IoT için Makine Öğrenmesi Tabanlı Saldırı Tespiti

Ayça Nur Kahya

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Haziran 2022

Machine Learning Based Intrusion Detection for IoT

Ayça Nur Kahya

MASTER OF SCIENCE THESIS

Department of Computer Engineering

June 2022

IoT için Makine Öğrenmesi Tabanlı Saldırı Tespiti

Ayça Nur Kahya

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Bilgisayar Mühendisliği Anabilim Dalı
Robotik Bilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır

Danışman: Dr.Öğr.Üyesi Esra N. Yolaçan

Haziran 2022

ETİK BEYAN

Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kılavuzuna göre, Dr. Öğr. Üyesi Esra N. Yolaçan danışmanlığında hazırlamış olduğum "IoT için Makine Öğrenmesi Tabanlı Saldırı Tespiti" başlıklı YÜKSEK LİSANS tezimin özgün bir çalışma olduğunu; tez çalışmamın tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; tezimde verdiğim bilgileri, verileri akademik ve bilimsel etik ilke ve kurallara uygun olarak elde ettiğimi; tez çalışmamda yararlandığım eserlerin tümüne atıf yaptığımı ve kaynak gösterdiğimi ve bilgi, belge ve sonuçları bilimsel etik ilke ve kurallara göre sunduğumu beyan ederim. 14/06/2022

Ayça Nur Kahya

İmza

ÖZET

İnternet ve iletişim teknolojilerinin gelişimi ile internete bağlanabilen akıllı cihazların sayısı gün geçtikçe artmaktadır. Bu cihazlar, bilgisayar ve sunucularda olduğu gibi savunma mekanizmalarına sahip değildir. IoT ağları, günümüzde farklı nitelikteki kritik ortamları izlemek için giderek daha popüler hale gelmektedir. IoT cihazlarının çok sayıda bulunması nedeniyle, bu ağların ve cihazların güvenliği kritik bir konudur. IoT için en çok kullanılan mesajlaşma protokollerinden olan MQTT protokolü, asenkron iletişim gerektiren yazılım uygulamalarında kullanılmaktadır. IoT cihazların ve ağların güvenliğini sağlamak için başvurulan yaklaşımlardan biri de ağa yapılan saldırıların tespit edilmesidir. MQTT protokolüne yapılan saldırıların tespit edilmesi için hem ağ katmanında hem de uygulama katmanında gerçekleşen verilerin önemi vardır. Toplanan verilerin çeşitli yöntemlerle değerlendirilecek bir saldırı olup olmadığı noktasında sonuca ulaştırılması gerekmektedir. Günümüzde bu amaçla makine öğrenimi gibi yenilikçi algoritmalara dayalı olarak siber saldırıların tespiti yapılmaktadır.

Bu çalışmada IoT cihazlarından gelen veriler makine öğrenmesi ve derin öğrenme ile sınıflandırılarak bir saldırı tespiti gerçekleştirilmiştir. Geliştirilen çoklu sınıflandırma sayesinde, sistemde meydana gelen davranışlar saldırı ise ne tür bir saldırı olduğunu da tespit edilebilmiştir. Çalışmada, bir MQTT broker'ında konulara abone olan IoT cihazlarını kullanarak sistemin bir prototipi oluşturulmuştur. Saldırı tespitini uygulamak ve makine öğrenimi modellerini eğitmek için MQTTset veri kümesinin nasıl kullanılabileceği bu çalışmada açıklanmıştır. MQTTset veri kümesine, bu çalışmada oluşturulan IoT ev ağından elde edilen verilerin eklenmesi ile veri kümesi genişletilmiştir, bu sayede dengeli bir veri kümesi elde edilmiştir. Mevcut saldırı tiplerine yeni saldırı çeşidi de eklenerek veri kümesinin zenginleştirilmesi hedeflenmiştir. Genişletilen veri kümesinin analizlerinde çeşitli makine öğrenmesi algoritmaları ve derin öğrenme algoritmalarına yer verilmiştir. Sistemin MQTT protokolüne gerçekleştirilen saldırılar kapsamında deneysel değerlendirmesi ve karşılaştırmalar bu çalışma kapsamında sağlanmıştır.

Anahtar Kelimeler: IoT Güvenliği, MQTT Protokolü, IoT Saldırıları, Saldırı Tespiti, Nesnelerin İnterneti, Siber Saldırılar, Makine Öğrenmesi, Derin Öğrenme, MQTTset veri kümesi

SUMMARY

With the development of the Internet and communication technologies, the number of smart devices that can connect to the Internet is increasing day by day. These devices don't have defense mechanisms like computers and servers. Novadays, IoT networks are becoming more and more popular for monitoring different types of critical devices. Due to the large number of IoT devices, the security of these networks and devices is a critical issue. MQTT protocol, one of the most used messaging protocols for IoT, is used in software applications which require asynchronous communication. One of the approaches used to ensure the security of IoT devices and networks is to detection of attacks on the network. The data that occurs at both the network layer and the application layer is important for detecting attacks on the MQTT protocol. The collected data should be evaluated by various methods and the conclusion should be reached at the point of whether there is an attack or not. For this purpose, cyber attacks are detected based on innovative algorithms such as machine learning.

In this study, attack detection was performed by classifying the data from IoT devices with machine learning and deep learning. Thanks to the developed multi-classification system, the changes in the system and the type of attack could be detected. In this Thesis, a prototype of the system is created by using IoT devices subscribed to topics in MQTT broker. This thesis explains how the MQTTset dataset can be used to implement intrusion detection and training of machine learning models. The dataset has been expanded by adding the data obtained from the IoT home network to the MQTTset dataset, thus a more balanced dataset has been obtained. It is aimed to enrich the dataset by adding a new attack type to the existing attack types. Machine learning algorithms and deep learning algorithms are included in the analysis of the expanded dataset. Experimental evaluation of the system's attacks on the MQTT protocol and their comparisons are provided in this thesis.

Keywords: IoT Security, MQTT Protocol, IoT Attacks, Intrusion Detection, Internet of Things, Cyber Attacks, Machine Learning, Deep Learning, MQTTset Dataset

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

Şekil **Sayfa**

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler	Açıklama
%	Yüzde
°C	Derece Santigrat
≥	Büyük Eşittir
±	Değer Toleransı

Kısaltmalar	Açıklama
6LowPAN	IPv6 over Low-Power Wireless Personal Area Networks
AB	AdaBoost
AC	Alternating Current
ACC	Accuracy
ACK	Acknowledgement
ADR	Attack Detection Rate
APV	Attack Predictive Value
ARFF	Attribute-Relation File Format
ARM	Acorn RISC Machine
ARP	Address Resolution Protocol
AUC	Area Under the Curve
BLE	Bluetooth Low Energy
C&C	Command and Control
CLI	Command Line Interface
CNN	Convolutional Neural Network
CoAP	Constrained Application Protocol
CSV	Comma Separated Values
DBM	Deep Boltzmann Machine
DDoS	A Distributed Denial of Service

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
DL	Deep Learning
DNN	Deep Neural Network
DNS	Domain Name System
DoS	Denial of Service Attack
DT	Decision Tree
DVM	Destek Vektör Makinesi
ELM	Extreme Learning Machines
EMQTT	Erlang Message Queuing Telemetry Transport
FAR	False Alarm Rate
FTP	File Transfer Protocol
GB	Gradient Boosting
GBM	Gradient Boosting Machines
GHz	Gigahertz
GPIO	General Purpose Input/Output
GRU	Gated Recurrent Unit
GS	Gradient Search
GTCS	Game Theory and Cyber Security
HDMI	High Definition Multimedia Interface
HT	Hoeffding Tree
HTTP	Hyper-Text Transfer Protocol
IBM	International Business Machines
IDE	Integrated Development Environment
IEEE	Institude of Electrical and Electronics Engineers
IMAP	Internet Message Access Protocol
IoT	Internet of Things
IP	Internet Protocol

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
IPV4/6	Internet Protocol Version 4/6
IPv6	Internet Protocol Version 6
IQR	Inter Quartile Range Technique
ITU	Uluslararası Telekomünikasyon Birliği
JDBC	Java Database Connectivity
Kb	Kilobayt
KDD	Knowledge Discovery Dataset
KNN	K-Nearest Neighbors
LAN	Local Area Network
LDDR	Low Power Double Data Rate
LOIC	Low Orbit Ion Cannon
LR	Logistic Regression
LSTM	Long Short-Term Memory
MCC	Mathews Correlation Coefficient
MIPI	Mobile Industry Processor Interface
MITM	Man In The Middle Attack
ML	Machine Learning
MLP	Multilayer Perceptron
MQTT BF	MQTT Brute Force Attack
MQTT	Message Queuing Telemetry Transport
MQTTSA	MQTT Security Assistant
NB	Naive Bayes
NFC	Near Field Communication
NLP	Natural Language Processing
NodeMCU	NodeMicro-Controller Unit
NPV	Normal Predictive Value

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
NSL KDD	Network Service Layer- Knowledge Discovery in Databases
OCSVM	One Class Support Vector Machine
P2P	Peer to Peer
PCA	Principal Component Analysis
PCAP	Packet Capture Data
PRC	Precision Recall Curve
QoS	Quality of Service
R2L	Root to Local Attack
REST	Representational State Transfer
RF	Random Forest
RFID	Radio Frequency Identification
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RPL	IPv6 Routing Protocol
RS	Random Search
SD	Secure Digital
SDA	Slow DoS Attack
SDRAM	Synchronous Dynamic Random Access Memory
SlowITE	Slow DoS against Internet of Things Environments
SO GAAL	Single Objective Generative Adversarial Active Learning
SQL	Structured Query Language
SSH	Secure Shell
SVC	Support Vector Classification
SVM	Support Vector Machine

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
SVR	Support Vector Regression
SYN	Synchronize
TCP SYNC	Synchronized Transmission Control Protocol
TCP	Transmission Control Protocol
TLS	Transport Layer Security
U2R	User to Root Attack
UDP	User Datagram Protocol
URL	Uniform Resource Locator
Vd	Ve diğerleri
WEKA	Waikato Environment for Knowledge Analysis
XGB	Extreme Gradient Boosting
XGBoost	Gradient Boosting
XMPP	Extensible Messaging and Presence Protocol
XSS	Cross-site Scripting Attack
YSA	Yapay Sinir Ağları

1. GİRİŞ VE AMAÇ

Teknolojinin gelişmesiyle birlikte internete bağlanan cihaz sayısı da artış göstermektedir. Nesnelerin İnterneti ile ilgili literatürde birçok tanım yer almaktadır. ITU'nun yapmış olduğu tanımıyla IoT, herhangi bir zamanda herhangi bir yerde her nesnenin birbirine bağlanabileceği bir teknolojidir. Britanyalı bilim adamı Kevin Ashton "Nesnelerin İnterneti" terimini ilk kez 1999 yılında Procter&Gamble şirketi için hazırlanan sunumda kullanmıştır. Bu sunumda RFID teknolojisinin faydaları anlatılmış ve IoT'nin çıkış noktasının RFID etkileri olarak kabul edilmiştir (Khalil vd., 2017). Ardından 2005 yılında ITU'nun konuya dair ilk raporu yayınlanmıştır. ITU; IoT'nin öğe tanımlama, algılayıcı ve kablosuz algılayıcı ağlar, gömülü sistemler ve nanoteknoloji gibi teknolojik geliştirmeleri bir araya getirerek dünyadaki objeleri algısal olarak bağlayacağını ileri sürmüştür (Yiğitbaşı, 2011). IBM'nin CEO'su Samuel J. Palmisano Smart Planet kavramını önermiştir (Bozdoğan, 2015). 2009 Ekim ayında düzenlenen CASAGRAS (2010) konferansında, gelişen internet ile bütünleşme ihtiyacı, istikrarlı, kararlı, ölçeklenebilir iletişim teknolojisi olması anlamında internet protokolünden yararlanılması ve IPv6 kullanımı vurgulanmıştır (Yiğitbaşı, 2011). IoT teknolojisi ilk ortaya atıldığı tarihten itibaren günümüze kadar önemli gelişmeler sağlamıştır. IoT teknolojisinde her nesne RFID, NFC, sensörler gibi algılama yöntemleri aracılığı ve Wifi, Wimax, Zigbee, Bluetooth, kızıl ötesi vb. kablosuz iletişim teknikleri ile nesneler hakkında bilgiler edinilebilmektedir. Bunun sayesinde nesnelerin sıcaklık, basınç, ışık, uzaklık gibi birçok durumunu sensörler aracılığıyla gözlemleyebilir, nesnelerin koşullara göre karar vermesi sağlanabilir. Nesneler bu bilgileri ve durumları saklayabilir veya paylaşabilirler. Ayrıca tüm elektronik cihazların uzaktan kontrol edilmesi ve izlenmesi IoT sayesinde mümkündür. IoT sistemler kullanıcılarına fayda sağlamak üzere fiziksel cihazlara iletişim kabiliyetleri katmıştır. Bu iletişimin üretimde kullanılmasıyla maliyet anlamında fayda sağlandığının görülmesiyle IoT uygulamalarının pazar payı artmaya devam etmektedir (Adıgüzel, 2016). Nesnelerin İnterneti teknolojisi çevre ve altyapı izleme, endüstriyel uygulamalar, enerji yönetimi, medikal servisler, bina otomasyonu, taşımacılık gibi birçok alanda kullanılmaktadır (Türkay, 2018). Nesnelerin İnternetinde artan cihaz sayısı ve kullanım alanının da genişlemesi bazı tehditleri de beraberinde getirmektedir. Siber saldırılar, bilgi sistemlerinde yetkisiz olarak değiştirme ve imha eylemleri,

bilginin bütünlüğüne yönelik her türlü tehdit olarak tanımlanmaktadır. İzinsiz Giriş Tespit Sistemleri ise bilgisayar güvenliğini sağlamaya yardımcı olmak için kötü amaçlı ağ etkinliklerini tanımlayan sistemlerdir. İnternete bağlanan cihazların sayısı artış gösterdiğinden tehditlerin sayısı da bununla birlikte artış göstermektedir. Artan tehditlerin sayısı, bilgisayar güvenliği konusunu yüksek öneme sahip bir görev haline getirmektedir. Siber saldırılarla mücadele için anti virüs sistemleri, güvenlik duvarları vb. çeşitli donanım ve yazılım çözümleri geliştirilmektedir. Bu çözümlerden biri de Saldırı Tespit Sistemleridir (Çakır, 2019). Saldırı tespitinin amacı; bilgisayar sistemlerinin hem sistem içindekiler hem de dışarıdan sızanlar tarafından yetkisiz ve kötüye kullanımını tespit etmektir. Genel olarak, IDS'ler izinsiz girişleri tespit etmek için istatistiksel, anormallik ve kural tabanlı kötüye kullanım modelleri kullanılmaktadır (Mukherjee vd., 1994).

Birçok IoT cihaz iletişimlerinde HTTP protokolünü kullanmaktadır. HTTP protokolü büyük ağ kaynaklarını kullanarak ağda gecikmelere sebep olmaktadır. Alternatif olarak MQTT protokolü, hizmet kalite servisi ile daha az yük ve minimum ağ kaynakları kullanmaktadır. Açık kaynaklı MQTT protokolü, makineden makineye ve düşük ağ bant genişliğinin mevcut olduğu IoT uygulamalarında sıklıkla kullanılır. Yayınla/abone ol modeli geleneksel istemci-sunucu mimarisine bir alternatif sunar (Siddharthan vd., 2022).

Kablolu ve kablosuz ağlarda saldırı tespitinde, makine öğrenmesi ve derin öğrenme yöntemleri kullanılmaktadır. Makine öğrenmesi, eğitim verilerinden öğrenilen özelliklere dayalı olarak sınıflandırma ve regresyon işlemlerine dayanmaktadır. Derin öğrenme, veri öğrenmenin karakterizasyonuna dayanan bir makine öğrenmesi yöntemidir. Derin öğrenme, analitik öğrenme için insan beynini simüle eden bir sinir ağıdır. Görüntü, ses, metin vb. verileri yorumlamak için insan beyni mekanizmasını taklit etmektedir (Xin vd., 2018). Yaygın olarak kullanılan ML algoritmalarından; YSA, Karar Ağaçları, Rastgele Orman, Naive Bayes, XGBoost, MLP, AdaBoost ve Hoeffding Ağacı bu çalışma kapsamında kullanılmıştır. DL yöntemlerinde; DBM, CNN ve LSTM yaygın olarak kullanılan modellerdir. Seçilecek katman ve düğüm sayısı gibi, aynı zamanda modeli ve entegrasyonu geliştirmek gibi birçok parametre vardır. Eğitim tamamlandıktan sonra farklı açılardan değerlendirilmesi gereken alternatif modeller bulunmaktadır (Xin vd., 2018).

Siber güvenlik kapsamında saldırı tespitinde birçok veri kümesi bulunmaktadır. Fakat bu veri kümeleri IoT ortamlarına içermiş oldukları nitelikler sebebiyle nadiren uygundur. Açık

kaynaklı olan MQTTset veri kümesindeki bazı özniteliklerin kaldırılması ve bu çalışmada oluşturulan test ortamından elde edilen verilerin eklenmesiyle değerlerde artışın gözlenmiştir. Derin öğrenme yöntemi olan LSTM'in bu çalışmaya entegre edilmesi ve farklı ML sınıflandırma algoritmalarının da eklenmesi göz önünde bulundurulduğunda bu tez çalışmasının literatüre katkı olması amaçlanmıştır.

Bu tez çalışmasında IoT ev ağı oluşturularak saldırı tespiti gerçekleştirilmiştir. Nesnelerin İnternetinde yaygın olarak kullanılan MQTT protokolü ile ev ağındaki cihazların birbirleriyle iletişim kurması sağlanmıştır. Makine öğrenmesi algoritmalarının kullanılarak yapılan çalışmalar bu tez çalışmasında incelenmiştir. MQTT özniteliklerini içeren ve saldırılara odaklanan veri kümelerinin karşılaştırması bu çalışma kapsamında yapılmıştır. MQTT özniteliklerini içeren MQTTset veri kümesinde, veri ön işleme ve öznitelik seçimi işlemleri yapılarak sonuçlar analiz edilmiştir. MQTTset veri kümesi genişletilerek iki sınıflı ve çok sınıflı sınıflandırma ile makine öğrenmesi yöntemleri uygulanmıştır. Genişletilen veri kümesinde DL yöntemlerinden LSTM ile saldırı tespiti yapılmış ve sonuçlar analiz edilmiştir. Ayrıca katmanlara yapılan saldırılar hakkında literatürde araştırmalar yapılarak özet niteliğinde tablolar halinde sunulmuştur.

Bu tez çalışmasının 2. Bölümünde saldırı tespitinde kullanılan, IoT izlerini ve MQTT özniteliklerini içeren çalışmalar ile ilgili literatür araştırmasına yer verilmiştir. 3. Bölümde Saldırı Tespit Sistemleri ve makine öğrenmesi yöntemleri anlatılmıştır. Yapılan çalışma ve yöntemler tüm aşamalarıyla 4. Bölümde anlatılmıştır. 5. Bölümde çalışma sonuçlarına ilişkin analizlere, karşılaştırmalara ve grafiklere yer verilmiştir. Sonuçların değerlendirilmesi ile gelecek çalışmalar 6. Bölümde sunulmuştur.

2. LİTERATÜR ARAŞTIRMASI

IoT sistemlerinde Saldırı Tespit Sistemleri konuları sürekli olarak saldırı tehditlerinin ortaya çıkması sebebiyle oldukça geniş bir alanı kapsamaktadır. Bu bölümde; IoT sistemlere yapılan saldırıların tespiti, veri kümeleri, makine öğrenmesi ve yapay zekâ konularını içeren çalışmalar incelenmiştir. Literatür araştırması IoT izlerini ve MQTT özniteliklerini içeren çalışmalar olmak üzere 2 kısımda incelenmiştir. MQTT özniteliklerini içeren çalışmalarda IoT izleri ve MQTT öznitelikleri yer alırken IoT izlerini içeren çalışmalarda sadece IoT izleri yer almaktadır.

2.1. IoT İzlerini İçeren Çalışmalar

Kaya vd. (2016) ML bazında kullandıkları Bayes Ağları, DVM, YSA, K en yakın komşu ve Karar Ağaçları algoritmaları ile KDDCup99 veri kümesinin saldırı tespitinde başarısını incelediler. KDDCup99 veri kümesinde yer alan saldırı tiplerinden DoS, Probe, U2R ve R2L veri sayısı ve saldırı türü bazında karşılaştırdılar. Yazarların gerçekleştirdikleri çalışmalar sonucunda normal davranışları ayırt etmede Karar Ağaçları başarılı olurken, DoS saldırıları tespitinde YSA ve Karar Ağaçları, U2R saldırılarında K en yakın komşu, R2L saldırılarında K en yakın komşu ve Karar Ağaçları, PROBE saldırılarında YSA ve Karar Ağaçları diğerlerine göre daha iyi sonuçlar verdiğini gördüler.

Canedo vd. (2016) YSA eğitilmesi ile IoT sisteminde gönderilen verilerin geçerli ve geçersiz olup olmadığını tespit etmek için yaptıkları çalışmada başarılı sonuçlar elde etmiştir. Test ortamında ESP8266 Wifi modülü, sıcaklık sensörü ve Arduino Uno kullandılar. Ağ geçidi uygulamak için de 10 uç cihaza, Raspberry Pi cihazı bağladılar. Bununla birlikte, çeşitli veri kurcalama saldırıları içeren zenginleştirilmiş veri kümeleri, pratik ayarlarda yüksek doğruluğu koruyup koruyamayacağını veya diğer gelişmiş öğrenme algoritmalarının gerekli olup olmadığını doğrulamak için YSA'yı eğitmek ve test etmek için kullanmışlardır. IoT sistemindeki anormallikleri tespit etmek için ML tabanlı çözümler önermişlerdir. Sinir ağları ile ağı geçersiz veri noktalarını tespit edecek şekilde eğitmişlerdir.

Manzoor vd. (2017) IoT ağlarında izinsiz giriş tespit sistemi için iki seviyeli bir anormal aktivite algılama modeli önerdikleri çalışmada, Rastgele Orman sınıflandırıcı kullanarak IoT Botnet veri kümesini test ettiler. Sundukları çalışmada YSA'nın görevi, kaynak kullanımını iyileştirmek ve karmaşıklığını azaltmaktır. YSA yöntemleri ile veri kümesinde, saldırının tespitinde tahmin yüzdesi daha yüksek olduğunu saptamışlardır.

Aytan vd. (2018) tarafından KDDCup99 veri kümesi kullanılarak Weka aracındaki ML algoritmaları ile saldırıların tespiti çalışmalarını yapmışlardır. ML algoritmalarından Geri Yayılma, Karar Ağacı, Ripper Kuralı ve Rastgele Orman Sınıflandırıcısını kullandılar. Yaptıkları çalışmada KDD99 veri kümesinde eğitim için 10.000 kayıt almışlardır. Bu kayıtlardan 5.000 normal veri, 5.000 saldırı verisidir. Test veri kümesi için ise 5.000 kayıt seçtiler. Bunların 2.500 tanesi normal, geri kalanı saldırı kaydıdır. Saldırı kayıtları DoS ve Bilgi Tarama Saldırılarından oluşmaktadır. Ayrıca diğer saldırı türlerini veri kümesinden silmişlerdir. Çalışmaları sonucunda en iyi saldırı tespiti ve eğitim süresinin Rastgele Orman Algoritmasına ait olduğu sonucuna vardılar.

Kaynar vd. (2018) KDDCup99 veri kümesi üzerinde Ki-Kare, Bilgi Kazancı, Kazanım Oranı, Gini Katsayısı, oneR, ReliefF, Genetik, İleriye Doğru ve Geriye Doğru öznitelik seçim algoritmaları ile yeni bir veri kümesi elde etmişlerdir. Veri kümesindeki sözel nitelikler, sayısal verilere dönüştürülmüş ve öznitelikler için normalizasyon işlemi Python kütüphaneleri kullanılmış ve yeni veri kümesi yazarlar tarafından oluşturulmuştur. K en yakın komşu, DVM, ELM ile analizler gerçekleştirmişlerdir. Tüm analizleri Matlab ortamında ve 10 kat çapraz doğrulama yöntemi ile deneyleri gerçekleştirdiler. Analizler sonucunda modeller daha hızlı çalışmış ve başarı oranının arttığını gördüler. Sınıflandırma algoritması olarak K en yakın komşu kullanan modellerin; ELM veya DVM kullanan modellere göre daha iyi başarı oranları elde etmişlerdir. ELM kullanan modellerin en hızlı çalıştığı, DVM kullanan modellerin ise diğer iki modele göre çok yavaş çalıştığı sonucuna vardılar.

Halimaa vd. (2019) IDS'in değerlendirilmesi için NSL–KDD veri kümesi ile DVM ve Naive Bayes sınıflandırma yöntemlerini kullanmıştır. Etiketleme işlemelerinden sonra nominal özniteliği, ikili özniteliğe dönüştürmek için ön işleme aşamasında kullanmışlardır. Ayrıca sayısal olmayan değerleri veri kümesinden kaldırdılar. Kullandıkları WEKA aracı örnekleri rastgele karıştırdıktan sonra 19.000 örnek toplamışlardır. Veri kümesinin performansını iyileştirmek için öznitelik indirgeme yönteminde CfsSubSetEval kullandılar. Bu yöntemler

sonucunda doğruluk oranları ve yanlış sınıflandırma oranlarını hesaplamışlardır. PCAP dosyalarından Paket, Tek yönlü, Çift yönlü olmak üzere 3 özellik çıkardılar. Ağırlıklı ortalama geri çağırma; paket tabanlı özellikler için ≥%75,31, tek yönlü akış özellikleri için ≥% 93.77 ve çift yönlü akış özellikleri için ≥% 98.85'e yükselme görmüşlerdir.

Şahingöz vd. (2019) ML algoritmalarından Karar Ağaçları, Rastgele Orman, K en yakın komşu, Adaboost, Gradyan Artırma, Doğrusal Ayrımcılık Analizi ve YSA kullanımı ile doğruluk, eğitim süreleri, çalıştırma sürelerini analiz ettiler. ML algoritmalarını uyguladıkları veri kümesi NSL-KDD'dir. Veri kümesi ön işlemeden geçirilerek sözel değerleri sayısal hale getirmişlerdir. Veri kümesini kullanarak, K Katlamalı Çapraz Doğrulama ile K değerini 5 seçtiler. K değerine bağlı olarak en yüksek doğruluk oranını AdaBoost vermiştir. Ancak eğitim süresi ve çalışma zamanı göz önüne aldıklarında Karar Ağacı Algoritmasının daha yüksek performans gösterdiğini gördüler.

DL ve ML sınıflandırıcılarının performanslarını kıyaslamak için Vinayakumar vd. (2019) birçok veri kümesi üzerinde test işlemleri gerçekleştirdiler. En iyi performans gösteren veri kümesinin KDDCup99 olduğu hiper paremetre seçim yöntemleri ile belirlenmiştir. DNN'lerin tüm deneyleri, [0.01–0.5] aralığında değişen öğrenme oranıyla analiz ettiler. Kıyaslama yapmak için NSL-KDD, UNSW-NB15, Kyoto, WSN-DS ve CICIDS 2017 gibi diğer veri kümelerine uyguladılar. Önerdikleri DNN modeli, IDS verilerini birçok gizli katmana geçirerek soyut ve yüksek boyutlu özellik gösterimini öğrenmiştir. DL'lerin, ML sınıflandırıcılarına göre daha iyi performans gösterdiğini analizler sonucunda ortaya koydular. Ayrıca ağ trafiğini ve ana bilgisayar düzeyindeki olayları etkin bir şekilde izlemek için gerçek zamanlı olarak kullanılabilen scale-hybrid-IDS-AlertNet adlı yüksek düzeyde ölçeklenebilir ve hibrit bir DL çerçevesi önerdiler.

Mahfouz vd. (2020) ağ trafiğindeki izinsiz girişleri belirlemek için bazı makine öğrenimi sınıflandırıcılarının kapsamlı bir analizini sağlamışlardır. Gerçekleştirdikleri deneysel ortamda iyi ve kötü amaçlı trafik verileri 8 günde oluşturulmuştur. Kötü amaçlı trafik için Kali Linux kullanarak sanal olarak kurulmuş 2 tane makineden saldırılar gerçekleştirmişlerdir. Kurban ağda ise Windows 2012, Windows 8.1, Windows 7, Ubuntu 16.4 ve Metasploitable 2 Linux sanal makinelerini tercih ettiler. Kali Linux makinelerinden gerçekleştirilen saldırılar; Botnet, Kaba Kuvvet, DDoS ve Sızma Saldırıları'dır. Hiper parametre optimizasyonu tekniklerinden GS, RS ve Bayesian kullandılar. Çalışmaların sonucunda GTCS adlı veri kümesini önerdiler. Ayrıca,

Saldırı Tespit Sistemleri'nde doğruluk ve yanlış alarm oranı sorununu ele almak için farklı öğrenme paradigmalarına sahip çoklu sınıflandırıcılardan oluşan bir topluluk ve uyarlanabilir sınıflandırıcı modeli önermektedir.

