T.C.
İSTANBUL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

LOG TABANLI SALDIRI TESPİT SİSTEMLERİNDE PERFORMANS ANALİZİ

Doğukan AKSU

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

DANIŞMAN
Dr. Öğr. Üyesi Muhammed Ali AYDIN

Haziran, 2018

İSTANBUL

Uygundur (05.07.2018)
Dr. Öğr. Üyesi Muhammed Ali AYDIN

Bu çalışma 06.06.2018 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Programında Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Jürisi

Dr. Öğr. Üyesi Muhammed Ali AYDIN (Danışman)
İstanbul Üniversitesi
Mühendislik Fakültesi

Prof. Dr. Ahmet SERTBAŞ
İstanbul Üniversitesi
Mühendislik Fakültesi

Doç. Dr. Yasin ÖZÇELEP
İstanbul Üniversitesi
Mühendislik Fakültesi

Dr. Öğr. Üyesi Şerif BAHTİYAR
İstanbul Teknik Üniversitesi
Bilgisayar ve Bilişim Fakültesi

Dr. Öğr. Üyesi Şafak Durukan ODABAŞI
İstanbul Üniversitesi
Mühendislik Fakültesi

20.04.2016 tarihli resmi gazetede yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, İstanbul Üniversitesi′nin abonesi olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü′nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır.

ÖNSÖZ

Bir hedefe ulaşmanın yolu, azim, çok çalışma ve disiplinden geçmektedir. Teze başladığım tarihten bu yana, motivasyon ve enerji kaynağı olan insanlar ve onların sağladığı çok değerli yardımlar olmasa idi bu görevimi başarı ile tamamlayamazdım.

Öncelikle "Boynuz kulağı geçmeli" deyimini eğitim ve öğretim hayatında ilke edinmiş ve bu ilke doğrultusunda öğrencilerinin yetişmesi için elinden gelen hiçbir fedakarlığı esirgemeyen, aynı zamanda tezimin şekillenmesinde büyük emeği geçen, gerek bilgi ve birikimlerini paylaşarak gerekse de gösterdiği yakınlık ve desteklerinden dolayı danışman hocam Dr. Öğr. Üyesi Muhammed Ali AYDIN 'a çok teşekkür ederim.

İlkokuldan başlayarak bugünlere gelmemde emeği geçen, özellikle ilk öğretmenim olan ve 5 yıl boyunca derslerime giren sevgili öğretmenim Fikret Yıldırm 'a, 6. sınıfta her ne kadar ben istemesemde beni sınıf başkanı yapan, her hafta kompozisyon ödevleri vererek bizlere yazma alışkanlığı kazandıran sınıf ve Türkçe öğretmenim İhsan Mercimekçi 'ye ve gerek derslerime giren gerekse çok değerli görüş ve yönlendirmeleri ile mesleki kariyerim ve hayata bakış açımın şekillenmesinde katkıda bulununan, sorgulamayı, araştırmayı ve değer üretmeyi öğreten tüm öğretmen ve hocalarıma teşekkür ederim.

Her zaman daha iyi ve başarılı olmamı dileyen ve bu uğurda benden hiçbir maddi ve manevi desteğini esirgemeyen aileme, en içten sevgi ve saygılarımla en büyük teşekkürlerimi ediyor ve şükranlarımı sunuyorum.

Haziran, 2018 Doğukan AKSU

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

TABLO LİSTESİ

Sayfa No

SİMGE VE KISALTMA LİSTESİ

Simgeler **Açıklama**

γ	: Pozitif Bir Normalleştirme Parametresi
K	: K Mesafe
S_b	: Sınıf Arası Dağılım Matrisi
S_t	: Toplam Dağılım Matrisi

Kısaltmalar **Açıklama**

DVM	: Destek Vektör Makineleri
DPI	: Deep Packet Inspection
FN	: False Negative
FP	: False Positive
ID	: Tanımlama Numarası
KA	: Karar Ağacı
KNN	: K Nearest Neighbours
SÖS	: Saldırı Önleme Sistemleri
STS	: Saldırı Tespit Sistemleri
TN	: True Negative
TP	: True Positive
YSA	: Yapay Sinir Ağları

ÖZET

YÜKSEK LİSANS TEZİ

LOG TABANLI SALDIRI TESPİT SİSTEMLERİNDE PERFORMANS ANALİZİ

Doğukan AKSU

İstanbul Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Muhammed Ali AYDIN

Hızla gelişmekte olan teknoloji, birçok yeni atak türünü beraberinde getirmenin yanı sıra, mevcut atakların yeni özellikler kazanarak farklılaşmasını sağlamıştır. Bu durum azımsanamayacak çokluktaki kurum, kuruluş ve şirketlerin maddi ve manevi kayıplara uğramasına sebebiyet vermiştir. Dolayısı ile verilerin büyük bir hızla dijitalleştiği günümüz dünyasında, veri güvenliğinin sağlanması, verilerin doğru bir şekilde iletilmesi ve veri bütünlüğünün bozulmadan korunması büyük bir öneme sahip olmuştur. Verilerin bütünlüğünü, gizliliğini, erişilebilirliğini ve güvenliğini tehdit eden tüm davranışlar saldırı olarak adlandırılmaktadır. Bu saldırılara karşı çeşitli güvenlik mekanizmaları geliştirilmiştir. En yaygın olarak kullanılan güvenlik mekanizmalarından biri ise "Saldırı Tespit Sistemleri" 'dir. Saldırı tespit sistemlerinin ana amacı sistemi tehdit eden bir davranış olduğunda bu saldırıyı tespit etmektir. Bu nedenle saldırı tespit sistemlerini alarm gibi düşünebiliriz. Olayların kayıt altına alınması ile oluşan veriler "log" olarak isimlendirilmektedir. Olayların geçtiği ortama bağlı olarak loglar; yazılım logları, işletim sistemi logları, ağ sunucusu logları vb. birçok çeşide ayrılmaktadır. Bu tez kapsamında ağ üzerindeki olayların kayıt altına alındığı loglar üzerinde saldırı tespit sistemleri incelemiş ve performans analizi çalışmaları yapılmıştır.

Haziran 2018, 92 sayfa.

Anahtar kelimeler: STS, SÖS, makine öğrenmesi, CICIDS2017 veriseti.

SUMMARY

M.Sc. THESIS

PERFORMANCE ANALYSIS OF LOG-BASED INTRUSION DETECTION SYSTEMS

Doğukan AKSU

İstanbul University

Institute of Graduate Studies in Science and Engineering

Department of Computer Engineering

Supervisor: Dr. Öğr. Üyesi Muhammed Ali AYDIN

Rapidly developing technology not only created many new attack types but also it varied existing attacks with new features. This situation has caused many institutions, organizations and companies to suffer financial and moral loss. Therefore, in today's world that all data are digitized at a high speed, providing data security, transmitting data correctly and protecting the integrity of data has become very important. All behaviors that threaten the integrity, confidentiality, accessibility and security of the data are called attacks. Various security mechanisms have been developed against these attacks. One of the most widely used security mechanisms is "Intrusion Detection Systems". The main purpose of intrusion detection systems is to detect a threatening behavior in the system. For this reason, we can think of intrusion detection systems as alarms. The recording of events is called "log". There are various log types depending on the events, such as software logs, operating system logs, network server logs etc. In this thesis, intrusion detection systems have been analyzed and performance analysis has been done on the network logs.

Haziran 2018, 92 pages.

Keywords: IDS, IPS, machine learning, CICIDS2017 dataset.

1. GİRİŞ

Gelişen teknoloji, sayısız imkan ve fırsatın yanında maalesef birçok riski de beraberinde getirmiştir. Önceden hiç görülmediği kadar internet kullanımı yaygınlaşmış ve internet kullanımının yaygınlaşmasının yanı sıra nesnelerin interneti teknolojisi popülerleşerek gelişmiştir. Bu durum, mevcut veri miktarını, lineer bir şekilde değil, eksponansiyel olarak arttırmış, dolayısı ile verilerin saklanması ayrı bir problem olmuş ve bulut mimarisi teknolojisinin gelişmesine olanak sağlamıştır. Dolayısı ile verilerin gizliliği, bütünlüğü ve erişilebilirliğinin korunmasına olanak sağlayan ağ güvenliği konusu büyük bir önem kazanmış ve kritik bir pozisyon almıştır.

Verilerin gizliliğini, bütünlüğünü veya erişilebilirliğini tehdit eden her davranış atak veya saldırı olarak adlandırılır. SYN saldırısı (SYN Flooding), DoS(Denial of Service) / DDoS(Distributed Denial of Service) atakları, Paket Süzme (Packet Sniffing), Zehirleme (Spoofing), Virüsler, Ajan yazılımlar(Spyware) gibi ağ üzerinde çeşitli atak türleri vardır.

İzinsiz giriş (intrusion), bir bilgi sistemine ait güvenlik politikasını veya yasal korumaları ihlal etme eylemidir. Sistemi kötüye kullanan veya sisteme izinsiz giriş yapmaya teşebbüs eden ya da izinsiz giriş yapan kişiler saldırgan(intruder) olarak isimlendirilmektedir. Bu saldırganlar hacker veya lamer olabilirler. Hacker ve lamer arasındaki temel fark ise hackerlar amaçları için kendi kodlarını yazarlarken, lamerler var olan kodları kullanarak sisteme zarar vermeyi amaçlarlar.

Ağ güvenliğini sağlamak amacıyla çeşitli araç ve teknikler kullanılmaktadır. Bunlar; güvenlik duvarları (firewalls), Saldırı Tespit Sistemleri (STS) ve Saldırı Önleme Sistemleridir (SÖS). Çoğu zaman saldırı tespit ve saldırı önleme sistemleri birlikte Saldırı Tespit ve Önleme Sistemleri(STÖS) adı altında kullanılabilirler.

Güvenlik duvarı; ağı dış saldırılardan koruyan yazılım veya donanımların birleşimidir. Güvenlik duvarı, özel bilgilerin korunmasında ilk savunma hattı olarak kabul edilmektedir.

Saldırı Tespit Sistemi, yazılım veya donanım veya her ikisinden oluşan bilgisayarlar ve

ağlar için bir tür güvenlik yönetim sistemidir. STS; bir ağda veya bilgisayar sisteminde gerçekleşen saldırıların çeşitli yöntemler kullanarak tespit edilmesini ve yetkili kişilerin uyarılmasını sağlar. Saldırı Tespit ve Önleme Sistemleri ise herhangi bir saldırıyı tespit ettikten sonra ilgili ağdaki bağlantıyı kesip trafiği engelleyebilirler.

Tezin kalan kısımları için; Genel Kısımlarda detaylı literatür taraması Ağ Üzerinde Gerçekleşen Atak Türleri, Saldırı (Intrusion) Nedir?, Güvenlik Araçları ve Yöntemleri alt başlıkları ile, Malzeme ve Yöntemler kısmında Kullanılan Veri Seti ve Makine Öğrenmesi yöntemleri, Bulgular başlığı altında elde edilen bulgular paylaşılmış ve Tartışma kısmında bu bulgular tartışılmıştır. Sonuç kısmı ile de tez sonlandırılmıştır.

2. GENEL KISIMLAR

2.1. LİTERATÜR ÖZETİ

Literatürdeki çalışmalar incelendiğinde öncelikle saldırı tespit sistemlerinin çeşitli tanımlamalarının olduğu görülmektedir. Bu tanımlamalara göre saldırı tespit sistemleri; bir ağdaki veya bilgisayar sistemindeki verilerin erişilebilirliğini, gizliliğini ve bütünlüğünü bozmak için sistemin güvenliğini tehdit eden davranışları analiz eden bir süreçtir [1], bilgisayar sistemindeki saldırıları tespit etmek için tasarlanmışlardır [2], olası güvenlik ihlali durumunda, ihlali engellemekten ziyade güvenlik uzmanına uyarı mesajı (alarm) verirler [3], bilgisayar sistemindeki dosyalara veya kaynaklara yetkisiz erişimi tespit ederler [4] veya bir ağa ya da bilgisayar sistemine yetkisiz erişimi algılamak için kullanılan yazılım araçlarıdır [5] gibi çeşitli tanımlamalara sahiptirler. Çeşitli tanımlamaların yanı sıra, literatürde birçok makine öğrenmesi algoritmaları kullanılarak gerçekleştirilmiş Saldırı Tespit Sistemleri mevcuttur.

Hossain Shahriar ve William Bond, "Towards An Attack Signature Generation Framework for Intrusion Detection Systems" başlıklı yapmış oldukları çalışmalarında, imza tabanlı Saldırı Tespit Sistemleri için mevcut atakların imzasından genetik algoritmalar kullanarak yeni atak imzaları üretmişlerdir [6]. Bu çalışmada anomali tabanlı saldırı tespiti bulunmayıp özellikle imza tabanlı STS 'ler üzerine geliştirildiğinden bilinen ataklar için yüksek doğruluk oranı vereceği ama yeni atak türleri için benzer bir başarı gösteremeyeceği düşünülmektedir. İmza tabanlı ve anomali tabanlı STS 'lere detaylı olarak karşılaştırmalı bir şekilde Saldırı Tespit Sistemleri başlığı altında yer verilmiştir.

Anuja S. Desai ve D. P. Gaikwad tarafından yapılmış "Real Time Hybrid Intrusion Detection System using Signature Matching Algorithm and Fuzzy-GA" başlıklı başka bir çalışmada ise; hem içeriden saldırıları hem de dış kaynaklı saldırıların tespiti için hibrit bir STS geliştirilmiştir. Bu çalışma, içerden yapılan saldırıların tespiti için imza eşleştirmesi yapıyorken, dış kaynaklı saldırı tespitini bulanık genetik algoritmalar ile gerçekleştirmektedir. Sunulan sistem çevrimiçi olarak çalışabildiği gibi çevrim dışı olarak da çalışabilmektedir. Çalışmada kullanılan veri seti konusunda yeterli bilgi

bulunmamaktadır ve Sonuç Tablosu toplam 50 kayıttan oluşmaktadır. Bunlara ek olarak çalışmanın başarı oranı verilmeyip bazı sistemlerden başarılı olduğu iddia edilmiştir [7].

Omar Al-Jarrah ve Ahmad Arafat, "Network Intrusion Detection System Using Attack Behavior Classification" isimli çalışmalarında, ağ ataklarının tanımlanma oranını maksimum yapmak için akıllı bir sistem kullanmışlardır. Bu sistem beş modülden oluşmaktadır; paket yakalama motoru, önişlemci, örüntü tanıma, sınıflandırma ve izleme ve alarm modülleridir. Sistem Darpa 1998 veri seti üzerinde test edilmiş ve %100 tanıma oranı elde edilmiştir. Ayrıca çalışmanın test sonuçları, sistemin SNORT gibi kural tabanlı sistemlerden çok daha hızlı bir şekilde saldırıları tespit ettiğini göstermiştir [8].

Jabez J ve B. Muthukumar 'ın "Intrusion Detection System (IDS): Anomaly Detection using Outlier Detection Approach" isimli çalışmasında, saldırı tespit etmek için Outlier Detection yaklaşımı olarak adlandırılan yeni bir yaklaşımın detaylarını paylaşmışlardır. Eğitim modeli, saldırı tespit sisteminin performansını arttıran dağıtılmış büyük veri kümelerinden oluşmaktadır. Önerilen yaklaşım, KDD veri seti kullanılarak test edilmiştir. Önerilen STS sisteminin literatürdeki yöntemlere nazaran daha az zamana ve alana ihtiyaç duyduğu savunulmuş, hatta ağdaki tüm anomali verisinin daha iyi bir performansla tespit edilebileceği iddia edilmiştir [9].

Neha G. Relan ve Dharmaraj R. Patil ise konu ile ilgili olarak yapmış oldukları çalışmalarında, C4.5 karar ağacı yöntemi ve bu yöntemi Pruning ile birlikte, NSL_KDD veri seti kullanarak test etmişlerdir. Sonuç olarak, C4.5 karar ağacının Pruning ile birlikte kullandıkları senaryoda %98,45 değerinde daha yüksek bir doğruluk oranı elde etmişlerdir [10].

M. H. Aghdam ve P. Kabiri 'nin "Feature Selection for Intrusion Detection System Using Ant Colony Optimization" başlıklı yayınlanmış makalelerinde, STS performansını arttırmak için, özniteliklerin karınca kolonisi optimizasyonu kullanılarak en uygun şekilde seçildiği bir STS önermişlerdir. Önerilen sistemde veri seti olarak KDD Cup 99 veri seti kullanılmış ve %98.9 bir doğruluk oranı elde edilmiştir [11].

U. Ravale ve arkadaşları, K-Means kümeleme algoritması ve SVM sınıflandırıcısının birlikte kullanılması ile hibrit bir STS yaklaşımı önermişlerdir. Bu yaklaşımda SVM 'nin RBF kernel fonksiyonu sınıflandırmada ve geniş heterojen veri setinin daha küçük

alt kümelere indirgenmesi için K-Means kümeleme yöntemi beraber kullanılmıştır. Bu yöntemde, KDD99 veri seti üzerinde gerçekleştirilen testler, özniteliklerin sayısının azaltılması ile karmaşıklık azalmış, performans ve doğruluk oranı ise artmıştır [12]. G.V. Naidammai ve M. Hemalatha çalışmalarında, veri madenciliği konseptini STS ile birleştirmişlerdir. Bu hibrit modelde EDADT, Yarı-Gözetimli Yaklaşım ve Değişen HOPERAA algortimalarını ve KDD Cup veri setini kullanmışlardır. Önerilen EDAADT ile %98.12 'lik bir doğruluk oranı elde edilmiştir [13].

Wei-Chao Lin ve arkadaşlarının yapmış olduğu başka bir çalışmada ise, CANN olarak adlandırılan küme merkezi ve en yakın komşu yaklaşımı önerilmiştir. Bu yaklaşımda, iki uzaklık ölçülür ve toplanır, ilki her bir veri örneği ve küme merkezi arasındaki mesafeye dayanıyorken, ikinci mesafe, aynı kümedeki veri ve onun en yakın komşusu arasındaki mesafeyi ifade eder. Daha sonra, bu yeni ve tek boyutlu mesafe tabanlı öznitelik, KNN sınıflandırıcı ile saldırı tespiti yapmak için her bir veri örneğini temsil etmek için kullanılır. Deneysel sonuçlar KDD-Cup 99 veri setine dayandırılmıştır ve CANN sınıflandırıcısının, KNN ve SVM ile kıyaslandığında benzer hatta daha iyi sonuçlar göstermesinin yanı sıra sınıflandırıcının eğitim ve test süresi içinde yüksek hesaplama verimliliği sağladığı öne sürülmüştür [14].

Raman Singha ve arkadaşları, ağ trafiği profili ile OS-ELM tabanlı bir STS sunmuşlardır. OS-ELM, hızlı ve doğru tek bir gizli katman ileri besleme sinir ağıdır. Çalışmada, eğitim bağlantıları ilk olarak protokol ve servis özellikleri temelinde sınıflandırılmıştır. Bu sınıflandırma, alfa profil oluşturma olarak adlandırılmış vc STS nin ölçeklenebilirliğini artırmanın yanında zaman karmaşıklığını azaltmıştır. Eğitim veri kümesinin boyutunu azaltmak için ise beta profil oluşturma kullanılmıştır. NSL-KDD 2009 veri seti kullanılmıştır [15].

L.Dhanabal ve S.P. Shantharajah 'ın sunmuş olduğu çalışmada, NSL-KDD veri seti analiz edilmiş ve ağ trafiği paternlerindeki anormallikleri saptamada çeşitli sınıflandırma algoritmalarının etkinliğini incelemek için kullanılmıştır. Yaygın olarak kullanılan ağ protokol yığınında bulunan protokollerin, anormal bir ağ trafiği oluşturmak için saldırganların kullandığı saldırılarla ilişkisi de analiz edilmiştir. Analiz, veri madenciliği aracı WEKA'da bulunan sınıflandırma algoritmaları kullanılarak yapılmıştır [16].

Aastha Puri ve Nidhi Sharma 'nın "a novel technique for intrusion detection system for

network security using hybrid svm-cart" başlığı ile yayınlamış oldukları makalede, SVM, sınıflandırma ve regresyon ağacı algoritmalarının kombinasyonundan oluşan hibrit bir yaklaşımı atakların sınıflandırılması için kullanmışlardır. Bu hibrit yaklaşım KDDCUP 99 veri seti üzerinde test edilmiş ve SVM-CART yöneminin KNN methoduna göre daha yüksek doğruluk ile çalıştığı yapılan testler sonucu ortaya konulmuştur [17].

Dubey ve arkadaşlarının önerdiği başka bir çalışmada ise, Saldırı Tespit Sistemleri ve STS 'lerin tasarım konseptlerinin araştırması yapılmıştır. Bu amaçla, KDD CUP 99 veri seti analizi kullanılarak bir STS geliştirmişlerdir. Çalışmanın ana odak noktası, sınıflandırıcıların performans iyileştirmeleri üzerinedlir. Önerilen STS; K-Means kümeleme algoritmasını, Bayes sınıflandırma algoritmasını ve son olarak geri yayılımlı sinir ağını(back propagation neural network) kullanmaktadır. Sistem, MATLAB IDE kullanılarak gerçeklenmiştir. Performans sonuçlarının doğrulanması için önerilen sınıflandırma yöntemi, geleneksel Bayes sınıflandırıcı ile karşılaştırılmıştır. Elde edilen performansa göre, önerilen sınıflandırma tekniği, geleneksel yöntemden optimum sınıflandırma doğruluğu ve hata oranı iyileştirmesi sağlamış, ancak tahmin süresi açısından benzer performans sergilemişlerdir. Öte yandan, geleneksel yöntemin hafıza tüketimi önerilen yöntemden daha fazla olmuştur [18].

Tseng ve arkadaşları, AODV yönlendirme protokolü üzerindeki saldırıları tespit edebilen bir Specification-Based saldırı tespit sistemi geliştirmişlerdir. Specification-Based saldırı tespit yaklaşımında; kritik nesnelerin doğru davranışları manuel olarak çıkarılmış, güvenlik kuralları olarak belirlenmiş ve bu kurallar, nesnelerin gerçek davranışları ile karşılaştırılmıştır. Genellikle nesnelerin beklenenin dışında, olağan dışı, beklenmeyen bir şekilde davranmasına neden olan saldırılar hakkında tam ve detaylı bir bilgi olmadan da tespit edilebilmişlerdir. Dolayısı ile bilinmeyen saldırılara da hitap edebileceği düşünülebilir. Bu makalede sunulan STS, AODV istek-cevap akışlarını izleyen dağıtık bir ağ monitörü mimarisi üzerine kurulmuştur. Ağ, tüm istek-cevap oturum ağaçlarını ve karşılık gelen yönlendirme tablolarını oluşturmak ve güncellemek için her RREQ, RREP ve RERR'yi izler ve denetler. İstek-cevap akışındaki kısıtlamalar sonlu durum makineleri kullanılarak belirlenir. Oturum ağaçlarını inşa etmek ve işlemek için prosedürleri tanımlar ve saldırıların başarılı bir şekilde tespit edilmesine örnekler sunar. Bu araştırma, Ad Hoc ağlarda yönlendirme saldırılarını tespit etmek için Specification-Based tespit tekniklerini uygulayan ilk çalışmalardan birisidir [19].

Peddabachigari ve Sandhya 'nın çalışmasında ise, saldırı tespiti için bazı yeni teknikler araştırılmış ve karşılaştırmaları KDD CUP 99 veri setine dayanılarak değerlendirilmiştir. Çalışmada öncelikle, DT ve SVM algoritmaları saldırı tespit modeller olarak sunulmuş ve akabinde SVM ve DT 'nin beraber kullanılması ile hibrit bir DT-SVM Saldırı Tespit modeli geliştirilmiştir. Bu modellerin birbirleri ile karşılaştırmalı sonuçlarına çalışmada yer verilmiştir. Deneysel sonuçlar, DT 'nin Normal, Probe, U2R ve R2L sınıfları için benzer veya daha iyi sonuçlar verdiğini ortaya çıkarmıştır. Hibrit DT-SVM yaklaşımı doğrudan SVM ile karşılaştırıldığında bütün sınıflar için eşit performans sergilemiştir. Bu yöntemlerin toplu kullanımı ise Probe ve R2L sınıfları için en iyi performans vermiştir. Bu çalışma, tekniklerin uygun bir şekilde bir araya getirilerek kullanımı ile daha yüksek başarım oranlarının elde edilebileceğini göstermiştir [20]].

Farah Jemili ve arkadaşları, Bayes sınıflandırıcı kullanarak kural tabanlı bir STS geliştirmişlerdir. Bu sistemde DARPA 99 veri seti kullanılmış ve sırasıyla DOS, PROBE, U2R, R2L saldırılarının sınıflandırılmasında; %99.62, %100, %98,63 ve %42.62 doğruluk oranları elde etmişlerdir [21].

Bu çalışmalara ek olarak, KDD CUP99 veri seti kullanılarak yapılan literatürdeki çalışmaların başarı oranlarının detaylı gösterimi Tablo 2.1 'de verilmiştir.

Tablo 2.1: KDD CUP99 ile yapılan çalışmalar ve elde edilen test sonuçlarındaki başarı oranlarının yüzdelik olarak gösterimi.

