
GAZİ ÜNİVERSİTESİ
Gazi
1926

YAZILIM TABANLI ARAÇSAL AĞLARA YÖNELİK GERÇEKLEŞTİRİLEN DDOS SALDIRILARININ DERİN ÖĞRENME TABANLI GERÇEK ZAMANLI TESPİTİ

Onur POLAT

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NİSAN 2023

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

İmza
Onur POLAT
07/04/2023

YAZILIM TABANLI ARAÇSAL AĞLARA YÖNELİK GERÇEKLEŞTİRİLEN DDOS SALDIRILARININ DERİN ÖĞRENME TABANLI GERÇEK ZAMANLI TESPİTİ

(Doktora Tezi)

Onur POLAT

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Nisan 2023

ÖZET

Araçsal tasarsız ağlar, (Vehicular Ad Hoc Networks -VANET) trafik verimliliğini ve güvenliğini destekleyerek sürücülere ve yolculara konforlu ulaşım için çeşitli uygulamalar ve hizmetler sunar. Ancak geleneksel VANET'ler, sayıları her geçen gün artan akıllı araçlar nedeniyle akıllı ulaşım sistemlerinin ölçeklenebilirlik, esneklik ve yönetim gibi temel gereksinimlerini karşılamada çeşitli teknik zorluklarla karşılaşmaktadır. Yazılım Tanımlı Ağlar (Software Defined Networks-SDN) esnek, programlanabilir, ölçeklenebilir ağ yapısı ile geleneksel VANET mimarisinde yaşanan sorunlara çözüm sunmaya adaydır. Yazılım Tanımlı Ağlar, kavramının geleneksel VANET'e uyarlanmasıyla oluşturulan mimari kısaca SD-VANET (Software Defined based Vehicular Ad Hoc Networks-SD-VANET) olarak adlandırılmaktadır. Bu yeni mimari, ağın kolayca ölçeklendirilmesine ve esnek ağ yönetimine olanak sağlamaktadır. SD-VANET mimarisi avantajlarına rağmen, aynı zamanda Dağıtık Hizmet Engelleme (Distributed Denial of Service-DDoS) gibi siber saldırı tehditlerine karşı da savunmasızdır. Bu çalışmada, SD-VANET kontrolcüsünü hedef alan DDoS saldırılarının tespiti için SD-VANET_Guard olarak adlandırdığımız bir güvenlik sistemi önerilmiştir. Bu sistem, VANET ağlarının Yazılım Tanımlı Ağ kontrolcüsü içinde görevlendirilerek, DDoS saldırılarının tespit edilmesi ve saldırılara karşı korunmasına yardımcı olacak şekilde tasarlanmıştır. Saldırı tespit metodolojisi, Ryu kontrolcüsü ile veri toplama, veri işleme ve ağ trafik sınıflandırmasını içermektedir. Deneysel sonuçlara göre, SD-VANET_Guard saldırı tespit sistemi, saldırı altındaki SD-VANET ağına gelen trafiğin yaklaşık %90'ının DDoS trafik akışına ait olduğunu tespit etmiştir. Önerilen sistem, SD-VANET mimarise yönelik gerçekleştirilen DDoS saldırılarının tespitinde umut verici sonuçlar elde edilebildiğini göstermektedir.

Bilim Kodu	:	92407
Anahtar Kelimeler	:	Yazılım Tanımlı Ağlar, DDoS saldırıları, VANET, Gerçek Zamanlı Saldırı Tespiti
Sayfa Adedi	:	132
Danışman	:	Doç. Dr. Hüseyin POLAT

DEEP LEARNING BASED REAL-TIME DETECTION OF DDOS ATTACKS ON SOFTWARE-DEFINED BASED VEHICULAR NETWORKS

(Ph. D. Thesis)

