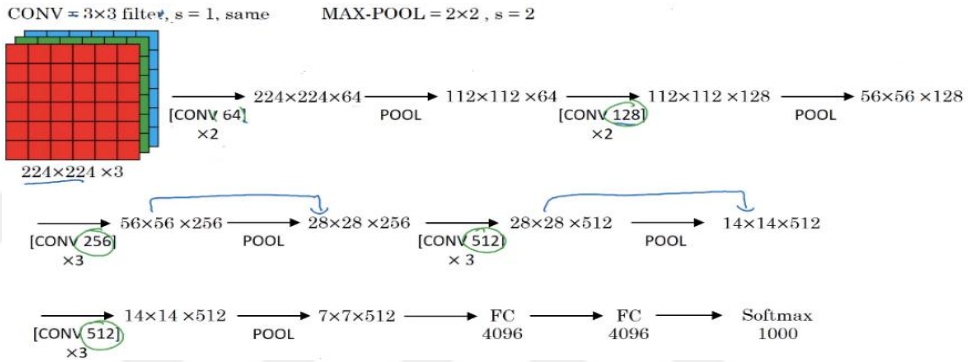
Anlaşıldığı üzere, bu aslında oldukça zordu ve GoogLeNets'in doğruluğunu yenmek için bazı insan eğitimi gerektirdi. Birkaç günlük eğitimden sonra, insan uzman (Andrej Karpathy)% 5.1 (tek model) ve% 3.6 (topluluk) ilk 5 hata oranına ulaşmayı başardı. Ağ, LeNet'ten esinlenen bir CNN kullandı ancak başlangıç modülü olarak adlandırılan yeni bir öğe uyguladı. Toplu normalleştirme, görüntü bozulmaları ve RMSprop kullandı.

Bu modül, parametre sayısını büyük ölçüde azaltmak için çok küçük kıvrımlara dayanmaktadır. Mimarileri 22 kat derin CNN'den oluşmaktaydı, ancak parametre sayısını 60 milyondan (AlexNet) 4 milyona düşürdü.

Bu tez çalışması için elimizdeki veri seti GoogleNet ile eğitildiğinde en yüksek değer %81.9 olarak Subspace Discriminant sınıflandırıcısıyla elde edildi.

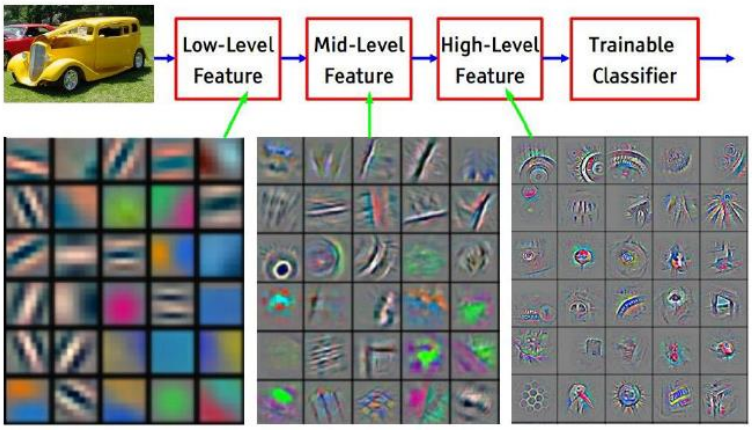
3.16. VGG-16

Bu basit bir ağ modelidir ve önceki modelden en önemli fark 2 veya 3 katlamalı ekleme kullanılmasıdır. Tam bağlantı (FC) katmanında, 7x7x512 = 4096 nöronların bir öznitelik vektörüne dönüştürülecektir. 1000 sınıfın Softmax performansı, iki FC katmanının çıktısında hesaplanır. Yaklaşık 138 milyon parametre hesaplandı. Şekil 3.6'de gösterildiği gibi diğer modellere benzer şekilde, matrisin girişten çıkışa kadar yükseklik ve genişlik boyutları azalırken derinlik değeri (kanal sayısı) artar [50].



Şekil 3.6. VGG-16 diyagramı

Modelin her evrişim katmanı çıkışında farklı ağırlıklara sahip filtreler hesaplanır ve katman sayısı artıkça filtrelerde oluşan öznitelikler görüntünün 'derinliklerini' simgelemektedir. Bununla ilgili en net görsel Şekil 3.7'de verilmiştir.



Şekil 3.7. Imagenet'te eğitilmiş evrişimsel yapay sinir ağının görselleştirilmesi [51]

Bu tez çalışması için elimizdeki veri seti VGG-16 ile eğitildiğinde en yüksek değer %83.2 ile Subspace Discriminant sınıflandırıcısıyla elde edildi.

3.17. VGG-19

Basit bir ağ modeli olup Şekil 3.8'de gruplandırıldığı gibi VGG-16'dan en önemli farkı evrişim katmanlarının 2'li, 3'lü ya da 4'lü kullanılması ve toplamda 19 adet katmana sahip olmasıdır [52]. Sonuç olarak, ağ çok çeşitli görüntüler için zengin özellik temsillerini öğrenmiştir. Ağın görüntü giriş boyutu 224x224'tür.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

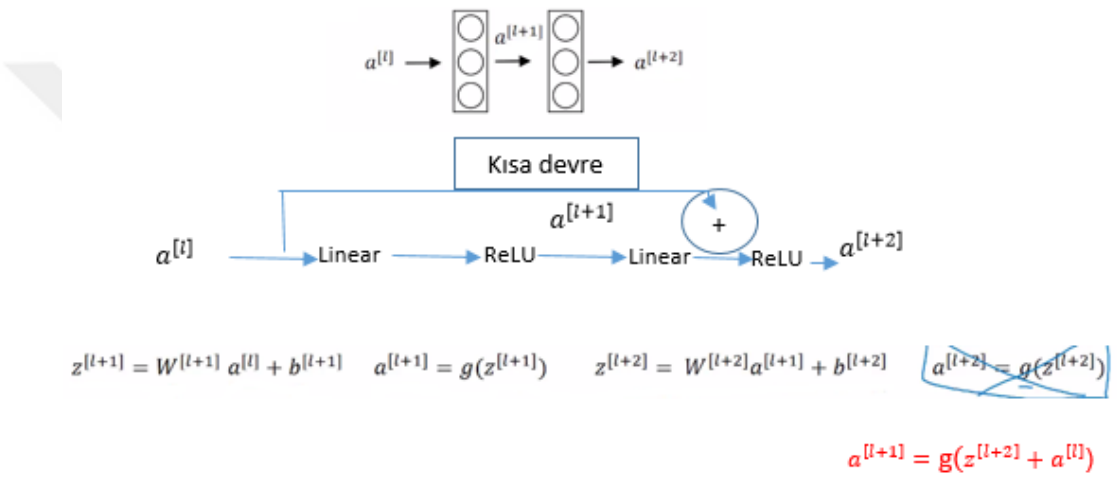
Şekil 3.8. VGG-19 mimarisinin detayları[52]

Bu tez çalışması için elimizdeki veri seti VGG-19 ile eğitildiğinde en yüksek değer %83.7 olarak Quadratic SVM sınıflandırıcısıyla elde edildi.

3.18. ResNet

Ağ modelinin gerçek anlamda derinleşmeye başladığı kendinden önceki modellerden farklı bir mantığı barındıran ResNet; artık değerlerin (residual value) sonraki katmanlara besleyen blokların (residual block) modele eklenmesiyle oluşmaktadır. ResNet bu özelliği ile klasik bir model olmaktan çıkmaktadır [53].