Yazarlar	Normal	DoS	Probe	U2R	R2L
Dewan ve arkadaşları [22]	99,82	99,49	99,72	99,47	99,35
Yongli Zhang ve Y. Zhu [23]	99,32	93,81	33,67	39,31	99,42
Jun Wang ve arkadaşları [24]	100	99,92	100	76	87,92
Qi Mu ve arkadaşları [25]	-	100	100	100	99,11
M. Bahrololum ve arkadaşları [26]	99,96	99,97	99,66	88,33	99,02
Ammar Alazab ve arkadaşları [27]	98,2	97,2	99,6	92,5	99,7
Vikas Sharma ve Aditi Nema [28]	-	99,98	88,19	51	94,70
Guisong Liu ve arkadaşları [29]	97,1	100	100	97,2	-
Xingchao Gong ve Xin Guan [30]	100	100	99,76	99,85	99,88
Wei-Chao Lin ve arkadaşları [14]	99,68	99,98	98,49	17,31	91,74
Chih-Fong Tsai ve Chia-Yİn Lin [31]	71,31	95,87	93,43	40	85,84

Tablo 2.1 'de görüldüğü üzere Normal, DoS, Probe, U2R ve R2L sınıflandırma oranları, yapılan çalışmalar ve kullanılan yöntemler ile sırasıyla aşağıda belirtildiği gibidir: Dewan ve arkadaşları, Bayes temelli, "Improved Self Adaptive Bayesian Algorithm-ISABA" ismini verdikleri bir yöntem geliştirmişler ve %99.82, %99.49, %99.72, %99.47 ve %99.35 doğruluk oranları elde etmişlerdir. Yongli Zhang ve Y. Zhu, DVM temelli bir yaklaşımla, %99.32, %93.81, %33.67, %39.31 ve %99.42 başarı oranları; Jun Wang ve arkadaşları, arı kolonisi ve DVM ile %100, %99.92, %100, %76 ve %87,92; Qi Mu ve arkadaşları, Boundary Incremental Support Vector Machine (B-ISVM) ismini verdikleri yöntem kullanarak, %100, %100, %100 ve %99,11 'lik bir başarı; M. Bahrololum ve arkadaşları, Karar Ağaçları ile %99.96, %99.97, %99.66, %88.33 ve %99.02 'lik; Ammar Alazab ve arkadşları, bilgi kazancı temelli bir yaklaşımla, %98.2, %97.2, %99.6, %92.5 ve %99.7 'lik; Vikas Sharma ve Aditi Nema, Karar Ağaçları ve genetik algoritmalar kullanarak, %99.98, %88.19, %51, %94.70 oranlarında; Guisong Liu ve arkadaşları, hiyerarşik temel bileşen analizi ve yapay sinir ağları (HPCANN-Hierarchical Principal Component Analysis Neural Networks) kullanarak, %97.1, %100, %100 ve %97.2 oranlarında; Xingchao Gong ve Xin Guan, uzman sistemler ile geri yayılımlı yapay sinir ağları kullanmış ve %100, %100, %99.76, %99.85, %99.88 'lik; Wei-Chao Lin

ve arkadaşları, CANN(Cluster center And Nearest Neighbor) olarak adlandırdıkları yöntemleri ile %99.68, %99.98, %98.49, %17,31 ve %91,74 oranlarında; Chih-Fong Tsai ve Chia-YİN Lin, TANN- Triangle Area Based Nearest Neighbors yöntemi ile, %71,31, %95.87, %93.43, %40 ve %85.84 'lük başarı oranları elde etmişlerdir.

Literatür çalışmaları incelendiğinde büyük bir çoğunluğun KDD CUP99 veri setinin kullanıldığı gözlemlenmiştir. Veri seti, herbir kayıt için 41 adet öznitelikten oluşmaktadır. KDD CUP99 ve NSL-KDD veri setlerinin özniteliklerinin detaylı bir açıklaması Tablo 2.2 'de verilmiştir:

Tablo 2.2: KDD CUP99 ve NSL-KDD Verisetlerinin Özniteliklerinin Listesi.

No	Öznitelik Adı	Açıklama	Türü
1	Duration	Bağlantının uzunluğu (sn.)	Sürekli
2	Protocol Type	Protokol türü (tcp,udb, vs.)	Ayrık
3	Service	Hedefteki ağ hizmeti, (http, telnet vs.)	Ayrık
4	Flag	Bağlantının normal veya hata durumu	Ayrık
5	Src-bytes	Kaynaktan hedefe veri bayt sayısı	Sürekli
6	Dst-bytes	Hedeften kaynağa veri bayt sayısı	Sürekli
7	Land	Eğer bağlantı aynı ana makineye/ makineden aynı bağlantı noktasına/ noktasından; Aksi takdirde 0	Ayrık
8	Wrong-fragment	Yanlış bölüm(fragment) sayısı	Sürekli
9	Urgent	Acil paketlerin sayısı	Ayrık
10	Hot	Sıcak göstergelerin sayısı	Ayrık
11	Num-failed-logins	Başarısız giriş denemesi sayısı	Ayrık
12	Logged-in	Başarılı giriş 1; Aksi takdirde 0	Ayrık
13	Num-compromised	compromised durum sayısı	Ayrık
14	Root-shell	Root shell gerçeklenirse 1, yoksa 0	Ayrık
15	Su-attempted	Eğer su root komutu denenirse 1; Aksi takdirde 0	Ayrık
16	Num-root	Root erişimlerinin sayısı	Ayrık
17	Num-file-creations	Dosya oluşturma işlemlerinin sayısı	Ayrık
18	Num-shells	shell prompt 'larının sayısı	Ayrık
19	Num-access-files	Erişim kontrol dosyalarındaki işlem sayısı	Ayrık
20	Num-outbound-cmds	Bir ftp oturumunda giden komut sayısı	Ayrık
21	Is-host-login	Oturum açma ana makine listesine aitse 1; Aksi takdirde 0	Ayrık
22	Is-guest-login	Oturum açma bir misafir girişi ise 1; Aksi takdirde 0	Ayrık
23	Count	Son iki saniyede mevcut bağlantı ile aynı istemciye yapılan bağlantı sayısı	Ayrık

No	Öznitelik Adı	Açıklama	Türü
24	Srv-count	Son iki saniyede mevcut bağlantı ile aynı hizmete yapılan bağlantı sayısı	Ayrık
25	Serror-rate	SYN hataları olan bağlantıların yüzdesi	Ayrık
26	Srv-serror-rate	SYN hataları olan bağlantıların yüzdesi	Ayrık
27	Rerror-rate	REJ hataları olan bağlantıların yüzdesi	Ayrık
28	Srv-rerror-rate	REJ hataları olan bağlantıların yüzdesi	Ayrık
29	Same-srv-rate	Aynı hizmetlere yapılan bağlantıların yüzdesi	Ayrık
30	Diff-srv-rate	Farklı hizmetlere yapılan bağlantıların yüzdesi	Ayrık
31	Srv-diff-host-rate	Farklı ana bilgisayarlara yapılan bağlantıların yüzdesi	Ayrık
32	Dst-host-count	Hedef ana bilgisayar için sayı	Ayrık
33	Dst-host-srv-count	Hedef ana bilgisayar için Srv-cout	Ayrık
34	Dst-host-same-srv-rate	Hedef ana bilgisayar için same-srv-rate	Ayrık
35	Dst-host-diff-srv-rate	Hedef ana bilgisayar için diff-srv-rate	Ayrık
36	Dst-host-same-src-port-rate	Hedef ana bilgisayar için same-src-port-rate	Ayrık
37	Dst-host-srv-diff-host-rate	Hedef ana bilgisayar için srv-diff-host-rate	Ayrık
38	Dst-host-serror-rate	Hedef ana bilgisayar için serror-rate	Ayrık
39	Dst-host-srv-serror-rate	Hedef ana bilgisayar için srv-serror-rate	Ayrık
40	Dst-host-rerror-rate	Hedef ana bilgisayar için rerror-rate	Ayrık
41	Dst-host-srv-rerror-rate	Hedef ana bilgisayar için srv-rerror-rate	Ayrık

KDD Cup 99 veri seti, 1999 yılından beri, STS değerlendirmeleri için referans veri seti olarak yaygın bir şekilde kullanılmaktadır [32]. KDD Cup 99 ’un eğitim ve test veri setinde sırasıyla; 4.898.431 ve 311.029 kayıt bulunmaktadır. Kayıtların her biri, 41 adet öznitelik içerir ve normal veya tam olarak belirli bir saldırı türüyle etiketlidir. Tüm saldırılar aşağıdaki dört kategoriden birine aittir:

- DoS: Ağın kaynaklarının ve hizmetlerin normal kullanıcılar için sınırlı veya kullanılamayacak şekilde tüketilmesi girişimidir.

- User to Root (U2R): Saldırganın bir makinedeki normal kullanıcı hesabına erişmeye başladığı ve güvenlik açıklarından yararlanarak makineye erişimin sağlandığı bir saldırıdır.
- Remote to Local (R2L): Bir saldırganın uzak bir sistemde hesabı olmadığında gerçekleşir, ancak bir ağ üzerinden bir sisteme paket gönderme olanağına sahip olduğu ve bu sistemin kullanıcısı olarak yerel erişim kazanmak için güvenlik açıklarından yararlandığı durumlarda oluşur.
- Probing: Saldırganın, güvenlik kontrollerini atlatma amacıyla, sistemler hakkında bilgi toplamak için ağı taradığı bir saldırı türüdür.

KDD CUP99 öznitelikleri aşağıdaki 3 gruba ayrılır:

1. Temel öznitelikler: Bu kategori bir TCP/IP bağlantısından çıkarılabilecek tüm öznitelikleri kapsar. Bu özniteliklerin çoğu, tespitte aşikar bir gecikmeye sebep olur.
2. Trafik öznitelikleri: Bu kategori, belirli bir aralığa göre hesaplanan öznitelikleri içerir.
3. İçerik öznitelikleri: DoS ve Probing saldırılarının çoğunda, devamlı ardışık desenler vardır. Bu durum saldırıların ana bilgisayarlara çok kısa bir sürede çok sayıda bağlantı kurmasından kaynaklanır. Bu saldırılardan farklı olarak, R2L ve U2R ataklarında devamlı ardışık modeller oluşturan davranışlar yoktur. R2L ve U2R saldırıları; paket payload 'larına gömülür ve normalde sadece tek bir bağlantı içerirler. Bu tür saldırıları tanımlamak, paket payload daki şüpheli davranışı tanımlamaya yarayacak bazı ilgili öznitelikler ile gerçekleştirilir. Bu özelliklere de içerik öznitelikleri denmektedir [33].

2.2. SALDIRI NEDİR?

Bir kaynağın gizliliğini, bütünlüğünü veya kullanılabilirliğini tehdit eden, sisteme ait güvenlik politikalarını ve yasal korumaları ihlal eden tüm eylemler "saldırı" olarak adlandırılmaktadır. Bu eylemi gerçekleştiren kişiye ise "saldırgan" denmektedir. Saldırganların bir sisteme izinsiz giriş yapabilmelerinin üç temel yöntemi vardır. Bu yöntemler aşağıdaki gibidir:

- Donanım ve güvenlik sistemini hedefleme: Bu yöntem, saldırganın, saldırdığı sistemin kullandığı donanım ve güvenlik yöntemlerine dair bazı bilgileri bildiğini varsayar.

- Yazılımların içermiş oldukları bug olarak isimlendirilen açıklıklar: Maalesef bilinen zayıflıklar bazen yeterli süre içerisinde giderilmez. Yeterli süre içerisinde giderilmeyen açıklıklar, saldırganların sisteme girebilmesi için açılmış bir kapı olur.

- Brute Force saldırılarından faydalanma: Bu yöntemde saldırgan, çeşitli algoritmalar kullanarak, olası kod kombinasyonlarını deneyerek güvenlik şifrelerini kırmayı, kullanıcının şifresini veya sosyal kimlik numarasını ele geçirmeyi amaçlar. Bir Brute Force saldırısında, otomatikleştirilmiş yazılım istenen verinin değerine göre çok sayıda ardışık tahmin üretmek için kullanılır. Brute Force saldırıları, suçluların şifreli verilerinin kırılması veya kurum, kuruluş ve organizasyonların ağ güvenliğini test etmek için güvenlik analistleri tarafından da kullanılmaktadır.

- Dictionary atakları: En yaygın olarak kullanılan brute force atak türlerinden birisidir. Bu ataklar sözlükteki tüm kelimeleri çeşitli kombinasyonlarla deneyerek sistemi kırmaya çalışırlar.

- Doğum günü atakları: Bir diğer brute force atak türüdür. Bu ataklar kişilerin aynı günde doğmuş olma paradoksuna dayanmaktadır. Hashler içerisinde çakışma bulma olasılığına dayanan istatistiksel bir yöntemdir.

Saldırıların başarısı genellikle hesaplama gücüne ve denenen kombinasyonlarının sayısına dayanmaktadır.

2.3. AĞ ÜZERİNDE GERÇEKLEŞEN ATAK TÜRLERİ

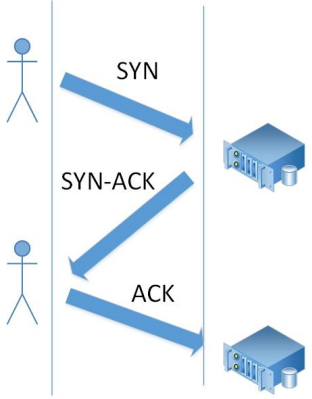
İnternet kullanımın hızla artması birçok fırsat ve imkan sağlamakla beraber kötü niyetli saldırıların meydana gelme riskini de arttırmıştır. Daha önceden hiç olmadığı kadar internet, hepimizin bildiği üzere artan talep ve kullanım alanları ile hayatımızı kolaylaştırmaktadır. İnsanlar, organizasyonlar, çeşitli kurum ve kuruluşlar verilerini ağ(LAN,WAN) veya internet üzerinden paylaşmaktadır. Bunun yanı sıra, internet üzerinden alım satım yapılmasına olanak veren ve e-ticaret siteleri olarak adlandırılan

binlerce web sitesi mevcuttur. TÜSİAD 'ın 2017 yılında yayınlamış olduğu e-ticaret raporuna göre [34], dünyada e-ticaret hacmi son 4 yılda 630 milyar dolardan 1,6 trilyon dolara ulaşmış, ülkemizde ise 2016 yılı e-ticaret hacmi 17,5 milyar Türk Lirasına ulaşmıştır. Aynı raporda, Türkiye 'de internet kullanan her üç kişiden birinin çevrimiçi alış veriş yaptığı bilgisine yer verilmiştir. Rakamlar; 2019 yılı tahminleri için Türkiye'de ki e-ticaret hacminin iki katına çıkabileceğini göstermektedir. Hal böyle iken, günümüzde internetin olmadığını hayal etmek bile insanlara büyük bir korku vermektedir. İnternetin sağlamış olduğu sayısız imkan ve kolaylıktan ötürü hayatımızdan çıkaramayacağımız büyük bir gerçektir. Bu sebeple kullanıcıların, finansal ve duygusal kayıplara uğramaması birinci öncelik olmalıdır. Kullanıcıların, internet kullanımı ve güvenlik zafiyetleri konusunda bilinçlendirilmesi, gerekli tedbirlerin alınması ve mevcut sistemlerin daha güvenli hale getirilmesi hususunda gerekli araştırma geliştirme çalışmalarına büyük bir hassasiyetle yaklaşılmalı ve asla göz ardı edilmemelidir.

Ağ üzerinde gerçekleşen saldırılar zaman ve kaynak tüketirler. Yaygın olarak kullanılan saldırılar sırasıyla; SYN Saldırısı, Flood Saldırıları, Paket Süzme, Zehirleme, Virüsler ve Ajan Yazılımlar alt başlıkları ile aşağıdaki gibidir.

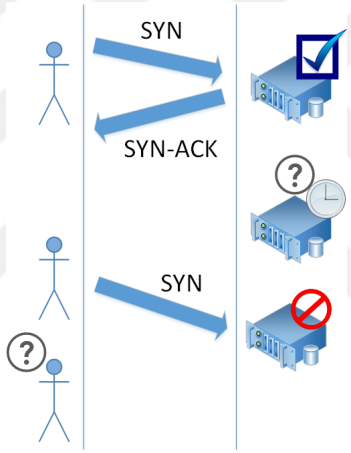
2.3.1. SYN Saldırısı

TCP (Transmission Control Protocol) bağlantıları için sunucu ve istemci arasında bir takım olaylar zinciri gerçekleştirilmektedir. Bu olaylar zinciri "üçlü el sıkışma" olarak adlandırılmaktadır. Üçlü el sıkışma aşağıdaki şekilde gösterildiği gibidir.



Şekil 2.1: Üçlü el sıkışma.

Şekil 2.1 'de gösterildiği üzere üçlü el sıkışma öncelikle istemcinin sunucuya kullanmış olduğu sistem altyapısı hakkında bilgileri içeren bir SYN (synchronize) mesajı göndermesi ile başlar. Sunucu istemcinin göndermiş olduğu SYN paketini alır ve ACK(acknowledgement) ile istemciye geri gönderir. İstemci, sunucunun göndermiş olduğu SYN+ACK paketini alır ve akabinde sunucuya ACK paketi gönderir. Böylece sunucu ve istemci arasında sağlıklı bir bağlantı kurulmuş olur. Kurulan bu bağlantının Şekil 2.2 'de görüldüğü üzere saldırgan sunucuya çeşitli paketler gönderip ACK bilgisini göndermez ise bağlantı yarı açık kalmış olur. Dolayısı ile bu yarı açık bağlantı ile saldırgan sunucunun çeşitli kaynaklarını tüketir. Çeşitli kaynakları tükenen sunucu normal kullanıcılara dahi hizmet veremez hale gelir. Bu durum SYN saldırısı olarak adlandırılmaktadır.



Şekil 2.2: SYN saldırısı.

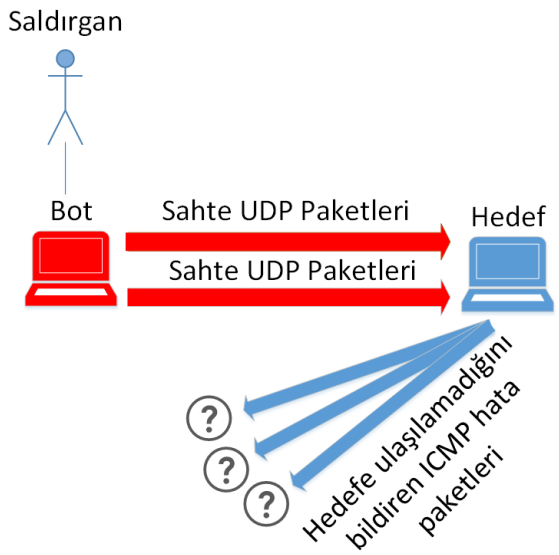
SYN saldırıları bir DDoS(Distrubuted Denial of Services) atağı gerçekleştirmenin en kolay ve en tehlikeli yollarından birisidir.

2.3.2. Flood Saldırıları

DoS(Servis Reddi) saldırılarının ilk versiyonu flood saldırılarıdır. Saldırgan, kurbanın kapasitesinin çok daha üzerinde trafik gönderir. Saldırganın daha fazla trafik gönderebilmesi için daha hızlı bir ağ bağlantısına sahip olması gererekmektedir. DoS ataklarının en düşük seviye ve tamamen önlenmesi en zor olan şeklidir. Örneğin UDP Flood saldırısı bir DoS saldırısıdır ve User Datagram ile sessionless/connectionless ağ protokollünü kullanır. UDP protokol saldırısı, uzak bir ana bilgisayarın rastgele portlarına

geniş miktarda UDP protokol paketinin gönderilmesi ile başlatılabilmektedir. Bunun sonucu olarak ana bilgisayar Şekil 2.3 'de gösterildiği gibi aşağıdaki adımları icra eder:

- Sunucu önce, belirtilen bağlantı noktasında istekleri dinleyen herhangi bir programın çalışıp çalışmadığını denetler.
- Hiçbir program bu bağlantı noktasında paket almıyorsa, sunucu gönderene ulaşılamayacağını bildiren bir ICMP (ping) paketiyle yanıt verir.



Şekil 2.3: UDP Flood Saldırısı.

2.3.3. Paket Sniffing Süzme

Ağ izleyicisi veya ağ analizcisi olarak da adlandırılan bir paket dinleyicisi, ağ trafiğini izlemek ve sorunlarını gidermek için bir ağ veya sistem yöneticisi tarafından yasal olarak kullanılabilmektedir. Sistem yöneticisi, paket dinleyicinin yakaladığı bilgileri kullanarak, hatalı paketlerin tanımlanması ve darboğazların belirlenmesi ile ağdaki veri iletiminin etkili bir şekilde sürdürülebilmesini sağlamaya çalışır. Basit bir paket dinleyicisi, belirli bir ağ üzerinden geçen tüm veri paketlerini kolayca yakalar. Tipik olarak, paket dinleyicisi sadece söz konusu makine için öngörülen paketleri yakalayacaktır. Bununla birlikte, paket dinleyicisi ayrıca, hedeften bağımsız olarak ağ üzerindeki tüm paketleri de yakalayabilir. Bir ağa bir paket dinleyicisi yerleştirerek, kötü niyetli bir saldırgan, tüm ağ trafiğini yakalayabilir ve analiz edebilir. Belirli bir ağ içinde kullanıcı adı ve şifre bilgisi genellikle

açık metinde iletilir, bu da bu ve benzeri bilgilerin iletilen paketlerin analiz edilerek görülebileceği anlamına gelir.

2.3.4. Spoofing Zehirleme

Spoofing saldırısı, bir kişinin veya programın bir tür yanlış bilgi sağlaması ve dolayısıyla meşru olmayan bir avantaj elde etmesi durumudur.

2.3.5. Zararlı Yazılımlar

1. Virüsler: Virüsler, kendini gerçek programlarda çoğaltan ve bir program her çalıştığında çalışan küçük bir yazılım parçasıdır. Çoğu diğer programları yeniden üretebilir ve bunlara saldırabilirler. E-posta virüsleri en yaygın olanlarıdır. E-posta virüsleri, e-posta iletilerinde gezinir, genellikle kendini kurbanın e-posta adres defterinde bulunan düzinelerce kişiye otomatik olarak göndererek çoğaltır.

2. Solucanlar: Kendisini çoğaltmak için bilgisayar ağlarını ve güvenlik açıklarını kullanan küçük yazılımlardır. Solucanlar, çok hızlı genişleyebilirler, ağı tararlar ve belirli bir güvenlik açığına sahip olan makineye kendilerini kopyalarlar.

3. Truva atları: Programı çalıştıran kullanıcının ayrıcalıklarından yararlanabilecek gizli fonksiyonlar içeren bir bilgisayar programıdırlar. Diski silebilir, kredi kartı numaralarınızı ve şifrelerinizi bilgisayar korsanlarına gönderebilirler.

2.3.6. Ajan yazılımlar

Casus yazılımlar; kişisel bir bilgisayarda kullanıcının izni ve bilgisi olmadan, kullanıcının bilgisayarı kullanma veya web tarama davranışları gibi kullanıcı hakkındaki bilgileri toplayan bilgisayar yazılımlarıdırlar. Aşağıdaki gibi üç sınıfa ayrılabilirler:

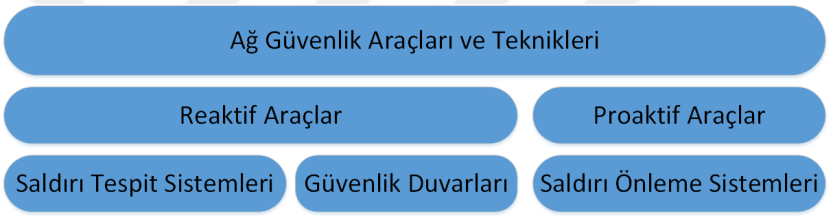
1. Zararsız ama can sıkıcı olanlar: Web tarayıcısının varsayılan ana sayfasında, pop-up vb. bazı reklamlara yer verenlerdir.

2. Bilgi toplayanlar: Bu sınıf genellikle, kullanıcı hakkında, en çok ziyaret edilen siteler gibi kullanıcı hakkında bir tür kullanıcının tercihlerine ve ilgi alanlarına yönelik bilgi toplamakla ilgilidir.

3. Kötü amaçlı olanlar: Sunucuya, özel ve gizli bilgileri göndermenin yanı sıra mahrem ve kritik öneme sahip bilgilerin toplanarak loglanmasını sağlarlar.

2.4. GÜVENLİK ARAÇLARI VE YÖNTEMLERİ

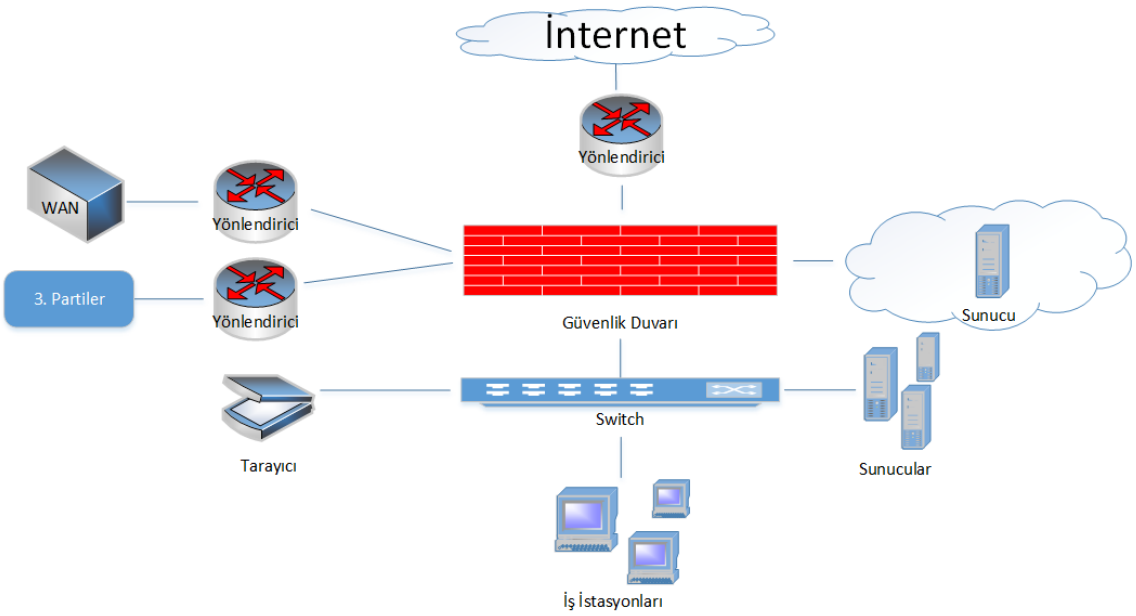
Ağ güvenliğini sağlamak için Şekil 2.4 'de gösterildiği gibi, Reaktif araçlar ve Proaktif araçlar olmak üzere temel iki tür araç ve teknik kullanılır. Reaktif yöntem ve araçlar, tanımlanmamış davranışları tespit ettikten sonra bir çeşit tepki veya uyarı üreten araç ve tekniklerdir. Güvenlik Duvarları ve STS bu kategoriye girmektedir. Proaktif yöntem ve araçlar ise tespit edilen tanımlanmamış davranışların sisteme zarar vermesini engellemek için kullanılan araç ve yöntemlerdir. SÖS 'ler bu sınıfa girmektedir.