Kiran vd. (2020) verileri normal ve değiştirilmiş kötü amaçlı saldırı verileri olarak sınıflandırmak için dört makine öğrenmesi algoritması ile testler yaptılar. Ağda farklı heterojen cihazlar ile ML modeli oluşturulurken çeşitli verilerin olmasını göz önünde bulundurdular. NodeMCU ESP8266, DHT11 sensörü ve kablosuz yönlendirici kullanılarak IoT ortamını simüle eden bir test yatağı oluşturdular. Saldırı sistemi, koklama ve zehirleme saldırıları gerçekleştiren bir dizüstü bilgisayar sistemi kullanılarak oluşturulmuştur. Normal aşamada, sensör değerleri NodeMCU tarafından yakalanır ve normal veriler olarak depolanan Think Speak sunucusuna iletilmiştir. Saldırı sisteminde, saldırgan verileri gizlice ele geçirerek, NodeMCU ile Think Speak sunucusu arasında iletildiğinde verileri değiştirir. Saldırı aşamasında, ağda ARP Zehirlenmesi kullanarak Ortadaki Adam Saldırısı gerçekleştirdiler ve yakaladıkları verileri saldırı verisi olarak etiketlediler. Naive Bayes, DVM, Karar Ağacı, AdaBoost gibi makine öğrenimi sınıflandırıcıları, verileri normal ve saldırı sınıflarına ayırmak için kullanmışlardır. Yazarlar Sensör 480 veri kümesini oluşturmuşlardır. Sundukları çalışmada verilerin %80'i ile sınıflandırıcı modeli oluşturulmuş ve kalan %20'si sınıflandırıcının performansını test etmek için kullanılmıştır. DVM, Naive Bayes ve AdaBoost tabanlı sınıflandırmalar da çok az sayıda yanlış sınıflandırma ile gerçekleştirdikleri çalışmada iyi performans göstermiştir.

Waskle vd. (2020) PCA ve Rastgele Orman Algoritmasını kullanarak IDS geliştirmek için bir yaklaşım önerdiler. Kullanmış oldukları KDDCup99 veri kümesinde PCA ile boyutsallığı azalttılar ve veri kümesini düzenlediler. Sınıflandırmak için Rastgele Orman kullandılar. Önerdikleri yaklaşım DVM, Naive Bayes ve Karar Ağaçları tekniklerine göre doğruluk açısından daha verimli çalışmıştır. Elde ettikleri sonuçlarda performans süresi 3.24 dakika, doğruluk oranı %96.78 ve hata oranı %0.21'dir.

Burukanlı vd. (2021) tarafından ML yaklaşımlarının saldırı tespiti üzerindeki performans ve analizleri karşılaştırılmıştır. Veri kümesi KDD10CORRECTED ve KDDTEST kullandılar. Sınıflandırıcı olarak ise Karar Ağacı, Topluluk Öğrenme ve DVM tercih etmişlerdir. Boyut indirgeme tekniği olarak PCA kullanarak 5 kat çapraz doğrulama tekniğini kullandılar. Matlab ortamında her sınıflandırma yönteminin özelliklerini kullanarak deneyleri

gerçekleştirdiler. Sonuç olarak KDD10CORRECTED veri kümesinde Torbalama Sınıflandırıcısı %99,99 en iyi başarı oranını vermiştir. KDDTEST veri kümesinde ise %97,90 ile Torbalama Sınıflandırıcı 'sının en iyi başarı oranını verdiği sonucuna ulaştılar. Bu iki veri kümesi birlikte kullanıldığında %100 başarı oranı ile Torbalama Sınıflandırıcısı olduğunu gördüler.

2.2. MQTT Niteliklerini İçeren Çalışmalar

Firdous vd. (2017) MQTT protokolüne tehdit modeli sunmuşlardır. MQTT aracısına karşı DoS saldırılarının etkisini ölçmek için deneyler yaptılar. TCP SYNC saldırısı, MQTT tabanlı bir IoT ağını modelleyip, dört DoS saldırı türünün IoT kaynaklarına karşı etkinliği test ettiler. Yazarlar, DoS saldırısının etkisinin uç düğümden (yani kurban IoT düğümü) daha çok ağ bant genişliği üzerinde olduğunu kesin olarak belirtmişlerdir. Ayrıca, düşman tarafından sömürülebilen MQTT aracı düğümünün güvenlik açığının buldular.

Ge vd. (2019) yaptıkları çalışmada, Derin Öğrenme Modelini kullanarak IoT için IDS mekanizması sundular. Saldırı ve hedef sanal makineleri, hava durumu istasyonu, akıllı buzdolabı, akıllı termostat, uzaktan kumandalı garaj kapısı ve akıllı ışıklar ile akıllı ev test ortamı kurmuşlardır. Test ortamındaki cihazlar MQTT protokolü ile haberleşmiştir. BoT-IoT veri kümesi kullanarak özellik çıkarma, etiket eşleştirme, şema birleştirme, özellik ön işleme işlemlerini gerçekleştirdiler. Verileri sınıflandırma için ikili, çoklu sınıflandırmayı gerçekleştirip sonuçlarını analiz ettiler. Yapmış oldukları ikili sınıflandırmada, DDoS/DoS ve Keşif Saldırıları için doğruluk, kesinlik, geri çağırma ve F1 puanı 0.99'a yakın sonuç vermiştir. Çoklu sınıflandırmada, DDoS/DoS saldırıları için 0.99'un üzerinde algılama doğruluğu, normal trafik sınıflandırmasında da 0.98'lik bir doğruluk vermiştir.

Ciklabakkal vd. (2019) IoT ağları için MQTT'ye yönelik saldırılara odaklanan anomali tabanlı IDS'nin tasarımını ve uygulamasını tanımlamışlardır. MQTT'ye yönelik saldırıları içeren bir veri seti oluşturarak IDS'in, IoT ağ davranışını normal veya anormal olarak sınıflandırmışlardır. Anomali algılama için K means, SO GAAL, OCSVM, Rastgele Orman, İzolasyon Orman ve Otomatik Kodlayıcı modellerinin performanslarının karşılaştırmalı bir analizini sağlamışlardır. Deney sonuçları, MQTT tabanlı IoT ağlarında ML tabanlı saldırı

tespitinin daha önce bilinen saldırılarla eğitilmediğinde bile etkileyici sonuçlar elde edebileceğini göstermektedir.

Alaiz-Moreton vd. (2019) MQTT protokolünü kullanarak bir veri kümesi oluşturup sınıflandırma modeli oluşturdular. Saldırıları sınıflandırmak için topluluk yöntemleri ve derin öğrenme modelleri kullandılar. Oluşturdukları ağa DoS ve MITM saldırıları gerçekleştirdiler. Saldırıları XGBoost, LSTM ve GRU Tekrarlayan Ağ ile sınıflandırdılar. LSTM kullanarak F-beta skor değerinin 0.9148, kesinlik değerinin 0.9276 olduğu sonucuna vardılar. XGBoost kullanarak M. Logaritmik loss değeri 0.079451 ve M. Sınıflandırma hata oranı 0.025651 değerlerini elde etmişlerdir.

Moustafa. (2019) ToN_IoT adlı ML tabanlı saldırı tespit yaklaşımı için MQTT veri kümesini oluşturmuştur. Yazar tarafından oluşturulan veri kümesi, Windows 7- 10 işletim sistemleri veri kümelerinin yanı sıra Ubuntu 14 -18 TLS ve ağ trafiği veri kümelerini içermektedir. Oluşturduğu sisteme; Tarama Saldırısı, DoS, DDoS, Fidye Yazılımı Saldırısı, Arka Kapı Saldırısı, Enjeksiyon Saldırısı, XSS, Şifre Saldırısı ve MITM Saldırılarını gerçekleştirmiştir. Açık kaynaklı olan veri kümesinde gönderilen alınan MQTT paketlerinin tamamını içermemektedir. Sensör ve broker'ı içeren hem MQTT hem de TCP için kimlik doğrulama aşamasını veri kümesine dahil etmemişlerdir. IoT cihazlarda kimlik doğrulama ve bağlantı kesme aşamaları veri kümesinde olması gerekmektedir. Tüm düğümler için tek bir TCP bağlantısı içerdiğinden farklı düğümler için saldırıyı tespit etmesi zorlaşmıştır.

Hindy vd. (2020) simüle edilmiş bir ağa saldırılar gerçekleştirerek ML modelleri önermişlerdir. MQTT-IOT-IDS2020 veri kümesini oluşturularak eğitim ve değerlendirme süreçleri için 6 farklı makine öğrenimi tekniği ile oluşturdukları veri kümesi test edilmiştir. Saldırgan ağda 4 farklı saldırı gerçekleştirilmiş ve her biri bağımsız olarak kaydedilmiştir. Gerçekleştirdikleri saldırılar; Agresif Tarama (Scan A), UDP Taraması (Scan sU), Sparta SSH Kaba Kuvvet ve MQTT BF'dır. Veriler tcpdump kullanılarak elde edildi. Paketler, Ethernet trafiği kaydedilerek ve ardından PCAP dosyalarına aktarılarak yazarlar tarafından toplanmıştır. Tcpdump komut satırında çalışan paket izleyicisidir. Bilgisayara gelen paketleri kaydetmek, incelemek ve filtrelemek gibi amaçlarla kullanılır. PCAP ise Wireshark tarafından geliştirilmiş veri dosyasıdır.

Vaccari vd. (2020) tarafından sunulan çalışmada ise MQTTset veri kümesi kullanılmıştır. Birçok MQTT niteliğini barındıran bu veri kümesi, ML modellerini eğitmek ve

test etmek için kullanmışlardır. MQTT broker olarak Eclipse Mosquitto v1.6.2 kullandılar. Test yatakları farklı 8 sensör ve bunların birbirleri ve MQTT ile haberleşmesinden oluşmaktadır. Sıcaklık, ışık yoğunluğu, nem, gaz, hareket, duman, kapı açma/kapama ve fan durumu sensörleri hazırladıkları test yatağında bulunmaktadır. Sensörler 2 ayrı oda da konumlandırılmıştır. Bu ağdan elde edilen verileri toplam 33 özellik altında PCAP dosyası olarak kaydetmişlerdir. Siber güvenlik alanında yaygın olarak kullanılan ML algoritmalarından Sinir ağı, Rastgele Orman, Naive Bayes, Karar Ağacı, Gradyan Artırma ve Çok Katmanlı Algılayıcı kullanılarak veri kümesi doğrulanmış ve algoritmaların performansları, doğruluk oranları karşılaştırılmıştır. Veri kümesi Rastgele Orman ve Karar Ağaçları Algoritmaları'nın doğruluk oranları yaptıkları çalışma için en yüksek değerleri göstermektedir.

Gao vd. (2021) MQTT tabanlı saldırıları tespit etmek için çevrimiçi-yarı denetimli öğrenme sinirsel anomali tespit sistemi sunmaktadır. İlk olarak MQTTset veri kümesinden sezgisel özellik seçimi ile 10 adet özelliği kaldırmışlardır. Karar Ağaçları yöntemi ile veri kümesindeki özelliklerin önem sırası belirlenmiştir. Önem derecesi diğerlerinden daha düşük olan 4 öznitelik kaldırılmıştır. MQTTset veri kümesinde %70 eğitim, %30 test kümesi olarak ele alınmıştır. Eğitim veri kümesinin sadece %17'si üzerinde testleri gerçekleştirmişlerdir. Sınıflandırma ve performans tespiti için; S-hFFN, İzolasyon Ormanı, Yerel Aykırı Değer Faktörü, Tek Sınıflı DVM, K en yakın komşu ve Alt Uzay Aykırı Değer Tespiti yöntemlerini uyguladılar. En yüksek doğruluk oranını S-hFNN vermiştir.

Siddharthan vd. (2022) IoT bağlamında MQTT saldırı tespiti için SENMQTT-SET veri kümesini oluşturdular ve analizlerini gerçekleştirdiler. Test ortamında 4 adet Raspberry Pi, 2 adet Node MCU, Yönlendirici ve Mosquitto MQTT broker'ını kullanmışlardır. LR ile öznitelik seçimi, optimizasyonda ise Multi-View Cascade Feature Generation Algoritmasını tercih ettiler. Ayrıca eğitim kümesinde K en yakın komşu, DVM, DT, GB, NB, RF kullanarak sınıflandırdılar. ML tahmin modeli ile saldırı ve normal olarak değerlendirme gerçekleştirdiler.

3. TEMEL KAVRAMLAR

Bu bölümde Nesnelerin İnternetinde saldırı türleri, katmanlara göre saldırılar, MQTT protokolü ile ilgili genel bilgiler, Saldırı Tespit Sistemlerinde sınıflandırma, makine öğrenmesi ve istatistiksel yöntemler anlatılmıştır.

3.1. Nesnelerin İnternetinde Saldırı Türleri

IoT cihazlarının kullanım alanının artması, saldırganların bu cihazlardaki saldıracak noktaların çoğalması ile IoT cihazları daha da savunmasız hale gelmektedir. Güvenli cihaz geliştirmek ve bunların haberleşmesini sağlamak zorlaşmıştır. Geliştirilen akıllı cihazların interneti kullanıyor olması bu cihazlara yapılan saldırılara karşı savunmasız hale gelmesine sebep olmaktadır (Kaşif, 2020). IoT'ye yönelik saldırılar fiziksel, ağ ve uygulama katmanlarına yönelik tehditler olmak üzere 3 sınıfta incelenmektedir (Kaşif, 2020). Bu çalışmada yapılan saldırılar Çizelge 3.7'de belirtilmiştir. Uygulandıkları katmanlara göre saldırıların sınıflandırılması Çizelge 3.1'de verilmiştir. Çizelge 3.1, Nesnelerin İnternetinde katmanlara yapılan saldırıların literatür araştırması sonucunda oluşturulmuştur. Burada beş ana katmandan bahsedilebilir. Bu katmanların sayıları projelere göre farklılık gösterebilmektedir. Bu bölümde Fiziksel, Veri Bağı, Ağ, Taşıma ve Uygulama Katmanına yapılan saldırılar açıklanmıştır.

Çizelge 3.1. Katmanlara göre saldırılar

Katmanlar	**Saldırılar**
Fiziksel Katman	Zararlı Kod Aşılama Rolling Code Saldırısı Blueborne Saldırısı
Veri Bağı Katmanı	Yayın Bozma Saldırısı Geri Çekilme Saldırısı
Ağ Katmanı	Solucan Deliği Saldırısı Sybil Saldırısı Tekrarlama Saldırısı Gider Deliği Saldırısı DNS Zehirlenmesi Oltalama Saldırısı
Taşıma Katmanı	SYN Akını
Uygulama Katmanı	SQL Aşılama Saldırısı Ortadaki Adam Saldırısı Arka Kapı Saldırısı Sıfırıncı Gün Saldırısı Arabellek Taşması Saldırısı

3.1.1. Fiziksel katmana yapılan saldırılar

Fiziksel Katmandaki tehditler, cihazların donanımsal olarak ele geçirilmesi, dinleme faaliyetleri ve haberleşmenin sekteye uğratılması gibi çeşitli saldırılardan oluşmaktadır (Kaşif, 2020). Nesnelerin İnternetinde Fiziksel Katmanlara yaygın olarak yapılan saldırılar Zararlı Kod Aşılama, Rolling Code ve Blueborne Saldırılarıyla ilgili bilgiler verilmektedir. Çizelge 3.2'de literatür araştırması sonucunda Fiziksel Katmana yapılan saldırılar özetlenmiştir.

Çizelge 3.2. Fiziksel katmana yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Fiziksel Katman	Zararlı Kod Aşılama	Bir saldırgan kötü amaçlı kod uygulamak ve yürütmek için yazılımdaki bir giriş doğrulama açığından yararlandığında meydana gelen bir saldırıdır (Taş vd., 2021).
	Rolling Code Saldırısı	Bir sistemin şifresini bulmak için kullanılan kod ile gerçekleştirilen saldırıdır. Bu kodun algoritmaları, kullanıcı kilit sistemini açmaya çalıştığında kodun sürekli değişeceği şekilde tasarlanmıştır (Mann vd., 2020).
	Blueborne Saldırısı	Bluetooth üzerinden cihaza girebilen ve cihaz üzerinde tam kontrolü ele geçirebilen bir virüstür. Bu saldırının insan etkileşimi veya herhangi bir internet bağlantısına ihtiyacı bulunmamaktadır (Mann vd., 2020).

3.1.2. Veri bağı katmanına yapılan saldırılar

Ortam erişimi ve hata kontrolünün yapıldığı yer Veri Bağı Katmanıdır. Bu katman iletişim ağında noktadan noktaya ve bir noktadan çok noktaya iletişimin güvenilir şekilde yapılmasında önemli görev üstlenmektedir (Çakıroğlu, 2008). Veri Bağı Katmanlarına yapılan Yayın Bozma ve Geri Çekilme Saldırılarıyla ilgili bilgiler verilmiştir. Çizelge 3.3'te veri bağı katmanına yapılan saldırılar özetlenmiştir.

Çizelge 3.3. Veri bağı katmanına yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Veri Bağı Katmanı	Yayın Bozma Saldırısı	Saldırgan anten ile sinyal üreterek iletişimde parazitler yapmayı amaçlar. Tüm ağ tiplerine yönelik olduğu için etkili bir saldırıdır (Taş vd., 2021).
	Geri Çekilme Saldırısı	Veri bağı protokollerinde düğümlerin aynı zamanda erişimini önlemek için bir geri çekilme zamanı konulmuştur. Saldırgan küçük bir geri çekilme zamanı seçerek, daha az uyku modunda kalarak ağdaki diğer düğümlere göre daha avantajlı hale gelmeyi amaçlamaktadır (Abdelzaher vd., 2004).

3.1.3. Ağ katmanına yapılan saldırılar

Algılayıcı düğümleri ile çıkış düğümü arasındaki çok atlamalı kablosuz yönlendirme protokolü Ağ Katmanında yer almaktadır (Kaşif, 2020). Yönlendirme, ağ topolojisi yönetiminin sağlandığı Ağ Katmanına sıklıkla yapılan Solucan Deliği, Sybil, Tekrarlama, Gider Deliği, DNS Zehirlenmesi ve Oltalama Saldırılarının açıklamaları bu kısımda verilmiştir. Çizelge 3.4'te Ağ Katmanına yapılan saldırılar özetlenmiştir.

Çizelge 3.4. Ağ katmanına yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Ağ Katmanı	Solucan Deliği Saldırısı	Birbirinden uzakta bulunan iki saldırgan düğümün aralarında oluşturdukları özel bir iletişim yolu ile gelen veri paketlerinin doğrudan diğerine aktarılması yoluyla yönlendirmenin bozulduğu saldırıdır (Karlof vd., 2003).
	Sybil Saldırısı	Saldırgan, ağda trafik kavşakları oluşturarak cihazların performansını düşürür. Bu saldırılarda tehlikeli düğüm diğer komşu düğümleri dolandırarak, kötü düğümlerde her Sybil düğümünü ayrı bir düğüm olarak görerek birden fazla sahte kimlik yaymaktadır (Mann vd., 2020).
	Tekrarlama Saldırısı	Saldırgan ağdaki trafiği inceleyerek, ilerde kullanmak üzere verilerin kopyasını saklamaktadır. Kullanıcı orada olmasa bile saldırgan cihazı kontrol edebilmektedir (Mann vd., 2020).
	Gider Deliği Saldırısı	Tüm trafiği belirli bir bölgeye çekerek yönlendirme yolunun cazip hale gelmesini sağlamaktadır. Yüksek bant genişliği ve bununla birlikte düşük gecikmenin avantajlı yönlendirme olduğunu belirtmektedir. Tüm ağ trafiğini aldıktan sonra, veri paketinde yapılan değişiklikler gibi gizli bilgileri değiştirir (Gagandeep vd., 2012).
	DNS Zehirlenmesi	DNS sunucusuna yetkisiz bir kaynaktan bilgi yüklenmesi, sunucu konfigürasyon yanlışlıkları, DNS protokolü açıkları olarak tanımlanmaktadır (Gül vd., 2016).
	Oltalama Saldırısı	Sosyal medya hesabı, banka hesabı, e-posta hesabı gibi kullanıcı adı ve şifre girilmesi gereken sitelerin bir kopyası yapılarak hesap bilgilerinin çalınmaya çalışıldığı bir dolandırıcılık türüdür (Hassija vd., 2019).

3.1.4. Taşıma katmanına yapılan saldırılar

Güvenilir veri taşıma, internet veya diğer harici ağlara erişim işlemi bu katmanda gerçekleştirilir. Uygulamalarda kablosuz algılayıcı ağ düğümlerinin harici bir ağa doğrudan bağlanması gerekmediğinden literatürde ulaşım katmanı ile ilgili çok fazla çalışma bulunmamaktadır (Çakıroğlu, 2008). Taşıma Katmanına yapılan SYN Akını Saldırıları Çizelge 3.5'te verilmiştir.

Çizelge 3.5. Taşıma katmanına yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Taşıma Katmanı	SYN Akını	SYN Akını saldırılarında düşman büyük sayıda SYN isteğini kurbana göndermektedir. Düğüm, kurulan bağlantının o anda durumunu kaydederek SYN/ACK ile cevap vermektedir. Saldırgan daha ileri eylemler gerçekleştirmez. Diğer bağlantı isteğini kurbana gönderir (Raymond vd., 2008).

3.1.5. Uygulama katmanına yapılan saldırılar

Uygulama Katmanı, IoT sistemlerde son kullanıcıya akıllı evler, akıllı sayıcılar, akıllı şebekeler, akıllı taşımacılık, kişisel bakım, sağlık gibi uygulamalar ve hizmetler sunmaktadır (Taş vd., 2021). Uygulama Katmanlarına yaygın olarak yapılan saldırılar MITM, Arka Kapı, SQL Aşılama ve Arabellek Taşması Saldırılarıyla ilgili bilgiler verilmiştir. Çizelge 3.6'da uygulama katmana yapılan saldırılar özetlenmiştir.

Çizelge 3.6. Uygulama katmanına yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Uygulama Katmanı	SQL Aşılama Saldırısı	Veri tabanı bulunan uygulamalara saldırmak amacıyla kullanılan bir saldırı tekniğidir. Bu saldırı ile saldırgan, uygulamanın gizlenmiş bilgilerine, kullanıcı verilerine ve yetkilerine erişerek kayıtlar üzerinde işlem yapabilir (Bossi vd., 2017).
	Ortadaki Adam Saldırısı	Son kullanıcı verilerinin kontrolünü ele geçirmek için başlatılan saldırı türüdür. Saldırgan iletişim kanalına erişim sağlayarak iki nokta arasındaki mesajları manipüle eder (Bhushan vd., 2017).
	Arka Kapı Saldırısı	Saldırganlar için arka kapı oluşturmayı hedefleyen zararlı yazılımlardır. Antivirüsler ile genellikle kolayca tespit edilirler. Arka kapı aracılığıyla bulaştığı sistemde saldırgan ve kurban arasında soket bağlantısı kurmaktadır (Saha vd., 2020).
	Arabellek Taşması Saldırısı	Bir programın geçici belleğinde tutabileceğinden daha fazla veri tutmaya çalıştığında oluşan bir saldırı yöntemidir. Bellek alanının dışına yazmak verileri bozabilmektedir. Bu açık öncelikle programların veya sistemin çökmesine neden olabilir (Mann vd., 2020).

IoT'ye yönelik saldırılar Fiziksel, Ağ ve Uygulama Katmanlarına yapılmaktadır. Çizelge 3.7'de bu çalışma kapsamında gerçekleştirilen saldırılar ve kısa bir şekilde açıklamaları yer almaktadır. Detaylı açıklama Bölüm 4.2'de yer almaktadır.

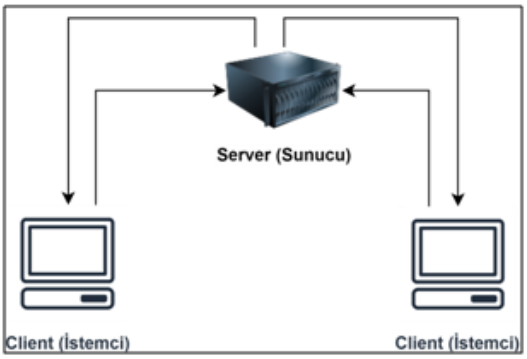
Çizelge 3.7. IoT ev ağına yapılan saldırılar

Katman	**Saldırılar**	**Açıklama**
Uygulama Katmanı	DoS	Haberleşme sisteminde var olan kaynakların kötücül bir yaklaşımla kullanılarak ve bu bileşenlerin haberleşemez hale getirilmesinin amaçlandığı saldırıdır (Arıs vd., 2015).
	Kaba Kuvvet Saldırıları	Kaba Kuvvet Saldırıları, hedef sistemde kayıtlı şifre ve parolaları kırmak için en sık tercih edilen siber saldırılardır (Kara., 2019).
	Botnet Saldırıları	Botmaster olarak bilinen saldırgan grubu tarafından kontrol edilirler. Bir web sitesi veya IP'ye erişmek için botlara komutlar vererek gerçekleştirilen bir saldırıdır (Vania vd., 2013).
	SlowDoS	Hizmet reddi yürüten bir ağ hizmetini hedeflemek için sunucuda birden çok bağlantı açarak, bu bağlantıları uzun süre açık tutmayı amaçlayan bir saldırıdır (Vaccari vd.,2020).

3.2. MQTT Protokolü

Nesnelerin İnterneti teknolojisindeki cihazlar kabiliyetlerine göre veri üretebilirler veya hem veri üretip hem de bu verileri işleyerek yönlendiriciye iletirler. Yönlendirici de cihazlardan toplamış olduğu bu verileri internet ortamına aktarır (Yalçınkaya, 2021). Bu verilerin güvenliğinin sağlanabilmesi amacıyla kullanılan standart ve protokollerin başlıcaları; MQTT, COAP, HTTP, XMPP, REST, IPV4/6, 6LowPAN ve RPL'dir.

MQTT protokolü, genellikle TCP/IP protokol takımı üzerinden çalışmaktadır. TCP/IP, günümüzde yaygın olarak kullanılan protokol takımlarındandır. MQTT protokolü, istemci – sunucu ağ mimarisini kullanmaktadır. Sunucu; bilgisayar ağlarında, diğer ağ bileşenlerinin erişebileceği, kullanımına veya paylaşımına açık kaynakları barındıran bilgisayar birimidir. Bir ağda birden fazla sunucu birimi bulunabilmektedir (Yalçınkaya, 2021). İstemci, bir ağ üzerinde, sunucu bilgisayarlardan hizmet alan kullanıcı bilgisayarlarıdır. Şekil 3.1'de örnek sunucu-istemci mimarisine ait model sunulmuştur.



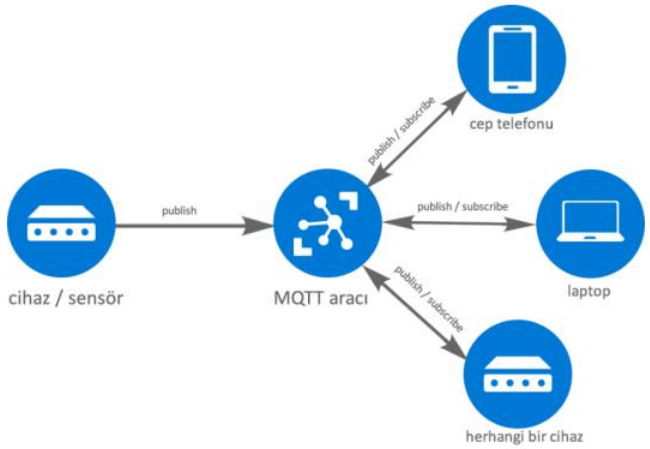
Şekil 3.1: İstemci-sunucu mimarisi modeli (Yalçınkaya, 2021).

MQTT protokolü, hafif cihazlar arasında mesaj ileten yayınlama (publish) – abone olma (subscribe) kalıbını kullanan bir ağ protokolüdür. Yayınlama – abone olma kalıbı, yayıncı olarak mesaj gönderenlerin, abone adlandırılan belirli alıcılara doğrudan gönderilecek mesajları programladığı bir mesajlaşma modelidir (Yalçınkaya, 2021). MQTT bağlantı mekanizması istemci ve broker arasında kurulmaktadır. İstemciler kendi arasında doğrudan bir bağlantı kurmamaktadır.

Çizelge 3.8’de verilen CONNECT mesajı ile istemci ve broker bağlantı kurarlar. Broker yanıt olarak CONNACK mesajını göndererek mesajdaki dönüş kodu, bağlantının durumunu belirler. Bağlantı başarılı ise istemci, broker’a mesaj yayınlama ve abone olma işlemini gerçekleştirebilir. Çizelge 3.8’de daha detaylı bilgiler yer almaktadır.

3.2.1. Broker/ yayıncı/ abone mekanizması

İstemci/sunucu modelinde (Bkz. Şekil 3.1) doğrudan bir bağlantı durumu söz konusudur. Mesajı gönderen bir yayıncı, abonenin varlığını bilmediği durumlarda da mesaj alıp gönderme işlemi gerçekleştirebilmektedir. Mesaj yayınlama ve abone olma işlemleri broker olarak adlandırılan merkez tarafından yapılmaktadır. Broker’lar bir IoT senaryosunun bileşenleri arasında bir köprüden oluşmaktadır. Bir cihazın bilgi göndermesi gerektiğinde, broker’lar tarafından yapılan iletişim sürecini kullanmak mümkündür (Oliveira vd., 2019). Şekil 3.2’de MQTT haberleşme modelinde de verildiği gibi broker yayınlanmış mesajları ilgili aboneye yayınlamaktadır. Broker tarafından mesajların filtrelenmesi de sağlanmaktadır.



Şekil 3.2: MQTT haberleşme modeli (Ebleme, 2019).

3.2.2. MQTT mesaj filtreleme

Broker, abonelerin yalnızca ilgilendikleri mesajları almasına olanak sağlamaktadır. Yayınla- abone ol modelinde mesajları farklı kriterlere göre filtrelemek mümkündür. Konu tabanlı filtreleme, içerik tabanlı filtreleme ve tür tabanlı filtreleme gibi mesaj filtreleme seçenekleri mevcuttur (Hillar, 2017).

- Konu tabanlı filtreleme de yayıncı, broker'dan bir mesaj yayınlamasını istediğinde hem konu hem de mesajı belirtmektedir. Broker, mesajı alarak ait olduğu konuya (topic) abone olan tüm abonelere iletmektedir. Aboneler birden fazla konuya abone olabilmektedir.
- İçerik tabanlı filtreleme, mesajların içeriğinde yer alan değerlere göre filtreleme işlemi yapılmaktadır. Burada DHT11 sensöründen okunan bilgilerde sıcaklık değerinin 25 dereceden daha yüksek olduğu değerlerin filtrelenmesi örnek verilebilir.
- Tür tabanlı filtreleme, pakette yer alan tür bilgisine göre filtreleme yapılabilmektedir. Bu çalışmada sıcaklık, nem ve uzaklık tür bilgisine örnek verilebilir.