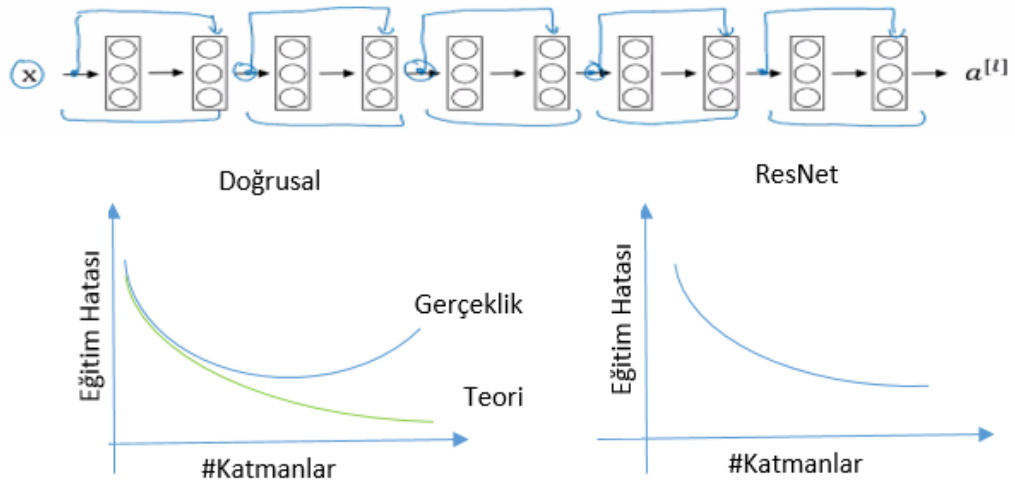
Doğrusal ve ReLU arasında iki katmanda bir eklenen bu değer Şekil 3.9'da gösterildiği gibi sistemdeki hesabı değiştirir. Önceden gelen a[l]değeri, a[l+2] hesabına eklenmiş olur.



Şekil 3.9. Doğrusal Ve ReLU hesaplama işlemleri

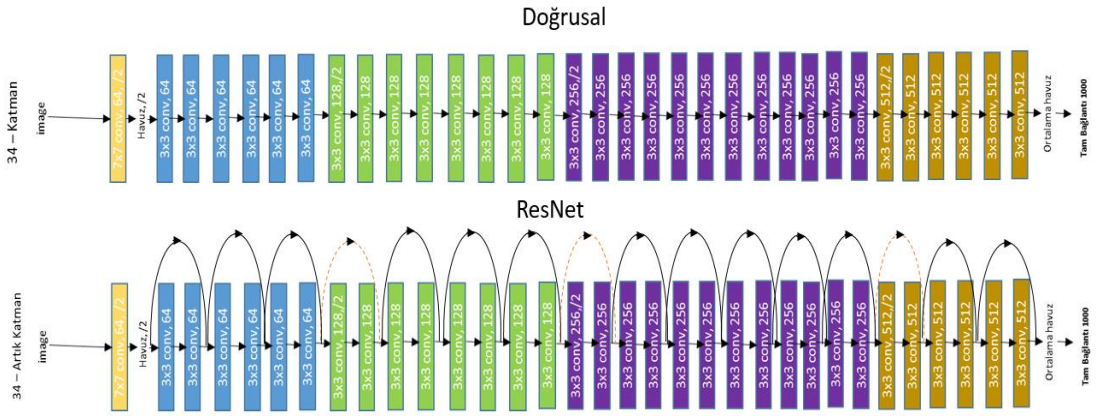
Teorik olarak, modelde katman sayısı arttıkça başarımın artacağı düşünülür. Ancak gerçekte böyle olmadığı deneyimlenmiştir. Buradan hareketle ResNet modeli oluşturulmuştur. Böylece w[l+2]=0 olduğu durumda yeni teoriye göre a[l+2]=b[l+2] olur. Bu aslında (vanishing gradient) istenmeyen bir durumdur.

Ancak Şekil 3.10'da gösterildiği gibi artık değer (residual) beslemesi yeni çıkış eşitliğini iki önceki katmandan gelen a[l] değeri o anki ağırlık 0 olsa bile öğrenme hatasını optimize eder. Daha hızlı eğitilir.



Şekil 3.10. Eğitim hatalarında karşılaşılan durumlar

Doğrusal ağ ve ResNet Ağı'nın açık gösterimi Şekil 3.11'de şematize edilmiştir. ResNet modeli 18 katman derinliğe sahip bir sinir ağı içerisinde yer alıyorsa bu ResNet-18, 50 katman derinliğe sahip bir sinir ağı içerisinde yer alıyorsa bu ResNet-50, eğer 101 katman derinliğe sahip bir sinir ağı içerisinde yer alıyorsa bu da ResNet-101 olarak ifade edilmektedir.

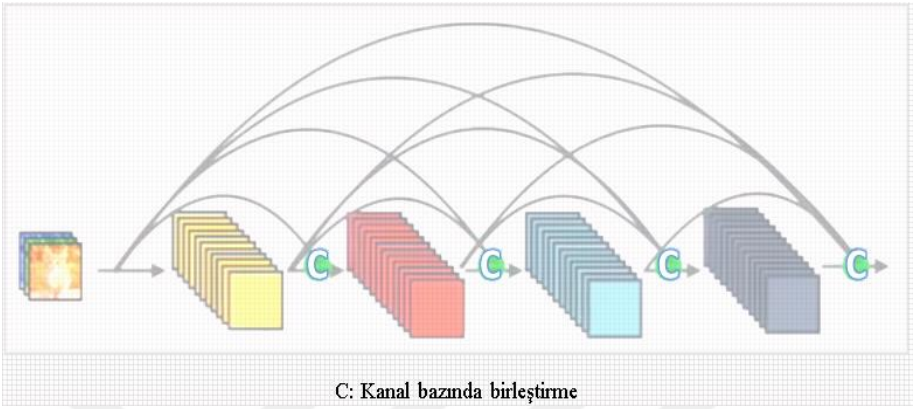


Şekil 3.11. Doğrusal ağ ve ResNet ağının açık gösterimi

Bu tez çalışması için elimizdeki veri seti ResNet-18 ile eğitildiğinde en yüksek değer %83.0 olarak Linear SVM sınıflandırıcısıyla, ResNet-50 ile eğitildiğinde %85.1 olarak Subspace Discriminant sınıflandırıcısıyla, ResNet-101 ile eğitildiğinde %88.5 olarak ise Subspace Discriminant sınıflandırıcısıyla elde edildi.

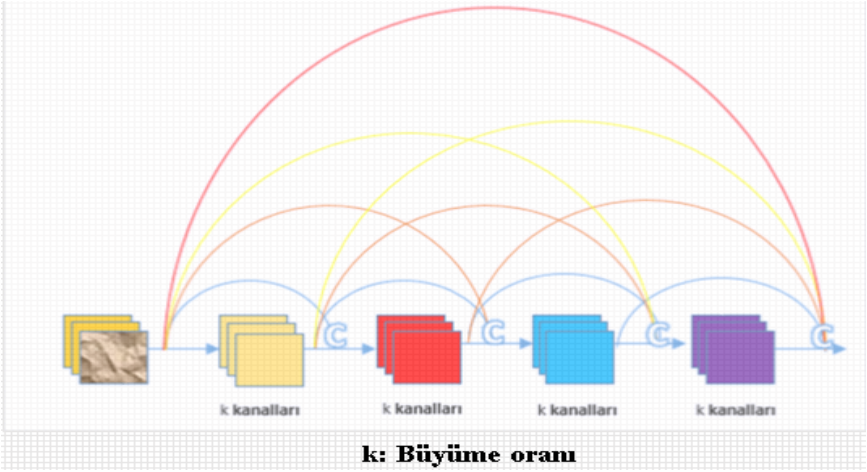
3.19. DenseNet-201

DenseNet'te Şekil 3.12'de gösterildiği gibi her katman, önceki tüm katmanlardan ek girişler alır ve kendi özellik eşlemelerinden sonraki tüm katmanlara geçer [54].