Şekil 2.4: Güvenlik araçları ve yöntemleri.

2.4.1. Güvenlik Duvarları

Güvenlik duvarları, ağı dış saldırılardan veya dış ağlardan koruyan yazılım veya donanımların birleşimidir. Özel bilgilerin korunmasında ilk savunma hattı olarak kabul edilmektedir. Şekil 2.5 'de görülebileceği gibi, güvenlik duvarı iki ağ arasında bir duvar görevi görebilir. Her iki ağa da erişmek isteyen kişi ilgili ağlara girmeden önce bu duvarı geçmek zorundadır. Güvenlik duvarı, iki ağ arasındaki trafik akışını kontrol etmek için kurulmuş bir sistemdir. Güvenlik duvarı IDS ile birlikte çalışarak tespit edilen tanımlanmamış davranışların, sisteme zarar vermesinin engellenmesinde kullanılmaktadır. Aşağıdaki gibi birçok güvenlik duvarı teknik ve yöntemi vardır:



Şekil 2.5: Güvenlik duvarı.

- Paket filtreleme: Ağ üzerinde gidip gelen her bir paket kontrol edilir ve tanımlı kurallara göre paketlerin iletilmesini sağlar veya engeller. Paket filtreleme, kullanıcılar için oldukça şeffaf ve etkili biçimde çalışmaktadır, ancak yapılandırılması zordur. Ayrıca, IP spoofing 'e karşı oldukça etkili bir yöntemdir.

- Application gateway: FTP ve Telnet sunucuları gibi belirli uygulamalara güvenlik mekanizmalarını uygulama noktasında oldukça başarılı olan bu yöntem, performans düşüşüne de neden olabilmektedir.

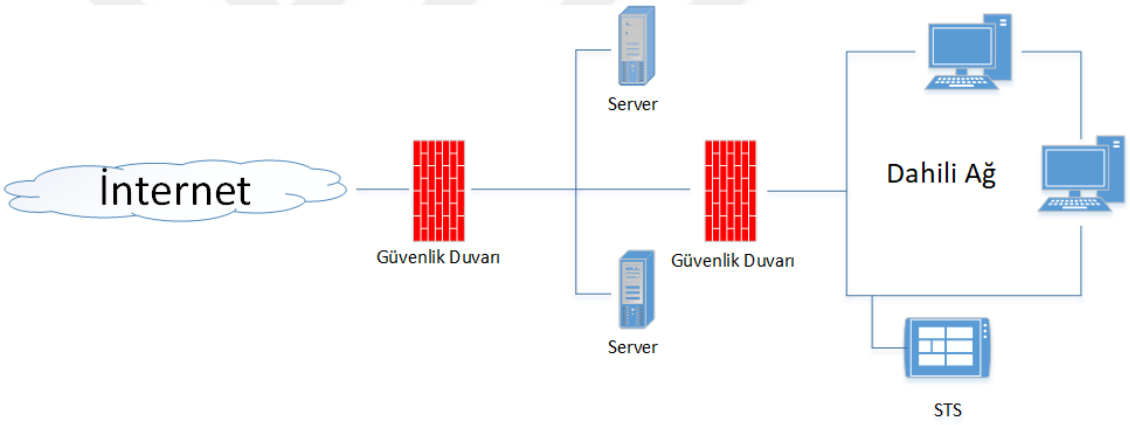
- Circuit-level gateway: TCP veya UDP bağlantısı kurulduğunda güvenlik mekanizmalarının uygulanması işlemidir. Bağlantı yapıldıktan sonra, paketlerin tekrar kontrol edilmesine gerek olmaksızın hostlar arasında iletimi sağlar.

- Proxy sunucusu: Proxy sunucusu, gerçek ağ adreslerini etkin bir şekilde gizler ve ağa giren ve çıkan tüm mesajları engeller.

2.4.2. Saldırı Tespit Sistemleri

Saldırı Tespit Sistemleri; bilgisayar veya ağ sistemlerinin başarılı bir şekilde çalışmalarının sürdürülebilmesi amacıyla, yazılım, donanım veya yazılım ve donanımın birlikte kullanılması ile oluşan bir güvenlik sistemidir. STS 'ler, dışarıdan yapılan

saldırıları veya kurum içerisinde yetkisiz giriş, kötüye kullanma vb. birçok aktivitenin bilgilerinin toplanıp, kaydedilmesini (loglama) ve kötü niyetli olan davranışların tespit edilmesini sağlarlar. Şekil 2.6 'da görülebileceği gibi STS 'ler, hem dışsal hem de içsel ağların dışarıdan gelebilecek saldırılardan korunmasını sağlarlar. Saldırı Tespit Sistemlerinin çalışma mantığı, güvenlik kameraları tarafından şüpheli veya yetkisiz bir aktivitenin tespit edilmesine benzetilebilir. Güvenlik duvarlarına benzer şekilde, gelen ve giden tüm ağ trafiğini denetlerler. Bununla birlikte, güvenlik duvarlarından farklı olarak, belirli paketleri engelleyerek veya bazı paketleri ileterek trafik akışını değiştirmezler. Daha ziyade, bir saldırının veya başka bir kötüye kullanımın göstergesi olabilecek kötü amaçlı davranışları ararlar ve güvenlik uzmanlarınca değerlendirilmesi için bu davranışları bir alarm ile kaydederler ve en önemlisi paket içeriklerini de kontrol edebilecek özelliğe sahiptirler.



Şekil 2.6: Saldırı tespit sistemi.

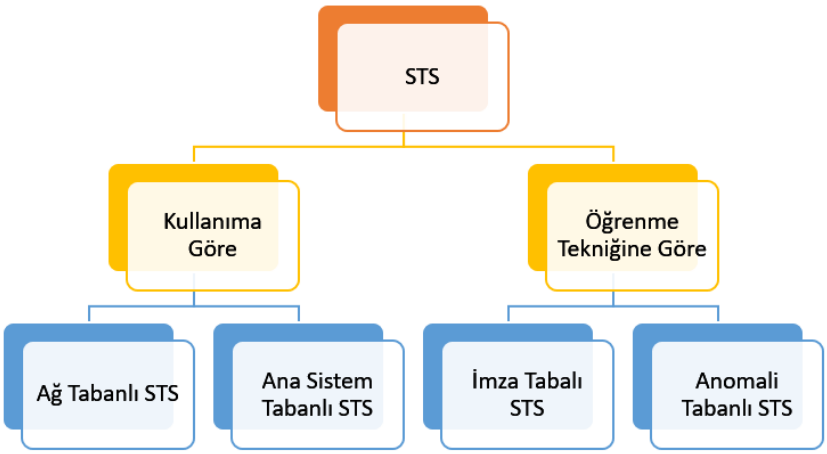
STS 'lerin birincil amacı bilgisayar sistemlerinin kötüye kullanımını tespit etmektir. İdeal bir STS, tanımlı olmayan davranışları tespit edebilir, güvenlik uzmanlarını e-posta veya çeşitli uyarı mekanizmaları ile sorun hakkında bilgilendirebilir ve bu tür kötüye kullanımın yarattığı riski minimize etmek amacıyla bağımsız hareket edebilir. STS 'nin bir diğer amacı, bir arıza durumunda kurtarma işlemini kolaylaştırmak, olayları kaydetmek (loglama) ve bir saldırıya dahil olan kaynakları, teknikleri ve sistem hakkındaki verileri, davranışların tanımlanması için toplamaktır. Bu bilgi ayrıca yasal amaçlarla ve saldırganın aleyhine delil olarak kullanılabilir.

İdeal bir Saldırı Tespit Sisteminin sahip olması gereken özellikleri aşağıdaki gibidir:

- Normal veya kabul edilebilir bir kullanıcı davranışı ile potansiyel olarak güvenli olmayan veya şüpheli davranışlar arasında kesin bir ayrım yapabilmelidir.
- Karmaşık büyük ağların içinde kolay ölçeklenebilir olmalıdır.
- Karmaşık yapıları ve modern heterojen ağ ve sistem mimarilerine kolaylıkla konuşlandırılabilmelidir.
- Daha önce tanımlanmamış yeni bir saldırılara dahi gereken minimum insan müdahalesi ile cevap verebilmelidir.
- Raporları ve grafikleri gerçek zamanlı olarak üretebilmeli ve yöneticinin ağa daha fazla zarar gelmesini önlemek için gecikme olmaksızın gerekli işlemi yapabilmesine fırsat vermelidir.
- Ağın ve diğer güvenlik mekanizmalarına uyumlu olmalıdır.
- İkincil seviye savunma sistemi olarak hareket etmeli ve aynı zamanda diğer güvenlik mekanizmalarına yönelik başarısızlıkları veya saldırıları tespit edebilmelidir.
- Güvenliği arttırarak izinsiz girişlere, saldırılara ve uygunsuz davranışlara ilgili güvenlik yerlerinde cevap verebilmelidir.
- Bir sistemin tüm bölümlerinde tanımlı olmayan davranışları tespit edebilmelidir.
- Sistemin bütünlüğünün bozulmadan korunması ve denetim bilgilerinin tehlikeye girmemesinin yanı sıra kendi sitemini de saldırılara karşı koruyabilmeli ve saldırılar karşısında sistemin işleyişinin olumsuz yönde etkilenmemesini veya etki oranını en aza indirebilecek kapasitede olmalıdır.
- Ağ arızaları, güvenilir olmayan iletim, yüksek sistem yükleri ve hizmet engelleme saldırılarına karşın işleyişini sürdürme kabiliyetine sahip olmalıdır.
- Sınırlı bir hafıza ve minimum sayıda kaynak tüketmelidir.
- STS tarafından üretilen bilginin, sistemin ikinci bir geri kazanımı ve herhangi bir hasar durumunda ağ profilinin çıkarılabilmesi için kullanılabilir ve aynı zamanda hukuki boyutta mahkemelerce delil olarak kullanılabilecek türden olmalıdır.

- Konuşlandırıldığı kurumun güvenlik politikasını yansıtmalı ve bu kuruluşun önceliklerini mevcut izleme düzeyini ve şeklini şekillendirmeye izin vermelidir.

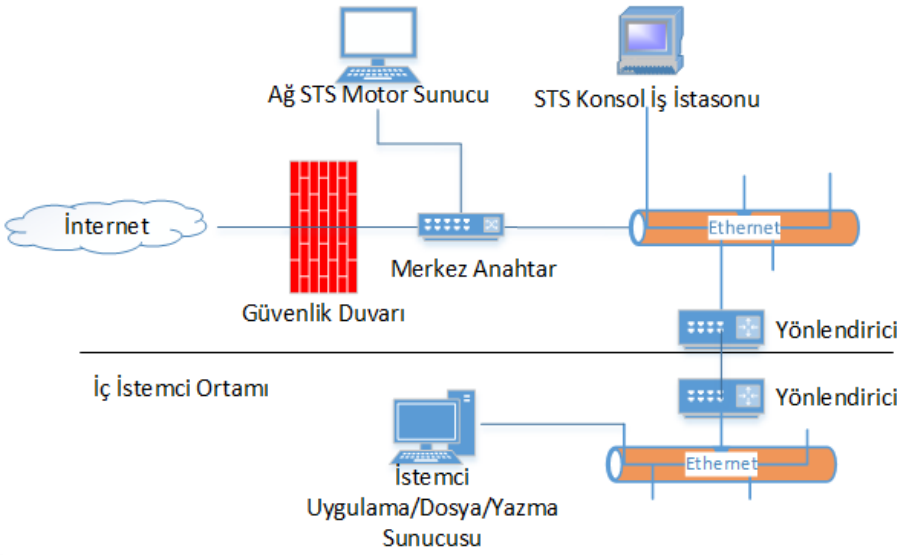
Saldırı Tespit Sistemleri Şekil 2.7 'de gösterildiği gibi, kullanımına göre; İstemci ve Ağ Tabanlı, saldırıları tespit etme yöntemlerine göre ise İmza (Kural) ve Anomali Tabanlı olarak sınıflandırılmaktadırlar. Bu sınıflandırmalara ek olarak birde Hibrit Saldırı Tespit Sistemleri bulunmaktadır.



Şekil 2.7: Saldırı tespit sistemlerinin sınıflandırılması.

2.4.2.1. İstemci Tabanlı (Host-Based) Saldırı Tespit Sistemleri

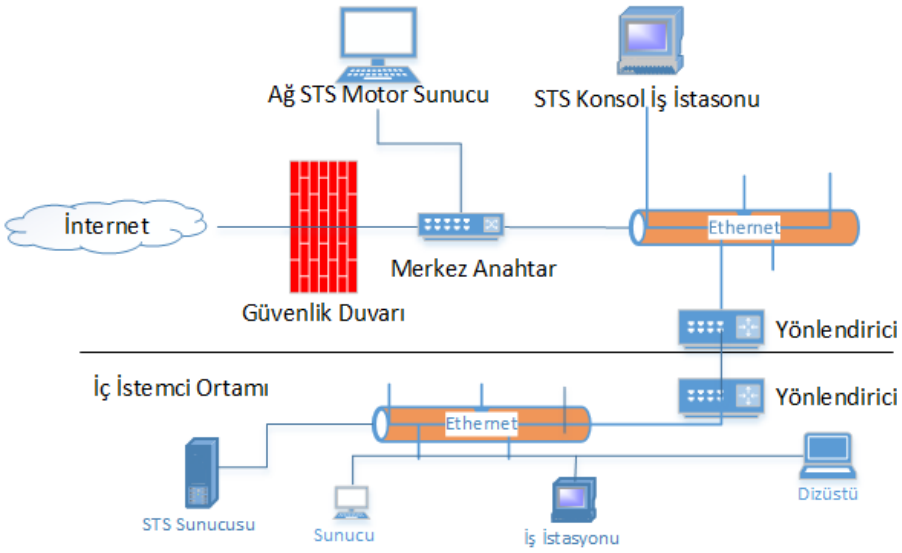
İstemci Tabanlı Saldırı Tespit Sistemleri, belirli bir bilgisayar üzerinde bulunur ve belirli bir bilgisayar sistemi için koruma sağlarlar. Bu saldırı tespit sistemleri, yerel olarak, Ağ Tabanlı STS 'lere kıyasla çok yönlü bir sistem oluşturan ana makinelere kurulurlar. Sunucular, iş istasyonları ve dizüstü bilgisayarlar gibi birçok farklı makineye kurulabilirler. Şekil 2.8 'de İstemci Tabanlı bir STS modeli gösterilmektedir. Bu model bir STS, uzaktan izleme, olayların kayıtlarının uzaktan depolanması ve bu kayıtların yeni veya mevcut istemcilere aktarılmasına olanak sağlar. Bu tür Saldırı Tespit Sistemleri şifreli paketlerdeki atakları da tespit edebilir.



Şekil 2.8: İstemci tabanlı saldırı tespit sistemi.

2.4.2.2. Ağ Tabanlı (Network-Based) Saldırı Tespit Sistemleri

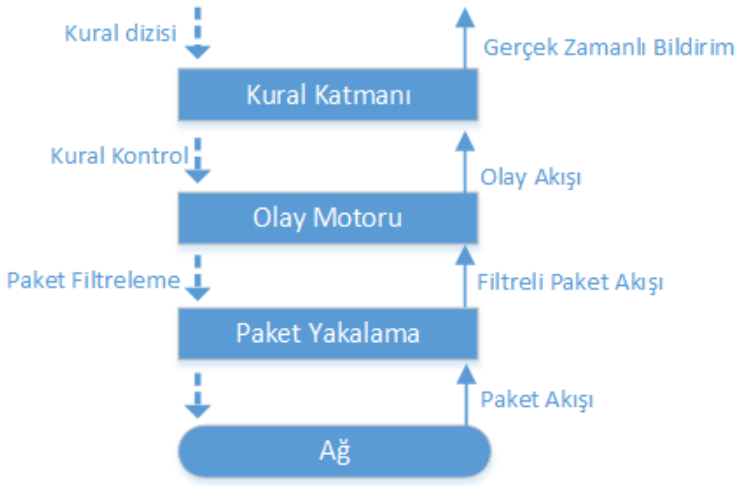
Ağ tabanlı STS 'ler; ağ trafiği paketlerini (TCP, UDP) yakalar ve olası bir olayın gerçekleşip gerçekleşmediğini belirlemek için trafiği, bir dizi kural, imza veya anomali tespiti ile analiz ederler. Bir Ağ Tabanlı STS, ağ kablosundaki paketleri izler ve bir bilgisayar korsanının sisteme girmeye çalışıp çalışmadığını veya servis reddi saldırısına (DoS) neden olup olmadığını bulmaya çalışır. Bu işlemi, çok sayıda TCP bağlantı isteğini (SYN) bir hedef makinedeki birçok farklı bağlantı noktasına kadar izleyerek, bir kullanıcının TCP bağlantı noktası taraması yapmaya çalışıp çalışmadığını analiz ederek gerçekleştirir. Kendi trafiğini izlemek için bir sistemde ya da tüm ağ trafiğini (hub, yönlendirici ve sonda) rastgele bir şekilde izlemek için bağımsız bir sistemde çalıştırabilir. Ağ Tabanlı STS 'ler, isminden de anlaşılacağı üzere ağ tabanlı oldukları içindir ki, sadece belirli bir ana bilgisayara giden paketlerin değil, ağ segmentindeki tüm makineler/sistemler 'in korunmasında kullanılmaktadırlar. Bir Ağ STS, aynı ağı kullanan birçok makineyi izler. Ağ tabanlı STS' ler, yönlendiriciler gibi pek çok aktif ağ elemanlarına da kurulabilirler. Cisco Secure IDS, Hogwash, Dragon, E-Trust IDS 'ler örnek Ağ Tabanlı Saldırı Tespit Sistemleridirler. Bunların yanı sıra, güçlü bir Ağ Tabanlı STS olan fakat fazla bilinmeyen açık kaynaklı bir STS ise Bro Saldırı Tespit Sistemidir.



Şekil 2.9: Ağ tabanlı saldırı tespit sistemi.

Bro Saldırı Tespit Sistemi

Bro, Vern Paxson tarafından 1995 yılında, Lawrence Livermore Ulusal Laboratuvarı tarafından tasarlanan ve o zamandan beri aktif olarak gelişmeye devam eden, Unix tabanlı bir Ağ Saldırı Tespit Sistemi 'dir. Kaliforniya Üniversitesi, Berkley ve LBL' de faaliyete geçmiştir. Bro STS, Şekil 2.10 'da gösterildiği gibi kural katmanı, olay motoru ve paket yakalama olmak üzere üç ana bileşenden oluşmaktadır [35].



Şekil 2.10: Bro temel bileşenleri.

Şekil 2.10 'da görüldüğü üzere katmanlı bir mimari kullanılmaktadır. Buradaki en düşük

katman olan paket yakalamanın görevi; paketleri filtrelemek ve protokol analizi gibi düşük seviyeli işlemleri gerçekleştiren “event engine” ’i filtreli paketler ile beslemektir. Olay motoru, kural katmanına iletilen bir olaylar akışı üretir. Kural katmanı, olayları kullanıcı tarafından belirlenen kurallara göre değerlendirir.

- Paket Yakalama: Paket yakalama katmanı, paketleri ağdan yakalamak için, farklı ağ bağlantı teknolojileri için platformdan bağımsız bir arayüz sağlayan libpcap kütüphanesi kullanır. Libpcap ayrıca analiz edilecek paketlerin miktarını azaltan bir filtre içerir. Filtre, kullanıcı tarafından belirlenmiş komut cümleciklerine göre çalışma zamanı sırasında oluşturulur. Örneğin, http trafiği için hiçbir kural tanımlı değilse, trafik filtrelenir ve daha sonra paket yakalama katmanındaki tüm http trafiğini görmezden gelir ve üst katmana hiçbir http trafiği gönderilmez. Bu ayrıca, ağa giren paketlerin önemli bir kısmının düşük bir seviyede reddedilmesine de izin verir. Böylece libpcap, Bro’nun uygulama protokolleriyle (ör., Parmak, ftp, telnet) ilişkili tüm paketleri yakalayacaktır. Ayrıca, Bro’da Libcap kullanmanın birçok avantajı vardır:
  - Bro’ nun platform bağımsız olmasına yardımcı olur.
  - Bro tcpdump dosyaları üzerinde çalışabilir ve çevrimdışı analiz de mümkün hale gelmiş olur.
  - Bro ’yu, FDDI, SLIP gibi kullanılan ağ teknolojilerinin ayrıntılarından izole etmiş olur.
- Olay motoru: Olay motoru katmanında, ağ paketlerinin düşük seviyeli analizi yer alır. Olay motoru ilk olarak paket başlıkları üzerinde bütünlük denetimleri gerçekleştirir, tek IP paketleri toplanır ve daha sonra TCP veri akışları yeniden birleştirilir. Taşıma seviyesinde TCP, UDP ve ICMP paketleri analiz edilir. Uygulama seviyesinde HTTP, SMTP, DNS, FTP ve daha fazlası analiz edilir. Bir analizör beklenmeyen bir durum ile karşılaşırsa, işleme için kural katmanına gönderilen bir olay oluşturur. Olaylar bu süreçte oluşturulur ve üçüncü katmanda yer alan kural cümleciği yorumlayıcı tarafından sorgulanacak bir sıraya yerleştirilir.
- Policy Layer: Bu katmanda, kullanıcı tarafından özel Bro dilinde yazılan kural cümleciği, olay motoru tarafından oluşturulan olayları ele alır. Bir olay, ardarda birkaç işleyici tarafından işlenebilir. Bro başladığında, hangi olay motoru

çözümleyicilerinin başlayacağını bulmak için etkinleştirilmiş olay işleyicilerinden bakar. Bro 'ya yeni özellikler eklemek için uygulamanın protokolleriyle ilişkili olayları tanımlamak ve kural cümleciği yorumlayıcısının işlevselliğini genişletmek için ilgili olay işleyicilerini yazmak gerekir.

Bro 'nun hangi ağ olaylarının alarm açısından uygun olduğunu belirlemek için kullandığı temel analizörler kural cümlecikleridir. Çeşitli eylemlerin ve faaliyetlerin nasıl rapor edileceği ile birlikte, incelemek için hangi aktivitelerin tanımlanması gerektiği kural cümlecikleri tarafından belirlenirler. Bro, olayları aktif veya şüpheli olarak sınıflandırmak için policy leri kullanır. Bu aktif ağ oturumları, yerel taraftaki bir bağlantıyı sıfırlamak için "rst" çağrısının çağrılması veya ana yönlendiricinin ACL(Erişim Kontrol Listesi) 'sine bir IP adresi bloğu eklenmesi gibi diğer politikalar veya uygulamalar yoluyla gerekli olduğu belirlenebilir, işaretlenebilir, izlenebilir veya yanıtlanabilir. Kural dosyaları Bro scripting dilini kullanır. Bro kural dosyaları bir @load komutu kullanılarak yüklenir. Birden çok policy scripts dosyasında bir şey yüklemenin zararı yoktur; çünkü, @load komutunun çalışma matığı gereği "zaten yüklenmemişse bu komut dosyasında yüklenir". Bro içerisindeki policy scripts ler aşağıdaki çizelgedeki gibidir. Bro 'nun varsayılan kural cümleciklerinin tümü Tablo 2.3 'de gösterildiği gibi varsayılan olarak yüklenir. Varsayılan olarak yüklü gelmeyen ikinci grup kuralları, Tablo 2.4 'de gösterilmiştir. Hangi analizörlerin yüklendiğini belirlemek için "$ BROHOME / SITE" dosyayı düzenlenebilir veya oluşturulabilir. Özel analizörlerin yazılması içinde bu dizine gidilmesi gereklidir. Bir analizörü devre dışı bırakmak için, "$ BROHOME / site / brohost.bro" dosyasının başına, "@load brolite.bro" satırından önce, "@unload policy.bro" eklenir. Ek analizöterler de "$ BROHOME / site / brohost.bro" dizinine @load ile eklenir.

Tablo 2.3: BRO varsayılan olarak açık kurallar.

Site	Yerel ve komşu ağları statik pingten tanımlar.
Alarm	Alarmlar için log dosyasını açar.
TCP	SYN / RST TCP paketleri için BPF filtresini başlatır.
Login	rlogin /telnet analizörlerinin devre dışı bırakıldığından emin olur.
Weird	Olağan dışı olayları tespit etmek için işlemleri başlatır.
Conn	Bağlantı olaylarına erişir ve kaydeder.
Hot	Hassas erişimin belirli formunu tanımlar.
Frag	Tcp fragmanlarını işler.
Print resources	Kaynak kullanım bilgilerinin yazdırılması ve çıkış ayarlamalarında yararlanılır.
Signatures	İmza policy engine dır.
Scan	Tespit edilen mekanizmaları tarar.
Trw	Daha fazla dedektif mekanizma sunar.
HTTP	Genel http analizörüdür.
HTTP-request	Http isteklerinin ayrıntılı analizidir.
HTTP-reply	Http-yanıtının ayrıntılı analizidir.
FTP	FTP analizörüdür.
Port mapper	RPC portmapper istekleri kayıt ve analiz eder.
SMTP	E-posta trafiğini kayıt ve analiz eder.
TFTP	Tftp oturumlarını loglar ve tanımlar.
Worm	Code Red gibi http tabanlı solucan kaynakları işaretler.
Software	Yazılım versiyonunu takip eder.
Blaster	Solucanları yok etmek için arar.
Synflood	Synflood'u arar.
Stepping	Harici bir ağdan giriş yapılıp yapılmadığını algılamak için kullanılır.
Reduce memory	Durum kaydetmek için daha kısa zaman aşımları ayarlar ve belleğe kaydeder.

Tablo 2.4: BRO kapalı varsayılan kurallar.