Bu çalışmada konu tabanlı filtreleme sıklıkla kullanıldığı için detaylı şekilde açıklanmıştır.

3.2.3. MQTT konu tabanlı filtreleme

MQTT konu tabanlı filtrelemeye olanak sağlamaktadır. Konu bir veya daha fazla seviyeden oluşmaktadır. Her konu düzeyi bir eğik çizgi (/) ile ifade edilmektedir. Aboneler, yayınlamadan veya abone olmadan önce istenen konuyu oluşturmasına gerek yoktur. Broker, önceden başlatma olmaksızın her geçerli konuyu kabul etmektedir. Konular büyük/küçük harfe duyarlıdır. Ayrıca tek bir eğik çizgi de bir konu olarak kabul edilmektedir.

Örneğin: Esp/dht/temperature

Burada "Esp" birinci seviye konudur, "dht" ikinci seviyedir ve hiyerarşi bu şekilde devam etmektedir. Abone, örnekte verildiği gibi tek bir konuya abone olabilir. Birden fazla konuya ait bilgileri görüntülemek istenildiğinde joker karakterleri kullanılmaktadır. Tek seviye joker ve çok düzeyli joker karakteri kullanımı mümkündür.

Örneğin: Esp/dht/+/oda1

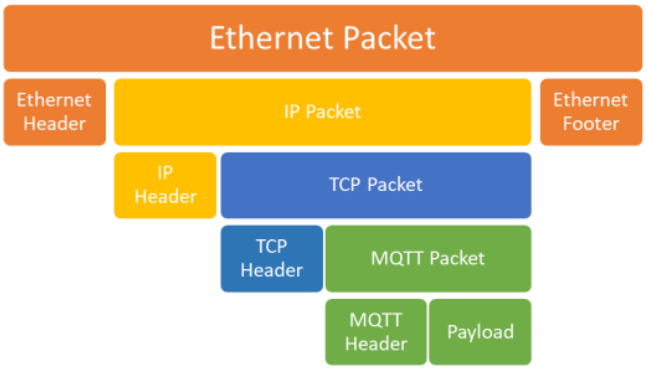
Konu tek seviye joker karakterinin kullanımının bir örneğidir. Bu konu ile “oda1” başlığı altındaki veriler alınabilmektedir.

Örneğin: Esp/dht/# veya Esp/hcsr/#

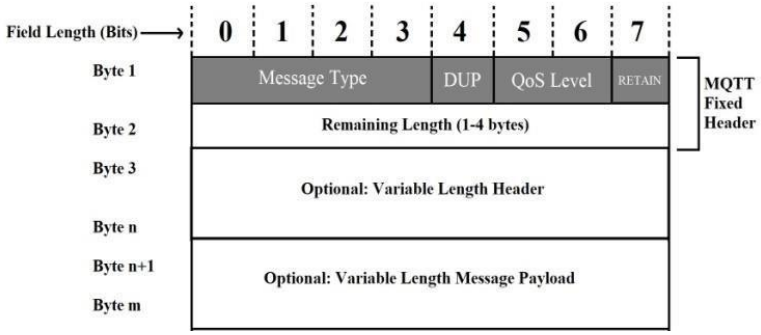
Konu çok düzeyli joker karakterinin kullanımının bu çalışmadaki örneğidir. Bu konu ile “dht” veya “hcsr” konusu altındaki bütün veriler görüntülenmektedir.

3.2.4. MQTT paket türleri

Gönderilecek olan mesaja önce TCP paket başlıkları eklenir ve mesaj TCP paketi özelliği kazanır. Daha sonra sırasıyla IP ve Ethernet başlıkları da eklenerek mesaj alıcıya gönderilir (Yalçınkaya, 2021). MQTT ile gönderilen bir mesaj Şekil 3.3’te görülen katmanlardan geçerek en son Ethernet Paketi olarak alıcı bilgisayara gönderilir.



Şekil 3.3: MQTT mesajının paketlenme hiyerarşisi (Ebleme, 2019).



Şekil 3.4: MQTT paket yapısı (Mishra, 2018).

Şekil 3.4'te verilen MQTT paket yapısında uygulamaya göre değişkenlik gösterebilmektedir. En az 2 bayt olacak şekilde sabit başlık alanı bulunmaktadır. DUP, mesajın kopyalandığını gösteren bayraktır. QoS Level; yayınlanan mesajın iletildiği QoS seviyesini göstermektedir. Retain, son alınan yayıncı mesajı sunucuya bildirirken yeni aboneye de ilk mesajı iletir. Remaining Length, mesajın kalan boyutudur. Header, opsiyonel olarak protokol versiyon bilgisi vb. içerir. Payload ise konu, mesaj, kullanıcı, şifre gibi bilgileri içermektedir (Mishra, 2018). Çizelge 3.8'de MQTT paket türleri, akış yönleri ve tanımları yer almaktadır.

Çizelge 3.8. MQTT paket türleri (Ebleme, 2019).

Paket Adı	**Değer**	**Akış Yönü**	**Tanım**
Rezerve	0	Forbidden	Rezerve edilmiş
CONNECT	1	İstemci → Sunucu	İstemci tarafından sunucuya bağlanma isteği
CONNACK	2	Sunucu → İstemci	Bağlantı onayı
PUBLISH	3	İstemci ↔ Sunucu	Mesaj yayınlar
PUBACK	4	İstemci ↔ Sunucu	Mesaj komisyoncu tarafından onaylanır
PUBREC	5	İstemci ↔ Sunucu	Mesajın komisyoncu tarafından alındığı bildirilir
PUBREL	6	İstemci ↔ Sunucu	Yayıncı mesaj alma bildirimi
PUBCOMP	7	İstemci ↔ Sunucu	İstemci ile komisyoncu haberleşmesinin tamamlandığını bildirir
SUBSCRIBE	8	İstemci → Sunucu	İstemciden aboneye mesaj iletir
SUBACK	9	Sunucu → İstemci	Aboneye mesajın ulaştığı bilgisi iletilir
UNSUBSCRIBE	10	İstemci → Sunucu	İstemciden sunucuya mesaj iletim bildirimi
UNSUBACK	11	Sunucu → İstemci	Sunucudan istemciye mesaj iletim bildirim onayı
PINGREQ	12	İstemci → Sunucu	İstemciden sunucuya haberleşme kontrol paketi
PINGRESP	13	Sunucu → İstemci	Sunucudan istemciye haberleşme kontrol paketi
DISCONNECT	14	İstemci → Sunucu	İstemci bağlantısı sonlandırır
Rezerve	15	Forbidden	Rezerve edilmiş

3.2.5. MQTT kalite servisi

Hizmet kalitesi (QoS) düzeyi, belirli bir mesajın teslim garantisini tanımlayan bir mesajın göndericisi ile alıcısı arasındaki bir anlaşmadır. QoS, istemciye ağ güvenilirliği ve uygulama mantığıyla eşleşen bir hizmet düzeyi seçme olanağı sağlamaktadır. MQTT, mesajların yeniden iletilmesini yönettiğinden ve teslimi garanti ettiğinden, QoS güvenilmez ağlarda iletişimi çok daha kolay hale getirmektedir (Anonim, 2015).
MQTT mesaj yapısında desteklenen 3 farklı servis kalitesi seviyesi bulunmaktadır (Ebleme, 2019). Servis kalite seviyelerine ait kısa bilgiler şu şekildedir:

QoS 0: En fazla bir defa, yayıncı mesajı sunucuya en fazla bir kez yayınlar. Mesaj sunucuda ve diğer uçlarda saklanmaz ve mesajın ulaşması kontrol edilmeyerek teslimat garantisi yoktur. Temel olarak TCP protokolü ile aynı garantiyi sağlar (Anonim, 2015). Şekil 3.5'te istemci ve broker arasındaki QoS 0 kalite seviyesi gösterilmiştir.

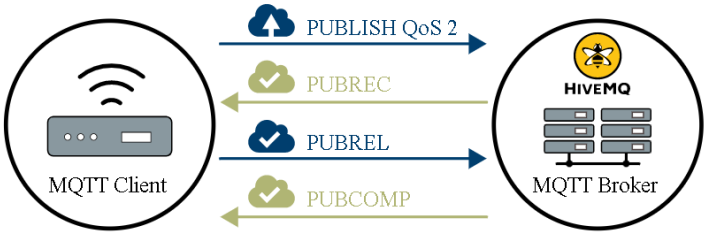
Şekil 3.5: MQTT QoS 0 (Anonim, 2015).

QoS 1: En azından bir defa, mesajın en az bir kere iletileceğini garanti eder. Ancak mesaj birden fazla iletilebilir. Yayıncı alıcıya mesajı bir kez gönderir. Gönderici, alıcıdan mesajın alındığını onaylayan bir PUBACK paketi alana kadar mesajı saklamaktadır (Anonim, 2015).

Şekil 3.6: MQTT QoS 1 (Anonim, 2015).

Şekil 3.6'da istemci ve broker arasındaki QoS 1 kalite seviyesi gösterilmiştir.

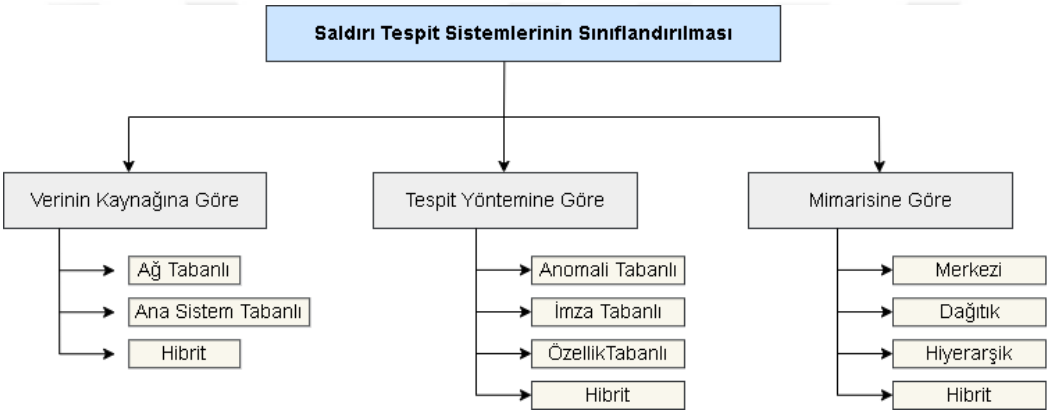
QoS 2: Kesinlikle bir defa, mesajın kesinlikle bir defa iletileceği garanti edilmektedir. En güvenli ancak en yavaş servis kalitesi seviyesidir. Gönderici ve alıcı, mesajın teslim olma durumunu koordine etmek için PUBLISH mesajının paket tanımlayıcısını kullanmaktadır. Alıcı, göndericiden bir QoS 2 PUBLISH paketi aldığında, yayınlama mesajını işleyerek göndericiye PUBLISH paketini onaylayan bir PUBREC paketi ile yanıt verir. Gönderici, alıcıdan bir PUBREC paketi alamazsa, onay alana kadar yinelenen (DUP) bayrağıyla PUBLISH paketini tekrar gönderir. Gönderici, alıcıdan bir PUBREC paketi aldığında, gönderen ilk PUBLISH paketini atmaktadır. Gönderici, PUBREL paketi ile yanıt verir. Alıcı PUBREL paketini aldıktan sonra, saklanan tüm durumları atarak PUBCOMP paketi ile yanıt vermektedir. Alıcı işlemi tamamlayıp PUBCOMP paketini gönderene kadar, alıcı orijinal PUBLISH paketinin paket tanımlayıcısını depolamaktadır (Anonim, 2015). Şekil 3.7'de istemci ve broker arasındaki QoS 2 kalite seviyesi gösterilmiştir.



Şekil 3.7: MQTT QoS 2 (Anonim, 2015).

3.3. Saldırı Tespit Sistemlerinin Sınıflandırılması

İlk Saldırı Tespit Sisteminin konsepti Anderson tarafından 1980 yılında önerilmiştir (Anderson, 1980). Bace vd. (2011) Saldırı Tespit Sisteminin görevini, bilgisayar sisteminde, ağ sisteminde gerçekleşen tüm olayları denetlemek ve kontrol etmek, güvenlik sorunları ortaya çıktığında ilgili personel ve birimleri uyarmak için alarm göndermek ve olası riskleri azaltmak için gerekli tedbirleri almak şeklinde tanımlamıştır.



Şekil 3.8: Saldırı tespit sistemlerinin sınıflandırılması (Balcı, 2021).

Şekil 3.8'de Saldırı Tespit Sistemlerinin sınıflandırmalarının bir özet hali sunulmuştur. Buna göre üç ana sınıflandırma gösterilmektedir. Bunlar: 1. Verinin Kaynağına Göre (Ağ, Ana Sistem, Hibrit), 2. Tespit Yöntemine Göre (İmza, Anomali, Özellik, Hibrit), 3. Mimarisine (Merkezi, Dağıtık, Hiyerarşik, Hibrit) göre olarak sınıflandırılmıştır.

3.3.1. Verinin kaynağı

Saldırı Tespit Sistemlerinde verinin kaynağına göre ağ tabanlı, ana sistem tabanlı ve hibrit olarak üçe ayrılabilir. Ağ tabanlı Saldırı Tespit Sistemlerinde ağın değişik konumlarına sensörler yerleştirilerek ağ trafiğindeki şüpheli durumları tespit etmek için ağı izlemektedir. Ana sistem tabanlı Saldırı Tespit Sistemlerinde sadece ağ trafiği ile uygulamalar, dosya sistemi değişiklikleri, sistem çağrıları, iletişim ve uygulama logları da takip edilmektedir. Hibrit yaklaşımlarda ise ağın değişik konumlarına sensörler ile sisteme bir donanım da yerleştirilir. Hem şüpheli ağ trafikleri hem uygulamalar arasındaki etkileşimler ve şüpheli durumlar tespit edilmektedir (Vacca, 2013).

3.3.2. Tespit yöntemi

Tespit yöntemlerine göre imza tabanlı, anomali tabanlı ve protokol analizi bazında sınıflandırma yapılır (Scarfone vd., 2007). İmza tabanlı yaklaşımlarda daha önceden oluşturulan veri tabanındaki saldırı imzaları ile sistemin şu anki davranışları kıyaslanır. Sistemde kaydedilmiş imza ile aynıysa sistem ikaz vermektedir. Veri tabanındaki imzalar ile eşleşen bir imza bulamadığından, bilinen saldırıların değişik türevleri ya da yeni tip saldırıların tespitinde yetersiz kalmaktadır (Liao vd., 2013). Anomali tabanlı Saldırı Tespit Sistemlerinde sistemin istatistiksel teknikler veya makine öğrenmesi algoritmaları kullanarak bir normal davranış profili oluşturulur. Daha sonra sistemin veya ağın anlık davranışlarını bu normal profil ile kıyaslayarak normal davranıştan sapan davranışlarda uyarı oluşturur. Yeni saldırıların tespitinde etkili bir yaklaşımdır (Balcı, 2021). Özellik tabanlı teknikler ise tıpkı anomali tabanlı sistemler gibi önceden tanımlanmış normal davranış modelinden farklı bir aktivitenin tespit edilmesini içerir. En büyük farkı faaliyetlerin kötü niyetli olduğunu bir insan tarafından onaylanmasının gerekliliğidir (Bütün, 2014). Hibrit çözüm teknikleri ise yukarıda adı geçen yöntemlerden birkaçını içererek zayıf yönlerinin diğer yöntemin güçlü yanları ile kapatılması esasına göre tasarlanır (Balcı, 2021).

3.3.3. Mimari yapı

Mimari yapısına göre merkezi, dağıtık, hiyerarşik ve hibrit olarak sınıflandırılır. Dağıtık yapılı bir Saldırı Tespit Sisteminde sensörler ağın her bir fiziki objesine yerleştirilir. Kısıtlı kaynağa sahip bu düğümlerin her birinin optimize edilmesi gerekmektedir. Ayrıca bütün düğümler komşu düğümünün bekçi köpeği olarak adlandırılır ve onları da gözlemlerler (Cervantes, 2015). Merkezi mimaride ise bütün tespit, izleme ve raporlama işlemleri merkezi bir birimde kontrol edilir (Güven, 2007). Dağıtık mimarinin karışıklığı, merkezi mimarinin de uçlardaki düğümlerde meydana gelebilecek saldırıların tespitinde gecikmelere sebep olabilecek olmasından dolayı hibrit çözümler de yaygın şekilde kullanılmaktadır. Hiyerarşi yapılar, bağımsız veya bazı düğümlerin diğerlerinden daha fazla işleme sorumluluğuna sahip olduğu şekliyle merkezi veya dağıtık yapıda başka bir mimari türüyle kombinasyon halinde olabilir. Merkezi olmayan mimariler hiyerarşik olarak gruplandırılmıştır (Balcı, 2021).

3.4. Nesnelerin İnternetinde Makine Öğrenmesi ve İstatiksel Yöntemler

İstatistiksel modelleme, ilk ML programı olan Samuel'in denetleyicisi programının tanıtıldığı 1950'lerde Makine öğreniminin ortaya çıkmasından yüzyıllar önce ortaya çıkmıştır (Ramzai, 2020).

Makine öğrenimi, bilgisayar ve verileri kullanarak tahmin üzerinde çalışır ve ayrıca hesaplama istatistikleriyle ilişkilendirilir. ML, yapay zekanın bir alt kategorisidir. ML, matematiksel optimizasyon ile güçlü bir bağlantıya sahip olduğu için farklı teoriler, yöntemler ve uygulama alanları içerir (Ekhlayef, 2020). Makine öğrenmesi, mevcut verilerden öğrenme sağlayarak yeni veriler üzerinde tahminler yapan algoritmalar içermektedir. Makine öğrenme algoritmalarında başarı hesaplama, algoritmanın eğitimi sırasında kullanılan verilere bağlıdır. Eğitim aşamasında ne kadar kapsamlı ve düzgün veriler kullanılırsa, test aşamasında o kadar sağlıklı sonuçlar elde edilir (Aytan, 2019).

Makine öğrenimi, önceden belirlenmiş kurallar olmadan verilerden öğrenmeyi kendi başlarına akıllı olma yeteneğine sahip öngörücü algoritmalar oluşturma ile ilgilenmektedir. İstatiksel modelleme, önce farklı değişkenler arasındaki ilişkileri bulabilen ve daha sonra diğer bağımsız değişkenlerin bir fonksiyonu olarak tanımlanabilecek bir olayı tahmin edebilen

modeller oluşturmaya odaklanmaktadır (Ramzai, 2020). Nesnelerin İnternetinde ML ve İstatiksel yöntemler birlikte kullanıldığı gibi ayrı olarak da yaygın kullanımı bulunmaktadır. Bu çalışmada ML yöntemleri ile ilgili uygulama yapıldığı için denetimli, denetimsiz ve yarı denetimli öğrenme olarak üçe ayrılmıştır.

3.4.1. Denetimli öğrenme

Denetimli öğrenme algoritmaları hem istenen girdilere hem de çıktılara sahip bir veri kümesinden bir model oluşturmak için kullanılır (Bertino vd., 2017). Kullanılan veriler eğitim verileridir ve tek veya birden çok girdi ve çıktı içeren eğitim örnekleri içerir (Kolias vd., 2017). Kullanılan yöntem denetleyici sinyal olarak bilinir. Denetlenen sinyalle ilgili sorunlarla uğraşırken, önce onunla ilişkili doğru etiketlere sahip bir dizi eğitim örneğiyle başlarız. Bir örnek, temsil ettiği her görüntü için doğru sayıyı içeren doğru etiketlerle elle yazılmış rakamların binlerce fotoğrafını çeken denetimli bir öğrenme algoritması eğittiğimiz zamandır (Kolias vd., 2017). Bu, algoritmanın el yazısıyla yazılmış rakamları sınıflandırmasına ve ayrıca görüntülerin ve ilgili sayıların ilişkisini öğrenmesine ve bu ilişkiyi daha önce makineye gösterilmemiş etiketler olmadan yeni görüntüleri sınıflandırmak için kullanmasına olanak tanır. Sınıflandırma ve regresyon denetimli öğrenmenin bir parçasıdır (Koroniotis vd., 2019). Sınıflandırmalar için algoritmalar, çıktıların sınırlı olan değer kümeleriyle sınırlı olduğu durumlarda kullanılır. Regresyon algoritmaları, çıktıların belirli bir aralıkta sayısal değerlere sahip olabileceği durumlarda kullanılır. Denetimli Öğrenme algoritmalarından en yaygın kullanılanlar; DVM, NB, RF, DT, KNN ve Regülasyon algoritmalarıdır (Tahsien vd., 2014).

3.4.2. Denetimsiz öğrenme

Denetimsiz öğrenme algoritmalarına yalnızca girdi içeren veri kümeleri verilir ve ardından algoritma veri noktalarının kümelenmesini bulunur. Bu nedenle algoritmalar etiketlenmemiş, kategorize edilmemiş veya sınıflandırılmamış test verilerinden öğrenir. Denetimsiz öğrenme algoritmaları, bir geri bildirim yaklaşımı yerine her yeni veri parçasındaki ortak noktaların yokluğuna veya varlığına dayanan verilerdeki ortak noktaları tespit eder. Etiketlenmemiş verilerin temel yapısını yararlı bir şekilde gruplamak, özetlemek ve bulmak denetimsiz öğrenmenin işlevidir (Ekhlayef, 2020). Denetimsiz öğrenme verileri gruplandırdığı için genellikle kümeleme amaçlı kullanılmaktadır. K-Means (Hartigan vd., 1979) ve PCA (J. Shlens, 2014) en popüler iki denetimsiz öğrenme algoritmasıdır.

3.4.3. Yarı denetimli öğrenme

Yarı denetimli öğrenme de veri etiketleri hakkında yalnızca kısmi bilgiler mevcuttur. Genellikle etiketlenmemiş büyük miktar veri topluluğunun az miktar etiketli verilerden nasıl etiketlendiğini öğrenmesini sağlayarak etiketsiz verilerin etiketlendirilmesini amaçlamaktadır. Bir başka deyişle yarı denetimli sınıflandırma, amacı her ikisinden de yararlanmak olan denetimli ve denetimsiz sınıflandırma arasında bir öğrenme yöntemidir (Ordin, 2010).

3.5. Nesnelerin İnternetinde Kullanılan Veri Kümeleri

Araştırmalar neticesinde birçok veri kümesi olduğu görülmüştür. Veri kümelerinden bazıları sadece IoT izlerini içerirken bazıları ise hem IoT izlerini hem de MQTT özniteliklerini içermektedir. Çizelge 3.9’da IoT izlerinin ve MQTT özniteliklerinin bulunduğu veri kümeleri verilmiştir.

Çizelge 3.9. Veri kümelerinin karşılaştırması

Veri Kümesi	**IoT İzleri**	**MQTT Öznitelikleri**	**Yıl**
DARPA 98	✕	✕	1998
DARPA 99	✕	✕	1999
KDDCUP 99	✕	✕	1999
NSL-KDD	✕	✕	2009
ISCX 2012	✕	✕	2012
ADFA-WD	✕	✕	2014
ADFA-LD	✕	✕	2014
CICIDS2017	✕	✕	2017
CICIDS2018	✕	✕	2018
Bot-IoT	✓	✕	2019
TON_IoT	✓	✕	2019
ARTEMIS	✓	✓	2019
Alaiz-Moreton vd. tarafından oluşturulan veri kümesi	✓	✓	2019
MQTTset	✓	✓	2020
MedBloT	✓	✕	2020
IoT-Botnet	✓	✕	2020
MQTT-IOT-IDS2020	✓	✓	2020
X-IIoTID	✓	✕	2021
SENMQTT-SET	✓	✓	2022

MQTT özniteliklerini MQTTset, ARTEMIS, MQTT-IOT-IDS2020, Alaiz-Moreton vd. (2019) tarafından oluşturulan veri kümesi ve SENMQTT-SET içermektedir. Ancak MQTT-IOT-IDS2020 ve ARTEMIS veri kümesinin MQTT öznitelik sayısı az olduğu için ve SENMQTT-SET veri kümesinin açık kaynaklı olmaması sebepleriyle tercih edilmemiştir. Alaiz-Moreton vd. (2019) tarafından oluşturulan veri kümesinde saldırı olarak DoS ve MITM tercih etmişlerdir. Daha fazla saldırı çeşiti olması sebebiyle MQTTset veri kümesi bu çalışmada kullanılmıştır.

MQTT niteliklerini içeren veri kümeleri için Çizelge 3.10'da uygulanan metotlar ve değerlendirme metriklerinin karşılaştırması verilmiştir.

Çizelge 3.10. Veri kümelerinin detaylı karşılaştırması

Veri Kümesi	**Metodlar**	**Değerlendirme Metrikleri**	**Referans**
Alaiz-Moreton vd. tarafından oluşturulan veri kümesi	XGBoost, GRU, LSTM	Log Loss, Cross Entropy Accuracy, F-Beta Score	(Alaiz-Moreton vd., 2019)
MQTTset	YSA, RF, DT, MLP, GB, NB	Accuracy, F1-Scor, Confusion Matrix	(Vaccari vd., 2020)
MQTT-IoT-IDS2020	SVM, Gaussian NB, KNN, DT, RF, LR	Accuracy, Precision, Recall, F1-Scor	(Hindy vd., 2020)
SENMQTT-SET	LR, KNN, SVM, NB, DT, RF, GB	Accuracy, ADR, APV, NPV, F1-Score, MCC, FAR, ROC AUC	(Siddharthan vd., 2022)
ARTEMIS	Autoencoder, SO_GAAL, RF, KNN, OCSVM, IF	ROC AUC, Accuracy	(Ciklabakkal vd., 2019)

Çizelge 3.11'de veri kümelerinde yer alan öznitelikler ve veri kümesi karşılaştırması verilmiştir.

MQTTset, ARTEMIS, MQTT-IoT-IDS2020 ve Alaiz-Moreton vd. tarafından oluşturulan veri kümelerinde ortak öznitelikler olduğu görülmüştür.

Çizelge 3.11. Veri kümelerindeki özniteliklerin karşılaştırması

No	Öznitelik Adı	MQTTset	ARTEMIS	MQTT-IoT-IDS2020	Veri kümesi (Alaiz-Moreton vd., 2019)
1	TCP Flags	✓		✓	
2	Time TCP Stream	✓			✓
3	TCP Segment Length	✓			✓
4	Acknowledge Flags	✓			✓
5	Conack Flag Reserved	✓			✓
6	Session Present	✓			✓
7	Return Code	✓		✓	✓
8	Clean Session Flag	✓			
9	Password Flag	✓		✓	✓
10	Conflag QoS Level	✓			✓
11	Conflag Reserved	✓		✓	✓
12	Will Retain	✓		✓	✓
13	User Name Flag	✓		✓	✓
14	Will Flag	✓		✓	✓
15	Connect Flags	✓			✓
16	DUP Flag	✓			✓
17	Header Flags	✓			✓
18	Keep Alive	✓			✓
19	Message Length	✓		✓	✓
20	Message	✓	✓		✓
21	Message Identifier	✓			✓
22	Message Type	✓	✓	✓	✓

Çizelge 3.11. Veri kümelerindeki özniteliklerin karşılaştırması (devam)

23	Protocol Name Length	✓			✓
24	Protocol Name	✓			✓
25	QoS Level	✓	✓		✓
26	Retain	✓			✓
27	Requested QoS	✓			✓
28	Granted QoS	✓			✓
29	Version	✓			✓
30	Will Message	✓			✓
31	Will Message Length	✓			✓
32	Will Topic	✓			✓
33	Will Topic Length	✓	✓		✓

4. MATERYAL VE YÖNTEM

4.1. Deneysel Ortam

Hazırlanmış olan IoT ev ağı ortamında nesnelerin birbirleriyle iletişim kurmasını sağlamak için kablosuz ağlar kullanılmıştır. Bu çalışmada NodeMCU ile Arduino IDE tabanlı IoT ev ağı prototipi tasarlanmıştır.

Bu bölümde öncelikle geliştirilen IoT ev ağı prototipinde kullanılan donanım bileşenleri ve kullanılan araçlar anlatılmıştır. Sonrasında IoT ev ağına gerçekleştirilen siber saldırılar, MQTTset veri kümesiyle ilgili detaylı bilgiler, bu çalışmanın akış diyagramı ve saldırı sınıflandırmasında kullanılan yöntemler hakkında bilgiler verilmiştir.

4.1.1. Donanım tasarımı ve bileşenleri

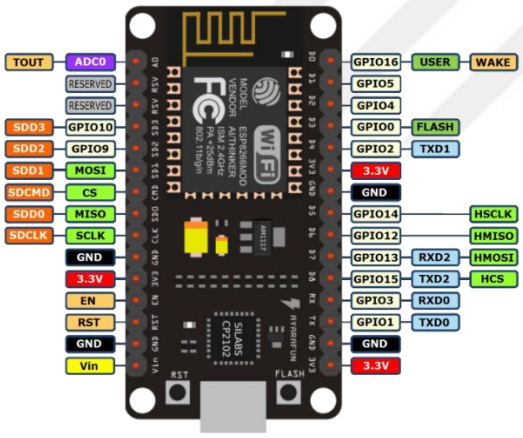
Arduino IDE tabanlı IoT akıllı ev prototipinde kullanılan sensörler ve özellikleri, donanım tasarımı detaylı şekilde anlatılmıştır. Bu çalışmada Arduino IDE, NodeMCU, DHT 11 sıcaklık ve nem sensörü, HC-SR04 mesafe sensörü ve Raspberry Pi 4 kullanılmıştır.