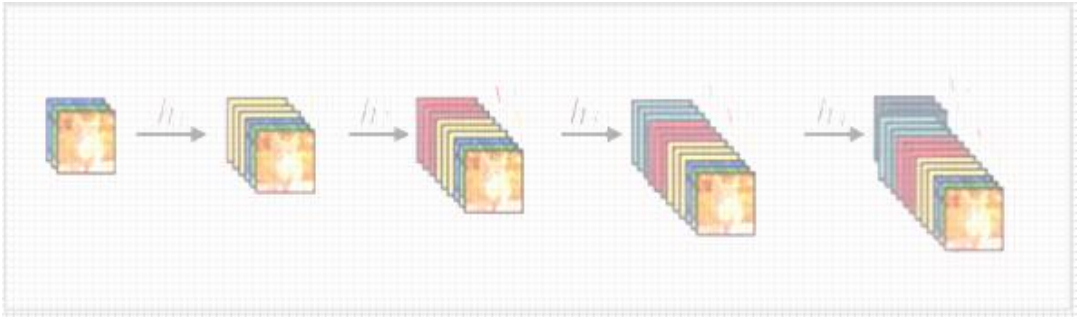
Şekil 3.12. DenseNet'teki bir Dense bloğu

Burada birleştirme kullanılır. Her katman, önceki tüm katmanlardan bir "kolektif bilgi" almaktadır. Başka bir açıdan bakacak olursak Şekil 3.13'de gösterildiği gibi her katman önceki tüm katmanlardan özellik haritaları aldığından, ağ daha ince ve kompakt olabilir, yani kanal sayısı daha az olabilir.



Şekil 3.13. DenseNet'te k büyüme oranı ile Dense bloğu

Büyüme oranı k, her katman için ilave kanal sayısını ifade etmektedir. Böylece, daha yüksek hesaplama verimliliğine ve bellek verimliliğine sahiptir. Şekil 3.14'de belirtildiği gibi ileri yayılma sırasında birleştirme kavramı gösterilmektedir:



Şekil 3.14. İleri yayılma sırasında birleştirme

Bu tez çalışması için elimizdeki veri seti DenseNet-201 ile eğitildiğinde en yüksek değer %85.0 olarak Subspace Discriminant sınıflandırıcısıyla elde edildi.

3.20. SqueezeNet

Derin öğrenme 18 katman derinliğinde evrişimsel bir sinir ağıdır. Sonuç olarak, ağ çok çeşitli görüntüler için zengin özellik temsillerini öğrenmiştir. Bu işlev, SqueezeNet v1.0'a benzer bir doğruluğa sahip olan, ancak tahmin başına daha az kayan nokta işlemi gerektiren bir SqueezeNet v1.1 ağı döndürür [55]. Ağın görüntü giriş boyutu 227 x 227'dir.

Bu tez çalışması için elimizdeki veri seti SqueezeNet ile eğitildiğinde en yüksek değer %82.3 ile Subspace Discriminant sınıflandırıcısıyla elde edildi.

3.21. MobileNet

Görüntü Sınıflandırma ve Mobil Vizyon için bir evrişimsel sinir ağları mimari modelidir. Başka modeller de var, ancak MobileNet'i özel yapan şey transfer öğrenmesini çalıştırmak veya uygulamak için çok daha az hesaplama gücü kullanmasıdır.

Bu, mobil cihazlara, gömülü sistemlere ve GPU'su olmayan veya düşük hesaplama verimliliğine sahip ve sonuçların doğruluğundan önemli ölçüde ödün veren bilgisayarlar için mükemmel bir uyum sağlar.

Ayrıca, tarayıcılar hesaplama, grafik işleme ve depolama konusunda sınırlamalara sahip oldukları için web tarayıcıları için de uygundur. Mimari yapısından bahsedecek olursak Hafif sıklet ve derin sinir ağları oluşturmak için derinlemesine ayrılabilir kıvrımlar kullanan aerodinamik bir mimariye dayanan mobil ve gömülü görüş uygulamaları için MobileNet'ler önerilmektedir [56].

Gecikme ve doğruluk arasında etkili bir şekilde transfer yapan iki basit küresel hiper parametre tanıtıldı. MobileNet'in ana katmanı Derinlikle Ayrılabilir Konvolüsyon olarak adlandırılan derinlemesine ayrılabilir filtrelerdir. Ağ yapısı, performansı artırmak için başka bir

faktördür. Son olarak, genişlik ve çözünürlük gecikme ve doğruluk arasında değişecek şekilde ayarlanabilir.

Bu tez çalışması için elimizdeki veri seti MobileNet ile eğitildiğinde en yüksek değer %85.3 olarak Subspace Discriminant sınıflandırıcısıyla elde edildi.

4. MATERYAL VE METOT

.Bu bölümde seminer çalışmalarımız ile birlikte başlanan ve tezin yazılmaya başlanması ile birlikte devam eden süreçte yaptığımız testler ve çalışmalarımıza yer vererek kendi önerimizi aktarmayı amaçladık. Bu tez çalışmamızda tez önerisinde değindiğimiz noktaları tek tek elden geçirerek en iyi sonuçları elde edebildiğimiz noktaya kadar çalışmalarımıza yön verdik. Çalışmalarımıza yön vererek gerçekleştirdiğimiz testler, çalışmalar ve sonuçları bu bölümde anlatarak kendi oluşturduğumuz yöntemin diğerleriyle karşılaştırmalı sonuçlarını paylaşmayı planlamaktayız.

4.1. Materyal

Bu kısımda kullanılan veri seti tanımlanmıştır. Bu veri seti literatürde Malimg veri seti olarak adlandırılmaktadır. Bu veri setinde toplamda 9339 kötücül yazılıma ait dump edilmiş veri bulunmaktadır. Dump edilmiş veriler gri seviyeli imge formunda bulunmaktadır ve bu işlem Şekil 4.1' de açıklanmıştır [57].

Şekil 4.1. Malimg veri kümesinin frekans dağılımı

Tablo 4.1' de Malimg veri seti açıklanmaktadır. Bu tablodan da görüldüğü gibi toplamda 9 farklı aileye ait 25 sınıf bulunmaktadır.

Tablo 4.1. Malimg veri kümesinde bulunan kötü amaçlı yazılım aileleri [57]

No	Aile	Aile Adı	Değişken Sayısı
01	Dialer	Adialer.C	122
02	Backdoor	Agent.FYI	116
03	Worm	Allaple.A	2949
04	Worm	Allaple.L	1591
05	Trojan	Alueron.gen!J	198
06	Worm AutoIT	Autorun.K	106
07	Trojan	C2Lop.p	146

Tablo 4.1. (Devamı)

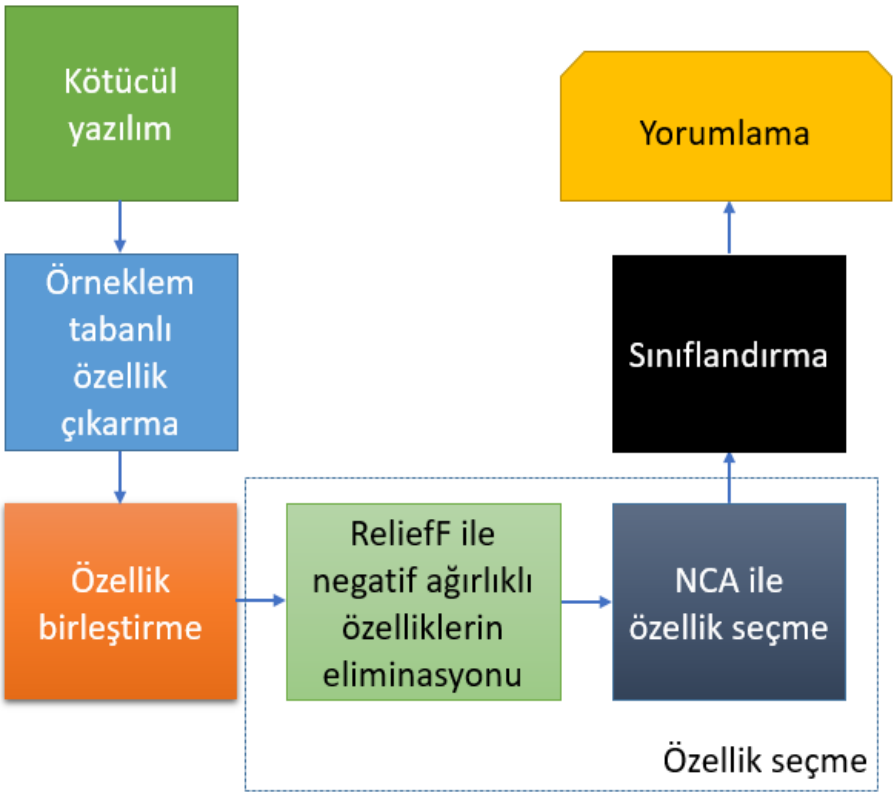
No	Aile	Aile Adı	Değişken Sayısı
08	Trojan	C2Lop.gen!G	200
09	Dialer	Dialplatform.B	177
10	Trojan Downloader	Dontovo.A.	162
11	Rogue	Fakerean	381
12	Dialer	Instantacces	431
13	PWS	Lolyda.AA 1	213
14	PWS	Lolyda.AA 2	184
15	PWS	Lolyda.AA 3	123
16	PWS	Lolyda.AT	159
17	Trojan	Malex.gen!J	136
18	Trojan Downloader	Obfuscator.AD	142
19	Backdoor	Rbot!gen	158
20	Trojan	Skintirim.N	80
21	Trojan Downloader	Swizzor.gen!E	128
22	Trojan Downloader	Swizzor.gen!I	132
23	Worm	VB.AT	408
24	Trojan Downloader	Wintrim.BX	97
25	Worm	Yuner.A	800