Policy	Açıklama	Varsayılan olarak kapalı olma nedeni
Drop	Eğer site düşmanları uzaktan düşürme özelliğine sahipse eklenir.	Gerekirse açılır.
ICMP	Icmp analizörü.	Aşırı CPU tüketimine sebebiyet verir.
DNS	DNS analizörü.	Aşırı CPU tüketimine sebebiyet verir.
Ident	Kimlik programı analizörü.	Eski olduğundan pek tercih edilmez.
Gnutella	Gnutella'yı çalıştıran ana bilgisayara bakar.	İstenirse açılır.
SSL	SSL analizörü.	Deneyseldir.
SSH-stepping	Gelen ve giden bağlantıların SSH olduğu adımların tespitidir.	Aşırı CPU ihtiyacı.
Analy	İstatistiksel analiz yapar.	Sadece çevrim dışı analiz için kullanılır.
Backdoor	Backdoor ları inceler.	Toplu trafik yakalarken etkilidir.
Passwords	Metin şifresi arar.	Site, metin şifresi sağlamazsa açılması gerekir.
File-flushed	Tüm log dosyalarının her N saniyede temizlenmesidir.	Gerçek zamanlı izleme yapmak için istendiğinde açılabilir.

2.4.2.3. İstemci Tabanlı ve Ağ Tabanlı Saldırı Tespit Sistemlerinin Karşılaştırılması

İstemci Tabanlı ve Ağ Tabanlı Saldırı Tespit Sistemlerinin fonksiyonlarla karşılaştırılması Tablo 2.5 'de verilmiştir.

Tablo 2.5: İstemci ve Ağ Tabanlı Saldırı Tespit Sistemlerinin Karşılaştırılması.

Fonksiyon	**İstemci Tabanlı STS**	**Ağ Tabanlı STS**
LAN açık koruma	Evet	Evet
LAN kapalı koruma	Evet	Hayır
Çok yönlülük	Çok	Az
Fiyat	Daha uygun	Yüksek
Paket reddi	Hayır	Evet
Başarısızlık oranı	Düşük	Yüksek
Yerel makine kayıt tarama	Evet	Hayır
Alarm fonksiyonu	Evet	Evet
Çapraz platform uyumu	Az	Çok
Tespit edilebilen atak türleri	- Şifreli ağ trafiği -Çalıştırılabilir login üzerine yazma vb.	- SYN atakları - Land, Smurf, Teardrop saldırıları vb.

2.4.2.4. Kural Tabanlı (Rule-Based) Saldırı Tespit Sistemleri

Kural Tabanlı STS 'ler, zararlı olduğu bilinen paket veya verilerin ağ üzerinde tespit edilmesi için kullanılırlar. Eğer ağın davranışları çok iyi biliniyorsa, çeşitli kurallar tanımlamak çok kolay bir işlem olacaktır. Bilinen atakların tespiti için kuralların tanımlanması ile bu atakların çeşitli imzaları oluşturulur ve kaydedilir.. Kural Tabanlı STS, belirlenmiş olan bu imzaları kullanarak ağ trafiğini analiz eder ve kayıtlı imzalar ile karşılaştırarak saldırıları tespit etmiş olur. Örneğin:

- Linux web sunucusu için "/etc/passwd" adresine bir HTTP isteği.
- Bilinen kötü amaçlı bir yazılım türü olan "morw.exe" adlı ekli dosya içeren bir e-posta.

Kural Tabanlı STS 'lerin en büyük dezavantajlarından birisi, sadece bilinen saldırıları tespit etmeleri, her saldırı için bir imzanın oluşturulması ve yeni saldırıların tespit edilememesidir. Genellikle düzenli ifadelere ve karakter eşleştirmelerine dayandıkları için "False Positive (FP)" oranları yüksektir. Bu iki yöntemde ağ üzerinde iletilen paketlerin dizelerine bakarlar. İmzalar, sabit bir davranış paterni olan saldırılara karşı iyi çalışırlarken, kendi kendini değiştiren davranışsal özelliklere sahip saldırı modellerine karşı iyi bir şekilde çalışmazlar. Saldırıların şifreli kanallardan saldırması imzanın tespitini karmaşıklaştırır. İmza tabanlı STS 'lerin başarılı olabilmesi için güncel bir imza

listesine sahip olmaları gerekmektedir. Analizin kalitesi, imza tabanının kalitesi ile doğru orantılı olacaktır. Bilinmeyen veya yeni saldırılara karşı sistemi koruyamazlar. Bilinen en popüler örneği SNORT Saldırı Tespit Sistemidir.

Snort Saldırı Tespit Sistemi

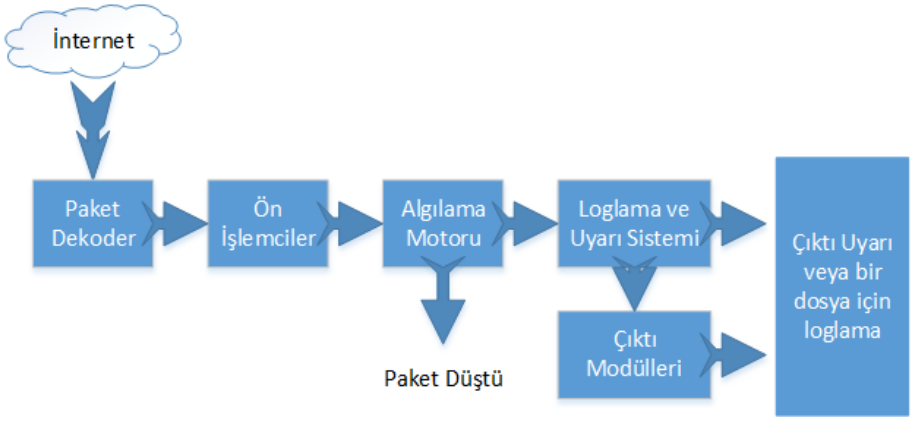
Snort, Marty Roesch tarafından yazılmış, en yaygın olarak kullanılan küçük, hafif bir açık kaynaklı STS 'dir. Gerçek zamanlı trafik analizi yapabilir. Snort, IP ağlarında paket loglayabilen ve gerçek zamanlı trafik analizi gerçekleştirebilen ücretsiz, açık kaynaklı bir Ağ Saldırı Önleme ve Saldırı Tespit Sistemi 'dir. Snort protokol analizi ve içerik arama/eşleme gerçekleştirmenin yanı sıra yaygın olarak tampon taşmaları, gizli port taramaları, web uygulama saldırıları ve işletim sistemi parmak izi teşebbüsleri gibi çeşitli saldırıları aktif olarak engellemek veya pasif olarak tespit etmek için kullanılır.

Özellikleri

- 1300'den fazla farklı saldırı türünü tespit edebilen engine kapasitesi.
- Paketleri okunabilir binary, XML ve diğer formatlarda kaydedebilme.
- Libpcap 26 paket yakalama kütüphanesini kullanan ağ tabanlı bir STS.
- Arabellek taşmaları, gizli port taramaları, CGI taramaları, SMB probları, NetBIOS sorguları, DDoS saldırıları, Truva atı saldırıları bazı virüs ve solucan türleri dahil olmak üzere tehditler için model eşleştirmesine dayalı olarak uyarı verme yeteneği.
- Linux, windows ve diğer işletim sistemleri üzerinde derlenip çalıştırılabilme özelliği.

Bileşenleri

Snort, mantıksal olarak çoklu bileşenlere ayrılmıştır. Bu bileşenler belirli saldırıları tespit etmek ve algılama sisteminden istenen bir formatta çıktı üretmek için birlikte çalışır. Snort tabanlı bir STS aşağıdaki ana bileşenlerden oluşur (Şekil 2.11).



Şekil 2.11: Snort bileşenleri.

- Paket Çözümleyici: Farklı tipteki ağ arayüzlerinden paketleri alır ve paketleri önişleme veya algılama motoruna gönderilmesi için hazırlar.
- Önişleyiciler: Paketin bir saldırgan tarafından kullanıldığını öğrenmek için algılama motorunun bazı işlemleri yapmadan önce Snort'un veri paketlerini düzenlemesine veya değiştirmesine yardımcı olur. Önişlemciler; herhangi bir STS 'nin, veri paketlerinin algılama motoruna gönderilmesinden önce hazırlanmalarını sağladıklarından çok önemlidir. Saldırganlar, STS 'yi farklı şekillerde kandırmak için farklı teknikler kullanırlar. Örneğin, HTTP paketlerinde "autorun / admin" imzasını bulmak için bir kural oluşturulmuş olsun. Bu dizede küçük değişiklikler yapan bir bilgisayar korsanı tarafından sistem kolayca kandırılabilir. Örneğin:
  - "autorun/./admin".
  - "artorun/,/admin"
- Algılama Motoru: Paket içinde herhangi bir saldırı etkinliğinin olup olmadığının tespiti Algılama Motorunun sorumluluğundadır. Algılama altyapısı, bu amaç için Snort kurallarını kullanır. Kurallar, eşleştirilmelerin ve kontrol işlemlerinin yapıldığı belirli bir veritabanından okunur. Bir paket herhangi bir kuralla eşleşirse, uygun önlem alınır aksi halde paket düşer. Önlemler, paketi kaydediyor veya uyarı oluşturuyor olabilir. Algılama motoru, Snort'un kritik-zamanlı çalışan bir parçasıdır. Algılama Motorundaki yük aşağıdaki faktörlere bağlıdır:
  - Snort Motorundaki tanımlanan kuralların sayısı.
  - Snort'un çalıştığı makinenin gücü.

  - Snort makinesinde kullanılan veri aktarım hızı.
  - Ağ üzerinde çalışan sistemlerin sayısı.

- Loglama ve uyarı verme sistemi: Algılama Engine'in bir paketin içerisinde tespit ettği olaya bağlı olarak, ilgili paket ya log dosyasına kaydedilir ya da duruma göre bir uyarı sistem tarafından oluşturulur. Bu durum, tamamen paketin içeriği ile ilgilidir. Loglar, metin(text) dosyalarında, tcpdump tarzı dosyalarda veya başka bir basit formada kaydedilir. Tüm log dosyaları varsayılan olarak /var/log/snort klasörü altında saklanır.

- Çıktı modülleri: Temel olarak bu modüller, kayıt ve uyarı sisteminin oluşturduğu çıktı türünü kontrol ederler. Çıktı modülleri veya eklentileri, Snort'un kayıt ve uyarı sistemi tarafından üretilen çıktıların ne ve nasıl bir şekilde kaydedilmek istendiğine bağlı olarak farklı işlemler yapmak için kullanılırlar.

Aşağıdaki Tablo 2.6 'da bileşenler, işlevleri ve açıklamaları ile birlikte gösterilmiştir.

Tablo 2.6: Snort bileşenleri ve açıklamaları.

Bileşenin Adı	**Açıklaması**
Paket çözümleyici	Algılama engine 'ina paketin iletilmeden önce hazırlanmasını sağlar.
Önişlemci	Paketlerin gerektiğinde tekrar düzenlenebilmesi için gerekli işlemleri yapar.
Algılama Engine	Tanımlanan kurallara göre paketlerin içerisindeki saldırıları tespit eder.
Loglama ve Uyarı Sistemi	Saldırı tespit edildiğinde uyarı göndermek ve loglama için kullanılır.
Çıktı Modülü	İşlermlerin, logların ve uyarıların önceden belirlenmiş formatta çıktılarını oluşturur ve saklar.

Snort biden fazla modda analiz işlemini gerçekleştirebilir:

- Sniffer modu, ağdaki paketleri okur ve bunları konsolda (ekran) sürekli bir akış şeklinde gösterir.

- Paket Loglayıcı modu; paketleri diske kaydeder.
- Ağ Saldırı Tespit Sistemi modu; Snort'un kullanıcı tanımlı bir kural setine bağlı olarak ağ trafiğini analiz etmesi ve çeşitli eylemler gerçekleştirmesini sağlayan en karmaşık ve yapılandırılabilir modudur.
- Satıriçi modu; Snort üzerinde sıralı özel kuralların kullanıldığı bazı paketlerin geçişine izin verilip bazılarına izin verilmeyen moddur.

Snort Uyarı Sistemi

-s anahtarı syslog 'a uyarıları göndermek için kullanılır. LOG AUTHPRIV ve LOG ALERT syslog uyarı mekanizması için varsayılan iki özelliktir. Syslog uyarı mekanizmasının çıktılarının çeşitli düzenlemeler yapılması istendiği takdirde, çıktı eklenti yönergelerinin (output plug-in directives) kural dosyalarında belirlenmesi gerekir. Örneğin, syslog 'a alarm göndermek ve varsayılan özellikleri loglamak için /snort -c snort.conf -l. /log -h 192.168.1.0/24 –s. komut satırı kullanılır. Bir diğer örnek olarak /var/log/snort dizininde varsayılan özellikleri loglamak ve hızlı bir alarm dosyasına alarmlar göndermek için ./snort -c snort.conf -A fast -h 192.168.1.0/24 komutu kullanılır. Snort bir uyarı mesajı oluşturduğunda, genellikle aşağıdaki gibi görünür: [**] [120: 56: 1] (snort_decoder): T / TCP Algılandı [**]

- İlk sayı, Snort 'un hangi bileşeninin bu alarmı ürettiğini gösteren üretici Tanımlama Numarası(ID) 'dır. Yukarıdaki örnek için tanımlama numarası 120 'dir ve bu sayı, örnek olayın Snort' un 120. bileşeninden geldiğini göstermektedir.
- İkinci sayı, imza ID 'si olarak da adlandırılan Snort ID 'sidir. Önişlemci SID'lerinin listesi etc/gen-msg.map adresinde bulunmaktadır. Kural tabanlı SID 'ler, sid seçeneği ile doğrudan kurallara yazılabilirler. Bu durumda, 56 bir T/TCP olayını temsil eder.
- Üçüncü sayı, revizyon ID'sidir. Bu sayı öncelikle imza yazarken kullanılır; çünkü kuralın her bir yorumu, bu sayıyı rev seçeneğiyle arttırmalıdır.

Ağ Saldırı Tespit modunda Snort çıkışını yapılandırmanın çeşitli yolları vardır. Toplam 7

uyarı modu olan Snort ’un bu modunun, -A seçeneği ile kullanılan altı modu aşağıdaki Tablo 2.7 ’de listelenmiştir.

Tablo 2.7: Snort uyarı modları.

Mod	Açıklama
-A fast	Hızlı uyarı modudur. Timestamp, alert message, source ve destination IPs/ports ’lar ile basit formatta bir uyarı yazar.
-A full	Tam uyarı modudur. Varsayılan olarak, herhangi bir mod belirtilmediğinde otomatik olarak kullanılır.
-A unsock	Başka bir programın dinleyebileceği bir UNIX soketine uyarı gönderir.
-A none	Uyarı vermeyi kapatır.
-A console	Konsola “fast-style” uyarılar gönderir.
-A cmg	“cmg style” uyarıları üretir.

Snort ’un hızlı modda analiz etmesi gerektiği durumda (1000 Mbps’lik bir bağlantıya sahip olmak gibi), birleştirilmiş log kaydını ve birleşik bir log okuyucu kullanılması gerekecektir. Bu durum, başka bir programın, veritabanına veri girişi yapmak gibi yavaş eylemler gerçekleştirdiği sırada, Snort’un uyarıları, ikili bir formda mümkün olduğunca hızlı bir şekilde kaydetmesine olanak sağlar. Kolayca çözümlenebilir olan ancak yine de biraz hızlı bir metin dosyası isteniyorsa, “hızlı” çıkış mekanizmasıyla ikili kayıt kullanmak denenebilir. Bu paketleri tcpdump formatında kaydedecek ve minimum uyarı üretecektir. Örneğin: ”./ snort -b - A Fast –c snort.conf”.

Snort Kuralları yazma

Snort esnek, oldukça güçlü basit ve hafif bir kural tanımlama dili kullanır. Snort kurallarını geliştirirken hatırlanması gereken bir dizi basit yönerge vardır. Çoğu Snort kuralları tek bir satırda yazılır. Bu durum 1.8’den önceki sürümlerde olması gereken bir zorunluluktu. Snort’un şu anki sürümlerinde, kurallar satırın sonuna bir ters eğik çizgi ekleyerek birden çok satıra yayılabilir.

Snort kuralları, iki mantıksal bölüme; kural başlığı ve kural seçeneklerine ayrılır:

- Kural Başlığı: Kuralın eylemini, protokolünü, kaynak ve hedef IP adreslerini, ağ maskelerini, kaynak ve hedef bağlantı noktası bilgilerini içerir.
- Kural Seçeneği: Bu bölüm, kural mesajının alınmasının gerekip gerekmediğini belirlemek için paketin hangi bölümlerinin denetlenmesi gerektiğine dair uyarı mesajları ve bilgileri içermektedir.

Örnek bir Snort kuralı şöyledir: Alert tcp any any -> 192.168.84.128/24 111 (content:”|00 01 70 b1|”; msg: “Accessing the device”;)

İlk paranteze kadar olan metin kural başlığı ve parantez içine alınmış bölüm ise kural seçeneğini ifade eder. Kural seçenekleri bölümündeki iki noktadan önceki kelimeler seçeneğin anahtar kelimeleri olarak adlandırılır. Bu kuraldaki tüm elemanlar, belirtilen kural eylemi için geçerli olmalıdır. Elemanlar, mantıksal VE(AND) durumu olarak düşünülebilir. Aynı zamanda, bir Snort kütüphanesindeki çeşitli kurallar, büyük bir mantıksal VEYA(OR) olarak düşünülebilir.

- kural başlığı, bir paket için “kim”, “nerede” ve “ne” sorularının cevaplarını tanımlayan bilgileri ve kuralda belirtilen tüm özelliklere sahip bir paketin görünmesi durumunda ne yapılacağını içerir. Kuraldaki ilk öğe kural eylemidir. Kural eylemi, Snort ’a kural ölçütleriyle eşleşen bir paket bulduğunda ne yapması gerektiğini bildirir. Snort ’un, alert, log, pass, activate ve dynamic olmak üzere 5 adet kullanılabilir varsayılan eylemi vardır. Ayrıca, inline modundaki bir Snort için bırakma, reddetme ve geri saymayı içeren ek seçenekler bulunur.
  - alert -belirtilen alarm modunda alarm üretir ve ilgili paketi loglar.
  - log - paket loglar.
  - pass - paketi görmezden elir.
  - dynamic - bir activate kuralı tarafından aktif hale getirilinceye dek kullanılmaz ve sonra bir log kuralı gibi davranır.
  - activate - alarm verir ve başka bir dinamik kural etkinleştirir.
  - drop - iptables oluşturur, paketi düşürür ve loglar.

  - reject - iptables oluşturur, paketi düşürür, loglar ve eğer TCP protokolü ise TCP reset gönderir.
  - sdrop - iptables oluşturur, paketi düşürür ama loglamaz.

  Bu kuralların haricinde, başka kurallar kullanıcı tarafından tanımlanabilir ve bu kurallar ile bir veya daha fazla çıktı eklentisi yapılandırılabilir.

- Protokol: Kural başlığından sonraki konu ismi protokoldür. Snort 'un şüpheli davranışların analizi için hali hazırda dört adet protokolü mevcuttur. Bunlar, TCP, UDP, ICMP ve IP protokolleridir.

- IP Adresleri: Herhangi bir anahtar kelime herhangi bir adres tanımlamak için kullanılabilir. Snort, kural dosyasındaki IP adresi alanlarına ana bilgisayar adı araması sağlayan bir mekanizmaya sahip değildir. Adresler düz bir sayısal IP adresi ve bir CIDR bloğu tarafından oluşturulur. CIDR bloğu, kuralın adresine ve kurala karşı test edilen tüm gelen paketlere uygulanması gereken ağ maskesini gösterir. /24 olan bir CIDR blok maskesi C Sınıfı ağını, /16 B sınıfı ağını ve /32 belirli bir makine adresini gösterir. Örneğin, /CIDR bileşimi 192.168.1.0/24 adresi, 192.168.1.1'den 192.168.1.255'e kadar adres bloklarını belirtir. Bu atama için kullanılan hedef kuralın, söz konusu aralıktaki herhangi bir adresle eşleşeceği anlamına gelir. IP adreslerine uygulanabilecek bir olumsuzlama operatörü vardır. Bu operatör, Snort'a listelenen IP adresiyle belirtilenler hariç herhangi bir IP adresini eşleştirmesini söyler. Olumsuzlama kullanıcı operatörü bir "!"ile gösterilir.

- Port Numaraları: Port numaraları, herhangi bir port tanımlama, statik port tanımlamaları, aralıklar ve olumsuzlama gibi çeşitli yollarla belirtilebilir. Herhangi bir port, tam anlamıyla herhangi bir port olan joker karakterdir. Statik portlar, portmapper için 111, telnet için 23, http için 80 gibi tek bir port numarası ile gösterilirler. Port aralıkları, aralık operatörü ile belirlenmektedir. Aralık operatörü, faklı anlamlar için çeşitli yollarla kullanılabilmektedirler. Örneğin, 192.168.1.0/24 1: 1024 şeklindeki bir tcp logu, herhangi bir porttan gelen günlük trafiğin, hedef portlarının 1 ile 1024 arasında değiştiğini ifade etmektedir.

- Kural Seçenekleri: Kural seçenekleri, Snort 'un saldırı tespit engine ninin temel yapısını oluşturur ve kullanım kolaylığını güç ve esneklik ile birleştirir. Tüm Snort kural seçenekleri, noktalı virgül (;) karakterini kullanarak birbirinden ayrılır. Kural

seçeneği anahtar kelimeleri bağımsız değişkenlerinden (:) karakteri kullanılarak ayrılır. Dört ana kural seçeneği kategorisi vardır:

- Genel: Bu seçenek, kural hakkında bilgi sağlar, ancak algılama sırasında herhangi bir etkisi yoktur.
- Yük(Payload)Yük: Paketlerdeki yükü arar ve birbirleri ile ilişkilendirebilir.
- Yüksüz(NON-payload): Paketin yüksüzlüğünü inceler.
- Tespit Sonrası(Post-detection): Bir kuralın "tetiklendikten" sonra gerçekleşecek kuralların özel tetikleyicileridirler.

Tablo 2.8: Snort 'un genel kural seçeneği anahtar kelimeleri.

Anahtar Kelime	Açıklama
Msg	Mesajı yazdırma, loglama ve uyarı verme
Reference	Harici saldırı tespit sistemlerine referansların dahil edilmesine izin verir.
Gid	Belirli bir kural tetiklendiğinde snort'un hangi bölümünün uyarı oluşturduğunu belirtmek için kullanılır.
Sid	Snort kurallarını benzersiz bir şekilde tanımlamak için kullanılır.
Rev	Snort kuralları düzenlemelerini benzersiz bir şekilde tanımlamak için kullanılır.
Classtype	Tespit edilen bir atağın sınıflandırılması için kullanılan bir kuraldır.
Priority	Kurallara öncelik vermek için kullanılır.
Metadata	Kural ile beraber veya kural hakkında bazı ek bilgiler ekleme veya kuralın içine gömmeye olanak tanır.

2.4.2.5. Anomali Tabanlı Saldırı Tespit Sistemleri

Anomali Tabanlı Saldırı Tespit yaklaşımı, kural tabanlı yaklaşımların aksine, bilinmeyen veya yeni türdeki atakların tespiti için sistemdeki anomali durumlarının tespitine dayanır. Anomali; bir sistemin normal davranışından belirgin bir şekilde farklı bir şekilde sapmalara verilen isimdir. Dolayısı ile bu yöntem, bilinen atak türleri için yüksek bir doğruluk oranı sunmasada, yeni ve farklı türdeki atakların tespitinde büyük bir başarı ile saldırı tespiti sağlarlar. Sistemde; CPU, disk aktiviteleri, kullanıcı girişleri ve

dosya aktiviteleri gibi rutin işlemlerin yer aldığı türden işlemlerde anomali durumların tespiti için kullanılır. Bu yaklaşımın sunduğu fayda ise, anomalinin arkasında yatan sebepleri anlamak için zaman harcamadan da anomalileri tespit edebilmesidir. Anomali tespiti, beklenen davranıştan, beklenmeyen şekildeki sapmaların tanımlanması işlemidir. Beklenen davranışlar, manuel olarak bir profilin geliştirilmesi ile veya bir profilden otomatik olarak geliştirilerek tanımlanabilir. Otomatik olarak bir profilin geliştirilmesi işlemi, ilgili bir davranışın geçerli örneklerinin istatistiksel formlarda ve belirli bir zaman aralığındaki sistem davranışlarının karakteristik özelliklerinin bir yazılım vasıtası ile toplanması ve işlenmesi ile oluşturulur. Unutulmamalıdır ki, beklenmeyen davranışın bir atak olması gerekmez; bu davranış, beklenen davranış kategorisine eklenmesi gereken, yeni ve yasal bir davranışta olabilir. Anormali tespit motorundaki olayların kaynağı, önceden tanımlanmış veya kabul edilen davranış modeli dışında kalan davranışlardır. Belirli davranışlar için normal işlem büyüklüğü yüzde 49 iken, aynı işlemin anomali durumunda CPU kullanım oranı yüzde 90 'lara kadar çıkabilmektedir.

Anomali tespit motorlarının bir diğer dezavantajı ise, kural tanımlama işleminin karmaşık olmasıdır. Bunun yanı sıra, analiz edilen her protol için tanımlanmalı, uygulanmalı ve doğruluğu test edilmelidir. Ayrıca kural geliştirme sürecinde, farklı protokollerdeki uygulama farklılıklarınında bir araya getirilmesi gerekmektedir. Ağdaki özel protokollerin analiz edilmesi büyük bir emek ve çaba gerektirmektedir. Diğer yandan, bir protokol oluşturulduğunda ve bir davranış tanımlandığında, imza tabanlı modelden daha hızlı ve kolay bir şekilde ölçeklenebilmektedir. Çünkü her saldırı ve potansiyel varyasyonları için yeni bir imza oluşturulması gerekmez. Öte yandan, normal davranış paterni içerisindeki kötü amaçlı aktivitelerin tespit edilememe durumu söz konusu olabilmektedir.