Onur POLAT

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

April 2023

ABSTRACT

Vehicular Ad Hoc Networks (VANETs) support traffic efficiency and safety, providing drivers and passengers with various applications and services for comfortable transportation. However, traditional VANETs face various technical challenges in meeting the basic requirements of intelligent transportation systems such as scalability, flexibility and management due to the increasing number of smart vehicles. Software Defined Networks (SDN) is a candidate to offer a solution to the problems experienced in traditional VANET architecture with its flexible, programmable, scalable network structure. The architecture created by adapting the concept of Software Defined Networks to traditional VANET is called SD-VANET (Software Defined based Vehicular Ad Hoc Networks-SD-VANET). This new architecture allows for easy network scaling and flexible network management. Despite its advantages, SD-VANET architecture is also vulnerable to cyber attack threats such as Distributed Denial of Service (DDoS). In this paper, we propose a security system, which we call SD-VANET_Guard, for the detection of DDoS attacks targeting the SD-VANET controller. This system is designed to be deployed in the Software Defined Network controller of VANET networks to help detect and protect against DDoS attacks. The attack detection methodology includes data collection, data processing and network traffic classification with Ryu controller. According to the experimental results, the SD-VANET_Guard intrusion detection system detected that about 90% of the traffic to the SD-VANET network under attack belongs to DDoS traffic flow. The proposed system shows promising results in detecting DDoS attacks against the SD-VANET architecture.

Science Code	:	92407
Key Words	:	Software Defined Networks, DDoS attacks, VANET, Real-Time Intrusion Detection
Page Number	:	132
Supervisor	:	Assoc. Prof. Dr. Hüseyin POLAT

TEŞEKKÜR

Doktora eğitimim boyunca tecrübe, akademik bilgi ve birikimini benden esirgemeyen, desteklerini her zaman yanımda hissettiğim çok kıymetli danışman hocam Doç. Dr. Hüseyin POLAT'a, aynı zamanda akademik eğitimim boyunca ve tez yazım sürecinde değerli desteklerini esirgemeyen kıymetli hocalarım Prof. Dr. Necaatin BARIŞÇI'ya, Doç. Dr. Halil Murat ÜNVER'e, Doç. Dr. Cemal KOÇAK'a, Doç. Dr. Muammer TÜRKOĞLU'na, Gazi Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Bölümü öğretim üyeleri kıymetli hocalarıma ve idari personellerine, kalbi şükranlarımı sunarım.

Akademik hayatım boyunca desteğini esirgemeyen sevgili eşim Mehtap POLAT, çocuklarım Müzeyyen Rahel ve Rukiye POLAT'a teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
ms	Milisaniye

Kısaltmalar	**Açıklamalar**
AP	Access Point
CPU	Central Processing Unit
DSA	Derin Sinir Ağı
DNS	Domain Name System
DVM	Destek Vektör Makinesi
DDoS	Distributed Denial of Service
DSRC	Dedicated Short Range Communication
EDR	Event Data Recorder
ESA	Evrişimsel Sinir Ağı
FP	False Positive
FN	False Negative
GDVM	Gelişmiş Destek Vektör Makinesi
GPS	Global Positioning System
GTB	Geçitli Tekrarlayan Birimler
HMF	Hidden Markov Filtering
ICMP	Internet Control Protocol
IP	Internet Protocol
K-EYK	K-En Yakın Komşu

Kısaltmalar	Açıklamalar
LDA	Linear Discriminant Analysis
LDAP	Lightweight Directory Access Protocol
LVQ1	Learning Vector Quantization
LR	Lojistik Regresyon
LEDEM	Learning-Driven Detection Mitigation
MC-SQLR	Microsoft SQL Server Resolution Protocol
MT-LNet	Multi-Task Learning Network
MRMR	Minimum Redundancy Maximum Relevance
MLP	Multiple Layer Perception
NTP	Network Type Protocol
OVS	OpenvSwitch
ONF	Open Network Foundation
ROG	Receiver Operating Characteristic
RCP	Resource Command Processor
SDN	Software Defined Network
SD-VANET	Software Defined based Vehicular Ad Hoc Networks
SSDP	Simple Service Discovery Protocol
SOM	Self-Organizing Maps
SBFS	Sequential Backward Floating Selection
SFS	Sequential Forward Selection
SBS	Sequential Backward Selection
TCP	Transmission Control Protocol
TSA	Tekrarlayan Sinir Ağı
TP	True Positive
UDP	User Datagram Protocol

Kısaltmalar	Açıklamalar
UPnP	Universal Plug and Play
UKSB	Uzun Kısa Süreli Bellek
VANET	Vehicular Ad Hoc Networks
VT	Variance Threshold
VM	Virtual Machine
YOK	Yığınlanmış Oto Kodlayıcı
YSA	Yapay Sinir Ağı
YSOK	Yığınlanmış Seyrek Oto Kodlayıcı
WAVE	Wireless Access in Vehicula Environments