4.2. Metot

Bu bölümde, daha önceki testlerde örneklem tabanlı bir yöntem kullanarak bulunan sonuçlardan bahsedilecektir. Bu kısımda temel olarak 3 ayrı yöntem kullanılmıştır ve sonuçlar elde edilmiştir. Bu yöntemlerden ilki örneklem tabanlı kötücül yazılım sınıflandırma yöntemidir. İkinci yöntemde ise transfer öğrenme kullanılarak derin öğrenme ağlarının performansları analiz edilmiştir. Üçüncü yöntemde, 9 derin ağ özellik çıkarıcı olarak kullanılmış ve yeni bir kompozit öğrenme modeli önerilmiştir.

4.2.1. Örneklem Tabanlı Kötücül Yazılım Sınıflandırma Yöntemi

Bu yöntemde, literatürde sıklıkla kullanılan makine öğrenmesi modellerinde olan örneklem tabanlı özellik çıkarma yöntemi kullanılmıştır. Özellik seçme aşamasında hibrit bir özellik seçici kullanılmıştır. Hibrit özellik seçicinin kullanılmasının temel amacı hem ReliefF (Rölyef) hemde NCA'nın (Neighborhood Component Analysis, Komşuluk Bileşen Analizi) verimliliğini bir arada kullanabilmektir. Önerilen yöntemin blok diyagramı Şekil 4.2' de verilmiştir.



Şekil 4.2. Sistemin blok diyagramı

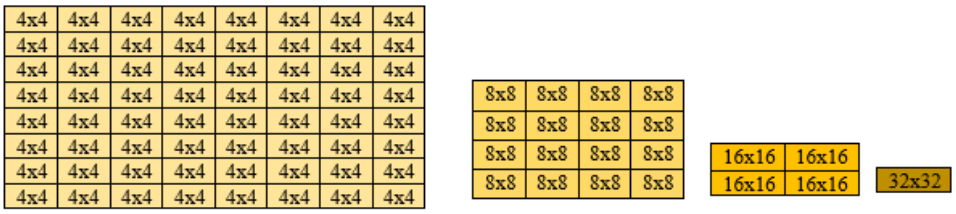
Şekil 4.2' de gösterildiği gibi yöntem 4 temel fazdan oluşmaktadır. Bu fazlar özellik çıkarma, özellik birleştirme, özellik seçme ve sınıflandırmadır. Yöntemin adımları aşağıdaki gibi verilmiştir.

Özellik Çıkarma: Özellik çıkarma aşamasında örneklem tabanlı bir yöntem kullanılmıştır. Kullanılan veri setinde bulunan kötücül yazılımlar 32 x 32 boyutunda gri seviyeli görüntü dosyası olarak kodlanmaktadır. Bu dosyalar byte-code yöntemi kullanılarak kodlanmıştır. Kötücül yazılım dosyaları görüntü formatında olduğu için, görüntü işleme yöntemleri kullanılarak bu dosyalardan özellik çıkartılabilmektedir. Bu çalışmada, özellik çıkarmak için başlangıç derin öğrenme yöntemine (inception network) benzer bir istatistiksel yöntem kullanılarak özellik çıkarma işlemi gerçekleştirilmiştir. Önerilen yöntemin özellik çıkarma aşamasının adımları aşağıdaki gibi verilmiştir.

Adım 0: Byte kod olarak kodlanmış kötücül yazılım veri setini yükle.

Adım 1: Her bir kötücül yazılımı ifade eden 32 x 32 boyutundaki gri seviyeli görüntüleri al.

Adım 2: Her bir görüntüyü sırasıyla 4 x 4, 8 x 8, 16 x 16 ve 32 x 32 boyutundaki bloklara Şekil 4.3' te gösterildiği gibi ayır. Bu adım örnekleme adımıdır. Bu adımda birden fazla blok boyutu kullanıldığı için başlangıç ağlarına da benzetilmektedir.



Şekil 4.3. Önerilen başlangıç örneklem (inception exemplar) yöntemi

Adım 3: 32 x 32 boyutundaki gri seviyeli görüntü dosyasını minimum-maksimum normalizasyon işlemini kullanarak normalize et. Normalizasyon denklemi Denklem 4.1' de verilmiştir.

$$G^N = \frac{G - G_{min}}{G_{max} - G_{min}} \tag{4.1}$$

Denklem 4.1' de G orijinal görüntü, G_{min} orijinal görüntünün minimum piksel değer, G_{max} orijinal görüntünün maksimum piksel değeri, G^N ise 0 ile 1 arasında normalize edilmiş görüntü.

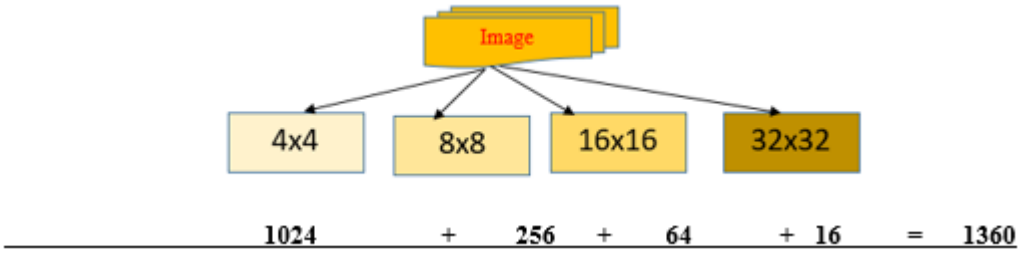
Adım 4: Şekil 4.3' de gösterilen her bir bloktan aşağıdaki denklemleri kullanarak 16 özellik çıkar. Kullanılan istatistiksel özelliklerin matematiksel ifadesi aşağıdaki gibi verilmiştir:

$$f1 = \min(\min(blok)) \tag{4.2}$$
$$f2 = \min(\max(blok)) \tag{4.3}$$
$$f3 = \max(\min(blok)) \tag{4.4}$$
$$f4 = \max(\max(blok)) \tag{4.5}$$
$$f5 = \max(\text{mean}(blok)) \tag{4.6}$$
$$f6 = \text{mean}(\text{mean}(blok)) \tag{4.7}$$
$$f7 = \text{std}(\text{std}(blok)) \tag{4.8}$$
$$f8 = \text{std}(\max(blok)) \tag{4.9}$$
$$f9 = \text{std}(\min(blok)) \tag{4.10}$$
$$f10 = \text{std}(\text{mean}(blok)) \tag{4.11}$$
$$f11 = \text{var}(\text{var}(blok)) \tag{4.12}$$
$$f12 = \text{var}(\text{mean}(blok)) \tag{4.13}$$
$$f13 = \text{var}(\text{std}(blok)) \tag{4.14}$$
$$f14 = \text{var}(\max(blok)) \tag{4.15}$$
$$f15 = \text{var}(\min(blok)) \tag{4.16}$$
$$[U, S, V] = SVD(blok), f16 = \text{S}(1,1) \tag{4.17}$$

Denklem 4-17' de $blok$ görüntüden elde edilen örtüşmeyen bloklardır ve Şekil 4.3' de verilmiştir, f çıkarılan özellikleri, $\min(.)$, $max(.)$, $mean(.)$, $std(.)$ ve $var(.)$ fonksiyonları sırasıyla minimum, maksimum, ortalama, standart sapma ve varyans fonksiyonlarını ifade etmektedir. $SVD(.)$ tekil değer ayrışımı, U, S ve V üniter, tekil ve dikey matrislerdir. Kullanılan bloklar iki boyutlu olduğu için fonksiyonlar iç içe kullanılmıştır.