2.4.2.6. Hibrit Saldırı Tespit Sistemleri

Bütün Saldırı Tespit Sistemleri 'nin bazı zayıflıkları bulunmaktadır. Bu nedenle, Hibrit Saldırı Tespit Sistemleri, çeşitli STS 'lerin özelliklerinin biraraya getirilmesi ile geliştirilmiştir. Hibrit STS 'ler, çeşitli STS 'lerin zayıflıklarının giderilmesini ve tespit yönteminin güçlü yönlerini birleştirmeyi amaçlamaktadır. Örneğin kural tabanlı sistemlerin eksik kaldığı, yeni ve güncel atakların tespiti noktasında anomali tabanlı tespit yöntemleri kullanılarak bu ataklarında tespit ediebileceği gelişmiş bir STS tasarlanabilir. Anomali tabanlı sistemler bazı sistemlerde yavaşlamalara sebep

olabileceğinden bazı kısımlarda pratiklik açısından kural tabanlı sistemler ile hibrit bir sistem geliştirilebilir. Hibrit Saldırı Tespit Sistemleri kullanılan platforma ve amaca göre tasarlanabilmektedirler. Böylece ihtiyaç doğrultusunda optimum çözüm elde edilmeye çalışılır.

2.4.2.7. Saldırı Tespit Sistemlerindeki Kısıtlamalar

STS 'ler, kullanıldıkları ağları, çeşitli saldırı ve tehlike arz eden davranışlardan korusa da, bu sistemlerin aşağıda belirtilen, belirli kısıtlamaları bulunmaktadır:

- Zayıf bir oturum açma ve tanımlama mekanizması için alternatif değillerdir.
- İnsan müdahalesi olmaksızın, izleme ya da saldırı incelemesi yapılamayabilir.
- Sistemin sunduğu bütünlük ve bilgi kalitesi ile ilgili bir sorunla baş edememektedirler.
- Yoğun bir ağdaki tüm trafik analiz edilemeyebilinir.
- Genellikle paket düzeyinde saldırılarla ilgili sorunlarla baş edemezler.
- STS, gerçekleşen bir saldırıyı tespit edemediğinde "False Negative" oluşur.
- STS, saldırı olmaksızın bir saldırı varmış gibi davranırsa "False Positive" oluşur.

2.4.3. Saldırı Önleme Sistemleri

Saldırı Önleme Sistemleri (SÖS) en basit ifade ile güvenlik duvarlarının engelleme özelliklerinin STS 'lerin paket denetimi özelliği ile birleştirilmesi ile geliştirilmişlerdir. SÖS, donanım ve yazılımın bir birleşimi veya tamamen ağ trafiğini izleyen ve gerçek zamanlı olarak, izinsiz giriş saldırılarını engelleyen bir donanım üzerinde kurulu yazılımdır. Bu sistemler, uyarı vermekten ziyade saldırının sisteme herhangi bir zarar vermesinden önce normal ağ trafiğinde kötü amaçlı paketlerin tespit edilmesi ve saldırıların otomatik olarak durdurulması için tasarlanan, proaktif savunma mekanizmalarıdır. Saldırı Önleme, saldırı tespitini gerçekleştirme ve tespit edilen olayları gerçek zamanlı olarak durdurma işlemidir. Saldırı tespiti gibi, olay hakkında bilgiler kaydedilir ve güvenlik uzmanına bildirilir. STS 'ler ile SÖS 'ler arasındaki temel fark,

STS ’ler yalnızca olayları izleyip gerekli durumlarda alarm veriyorlarken, SÖS ’ler ise saldırıları durdurmaya çalışırlar.

3. MALZEME VE YÖNTEM

Bu bölümde, sırasıyla tezin gerçeklenmesinde kullanılan veri seti ve makine öğrenmesi yöntemlerine değinilmiştir.

3.1. SALDIRI TESPİT DEĞERLENDİRME VERİ SETİ (CICIDS2017)

Saldırı Tespit Sistemleri ve Saldırı Önleme Sistemleri, karmaşık ve sürekli büyüyen ağ saldırılarına karşı en önemli savunma araçlarıdır. Güvenilir test ve doğrulama veri setlerinin eksikliğinden dolayı, özellikle anomali tabanlı saldırı tespit yaklaşımlarının tutarlı ve doğru performans değerlendirmelerinin yapılması güç bir hal almıştır.

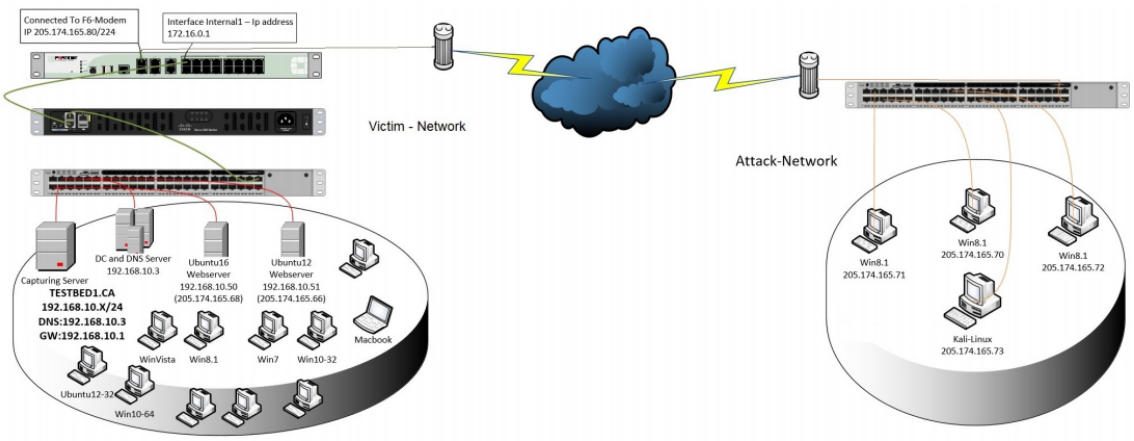
Değerlendirmelerimiz, 1998 yılından itibaren mevcut olan veri setlerinin güncel olmamalarının yanı sıra, kullanılmalarının da güvenilir olmadığını ortaya çıkarmıştır. Bahse mevzu bu veri setlerinin bir kısmı trafiğin çeşidinden yoksunken, diğerleri de paket yükü verilerini, mevcut eğilimleri yansıtmayacak şekilde anonimleştirmişlerdir. Bazıları da öznitelik kümesinden ve metadata 'dan yoksunlardır.

CICIDS2017 [36] veri seti, gerçek dünya verilerini (PCAP) yansıtan iyi huylu ve en güncel yaygın saldırıları içermektedir. Ayrıca, time stamp, kaynak ve hedef IP 'lere, kaynak ve hedef bağlantı noktalarına, protokollere ve ataklara (CSV dosyaları) dayalı olarak etiketli akışlarla CICFlowMeter kullanan ağ trafiği analizinin sonuçlarını da içermektedir.

Bu veri kümesi oluşturulurken, gerçekçi bir arka plan trafiği oluşturmak, en önemli önceliklerden birisi olmuştur. İnsan etkileşimlerinin soyut davranışını gözlemlemek ve doğal zararsız normal bir arka plan trafiği oluşturmak için, Gharib ve arkadaşlarının önermiş olduğu B-Profil 'i sistemini kullanmışlardır [37]. Bu veri seti için, HTTP, HTTPS, FTP, SSH ve e-posta protokollerine dayalı 25 kullanıcının soyut davranışı oluşturulmuştur.

Veriseti oluşturulurken, Şekil 3.1' de görüldüğü gibi tamamen birbirinden ayrı iki ağ,

Saldırı-Ağı ve Kurban-Ağı olmak üzere bir test ortamı tasarlanmış ve geliştirilmiştir.



Şekil 3.1: Test ortamı altyapısı.

Şekil 3.1 'de, Kurban-Ağı 'nın, güvenlik duvarı, anahtalar, yönlendiriciler ve kişisel bilgisayarları içeren yüksek bir güvenlik mimarisinden oluştuğunu; diğer taraftan Saldırı-Ağı 'nın ise bir router, switch ve çeşitli atak senaryosunu çalıştırabilecek, farklı işletim sistemlerine sahip birkaç bilgisayardan oluştuğunu göstermektedir.

Veri toplama dönemi, toplam 5 gün sürmüş ve 3 Temmuz 2017 Pazartesi günü saat 09: 00 'da başlayıp, 7 Temmuz 2017 Cuma günü saat 17: 00 'da sona ermiştir. Pazartesi normal gündür ve sadece iyi huylu trafiği içerir. Uygulanan saldırılar, Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet ve DDoS saldırılarını içermektedirler. Salı, Çarşamba, Perşembe ve Cuma günleri sabah ve öğleden sonra uygulanmışlardır.

Gharib ve arkadaşları [37], önermiş oldukları veri seti değerlendirme framework 'ünde, güvenilir bir ölçüt olması düşüncesiyle, veri kümesi oluşturmak için gerekli, onbir kriter belirlemişlerdir. Daha önceki STS veri kümelerinin hiçbirisinin bu 11 kriterin tümünü kapsayamadığını ileri sürmüşlerdir. CICIDS2017 veri seti bu kriterlerin tamamı göz önünde bulundurularak hazırlanmış bir veri seti olma özelliği taşımaktadır. İlgili kriterlerin kısa özetleri aşağıda ifade edildiği gibidir:

- Bütün bir ağ yapılandırması: Bütün bir ağ topolojisi olarak; Modem, Güvenlik Duvarı, Anahtarlar, Yönlendiriciler ve Windows, Ubuntu ve Mac OS X gibi çeşitli işletim sistemlerinin bir arada bulunduğu ağ yapılandırmalarıdır.

- Bütün bir trafik: Bir kaynaktan bir hedefe gelen ve giden paket akışıdır.
- Etiketli Veri Kümesi: Veri setinin etiketli olması.
- Tam etkileşim: Değerlendirmelerin doğru yorumlanması için, hayati özelliklerden biri, anomali davranışlar için mevcut bilgi miktarıdır. Bu nedenle, LAN 'lar arasındaki veya içindeki gibi bütün ağ etkileşimlerine sahip olmak, iyi bir veri seti için en önemli gereksinimlerden birisidir.
- Tam Yakalama: Tam bir trafik veri setinde bile, önerilen tespit sistemlerini değerlendirmek isteyen araştırmacıların tüm trafiği yakalaması çok önemlidir. Bazı veri setlerinin trafiği kısmen yakaladığı gözlemlenmiş ve işlevsel olmayan veya etiketlenmemiş trafiğin bazı kısımları veri setinden çıkarılmıştır, Oysaki tüm trafiğin bir saldırı tespit sisteminin false-positive yüzdesini hesaplamak için oldukça etkili olduğu görülmektedir.
- Erişilebilir Protokoller: Veri aktarımının düzensiz bir şekli olan ve HTTP, FTP gibi bazı protokolleri kapsayabilen Bursty traffic gibi STS sistemini test etmek için hayati önem taşıyan birçok farklı trafik türü vardır. Etkileşimli trafik, kullanıcılarla gerçek zamanlı etkileşim içeren uygulamalar (ör. Web 'de tarama, çevrimiçi satın alma vb.) gibi kısa istek ve yanıt çiftlerinden oluşan oturumları içerir. Kullanıcının VoIP ve Video konferans gibi verilerinin zamanında teslim edilmesinin beklendiği, gecikmeye duyarlı trafiklerin yanında, zamanında teslimatın önemli olmadığı haber ve posta trafiği gibi gerçek zamanlı olmayan trafikler bulunmaktadır. Tam bir veri kümesinin hem normal hem de anormal trafiği olmalıdır.
- Saldırı Çeşitliliği: Son yıllarda, tehditler kapsamlarını uygulamalar ve uygulama saldırıları gibi karmaşık senaryolar şeklinde genişlettiler. Saldırı türleri her gün değişiyor ve güncelleniyor. Bu nedenle, STS ve SÖS sistemlerini yeni saldırılar ve tehdit senaryoları ile test etme ve analiz etme becerisine sahip olmak, çevrimdışı bir veri setinin desteklemesi gereken en önemli gereksinimlerden birisidir. Çalışmada saldırılar, 2016 McAfee raporuna dayanılarak Brute force, DoS, Scan veya enumeration, Backdoors, DNS ve diğer saldırılar (örn. Heartbleed, Shellshock ve Apple SSL kütüphane hatası) olmak üzere yedi ana gruba kategorize edilmiştir.
- Anonimlik: Hem IP hem de payload erişilebilir olduğunda gizlilik sorunları ortaya çıkar. Bu nedenle, veri kümelerinin çoğu, özellikle derin paket incelemesi (DPI)

gibi bazı tespit mekanizmaları için veri setinin kullanışlılığını azaltan payload larını tamamen kaldırmışlardır.

- Heterojenlik: STS alanında,veri kümesi oluşturmak için ağ trafiği, işletim sistemi logları veya ağ trafiği logları gibi farklı kaynaklara sahip veri setleri oluşturulmalıdır. Bir kaynağa sahip homojen bir veri kümesi, belirli bir tipteki tespit sistemlerinin analizi için faydalı olabilirken, heterojen veri kümesi, algılama sürecinin tüm yönlerini kapsayan eksiksiz bir test için kullanılabilir.

- Öznitelik kümesi: Veri kümesi sağlamanın temel amacı, diğer araştırmacıların önerilen sistemlerini test etmesi ve analiz etmesidir. Temel zorluklardan biri, ilgili özelliklerin nasıl hesaplanacağı ve analiz edileceğidir. Öznitelik çıkarma uygulamalarını kullanan trafik veya loglar gibi farklı türdeki veri kaynaklarından özellikler elde edilmelidir.

- Metadata: Uygun dokümantasyonun eksikliği, bu alandaki mevcut veri setlerindeki temel eksikliklerden biridir. Veri setlerinin çoğunun, tamamlanmamış olsalar dahi, dokümantasyonları yoktur. Ağ yapılandırması, saldırganın ve mağdurun makinelerinin işletim sistemleri, saldırı senaryoları ve diğer kritik öneme sahip bilgiler hakkındaki mevcut yetersiz bilgi, araştırmacılara yönelik, veri setinin kullanılabilirliği ile azaltılabilir.

3.2. MAKİNE ÖĞRENMESİ YÖNTEMLERİ

Makine Öğrenmesi, mevcut veriden matematiksel ve istatistiksel yöntemler kullanarak çeşitli çıkarımlar yapan metotlardan oluşmaktadır. Makine Öğrenmesi, çok kapsamlı bir kullanım alanı olmasına ek olarak, yapay sinir ağları (YSA), naive bayes, derin öğrenme, KNN, k means, destek vektör makineleri (DVM), karar ağaçları, genetik algoritmalar gibi farklı pek çok öğrenme yöntemini içerisinde barındırmaktadır. Genellikle tahmin, çıkarım ve sınıflandırma için kullanılırlar. Gözetimsiz, yarı gözetimli ve gözetimli yöntemler olmak üzere 3 gruba ayrılırlar. Gözetimli öğrenmede, kullanılan algoritmaya bağlı olarak, model oluşturmak için eğitim verileri kullanılır ve daha sonra eğitim verisi içerisinde yer almayan test verileri ile modelin başarısı test edilir. Diğer taraftan, gözetimsiz öğrenme tekniklerinde modeli eğitmek için bir etiketli bir eğitim verisi bulunmaz ve modelin kendi kendine öğrenmesi beklenir. Bu sebeple, genellikle gözetimsiz öğrenme tekniklerinin

doğruluk oranı, gözetimli öğrenme yöntemlerine göre daha düşüktür.

Makine öğrenmesinde, değişken seçimi, özellik seçimi ve değişken alt kümesi seçimi olarakta bilinen özniteliklerin seçimi, öğrenme modelinin oluşturulması için tüm özeniteliklerden en çok işe yarar olanlarının seçilmesi işlemidir. Özniteliklerin seçilmesi genellikle, araştırmacıların veya kullanıcıların yorumlamalarını kolaylaştırmak için modellerin basitleştirilmesinde, eğitim sürelerinin kısaltılmasında, gereksiz boyut fazlalığından kaçınılmada ve uyumun azaltılarak genellemenin sağlanmasında kullanılır.

Veriler, çoğunlukla gereksiz veya ilgili olmayan birçok öznitelik içerir. Özniteliklerin seçimi ile bu ilgisiz ve gereksiz olan özellikler veri kaydından büyük bir bilgi kaybı olmadan çıkarılmış olur. İlgisiz ve gereksiz kavramları birbirlerinden farklı kavramlardır. Çünkü ilgili bir özellik, güçlü bir şekilde ilişkili olduğu başka bir özniteliğin yanında gereksiz olabilir.

Yüksek boyutlu veriler sınıflandırma için iyi değillerdir. Verilerin işlenmesi için gereken zaman ve alan karmaşıklığını arttırırlar. Dahası, gereksiz veya ilgisiz özniteliklerin varlığı, öğrenme yöntemlerinin verimli bir şekilde çalışamamalarına yol açabilir. Bu problemin en yaygın olarak kullanılan çözümü ise özniteliklerin seçilmesidir. Özniteliklerin seçilmesi, girdi öznitelik kümesinden özniteliklerin alt kümelerinin belirlenmesi işlemi ile gerçeklenir ve böylece boyutta azaltılmış olur. Genellikle hesaplama maliyetini azaltmak için ve yüksek boyut ile gelen problemlerin çözümü için, ilgisiz ve gereksiz özniteliklerin, öznitelik kümesinden çıkarılması için kullanılır.

Öznitelik seçimi yöntemleri üç ayrı kategoriye ayrılabilmektedirler. Bunlar; filter-based, wrapper-based and embedded yöntemlerdir. Filter-based yöntemler, öznitelikleri öğrenme algoritmalarından önce önişlem olarak, puanlarına göre sıralar ve en yüksek puanlı öznitelikleri seçer. Wrapper-based yöntemler, öğrenme algoritmasını kullanarak öznitelikleri puanlar. Embedded yöntemler ise öznitelik seçimi ile öğrenme algoritmalarını birleştirir.

Filtre tabanlı öznitelik seçimi, genellikle, bazı performans kriterlerini maksimum seviyeye çıkaran özniteliklerin ikili seçimi şeklinde çalışmaktadır. Fisher score, ReliefF, Lablacian score, Hilbert Schmidt Independence Criterion (HSIC), ve Trace Ratio criterion Filter-based öznitlik seçimi yöntemlerinderdirler. Bu yöntemlerin içerisinde genel olarak

iyi performans sergilemesi dolayısı ile en çok tercih edilen yöntemlerden biri ise Fisher Score öznitelik seçimi yöntemidir.

Detaylı olarak incelendiğinde, d özniteliğin bir kümesi, S ile ifade edilsin. Burada filter-based öznitelik seçmenin amacı, bazı F kriterlerini maksimum yapan, T ile ifade edilen, $|\cdot|$ ile kümenin en önemli elemanlarının sayısının olduğu yerde, m < d özniteliklerinin alt kümesinin seçilmesidir.

$$T^* \equiv argmax_{T \subseteq S} F(T), s.t. |T| \equiv m \tag{3.1}$$

Denklem 3.1, bir çeşit birleşimsel optimizasyon problemidir. Denklem için evrensel optimal çözümün bulunması NP hard(non-deterministic polynomial-time hardness) olarak ifade edilir. Bu problemin çözümü için en yaygın yaklaşımlardan biri olan heuristic yaklaşım, ilk olarak F kriterine bağlı olarak, her bir özellik için bir puan hesaplar ve sonra yüksek skorlara sahip en üstteki m özniteliği seçer. Lakin, heuristic algoritma ile seçilen özniteliklerin standartların altında yetersiz olma gibi bir durumları söz konusudur. Diğer yandan, heuristic algoritma her bir özniteliğin skorunu bağımsız olarak tek başına hesapladığı için özniteliklerin bir araya gelerek oluşturdukları kombinasyonlarının değerlendirmesini göz ardı etmiş olur. Örneğin, a ve b iki farklı özniteliğimiz olsun. Bu iki özniteliğin bağımsız olarak değerlendirildiğinde skorları düşük olsun. a ve b özniteliklerinin biraraya gelmesi ile ab ikili kombinasyonun ise skor değeri yüksek olabilir. Bu durumda heuristic yaklaşımlar bu öznitelikleri bireysel ve bağımsız ele aldığı için elimine etmiş olacaklar. Oysaki elimine edilmemeliydi. Diğer taraftan, tam tersi durumda da yine söz konusu durum geçerli olacaktır. Bu ve benzeri birçok senaryo bize öğrenme metotları için öznitelik seçmenin ne denli kritik bir öneme sahip olduğunu ve hassas yapılması gerektiğini göstemektedir.

Bu tez kapsamında, Fisher Score Öznitelik Seçme Algoritması ve KNN, SVM ve DT makine öğrenmesi teknikleri CICIDS2017 veri seti ile birlikte kullanılmıştır. Bu çalışmalara ek olarak ayrıca TBA (Temel Bileşen Analizi) ve Derin Öğrenme algoritmaları kullanılarak DDoS saldırılarının tespiti yapılmıştır.

3.2.1. Fisher Score

Fisher score 'un ana fikri özniteliklerin bir alt kümesini bulmaktır. Öyle ki seçilen özniteliklerin kapsamakta olduğu uzayda, farklı sınıflara ait veri örneklerinin arasındaki mesafe mümkün olduğunca büyükken, aynı sınıfa ait veri örneklerinin arasındaki mesafe ise mümkün oldukça küçük olacak şekilde alt kümeler bulunmaya çalışılır. Özellikle, seçilen m öznitelik göz önüne alınmış olsun. $X \varepsilon R^{dxn}$ girdi veri matrisi $Z \varepsilon R^{dxn}$ olacak şekilde küçültülür ve Fisher Score aşağıdaki gibi hesaplanır [38],

$$F(Z) = tr\{(S_b^{\sim})(S_t^{\sim} + \gamma I)^{-1}\} \tag{3.2}$$

γ pozitif bir normalleştirme parametresi, Sb ve St sırasıyla sınıf arası dağılım matrisi ve toplam dağılım matrisidir.

$$\begin{aligned} S_b^{\sim} &= \sum_{k=1}^{c} n_k(\mu_k^{\sim} - \mu^{\sim})(\mu_k^{\sim} - \mu^{\sim})^T \\ S_t^{\sim} &= \sum_{i=1}^{n}(z_i - \mu^{\sim})(z_i - \mu^{\sim})^T, \end{aligned} \tag{3.3}$$

$\mu_k^{\sim}$ ve n_k sırasıyla, mean vector ve indirgenmiş veri boyutundaki k. sınıfın boyutunu ifade eder.

Her öznitelik için Fisher Score hesaplandıktan sonra, Fisher Score metodu, en yüksek skora sahip m özniteliği seçer. Bu noktada, heuristic yöntemlerde; öznitelik seçme işlemi, her bir öznitelik için ayrı ayrı ve bağımsız olarak yapıldığından ötürü heuristic yöntemler ile öznitelik seçimi standartların daha altındadır. Çünkü nispeten düşük bireysel puanlara sahip olan ama bir bütün olarak bir araya getirildiklerinde çok yüksek bir skora sahip olan bu öznitelikleri seçme konusunda başarısız olurlar. Ek olarak, gereksiz öznitelikleri elimine etme konusunda da başarısız olurlar. Bu ve benzeri durumların söz konusu olması dolayısı ile Fisher Score, bu tez kapsamında kullanılan öznitelik seçme yöntemi olmuştur.

3.2.2. Temel Bileşen Analizi

Temel Bileşen Analizi (TBA), çok sayıda birbiri ile ilişkili değişken içeren veri setleri için veri setlerindeki değişimleri olabildiğince koruyarak, veri kümerlerinin

boyutlarının azlatılmasına olanak veren bir dönüşüm tekniğidir. Kısaca, veri setinin daha az özellik kullanılarak ifade edilmesine olanak sağlar. Dolayısı ile TBA, verilen verilerde daha az değişkenle ifade edilebilecek en iyi dönüşümü bulmayı amaçlamaktadır. Dönüşüm uygulanarak elde edilen değişkenler, ilk değişkenlerin temel bileşenleri olarak adlandırılırlar. En büyük varyans değerine ilk temel bileşen sahiptir ve diğer temel bileşenlerin varyans değerleri sırasıyla azalarak devam eder. Gürültüye karşı düşük hassasiyet, bellek miktarının yetersiz olması ve kapasite gereksinimleri için daha az boyutlu alanlarda daha verimli çalışma TBA 'nın başlıca avantajları arasındadır.

3.2.3. KNN

Örüntü tanımada, k-en yakın komşular algoritması (KNN), sınıflandırma ve regresyon için kullanılan parametrik olmayan bir yöntemdir [39]. Her iki durumda da, giriş, özellik alanındaki en yakın eğitim örneklerinden oluşur. Çıkış, KNN 'nin sınıflandırma veya regresyon için kullanılmasına bağlı olarak aşağıdaki gibidir:

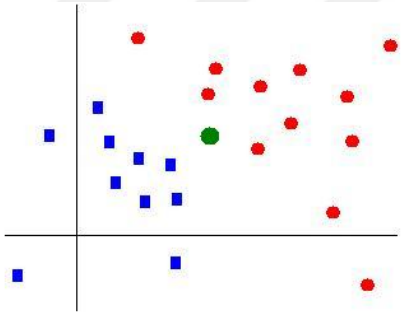
- Sınıflandırmada, çıktı bir sınıfın üyesidir. K değerine bağlı olarak nesne, en yakın k komşuluk değeri dikkate alınarak, sınıflandırılır ve en yakın k komşu sınıfına dahil edilir. Eğer k=1 ise en yakın komşuya göre sınıflandırılır.
- Regresyonunda ise çıktı nesnenin özellik değeridir. Bu değer, k en yakın komşularının değerlerinin ortalamasıdır.

Bir tekniğin parametrik olmaması, temel veri dağıtımında herhangi bir varsayım yapmıyor anlamına gelmektedir. Diğer bir ifade ile, model yapısı verilerden belirlenir. Bu aslında oldukça yararlıdır, çünkü uygulamaya yönelik "gerçek dünya" verilerinin çoğu, tipik teorik varsayımlara uymamaktadırlar (örneğin, doğrusal regresyon modellerinde olduğu gibi). Bu nedenle, KNN, verilerin dağılımı hakkında çok az bilgiye sahip olan ya da hiç bilgisi olmayan bir sınıflandırma çalışması için ilk seçeneklerden biri olmaya adaydır. KNN aynı zamanda, "hevesli" olarak ifade edebileceğimiz algoritmalara nazaran "tembel" bir öğrenme yöntemidir. Buradaki tembelin anlamı, KNN 'in hiçbir şey yapmaması değil, herhangi bir genelleme yapmak için eğitim veri noktalarını kullanmamasıdır. Dolayısı ile KNN 'in belirgin bir eğitim fazı yoktur veya çok azdır. Bu aynı zamanda eğitim aşamasının oldukça hızlı olduğu anlamına gelmektedir. Genelleme

eksikliğinden dolayı KNN tüm eğitim verisini tutar. Bu, test aşamasında tüm veya çoğu eğitim verisinin olmasını gerektirir.