1. GİRİŞ

Dünya üzerindeki insan popülasyonunun artması ile birlikte trafikte kullanılan araç sayısı da gün geçtikçe artmaktadır. Modern dünyada, seyahatlerin çoğunlukla özel araçlar ile gerçekleştirilmesi ve karayolu taşımacılığının yaygınlaşması, ulaşım altyapılarını ve ulaşım güvenliğini gittikçe zorlamaktadır. Araç sayısındaki artış, beraberinde trafik kazaları ve yaralanmalarında artışı getirmektedir. Bu nedenle insan hayatına ciddi zarar verebilecek trafik kazalarını önlemek, güvenli ve konforlu bir seyahat ve karayolu taşımacılığı için akıllı ulaşım sistemlerine ihtiyaç duyulmaktadır. Akıllı ulaşım sistemleri, araç sürücüleri arasında trafik bilgisi, kaza durumu, acil durumlar, yol yapımı gibi önemli ve faydalı bilgi alışverişi için kullanılabilir [1]. Otomatik bilgi alışverişi, sürücülerin bir kazayı önlemek için trafiğe daha fazla hâkim olmasını, erken uyarılar almasını ve seyahat için alternatif bir rota oluşturulmasını sağlayabilir.

Trafik güvenliğinin sağlanması ve sürücülerin daha bilinçli ve güvenli bir şekilde araç kullanabilmesine destek olmak için ilk olarak Mobil Tasarsız Ağ (Mobile Ad Hoc Network-MANET), teknolojisi kullanılmaya başlandı. Fakat MANET teknolojisi zamanla yönlendirme, kapasite ve enerji yönetimi konularında zorluklarla karşı karşıya kalmıştır. MANET'in karşılaştığı bu zorlukların üstesinden gelmek için Araçsal Tasarsız Ağ (Vehicular Ad Hoc Network-VANET) teknolojisi geliştirilmiştir. Araçlar arasındaki veri paylaşımında, IEEE 802.11p kablosuz erişim teknolojilerini destekleyen VANET, trafik güvenliğini artıran ve rahat bir sürüş deneyimi sağlayan birçok uygulamaya olanak tanıdığı için akıllı ulaşım için önemli bir teknoloji olarak görülmektedir. VANET, araçların birbirleriyle doğrudan veya dolaylı olarak iletişim kurabildiği bir ağdır. Araçlar, çevredeki diğer araçların, trafik işaretlerinin ve diğer altyapı elemanlarının konum bilgilerini alabilir ve bu bilgileri diğer araçlarla paylaşabilir. Bu sayede sürücüler trafik akışını daha iyi yönetebilir ve kaza riskini azaltabilirler. VANET ayrıca acil durum çağrılarını ve kaza bildirimlerini de otomatik olarak gönderebilir ve alabilir. Bu sayede acil durumlarda hızlı müdahale edilebilir ve kurtarma işlemleri hızlandırılabilir. VANET teknolojisi, ileri sürüş destek sistemleri ve otonom araç teknolojisi geliştikçe daha da önem kazanmaktadır.

VANET'te araçlar birbirleriyle (vehicle-to-vehicle-V2V) ve Yol Kenarı Üniteleri (YKÜ-Road Side Unit-RSU) aracılığıyla araç ile altyapı (vehicle-to-infrastructure) arasında doğrudan iletişim kurabilir. VANET teknolojisi sayesinde araçlar sıkışık bir trafikten kaçınmak için birbirleriyle mesajlaşabilir ve kendilerine yoğun olmayan bir trafik rotası belirleyebilirler [2]. VANET ayrıca internet erişimi sayesinde sürücüler ve yolcular için ek kolaylıklar sunan uygulamalara da olanak sağlamaktadır [3].

VANET teknolojisi araçtan araca ve araçtan yol bilgi sistemlerine veri aktarabilmesine olanak tanıdığı için daha güvenli ve daha konforlu bir sürüş olanağı sağlayabilmeyi vaat etmektedir. Ancak bütün ağ sistemlerinde olduğu gibi VANET teknolojisinin de yaygın bir şekilde kullanılabilmesi için çözüm bulunması gereken bazı sorunları vardır. Bu çözüm bekleyen sorunlar; ağ trafik sürekliliğinin sağlanması, birçok araçtan ve yol bilgi sistemlerinden oluşan dağıtık yapının yönetilmesi, akıllı bilgi sistemlerinden gelen yoğun verinin işlenmesi ve gerçek zamanlı bilgi akışının sağlanması (yol tıkanıklığı, kaza raporları, mevcut park yerleri, en yakın hastane, benzin istasyonları vb) gibi vektörlerdir [4]. Ayrıca mevcut VANET mimarisi 5G teknolojisin talep ettiği örneğin; park alanı, acil durum uyarısı, multimedya aktarımı gibi içeriğe dayalı bilgilerin değişimi için de uygun değildir.