Özellik birleştirme: Özellik birleştirme aşamasında farklı boyuttaki bloklar kullanılarak çıkartılan özellikler birleştirilir ve her bir örnek için 1360 özellik elde edilir.

Adım 5: Çıkarılan özellikleri birleştir. Şekil 4.4' te gösterildiği gibi birleştirilmiştir.



Şekil 4.4. Özellik birleştirme aşaması

Özellik Seçme: Bu aşamada literatürde yaygın olarak kullanılan yöntemlerden birisi olan Rölyef (ReliefF) yöntemi kullanılmıştır. ReliefF yöntemi kullanılarak gereksiz özellikler elimine edilmiştir. Gereksiz özelliklerin belirlenmesi aşamasında ise ReliefF yönteminin ürettiği ağırlıklar kullanılmıştır. ReliefF yönteminin sağladığı en önemli avantaj negatif ve pozitif ağırlıklar üretmesidir. Üretilen negatif ağırlıklar gereksiz özellikleri göstermektedir. Negatif özellikler elimine edildikten sonra seçilen özelliklere tekrardan rölyef algoritması uygulanır ve 64, 128, 256 ve 512 özellik seçilir. 10 adet en yakın komşuya sahip ReliefF algoritmasını kullanan tahmin edicileri sıralar. Giriş matrisi X, tahmin değişkenleri içerir ve y vektörü bir yanıt vektörü içerir. İşlev, en önemli işlemci dizinlerini içeren a'yı ve işlemcilerin ağırlıklarını içeren b'yi döndürür. Eğer y sayısalsa, relieff varsayılan olarak regresyon için RReliefF analizini gerçekleştirir. Aksi takdirde, sınıf başına k en yakın komşuyu kullanarak sınıflandırma için ReliefF analizine yardımcı olur. Şekil 4.2 üzerinde görsele dökülen bu işlemler adım adım gerçekleştirilmektedir. Bu işlemden sonra Komşuluk bileşen analizi (NCA) işlemine sokularak özellikler elde edilir.

Sınıflandırma: Örneklem tabanlı kullanılan bu yöntem ile veri seti üzerinde özellik çıkarma, özellik birleştirme, özellik seçme ve sınıflandırma olarak 4 basamaktan oluşan işlemler uygulanmıştır. Şekil 4.2'deki diyagramda belirtilen bu işlemler sonucunda toplamda 1360 adet

özellik seti çıkarılarak gereksiz(-) olanlar göz ardı edilip elenmiştir. Sınıflandırma aşamasında 5 adet sınıflandırıcı kullanılmıştır ve bu sınıflandırıcı karar ağacı (Fine Tree), doğrusal ayırıcı (Linear Discriminant Analysis), destek vektör makineleri (SVM), k en yakın komşu yöntemi (Fine kNN), torbalı ağaçlar (BT) sınıflandırıcılarıdır. Sonuçları elde etmek için sırasıyla 64, 128, 256 ve 512 özellik seçilmiştir. Elde edilen sonuçlar Tablo 4.2' de gösterilmiştir [58].

Tablo 4.2. Sınıflandırma Tablosu

Sınıflandırıcı	64 özellik	128 özellik	256 özellik	512 özellik
Fine Tree	78.4	78.8	77.9	*80.6*
LDA	77.9	80.9	81.5	*84.4*
SVM	82.4	83.3	83.6	*85.4*
Fine kNN	83.2	82.1	82.8	*83.6*
BT	**84.9**	**84.6**	**84.8**	***85.7***

Tablo 4.2'deki sonuçlara göre en yüksek değerler Torbalı ağaçlar(Bagged Trees) sınıflandırıcısında elde edilmiştir. Bagged Trees sınıflandırıcısında elde edilen en yüksek değer 512 özelliğin kullanıldığı sınıflandırma işleminde elde edilmiştir. Diğer sonuçlara da bakarak bir genelleme yapacak olursak özellik sayısı arttıkça başarı oranının yükseldiği ve en yüksek değerlerin %85.4 SVM ve %85.7 BT sınıflandırıcısına ait olduğu görülmektedir.

4.2.2. Transfer Öğrenme Tabanlı Kötücül Yazılım Sınıflandırma

Önerilen yöntemin yanı sıra Malimg kötücül yazılım veri setine derin öğrenme ağları uygulanmış. Bu aşamada transfer öğrenme modeli kullanılmıştır. Transfer öğrenme modelinde derin ağlar ImageNet [59] veri seti kullanılarak eğitilir ve elde edilen optimum ağırlıklar diğer sınıflandırma problemlerini çözmek için kullanılır. Derin öğrenme sonuçlarını elde etmek için ResNet-18, ResNet-50, ResNet-101, GoogLeNet, VGG-16, VGG-19, MobileNet, SqueezeNet ve DenseNet-201 derin ağları kullanılmıştır. Bu ağların ortak özellikleri bilgisayarda görü alanında sıklıkla kullanılan derin öğrenme modelleri olmalarıdır. Bu ağlar hem sınıflandırma hem de özellik çıkarıcı olarak kullanılmaktadır. Bu tez çalışmasında, derin ağlar özellik çıkarıcı olarak kullanılmıştır ve her bir ağ kullanılarak 1000 özellik elde edilmiştir. Elde edilen özellikler MATLAB Classification Learner (Sınıflandırma Öğreticisi) kullanılarak sınıflandırılmıştır. Bahsedilen araçta 23 adet sınıflandırıcı bulunmaktadır. Tablo 4.3' te en iyi performansa sahip 4 sınıflandırıcının sonuçları verilmiştir.

Tablo 4.3. Derin öğrenme yöntemlerine ait test sonuçları

	Subspace Discriminant	Quadratic SVM	Linear SVM	Cubic SVM
ResNet-18	82.0	82.8	**83.0**	82.8
ResNet-50	**85.1**	84.5	82.7	83.4
ResNet-101	**88.5***	84.8	84.3	83.3
GoogleNet	**81.9**	80.1	80.0	79.9
VGG16	**85.7**	82.0	82.5	*82.3*
VGG19	82.3	**83.7**	81.8	*81.3*
MobileNet	**85.3**	82.7	83.5	*82.2*
SqueezeNet	**82.3**	82.2	80.7	*82.0*
DenseNet201	**85.0**	84.5	83.9	*84.0*

Tablo 4.3' teki sonuçlardan da görüldüğü gibi en yüksek başarım oranının **%88.5** ile **ResNet-101'**e ait olduğu görülmektedir. Derin öğrenme modellerinin yüksek başarıma sahip oldukları görülmektedir. Bu başarımı arttırmak için bir sonraki bölümde kompozit derin ağ tabanlı bir yöntem önerilmiştir.