KNN algoritması beş adımdan oluşur:

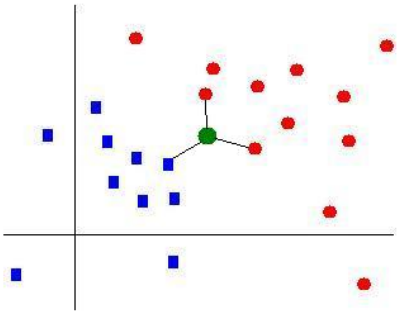
- K değeri belirlenir.
- Öklit uzaklıkları ile hedef nesnenin diğer nesnelerden uzaklığı hesaplanır.
- Referans noktasından minimum uzaklıklar ile en yakın komşular belirlenir.
- En yakın komşu sınıfları toplanır.
- En yakın komşu sınıfı seçilir.



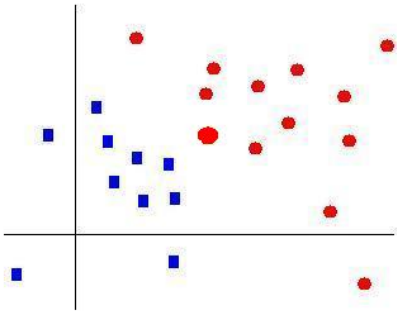
Şekil 3.2: KNN sınıflandırma örneği - birinci sınıfa ait örnekler mavi, ikinci sınıfa ait olanlar kırmızı, yeşil ile de sınıflandırılmak istenen nesne gösterilmiştir.

Örneğin, Şekil 3.2 'de görüldüğü gibi, mavi ve kırmızı olmak üzere iki adet sınıf, iki adet mavi sınıfına ait, iki adet kırmızı sınıfına ait ve birde sarı renk ile gösterilen ve hangi sınıfa ait olduğu bilinmeyen örnek bir veri olsun. Yeşil nesnenin sınıflandırılması için aşağıdaki adımlar, KNN tarafından sırasıyla uygulanır.

1. Öncelikle k değeri rastgele olarak belirlenir. Örneğin üç olsun.
2. Sarı renk ile gösterilen ve sınıflandırılmak istenen verinin diğer nesneler ile uzaklığı öklit ile hesaplanır.
3. Uzaklıklar sıralanır ve k adet nesne seçilir.
4. İki adet kırmızı ve bir adet mavi nesne olarak sınıf kategori sayıları belirlenir.



Şekil 3.3: En yakın komşuların gösterimi.



Şekil 3.4: Nesnenin en yakın sınıfa dahil edilmesi.

5. Çoğunluğun olduğu sınıf kırmızı olduğu için kırmızı sınıfına nesne dahil edilir.

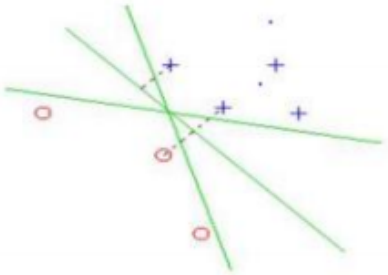
KNN algoritmasının en büyük dezavantajı uygun k değerinin bulunabilmesidir. Uygun k değeri deneme yanılma yolu ile manuel olarak seçilir. Bunun yanında veri seti ne denli büyük olursa algortima seçilen uygun k değeri ile daha iyi sonuç verebilir. Yine de sınıflandırma problemlerinde başta gelen algoritmalardan birisidir.

3.2.4. Destek Vektör Makineleri

İstatistiksel yöntemlere dayanan Destek Vektör Makineleri (DVM); 1998 yılında ilk defa Vapnik tarafından ortaya atılmıştır [40]. Gözetimli öğrenme türüdür. Öğrenme, örüntü tanıma, sınıflandırma, regrasyon ve analiz işlemleri için birçok farklı alanda kullanılmaktadır.

Etiketli bir veri setini girdi olarak aldığından dolayı gözetimli bir öğrenme yöntemidir. Veri setine bağlı olarak çıktıların sınıf sayısı değişir. Örneğin iki sınıftan oluşan bir veri seti girdi olarak verildiğinde iki sınıflı bir çıktı verisi üretilir. Buna bağlı olarak girdi olarak verilen örnekler bu sınıflara göre kategorize edilir. Eğitim aşamasında girdi veri setine bağlı olarak bir model oluşturulur ve bu model üzerinden sınıflandırma işlemi

gerçeklenir. Örneklerin her biri uzayda bir noktayı temsil eder ve model, uzayı sınıf sayısına bağlı olarak böler. Böylece uzayda Şekil 3.5 'de gösterildiği gibi hiperdüzlem oluşmuş olur. DVM, hiperdüzlem ile eğitim verisi içerisindeki örnekleri ayrır ve yeni örnekleri aynı uzay içerisinde sınıflandırmaya çalışır.



Şekil 3.5: Hiperdüzlem ile verilerin ayrılması.

İki sınıfın örneklerinin birbirine en yakın olduğu noktadan birbirine paralel olacak şekilde iki hiperdüzlem çizilir ve sonra bu iki hiperdüzlemi tam ortalayacak şekilde bir hiperdüzlem çizilir. Örnekler, uzayda düzenli bir şekilde dağılım göstermeleri durumunda bu hiperdüzlem doğrusal olur.

Doğrusal olarak ayrılabilen ve iki sınıftan oluşan k tane örnekten oluşan X eğitim verisi x_i, y_i, i=1,.....,k olduğu durumda, optimum hiperdüzleme ait eşitsizlikler:

$$\begin{aligned} w.x_i + b &\geqslant +1 \quad \text{her} \quad y = +1 \quad \text{için} \\ w.x_i + b &\leqslant +1 \quad \text{her} \quad y = -1 \quad \text{için} \end{aligned} \tag{3.4}$$

şeklinde olur. $x \varepsilon R^N$ olup N-boyutlu bir uzayı, $y \varepsilon \{-1, +1\}$ ise sınıf etiketlerini, w ağırlık vektörünü (hiperdüzlemin normali) ve b eğilim değerini ifade eder. Optimum hiperdüzlemin belirlenebilmesi için, bu düzleme paralel ve sınırlarını oluşturacak iki hiperdüzlemin belirlenmesi gerekir. Bu hiperdüzlemleri oluşturan noktalar destek vektörleri olarak adlandırılır ve bu düzlemler $w.x_i + b = \pm 1$ denklemi ile ifade edilirler [41].

Sınıflandırma işleminden önce en uygun hiperdüzlemin bulunması gerekir. Hiperdüzlemin ayırdığı alanlar, pozitif ve negatif olmak üzere farklı sınıfları ifade eder. Girdi olarak verilen X örnek, destek vektör makineleri ile formüle edilir ve f(x)

fonksiyonun sonucuna bağlı olarak sınıflardan birine kategorize edilir.

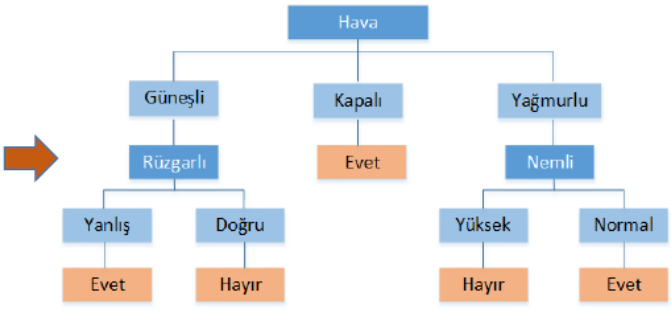
Destek vektör makineleri, sınıflandırma problemlerinin çözümünde en sık kullanılan tekniklerden biridir.

3.2.5. Karar Ağaçları

Karar ağaçları; bir ağaç yapısı şeklinde sınıflandırma veya regresyon modelleri oluşturur. Veri kümesini küçük ve daha küçük alt kümelere ayırırken, aynı zamanda ilişkili bir karar ağacı da aşamalı olarak geliştirilir. Sonuç olarak karar düğümleri ve yaprak düğümleri ile ağaç oluşur. Bir karar düğümden 2 veya daha fazla dallanma meydana gelir. Yaprak düğümler ise kararı veya sınıflandırmayı temsil eder. Ağacın en üstteki karar düğümü kök düğüm olarak adlandırılır. Karar ağaçları hem kategorik hem de sayısal verileri işleyebilir.

Karar ağacı örneği Şekil 3.6 'da verilmiştir. Verilen bir veri kümesi için Hava durumu dikkate alınarak golf oynama durumu için evet ve hayır cevapları aranmaktadır. Bu durumda, Hava düğümünün sırasıyla Güneşli, Kapalı ve Yağmurlu olmak üzere dallanmaları ve bunlara bağlı olarak oluşan yaprak düğümleri gösterilmiştir. Yaprak düğüm "Oynama" için Evet veya Hayır şeklinde oluşmaktadır.

Hava	Sıcaklık	Nemli	Rüzgarlı	Golf Oynama
Yağmurlu	Sıcak	Yüksek	Yanlış	Hayır
Yağmurlu	Sıcak	Yüksek	Doğru	Hayır
Kapalı	Sıcak	Yüksek	Yanlış	Evet
Güneşli	Ilık	Yüksek	Yanlış	Evet
Güneşli	Serin	Normal	Yanlış	Evet
Güneşli	Serin	Normal	Doğru	Hayır
Kapalı	Serin	Normal	Doğru	Evet
Yağmurlu	Ilık	Yüksek	Yanlış	Hayır
Yağmurlu	Serin	Normal	Yanlış	Evet
Güneşli	Ilık	Normal	Yanlış	Evet
Yağmurlu	Ilık	Normal	Doğru	Evet
Kapalı	Ilık	Yüksek	Doğru	Evet
Kapalı	Sıcak	Normal	Yanlış	Evet
Güneşli	Ilık	Yüksek	Doğru	Hayır

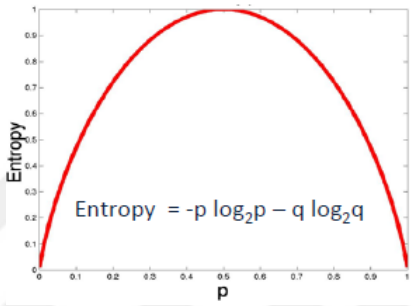


Şekil 3.6: Örnek Karar Ağacı Oluşturma.

Yukarıdan aşağıya geri izleme olmadan mümkün olan dallanmaların hesaplanmasını yapan greedy arama ile çalışan, ID3 olarak adlandırılan algoritma, bir karar ağacı oluşturulurken kullanılan temel algoritmadır. ID3 algoritması, ağaç oluştururken Entropi ve Bilgi Kazancı olmak üzere iki hesaplama kullanır. ZeroR modelinde hiçbir belirleyici bulunmazken, OneR modelinde tek ve en iyi belirleyici bulunmaya çalışır, Naive Bayesian yöntemi, Bayes kuralını ve belirleyici ile varsayımlar arasındaki bağımsızlığı kullanan tüm belirleyicileri içerir, fakat karar ağacı, belirleyici ile varsayımlar arasındaki

bağımlılığı kullanan tüm belirleyicileri içerir.

Entropi: Karar ağacı, kök düğümden yukarıdan aşağıya doğru oluşturulur ve verilerin, benzer değerlere (homojen) sahip örnekler içeren alt kümelere ayrılmasını içerir. ID3 algoritması, bir örneğin homojenliğini hesaplamak için entropi kullanır. Örnek tamamen homojense, entropi sıfırdır ve eğer örnek eşit olarak bölünmüşse, entropisi 1 'e eşit olur. Şekil 3.7 'de entropi hesabı grafik ile gösterilmiştir. P değerinin 0.5 olduğu anda Entropi hesabı aşağıdaki formül ile yapılır ([42]).



Şekil 3.7: Entropinin grafiksel gösterimi.

Bilgi kazancı: Bilgi kazancı, Veri kümesinin bir öznitelik ile bölünmesinden sonra entropide meydana gelen azalmaya dayanır. Karar ağacının oluşturulması tamamen en yüksek bilgi kazancı sağlayan özelliği bulmak ile ilgilidir. Bilgi kazancı adımnları aşağıdaki gibidir;

1. Adım: Hedefin entropisi hesaplanır.
2. Adım: Veri seti farklı özelliklere göre bölünür. Her bir dallanama için entropi hesaplanır. Daha sonra toplam entropi elde etmek için orantılı olarak eklenir. Ortaya çıkan entropi, bölünmeden önceki entropiden çıkarılır. Sonuç Bilgi Kazanımı veya entropi azalmasıdır.
3. Adım: Karar düğümü olarak en yüksek bilgi kazancına sahip öznitelik seçilir, veri kümesi dallarına bölünür ve aynı işlem her bir dal için tekrarlanır.
4. Bu adım ikiye ayrılır:
   (a) Entropisi 0 olan dal, yaprak düğüm olur.

(b) Entropisi 0 dan büyük olan dallara daha fazla bölünmesi gerekir.

5. Tüm veriler sınıflandırılıncaya kadar algoritma, yaprak düğüm olmayan düğümler için ardışık olarak çalıştırılır.

Karar ağaçları kolaylıkla kök düğümden başlayıp yaprak düğümlere kadar grafikten bir dizi kural kümesine dönüştürülebilirller.

3.2.6. Derin Öğrenme

Derin öğrenme algoritmalarındaki temel anlayış incelenen veride mevcut temsillerin otomatik olarak çıkarılmasıdır [43][44][45]. Gözetimsiz veride mevcut karmaşık temsillerin çıkarılmasında derin öğrenme algoritmaları kullanılmaktadır. Bu algoritmalar yapay zekanın kapsamından yer alan ve gözlemleme, analiz etme, öğrenme, karar verme gibi insan beyninin yeteneklerinden esinlenerek özellikle oldukça karmaşık problemlerin çözümlenmesinde kullanılmaktadır. Karar Ağaçları, Destek Vektör Makineleri veya Duruma-Bağlı Çıkarım (Case-based reasoning) gibi sığ yapıdaki Makine Öğrenme mimarileri, karmaşık yapıdaki girdi bilgisinden mantıklı ilişkilerin kurulmasında zaman zaman yetersiz kalmaktadır. Bununla birlikte Derin Öğrenme Mimarileri, verideki yakın ilişkilerin ötesinde küresel anlamda yerel olmayan ilişkilerin çıkarılması ve buna bağlı genelleştirilmiş kuralların belirlenmesinde başarılı sonuçlar ürettikleri görülmektedir [46]. Derin Öğrenme bu özelliği ile yapay zekaya doğru atılmış en önemli adımlardan biri olmakla birlikte insan müdahalesi olmadan doğrudan gözetimsiz veriden temsilleri çıkarması büyük önem taşımaktadır.

Derin Öğrenme algoritmaları soyut temsillerin elde edilmesi şeklinde olmaktadır. Soyut temsillerin temel faydası, girdinin kendine ait farklılıkların, modelin oluşturulmasındaki etkisini azaltmasıdır; bu şekilde değerlendirildiğinde veri içerisindeki local değişken faktörlerin (örn. aynı imgenin farklı renk tonları) etkileyemediği soyut öz niteliklerin elde edilmesi en önemli amaçtır. Böylece farklı kaynaklardan gelen benzer/aynı tipteki verinin ayrıştırılması sağlanmaktadır.

Derin Öğrenme algoritmaları birbirini takip eder tarzdaki katmanlardan oluşan bir mimariye sahiptir. Her katmanda girdilere lineer olmayan dönüşüm fonksiyonları uygulanarak katmana ait çıktı temsilleri/özellikleri elde edilmektedir. Burada hedeflenen

birinci katmandan en son katmana hiyerarşik olarak verinin belirtilen dönüşüm fonksiyonlarına sokularak karmaşık ve soyut özelliklerinin (temsillerinin) elde edilmesini sağlamaktır. İlk katman verinin gözlemlenen ilk değerleri olurken devamındaki katmanlar önceki katmaların çıktılarını girdi olarak alır ve işlerler.

Üst üste veya ardı sıra lineer olmayan dönüşüm fonksiyonlarını uygulandığı katman yapısı Derin Öğrenme algoritmalarının temelini oluşturmaktadır. Katman sayısının artırılması inşa edilecek doğrusal olmayan dönüşümlerin karmaşıklığını da artıracaktır. Derin Öğrenmeyi algoritmaları en son katmanda elde edilen verinin soyut saklı özelliklerini bir bağlamda çoklu seviyede elde edilen soyut temsillerinden öğrenmektedir. Böylelikle son katman çıktısı, soyut özellikleri, verinin yüksek seviyeli doğrusal olmayan bir fonksiyona sokulması sonucu olarak elde edilir.

Katmanlardaki doğrusal olmayan dönüşüm (non-linear transformation) veride mevcut bulunan ve veriye ait açıklayıcı faktörlerin çıkarılmasına çalışmaktadır. Temel Bileşen Analizi (TBA) benzeri doğrusal bir dönüşüm fonksiyonunun kullanılmamasının sebebi ardı sıra gelen katmanlarda uygulanacak doğrusal dönüşüm sonucu elde edilecek nihai fonksiyonun da doğrusal olmasıdır ve katmanlı yapı kullanımını anlamsız hale getirmektedir. Derin Öğrenme algoritmalarının her bir katmanında uygulanan doğrusal olmayan dönüşüm ile verideki değişkenlik faktörünün sebep olacağı karmaşıklığın giderilmesini sağlar. Bu kavramın uygun bir eğitim kıstası haline dönüştürülmesi Derin Öğrenme algoritmaları için devam eden önemli bir araştırma konusudur [43].

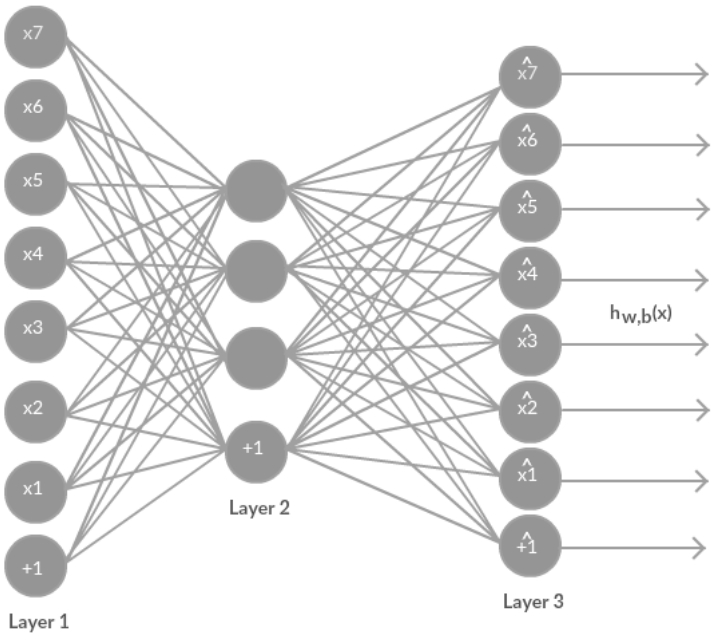
Derin Öğrenme algoritmaları, veriye ait sınıflandırıcıların belirlenmesi, veri indekslemesi veya çok boyutlu sensör verisi yerine daha soyut temsillerin kullanılabileceği ortamlarda kullanılacak temsilleri final katmanının çıktısından sağlamaktadır.

Derin mimarilerde parametrelerin öğrenilmesi zor bir optimizasyon görevidir, örnek olarak birçok katmana sahip sinir ağlarında. 2006 yılında Hinton tarafından gözetimsiz açgözlü katman tabanlı (greedy layer-wise) öğrenme yöntemi ile bu probleme bir çözüm önerisi getirilmiştir [47]. Başlangıçta duyulara ait veri ile mimarinin ilk katmanı beslenmiş ve ilk katman bu veriler ile eğitilmiştir. Birinci katmanın çıktıları ikinci katman için girdi olarak kullanılmıştır. Benzeri tekrar ile istenen sayıdaki katman beslenerek eğitim tamamlanmıştır. Son katmandan elde edilen veriye ait soyut özellikler ile farklı görevler yerine getirilmesi sağlanmıştır. Parametrelere ait ince ayarlama (fine tuning) ise

gözetimli olarak yapılabilmektedir.

Derin öğrenmede en çok kullanılan temel bloklar gözetimsiz olarak eğitimin gerçekleştirildiği tek katmanlı Autoencoders (Otomatik Kodlayıcılar) ile Restricted Boltzmann Machines (Kısıtlı Boltzmann Makineleri) 'dir. Bu iki algoritma ile yığıt Otomatik Kodlayıcılar (stacked Autoencoders) [48][49] ve Derin Kanaat Ağları (deep belief networks) [47] oluşturulmaktadır. Ardı sıra gelen katmanlarda kullanılan Otomatik Kodlayıcılar ile Yığıt Otomatik Kodlayıcılar oluşturulurken, benzer şekilde Kısıtlı Boltzmann Makinelerinin kullanımı ile Derin Kanaat Ağları inşaa edilmektedir.

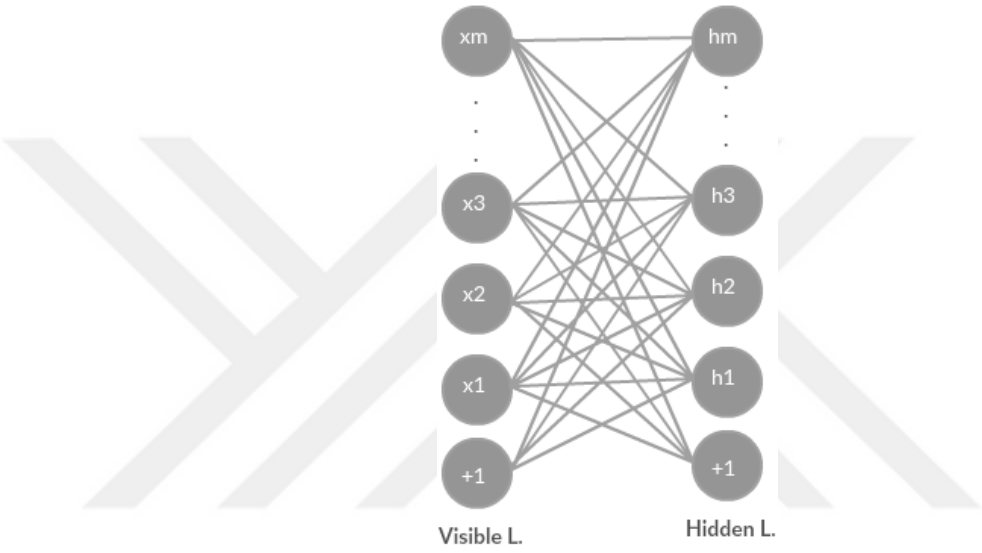
Otomatik Kodlayıcıların oluşturduğu ağlarda temel olarak üç katman yer alır; girdi katmanı, saklı katman ve çıktı katmanı. Otomatik Kodlayıcılar girdi katmanındaki veriden elde edecekleri soyut özellikler/temsiller ile çıktı katmanında tekrar girdi katmanı elde etmek için dizayn edilmişlerdir. Başka bir değişle çıktı katmanında hedeflenen çıktı aslında girdi katmanındaki verinin kendisidir. Temel bir otomatik kodlayıcı tekrar inşa (reconstruction error) 'da ortaya çıkan hatanın azaltılmasını sağlayarak parametre güncellemelerini yapmaktadır. Hatanın azaltılmasında genel olarak İstatistiksel Dereceli Azalma (Stocastic Gradient Descent) algoritması kullanılmaktadır.



Şekil 3.8: Otomatik kodlayıcı diyagramı.

Bir diğer gözetimsiz öğrenme algoritması olan Derin Kanaat Ağlarının (Deep Belief

Network) oluşturulansında kullanılan tek katmanlı Kısıtlı Boltzmann Makineleri Boltzmann Makinelerinin en çok bilinen versiyonlarıdır. Şekil 3.9 'da kısıtlı boltzman makinesi diyagramı gösterilmiştir. Bu yapıda görünen katman ve saklı katman yapısı yer alır. Boltzmann Makinelerinde aynı katmanda yer alan birimlerin kendi içindeki etkileşimlerinin kaldırılması ile Kısıtlı Boltzmann Makineleri elde edilmektedir[50]. Contrastive Divergence algoritması Boltzmann Makilerinin eğitiminde kullanılan başlıca algoritmalardan biridir.



Şekil 3.9: Kısıtlı boltzmann makinesi diyagramı.

Derin Öğrenme algoritmaları anlamlı soyut özellikleri işlenmemiş veriden elde etmektedir ve bunu yüksek seviyedeki katmanların daha alt seviyelerdeki daha düşük soyut özellikleri kullanarak yapan çok katmanlı hiyerarşik öğrenme yaklaşımı kullanmaktadır. Derin Öğrenme algoritmaları etiketli veriyi kullanabildiği gibi daha fazla miktarlarda bulunan etiketsiz veriyi güdümsüz olarak kullanabilmektedir[43][45][51], bu yapısı ile fazla miktarlarda elde edilebilen veriden anlamlı özellik/temsillerin ve örüntünün çıkarılmasını sağlamaktadır. Derin Öğrenme algoritmalarının yerel olmayan ve küresel ilişkilerin elde edilmesinde, veriye ait örüntünün çıkarılmasında sığ öğrenme mimarilerine göre daha başarılı oldukları görülmektedir[48].

3.3. YÖNTEM

Bu çalışmada, ağ üzerinde gerçekleşen DDoS ataklarının tespiti için, Kanada Siber Güvenlik Enstitüsü tarafından geliştirilen CICIDS2017 veriseti kullanılmıştır. Gerçek dünya verileri (PCAP) gibi zarasız ve en ileri düzeydeki yaygın saldırılar CICIDS2017 veri kümesinde bulunmaktadır. Ayrıca veri kümesi, IP protokollerinin kaynak ve hedef bağlantı noktaları, timestampe (CSV dosyaları), kaynak ve hedef adreslerine dayalı olarak etiketli akışlar kullanılarak ağ trafiği analizinin sonuçlarını içerir.