- ***Arduino IDE:***

Arduino tarafından tanıtılan ve Arduino cihazındaki kodu düzenlemek, derlemek ve yüklemek için kullanılan bir yazılımdır. MAC, Windows, Linux gibi işletim sistemlerinde kolayca kullanılabilen Arduino Java platformunda çalışmaktadır. Arduino IDE kodu yazmak için ve sonra kodu derlemek, modüle yüklemek için kullanılmaktadır (Fezari vd., 2018). Arduino IDE'de yazılım kütüphanelerinden; "DHT.h", "ESP8266Wifi.h", "Ticker.h", "Ultrasonic.h" ve "AsyncMqttClient.h" kütüphaneleri bu çalışmada kullanılmıştır. Arduino IDE ile NodeMCU kullanılabilmesi için;
"Ek Devre Kartları Yöneticisi URL'leri" kısmında "http://arduino.esp8266.com/stable/package_esp8266com_index.json" eklenmiştir. Arduino IDE ile yazılan kodlamada IP adres, port ve konu (topic) bilgileri tanımlanmıştır.

- ***NodeMCU:***

NodeMCU, Espressif'in ESP8266 wifi yongası için geliştirilmiş açık kaynaklı bir LUA tabanlı yazılımdır. NodeMCU yazılımı, ESP8266 geliştirme kartı/kiti ile gelmektedir. Ayrıca açık kaynaklı olması sebebiyle tercih edilmiştir. ESP8266 NodeMCU güç, yerleşik Mikro USB konektörü aracılığıyla sağlanmaktadır. ESP8266 NodeMCU; 32 Kb RAM, 80 Kb DRAM ve 200 Kb Flash Bellek bulunmaktadır (Parihar, 2019). Şekil 4.1'de NodeMCU pin şeması verilmiştir. Bu çalışmada NodeMCU üzerinde, DHT11 ve HC-SR04 sensörleri entegrasyonu sağlanmıştır. NodeMCU, Arduino IDE ile programlanmış ve gerekli kütüphanelerin kurulumu yapılmıştır. NodeMCU ve Raspberry Pi aynı IP adresi 1883 portu üzerinden yayın (publish) yapmaktadır.



Şekil 4.1: NodeMCU pin şeması (Kahraman, 2018).

- ***Raspberry Pi 4:***

Raspberry Pi, Raspberry Pi Vakfı tarafından geliştirilen ve ilk olarak Şubat 2012'de piyasaya sürülen düşük maliyetli bir ARM bilgisayar sistemidir. Bu çalışmada kullanılan Raspberry Pi 4 Model B kullanılmıştır. Çizelge 4.1'de Raspberry Pi cihaz özellikleri verilmiştir. Raspberry Pi üzerine Kali Linux işletim sistemi kurulumu yapılarak bir bilgisayar haline getirilmiştir. Bu cihaz NodeMCU ile entegre şekilde çalışmaktadır. Ayrıca Gateway olarak 192.168.0.1 ve Static IP 192.168.0.53 olarak belirlendi. Router ile aynı ağda olması sağlandı.

Çizelge 4.1. Raspberry Pi cihaz özellikleri (Anonim).

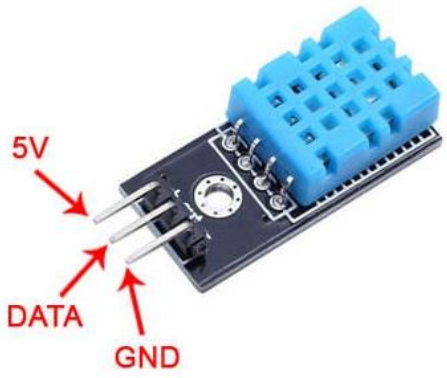
Cihaz	**Marka ve Model**
İşlemci	Quad core Cortex-A72 (ARM v8) 64-bit @1.5GHz
Ram	2GB LPDDR4-3200 SDRAM
Genişletme yuvaları	40-pin GPI0 başlığı
Görüntü çıkışı	Micro-HDMI
Kamera desteği	MIPI kamera yuvası
Depolama	1 Mikro SD yuvası
Güç	5V/2.5A DC güç girişi
Kablosuz adaptörü	2.4GHz ve 5GHz IEEE 802.11ac kablosuz LAN, Bluetooth 5.0, BLE

- ***DHT11 Sensör Modülü:***

DHT11 sıcaklık/nem sensörü, ortamdaki sıcaklık ve nem değerlerini ölçerek topladığı verileri dijital olarak düzenleyip çıkış verebilen bir sensördür (Erzi,2021). DHT 11 sensör ve sensör modülü olarak ikiye ayrılmaktadır. Sensörde 4 adet çıkış mevcutken, sensör modülünde 3 adet çıkış bulunmaktadır. Bu çalışmada kullanılan DHT 11 sensör modülü Şekil 4.2'de verilmiştir. DHT 11 sensör ve modülü akıllı ev projelerinde sıklıkla tercih edilmektedir. 8 bit mikroişlemci içermektedir.

DHT 11 özellikleri şu şekildedir: (Anonim)

- ✓ Sıcaklık ölçümü: 0ºC-50ºC ±2°C
- ✓ Nem ölçümü: %20 – %90 RH ±5%RH
- ✓ Çalışma Akımı (çalışma): 0.5mA (2.5mA max.)
- ✓ Çalışma Akımı (bekleme): 100uA (150uA max.)
- ✓ Çalışma Voltajı: 5V



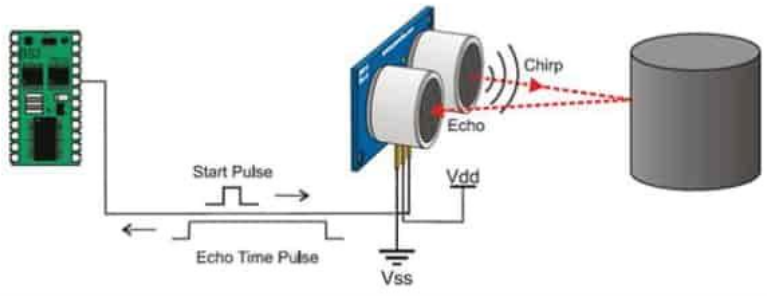
Şekil 4.2: DHT11 sensör modülü (Erzi, 2021).

Bu çalışmada DHT 11 sensörü ve NodeMCU bağlantısı gerçekleştirilmiştir. DHT 11 sensöründen iletilen sıcaklık ve nem bilgilerinin görüntülenmesi Arduino Serial Monitör ve MQTTLens programı ile gerçekleştirilmiştir.

Çizelge 4.2'de DHT11 sensörü izleme yöntemi ve detaylı bilgiler yer almaktadır. DHT11 sensörü 192.168.0.53 IP adresi ve 1883 portu üzerinden sıcaklık ve nem bilgilerini MQTT broker'a yayınlamaktadır. DHT11 sensörü için konu (topic) sıcaklık için "Esp/dht/temperature" ve nem bilgileri için "Esp/dht/humidity" olarak belirlenmiştir.

- ***HC-SR04 Ultrasonik Sensör:***

Ultrasonik sensörler mesafe ve varlık tespitinde sonar sistemini kullanarak değerler üreten cihazlardır. Radarla çalışma mantığı benzerlik gösteren ultrasonik sensörler ses dalgalarını kullanarak mesafeyi ölçmektedir. Endüstriyel anlamda en çok tercih edilen HC-SR04'ün ultrasonik dalga frekansı 40kHz'dir. Şekil 4.3'te ultrasonik sensörün çalışma prensibi verilmiştir. Burada mesafe ölçümünde mikro denetleyici, ultrasonik sensöre tetikleme sinyali göndermektedir. Tetikleme tamamlandığında HC-SR04 akustik dalgalar üretir. Akustik ile zaman sayacı da başlatılır. Yansıyan sinyal ulaştığında zamanlayıcıyı durdurarak ultrasonik sensör değerler üretir. Ultrasonik sensörler birçok mikro denetleyici ile kolaylıkla kullanılabilmektedir. Bu çalışmada HC-SR04 sensörü NodeMCU ile kullanılmıştır. HC-SR04 sensörünü tetiklemek ve sinyal darbe genişliğini manuel olarak ölçmek için "NewPing" kütüphanesi kullanılmıştır. HC-SR04 sensörü 192.168.0.53 IP adresi ve 1883 portu üzerinden uzaklık bilgisini MQTT broker'a yayınlamaktadır. HC-SR04 sensörü için konu uzaklık için "Esp/hcsr/distance" olarak belirlenmiştir.



Şekil 4.3: HC-SR04 sensörü çalışma prensibi (Akgul, 2021).

Çizelge 4.2'de sensörler ve izleme yöntemleri hakkında detaylı bilgiler verilmiştir.

Çizelge 4.2. Sensörler ve izleme yöntemleri

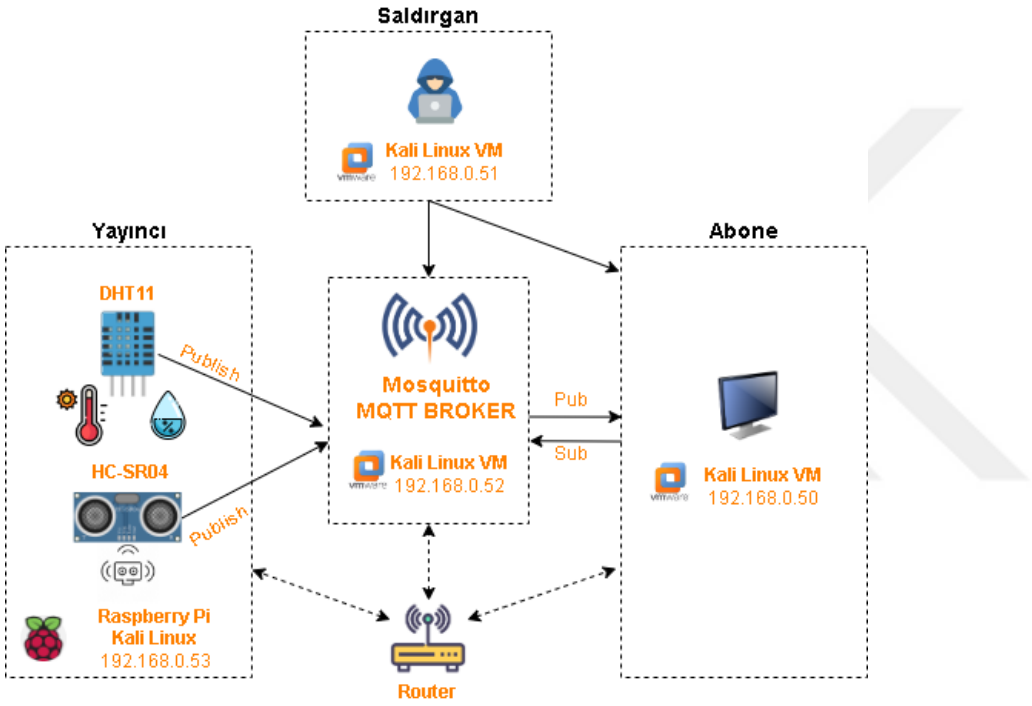
Sensör	**IP Adresi**	**Zaman**	**Konu**	**İzleme Yöntemi**
DHT11 Sıcaklık/Nem Sensörü	192.168.0.53	Periyodik Davranış	Esp/dht/temperature Esp/dht/humidity	Serial Monitör, Wireshark, MQTTLens
HC-SR04 Mesafe Sensörü	192.168.0.53	Rastgele Davranış	Esp/hcsr/distance	Serial Monitör, Wireshark, MQTTLens

4.1.2. Akıllı ev ağı kurulumu

Bu bölümde akıllı ev prototipinde kullanılan makinelerin birbirleriyle haberleşmesi ve kurulumlar ile ilgili detaylar paylaşılmıştır. Kali Linux işletim sistemleri, Vmware sanallaştırma yazılımı ile sağlanmıştır. Kullanılan tüm makinelerde statik IP ataması yapılmıştır. Ağda yer alan broker, mesajları aboneye iletmekten, hedeflerine yönlendirmekten ve mesajları yayıncılardan almaktan sorumludur. Ağda bulunan 2 adet sensör, simüle edilerek farklı senaryolarda ağ trafiği dinlenmiş ve kaydedilmiştir. Normal çalışma sırasında sensörler "Yayınla" komutunu kullanarak rastgele mesajlar göndermektedir. Mesajların uzunluğu ve konu bilgileri değişkenlik gösterdiği için rastgele bir şekilde mesajlar oluşturulur. Normal trafik Wireshark aracılığıyla kaydedilir. Farklı saldırılar altında kayıt alınması süreci sırasıyla gerçekleştirilir. Wireshark aracı ile dinleme ve kayıt alma işlemi "192.168.0.52" IP'li bilgisayarda gerçekleşmektedir. Makine Öğreniminde kullanım kolaylığı olması sebebiyle ağ trafiği CSV formatında kaydedilmiştir.

Yayınla işlemleri "192.168.0.53" IP adresli Raspberry Pi üzerinde kurulmuş Kali Linux makinesinde gerçekleştirilmiştir. Bu makinede Arduino IDE kullanılarak devamlı olarak yayın yapacak şekilde bir yazılım geliştirilmiştir. Bu yazılım sensörlerden verileri okuyarak Broker'a mesajların iletilmesini sağlamaktadır. Abone işlemleri Kali Linux "192.168.0.50" IP adresli sanal makinesinde MQTTLens aracı ile sağlanmıştır. Konu bilgisi ve bilgisayarın IP adresi ile abone işlemi gerçekleştirilmiştir. Broker'ın konfigürasyon ayarları "mosquitto.conf" içerisinde

yapılmıştır. Burada broker'ın kullanıcı adı ve şifre bilgileri kaydedilmiştir. Şifre ve kullanıcı adı bilgileri ".txt" olarak kaydedilen başka bir dosyadan entegre edilmiştir. Saldırı senaryoları Kali Linux "192.168.0.51" IP'li sanal makinesinde kurgulanmıştır. Saldırı araçlarının kurulumu bu sanal makinede sağlanmıştır. Bölüm 3.2'de MQTT broker, yayıncı ve abone çalışma mekanizması ve projede nasıl kullanıldığına dair detaylı bilgiler verilmiştir.



Şekil 4.4: Akıllı ev ağı haberleşme yapısı

Şekil 4.4 ağ bileşenlerini ve MQTT haberleşme yapısını özetlemektedir.

4.2. Kullanılan Araçlar

Çalışmada oluşturulan IoT ev ağına çeşitli saldırılar gerçekleştirilmiş ve bu saldırılar araçlar aracılığıyla izlenmiş ve veriler kaydedilmiştir. Bu bölümde kullanılan saldırı çeşitleri ve verilerin elde edilmesinde kullanılan araçlardan bahsedilmiştir. Çizelge 4.3'te bu çalışmada kullanılan araçların kullanım amaçlarının özeti verilmiştir.

Çizelge 4.3. Kullanılan araçlar ve kullanım amaçları

Araç	**Kullanım Amacı**
MQTTLens	MQTT İstemcisi, Verilerin İzlenmesi
Wireshark	Ağ Trafiğinin Dinlenmesi
WEKA	Öznitelik seçimi, ML, DL
Mosquitto	MQTT Broker
LOIC	DoS Saldırısı
RUDY	Yavaş DoS Saldırısı
MQTT_SLOWITE	Yavaş DoS Saldırısı
HYDRA	Kaba Kuvvet Saldırısı
UFONet	Botnet Saldırısı
MQTTSA	Hatalı Biçimlendirilmiş Veriler
IoT-FLOCK	Zararlı MQTT Verileri

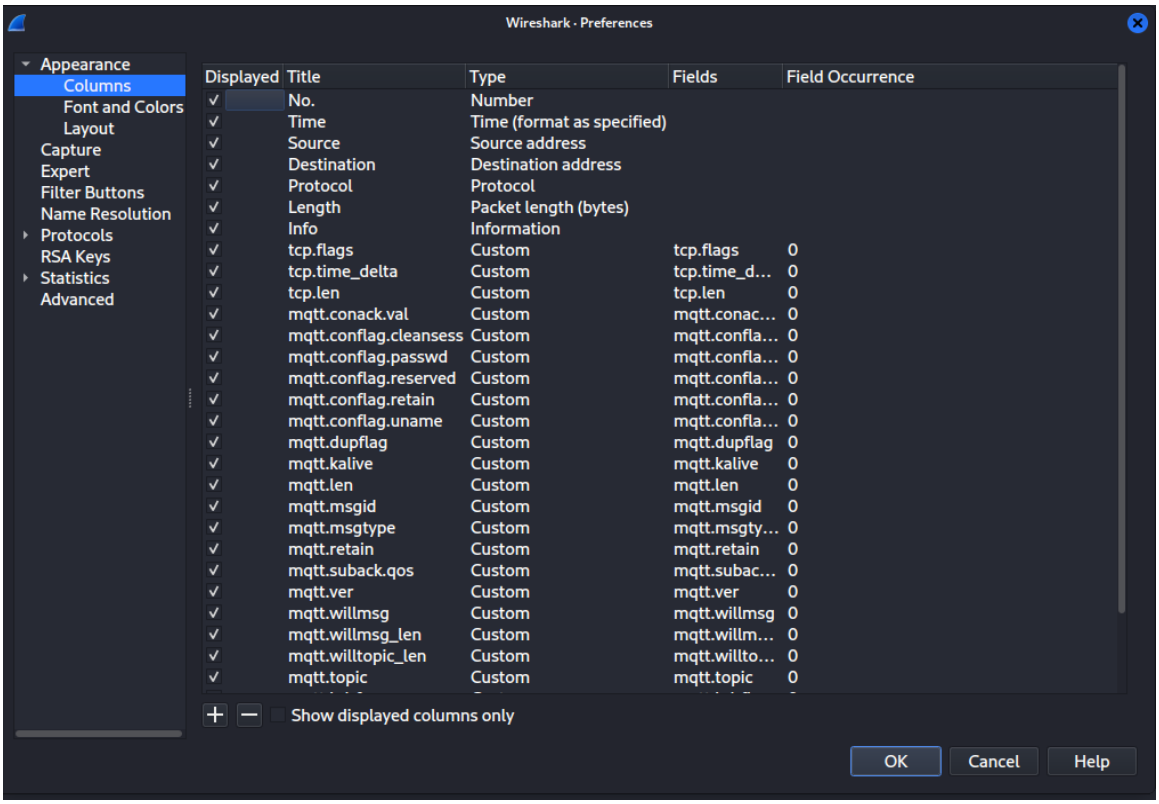
4.2.1. MQTTLens

MQTTLens (Sandro vd., 2018), MQTT broker'a bağlanabilen ve MQTT konularına abone olabilen, yayınlayabilen bir Google Chrome uygulamasıdır. Chrome tarayıcısında eklenti olarak entegre edilebilmektedir. MQTTLens aracı, MQTT broker'a mesaj yayınlamak ve konulara abone olmak için kullanılmıştır. Çizelge 4.2'de verilen konulara abone olunarak mesajlar görüntülenmiştir.

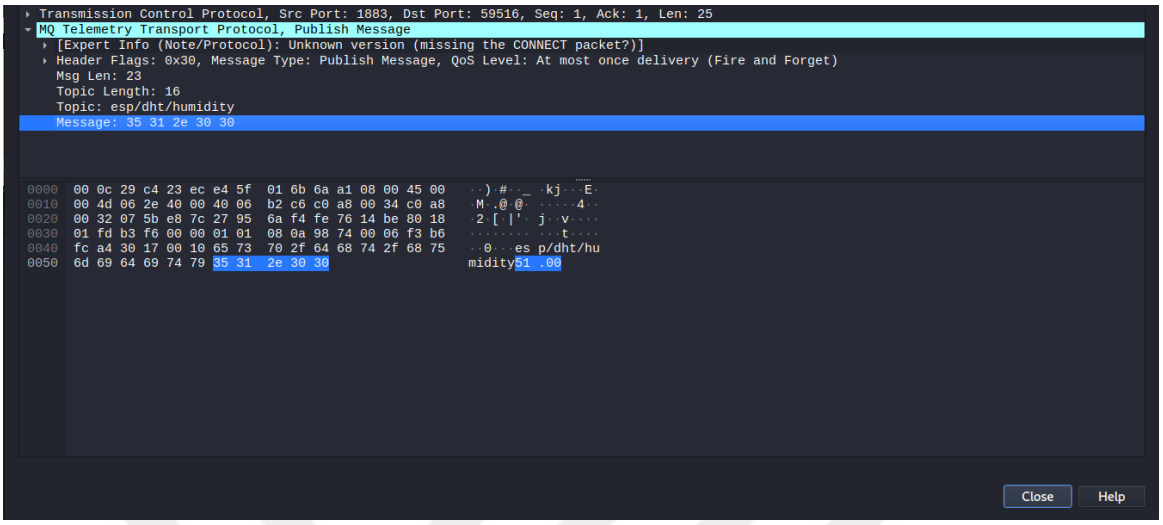
4.2.2. Wireshark

Wireshark ağ paket çözümleyicisi olduğu için ağ paketlerini yakalayarak ve bu paket verilerini olabildiğince ayrıntılı olarak görüntülemeyi sağlamaktadır. Paket dinleyicileri ağ trafiğini izleme amacıyla genellikle ağ ve sistem yöneticisi olan kişiler kullanmaktadır. Paket dinlemek için kullanılabilecek yazılımlara örnek olarak Wireshark, Tcpdump ve Network Miner verilebilir (Dal, 2019). Açık kaynaklı paket analizörü olması sebebiyle bu çalışmada tercih edilmiştir. Kali Linux işletim sisteminde kurulu bir şekilde gelmektedir.

Wireshark ile MQTT paketleri görüntülenebilmektedir. Wireshark üzerinde MQTT paketlerinin dinlenebilmesi için Tercihler sekmesinde MQTT protokol port bilgisinin kaydedilmesi gerekmektedir. MQTT öznitelikleri biliniyorsa Wireshark ile kolon olarak eklenebilmektedir. Şekil 4.5'te verildiği gibi bilinen ve görüntülenmesi istenilen öznitelikler eklenmiştir. Ayrıca TCP protokolu üzerinden paketleri dinlemek için aynı işlem gerçekleştirilmektedir.



Şekil 4.5: Wireshark tercihlerin ayarlanması



Şekil 4.6: Wireshark örnek paket yapısı

Wireshark ile görüntülenen MQTT paket örneği Şekil 4.6'da verilmiştir. MQTT paket örneğinde 1883 portundan "Esp/dht/humidity" konusunda nem değeri görüntülenmektedir.

4.2.3. WEKA

Bu çalışmada Waikato Üniversitesi'nde java programlama diliyle geliştirilmiş WEKA aracı kullanılmıştır. WEKA, kullanımı ücretsiz, pek çok sınıflandırma, regresyon, demetleme, bağıntı kuralları, yapay sinir ağları algoritmaları, öznitelik seçimi ve ön işleme metotları barındıran, yaygın kullanılan bir veri madenciliği aracıdır. Sınıflandırma işlemlerinde; veri kümesini istenilen oranda eğitim ve test kümesi olarak bölme işlemleri gerçekleştirilmektedir. Ayrıca test kümesi dışarıdan da eklenebilmektedir.

Explorer, Experimenter, Knowledge Flow ve Simple CLI adı verilen 4 temel uygulamayı barındırır (Coşkun vd., 2011). Package Manager sayesinde var olan kütüphaneler indirilebilir veya dışarıdan kütüphane entegre edilebilmektedir. WEKA'da grafik oluşturmak için Visualization sekmesi kullanılmaktadır. Desteklediği temel veri kaynakları, metin tabanlı arff, csv, c45, libsvm, svmlight, Xarff formatlarıdır; jdbc sürücüsü bulunan veri tabanlarına direk bağlantı yapabilir ve internet üzerinden http protokolünü kullanarak bu formatlara uygun dosyaları okuyabilme yeteneğine sahiptir (Coşkun vd., 2011). Makine öğrenmesi ve istatistik için birçok kütüphane WEKA'da yer almaktadır. Bu çalışmada "wekaDeeplearning4j"

kütüphanesi WEKA'ya entegre edilmiştir. ARFFViewer kullanılarak veriler farklı formatlarda kaydedilmiştir.

WekaDeeplearning4j kütüphanesi sinir ağlarını eğitmek ve dağıtmak için geliştirilmiş bir araç setidir (Lang vd., 2019). Bu kütüphane derin öğrenme uygulamaları ile verileri yükleyerek modelleri eğitmeyi sağlamaktadır. Eğitim kümesi yüklendikten sonra Classify sekmesinden "weka.classifiers.functions.Dl4jMlpClassifier" seçimi yapılmaktadır.

4.2.3. Mosquitto broker

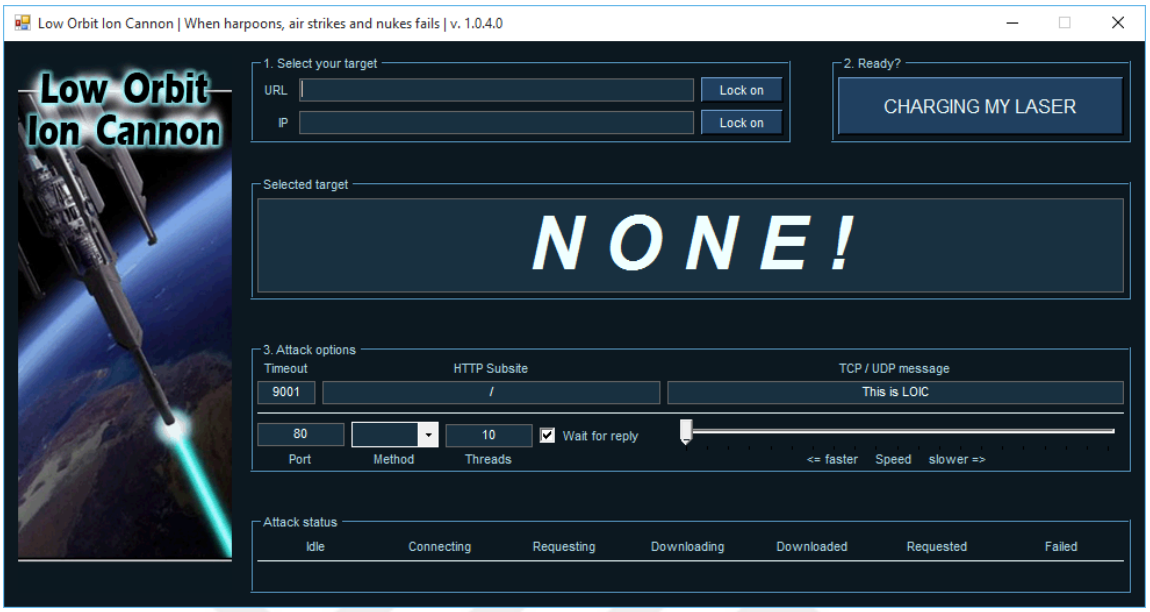
MQTT broker'lar, cihazlar arasında bağlantı kurulmasına ve mesaj gönderilmesine izin veren programlanabilir araçlardır. Broker'lar, sağladıkları iletişim protokolü soyutlaması ve çalışması nedeniyle IoT'ye uygundur. IoT, sensörler, elektronik cihazlar, akıllı telefonlar arasında bir bağlantı olarak yapılandırılır. MQTT broker'ların IoT cihazları ile iletişim kurabilmesi için ücretsiz araçlar mevcuttur (Oliveira vd., 2019). Çalışmalarda sıklıkla tercih edilen Mosquitto ve RabbitMQ broker'larıdır. Bunların yanısıra ActiveMQ, Mosca, EMQTT, Ponte, OpenMQ, Kafka ve Flume broker'ları da bulunmaktadır. Mosquitto ve RabbitMQ broker'larının performans değerlendirmelerinde seçilme sebepleri; ölçeklenebilirlik ve güvenirlik için bir bulut ortamına sahip olmasıdır. Bulut kullanılabilirliği ve yapılandırması ile ilgili olarak, diğer broker'ların Mosquitto ve RabbitMQ ile karşılaştırıldığında dezavantajlarının olduğu çalışmalarda bahsedilmektedir. Kullanılan broker'ların bir kısmı ise ücretlidir (Oliveira vd., 2019).

4.3. Gerçekleştirilen Siber Saldırılar

Bu çalışmada gerçekleştirilen siber saldırılar bu bölümde detaylandırılmıştır. Siber saldırıların gerçekleştirildiği araçlar ve kullanımlarına dair bilgiler verilmiştir.

4.3.1. Hizmet reddi saldırıları (DoS)

IoT ağlarında mesaj güvenliğini uçtan uca şifreleme teknikleriyle sağlanmaktadır. Ancak IoT'de işleyişi bozabilecek, kaynakları meşgul edebilecek ve nesnelerin enerjisini tüketebilecek eylemlerde söz konusudur. Bu saldırıların en önemlisi DoS/DDoS saldırılarıdır (Raza vd., 2013). DoS saldırıları, bir haberleşme sisteminde var olan kaynakların kötücül bir yaklaşımla kullanılarak bileşenlerin haberleşemez hale gelmesidir. Saldırının birden fazla noktadan yapılıyorsa DDoS saldırıları denilmektedir. Bu saldırılarda TCP/IP protokol yapısındaki açıklardan faydalanılarak ve kaynaklar tüketilerek haberleşme engellenmektedir. DoS ve DDoS saldırılarını gerçekleştiren bazı uygulamalar mevcuttur. Bu çalışmada LOIC saldırı kullanılmıştır. LOIC saldırı aracı C# ile yazılmış açık kaynaklı hizmet reddi saldırı uygulamasıdır. LOIC ile DoS ve DDoS saldırıları gerçekleştirilmektedir. Hedef sunucuya TCP, UDP veya HTTP paketleri göndererek haberleşmenin durdurulması sağlanmıştır.