Sonuçta geleneksel VANET mimarisi ağ yönetim, ölçekleme, yönlendirme gibi konular açısından, gelişmiş ulaşım sistemlerinin gerektirdiği bilgi ve iletişim teknolojilerinin temel gereksinimlerini karşılamada yetersiz kalmaya başlamıştır. Akıllı şehirler kavramı geliştikçe otomasyona ihtiyaç duyulmakta ve gerekli hizmetlerin etkin bir şekilde verilmesi gerekmektedir. Yaşanan sıkıntılar araçsal ağlarda bilgi, konfor, güvenlik gibi servislerin desteklenmesiyle aşılabilir. Fakat araçsal ağların karmaşık ve esnek olmayan mimarisi, yüksek mobilite, aralıklı bağlantı, uygulamaların heterojenliği gibi bir dizi zorluklar içermektedir. Geleceğimizi şekillendirecek akıllı şehirler ve akıllı araç teknolojisinin gereksinimlerine verimli cevap veren, internet ile entegre olabilen, gerçek zamanlı trafik bilgisi elde edilebilen yeni kablosuz iletişim ağ yapılarının kurulması gerekmektedir [5].

Esnek, programlanabilir, ölçeklenebilir ve yönetim kolaylığı sağlayan Yazılım Tanımlı Ağ paradigması geleneksel VANET mimarisinde yaşanan sıkıntılara çözüm sunmaya adaydır. Yazılım Tanımlı Ağ'da kontrol ve veri düzlemlerinin ayrı yapılandırılmış olması, ağın merkezi bir kontrolcü tarafından yönetilmesi, VANET'de düğüm sayısı hızla artsa bile bu ağın daha etkin ve kolay şekilde yönetilmesine olanak sağlar. Yazılım Tanımlı Ağ, ağın donanımından bağımsız olarak yönetim olanağı sağladığından, ağın ölçeklenmesi ve esnekliği daha kolaydır. Ağdaki tüm cihazlarla ilgili yapılandırmalar merkezi kontrolcü sayesinde kolayca gerçekleştirilir. Ayrıca ağ yöneticileri ağdaki hataları daha hızlı bir şekilde tespit edip çözebilirler [6].

SD-VANET mimarisinin getirdiği birçok avantaj mevcut olsa da, bu mimari hâla kendine has bazı güvenlik tehditlerine karşı çözüm geliştirilmesini beklemektedir. Özellikle Dağıtık Hizmet Engelleme (Distributed Denial of Service -DDoS) saldırıları SD-VANET ağlar için büyük tehlike oluşturmaktadır.