Burada, DDoS ve normal trafik toplamı olarak 225,746 kayıt bulunmaktadır. Her kayıt 80 adet öznitelik içermektedir. Bunlar, Protocol, Flow ID, Source IP, Destination IP, Source Port, Destination Port, Flow Duration, Total Fwd Packets, Total Backwards Packet vs. dir. Şekil 3.10 verisetinin örnek bir kısmını göstermektedir.

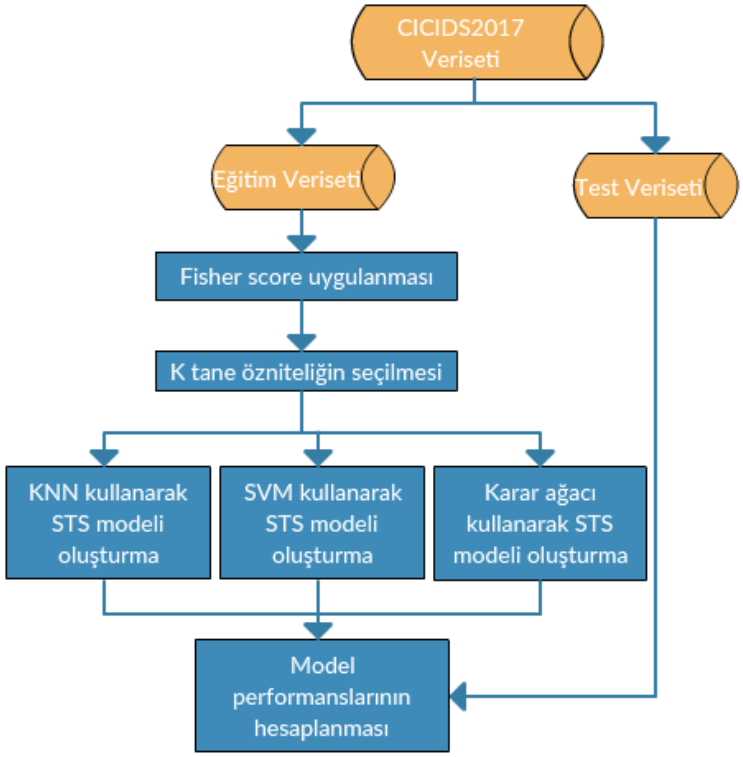
Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets
192.168.10.16-199.244.48.55-41936-443-6	192.168.10.16	41936	199.244.48.55	443	6	143347	47	60
192.168.10.16-54.210.195.63-42970-80-6	192.168.10.16	42970	54.210.195.63	80	6	50905	1	1
192.168.10.16-199.244.48.55-41944-443-6	192.168.10.16	41944	199.244.48.55	443	6	143899	46	58
192.168.10.3-192.168.10.17-53-12886-17	192.168.10.17	12886	192.168.10.3	53	17	313	2	2
192.168.10.16-199.244.48.55-41942-443-6	192.168.10.16	41942	199.244.48.55	443	6	142605	45	58
192.168.10.3-192.168.10.17-53-33063-17	192.168.10.17	33063	192.168.10.3	53	17	253	2	2
192.168.10.16-199.244.48.55-41940-443-6	192.168.10.16	41940	199.244.48.55	443	6	142499	46	53
192.168.10.16-199.244.48.55-41938-443-6	192.168.10.16	41938	199.244.48.55	443	6	23828	27	31
192.168.10.16-199.244.48.55-41946-443-6	192.168.10.16	41946	199.244.48.55	443	6	119090	23	28
192.168.10.25-17.253.14.125-123-123-17	192.168.10.25	123	17.253.14.125	123	17	63021198	2	2
192.168.10.16-199.244.48.55-41946-443-6	192.168.10.16	41946	199.244.48.55	443	6	23841	26	31
192.168.10.3-192.168.10.9-53-65431-17	192.168.10.9	65431	192.168.10.3	53	17	227	2	2
192.168.10.16-199.244.48.55-41950-443-6	192.168.10.16	41950	199.244.48.55	443	6	199037	47	63
192.168.10.16-199.244.48.55-41948-443-6	192.168.10.16	41948	199.244.48.55	443	6	142923	46	60
192.168.10.3-192.168.10.9-53-58966-17	192.168.10.9	58966	192.168.10.3	53	17	222	2	2
192.168.10.9-5.79.75.140-8503-443-6	5.79.75.140	443	192.168.10.9	8503	6	410591	1	1
192.168.10.16-199.244.48.55-41956-443-6	192.168.10.16	41956	199.244.48.55	443	6	446800	23	28
192.168.10.9-5.79.75.140-8503-443-6	192.168.10.9	8503	5.79.75.140	443	6	56	2	0
192.168.10.16-199.244.48.55-41954-443-6	192.168.10.16	41954	199.244.48.55	443	6	141851	47	55
192.168.10.16-199.244.48.55-41952-443-6	192.168.10.16	41952	199.244.48.55	443	6	145376	51	60
192.168.10.16-199.244.48.55-41958-443-6	192.168.10.16	41958	199.244.48.55	443	6	142904	47	56
192.168.10.1-192.168.10.3-53-61539-17	192.168.10.3	61539	192.168.10.1	53	17	23537	1	1
192.168.10.3-192.168.10.15-53-51561-17	192.168.10.15	51561	192.168.10.3	53	17	24027	2	2
192.168.10.16-199.244.48.55-41956-443-6	199.244.48.55	443	192.168.10.16	41956	6	23936	29	23
192.168.10.16-199.244.48.55-41964-443-6	192.168.10.16	41964	199.244.48.55	443	6	142508	50	59
192.168.10.16-199.244.48.55-41962-443-6	192.168.10.16	41962	199.244.48.55	443	6	142561	48	59
192.168.10.16-199.244.48.55-41960-443-6	192.168.10.16	41960	199.244.48.55	443	6	142389	52	60
192.168.10.5-23.60.139.27-55012-80-6	192.168.10.5	55012	23.60.139.27	80	6	82749413	24	21

Şekil 3.10: Veriseti kayıtlarının örnek bir kısmı.

3.3.1. Fisher Score, KNN, DVM ve Karar Ağaçları ile Saldırı Tespiti

Öncelikle, bu kısımda veri setinin tamamı kullanılmayıp, 225711 örneğin 52972 adedi kullanılmış ve bu verisetine öncelikle fisher score yöntemi uygulanarak en iyi öznitelikleri belirlenmiş ve KNN, DVM, ve Karar ağaçları sınıflandırma yapılmıştır. Veri setinin tamamı, bu bölümdeki makine öğrenmesi tekniklerinin yanı sıra, derin öğrenme algoritmasınında uygulandığı bir sonraki bölümde kullanılmıştır. Veri setinin

bir kısmı (%10) test verisi olarak bir kısmı ise (%90) eğitim verisi olacak şekilde ikiye bölünmüştür. Eğitim verilerinin içerisindeki özniteliklerin her biri için Fisher Score değerleri hesaplanmıştır. Hesaplanan değerlerin en yüksek olduğu k adet öznitelik seçilmiştir ve bu öznitelikler ile veri setinin boyutu küçültülmüştür. Bu işlemin ardından sırasıyla, k en yakın komşuluk, destek vektör makineleri ve karar ağaçları ile her bir yöntem için kendi içerisinde saldırı tespit modelleri oluşturulmuştur. Son olarak, oluşturulan bu saldırı tespit modellerinin her biri için Accuracy, Recall, Precision, F1 değerleri hesaplanmıştır. Kullanılan makine öğrenmesi yöntemine bağlı olarak saldırı tespit sistemlerinin performans analizlerinin tespiti için önerilen çalışmanın akış diyagramı Şekil 3.11 'de gösterilmiştir.



Şekil 3.11: Fisher skor, KNN, DVM ve karar ağaçları ile saldırı tespiti

Çalışmada öznitelik sayısı 80 den 10 'a 10 ar 10 ar azaltılarak öğrenme algoritmalarının başarı oranları 100 iterasyon için ayrı ayrı hesaplanmıştır ve böylece maksimum doğruluğa ulaşılan öznitelikler ile öznitelik sayısı belirlenmiştir. KNN 'in k değeri 1 olarak seçilmiştir. Öznitelik sayısına bağlı algoritmaların doğruluk oranlarının grafiği ve sınıflandırıcıların performansı sırasıyla aşağıdaki şekil ve Tablo 3.1 'de gösterilmiştir.

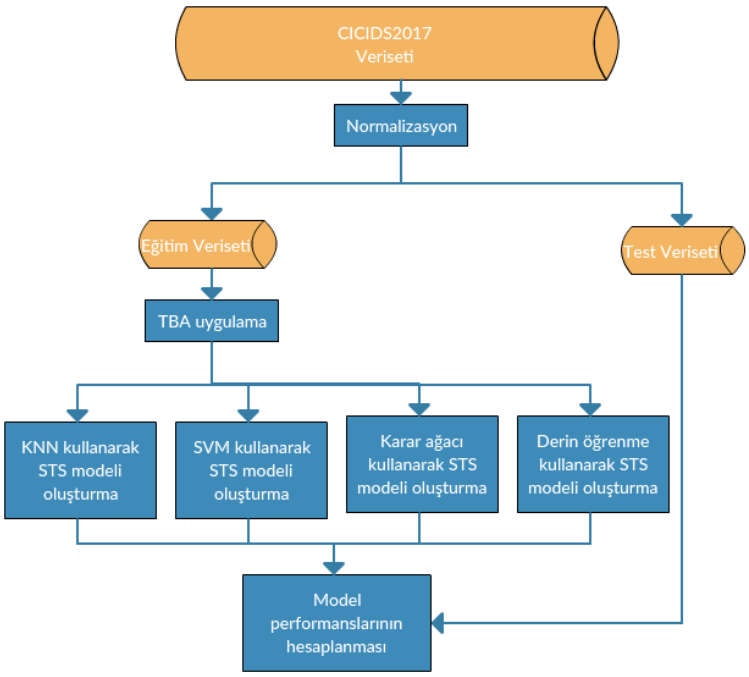
Tablo 3.1: Sınıflandırıcı performansları.

Algoritma	Öznitelik Sayısı	Doğruluk	Geri çağırma	Hassasiyet	F1
KNN	80	0.9572	0.9589	0.9566	0.9577
	30	0,9997	0.9985	0.9968	0,9997
DVM	80	0,6069	0,7142	0.5956	0.6463
	30	0,5776	0,8097	0,5654	0,6564
KA	80	0,99	0,99	0,99	0,99
	30	0,99	0,99	0,99	0.99

Tablo 3.11 'de 80 ve 30 öznitelik için sınıflandırıcıların performansları gösterilmektedir. Doğruluk, geri çağırma, hassasiyet ve F1 skor değerleri her bir algoritma için hesaplanmış ve gösterilmiştir. K en yakın komşuluk için 30 öznitelik ile daha iyi sonuçlar veriyor iken, karar ağaçları için hesaplama değerleri değişmemiştir. Diğer taraftan destek vektör makineleri hem 30 hem de 80 öznitelik için başarı oranları en düşük çıkan öğrenme yöntemi olmuştur.

3.3.2. TBA, KNN, DVM, Karar Ağaçları ve Derin Öğrenme ile Saldırı Tespiti

Bu tez kapsamında çeşitli makine öğrenmesi yöntemlerinin fisher score öznitelik seçme algoritması ile birlikte kullanarak karşılaştırmalı sonucunu vermenin yanı sıra Derin Öğrenme algoritması kullanarak DDoS saldırılarının tespiti gerçekleştirilmiştir. Şekil 3.12 'de akış diyagramı gösterilmiştir. Öncelikle toplamda 225711 DDoS atak ve normal davranışları içeren kayıtlar CICIDS2017 verisetinden alınmıştır. Alınan kayıtlar normalleştirilmiş ve %77 'si eğitim verisi olarak %33 'ü ise test verisi olarak ikiye ayrılmıştır. Bu işlemin ardından eğitim verisi TBA uygulanarak veriseti boyutu küçültülmüştür. KNN, DVM, karar ağaçları ve derin öğrenme yöntemleri kullanılarak STS modelleri oluşturulmuştur. Oluşturulan modeller test verisi kullanılarak test edilmiş ve performans metrikleri ölçülmüştür.



Şekil 3.12: TBA, KNN, DVM, karar ağaçları ve derin öğrenme ile saldırı tespiti akış diyagramı.

Şekil 3.12 'de görüldüğü üzere, ilk adım normalizasyon adımıdır. Bu adımda, gözlemlenen tüm özellik değerleri 1 uzunluğa sahip olacak şekilde yeniden boyutlandırılmıştır. Bunun sebebi verisetindeki bazı özniteliklerin farklı formlarda olmasından dolayıdır. Örneğin kullanmış olduğumuz verisetinde Ip Adresleri nokta(.) içermektedir. Dolayısı ile bu noktaların kaldırılması ve bu değerlerinde 0 ve 1 aralığında bir değere dönüştürülmelsi gerekir. Sadece Ip Adresleri değil, verisetindeki tüm öznitelikler, normalleştirilmiş ve 0 ile 1 arasında bir değere dönüştürülmüştür. Ayrıca, NaN ve boş değerler elimine edilmiştir. Böylece, veri seti kullanılıp işlem yapmak için hazır hale getirilmiştir.

İkinci adımda, toplam 225711 kayıt içeren ve işlenmek için hazır hale getirilen veriseti eğitim ve test verisi olmak üzere ikiye bölünmüştür. Eğitim verisi içerisinde 27990 DDoS atak ve 123236 normal davranışa ait kayıt bulunmaktadır. Diğer taraftan test verisi ise 13844 DDoS ve 60641 normal davranışa ait kayıt içermektedir.

Üçüncü adımda, eğitim verisetinin TBA kullanılarak boyutu azaltılmış ve STS modelleri oluşturulmuştur. Derin öğrenme modeli 50 iterasyonda test edilmiş ve ince ayar (fine tuning) yapılmıştır.

Son olarak, oluşturulan tüm STS modelleri test verileri kullanılarak test edilmiştir. Bu işlem ile birlikte modellerin performans metriklerinin hesaplanması için TP, TN, FP, FN değerleri hesaplanmıştır ve hesaplanan bu değerler Tablo 3.2 'de gösterilmiştir.

Tablo 3.2: Karışıklık Matrisi

Gerçek Sınıf \Tahmin Edilen Sınıf	Normal (Benign)	Anomali (DDoS)
Normal (Benign)	TN	FP
Anomali (DDoS)	FN	TP

- TN: Zararsız olarak sınıflandırılmış zararsız davranış.
- FP: DDoS olarak sınıflandırılmış zararsız davranış.
- FN: Zararsız olarak sınıflandırılmış DDoS.
- TP: DDoS olarak sınıflandırılmış DDoS.

Tablo 3.3: Performans metrikleri.

Ölçüm	**Formül**
Doğruluk	(TP+TN) / (TP+FP+FN+TN)
Geri çağırma	TP / (TP+FN)
Kesinlik	TP/(TP+FP)
F1 skor	2TP / (2TP+FP+FN)

Doğruluk, doğru tahmin edilen gözlemlerin oranı anlamına gelir. Geri çağırma, duyarlılık veya doğru pozitif oranlar olarak adlandırılabilir ve doğru tahmin edilen pozitif olayların oranını ifade eder. Kesinlik, doğru pozitif gözlemlerin oranıdır. Kesinlik ve geri çağırmanın ortalaması F1 skorunu ifade eder.

4. BULGULAR

Bu çalışmadaki bütün işlemler, Intel(R) Core(TM) i7-5700HQ CPU @2.70 GHz işlemci ve 16 GM Ram kapasitesine sahip kişisel bilgisayar üzerinde gerçekleştirilmiştir.

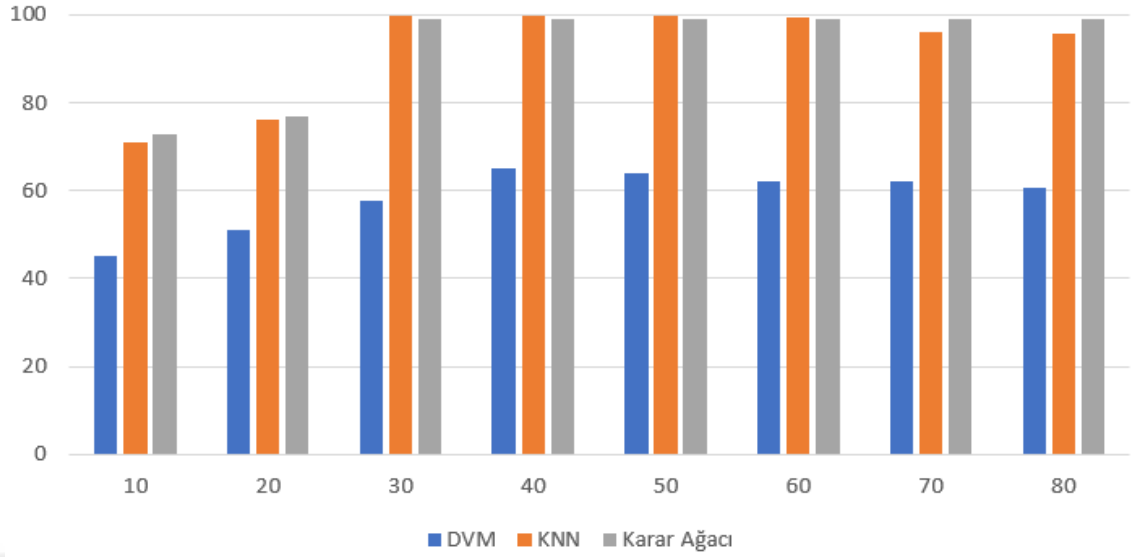
4.1. FİSHER SCORE, KNN, DVM VE KARAR AĞAÇLARI İLE SALDIRI TESPİTİ BULGULARI

Bu kısımda, veri setinin tamamı kullanılmayıp, 225711 örneğin 52972 adedi kullanılması sonuçları ile verisetine öncelikle fisher score yöntemi uygulanarak en iyi özniteliklerin belirlenmesi ve KNN, DVM, ve Karar ağacı sınıflandırma yöntemlerinin bulgularına yer verilmiştir.

Fisher skor yöntemi en iyi özniteliklerin seçilmesi için kullanılmıştır. En iyi performans gösteren öznitelikler sınıflandırma için önem sırasına göre Tablo 4.1 'de listelenmiştir.

'Fwd Packet Length Mean' saldırı tespit sınıflandırma işlemleri için en belirgin özelliklere sahip olan öznitelik olmuştur. Sırasıyla bu özniteliği, ' Avg Fwd Segment Size', ' Fwd Packet Length Max', ' Bwd IAT Min', 'Total Length of Fwd Packets', ' Subflow Fwd Bytes' ' Fwd Packet Length Std', ' Destination Port', ' Protocol', 'Fwd PSH Flags' öznitelikleri önem sıralarına göre takip etmiştir.

Öznitelik sayısının değişimine bağlı olarak sınıflandırma algoritmalarının performansı Şekil 4.1 'de verilmiştir.



Şekil 4.1: Öznitelik sayısına bağlı olarak doğrulukların gösterimi.

Şekil 4.1 öznitelik sayısının, 80 ile 10 arasında, 10 ar 10 ar azaltılması ile sınıflandırıcıların performansının değişimini göstermektedir. Mavi renk ile destek vektör makineleri, kırmızı ile k en yakın komşuluk ve yeşil renk ile de karar ağacı algoritmaları temsil edilmiştir. Öznitelik sayısının 10 'a düştüğü andaki makine öğrenmesi tekniklerinin performansında belirgin bir azalma olduğu yukarıdaki şekilde belirgin bir şekilde görülmektedir. Bunun sebebi olarak kullanılan verisetinde 10 özniteliğin yüksek doğruluk oranı ile sınıflandırmanın yapılması için yeterli olmadığı ve yüksek doğruluk için algoritmaların 10 öznitelikten daha fazla özniteliğe ihtiyaç duyduğu sonucu çıkarılabilir. Bu grafiğin elde edilmesinde öznitelik seçme algoritması olan Fisher skor metotu kullanılmıştır.

Tablo 4.1: Algoritma performanslarının karşılaştırımı.

Yöntem	Öznitelik Sayısı	Doğruluk	Geri Çağırma	Kesinlik	F1 Skoru
KNN	**80**	0.9572	0.9589	0.9566	0.9577
	30	0.9997	0.9985	0.9968	0.9997
DVM	**80**	0.6069	0.7142	0.5956	0.6463
	30	0.5776	0.8097	0.5654	0.6564
Karar Ağacı	**80**	0.99	0.99	0.99	0.99
	30	0.99	0.99	0.99	0.99

KNN makine öğrenmesi yönteminin 80 öznitelik ile sınıflandırmada doğruluk oranı 0.9572 iken, öznitelik sayısı 30 'a düşürüldüğünde bu oran 0,9997 'ye çıkmıştır. Öznitelik sayısının azaltılması ile gereksiz ve ilgisiz öznitelikler KNN sınıflandırma algoritmasının eğitiminde kullanılmadığı için daha yüksek doğruluğun elde edildiği düşünülmektedir. Karar ağacı için öznitelik sayısının 80 den 30 düşürülmesi algoritmanın performans değerlerinde herhangi bir değişikliğe sebep olmamış ve tüm performans metrikleri için aynı olan 0,99 değeri elde edilmiştir. Karar ağacı algoritmasında ağaç oluşturulurken en iyi 30 öznitelik ile bu ağaç oluşturulduktan sonra eklenen öznitelikler ağacın sınıflandırma şeklini değiştimediği düşünülmektedir. Destek vektör makinesi için öznitelik sayısının azaltılması, 0,6069 olan doğruluk oranının, 0,5776 'ya azalmasına sebebiyet vermiştir. Destek vektör makinesinin sınıflandırma işlemini gerçekleştirirken veri setindeki örneklere göre optimum hiperdüzlemi çizip başarılı bir sınıflandırma işlemi gerçekleştirebilmesi için doğruluk ile öznitelik sayısının doğru orantılı olduğu düşünülmektedir. Algoritmaların sınıflandırm işlemi için çalışma yöntemleri belirli olsa dahi sınıflandırma performansları veriseti ile yakından ilişkili olduğundan yorumlamaları genellememiz doğru bulunmamıştır. Yorumlamalar kullanılan verisetlerine bağlı olarak değişebilmektedir.

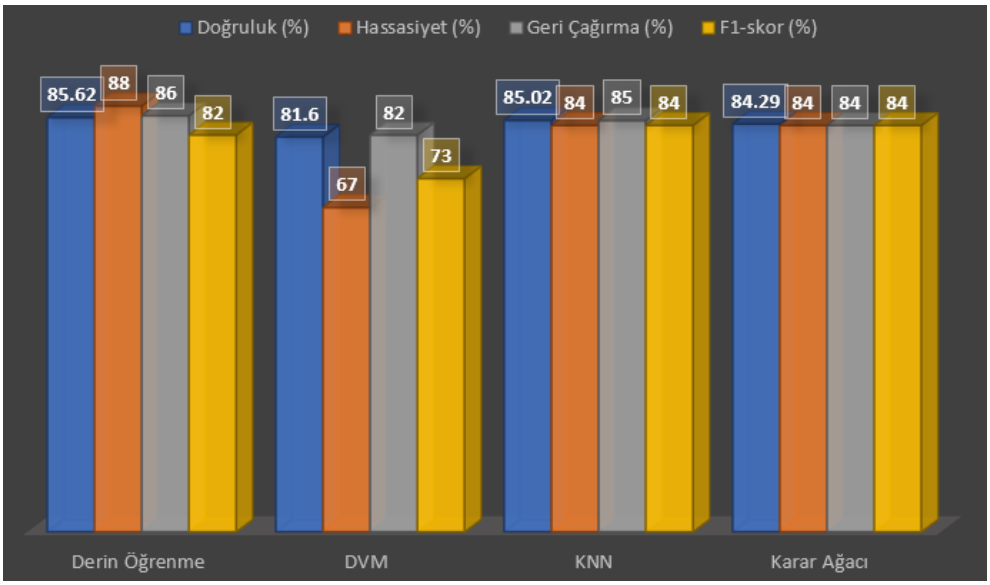
4.2. TBA, KNN, DVM, KARAR AĞAÇLARI VE DERİN ÖĞRENME İLE SALDIRI TESPİTİ BULGULARI

DDoS saldırılarından ve normal davranışlardan oluşan 225711 kayıt, % 77 'si yani 151226 tanesi eğitim, % 33 'ü yani 74485 adedi ise test için kullanılmak üzere ikiye bölünmüştür. Derin Öğrenme modeli 50 iterasyon boyunca eğitilmiş ve modelin performans değerleri hesaplanmıştır. Ayrıca, DVM, KNN ve karar ağacı yöntemleri sınıflandırma algoritmaları olarak kullanılmışlardır ve tüm yöntemlerin performans ölçümleri karşılaştırmalı olarak Tablo 4.2 'de sunulmuştur.

Tablo 4.2: CICIDS2017 veri seti ile kullanılan algoritmaların performans ölçümleri.