4.2.3. Kompozit Derin Ağlar Tabanlı Kötücül Yazılım Sınıflandırma Yöntemi

Tablo 4.3' te derin öğrenme modellerinin başarımları verilmektedir. Bu yöntemin temel amacı, derin ağların belirgin özelliklerini kullanarak yüksek başarıma sahip bir kötücül yazılım sınıflandırma yöntemi önermektir. Bu amaçla literatürde sıklıkla kullanılan 8 derin öğrenme ağı kullanılmıştır. Bu ağlar Tablo 4.3' te gösterilmiştir. Sadece GoogLeNet düşük başarıma sahip olduğu için göz ardı edilmiştir. Her bir ağdan 1000 özellik çıkarılmıştır. Çıkarılan özellikler birleştirilerek 8000 özellik elde edilmiştir. Ardından ReliefF özellik seçicisi kullanılarak en anlamlı 1000 özellik elde edilmiştir. Seçilen 1000 özellik sınıflandırıcıların girişi olarak kullanılmıştır. Önerilen yöntemin adımları aşağıdaki gibidir.

Adım 1: Her bir ağı transfer modunda kullanarak 1000 özellik çıkar.

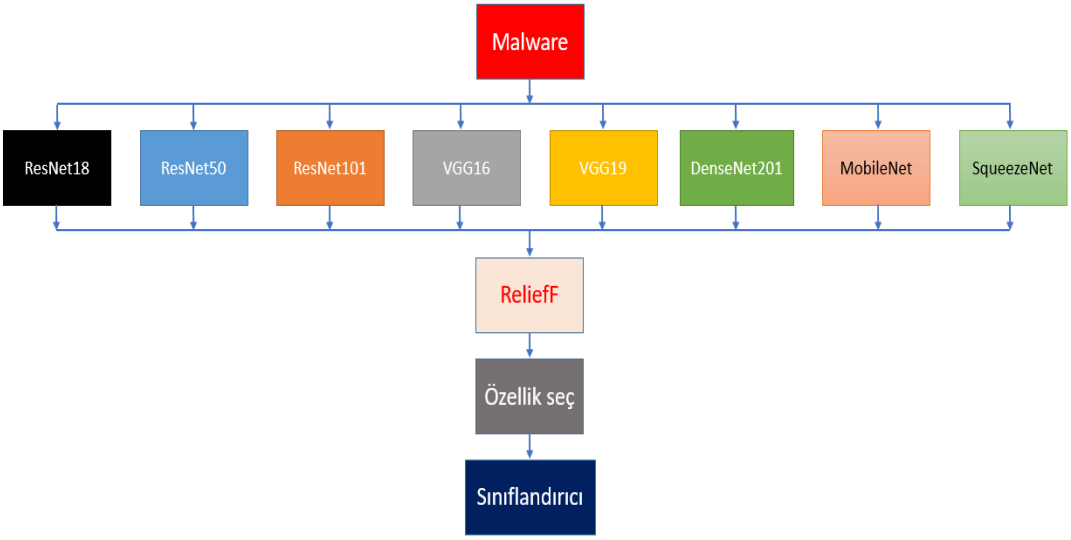
Adım 2: Özellikleri birleştir ve 8000 özellik boyutunda final vektörünü elde et.

Adım 3: Final vektörüne ReliefF uygula ve sıralanmış indisleri elde et.

Adım 4: Sıralanmış indisleri kullanarak en anlamlı 1000 özelliği seç.

Adım 5: Seçilen özellikleri sınıflandırıcıya gönder.

Yöntemin blok diyagramı da şekil 4.5'de gösterilmiştir.



Şekil 4.5. Kompozit derin ağlar tabanlı kötücül yazılım sınıflandırma yöntemine ait blok diyagram

Tablo 4.4'de kompozit derin ağlar tabanlı yönteme ait sonuçlar verilmiştir.

Tablo 4.4. Kompozit derin ağlar tabanlı yönteme ait test sonuçları

	Subspace Discriminant	**Quadratic SVM**	**Linear SVM**	**Cubic SVM**	**LDA**
Kompozit Derin Ağlar Tabanlı Yöntem	**93.6**	93.3	92.5	92.9	93.5

Bu test sonuçlarına göre bir kıyaslama yapıldığında en yüksek başarım oranının Subspace Discriminant sınıflandırıcısına ait olduğu tespit edilmiş olup %93.6 başarım oranıyla diğer yöntemlere göre daha yüksek sonuçlar elde edilmiştir.

5. BULGULAR VE TARTIŞMA

Bilinen kötü amaçlı yazılım örneklerinden çoğunun mevcut olanlardan türetildiği iyi bilinen bir gerçektir. Bu örneklerin gizleme teknikleriyle donanmış olması durumunda dahi ortak güvenlik çözümleriyle olası tehditlerden kolayca kaçınılabilir. Örneğin yapay zekâ teknikleri de siber güvenlik savunma tekniklerinden birisidir. Bu teknikler kullanılarak özellikle kötücül yazılım aktif olarak tespit edilebilmektedir. Literatürden de bilindiği üzere makine öğrenmesi teknikleri sinyal ve görüntü işleme teknikleriyle sıklıkla kullanılmaktadır ve bu alanda birçok akıllı model bulunmaktadır. Özellikle imge tanımlayıcılarının önerilmesiyle birlikte görüntü işleme ve bilgisayarlarla görü alanında büyük gelişmeler yaşanmıştır. İmge tanımlayıcıların verimliliğinden dolayı bu yöntemler bilgi güvenliğinde de kullanılmaya başlanılmıştır. Özellikle klasik makine öğrenmesi gibi yöntemler, örneklem tabanlı yöntemler ve derin ağ tabanlı yöntemler kullanılarak kötücül yazılım özellikleri çıkarılmıştır. Çıkarılan özellikler konvansiyel sınıflandırıcılar kullanılarak öğrenme modelleri oluşturulmuştur. Bu modellerin de oldukça başarılı sonuçlar ortaya koyduğu görülmüştür.

Günümüzde derin öğrenme yöntemleri de yaygın olarak kullanılmaktadır. Bu yöntemler kullanılarak bilgi güvenliği modelleri tasarlanmıştır. Özellikle CNN, LSTM ve GRU ağları iyi bir öğrenme modeli olarak karşımıza çıkmaktadır. Ancak bu derin öğrenme modellerinin yanı sıra en çok kullanılan yöntemler arasında derin ağ tabanlı GoogLeNet, ResNet, VGG, SqueezeNet, MobileNet, DenseNet gibi modeller de yer almaktadır.

Literatürde hem hafif sıklet hem de ağır sıklet yöntemler kötücül yazılım sınıflandırılması ve tespiti için kullanılmaktadır. Ağır sıklet yazılımlarının en önemli avantajı yüksek doğruluk performansına sahip olmaları ve büyük veriyi başarılı olarak yorumlayabilmeleridir. Dezavantajları ise, yüksek hesaplama maliyeti ve az veride düşük performans olarak görülmektedir.

Hafif sıklet yöntemler ise hızlı çalışırlar, düşük maliyetlidirler, seçilmiş az veri setleri için yüksek performans sağlayabilirler ancak yüksek performansı garanti etmezler.

Yöntemlerin genel avantaj ve dezavantajları ortaya konduğunda optimum yönteme ulaşmak için yeni bir yaklaşımın oluşturulması gerekliliği ortaya çıkmıştır. Bu gereklilik sonrası oluşturulan kompozit bir yöntem ile bu sorun çözüme kavuşturulmaya çalışılmış ve olumlu sonuçlar ortaya çıkarılmıştır.