Şekil 4.7: LOIC saldırı aracı

Şekil 4.7'de LOIC saldırı aracının ekran görüntüsü verilmiştir. URL veya IP adresi girildikten sonra "Lock on" seçilmektedir. Opsiyonlarda port, method ve thread ayarları yapılabilmektedir. Saldırıyı başlatmak için "Charging my laser" seçilmektedir. Bu çalışmada saldırıyı görüntülemek için Wireshark ile port ve protokol bazında filtre uygulanmıştır. Filtre işleminden sonra CSV formatında veriler kaydedilmiştir.

4.3.2. Yavaş hizmet reddi saldırıları

Nesnelerin İnternetinde Yavaş DoS (SlowITe) Saldırısı, MQTT protokolünü hedefleyen yeni bir hizmet reddi tehdididir. Bu tür tehdit, hizmet reddi yürüten bir ağ hizmetini hedeflemek için minimum saldırı bant genişliği ve kaynakları kullanan Yavaş DoS Saldırıları kategorisine aittir. Yavaş DoS Saldırıları yalnızca TCP tabanlı protokolleri hedefleyebildiğinden, MQTT hizmeti, TCP üzerinden çalıştığı için SlowITe tarafından bir hedef olarak kabul edilir (Vaccari vd., 2020). Özellikle SlowITe'nin amacı, MQTT aracısının aynı anda yönetebildiği tüm kullanılabilir bağlantıları ele geçirmek için sunucuyla çok sayıda bağlantı başlatmaktır.

Mevcut tüm bağlantılar (uygulama katmanında) saldırgan tarafından kurulduğunda, DoS durumuna ulaşılır. Bu noktada saldırganın amacı, aynı anda mümkün olan en düşük bant

genişliğini kullanarak MQTT aracısını mümkün olduğunca uzun süre meşgul tutmak, böylece DoS durumunu korumaktır.

Örneğin, Slowloris gibi diğer SDA'lar için bu, HTTP'de GET isteğinin başlangıcını sunucuya göndererek yapılır. Bununla birlikte, daha önce belirtildiği gibi, istemcinin MQTT aracısına kimliğini doğrulamak için CONNECT paketine ihtiyaç vardır. Bu nedenle, saldırının böyle bir paketi sunucuya göndermesi gerekir. SlowITe, broker ile iletişim başlatmak için MQTT tarafından desteklenen CONNECT paketinden yararlanmaktadır (Vaccari vd., 2020).

Sunucunun tüm bağlantılarını düşük oranlı tekniklerle ele geçirmek olduğundan, bant genişliği, CPU veya bellek açısından sunucu üzerindeki etkisi ihmal edilebilir düzeydedir. Yavaş DoS Saldırılarının ortak bir özelliğidir. Saldırıları tespit etmeyi zorlaştırdığından ihmal edilmemelidir. Kali Linux ve Windows işletim sistemlerinde kullanılan Yavaş DoS Saldırı araçları bulunmaktadır. Bu çalışmada Kali Linux işletim sistemi tercih edildiği için RUDY ve Slowloris saldırı araçları kullanılmıştır.

- ***RUDY:***

RUDY düşük ve yavaş bir saldırı olarak kategorize edilmektedir. Yüksek hacimli hızlı isteklerle bir sunucuyu meşgul etmek yerine birkaç uzatılmış istek oluşturmaya odaklanmaktadır. Başarılı bir RUDY saldırısı, kurbanın web sunucusunun meşru trafik için kullanılamaz hale gelmesine neden olmaktadır. Bu araç, yükü her biri 1 bayt kadar küçük paketlere bölerek bu paketleri sunucuya her biri yaklaşık 10 saniyelik rastgele aralıklarla gönderir. Araç, süresiz olarak veri göndermeye devam eder. Saldırının davranışı, form verisi gönderen yavaş bağlantı hızına sahip bir kullanıcının davranışına benzer olduğundan, web sunucusu paketleri kabul etmek için bağlantıyı açık tutmaktadır (Chaddha, 2018).

Şekil 4.8: RUDY saldırı aracı

Şekil 4.8'de RUDY saldırı aracı ekran görüntüsü verilmiştir. Burada "-t" veya "--target" yazılması zorunlu olup IP adres/Web adres olarak yazılmıştır. Opsiyonel verilebilen "-d" veya "-- delay" değeri varsayılan olarak 5, "-n" veya "--numberOfConnections" değeri 500 olarak verilmiştir. Port değeri girilmediği takdirde varsayılan olarak 9050 portunu hedef alacağı için "--torPort" ile 1883 değeri verilmiştir. Port değerinin girilmesi için "-o" veya "--torPort" kullanılmaktadır. Saldırının kurulumu, kullanım örnekleri ve opsiyonlar ile ilgili detaylı bilgiler Github adresi üzerinden erişilebilmektedir (Chadda, 2018). Saldırılar anlık olarak Wireshark aracı ile izlenmiş ve kaydedilmiştir. Veriler CSV formatında kaydedilmiştir.

- ***MQTT_SLOWITE:***

MQTT_SLOWITE, Github'da bulunan ücretsiz ve açık kaynaklı bir araçtır. Bu aracı kullanarak Yavaş DoS Saldırısı gerçekleştirilmektedir. Pisa Üniversitesi'nde geliştirilmiş olan bu program Python ile yazılmıştır. Eclipse Paho MQTT Python istemci kütüphanesini kullanarak mesaj yayınlamak, abone olmak ve MQTT broker'a Yavaş DoS saldırısı gerçekleştirmek için yazılmıştır (Carli vd., 2020).

Kaynak kodlar Python ile yazıldığı için Python yazılım paketlerinin kurulumları gerçekleştirilmiştir. Mosquitto MQTT broker kurulumu yapıldıktan sonra Şekil 4.9'da verildiği gibi IP ve port bilgileri yazılmıştır. Github'da paylaşılan kaynak kodlardan "MQTT_SlowDoS.py" çalıştırılarak SlowITE saldırısı başlatılmıştır.

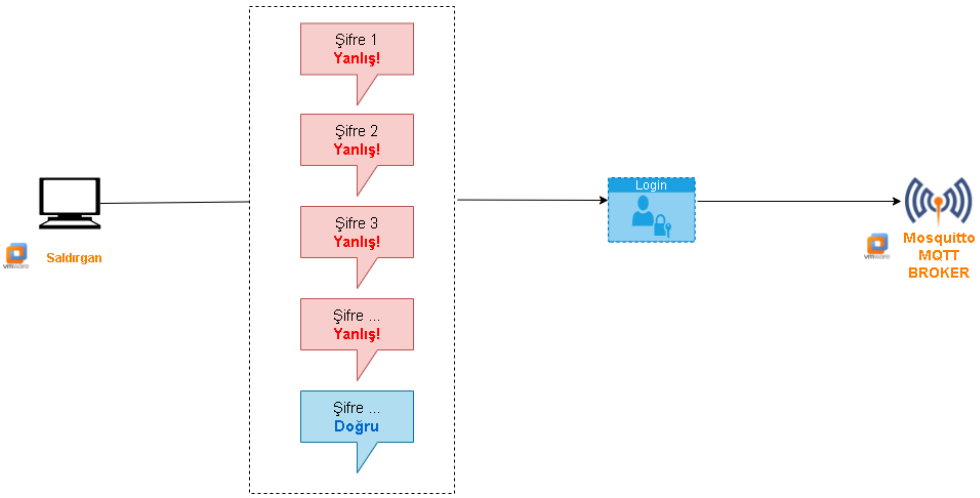
```
┌──(root💀kali)-[/home/…/TOOLS/MQTT_SlowITe-main/MQTT_SlowITe-main/MQTT_attack_slowITe]
└─# python3 MQTT_SlowDoS.py -a 192.168.0.52 -p 1883

Requesting connections ...
```

Şekil 4.9: MQTT SLOWITE saldırıları

Şekil 4.9'da görüldüğü gibi -a MQTT broker IP adresi, -p broker port bilgisidir.

4.3.3. Kaba kuvvet saldırıları

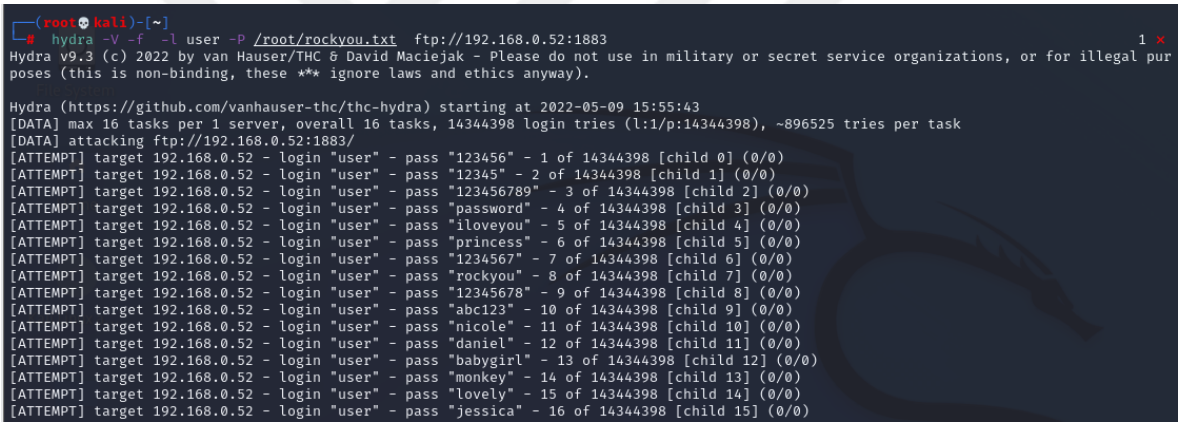
Kaba Kuvvet Saldırıları, hedef sistemde kayıtlı şifre ve parolaları kırmak için en sık tercih edilen siber saldırılardan biridir. Kayıtlı şifre ve parolalar kullanıcı bilgileri, kredi kart bilgileri, sosyal medya hesap bilgileri vb. olabilmektedir. Kaba Kuvvet Saldırısında hedef sistem üzerinde, yönetici yetkisine erişim sağlayarak saldırganın istediği dosyalara tam erişim sağlaması ile sonuçlanmaktadır. Saldırgan kamera, fare veya klavye gibi araçları da tam erişim sayesinde kontrol edebilmektedir. Yönetici hesabı şifre ile korunuyorsa, Kaba Kuvvet Saldırısı ile bu şifre kırılmaktadır (Kara, 2019). Bu çalışmada da olduğu gibi harfler ve özel karakterler bulunduran kullanıcı adı ve şifre listesi hazırlanmaktadır. Bu liste ile şifreye ulaşmak için saldırı işlemleri gerçekleştirilmektedir. Şekil 4.10'da Kaba Kuvvet Saldırılarının çalışmada uygulanan diyagramı verilmiştir.



Şekil 4.10: Kaba kuvvet saldırıları

- ***HYDRA:***

Kali Linux işletim sisteminde, Kaba Kuvvet Saldırıları gerçekleştirmek için yüklü olarak bulunan Hydra aracı kullanılmıştır. Bu araç birçok protokolü desteklemektedir. Ayrıca Github üzerinden indirilerek çalıştırılması sağlanabilmektedir. Yaygın olarak kullanılan "rockyou" şifre listesi bu çalışmada kullanılmıştır. Bu çalışmada saldırgan, oluşturulmuş şifre listesi ile Mosquitto broker'a giriş işlemi için testler yapmaktadır. Şifre bulunana kadar test işlemi devam etmektedir. Doğru şifre eşleşmesi yakalanırsa saldırı otomatik olarak durmaktadır. Saldırgan ve broker makine Kali Linux işletim sisteminde çalışmaktadır.

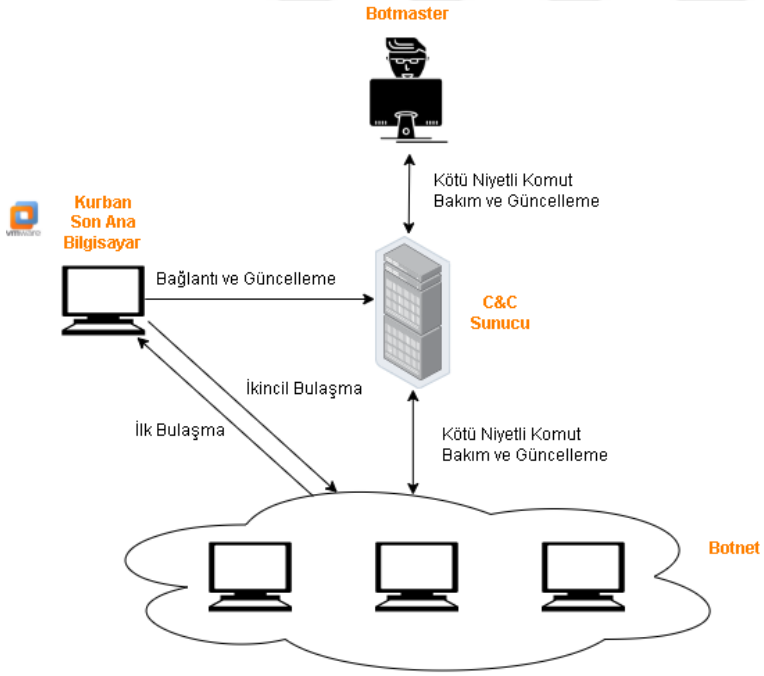


Şekil 4.11: HYDRA saldırı aracı

Şekil 4.11'de verildiği gibi "-V" ayrıntılı modda görüntülemek, "-f" şifre eşleşmesi bulunduğunda saldırıyı sonlandırmak, "-l" oturum açılacak olan kullanıcı adı (kullanıcı listesi de dahil edilebilir), "-p" listeden ya da belirli bir şifre ile giriş testi yapmak için kullanılmıştır. Saldırı FTP veya IMAP olarak başlatılmaktadır. FTP, Dosya Transfer Protokolü olarak dosya transfer işlemi için kullanılan bir protokoldür. İnternete bağlı iki bilgisayar arasında dosya transferini sağlamaktadır. IMAP ise her yerden ve her cihazdan e-posta erişimini sağlamaktadır. Bu çalışmada e-posta ile ilgili işlem yapılmadığından FTP olarak saldırılar yapılmıştır.

4.3.4. Botnet saldırıları

Botnet, güvenlik savunmaları ihlal edilmiş ve kontrolünün kötü niyetli bir tarafa, blackhat topluluğuna bırakılmış, internete bağlı bilgisayarlardan oluşan bir koleksiyondur. Güvenliği ihlal edilmiş bilgisayar grubu, "Botmaster" olarak bilinen bir veya saldırgan grubu tarafından kontrol edilmektedir. Botnet operatörleri, bu tehlikeli faaliyetlerin etkisini artırmak için birçok botun toplam gücünü kullanabilir. Tek bir bot internet için bir tehlike olmayabilir, ancak bir bot ağı kesinlikle büyük arızalar yaratabilmektedir. Botnet kapsamındaki başlıca saldırılar, DDoS, Tarama, Kimlik Avı, Tıklama Sahtekarlığı, Spam göndermedir (Vania vd., 2013). Botnet Saldırısı beş aşamadan oluşmaktadır: ilk bulaşma, ikincil bulaşma, bağlantı, kötü niyetli komut/kontrol, güncelleme ve bakım. Şekil 4.12'de Botnet yaşam döngüsü verilmiştir.



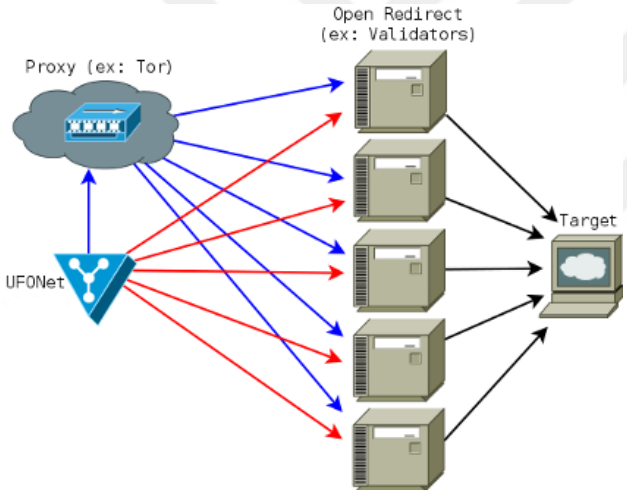
Şekil 4.12: Botnet yaşam döngüsü (Vania vd., 2013)

1. **İlk bulaşma:** Saldırgan, başlangıçta bilinen güvenlik açığı için bir hedef alt ağı tarar ve farklı yöntemlerle kurban makinelere bulaşır. Web sayfalarından ikili indirme, e-posta eki, USB otomatik çalıştırma ve bazı kötü amaçlı programlar gibi yöntemler kullanır (Vania vd., 2013).
2. **İkincil bulaşma:** Etkilenen ana bilgisayarlar, kabuk kodu olarak bilinen komut dosyasını yürütür. Bu ikili dosyayı FTP, HTTP veya P2P aracılığıyla belirli bir konumdan indirir. Kod yüklenir ve hedeflenen makine "Zombie" olarak da bilinen "Bot" haline gelir (Vania vd., 2013).
3. **Bağlantı:** Bağlantı aşamasında bot programı komuta kontrol (C&C) kanalı kurar ve zombiyi komuta kontrol sunucusuna bağlar. C&C kanalının kurulmasıyla zombi, saldırganın botnet ordusunun bir parçası olur (Vania vd., 2013).
4. **Komuta ve kontrol:** Botmaster, komutları bot ordusuna yaymak için C&C kanalını kullanır. Komut iletişimi, tek hata noktasından kaçınmak için IRC tabanlı, HTTP tabanlı, DNS tabanlı veya P2P protokolü kullanılarak olabilir. Bot programları, BotMaster tarafından gönderilen komutları alır ve yürütür. C&C kanalı, bot yöneticisinin çeşitli yasa dışı faaliyetleri yürütmek için çok sayıda botun eylemini uzaktan kontrol etmesini sağlar (Vania vd., 2013).
5. **Güncelleme ve bakım:** Botların canlı ve güncel olması için komut verilir. Böylece, bulunacak ve kontrol edilecek herhangi yeni bir çözüm tespit edildiğinde, kontrol merkezi (Botmaster) onu yeni stratejilerle güncelleyebilir veya yeni işlevler ekleyebilir (Vania vd., 2013).

Bu çalışmada kullanılan Botnet saldırı aracı UFONet, DoS ve DDoS saldırılarının gerçekleştirilmesine olanak sağlamaktadır. Bu araç açık kaynak kodlu bir yazılımdır.

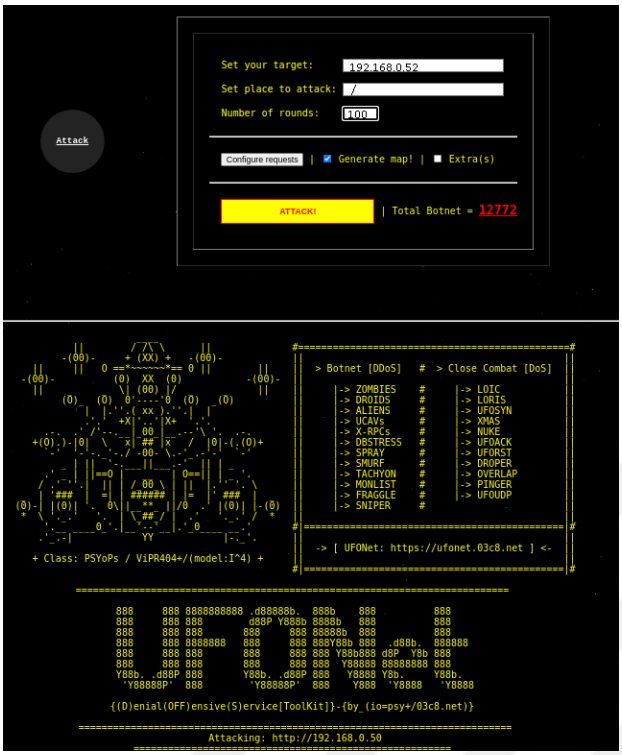
- ***UFONet:***

UFONet bir web arayüz sayesinde kolaylıkla yönetilmektedir. P2P ve kripto grafik - yıkıcı araç takımı olarak bilinmektedir. DoS ve DDoS saldırılarının yanı sıra, küresel bir istemci/sunucu ağı oluşturarak içerik yayınlamak ve almak için şifreli bir DarkNET olarak da çalışmaktadır. Üçüncü taraf web sitelerinde bir Botnet gibi davranmak için Açık Yönlendirme Vektörlerinin sömürülmesi yoluyla Katman 7'de (APP/HTTP) ve protokolü kötüye kullanarak Katman3'te (Ağ) saldırılar gerçekleştirir. Katman 7'de DDoS saldırıları başlatırken, Katman 3'te TCP/SYN sel saldırılarını başlatmaktadır (Anonim, 2007).



Şekil 4.13: UFONet DDoS saldırı şeması (Anonim., 2007)

Şekil 4.13'te verilen şema, HTTP/WebAbuse DDoS saldırısını gerçekleştirirken isteklerin nasıl yapıldığını gösteren bir mimari örneğidir.



Şekil 4.14: UFONet arayüzü

Şekil 4.14'teki UFONet web ara yüzünde hedef adres ve tur sayısı bilgileri verildiği gibidir. Toplam BotNet sayısını girilen bilgilere göre otomatik olarak hesaplamaktadır.

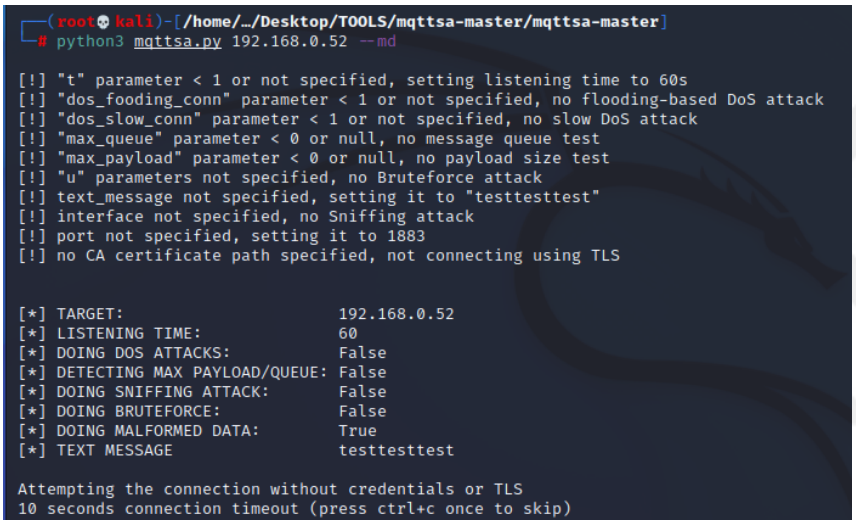
4.3.5. Hatalı biçimlendirilmiş veriler

Hatalı biçimlendirilmiş veriler için hedeflenen bilgisayarda hatalı biçimlendirilmiş paketler oluşturup broker'a göndermeyi amaçlamaktadır. MQTTSA aracını kullanarak hatalı biçimlendirilmiş CONNECT ve PUBLISH paketleri gönderilmiştir.

- ***MQTTSA:***

MQTTSA, Python ile yazılmış açık kaynak kodlu bir yazılımdır. MQTT mesajlaşma protokolünün açık kaynaklı bir istemci uygulamasını sağlayan bu araç, Paho3 ve Wireshark5'in komut satırı aracına bir arabirim sağlayan pyshark4 kütüphaneleri ile yazılmıştır (Palmieri vd.,

2019). Bilinen güvenlik açıklarını ortaya çıkarmak için bir dizi saldırı modelini otomatik olarak başlatarak MQTT broker'lardaki olası güvenlik açıklarını tespit etme işlevi de bulunmaktadır. MQTTSA aracı ile veri ayrıştırma ve sızma, kimlik bilgisi alma, broker parmak izi, Kaba Kuvvet kimlik doğrulama, veri kurcalama, DoS saldırıları ve bunlara bağlı olarak raporlar oluşturulabilmektedir. Bu çalışmada MQTTSA aracı Github üzerinden indirilerek Kali Linux işletim sisteminde çalıştırılmıştır.

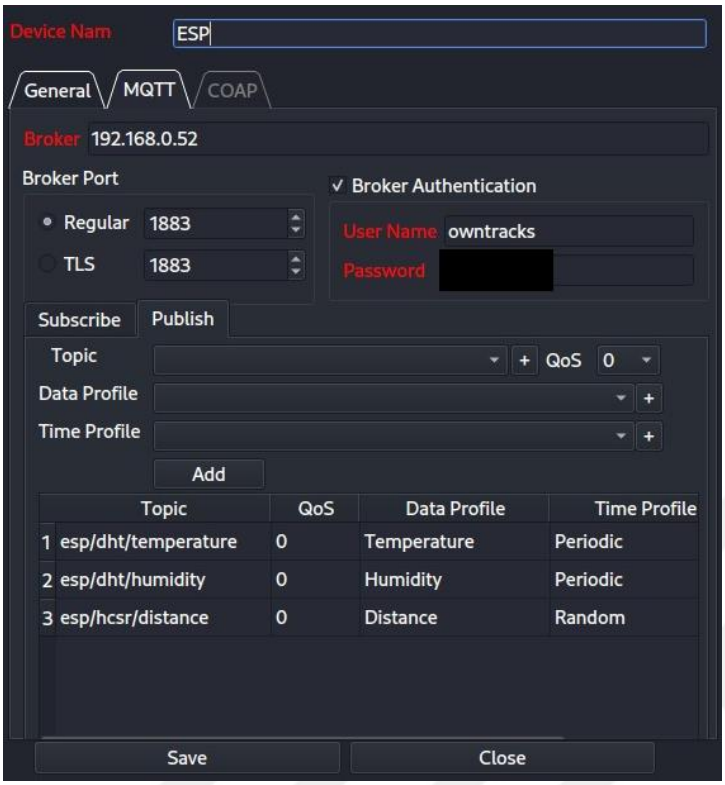


Şekil 4.15: MQTTSA aracı

Şekil 4.15'te verildiği gibi hedef adres, port bilgisi eklenmiştir. Ayrıca hatalı biçimlendirilmiş olmasını sağlamak için "--md" eklenmiştir.

4.3.6. Zararlı MQTT verileri

Kötü amaçlı bir IoT cihazı, sunucunun tüm kaynaklarını ele geçirmek için, bağlantı yuvaları, ağlar veya diğer kaynaklar açısından periyodik olarak büyük miktarda kötü amaçlı MQTT verisi gönderir. Bu saldırı birden çok bağlantı oluşturmak yerine tek bir bağlantı kullanarak kaynakları doyurmaya çalışır (Vaccari vd., 2020). Bu saldırı, bu durumda IoT-Flock aracının içindeki bir modül kullanılarak oluşturulmuştur (Ghazanfar vd., 2020). IoT- Flock Github üzerinden erişilen proje QTCreator 5'e Linux işletim sisteminde entegre edilmiştir.



Şekil 4.16: IoT-Flock aracı

Şekil 4.16'da IoT-Flock aracında yer alan "MQTT Publish Flood" modülünün kullanımı hakkında detaylar verilmiştir. Zararlı MQTT verileri IoT-Flock ile oluşturulmuş ve Wireshark aracı ile kaydedilmiştir.

4.4. MQTTset Veri Kümesi

Bu bölümde MQTTset veri kümesi hakkında bilgiler, veriler üzerinde gerçekleştirilen işlemler, çalışmanın akış diyagramı detaylı şekilde anlatılmıştır.

MQTTset veri kümesi, MQTT ve CoAP protokollerini temel alan ağları taklit edebilen bir ağ trafiği oluşturma aracı olan IoT-Flock kullanılarak oluşturulmuştur (Vaccari vd., 2020). Oluşturdukları MQTT ağ trafiğinde PCAP dosyası olarak verileri kaydetmişlerdir. Veri kümesi 11.915.716 ağ paketi ve toplam 1.093.676.216 bayt boyutuyla temsil edilmiştir. 3 saldırı, 1 normal, 1 bozulmuş ve 1 meşru olmak üzere 6 farklı sınıf içeren MQTTset veri kümesinin zenginleştirilmesi amacıyla bu çalışmada MQTTset verilerine ek olarak Botnet saldırısı

eklenmiştir. Elde edilen veriler yeni bir sınıf olarak eğitim ve test verilerine eklenmiştir. Ayrıca DoS, Kaba Kuvvet, Yavaş DoS, bozulmuş veriler, zararlı veriler ve meşru veriler için ağdan toplanan veriler MQTTset verilerine eklenmiştir. Çizelge 4.4'te MQTTset veri kümesindeki eğitim ve test verilerinin toplam adetleri görülmektedir. Bu çalışmada ev ağından toplanan toplam veri adetleri ile karşılaştırması Çizelge 4.4'te verilmiştir. Botnet saldırısı için veriler MQTTset veri kümesinde bulunmamaktadır.

Çizelge 4.4. MQTTset veri kümesi ve ağdan toplanan veriler

Saldırı Adı	**MQTTset Toplam Veri Adetleri**	**Ev Ağından Toplanan Toplam Veri Adetleri**
DoS Saldırısı	128,233	11,191
SlowITE Saldırısı	9,202	112,353
Kaba Kuvvet Saldırısı	14,501	107,982
Bozulmuş Veriler	10,924	96,261
Zararlı Veriler	613	32,671
Meşru Veriler	165,463	64,183
Botnet Saldırısı	-	139,424

Örneğin DoS saldırısı için MQTTset veri kümesinde toplam 128,233 adet veri bulunmaktadır. Ev ağından toplanan 11,191 veri ile birleştirilerek toplam veri adedi üzerinde 5 kat çapraz doğrulama ile eğitim ve test kümesi seçimi yapılmıştır. Çizelge 4.5'te MQTTset veri kümesinde yer alan özniteliklerin adı, açıklaması ve protokol katmanı verilmiştir.