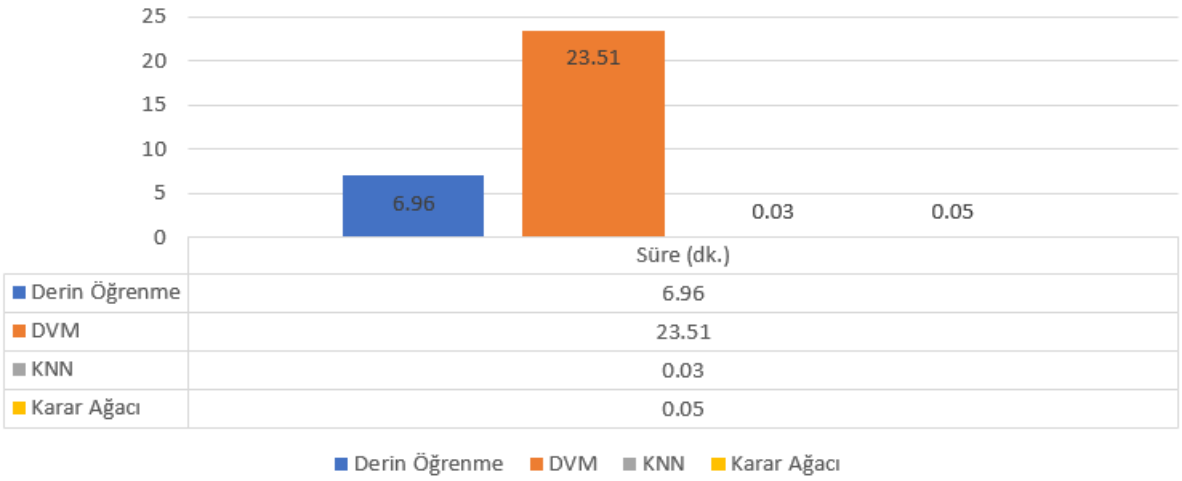
Metod	TP	TN	FP	FN	Doğruluk	Hassasiyet	Geri çağırma	F1 skor	Süre (dk.)
Derin Öğrenme	3000	60778	3	10704	0.8562	0.88	0.86	0.82	6.96
SVM	0	60781	0	13704	0.8160	0.67	0.82	0.73	23.51
KNN	6375	56955	3826	7329	0.8502	0.84	0.85	0.84	0.03
Karar Ağacı	7503	55304	5477	6201	0.8432	0.84	0.84	0.84	0.05

Tablo 4.2, derin öğrenme, DVM, KNN ve karar ağacı yöntemleri kullanılarak geliştirilen STS modellerinin performans ölçümlerini göstermektedir. Performans ölçümü için doğruluk, hassasiyet, geri çağırma ve f1 skor değerleri kullanılmıştır. Şekil 4.2 'de algoritmaların performans ölçüm değerleri karşılaştırmalı olarak gösterilmiştir.



Şekil 4.2: Algoritmaların başarılarının karşılaştırılması.

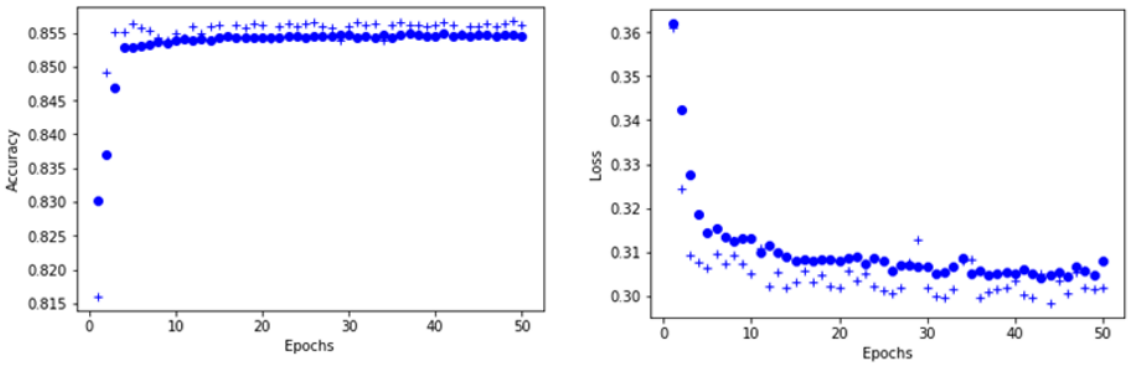
Derin öğrenme algoritmasının sırasıyla %85.62, %88 ve %86 ile en yüksek doğruluk, hassasiyet ve geri çağırma oranlarına sahip olduğu Şekil 4.2 'de görülmektedir. Aksine, en düşük doğruluk, hassasiyet, geri çağırma ve f1 skoru SVM tarafından gerçekleştirilmiştir. KNN'nin doğruluk ve geri çağırma oranları, karar ağacı algoritmasının doğruluğu ve geri çağırma oranlarından daha yüksektir. Bununla birlikte, KNN ve karar ağacı, hassasiyet ve f1 skor oranları için aynı sonuçlara sahiptir. Bu ölçümlere ek olarak, derin öğrenme, DVM, KNN ve karar ağacı algoritmalarının işlem süreleri sırasıyla 6.96, 23.51, 0.03 ve 0.05 dakika olarak ölçülmüş ve Şekil 4.3 'de gösterilmiştir.



Şekil 4.3: Algoritmaların sürelerinin karşılaştırılması.

Yulkarıdaki şekilde görülmekte olduğu üzere, SVM veri kümesini işlemek için en fazla CPU süresini harcar. Derin öğrenme, 50 epok boyunca çalıştırıldığı için 6.96 dakika harcamıştır. KNN ve Karar ağacı işlem süreleri hem çok az hem de birbirlerine çok yakındır. Süre olarak değerlendirildiğinde KNN en iyi performansı sergilemiştir.

Şekil 4.4, iterasyona bağlı olarak kaybın ve doğruluğun değişimini göstermektedir.



Şekil 4.4: Derin öğrenme STS modelinde doğruluk ve kaybın Epok 'a göre değişimi.

Şekil 4.4 'de, mavi noktalar kayıp ve doğruluk anlamına gelirken, mavi artılar ise doğrulama kaybı ve doğrulama doğruluklarını göstermektedir. Şekil, iterasyon sayısının 0 'dan 50 'ye arttıkça doğruluğun artarken kaybın azalmasını göstermektedir. Detaylı sonuç Tablo 4.3 'te gösterilmiştir.

Tablo 4.3: Derin öğrenme STS modelinin 50 epokta performans değerleri.

Epok	Kayıp	Doğruluk	Doğrulama Kaybı	Doğrulama Doğruluğu
1	0.3619	0.8303	0.3611	0.8160
2	0.3425	0.8370	0.3245	0.8492
3	0.3276	0.8470	0.3093	0.8551
4	0.3187	0.8528	0.3078	0.8551
5	0.3145	0.8528	0.3065	0.8563
6	0.3154	0.8531	0.3095	0.8558
7	0.3135	0.8533	0.3075	0.8553
8	0.3127	0.8537	0.3093	0.8542
9	0.3133	0.8535	0.3074	0.8539
10	0.3130	0.8539	0.3052	0.8550
11	0.3099	0.8540	0.3108	0.8542
12	0.3115	0.8538	0.3022	0.8559
13	0.3099	0.8540	0.3054	0.8549
14	0.3090	0.8539	0.3019	0.8560
15	0.3081	0.8543	0.3033	0.8562
16	0.3083	0.8545	0.3057	0.8545
17	0.3081	0.8544	0.3031	0.8562
18	0.3084	0.8544	0.3049	0.8557
19	0.3082	0.8544	0.3023	0.8564
20	0.3081	0.8543	0.3020	0.8561
21	0.3086	0.8543	0.3059	0.8542
22	0.3091	0.8542	0.3034	0.8559
23	0.3073	0.8544	0.3051	0.8563
24	0.3086	0.8544	0.3023	0.8559
25	0.3080	0.8543	0.3012	0.8563

Epok	Kayıp	Doğruluk	Doğrulama Kaybı	Doğrulama Doğruluğu
26	0.3058	0.8546	0.3007	0.8566
27	0.3071	0.8546	0.3021	0.8559
28	0.3070	0.8544	0.3077	0.8558
29	0.3066	0.8547	0.3128	0.8540
30	0.3068	0.8547	0.3018	0.8566
31	0.3052	0.8543	0.3000	0.8566
32	0.3055	0.8546	0.2998	0.8563
33	0.3066	0.8544	0.3017	0.8564
34	0.3087	0.8547	0.3081	0.8539
35	0.3050	0.8543	0.3083	0.8562
36	0.3057	0.8546	0.2996	0.8565
37	0.3047	0.8548	0.3011	0.8562
38	0.3052	0.8547	0.3017	0.8562
39	0.3055	0.8545	0.3020	0.8559
40	0.3050	0.8546	0.3036	0.8561
41	0.3061	0.8549	0.3003	0.8566
42	0.3051	0.8546	0.2998	0.8562
43	0.3043	0.8548	0.3053	0.8550
44	0.3050	0.8546	0.2984	0.8559
45	0.3056	0.8548	0.3034	0.8560
46	0.3045	0.8546	0.3005	0.8563
47	0.3068	0.8544	0.3054	0.8560
48	0.3058	0.8547	0.3019	0.8564
49	0.3049	0.8547	0.3015	0.8567
50	0.3079	0.8545	0.3021	0.8563

Çeşitli makine öğrenmesi algoritmaları kullanılarak CICIDS2017 veriseti içerisindeki DDoS ataklarının tespit edilmesi noktasında, verilerin DDoS veya normal davranış şeklinde sınıflandırılması gerçeklenmiştir. Sınıflandırma sonuçlarına bağlı olarak kullanılan makine öğrenmesi yöntemleri karşılaştırılmıştır. Karşılaştırma neticesinde en yüksek doğruluk derin öğrenme yöntemi ile elde edilirken, en düşük doğruluk ile

sınıflandırma DVM ile elde edilmiştir. En kısa sürede sınıflandırma sonucu veren teknik ise KNN algoritması olmuştur. Bunun sebebi olarak KNN yönteminin sınıflandırma işleminde en basit şekli ile örneklerin sınıflandırılırken k en yakın komşusuna bakarak hızlıca bu sınıflandırmayı yapması neticesinde hızlı sonuç verdiği, diğer taraftan DVM 'nin sınıflandırmada örnekleri ayıran ideal hiperdüzlemi çizmede zorlandığı bununda veriseti içerisindeki örneklerin uzayda çok farklı ve çeşitli şekillerde dağılım göstermesinden kaynaklandığı, derin öğrenme yöntemi ile elde edilen doğruluğun gizli katman sayısına ve katmanlardaki nöron sayıları ile doğrudan ilişkili olduğu düşünülmektedir. Hali hazırda bu parametrelerin otomatik olarak ayarlanması yani literatürde ince ayar olarak geçen bu ayarın yapılabilmesi için bir optimizasyon algoritması bulunamamış fakat çeşitli yapay zeka yaklaşımları ile genetik algoritmalar veya arama algoritmaları ile bu işlemin kullanılan veritabanına bağlı olarak yapılabileceği düşünülmektedir.

5. TARTIŞMA VE SONUÇ

Saldırı Tespit Sistemleri üzerine hali hazırda birçok çalışma yapılmasına rağmen, verilerin büyük bir hızla dijitalleştiği, nesnelerin interneti uygulamalarının arttığı, bulut mimarilerinin popülerleştiği ve basit bir masaüstü uygulaması dahi olsa arka planda web servisler ile bağlantılı olarak çalıştığı ve internet ortamında veri sakladığı günümüzde gizliliğin ve güvenliğinin sağlanarak duygusal ve finansal kaybın yaşanmaması adına STS 'ler hakkında daha fazla çalışma yapılması gerekmektedir. Son zamanlardaki teknolojik ilerleme ve gelişmeler, kullanılan araçların çeşitlilikle artması saldırganların hedeflerinde ve kabiliyetlerinde de artış meydana getirmiştir. Dolayısı ile karmaşık ve yeni atak türleri ortaya çıkmıştır. Bu yeni ve karmaşık atak tülerine karşı savunma mekanizmalarından birisi olan STS 'lerin sürekli güncel ve çeşitli atak senaryolarına karşı dayanıklı olabilmesi şarttır. STS çalışmaları henüz olgunlaşmamış ve başlangıç aşamasındadır. Hali hazırda mevcut bir saldırı tespitinin nasıl yapılacağını belirten bir standart maalesef yoktur. Mevcut STS 'lerin dahi tamamen otomatik olarak değil en iyi ihtimalde bile ufak insan müdahalesi gerektiren türden olduğu görülmüştür. Ağ güvenliğini sağlamak ve daha güvenilir hale getirmek için belirli araç ve tekniklere ihtiyaç vardır. Bu yöntemlerin en etkili olan türleri ve makine öğrenmesi teknikleri ile saldırı tespit sistemlerinin geliştirilmesi bu tez kapsamında incelenmiş ve analiz edilmiştir.

Bu tez kapsamında öncelikle literatürde var olan saldırı tespit yöntemleri ve saldırı tespit sistemleri incelenerek analiz edilmiştir. Analiz ve incelemenin akabinde DVM, KNN, karar ağacı algoritmalarının performans ölçümleri CICIDS2017 veri setine bağlı olarak, veri setinden alınan 26167 DDoS ve 26805 normal davranıştan oluşan 28972 veri üzerinde, karşılaştırmalı biçimde gösterilmiştir. En iyi öznitelikler Fisher Score yöntemi kullanılarak belirlenmiştir. Böylece ilgisiz ve gereksiz olan öznitelikler verisetinden çıkarılmış ve gereksiz zaman ve alan kaybı engellenmiştir. Öznitelik sayısına bağlı olarak algoritmaların doğruluk oranları karşılaştırmalı olarak gösterilmiştir. Veri setinin %60 oranında küçültülmesine karşın, KNN 'in başarısı artmış, Karar Ağacının değişmemiş ve DVM 'nin başarısı ise azalmıştır. Sonuçlar, Karar Ağacı algoritmasının her iki 80 ve 30 öznitelik için aynı başarı oranına sahip olduğunu, DVM 'nin başarı oranının azalırken,

KNN ’nin başarı oranının arttığını göstermiştir.

Bu işlemlerin yanı sıra, CICIDS2017 verisetindeki DDoS ataklarını ve normal davranışları içeren 225711 verinin tamamı kullanılarak DDoS ataklarının tespiti için Derin Öğrenme ve yukarıda belirtilen sınıflandırma algoritmaları beraber kullanılarak STS modelleri oluşturulmuştur. Ayrıca TBA kullanılarak verisetinin boyutu azaltılmıştır. Boyut azaltma işlemi ile veriseti içerisindeki sınıflandırma için gereksiz ve ilgisiz olan öznitelikler ayrıştırılmış böylece allgoritmaların performansları arttırılmıştır. Oluşturulan STS modellerinin performans ölçümleri karşılaştırılmıştır. Karşılaştırmalar sonucunda en yüksek doğruluk derin öğrenme algoritması ile elde edilmiştir. Derin öğrenme tekniği içerisinde literatürde ince ayar olarak adlandırılan optimum parametrelerin seçilmesi noktasında hali hazırda genel geçer bir yöntemin bulunmadığı yine de bu işlemin çeşitli sezgisel yaklaşımlar, aramaalgoritmaları ve genetik algoritmalar ile gerçeklenebileceği düşünülmüştür. KNN en performanslı çalışan algoritma olmuştur. Bunun altında yatan sebebin ise algoritmanın sınıflandırma yaparken en yakın k komşuya göre yapmasındaki basitliğinden kaynaklandığı düşünülmektedir. DVM ’ler birçok sınıflandırma uygulamasında başarı ile kullanılmasına karşın bu çalışmada en düşük performansı gösteren algoritma olmuştur. Bunun nedeni ise veriseti içerisindeki verilerin uzayda çok dağınık bir yer kaplaması ve optimum hiperdüzlemin bulunmasındaki zorluktan kaynaklandığı düşünülmektedir. Algoitmaların doğruluklarının ve işlem sürelerinin karşılaştırmalı özet bir gösterimi Tablo 5.1’de gösterilmiştir.

Tablo 5.1: Sonuçların özet gösterimi

Algoritma	**Doğruluk**	**Süre(dk.)**
Derin Öğrenme	0.8562	6.96
KNN	0.8502	0.03
Karar Ağacı	0.8432	0.05
SVM	0.8160	23.51

Bu tez çalışması kapsamında DDoS ataklarının tespiti için CICIDS2017 veriseti kullanılarak farklı makine öğrenmesi teknikleri ile başarılı olarak çeşitli STS modelleri geliştirildiği düşünülmektedir. Geliştirilen STS modellerinin doğruluk oranlarının ve

performanslarının arttırılması için çeşitli matematiksel ve istatistiki farklı yaklaşımların kullanılabileceği gözlemlenmiştir. Yapılan çalışmalara ek olarak, CICIDS2017 veriseti içerisinde bulunan diğer atak türleride dahil olmak üzere, büyük veri teknolojilerinden Hadoop, Spark gibi teknolojiler kullanılarak saldırıların tespiti ve sınıflandırılmalarının yapılması, ayrıca CPU yerine GPU kullanılarak işlem sürelerinin karşılaştırılmalı sunulması bu tezin devamı niteliğinde yapılması planlanan çalışmalardandır.

KAYNAKLAR

[1] Teresa F Lunt. *Automated audit trail analysis and intrusion detection: A survey*. SRI International, Business Intelligence Program, 1989.

[2] Dorothy E Denning. An intrusion-detection model. *IEEE Transactions on software engineering*, SE-13(2):222–232, 1987.

[3] Mark Crosbie and Eugene H Spafford. Defending a computer system using autonomous agents. 1995.

[4] David Endler. Intrusion detection. applying machine learning to solaris audit data. In *Computer Security Applications Conference, 1998. Proceedings. 14th Annual*, pages 268–279. IEEE, 1998.

[5] Animesh Patcha and Jung-Min Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer networks*, 51(12):3448–3470, 2007.

[6] Hossain Shahriar and William Bond. Towards an attack signature generation framework for intrusion detection systems. In *Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence & Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2017 IEEE 15th Intl*, pages 597–603. IEEE, 2017.

[7] Anuja S Desai and DP Gaikwad. Real time hybrid intrusion detection system using signature matching algorithm and fuzzy-ga. In *Advances in Electronics, Communication and Computer Technology (ICAECCT), 2016 IEEE International Conference on*, pages 291–294. IEEE, 2016.

[8] Omar Al-Jarrah and Ahmad Arafat. Network intrusion detection system using attack behavior classification. In *Information and Communication Systems (ICICS), 2014 5th International Conference on*, pages 1–6. IEEE, 2014.

[9] J Jabez and B Muthukumar. Intrusion detection system (ids): Anomaly detection using outlier detection approach. *Procedia Computer Science*, 48:338–346, 2015.

[10] Neha G Relan and Dharmaraj R Patil. Implementation of network intrusion detection system using variant of decision tree algorithm. In *Nascent Technologies in the Engineering Field (ICNTE), 2015 International Conference on*, pages 1–5. IEEE, 2015.

[11] Mehdi Hosseinzadeh Aghdam and Peyman Kabiri. Feature selection for intrusion detection system using ant colony optimization. *IJ Network Security*, 18(3):420–432, 2016.

[12] Ujwala Ravale, Nilesh Marathe, and Puja Padiya. Feature selection based hybrid anomaly intrusion detection system using k means and rbf kernel function. *Procedia Computer Science*, 45:428–435, 2015.

[13] GV Nadiammai and M Hemalatha. Effective approach toward intrusion detection system using data mining techniques. *Egyptian Informatics Journal*, 15(1):37–50, 2014.

[14] Wei-Chao Lin, Shih-Wen Ke, and Chih-Fong Tsai. Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-based systems*, 78:13–21, 2015.

[15] Raman Singh, Harish Kumar, and RK Singla. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Systems with Applications*, 42(22):8609–8624, 2015.

[16] L Dhanabal and SP Shantharajah. A study on nsl-kdd dataset for intrusion detection system based on classification algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(6):446–452, 2015.

[17] Aastha Puri and Nidhi Sharma. A novel technique for intrusion detection system for network security using hybrid svm-cart. 2017.

[18] Shreya Dubey and Jigyasu Dubey. Kbb: A hybrid method for intrusion detection. In *Computer, Communication and Control (IC4), 2015 International Conference on*, pages 1–6. IEEE, 2015.

[19] Chin-Yang Tseng, Poornima Balasubramanyam, Calvin Ko, Rattapon Limprasittiporn, Jeff Rowe, and Karl Levitt. A specification-based intrusion detection system for aodv. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 125–134. ACM, 2003.

[20] Sandhya Peddabachigari, Ajith Abraham, Crina Grosan, and Johnson Thomas. Modeling intrusion detection system using hybrid intelligent systems. *Journal of network and computer applications*, 30(1):114–132, 2007.

[21] Farah Jemili, Montaceur Zaghdoud, and Mohamed Ben Ahmed. A framework for an adaptive intrusion detection system using bayesian network. In *Intelligence and Security Informatics, 2007 IEEE*, pages 66–70. IEEE, 2007.

[22] Dewan Md Farid and Mohammad Zahidur Rahman. Anomaly network intrusion detection based on improved self adaptive bayesian algorithm. *JCP*, 5(1):23–31, 2010.

[23] Yongli Zhang and Yanwei Zhu. Application of improved support vector machines in intrusion detection. In *e-Business and Information System Security (EBISS), 2010 2nd International Conference on*, pages 1–4. IEEE, 2010.

[24] Jun Wang, Taihang Li, and Rongrong Ren. A real time idss based on artificial bee colony-support vector machine algorithm. In *Advanced computational intelligence (IWACI), 2010 third international workshop on*, pages 91–96. IEEE, 2010.

[25] Qi Mu, Yikun Chen, and Yongjun Zhang. Incremental svm algorithm to intrusion detection base on boundary areas. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 198–201. IEEE, 2012.

[26] M Bahrololum, E Salahi, and M Khaleghi. Machine learning techniques for feature reduction in intrusion detection systems: A comparison. In *Computer Sciences and Convergence Information Technology, 2009. ICCIT'09. Fourth International Conference on*, pages 1091–1095. IEEE, 2009.

[27] Ammar Alazab, Michael Hobbs, Jemal Abawajy, and Moutaz Alazab. Using feature selection for intrusion detection system. In *Communications and Information Technologies (ISCIT), 2012 International Symposium on*, pages 296–301. IEEE, 2012.

[28] Vikas Sharma and Aditi Nema. Innovative genetic approach for intrusion detection by using decision tree. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, pages 418–422. IEEE, 2013.

[29] Guisong Liu, Zhang Yi, and Shangming Yang. A hierarchical intrusion detection model based on the pca neural networks. *Neurocomputing*, 70(7-9):1561–1568, 2007.

[30] Xingchao Gong and Xin Guan. Intrusion detection model based on the improved neural network and expert system. In *Electrical & Electronics Engineering (EEESYM), 2012 IEEE Symposium on*, pages 191–193. IEEE, 2012.

[31] Chih-Fong Tsai and Chia-Ying Lin. A triangle area based nearest neighbors approach to intrusion detection. *Pattern recognition*, 43(1):222–229, 2010.

[32] KDD Cup. Data/the uci kdd archive, information and computer science. *University of California, Irvine*, 1999.

[33] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the kdd cup 99 data set. In *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*, pages 1–6. IEEE, 2009.

[34] Tusiad. Türkiyedeki E-Ticaret Pazarına Dair Kapsamlı Rapor. [Online; accessed 25 April 2017].

[35] Robin Sommer. Bro: An open source network intrusion detection system. In *DFN-Arbeitstagung über Kommunikationsnetze*, pages 273–288. Citeseer, 2003.

[36] Canadian Institute for Cybersecurity. CICIDS2017 intrusion detection evaluation dataset(cicids2017). http://www.unb.ca/cic/datasets/ids-2017.html. Online; accessed 07 July 2017.

[37] Amirhossein Gharib, Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. An evaluation framework for intrusion detection dataset. In *Information Science and Security (ICISS), 2016 International Conference on*, pages 1–6. IEEE, 2016.

[38] Quanquan Gu, Zhenhui Li, and Jiawei Han. Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725*, 2012.

[39] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[40] Vladimir Vapnik. *Statistical learning theory. 1998*. Wiley, New York, 1998.

[41] Taşkın Kavzoğlu and İsmail Çölkesen. Destek vektör makineleri ile uydu görüntülerinin sınıflandırılmasında kernel fonksiyonlarının etkilerinin incelenmesi. *Harita Dergisi*, 144(7):73–82, 2010.

[42] Steven J Phillips, Robert P Anderson, and Robert E Schapire. Maximum entropy modeling of species geographic distributions. *Ecological modelling*, 190(3-4):231–259, 2006.

[43] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[44] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[45] Yoshua Bengio. Deep learning of representations: Looking forward. In *International Conference on Statistical Language and Speech Processing*, pages 1–37. Springer, 2013.

[46] Maryam M Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1):1, 2015.

[47] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[48] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.

[49] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, pages 3–10, 1994.

[50] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[51] Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5):1–41, 2007.

ÖZGEÇMİŞ

Kişisel Bilgiler	
Adı Soyadı	Doğukan AKSU
Doğum Yeri	Çarşamba/Samsun
Doğum Tarihi	19.09.1992
Uyruğu	[X] T.C. [ ] Diğer:
Telefon	02124737070-17458
E-Posta Adresi	d.aksu@istanbul.edu.tr
Web Adresi	http://aves.istanbul.edu.tr/d.aksu

Eğitim Bilgileri	
Lisans	
Üniversite	İstanbul Üniversitesi
Fakülte	Mühendislik Fakültesi
Bölümü	Bilgisayar Mühendisliği
Mezuniyet Yılı	2015

Yüksek Lisans	
Üniversite	İstanbul Üniversitesi
Enstitü Adı	Fen Bilimleri
Anabilim Dalı	Bilgisayar Mühendisliği Anabilim Dalı
Programı	Bilgisayar Mühendisliği Programı

Makale ve Bildiriler

Makaleler

Aksu, D. and Aydin, M.A., 2018, "Detecting DDoS Attacks Using by Deep Learning Approach with Comparative Analysis of Various Machine Learning Algorithms and PCA Dimension Reduction Technique, *Entropy*, SCI(gönderildi).

Bildiriler

Aksu, D., Abdulwakil, A. and Aydin, M.A., 2017, Detecting Phishing Websites Using Support Vector Machine Algorithm, *PressAcademia Procedia*, 5(1), pp.139-142.

Aksu, D. and Aydin, M.A., 2017, Human Computer Interaction by Eye Blinking on Real Time, *Computational Intelligence and Communication Networks (CICN), 2017 9th International Conference on*, pp. 135-138, IEEE.

Aksu, D., Turgut, Z., Ustebay, S. and Aydin, M.A., 2017, Phishing Analysis of

Websites Using Classification Techniques, *International Telecommunications Conference (itelcon)*, Springer.

Aksu, D.,Ustebay, S., Aydin, M.A. and Atmaca, T., 2018, "Intrusion Detection with Comparative Analysis of Supervised Learning Techniques and Fisher Score Feature Selection Algorithm, *ISCIS*, Conference(kabul edildi).