6. SONUÇLAR

Bu tez çalışmasında bilgi güvenliği ve yapay zekânın ilişkisi irdelenmiştir. Günümüzde bilgi güvenliği sadece kriptoloji ve steganografi gibi konuları kapsayan bir bilim dalı değildir. Bilgi güvenliğinin sağlanması için, olaylara bir veri bilimcisi bakışıyla bakılması gerekmektedir. Özellikle kötücül yazılım tespitinde yapay zeka ve makine öğrenmesi yöntemlerinden faydalanılarak erken tespit sistemleri geliştirilebilmektedir. Bu tez çalışmasında hem klasik makine öğrenmesi modelleri hemde derin öğrenme tabanlı yöntemler kullanılarak kötücül yazılım sınıflandırma yöntemleri önerilmiştir. Deneysel sonuçları elde etmek için Maligm veri seti kullanılmıştır. Maligm veri seti 25 sınıfta 9339 kötücül yazılımdan oluşmaktadır. Bu veri setine uygulanmak üzere 3 adet yöntem kullanılmıştır. Bu yöntemlerden ikisi bizim önerdiğimiz yöntemlerdir. İlk olarak önerilen yöntem istatiksel özellik çıkarma yöntemlerini kullanan örneklem tabanlı kötücül yazılım sınıflandırma yöntemidir. Bu yöntemde değişken boyutlu örneklemler kullanılarak özellikler elde edilmiş ve elde edilen özelliklerden en anlamlı verileri elde etmek için RFNCA hibrit özellik seçicisi kullanılmıştır. Seçilen anlamlı özellikler konvansiyonel sınıflandırıcılar kullanılarak sınıflandırılmış ve %85.7 başarım oranı elde edilmiştir. İkinci yöntemde literatürde sıklıkla kullanılan 9 adet derin öğrenme ağı özellik çıkarıcı olarak kullanılmış ve elde edilen sonuçlar gösterilmiştir. Bu sonuçlar derin öğrenme yöntemlerinin daha iyi sonuçlar verdiğini göstermiştir. Bu avantajları kullanabilmek için 8 adet derin öğrenme ağı bir arada kullanılarak kompozit öğrenme ağı önerilmiştir. Önerilen ağda derin öğrenme ağları özellik üreteci olarak kullanılmış ve 8000 özellik elde edilmiştir. Bu özellikler ReliefF özellik seçiciyle 1000 özelliğe indirgenmiştir. Sonuçlar, bu yöntemin başarımının %93.6 başarıma ulaştığını açık bir şekilde göstermektedir.

ÖNERİLER

Literatürde hem hafif sıklet hem de ağır sıklet yöntemler(derin öğrenme yöntemleri) kötücül yazılım sınıflandırılması ve tespiti için kullanılmaktadır. Ağır sıklet yazılımlarının en önemli avantajı yüksek doğruluk performansına sahip olmaları ve büyük veriyi başarılı olarak yorumlayabilmeleridir. Dezavantajları ise, yüksek hesaplama maliyeti ve az veride düşük performans olarak görülmektedir. Hafif sıklet yöntemler ise hızlı çalışırlar, düşük maliyetlidirler, seçilmiş az veri setleri için yüksek performans sağlayabilirler ancak yüksek performansı garanti etmezler.

Çalışmalarımız ve literatür taraması doğrultusunda makine öğrenmesi ve derin öğrenme yöntemleri bir arada kullanılarak yüksek doğruluğu garanti eden bir sistem geliştirmek amaçlanmıştır. Amaçlarımız doğrultusunda elde ettiğimiz bu sonuçlar ile kötücül yazılım tespiti oldukça hızlı ve yüksek bir başarım oranıyla tespit edilecek olup bu alanda geliştirilecek başka ürünlere de örnek olması planlanmaktadır. Bu çalışmalarımızdan hareketle kötücül yazılım tespiti alanında herhangi bir ürün geliştirmek isteyen araştırmacılara ise yol göstereceği düşünülmektedir.

KAYNAKLAR

[1] Kaspersky Security Bulletin, https://securelist.com/kaspersky-security-bulletin-2019-statistics/95475/, Erişim: 30 Aralık 2019

[2] ZDNet Rapor, https://www.zdnet.com/article/cybercrime-drains-600-billion-a-year-from-the-global-economy-says-report/, Erişim: 20 Ekim 2019

[3] Idika, N., Mathur, A. (2007). A survey of malware detection techniques. Purdue University

[4] Tang, A. Sethumadhavan, S. and Salvatore J. S. (2014). Unsupervised anomaly-based malware detection using hardware features. CoRR, abs/1403.1631, 2014

[5] Demme, J., Maycock, M., Schmitz, J., Tang, A., Waksman, A., Sethumadhavan, S., Stolfo, S. (2013). On the feasibility of online malware detection with performance counters. Proceedings-International Symposium on Computer Architecture. 41. 559-570. 10.1145/2508148.2485970.

[6] Tian, R. Islam, R. Batten, L. Versteeg, S. (2010). Differentiating malware from cleanware using behavioural analysis. In: 2010 5th international conference on malicious and unwanted software. 2010. p. 23–30.

[7] Firdausi, I., Lim, C., Erwin, A., Nugroho, A.S. (2010). Analysis of machine learning techniques used in behavior-based malware detection 2010 second international conference on advances in computing, control, and telecommunication technologies , pp. 201-203

[8] Damodaran, A. Troia, F.D. Visaggio, C.A. Austin, T.H. Stamp, M. (2017). A comparison of static, dynamic, and hybrid analysis for malware detection J Comput Virol Hacking Tech, 13 (1), pp. 1-12

[9] Burnap, P., French, R., Turner, F., Jones, K. (2017). Malware classification using self organising feature maps and machine activity data. Computers & Security. 73. 10.1016/j.cose.2017.11.016

[10] Tobiyama, S., Yamaguchi, Y., Shimada, H., Ikuse, T., Yagi, T. (2016). Malware detection with deep neural network using process behavior. In: 2016 IEEE 40th annual Computer Software and Applications Conference (COMPSAC), vol. 2. 2016. p. 577–82

[11] Ahmed, F. Hameed, H. Shafiq, M.Z. Farooq, M. (2009). Using spatio-temporal information in API calls with machine learning algorithms for malware detection Proceedings of the 2nd ACM workshop on security and artificial intelligence, ACM, pp. 55-62

[12] Dini, G. Martinelli, F. Saracino, A. Sgandurra, D. (2012). MADAM: a Multi-level Anomaly Detector for Android Malware Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 240-253

[13] Scaife, N. Carter, H. Traynor, P. Butler, KRB. (2016). Cryptolock (and drop it): Stopping ransomware attacks on user data. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS). 2016. p. 303–12

[14] Continella, A., Guagnelli, A., Zingaro, G., Pasquale, G., Barenghi, A., Zanero, S., Maggi, F. (2016). Shieldfs: A self-healing, ransomware-aware file system Proceedings of the 32nd annual conference on computer security applications, ACM, New York, NY, USA, pp. 336-347 ACSAC '16

[15] Pascanu, R. Stokes, J.W. Sanossian, H. Marinescu, M. Thomas, A. (2015). Malware classification with recurrent networks, 2015 I. EEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2015. p. 1916–20

[16] Dahl, G.E., Stokes, J.W., Deng, L., Yu, D. (2013). Large-scale malware classification using random projections and neural networks 2013 IEEE international conference on acoustics, speech and signal processing, IEEE, pp. 3422-3426

[17] Huang, W., Stokes, J.W. (2016). Mtnet: A multi-task neural network for dynamic malware classification Proceedings of the 13th international conference on detection of intrusions and malware, and vulnerability assessment, vol. 9721, Springer-Verlag New York, Inc., New York, NY, USA, pp. 399-418 DIMVA 2016

[18] Bowles, S., Hernandez-Castro, J. (2015). The first 10 years of the Trojan Horse defence. Computer Fraud & Security. 2015. 10.1016/S1361-3723(15)70005-9.

[19] O'Hanley, R. (2005). Spyware, Adware, Malware — It's All Sleazeware to Me. Information Systems Security. 13. 2-4. 10.1201/1086/44954.13.6.20050101/86214.1.

[20] Güvenlik Çözümleri, https://www.trendmicro.de/cloud-content/us/pdfs/security-intelligence/white-papers/wp-backdoor-use-in-targeted-attacks.pdf, Erişim: 26 Nisan 2019

[21] Megira, S., R Pangesti, A., Wibowo, F. (2018). Malware Analysis and Detection Using Reverse Engineering Technique. Journal of Physics: Conference Series. 1140. 012042. 10.1088/1742-6596/1140/1/012042.