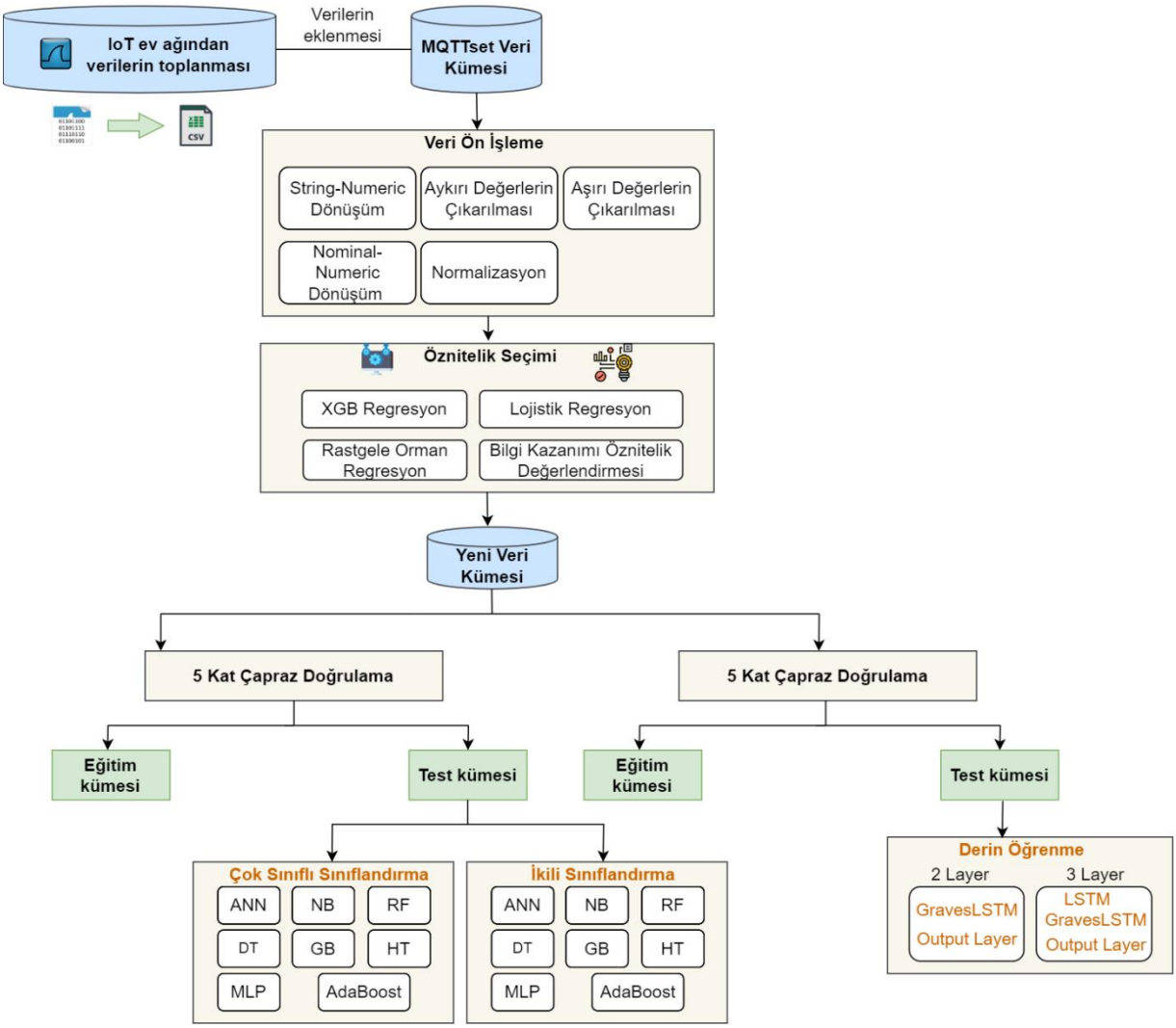
Çizelge 4.5. MQTTset veri kümesi öznitelik listesi (Vaccari vd., 2020)

No	Nitelik Adı	Açıklama	Protokol Katmanı
1	tcp.flags	TCP Flags	TCP
2	tcp.time_delta	Time TCP stream	
3	tcp.len	TCP Segment Length	
4	mqtt.conack.flags	Acknowledge Flags	MQTT
5	mqtt.conack.flags.reserved	Reserved	
6	mqtt.conack.flags.sp	Session Present	
7	mqtt.conack.val	Return Code	
8	mqtt.conflag.cleansess	Clean Session Flag	
9	mqtt.conflag.passwd	Password Flag	
10	mqtt.conflag.qos	QoS Level	
11	mqtt.conflag.reserved	Reserved	
12	mqtt.conflag.retain	Will Retain	
13	mqtt.conflag.uname	User Name Flag	
14	mqtt.conflag.willflag	Will Flag	
15	mqtt.conflags	Connect Flags	
16	mqtt.dupflags	DUP Flag	
17	mqtt.hdrflags	Header Flags	
18	mqtt.kalive	Keep Alive	
19	mqtt.len	Message Length	
20	mqtt.msg	Message	
21	mqtt.msgid	Message Identifier	
22	mqtt.msgtype	Message Type	
23	mqtt.proto_len	Protocol Name Length	
24	mqtt.protoname	Protocol Name	
25	mqtt.qos	QoS Level	
26	mqtt.retain	Retain	
27	mqtt.sub.qos	Requested QoS	
28	mqtt.suback.qos	Granted QoS	

Çizelge 4.5. MQTTset veri kümesi öznitelik listesi (Vaccari vd., 2020) (devam)

29	mqtt.ver	Version	MQTT
30	mqtt.willmsg	Will Message	
31	mqtt.willmsg_len	Will Message Length	
32	mqtt.willtopic	Will Topic	
33	mqtt.willtopic_len	Will Topic Length	
34	target	Sınıf	

IoT ev ağından elde edilen veriler ve MQTTset veri kümesi verileri birleştirilmiştir. Bu işlem sonucunda veri ön işleme, öznitelik seçimi gerçekleştirilmiştir. Şekil 4.16'da verildiği gibi veri ön işleme yapılırken string-numeric dönüşümler, nominal-numeric dönüşümler, aykırı ve aşırı değerlerin çıkarılması ve normalizasyon işlemleri uygulanmıştır. WEKA uygulamasında birçok sınıflandırma algoritmasında ve DNN çalışmalarında nominal değer kullanımı (target dışında) yapılamamaktadır. WEKA kullanılan çalışmalarda da veri ön işleme sıklıkla kullanıldığı görülmüştür. Öznitelik seçiminde Lojistik regresyon, rastgele orman regresyon, XGBoost regresyon ve bilgi kazanımı öznitelik değerlendirmesi uygulanmıştır. Elde edilen yeni veri kümesinde WEKA aracılığı ile 5 kat çapraz doğrulama uygulanmıştır. Test kümesi ile çok sınıflı sınıflandırma ve ikili sınıflandırma olarak iki ayrı şekilde testler yapılmıştır. Çok sınıflı ve ikili sınıflandırmada kullanılan algoritmalar Şekil 4.16'da verilmiştir. Bölüm 4.4'te sınıflandırma algoritmaları ve kullanım yöntemlerine dair detaylı bilgiler yer almaktadır. 5 kat çapraz doğrulama sonucunda uygulanan bir diğer yöntem DL'dir. 2 katmanlı ve 3 katmanlı DL'ye bu çalışmada yer verilmiştir. Şekil 4.17'de veri kümesi ve ağ üzerinde gerçekleştirilen işlemler özetlenmiştir.



Şekil 4.17: Çalışmanın akış diyagramı

4.4.1. Veri kümesi ön işleme

Veri ön işleme, herhangi bir ML modeli tarafından kullanılmadan önce veri toplama sürecinin ilk adımıdır. Genellikle ham verileri ML modelinin işleyebileceği bir yapıya dönüştürmek için kullanılır ve ayrıca modelin kalitesinin artmasına yardımcı olur (Mahfouz vd., 2020). Sınıflandırma ve sonuç değerlendirme, önerilen modeldeki en önemli aşamalardır. Veri kümesinin performansı üzerinde oldukça etkilidir. Veri kümeleri sembolik özellikler içerebilmektedir. Fakat sınıflandırıcılar, bu sembolleri işleyemez. Ön işlemeye bu sebeple

ihtiyaç duyulmaktadır. Tüm sayısal olmayan veya sembolik özellikler veri kümesinden kaldırılır veya değiştirilir (Halimaa vd.,2019). Veri ön işleme de birçok yöntem kullanılabilmektedir.

Akıllı ev ağından elde edilen veriler üzerinde semboller kaldırılmış, sayısal olmayan (nominal ve string) değerler sayısal (numeric) değerlere dönüştürülmüştür. Bu işlemler WEKA aracında yer alan filters→ unsupervised→ attribute içerisindeki fonksiyonlar yardımıyla gerçekleştirilmiştir. Bunun yanısıra bu çalışmada normalleştirme, standartlaştırma ve öznitelik seçimi yöntemleri kullanılmıştır.

- ***Normalleştirme:***

Normalleştirme, genellikle ML modelleri için veri hazırlamanın bir parçası olarak uygulanan bir ön işleme yöntemidir. Normalleştirmenin amacı, gerçek değer aralıklarındaki farklılıkları bozmadan ve bilgi kaybetmeden ortak bir ölçek kullanarak bir veri kümesindeki sayısal verilerin değerlerini bir aralığa (genellikle 0–1) ayarlamaktır (Mahfouz vd., 2020). WEKA aracında filters → unsupervised → attribute altında yer alan "Normalize" fonksiyonu ile 0-1 aralığına değerler ayarlanmıştır.

- ***Aykırı ve uç değerlerin ayrıştırılması:***

Aykırı veya uç değerler, beklenen değer aralığının dışında kalan değerlerdir. Sınıflandırıcıları etkilemesi sebebiyle, bu değerlerin belirlenmesi ve çıkarılmasıyla büyük ölçüde fayda sağlamaktadır. Aykırı değerlerin olumsuz etkilerinden bazıları, hata varyansını artırabilmeleri, normalliği azaltabilmeleri, yanlılık gösterebilmeleri veya önemli ölçüde ilgi çekici olabilecek tahminleri etkileyebilmeleridir (Nair vd., 2019). WEKA aracı ile IQR tekniği sayesinde aykırı ve uç değerler ayrıştırılıp veri kümesinden çıkarılmıştır. WEKA aracında filters → unsupervised → attribute → InterqualiteRange fonksiyonu yer almaktadır.

4.4.2. Veri kümesi öznitelik seçimi

Öznitelik seçimi, veri kümesinin üzerinde çalıştığı tahmine dayalı modelleme problemiyle en ilgili özniteliklerini seçme sürecidir. Özellik seçme teknikleri, daha az veri gerektirirken daha iyi doğruluk ve daha az karmaşıklık sağlayacak özellikleri seçerek doğru bir tahmine dayalı model oluşturmaya yardımcı olmaktadır (John vd., 1994).

- ***Lojistik regresyon:***

Lojistik regresyon özellikler arasındaki, ilişkiyi açıklayan, ikili sınıflandırma problemlerinde sıklıkla kullanılan bir yaklaşımdır. Bu yaklaşımda sınıf etiketleri 0,1 şeklinde verilmektedir. Geleneksel sınıflandırma yöntemlerinden biri olan lojistik regresyon tıbbı sınıflandırma çalışmalarında da kullanılmaktadır. Bu çalışmada Python paketlerinden olan scikit-learn kullanılarak Kaggle platformunda yazılım gerçekleştirilmiştir. Bu yazılım ile lojistik regresyon yöntemi kullanılarak öznitelik seçimi yapılmıştır. Sonuç olarak en önemli özniteliklerin 22 adet olduğu görülmüştür.

- ***XGB regresyon:***

Boosting, regresyon ve sınıflandırma problemleri için kullanılan bir makine öğrenme tekniğidir. Her adımda bir öğrenici oluşturur ve bunu toplam modelde biriktirir. Her adım için öğrenici, kayıp fonksiyonunun gradyan yönünü temel alıyorsa, Gradyan Artırma Makineleri olarak adlandırılabilir. Rastgele Orman ve Gradyan Artırma Makineleri arasındaki temel fark, RF'de ağaçlar birbirinden bağımsız olarak inşa edilirken, GBM'nin halihazırda inşa edilmiş olanları tamamlamak için yeni bir ağaç eklemesidir (Pan, 2018). Bu çalışmada XGB regresyon uygulamak için Python'da scikit-learn paketi kullanılmıştır. XGB regresyon ile en önemli özniteliklerin 21 adet olduğu görülmüştür.

- ***Rastgele orman regresyon:***

Rastgele orman, her ağacın girdi vektöründen bağımsız olarak örneklenen rastgele bir vektör kullanılarak oluşturulduğu ağaç tahmincilerinin bir kombinasyonundan oluşan bir sınıflandırma ve regresyon yöntemidir. Regresyonda, ağaç tahmincisi, rastgele orman sınıflandırıcısı tarafından kullanılan sınıf etiketlerinin aksine sayısal değerler almaktadır (Singh vd., 2017). Bu çalışmada kullanılan rastgele orman regresyonu, Python paketi olan scikit-learn ile gerçekleştirilmiştir. Öznitelik sıralaması dikkate alındığında en önemli öznitelik sayısı 21 adet olarak görülmüştür.

- ***Bilgi kazanımı öznitelik değerlendirmesi:***

Bilgi kazanımı yönteminde örneklerin sınıflandırılması işleminde özniteliklere ait etkin değerlerin aralıkların bulunmasında kullanılmaktadır. Bilgi kazanımında belirli bir öznitelik dikkate alınarak örneklerin sınıflandırılması işlemleri gerçekleştirilir. Entropinin (belirsizliğin) indirgenmesi hedeflenir. Sınıflandırmada bilgi kazanımının yüksek olması durumlarında belirsizlik azalacağı için etkin sınıflandırma sağlanmaktadır (Kumova vd., 2016). Bu çalışmada WEKA aracı ile InfoGainAttributeEval kullanılarak veri kümesindeki özniteliklerin önem sıralaması Ranker metodu ile yapılmıştır. Ranker sonucunda 21 özniteliğin ilk sıralarda yer aldığı görülmüştür.

XGB regresyon, lojistik regresyon, rastgele orman regresyon ve bilgi kazanımı öznitelik değerlendirmesi sonucunda kesişen öznitelikler 21 adet olarak bulunmuştur. Bulunan 21 öznitelik Çizelge 4.6'da koyu renk yazı ile ifade edilmiştir. Diğer 13 öznitelik MQTTset veri kümesinden çıkarılarak ML ve DNN uygulamaları gerçekleştirilmiştir.

Çizelge 4.6. MQTTset veri kümesi öznitelik seçimi sonuçları

No	Nitelik Adı	Açıklama	Protokol Katmanı
1	**tcp.flags**	**TCP Flags**	TCP
2	**tcp.time_delta**	**Time TCP stream**	
3	**tcp.len**	**TCP Segment Length**	
4	mqtt.conack.flags	Acknowledge Flags	MQTT
5	mqtt.conack.flags.reserved	Reserved	
6	mqtt.conack.flags.sp	Session Present	
7	**mqtt.conack.val**	**Return Code**	
8	**mqtt.conflag.cleansess**	**Clean Session Flag**	
9	**mqtt.conflag.passwd**	**Password Flag**	
10	mqtt.conflag.qos	QoS Level	
11	**mqtt.conflag.reserved**	**Reserved**	
12	**mqtt.conflag.retain**	**Will Retain**	
13	**mqtt.conflag.uname**	**User Name Flag**	
14	mqtt.conflag.willflag	Will Flag	
15	mqtt.conflags	Connect Flags	
16	**mqtt.dupflags**	**DUP Flag**	
17	mqtt.hdrflags	Header Flags	
18	**mqtt.kalive**	**Keep Alive**	
19	**mqtt.len**	**Message Length**	
20	mqtt.msg	Message	
21	**mqtt.msgid**	**Message Identifier**	
22	**mqtt.msgtype**	**Message Type**	
23	mqtt.proto_len	Protocol Name Length	
24	mqtt.protoname	Protocol Name	
25	mqtt.qos	QoS Level	
26	**mqtt.retain**	**Retain**	
27	mqtt.sub.qos	Requested QoS	
28	**mqtt.suback.qos**	**Granted QoS**	

Çizelge 4.6. MQTTset veri kümesi öznitelik seçimi sonuçları (devam)

29	**mqtt.ver**	**Version**	MQTT
30	**mqtt.willmsg**	**Will Message**	
31	**mqtt.willmsg_len**	**Will Message Length**	
32	mqtt.willtopic	Will Topic	
33	**mqtt.willtopic_len**	**Will Topic Length**	
34	**target**	**Sınıf**	

4.4.3. Veri kümesinin dengelenmesi

Dengesiz veri kümeleri, her sınıfta orantısız bir örnek oranının olduğu ML sınıflandırmasında yaygın bir sorundur. Bu sorun, özellikle iki veya daha fazla sınıfı içeren denetimli makine öğrenimi bağlamında geçerlidir. ML sınıflandırma algoritmalarının çoğu, her sınıf için neredeyse eşit sayıda örnek olduğu hipotezine dayalı olarak tasarlandığından, dengesiz bir veri kümesi, tahmine dayalı modelleme için bir zorluk teşkil etmektedir (Liu vd., 2011). Bu, özellikle düşük performansa sahip hatalı bir tahmin modeliyle sonuçlanır. Bu nedenle, iyi bir tahmin modeli oluşturmak için dengeli bir veri seti gereklidir. Dengesizlik oldukça fazla ise, geleneksel öğrenme algoritmaları ile verimli sınıflandırıcılar geliştirmek zor olmaktadır. MQTTset veri kümesinde sınıflar arasında dengesizlik olduğundan akıllı ev ağından elde edilen veriler eklenerek dengeli hale getirilmiştir.

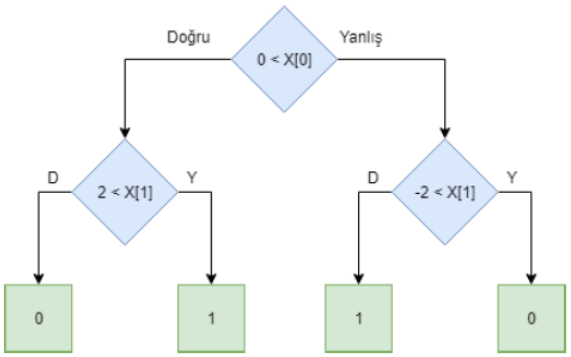
4.5. Makine Öğrenmesi Sınıflandırma Algoritmaları

Saldırı Tespit Sistemlerinde makine öğrenim teknikleri anomali tespiti için yüksek başarı sergilemektedir. Yeni saldırıların daha iyi tespit edilmesi için veri madenciliği kombinasyonuyla birlikte makine öğrenme teknikleri kullanılmaktadır. Genel olarak bir Saldırı Tespit Sisteminin başarısı saldırıyı etkin bir şekilde tespit etme yeteneğine bağlıdır. Saldırı Tespit Sistemleri herhangi bir saldırı durumunda sistem sorumlularını ikaz edebilecek alarmlar üretir.
Herhangi bir önleme faaliyetinde bulunmazlar. Saldırı önleme sistemleri ise saldırıyı tespit etme, izole etme ve engelleme yeteneğine sahiptirler.

Makine öğrenmesi tekniklerinde sistemin performansını etkileyebilecek aşamalardan birisi sınıflandırma işlemleridir. Elde edilen özniteliklere göre her sınıfa ait örnekler belli bir bölgeye konumlandırılır. Sınıflar arasındaki ayrım işlemi her sınıfa ait örneklerin bu konumlarına göre yorumlanmaktadır. Elde edilen bulgulara göre uygun sınıflandırıcı veya birden çok sınıflandırıcılar seçilmektedir (Aytan, 2019). Veri sınıflandırma için kullanılabilecek birçok ML tekniği vardır. Bu çalışmada Karar Ağaçları, Naive Bayes, Rastgele Orman, XGBoost, YSA, MLP, Hoeffding Ağacı ve AdaBoost sınıflandırma yöntemleri kullanılmıştır.

4.5.1. Karar ağacı

Karar ağacı, kökten başlayarak, en uçtaki düğümde bir Boolean sonucuna ulaşana kadar veri bölme dizisinin temsil edildiği ağaç tabanlı bir yöntemdir. Ağaç, dal ve düğümlerden meydana gelen karar ağacında; sınıflandırılacak özellikler bir düğüm ile, bu düğümlerin alabileceği değerler ise dal ile temsil edilmektedir. Ağacın düğümlerinde karar değişkenleri ve yapraklarında tahmin edilen hedef değerler bulunur. Sınıflandırma ve kümeleme problemlerinde sıklıkla tercih edilen karar ağaçları, insan dillerine veya programlama dillerine kolayca çevrilebilmesinden dolayı denetimli öğrenme algoritmalarının en popüleridir (Dey, 2016) (Yang, 2019). Karar ağaçlarını temel alarak geliştirilen birçok algoritma bulunmaktadır. Bu algoritmalar birbirlerinden kök, düğüm ve dallanma kriterine göre farklı kategorilere ayrılmaktadır. Yaygın olarak bilinen karar ağacı algoritmaları ID3, C4.5 ve C5'tir. C4.5, Karar Ağacı oluşturmak için kullanılan verimli ve popüler olarak kullanılan algoritmalardan biridir (Tuncer vd., 2009). Karar ağacı kullanmanın birçok avantajı vardır. Bir problemi anlamak ve yorumlamak kolaydır. Alternatif tekniklerin çoğunda, veriler küçük bir işlemden sonra kullanılabilir hale gelir. Karar ağacının ön işleme aşaması, diğer alternatif yöntemlerden daha kısa ve basittir. Bu yöntem hem sayısal hem de sınıf verilerini işlemek için kullanılabilir. Çoğu makine öğrenmesi algoritması, sayısal uygulamalar veya sınıflandırma problemleri için yararlıdır (Tuncer vd., 2009).

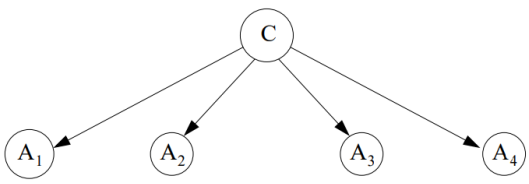


Şekil 4.18: Karar ağacı yapısı örneği (Gülaçar, 2018).

Şekil 4.18'de örnek bir ağaç yapısı gösterilmektedir. Tahmin işlemi sırasında, kök düğümünden başlayarak karar değişkenlerinin girdi kümesindeki karşılıklarına göre alt düğümlere ve sonrasında yapraklara ulaşır. Bir yaprağa ulaştığında gezintisini sonlandırır ve bulunduğu yapraktaki değeri tahmin sonucu olarak döner. Bu çalışmada karar ağaçları ile sınıflandırmada WEKA kullanılmıştır. WEKA'da karar ağacı sınıflandırma "weka.classifiers.trees.J48" içerisinde yer almaktadır.

4.5.2. Naive bayes

Naive Bayes algoritması, bir veri kümesindeki değerlerin frekansını ve kombinasyonlarını sayarak olasılık kümesini hesaplayan basit bir olasılık sınıflandırıcıdır. Naif Bayes sınıflandırma yöntemi, öznitelik değerlerinin verilen hedef değerden koşullu olarak bağımsız olduğu bir varsayıma dayanmaktadır (Saritas vd., 2019). Şekil 4.19'da, Naive Bayes'in bir örneği verilmiştir. Naive Bayes'te, her bir öznitelik düğümünün, sınıf düğümü dışında ebeveyni yoktur.

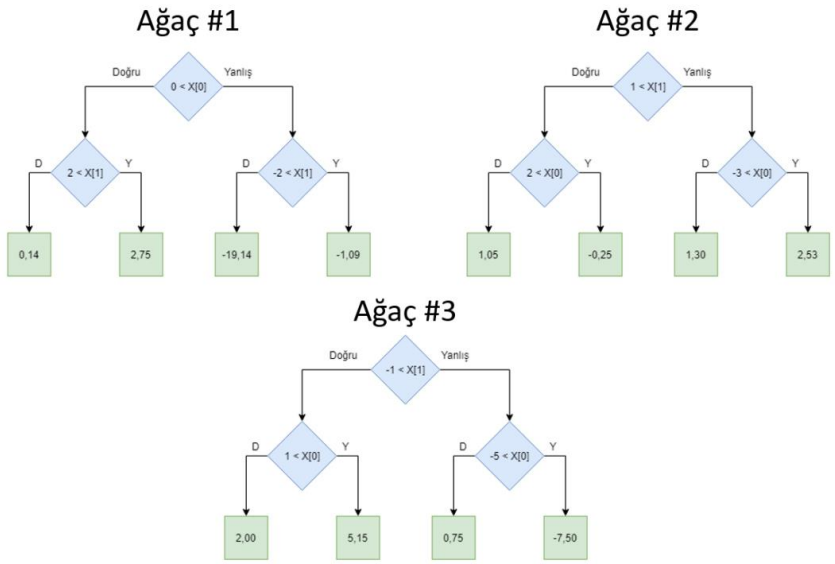


Şekil 4.19: Naive bayes örneği (Zhang, 2004).

4.5.3. Rastgele orman

Rastgele Orman, birbirlerinden bağımsız, aynı dağılım kullanarak eğitim verilerinden rastgele örneklemeye dayanarak karar ağaçlarından oluşmuş bir topluluktur. Rastgele Orman oluşturulmak istenen ağaç sayısınca sınıflandırma ağacının veya hedefine uygun olarak regresyon ağacının topluluklarından oluşmaktadır. Bu nedenle topluluk yöntemlerinden en çok tercih edilen algoritmalarındandır. Yöntem, çok sayıda tahmin edici ağaçlardan rastgele bir alt kümesi seçilerek topluluklar oluşturmaktır (Breiman, 2001). Rastgele seçilmiş alt veri kümeleri üzerinde Karar Ağacı Regresyon yöntemini uygulayarak ağaçlar oluşturur. Tüm ağaçlara birer tahmin değeri üretir. Üretilen tahminlerin ortalamasını alarak kendi tahmin değeri olarak kabul eder.

Bu yöntemde veri kümesi eğitimi esnasında birden fazla karar ağacı oluşturulur ve sonrasında kestirim sırasında karar ağaçlarının sınıflandırılması sonuçları kullanılarak, girdinin sınıfına çoğunluk oyunun belirlenmesi ile karar verilmektedir. Rastgele ormanlarda, karar ağaçlarında olduğu gibi aşırı uyum sorunu bulunmamaktadır (Kalaycı, 2018). Şekil 4.19'da rastgele orman sınıflandırma yönteminde ağaç yapısı örneği verilmiştir. Bu çalışmada rastgele orman sınıflandırma yöntemi için WEKA kullanılmıştır. WEKA'da rastgele orman sınıflandırma "weka.classifiers.trees.RandomForest" içerisinde yer almaktadır.



Şekil 4.20: Rastgele orman sınıflandırma örneği (Gülaçar, 2018).

Örneğin Şekil 4.20'de bulunan 3 ağaçlı orman üzerinde X= [-1, 2] girdi vektörü çalıştırıldığında, algoritmanın üreteceği tahmin değeri -4,75 olacaktır (Gülaçar., 2018). Rastgele Orman, kategorik, sürekli, bazen her ikisinin de yer aldığı veri kümelerinde sıklıkla kullanılmaktadır (Korkmaz vd., 2018).

4.5.4. Yapay sinir ağları

Yapay Sinir Ağları, temelinde biyolojik sinir hücrelerinin bulunduğu, sıklıkla tercih edilen, nöronları yapı taşları olarak kullanan sınıflandırma aracıdır. Burada yapay nöronlar, biyolojik nöronlara benzer olarak tasarlanmıştır. Nöronlara verilmiş girişler, ağırlıklar ile çarpılarak sonrasında yer alan toplama biriminde toplanırlar. Burada bahsedilen toplam ise transfer fonksiyonuna giriş olarak verilmektedir. Transfer fonksiyonundaki çıkış değeri, nöronun çıkış değeri olarak belirlenir. YSA mimari yapısında birçok transfer fonksiyonu kullanılmaktadır. Örneğin; sigmoid fonksiyonu sınıflandırma için kullanılırken, gaussian (normal) fonksiyonu, fonksiyon yakınlaştırma için kullanılabilmektedir. Tek bir nöron normalde sadece doğrusal problemler için kullanılır (Özgür vd., 2012).

YSA, giriş ve çıkış vektörleri arasındaki ilişkileri öğrenen akıllı sistemlerdir. Akıllı olduğu için sistemdeki yeni girişlerin öğrendiği modelle bağını kurabilmektedir. Saldırı Tespit Sistemlerinde YSA'nın sistem davranış izleriyle veya aynı eğilimdeki bir veri seti ile eğitilmesi gerekmektedir. Devamında anormal veya normal olarak belirtilen akışlar YSA modeline verilir. Bu verilerle gerçekleşen öğrenme kullanıcının hareket veya komutunu, bir sonraki hareket veya komutuna yakınsama ile eğitilmesi işlemlerini kapsar (Frank, 1994). Bu çalışmada YSA'yı test edebilmek için WEKA aracı kullanılmıştır. WEKA'da "weka.classifiers.functions.MultilayerPerceptron" seçildikten sonra, sinir ağında gizli katmanlar girilmiştir. Her gizli katman WEKA'da virgülle ayrılarak yazılmaktadır. Vaccari vd. (2020) yapmış oldukları çalışmada gizli katmanlardaki nöron sayısı sırasıyla 50, 30 ve 20 olarak belirlenmiştir. 6 sınıflı olarak yaptıkları çalışmada düğüm sayısını 6 olarak belirlediler. Bu çalışmada başlangıçta 6 düğüm ile testler yapıldı. Veri kümesinin genişletilmesiyle 7 düğümle de test işlemleri gerçekleştirildi.