[22] Garg, N., Kumar, D., Khera, Y., Sujay, P., Jain, P. (2018). Towards the ımpact of hackıng on cyber securıty. IIOAB Journal. 9. 61-77.

[23] Robin, M. (2018). Cyber Security.

[24] Güvenlik, https://www.trendmicro.com/vinfo/us/security/definition/rootkit, Erişim: 28 Nisan 2019

[25] Tehdit Raporu, https://www.symantec.com/security-center/threat-report, Erişim: 28 Nisan 2019.

[26] Aggarwal, S., Houshmand, S., Weir, M. (2018). New Technologies in Password Cracking Techniques. 10.1007/978-3-319-75307-2_11.

[27] Sinha, S., (2017). Building an Nmap Network Scanner. 10.1007/978-1-4842-2541-7_24.

[28] Pandey, B., Singh, A., Balani, L. (2015). Ethıcal hackıng (Tools, Techniques and Approaches). 10.13140/2.1.4542.2884.

[29] Ambavkar, P., Patil, P., Meshram, P., Pamu, K., Swamy. (2012). WPA Exploitation In The World Of Wireless Network. International Journal of Advanced Research in Computer Engineering & TechnologyJune 28, 2012. Volume 2. 320-324.

[30] Saeed, F. (2014). Assessment of open source web application security scanners. Journal of Theoretical and Applied Information Technology. 61. 281-287.

[31] Al-khateeb, S., Agarwal, N. (2019). Social cyber forensics: leveraging open source information and social network analysis to advance cyber security informatics. Computational and Mathematical Organization Theory. 10.1007/s10588-019-09296-3.

[32] Tranquillo, J., Kline, W., Hixson, C. (2016). Making Sense of Canvas Tools: Analysis and Comparison of Popular Canvases.

[33] Raju, B. (2016). MITM Attacks through ARP poisoning. 10.13140/RG.2.2.30881.61289.

[34] Obunadike, G., Isah, A., Alhassan, J. (2018). Optimized Naïve Bayesian Algorithm for Efficient Performance.

[35] Sharma, H., Kumar, S. (2016). A Survey on Decision Tree Algorithms of Classification in Data Mining. International Journal of Science and Research (IJSR). 5.

[36] Sharma, P., Aggarwal, A., Gupta, A., Garg, A. (2019). Leaf Identification Using HOG, KNN, and Neural Networks: Proceedings of ICICC 2018, Volume 2. 10.1007/978-981-13-2354-6_10.

[37] Cutler, A., Cutler, D., Stevens, J. (2011). Random Forests. 10.1007/978-1-4419-9326-7_5.

[38] Random Forest Algoritması, https://devhunteryz.wordpress.com/2018/09/20/rastgele-ormanrandom-forest-algoritmasi/, Erişim Tarihi: 30 Nisan 2019.

[39] Bashari Rad, B. Shahpasand, M. Kazem Hassan Nejad, M. (2018). Malware classification and detection using artificial neural network. Journal of Engineering Science and Technology. 13. 14-23.

[40] Hoon Kim, C., Espoir, K., & Kang, S. J. (2018). Classifying malware using convolutional gated neural network. 40-44. 10.23919/ICACT.2018.8323640.

[41] Silva, C., Bouwmans, T., Frelicot, C. (2015). "An eXtended Center-Symmetric Local Binary Pattern for Background Modeling and Subtraction in Videos", VISAPP 2015, Berlin, Germany, March 2015

[42] Bouwmans, T., Silva, C., Marghes, C., Zitouni, M. S., Bhaskar, H., Frelicot, C. (2018). On the Role and the Importance of Features for Background Modeling and Foreground Detection. Computer Science Review. 28. 26–91. 10.1016/j.cosrev.2018.01.004.

[43] Sharma, P., Aggarwal, A., Gupta, A., Garg, A. (2019). Leaf Identification Using HOG, KNN, and Neural Networks: Proceedings of ICICC 2018, Volume 2. 10.1007/978-981-13-2354-6_10.

[44] Gandotra, E., Bansal, D., Sofat, S. (2014). Malware Analysis and Classification: A Survey. Journal of Information Security. 05. 56-64. 10.4236/jis.2014.52006.

[45] Mohebbi, A., Douzandegan, Y. (2017). Linear Regression Analysis of Title Word Count and Article Time Cited using R. Journal of Scientometric Research. 6. 15-22. 10.5530/jscires.6.1.3.

[46] Singhal, Y., Jain, A. Batra, S. Varshney, Y. Rathi, M. (2018). Review of Bagging and Boosting Classification Performance on Unbalanced Binary Classification. 10.1109/IADCC.2018.8692138.

[47] Sultana, F. Sufian, A. Dutta, P. (2018). Image Classification using CNN.

[48] Krizhevsky, A., Sutskever, I. Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[49] Ballester, P., Araujo, R. (2016). On the performance of GoogLeNet and AlexNet applied to sketches.

[50] Tammina, S. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. International Journal of Scientific and Research Publications (IJSRP). 9. p9420. 10.29322/IJSRP.9.10.2019.p9420.

[51] Zeiler, M. Fergus, R. (2013). Visualizing and Understanding Convolutional Neural Networks. ECCV 2014, Part I, LNCS 8689. 8689. 10.1007/978-3-319-10590-1_53.

[52] VGG-19 Sinir Ağları Nedir?, https://www.quora.com/What-is-the-VGG-19-neural-network, Erişim: 7 Nisan 2020

[53] He, K. Zhang, X. Ren, S. Sun, J. (2016). Deep Residual Learning for Image Recognition. 770-778. 10.1109/CVPR.2016.90.

[54] Huang, G., Liu, Z., van der Maaten, L., Weinberger, K. (2017). Densely Connected Convolutional Networks. 10.1109/CVPR.2017.243.

[55] Iandola, F. N. "SqueezeNet." https://github.com/forresti/SqueezeNet, Erişim: 28 Nisan 2020.

[56] Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W. Weyand, T., Andreetto, M., Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

[57] Lakshmanan, N., Karthikeyan, S., Gregoire J., and Bangalore S. M. (2011). Malware images: visualization and automatic classification. In Proceedings of the 8th international symposium on visualization for cyber security. ACM, 4.

[58] Yıldız, A.M., Tuncer, T., Doğan, Ş., Akbal, E. (2019). Örneklem ve Makine Öğrenmesi Tabanlı Kötücül Yazılım Tespiti, 4. Uluslararası Mühendislik ve Doğa Bilimleri Konferansı, Dicle Üniversitesi, Diyarbakır.

[59] Krizhevsky, A., Sutskever, I., Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

ÖZGEÇMİŞ

Arif Metehan YILDIZ

KİŞİSEL BİLGİLER

Doğum Yeri	**:** Maden
Doğum Yılı	**:** 1995
Uyruğu	**:** T.C.
Adres	**:** Rüstem Paşa Mahallesi, Çaldıran Sokak, No:42, Daire : 3, Merkez/Elazığ
E-posta	**:** a.metehanyildiz@gmail.com
Yabancı Diller	**:** İngilizce (Düzey: B2)

EĞİTİM BİLGİLERİ

Lisans	**:** Fırat Üniversitesi, Teknoloji Fakültesi, Adli Bilişim Mühendisliği Bölümü, 2018
Lise	**:** Mehmet Koloğlu Anadolu Lisesi, Elazığ, 2013

ARAŞTIRMA DENEYİMİ

- ✓ Pc-3000 Express, Tableau TD3, Ultra Dock
- ✓ C++, Java, HTML, MATLAB
- ✓ Forensics Explorer, EnCase, Magnet Axıom, FTK, X-Ways, Autopsy, MobilEdit Forensics

İŞ DENEYİMİ

2016 – 2016	**:** FORDEFENCE Adli Bilişim Laboratuvarı
2017 – 2017	**:** TRT WORLD IT
2018 – 2018	**:** MyDISK Adli Bilişim Ve Veri Kurtarma

AKADEMİK FAALİYETLER

Makaleler:

1. MALWARE DETECTION METHOD BASED ON EXEMPLAR AND MACHINE LEARNING