4.5.5. XGBoost

XGBoost, regresyon ve sınıflandırma görevlerinde kullanılan bir makine öğrenimi tekniğidir. Tipik olarak karar ağaçları olan zayıf tahmin modelleri topluluğu şeklinde bir tahmin modeli verir. Gradyan destekli bir ağaç modeli, diğer artırma yöntemlerinde olduğu gibi aşamalı bir şekilde oluşturulur, ancak keyfi bir türevlenebilir kayıp fonksiyonunun optimizasyonuna izin vererek diğer yöntemleri genelleştirir. Karar ağaçlarının performansını iyileştirmeyi amaçlayan topluluk ailesinin bir üyesidir (Muratlar., 2020).

Input: $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$

M - iterasyon sayısı

p_i - sınıf 1'e ait elemanların oranı

$f_0(x) = \frac{p_i}{1-p_i}$

for $i \leftarrow 1$ **to** M **do**

  $g_i = \frac{dL(y_i, f_m(x_i))}{df_m(x_i)}$

  $h_m(x) = \text{decesion\_tree}(\{(x_1, g_1), (x_2, g_2), ..., (x_n, g_n)\})$

  $v_m = \arg\max_v Q[f_{m-1}(x) + vh_m(x)]$

  $f_m(x) = f_{m-1}(x) + 0.1 v_m h_m(x)$

Return $f_m(x)$

Şekil 4.21: XGBoost sınıflandırma yöntemi (Gülaçar, 2018).

XGBoost sınıflandırma yöntemi, Karar Ağacı sınıflandırma yöntemini kullanarak veri üzerinde Şekil 4.21'de gösterilen algoritmayı uygulayıp tahmin üreten yöntemdir. XGBoost sınıflandırma WEKA'da hazır olarak gelmediği için "weka-xgboost" paketi entegre edilerek test işlemleri gerçekleştirilmiştir.

4.5.6. MLP

MLP, izinsiz giriş tespit sistemlerinin tespit oranını artırmak için yaygın olarak kullanılan makine öğrenimi tekniklerinden biridir. MLP'lerin örüntü sınıflandırma problemleri için eğitim süreci iki görevden oluşur; birincisi probleme uygun mimarinin seçilmesi, ikincisi ise ağın bağlantı ağırlıklarının ayarlanmasıdır. MLP, sistem kullanıcılarının tipik özelliklerini tanımlar ve yerleşik kullanıcı davranışından istatistiksel olarak önemli sapmaları tanımlar. Ayrıca MLP sinir ağı esnek ve genişletilebilir yapılara sahiptir. Bir ortamın davranışına ilişkin genel bir bilgi modeli oluşturabilir. MLP'nin eğitim yoluyla gelen model ve hedef çıktı arasındaki ilişkiyi tanımlamasını sağlamak için nöron parametrelerinin tahmin edilmesini gerektirir (Ghanem vd., 2019). MLP sınıflandırma yöntemi WEKA aracıyla gerçekleştirilmiştir. WEKA'da "weka.classifiers.functions.MultilayerPerceptron" seçilerek sonuçlar görüntülenmiştir.

4.5.7. Hoeffding ağacı

Hoeffding Ağacı, adım adım büyüyen verilerle karar ağacı oluşturmaya yönelik bir algoritmadır. Hoeffding Ağacı, büyük veri akışlarıyla çalışır ve veri dağılımının zaman içinde değişmediğini varsaymaktadır. Hoeffding Ağacı, gerekli doğruluk dahilinde bazı yararlı bilgileri değerlendirmek için gerekli örnek sayısını hesaplayan Hoeffding sınırı ilkesine dayanmaktadır. Bölme için optimal bir özelliğin mevcut olduğuna dair yeterli istatistiksel bilgi toplandıktan sonra, Hoeffding sınırına dayalı olarak ağacın bir düğümü genişletilir. Hoeffding Ağacı, her bir örneği okumak için yalnızca küçük sabit bir zaman aldığından, çevrimiçi veri tabanlarında madencilik yapmak için kullanılmaktadır. Hoeffding Ağaçları, küçük hesaplama yükü olan karmaşık karar ağaçları oluşturmak için kullanılabilir (Lavanya vd., 2017). Bu çalışmada Hoeffding Ağacı ile sınıflandırma işlemleri WEKA'da "weka.classifiers.trees.HoeffdingTree" ile yapılmıştır.

4.5.8. AdaBoost

Boosting, ilk olarak 1997 yılında Freund ve Schapire tarafından sunulan bir topluluk modelleme tekniğidir. Bu algoritmalar, bir dizi zayıf öğreniciyi güçlü öğrenicilere dönüştürerek tahmin gücünü geliştirir. Yükseltme algoritmaları, ilk olarak bir model oluşturur, ardından ilk modelde bulunan hataları düzeltmek için ikinci bir model oluşturmaktadır. Bu prosedür, hatalar en aza indirilinceye ve veri kümesi doğru bir şekilde tahmin edilinceye kadar devam ettirilir. Güçlendirme algoritmaları benzer şekilde çalışmaktadır. Sonuç çıktısına (güçlü öğrenenler) ulaşmak için birden fazla modeli (zayıf öğrenenler) birleştirirler. Temel olarak 3 tür artırma algoritması vardır: AdaBoost algoritması, Gradyan azalma algoritması, Aşırı eğimli azalma algoritması. AdaBoost, makine öğreniminde kullanılan bir tekniktir. AdaBoost algoritması, bir model oluşturarak tüm veri noktalarına eşit ağırlıklar verir. Daha sonra yanlış sınıflandırılmış noktalara daha yüksek ağırlıklar vermektedir. Bir sonraki modelde daha yüksek ağırlıklara sahip tüm noktalara daha fazla önem vermektedir. Bir lowe hatası alana kadar eğitim modellerini tutmaktadır (Saini, 2021). Bu çalışmada AdaBoost sınıflandırma yöntemi "weka.classifiers.meta.AdaBoostM1" ile yapılmıştır.

4.6. Derin Öğrenme

Derin öğrenme, insan beyninde bulunan nöronların yapısından esinlenen yapay sinir ağlarını kullanan makine öğreniminin bir alt alanıdır. YSA olarak da bilinen bu algoritmalar, daha yüksek sayıda gizli katman nedeniyle geleneksel Çok Katmanlı Algılayıcı ile karşılaştırıldığında daha iyi tahmin yeteneklerine sahiptir. Öncelikle YSA'lar, çıktıyı tahmin etmek için aktivasyon fonksiyonlarını kullanarak girdi verilerini işleyen komşu bir katmanla bağlantılı nöronlardan oluşur (Khan vd., 2021). Derin öğrenme metotlarının görüntü tanıma ve işleme problemlerinde oldukça başarılı sonuçlar elde etmektedir. Bunun sebeplerinden bir tanesi yapısının görsel korteksle benzer olmasıdır. Yapıları evrişimsel katmanları, havuzlama katmanlarını ve doğrusal olmayan katmanları içermektedir (Ahmed vd., 2019). DNN yapılarında her katmanda belirli sayıda nöronlar bulmaktadır. Katmanlar kendinden bir önceki katmanın çıkışını girdi olarak alarak çıkışını hesaplamaktadır. Eğitimde DNN'ye belirli bir veri kümesi girdi olarak verilir ve ilk olarak ağırlıklar rastgele olarak ağ yapısı başlatılır. Ancak başka bir yöntem ise daha önceden belirli bir veri kümesi üzerinde eğitilmiş bir ağın üzerinde hedef veri kümesine uygun şekilde ayarlar yaparak kullanmaktır. Bu yöntem transfer öğrenmesi olarak bilinmektedir (Fawaz vd., 2019). Giriş katmanı, verilerin özniteliklerinin ağa giriş olarak verildiği katmandır. Giriş katmanında öznitelik sayısı kadar nöron bulunmaktadır. Gizli katmanda, işlenmemiş veri giriş katmanından alınarak işlenir. Sonraki adımda çıktıyı veren çıkış katmanına gönderilir. Çıkış katmanı ise sonucun üretildiği katmandır (Fawaz vd., 2019).

Sinir ağında katmanlar arasındaki düğümlerin birbirine bağlanması, ileri beslemeli ve tekrarlayan sinir ağı olmak üzere iki temel sınıfa ayrılmaktadır. İleri beslemeli sinir ağlarında, verilen girdilerden çıktılara bilgi hareketi sadece tek yönlü olmaktadır. Tekrarlayan sinir ağlarında durum; bilgilerin bir kısmının ters yönde hareket etmesi ile sonuçlanmaktadır. Uygulama türüne bağlı olarak, çeşitli DNN mimarileri tercih edilmektedir. Bunlar arasında Tekrarlayan Sinir Ağları, Uzun Kısa Süreli Bellek, Evrişimsel Sinir Ağı vb. bulunmaktadır (Selvin vd., 2017).

CNN algoritmaları, görüntü ve ses işleme alanlarında, NLP, biyomedikal gibi birçok alanda uygulamaları bulunmaktadır. Görüntü işleme alanlarındaki çalışmalar sonucunda en iyi sonuçları CNN ile elde etmişlerdir. RNN ise birimler arasındaki bağlantıların yönlendirilmiş bir döngü oluşturduğu yapay sinir ağı sınıfıdır. Bu döngü ile, dinamik zamansal davranış

sergilemesine olanak tanıyan bir ağ iç durumu oluşturmaktadır. RNN'ler görüntü tabanlı verilerde ve NLP çalışmalarında yaygın olarak tercih edilmektedir. Uzun vadeli bağımlılıkları öğrenen özel bir RNN türü olarak bilinen LSTM; 3 kapı, blok girişi, sabit hata döngüsü, çıkış aktivasyon fonksiyonu ve gözetleme bağlantılarından oluşmaktadır. Giriş, unutma ve çıkış kapı bağlantıları bulunmaktadır. Bloktan elde edilen çıktı tekrar bloğun girişine ve tüm kapılara bağlanmaktadır. LSTM de durum sıfırlamak için unutma kapısı, zamanlamaları öğrenmeyi kolay hale getirmek için gözetleme bağlantıları ilk geliştirilen mimariye eklenmiştir (Şeker vd., 2017).

Alaiz-Moreton vd. yapmış oldukları çalışmada LSTM ağları ile sınıflandırma işlemi gerçekleştirdiler. Bu çalışmadan esinlenerek ve sonuçların olumlu yönde etkileneceği düşünülerek bu tez çalışmasında LSTM uygulanmıştır. LSTM uygulamaları,WEKA aracına Deep Learning4j paketi yüklenerek gerçekleştirilmiştir. Bu paket ile MQTTset veri kümesinde 2 katmanlı ve 3 katmanlı DNN testleri yapılmıştır. IoT ev ağından elde edilen veriler ve MQTTset veri kümesi verilerinin birleşiminden elde edilen veri kümesi üzerinde de DNN testleri yapılmıştır. Bu testlerde 5 katmanlı çapraz doğrulama ve LSTM ile sınıfların sonuçları analiz edilmiştir.

4.7. Değerlendirme Metrikleri

Çalışmanın bu bölümünde makine öğrenmesi ve derin öğrenme saldırı tespitindeki başarıların ölçümü için kullanılan değerlendirme ölçütleri hakkında bilgiler verilmektedir. Bölüm 4.4 ve 4.5'te algoritmaların test edilmesi sonrasında oluşturulan konfüzyon matrisinden yararlanılarak doğruluk, duyarlılık, kesinlik ve F1 skor ölçütleri ile performansları değerlendirilmek üzere hesaplanmıştır.

4.7.1. Doğruluk (Accuracy)

Algoritmanın doğru yaptığı sınıflandırmadaki örnek sayısının, toplam örnek sayısına oranı doğruluk değerini vermektedir. Bu değer Denklem 4.1 ile hesaplanmaktadır.

DP $(Doğru\ Pozitif)$: Saldırı sınıfındaki tahmin edilen örnek sayısı,
DN $(Doğru\ Negatif)$: Normal sınıftaki tahmin edilen örnek sayısı,
YP $(Yanlış\ Pozitif)$: Normal sınıftaki saldırı olarak tahmin edilen örnek sayısı,
YN $(Yanlış\ Negatif)$: Saldırı sınıfındaki normal olarak tahmin edilen örnek sayısı,

$$Doğruluk = \frac{DP+DN}{DP+YN+YP+DN} \tag{4.1}$$

4.7.2. Kesinlik (Precision)

Veri kümesinin saldırı sınıfında olup saldırı olarak sınıflandırdığı değerlerin sayısının, saldırı olarak tahmin edilen tüm örneklerin sayısına oranına eşittir. Bu değer Denklem 4.2 ile hesaplanmaktadır.

$$Kesinlik = \frac{DP}{DP+YP} \tag{4.2}$$

4.7.3. Duyarlılık (Recall)

Veri kümesinin saldırı sınıfında bulunan ve saldırı olarak tahmin edilen örneklerin sayısının, saldırı olan tüm örneklere oranı duyarlılık değerine eşittir (Çavuşoğlu vd., 2019). Bu değer Denklem 4.3 ile hesaplanmaktadır.

$$Duyarlılık = \frac{DP}{DP+YN} \tag{4.3}$$

4.7.4. F1 Skoru

Kesinlik ve duyarlılık değerlerinin birlikte kullanılarak hesaplanmaktadır. Bu değer elde edilen kesinlik ve duyarlılık değerlerin harmonik ortalaması alınarak elde edilen F1 Skor değeridir (Çavuşoğlu vd., 2019). Bu değer Denklem 4.4 ile hesaplanmaktadır.

$$F1\ Skor = 2\ x\ (\frac{Kesinlik\ x\ Duyarlılık}{Kesinlik+Duyarlılık}) \tag{4.4}$$

4.7.5. MCC

MCC, dengesiz dağılmış veri kümelerinde bile en gerçekçi sonucun elde edilmesini sağlayan bir metriktir. MCC değerinin 1'e yaklaşması doğru olarak sınıflandırmanın yapıldığını gösterir (Liu vd., 2015). Bu değer Denklem 4.5 ile hesaplanmaktadır.

$$MCC = \frac{(DPxDN)-(YPxYN)}{\sqrt{(DP+YP)(DP+YN)(DN+YP)(DN+YN)}} \tag{4.5}$$

4.7.6. ROC AUC

ROC analizi veri kümesinde gerçekleştirilen sınıflandırma işleminin performansının ölçümünde kullanılan bir analizdir. Burada sınıflandırma işleminin sonucunda, doğru tahminin oranını farklı bir yaklaşım kullanarak incelenmesidir. ROC eğrisi, Duyarlılık ve Özgüllük değerleriyle ifade edilmektedir. ROC eğrisinin, Y ekseninde hesaplanan duyarlılık değeri yer alırken X ekseninde hesaplanan 1- Özgüllük değeri yer almaktadır. Bu değerlerin kesişimleriyle bulunan noktaların birleşimiyle ROC eğrisi elde edilmektedir (Swets, 1979).

ROC eğrisinin altında kalan alan AUC değeridir. AUC değeri Denklem 4.6'daki gibi hesaplanmaktadır (Hossin vd., 2015).

AUC sonuçlarında 1 en iyi değeri, 0,5 ve altındaki değerler ise başarısız bir sınıflandırma yapıldığını göstermektedir.

S_p: Tüm pozitif örneklerin toplamı,

n_p : Pozitif örneklerin sayısı,

n_n : Negatif örneklerin sayısı,

$$AUC = \frac{S_p - n_p\ (n_n + 1)/2}{n_p n_n} \tag{4.6}$$

4.7.7. PRC AUC

PRC eğrisinde duyarlılık ve kesinlik arasındaki ilişki incelenmektedir. Kesim noktalarının tamamında, bu iki değer hesaplanır ve bir eğri elde edilir. Eğri çizilirken X ekseninde elde edilen duyarlılık değeri, Y ekseninde de kesinlik değeri bulunur. Bu değerlerin kesişimiyle bulunan noktaların birleşimiyle PRC eğrisi oluşturulur. Sonucun 1' yakın olması doğru bir tahmin yapıldığını göstermektedir (Çavuşoğlu vd., 2019). PRC AUC değeri, X ve Y arasındaki korelasyonu hesaplayan belirli bir eşiğin ayarlanması durumunda ortalamasını alan bir metriktir (Vujović, 2021).

X ekseninde → Duyarlılık

Y ekseninde → Kesinlik

5. BULGULAR VE TARTIŞMA

Çalışma sonuçları iki farklı bölümde sunulmuştur. İlk bölümde; öncelikle MQTTset veri kümesi üzerinde makine öğrenmesi yöntemleri uygulanmıştır. Daha sonra MQTTset ve IoT ev ağından elde veriler üzerinde makine öğrenmesi yöntemleri uygulanarak sonuçlar karşılaştırılmıştır. Elde edilen değerler tablolar ve grafikler ile sunulmuştur.

İkinci bölümde; MQTTset veri kümesi ile derin öğrenme uygulamaları gerçekleştirilmiştir. Makine öğrenmesi yöntemlerine göre değerlerde artış gözlendiği için MQTTset ve IoT ev ağından elde edilen verilerle de derin öğrenme uygulamaları yapılmış ve sonuçlar tablolar ile karşılaştırılmıştır.

5.1. Makine Öğrenmesi ile Sınıflandırma

Bu bölümde makine öğrenmesi ile çok sınıflı ve iki sınıflı sınıflandırma sonuçlarının karşılaştırması yapılmıştır. Makine öğrenmesi sınıflandırma işlemlerinde 5 kat çapraz doğrulama ile hem eğitim hem de test kümesinin seçim işlemi gerçekleştirilmiştir.

5.1.1. Çok sınıflı sınıflandırma uygulama sonuçları

Bölüm 4.4'te sunulan makine öğrenmesi sınıflandırma yöntemleri ile MQTTset veri kümesi çok sınıflı sınıflandırma ile test edilmiştir. Çizelge 5.1'de MQTTset veri kümesi ile çok sınıflı sınıflandırma sonuçları sunulmuştur. Vaccari vd. (2020) tarafından yapılan çalışmada HT ve AdaBoost sınıflandırma bulunmamaktadır. Bu çalışma kapsamında HT ve AdaBoost algoritmaları ile sınıflandırma eklenmiştir. Diğer algoritmaların sınıflandırma sonuçları ile karşılaştırıldığında sonuçların daha yüksek olduğu görülmüştür.

Çizelge 5.1. MQTTset çok sınıflı sınıflandırma sonuçları

Algoritmalar	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skor**	**MCC**	**ROC AUC**	**PRC AUC**
Rastgele Orman	0.915	0.937	0.935	0.914	0.908	0.901	0.923
Naive Bayes	0.643	0.786	0.669	0.687	0.683	0.803	0.855
Karar Ağaçları	0.915	0.891	0.869	0.914	0.784	0.921	0.906
XGBoost	0.879	0.855	0.870	0.872	0.821	0.835	0.842
MLP	0.903	0.880	0.848	0.842	0.759	0.912	0.921
YSA	0.904	0.852	0.840	0.902	0.810	0.823	0.827
Hoeffding Ağacı	0.916	0.927	0.917	0.921	0.887	0.907	0.933
AdaBoost	0.921	0.932	0.920	0.917	0.925	0.918	0.938

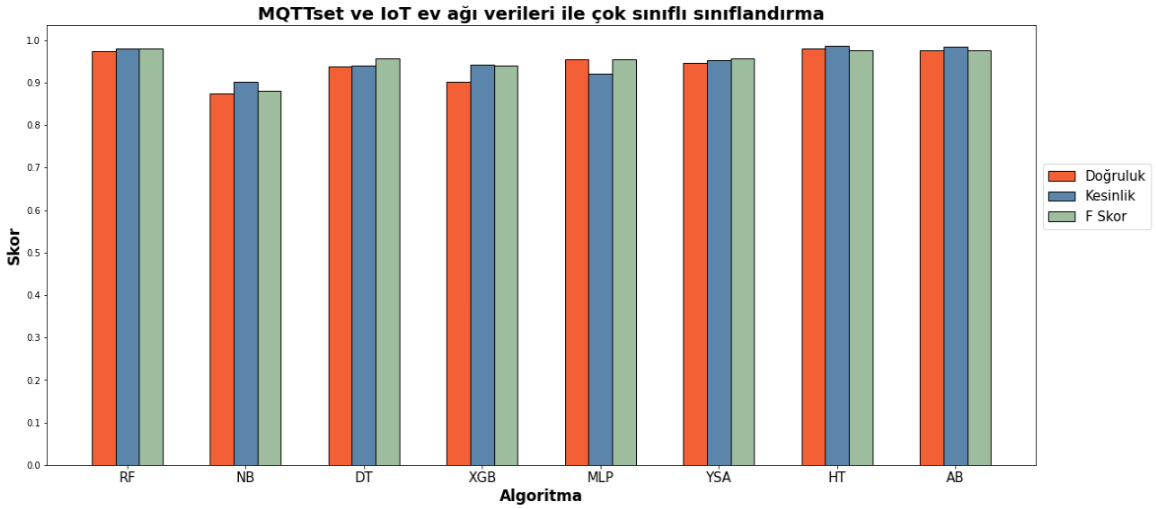
Çizelge 5.2'de MQTTset ve IoT ev ağı verileri ile çok sınıflı sınıflandırma sonuçları verilmiştir. Ev ağından elde edilen veriler ve yeni eklenen botnet saldırıları ile sonuçlarda artış gözlenmiştir. Çizelge 5.1 ve Çizelge 5.2 karşılaştırıldığında veri kümesindeki yapılan genişletme ve iyileştirmelerin olumlu yönde olduğu görülmüştür.

Çizelge 5.2. MQTTset ve IoT ev ağı verileri ile çok sınıflı sınıflandırma sonuçları

Algoritmalar	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skor**	**MCC**	**ROC AUC**	**PRC AUC**
Rastgele Orman	0.975	0.980	0.977	0.980	0.977	0.998	0.993
Naive Bayes	0.875	0.902	0.878	0.880	0.861	0.985	0.982
Karar Ağaçları	0.939	0.940	0.938	0.958	0.928	0.987	0.971
XGBoost	0.901	0.942	0.935	0.940	0.895	0.988	0.981
MLP	0.955	0.920	0.920	0.955	0.948	0.990	0.989
YSA	0.946	0.952	0.940	0.956	0.965	0.985	0.979
Hoeffding Ağacı	0.980	0.987	0.977	0.977	0.973	0.998	0.991
AdaBoost	0.976	0.985	0.986	0.976	0.972	0.998	0.991

Şekil 5.1'de MQTTset ve IoT ev ağı verileri ile çok sınıflı sınıflandırma sonuçları verilmiştir. Grafikte X ekseninde uygulanan sınıflandırma algoritmaları ve Y ekseninde sonuç değerleri bulunmaktadır.

Şekil 5.1'de görüldüğü gibi en yüksek sonuçları RF, HT ve AdaBoost sınıflandırma algoritmaları vermektedir. Diğer sınıflandırma algoritmalarının sonuçlarında iyileşmeler olduğu görülmüştür.



Şekil 5.1: Çok sınıflı sınıflandırma doğruluk, kesinlik ve F skor değerleri

5.1.2. İki sınıflı sınıflandırma uygulama sonuçları

Çok sınıflı sınıflandırma bazı veri kümelerinde iyi sonuçlar verirken bazı veri kümelerinde ise iki sınıflı sınıflandırma daha yüksek sonuçlar vermektedir. Bu çalışma kapsamında veri kümelerinin iki sınıflı hale getirilmesi sağlanmış ve çok sınıflı sınıflandırma da uygulanan tüm sınıflandırma algoritmaları test edilmiştir. Bölüm 4.4'te sınıflandırma algoritmaları ile ilgili detaylı bilgiler yer almaktadır. Çok sınıflı yapıdan iki sınıflı yapıya dönüşüm işlemleri WEKA ile gerçekleştirilmiştir. İki sınıflı sınıflandırmada veri kümesinde target bilgisi saldırı ve normal olarak tüm verilerde dönüşüm sağlanmıştır. Çizelge 5.3'te MQTTset veri kümesi ile gerçekleştirilmiş iki sınıflı sınıflandırma sonuçları verilmiştir. Çizelge 5.1 ve Çizelge 5.3 karşılaştırıldığında sonuçlarda artış görülmüştür.

Çizelge 5.3. MQTTset iki sınıflı sınıflandırma sonuçları

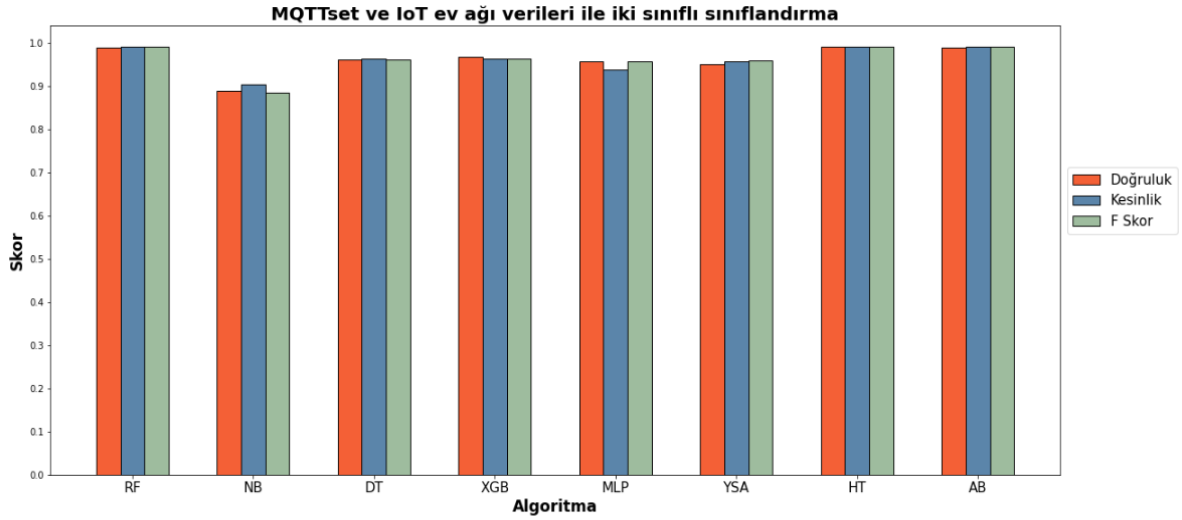
Algoritmalar	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skor**	**MCC**	**ROC AUC**	**PRC AUC**
Rastgele Orman	0.961	0.952	0.963	0.961	0.923	0.985	0.989
Naive Bayes	0.811	0.863	0.811	0.804	0.672	0.920	0.898
Karar Ağaçları	0.860	0.890	0.860	0.857	0.749	0.948	0.922
XGBoost	0.821	0.868	0.865	0.852	0.765	0.920	0.932
MLP	0.848	0.883	0.844	0.730	0.736	0.926	0.932
YSA	0.855	0.823	0.845	0.850	0.890	0.823	0.852
Hoeffding Ağacı	0.860	0.890	0.860	0.857	0.749	0.948	0.922
AdaBoost	0.868	0.895	0.875	0.856	0.878	0.892	0.898

İkili sınıflandırma yönteminin daha yüksek sonuçlar verdiği test işlemleri sonucunda görüldüğü için MQTTset ve IoT ev ağı verileri için de aynı dönüşüm işlemleri gerçekleştirilmiştir. Çizelge 5.4'te MQTTset ve IoT ev ağı verileri ile iki sınıflı sınıflandırma sonuçları sunulmuştur. Çizelge 5.2 ile Çizelge 5.4 karşılaştırıldığında ikili sınıflandırma sonuçlarının daha yüksek olduğu görülmektedir.

Çizelge 5.4. MQTTset ve IoT ev ağı verileri ile iki sınıflı sınıflandırma sonuçları

Algoritmalar	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skor**	**MCC**	**ROC AUC**	**PRC AUC**
Rastgele Orman	0.990	0.991	0.992	0.992	0.980	0.999	0.994
Naive Bayes	0.889	0.904	0.898	0.886	0.875	0.986	0.983
Karar Ağaçları	0.962	0.965	0.962	0.962	0.959	0.990	0.990
XGBoost	0.968	0.965	0.967	0.965	0.968	0.991	0.988
MLP	0.958	0.938	0.925	0.958	0.950	0.992	0.990
YSA	0.952	0.958	0.945	0.959	0.970	0.987	0.985
Hoeffding Ağacı	0.991	0.992	0.993	0.992	0.979	0.999	0.999
AdaBoost	0.990	0.992	0.992	0.992	0.980	0.999	0.998

Şekil 5.2'de iki sınıflı sınıflandırma doğruluk, kesinlik ve F skor değerlerinin grafiksel gösterimi sağlanmıştır. En yüksek değerlerin RF, HT ve AdaBoost sınıflandırma algoritmalarının verdiği görülmektedir.



Şekil 5.2: İki sınıflı sınıflandırma doğruluk, kesinlik ve F skor değerleri

5.2. Derin Öğrenme ile Sınıflandırma

Derin öğrenme ile sınıflandırma işlemlerinde 5 kat çapraz doğrulama ile hem eğitim hem de test kümesinin seçim işlemi gerçekleştirilmiştir. Derin öğrenme testleri WEKA ile Deep Learning4J paketi yüklenerek yapılmıştır. DL4J modellerinin yapısı Dinler vd. (2021) tarafından detaylı şekilde açıklanmıştır. Yaptıkları testler sonucunda GravesLSTM 2-Layer ve LSTM & GravesLSTM 3-Layer'ın daha yüksek sonuçlar verdiği görülmektedir. Buradan yola çıkarak Çizelge 5.5'te verilen 2 ve 3 katman bilgileriyle test işlemleri gerçekleştirilmiştir.

Çizelge 5.5. Derin öğrenme katman bilgileri

<table>
<tr><th>Katmanlar</th><th>Katman Detayları</th><th>Epochs</th><th>Ağ Yapılandırma</th></tr>
<tr><td rowspan="2">GravesLSTM
2-Layer</td><td>GravesLSTM Layer</td><td rowspan="2">10</td><td rowspan="4">Optimizasyon algoritması:
Stochastic_Gradient_Descent,
Updater: Adam,
biasUpdater: Sgd,
weight initialization method:
XAVIER,
zooModel: CustomNet</td></tr>
<tr><td>Output Layer</td></tr>
<tr><td rowspan="3">LSTM &
GravesLSTM
3-Layer</td><td>LSTM Layer</td><td rowspan="3">10</td></tr>
<tr><td>GravesLSTM Layer</td></tr>
<tr><td>Output Layer</td></tr>
</table>

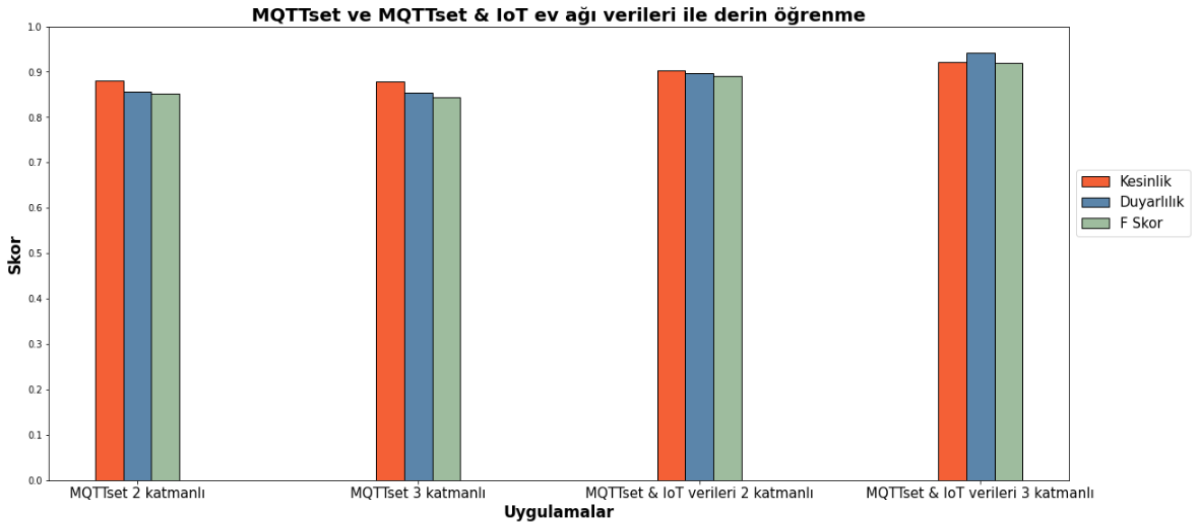
DL4J ile derin öğrenmede epochs değeri 10, ağ yapılandırma ayarları Çizelge 5.5'te verildiği gibi uygulanmıştır. İlk olarak MQTTset veri kümesi ile 2 ve 3 katman ile derin öğrenme işlemleri gerçekleştirilmiştir. İkinci olarak MQTTset ve IoT ev ağı verileri ile Çizelge 5.5'te verildiği gibi 2 ve 3 katmanda testler yapılmıştır.

Derin öğrenme 2 katmanla yapılan uygulamalarda ilk katman GravesLSTM, ikinci katman Output Layer olarak seçilmiştir. MQTTset veri kümesi ile yapılan uygulamalarda iki katmanında çıktı sayıları 6 olarak seçilmiştir. MQTTset ve IoT ev ağı verileri ile yapılan uygulamalarda ilk ve ikinci katmanda çıktı sayıları 7 olarak ayarlanmıştır. MQTTset veri kümesinde 6 farklı sınıf yer alırken diğerinde (MQTTset ve IoT ev ağı verileri) 7 farklı sınıf yer almaktadır. Bu nedenle çıktı sayıları bahsedildiği gibi seçilmiştir. Çizelge 5.6'da derin öğrenme uygulamalarının sonuçları sunulmuştur. MQTTset ve IoT ev ağı verileri ile yapılan uygulama sonuçlarının, MQTTset veri kümesi ile yapılan uygulamalardan daha iyi sonuçlar verdiği görülmüştür.

Çizelge 5.6. MQTTset ve IoT ev ağı verileri ile derin öğrenme sonuçları

Algoritmalar	**Kesinlik**	**Duyarlılık**	**F1 Skor**	**MCC**	**ROC AUC**	**PRC AUC**
MQTTset 2 katmanlı	0.880	0.857	0.851	0.767	0.962	0.940
MQTTset 3 katmanlı	0.878	0.854	0.844	0.761	0.944	0.918
MQTTset ve IoT ev ağı verileri ile 2 katmanlı	0.903	0.896	0.890	0.896	0.974	0.976
MQTTset ve IoT ev ağı verileri ile 3 katmanlı	0.920	0.941	0.920	0.947	0.981	0.987

Şekil 5.3'te derin öğrenme ile yapılan tüm uygulamaların sonuçları grafik ile sunulmuştur.



Şekil 5.3: Derin öğrenme kesinlik, duyarlılık ve F skor değerleri

6. SONUÇ VE ÖNERİLER

Bu tez çalışmasında IoT'de sıklıkla kullanılan MQTT protokolü ile haberleşen IoT ev ağı prototipi oluşturulmuştur. IoT ev ağında farklı ağ bağlantısı olan sensörlerden sıcaklık, nem, uzaklık bilgileri alınarak kaydedilmiştir. Bu ağa Botnet, DoS, Kaba Kuvvet ve SlowDoS saldırıları gerçekleştirilmiş ve elde edilen veriler MQTTset veri kümesine eklenmiştir. Ayrıca zararlı, bozulmuş ve meşru veriler de entegre edilmiştir. MQTTset veri kümesine Botnet saldırıları eklenmiş ve diğer saldırılara da veriler eklenerek genişletilmesi sağlanmıştır. MQTTset veri kümesi üzerinde veri ön işleme ve öznitelik seçimi işlemleri gerçekleştirilmiştir. MQTTset veri kümesi 21 özniteliğe indirgenmiştir. MQTTset veri kümesi 5 kat çapraz doğrulama kullanılarak makine öğrenmesi ve derin öğrenme algoritmaları ile değerlendirilmiştir.

MQTTset veri kümesi üzerinde çok sınıflı ve iki sınıflı sınıflandırma ile ayrı ayrı testler yapılmıştır. Çok sınıflı sınıflandırmada en yüksek doğruluk değerini AdaBoost, en yüksek kesinlik değerini RF, en yüksek duyarlılık değerini RF, en yüksek F1 Skor değerini HT algoritmaları vermiştir. İki sınıflı sınıflandırmada en yüksek doğruluk, kesinlik, duyarlılık ve F1 Skor değerlerini RF algoritması vermiştir.

MQTTset ve IoT ev ağı verileri de kendi içerisinde çok sınıflı ve iki sınıflı sınıflandırma sonuçlarına göre değerlendirilmiştir. MQTTset ve IoT ev ağı verileri ile çok sınıflı sınıflandırmada en yüksek doğruluk değerlerini sırasıyla HT, AdaBoost, RF algoritmaları vermiştir. En yüksek kesinlik HT, duyarlılık AdaBoost ve F1 Skor değerini RF algoritmalarının verdiği görülmüştür. MQTTset ve IoT ev ağı verileri ile iki sınıflı sınıflandırmada en yüksek doğruluk değerlerini RF ve AdaBoost, kesinlik değerlerini HT ve AdaBoost, duyarlılık değerini HT, F1 Skor değerlerini ise RF, HT, AdaBoost algoritmaları vermektedir.

Derin öğrenme yöntemlerinde 2 ve 3 katmanlı uygulamalarla test edilmiştir. MQTTset ve IoT ev ağından elde edilen verilerin birleşimiyle de aynı yöntemler ve algoritmalar kullanılarak test ve analiz işlemleri gerçekleştirilmiştir. MQTTset veri kümesinde 3 katmanla yapılan derin öğrenme sonuçlarında sadece kesinlik değerinin 2 katmanla yapılandan daha yüksek olduğu görülmüştür. MQTTset veri kümesinde genel olarak sonuçlara bakıldığında 2 katmanla yapılan derin öğrenme uygulamaları daha yüksek sonuçlar vermiştir. MQTTset ve IoT

ev ağı verileri ile derin öğrenme uygulamalarında en yüksek değerler 3 katmanla yapılan testlerden elde edilmiştir. Derin öğrenme çerçevesinde MQTTset ve IoT ev ağı verilerinin daha yüksek sonuçlar verdiği testler ve analizler sonucunda tespit edilmiştir. Ayrıca bu tez çalışmasında katmanlara göre saldırıların sınıflandırılmasında literatür araştırması sonucunda tablolar oluşturulmuştur.

Gelecek çalışmalarda IoT'de kullanılan diğer protokoller için makine öğrenmesi ve derin öğrenme yöntemleri bu çalışmada olduğu gibi test edilip protokol bazında karşılaştırmalar yapılabilir. Bu çalışma kapsamında uygulanan makine öğrenmesi algoritmaları dışındaki algoritmaların da test edilmesi sağlanabilir. İlerleyen çalışmalarda, IoT ağlarda kullanılan iletişim protokollerine karşı gerçekleştirilen saldırıların çeşitlerinin tümü değerlendirilerek, yeni ve daha kapsamlı bir veri kümesi oluşturulması ve bu alanda yapılan çalışmalar için kullanıma açılması hedeflenmektedir.

KAYNAKLAR DİZİNİ

Abdelzaher, T. R., Prabh, S., Kiran, R., 2004, On real-time capacity limits of multihop wireless sensor networks, In 25th IEEE International Real-Time Systems Symposium, p. 359-370.

Adıgüzel, Ö. Y., 2016, Endüstri 4.0 ve nesnelerin interneti, Lisans tezi, Karabük Üniversitesi Teknoloji Fakültesi Mekatronik Mühendisliği Bölümü, s. 7-77.

Ahmed, O., Brifcani, A., 2019, Gene expression classification based on deep learning, In 2019 4th Scientific International Conference Najaf, p.45-149.

Akgul, F., 2021, https://maker.robotistan.com/ultrasonic-sensor/, erişim tarihi: 1.05.2022.

Alaiz-Moreton, H., Aveleira-Mata, J., Ondicol-Garcia, J., Muñoz-Castañeda, A. L., García, I., Benavides, C., 2019, Multiclass classification procedure for detecting attacks on MQTT-IoT protocol.

Anderson, J. P., 1980, Computer security threat monitoring and surveillance, Technical Report, James P. Anderson Company.

Arıs, A., Oktug, S. F., Yalçın, S. B. Ö., 2015, Nesnelerin interneti güvenliği: Servis engelleme saldırıları, In 23th Signal Processing and Communications Applications Conference (SIU).

Aytan, B., 2019, Siber savunma alanında yapay zeka tabanlı saldırı tespiti ve analizi, Yüksek lisans tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, s. 1-74.

Aytan, B., Barışçı, N, 2018, Siber savunma alanında yapay zekâ tabanlı saldırı tespiti ve analizi, In Proceeding of the 2nd International Symposium on Innovative Approaches in Scientific Studies.

Bace, R. G., Mell, P., 2011, NIST special publication on intrusion detection systems, Publications of National Institute of Standards and Technology, p.1-53.

Balcı, Y., 2021, Nesnelerin interneti ekosisteminde yapay zeka destekli saldırı tespit sistemi, Yüksek lisans tezi, Milli Savunma Üniversitesi Hezarfen Havacılık ve Uzay Teknolojileri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, s. 1-66.

Bertino, E., Islam, N., 2017, Botnets and internet of things security, p.76-79.

KAYNAKLAR DİZİNİ (devam)

Bhushan, B., Sahoo, G., Rai, A. K., 2017, Man-in-the-middle attack in wireless and computer networking, A review, In 2017 3rd International Conference on Advances in Computing, Communication & Automation, p.1-6.

Bossi L., Bertino, E., Hussain, S. R., 2017, A system for profiling and monitoring database access patterns by application programs for anomaly detection, IEEE Transactions on Software Engineering, p.415-431.

Bozdoğan, Z., 2015, Nesnelerin interneti için tasarım mimarisi, Yüksek lisans tezi, Düzce Üniversitesi Fen Bilimleri Enstitüsü, s. 1-37.

Breiman, L., 2001, Random forests. Machine learning, 45, 1, p. 5-32.

Burukanlı, M., Çıbuk, M., Budak, Ü., 2021, Saldırı tespiti için makine öğrenme yöntemlerinin karşılaştırmalı analizi, Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, s. 613-624.

Canedo, J., Skjellum, A., 2016, Using machine learning to secure IoT systems, In 2016 14th annual conference on privacy, security and trust (PST), p. 219-222.

Carli, F., Boschi, G., Petri, P., 2020, https://github.com/gianluca2414/MQTT_SlowITe, erişim tarihi: 17.01.2022.

Chaddha, S., 2018, https://github.com/sahilchaddha/rudyjs, erişim tarihi: 13.09.2021.

Ciklabakkal, E., Donmez, A., Erdemir, M., Suren, E., Yilmaz, M. K., Angin, P., 2019, ARTEMIS: An intrusion detection system for MQTT attacks in internet of things, In 2019 38th Symposium on Reliable Distributed Systems (SRDS), p. 369-3692.

Çakır, B., 2019, Zero-day attack detection with deep learning, Master of science, Middle East Technical University The Graduate School of Natural and Applied Sciences Computer Engineering, p. 1-95.

Çakıroğlu, M., 2008, Kablosuz algılayıcı ağlar için dinamik kanal atlamalı güvenlik sistemi tasarımı, Doktora tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı, s. 56-73.

Çavuşoğlu, Ü., Kaçar, S., 2019, Anormal trafik tespiti için veri madenciliği algoritmalarının performans analizi, Academic Platform Journal of Engineering and Science, 7, 2, s. 205-216.

KAYNAKLAR DİZİNİ (devam)

Dal, F., 2019, Nesnelerin interneti sistemlerine yapılan saldırıların analizi üzerine tuzak sistemler ile bir durum çalışması, Yüksek lisans tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü Adli Bilimler Anabilim Dalı, s. 27-35.

Dey, A., 2016, Machine learning algorithms, A review, International Journal of Computer Science and Information Technologies, p. 1174-1179.

Dinler, Ö. B., Şahin, C. B., Abualigah, L., 2021, Comparison of performance of phishing web sites with different DeepLearning4J models, Avrupa Bilim ve Teknoloji Dergisi, p. 425-431.

Ebleme, M. A., 2019, Nesnelerin interneti uygulama katmanı haberleşme protokollerinin başarım analizi, Yüksek lisans tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı, s. 7-14.

Ekhlayef, O. K., 2020, Intrusion detection and classification in internet of things networks using machine learning, Yüksek lisans tezi, Altınbaş University Institute of Graduate Studies Information Technologies, p. 2-6.

Erzi, H., 2021, Nesnelerin interneti teknolojisi ile akıllı ev kontrolü için uygulama geliştirilmesi, Yüksek lisans tezi, İnönü Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, s. 11-17.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., Muller, P. A., 2019, Deep learning for time series classification, A review, Data mining and knowledge discovery, p. 917-963.

Fezari, M., Dahoud, A., 2018, Integrated Development Environment “IDE” for Arduino, WSN applications, p. 1-12.

Firdous, S. N., Baig, Z., Valli, C., Ibrahim, A., 2017, Modelling and evaluation of malicious attacks against the iot mqtt protocol, In 2017 IEEE International Conference on Internet of Things and IEEE Green Computing and Communications and IEEE Cyber, Physical and Social Computing and IEEE Smart Data, p. 748-755.

Frank, J., 1994, Artificial intelligence and intrusion detection: current and future directions, Division of Computer Science University of California at Davis, p. 1-12.

Gagandeep., Aashima., 2012, Study on sinkhole attacks in wireless ad hoc networks, International Journal on Computer Science and Engineering, 4, p. 1078-1084.

KAYNAKLAR DİZİNİ (devam)

Gao, Z., Cao, J., Wang, W., Zhang, H., Xu, Z., 2021, Online-semisupervised neural anomaly detector to identify MQTT-based attacks in real time, Security and Communication Networks.

Ge, M., Fu, X., Syed, N., Baig, Z., Teo, G., Robles-Kelly, A., 2019, Deep learning-based intrusion detection for IoT networks. In 2019 IEEE 24th pacific rim international symposium on dependable computing, p. 256-265.

Ghanem, W. A., Jantan, A., 2019, A new approach for intrusion detection system based on training multilayer perceptron by using enhanced Bat algorithm, Neural Computing and Applications, p. 1-34.

Ghazanfar, S., Hussain, F., Rehman, A. U., Fayyaz, U. U., Shahzad, F., Shah, G. A., 2020, Iot-flock: An open-source framework for iot traffic generation, In 2020 International Conference on Emerging Trends in Smart Technologies (ICETST), p. 1-6.

Gül, M., Aydın, M. A., 2016, Exploits in information security, In 2016 24th Signal Processing and Communication Application Conference, p. 1729-1732.

Gülaçar, H., 2018, Nesnelerin interneti platformları için makine öğrenmesi tabanlı bir tahmin modülü, Yüksek lisans tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, s. 18-26.

Halimaa, A., Sundarakantham, K., 2019, Machine learning based intrusion detection system, International conference on trends in electronics and informatics, p. 916-920.

Hartigan, J. A. Wong M. A., 1979, Algorithm AS 136: A k-means clustering algorithm, Journal of the Royal Statistical Society, 28, p. 100-108.

Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P., Sikdar, B., 2019, A survey on IoT security: Application areas, security threats and solution architectures, IEEE Access, 7, p. 82721-82743.

Hillar, G. C., 2017, MQTT essentials-a lightweight IoT protocol, Packt Publishing Ltd.

Hindy, H., Bayne, E., Bures, M., Atkinson, R., Tachtatzis, C., Bellekens, X., 2020, Machine learning based IoT Intrusion Detection System: an MQTT case study (MQTT-IoT-IDS2020 Dataset), In International Networking Conference, p. 73-84.

Hossin, M., Sulaiman, M. N., 2015, A review on evaluation metrics for data classification evaluations, International journal of data mining & knowledge management process, 5, p. 1-11.

KAYNAKLAR DİZİNİ (devam)

Kahraman, H., 2018, http://www.arduinorobotik.com/Makale/MakaleDetay/36-nodemcu-ders-1-arduino-nodemcu-esp8266-12e-kurulumu, erişim tarihi: 1.05.2022.

Kalaycı, T. E., 2018, Kimlik hırsızı web sitelerinin sınıflandırılması için makine öğrenmesi yöntemlerinin karşılaştırılması, Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 24, 5, p. 870-878.

Kara, İ., 2019, Detection, technical analysis of brute force attack, p. 1-9.

Karlof, C., Wagner, D., 2003, Secure routing in wireless sensor networks attacks and countermeasures, Ad Hoc Networks, 1,2, p. 293-315.

Kaşif, A., 2020, Ev otomasyon sistemleri için güvenlik ve gizlilik odaklı ağ geçidi tasarımı, Yüksek lisans tezi, Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, s. 12-19.

Kaya, Ç., Yıldız, O., Ay, S., 2016, Performance analysis of machine learning techniques in intrusion detection, Signal Processing and Communication Application Conference, p. 1473-1476.

Kaynar, O., Arslan, H., Görmez, Y., Işık, Y. E., 2018, Makine öğrenmesi ve öznitelik seçim yöntemleriyle saldırı tespiti, Bilişim Teknolojileri Dergisi, 11, 2, s. 175-185.

Khalil, E. A., Özdemir, S., 2018, Nesnelerin internetine genel bir bakış: Kavram, özellikler, zorluklar ve fırsatlar, Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 24, 2, s. 311-326.

Khan, M. A., Jan, S. U., Ahmad, J., Jamal, S. S., Shah, A. A., Buchanan, W. J., 2021, A deep learning-based Intrusion Detection System for MQTT enabled IoT sensors, p. 7016.

Kiran, K. S., Devisetty, R. K., Kalyan, N. P., Mukundini, K., Karthi, R., 2020, Building a intrusion detection system for iot environment using machine learning techniques, Procedia Computer Science, 171, p. 2372-2379.

Kolias, C., Kambourakis, G., Stavrou, A., Voas, J., 2017, DDoS in the IoT: Mirai and other botnets, p. 80-84.

Korkmaz, D., Çelik, H. E., Kapar, M., 2018, Sınıflandırma ve regresyon ağaçları ile rastgele orman algoritması kullanarak botnet tespiti, Van Yüzüncü Yıl Üniversitesi Fen Bilimleri Enstitüsü Dergisi, s. 297-307.

KAYNAKLAR DİZİNİ (devam)

Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B., 2019, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset, Future Generation Computer Systems, p. 779-796.

Kumova, S., Karaoğlan, B., Kışla, T., 2016, Attribute value-range detection in identification of paraphrase sentence pairs, In 2016 24th Signal Processing and Communication Application Conference, p. 1393-1396.

Lang, S., Bravo-Marquez, F., Beckham, C., Hall, M., Frank, E., 2019, Wekadeeplearning4j: A deep learning package for weka based on deeplearning4j, Knowledge-Based Systems, 178, p. 48-50.

Lavanya, K., Bajaj, S., Tank, P., Jain, S., 2017, Handwritten digit recognition using hoeffding tree, decision tree and random forests, A comparative approach, In 2017 International Conference on Computational Intelligence in Data Science, p. 1-6.

Liao H., Lin C., Lin Y. Tung K., 2013, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications, 36, 1, p. 16-24.

Liu, Y., Cheng, J., Yan, C., Wu, X., Chen, F., 2015, Research on the Matthews Correlation Coefficients Metrics of Personalized Recommendation Algorithm Evaluation, International Journal of Hybrid Information Technology, 8, p. 163-172.

Liu, Y., Yu, X., Huang, J. X., An, A., 2011, Combining integrated sampling with SVM ensembles for learning from imbalanced datasets, Information Processing & Management, p. 617-631.

Mahfouz, A., Abuhussein, A., Venugopal, D., Shiva, S., 2020, Ensemble classifiers for network Intrusion Detection using a novel network attack dataset, Future Internet, 12,11, p. 180.

Mann, P., Tyagi, N., Gautam, S., Rana, A., 2020, Classification of various types of attacks in IoT environment, In 2020 12th International Conference on Computational Intelligence and Communication Networks, p. 346-350.

Manzoor, I., Kumar, N., 2017, A feature reduced intrusion detection system using ANN classifier, Expert Systems with Applications, 88, p. 249-257.

Mishra, B., 2018, TMCAS: An MQTT based collision avoidance system for railway networks, 18th International Conference on Computational Science and Applications, p. 1-6.

KAYNAKLAR DİZİNİ (devam)

Moustafa, N., 2019, New generations of Internet of Things datasets for cybersecurity applications based machine learning: TON_IoT datasets, In Proceedings of the Research Australasia Conference, p. 21-25.

Mukherjee, B., Heberlein, L. T., Levitt, K. N., 1994, Network intrusion detection, IEEE network, 8, 3, p. 26-41.

Muratlar, E. R., 2020, https://www.veribilimiokulu.com/gradient-boosted-regresyon-agaclari/, erişim tarihi: 3.12.2021.

Nair, P., Kashyap, I., 2019, Hybrid pre-processing technique for handling imbalanced data and detecting outliers for KNN classifier, In 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing, p. 460-464.

Oliveira, D. L., Veloso, A. F. D. S., Sobral, J. V., Rabêlo, R. A., Rodrigues, J. J., Solic, P., 2019, Performance evaluation of MQTT brokers in the Internet of Things for smart cities, In 2019 4th International Conference on Smart and Sustainable Technologies, p. 1-6.

Ordin, B., 2010, Nonsmooth optimization algorithm for semi-supervised data classification, Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications & Algorithms, 17, p. 741-749.

Özgür, A., Erdem, H., 2012, Saldırı tespit sistemlerinde kullanılan kolay erişilen makine öğrenme algoritmalarının karşılaştırılması, Bilişim Teknolojileri Dergisi, s. 41-48.

Palmieri, A., Prem, P., Ranise, S., Morelli, U., Ahmad, T., 2019, MQTTSA: A tool for automatically assisting the secure deployments of MQTT brokers, In 2019 IEEE World Congress on Services, p. 47-53.

Pan, B., 2018, Application of XGBoost algorithm in hourly PM2, 5 concentration prediction, In IOP conference series: earth and environmental science, 113, p. 012127.

Parihar, Y. S., 2019, Internet of Things and Nodemcu, Journal of Emerging Technologies and Innovative Research, p. 1085.

Ramzai, J., 2020, https://towardsdatascience.com/clearly-explained-how-machine-learning-differs-from-statistical-modeling-967f2c5a9cfd, erişim tarihi: 01.09.2021.

Raymond, D. R., Midkiff, S. F., 2008, Denial-of-service in wireless sensor networks: Attacks and defenses, IEEE Pervasive Computing, 7, 1, p. 74-81.

KAYNAKLAR DİZİNİ (devam)

Saha, A., Subramanya, A., Pirsiavash, H., 2020, Hidden trigger backdoor attacks, Proceedings of the AAAI Conference on Artificial Intelligence, 34, p. 11957-65.

Saini, A., 2021, https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/, erişim tarihi: 5.04.2022.

Sandro., Gardella, P., 2018, https://github.com/sandro-k/MQTTLensChromeApp, erişim tarihi: 15.04.2022.

Saritas, M. M., Yasar, A., 2019, Performance analysis of ANN and Naive Bayes classification algorithm for data classification, International journal of intelligent systems and applications in engineering, p. 88-91.

Scarfone, K., Mell, P., 2007, Guide to intrusion detection and prevention systems, Technical report, National Institute of Standards and Technology, p. 800-94.

Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., Soman, K. P., 2017, Stock price prediction using LSTM, RNN and CNN-sliding window model, In 2017 international conference on advances in computing, communications and informatics, p. 1643-1647.

Shlens, J., 2014, A tutorial on principal component analysis, Arxiv preprint arXiv:1404.1100.

Siddharthan, H., Deepa, T., Chandhar, P., 2022, SENMQTT-SET: An Intelligent Intrusion Detection in IoT-MQTT networks using ensemble multi cascade features, 10, p. 33095-33110.

Singh, B., Sihag, P., Singh, K., 2017, Modelling of impact of water quality on infiltration rate of soil by random forest regression, Modeling Earth Systems and Environment, p. 999-1004.

Swets, J. A., 1979, ROC analysis applied to the evaluation of medical imaging techniques, Investigative radiology, p. 109-121.

Şahingöz, O. K., Çebi, C. B., Bulut, F. S., Fırat, H., Karataş, G., 2019, Saldırı tespit sistemlerinde makine öğrenmesi modellerinin karşılaştırılması, Erzincan Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 12, 3, s. 1513-1525.

Şeker, A., Diri, B., Balık, H. H., 2017, Derin öğrenme yöntemleri ve uygulamaları hakkında bir inceleme, Gazi Mühendislik Bilimleri Dergisi, s. 47-64.

Tahsien, S. M., Karimipour, H., Spachos, P., 2020, Machine learning based solutions for security of Internet of Things, A survey, J. Netw. Comput. Appl., 161, p. 102630.

KAYNAKLAR DİZİNİ (devam)

Taş, O., Kiani, F., 2021, Nesnelerin interneti ve kablosuz algılayıcı ağların güvenliğine yapılan saldırıların tespit edilmesi ve önlenmesi, Politeknik Dergisi, 24, 1, s. 219-235.

Tuncer, T., Tatar, Y., 2009, Karar ağacı kullanarak saldırı tespit sistemlerinin performans değerlendirmesi, 4. İletişim Teknolojileri Ulusal Sempozyumu, p. 41-48.

Türkay, B., 2018, https://medium.com/@Barturkay/nesnelerin-interneti-iot-uygulamalarının-günümüzdeki-yeri-736cd99e37d9, erişim tarihi: 17.12.2021.

Vacca, J., 2013, Computer and information security handbook, Morgan Kaufmann.

Vaccari, I., Aiello, M., Cambiaso, E., 2020, SlowITe, a novel denial of service attack affecting MQTT, 20, p. 2932.

Vaccari, I., Chiola, G., Aiello, M., Mongelli, M., Cambiaso, E., 2020, MQTTset, a new dataset for machine learning techniques on MQTT, Sensors, 20, p. 6578.

Vania, J., Meniya, A., Jethva, H. B., 2013, A review on botnet and detection technique, International Journal of Computer Trends and Technology, p. 23-29.

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., Venkatraman, S., 2019, Deep learning approach for intelligent intrusion detection system, IEEE Access, 7, p. 41525-41550.

Vujović, Ž., 2021, A case study of the application of WEKA software to solve the problem of liver inflamation.

Waskle, S., Parashar, L., Singh, U., 2020, Intrusion detection system using PCA with random forest approach, In 2020 International Conference on Electronics and Sustainable Communication Systems, p. 803-808.

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., vd., 2018, Machine learning and deep learning methods for cybersecurity, p. 35365-35381.

Yalçınkaya, S., 2021, https://www.siberkavram.com/2021/05/mqtt-protokolunun-tarihcesi-genel.html, erişim tarihi: 16.04.2020.

Yang, F., 2019, An extended idea about decision trees, 2019 International Conference on Computational Science and Computational Intelligence, p. 349-354.

Yiğitbaşı, H, Z., 2011, Nesnelerin interneti ve makineden makineye kavramları için kilit öncül-IPv6, Ulusal IPv6 Konferansı, 171, s. 103-108.

KAYNAKLAR DİZİNİ (devam)

Zhang, H., 2004, The optimality of naive bayes, 1, p. 3.

Anonim, 2007, https://ufonet.03c8.net/#docs/, erişim tarihi: 10.08.2021.

Anonim, 2015, https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/, erişim tarihi: 4.05.2022.

Anonim, https://www.direnc.net/dht11-arduino-sensor-nem-ve-sicaklik-sensoru, erişim tarihi: 1.05.2022.

Anonim, https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications, erişim tarihi: 1.05.2022.

Anonim, https://www.wireshark.org/docs/dfref/m/mqtt.html, erişim tarihi: 20.04.2022.