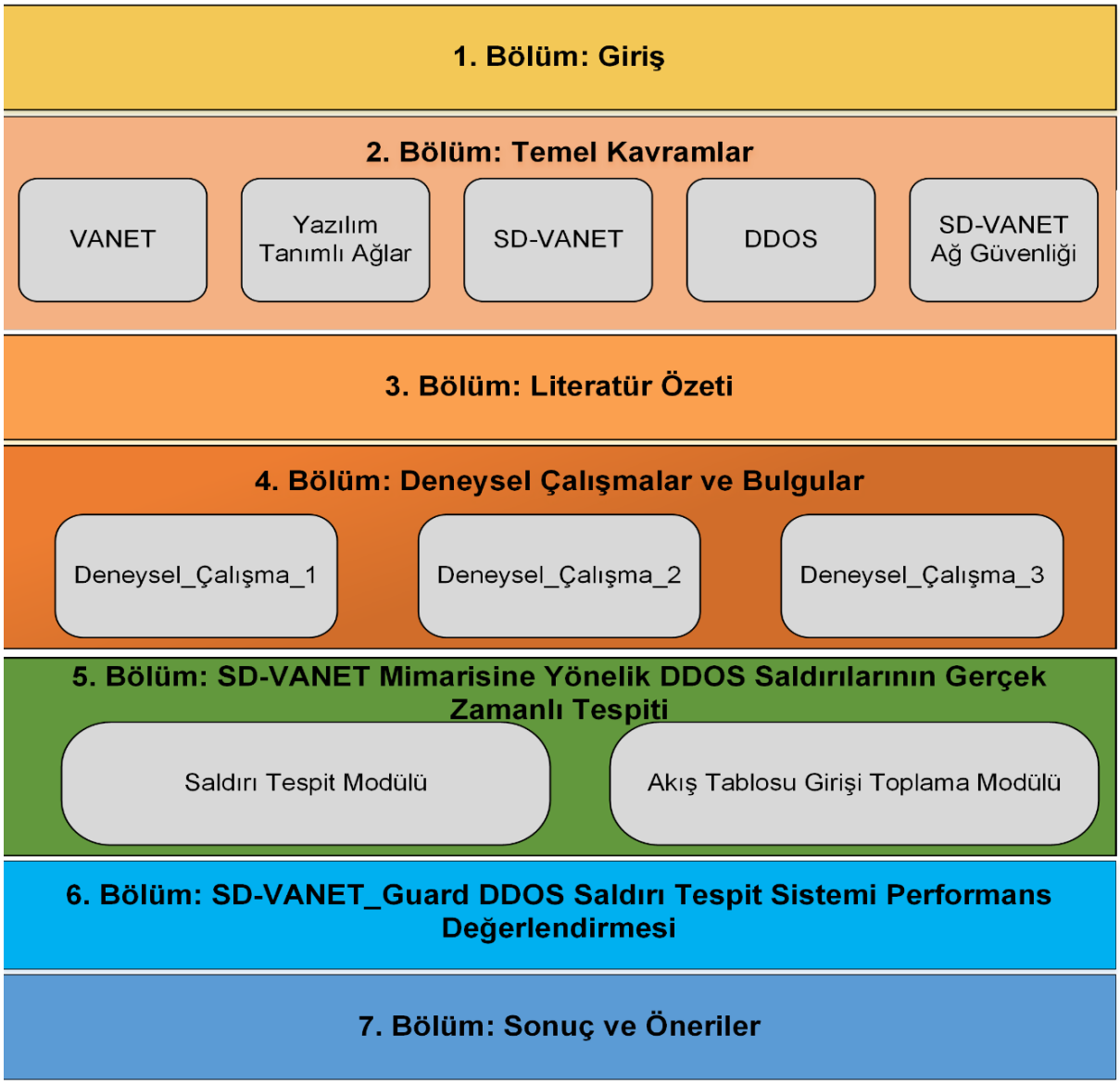
DDoS saldırılarında, ağ üzerindeki kontrolcü, anahtar, sunucular veya araçlar hedef alınarak SD-VANET mimarisininin kullanılamaz hale getirilmesi amaçlanır. Kontrolcüyü hedef alan DDoS saldırısında kontrolcünün işlem ve iletişim kapasitesi hedef alınırken, aynı zamanda veri düzleminde bulunan anahtarların da işlemci, bellek gibi kaynaklarının tüketilmesi amaçlanır. Ayrıca SD-VANET mimarisi üzerindeki sunuculara yönelik yapılan saldırılar, son kullanıcıların servislere erişimini engelleme amacıyla gerçekleştirilir.

SD-VANET mimarisinde kontrolcü ağın beyni konumunda olduğu için saldırganların ana hedefi haline gelmiştir. Eğer bir saldırgan, kontrolcüyü hedef alan DDoS saldırısı gerçekleştirirse, bütün ağ olumsuz etkilenebilir, hatta bazı durumlarda SD-VANET mimarisi tamamen çökebilir [7]. SD-VANET mimarisine karşı gerçekleştirilebilecek bu tür saldırılar trafik güvenliğini ciddi bir şeklide tehlikeye sokabilir. SD-VANET uygulamaları, araçlar arasında güvenlik açısından kritik veri alışverişini mümkün kılmaktadır. Bu yüzden SD-VANET'lerin akıllı ulaşım sistemlerine büyük çapta uygulanabilmesi için ilgili güvenlik tehditlerinin ele alınması gerekir [8]. Ağın sürekliliğini ve kullanılabilirliğini korumak açısından DDoS türündeki saldırıların etkin bir şekilde tespit edilerek buna karşı hızlı ve etkili önlemlerin alınması gerekir.

Bu çalışmada, SD-VANET kontrolcüsünü hedef alan DDoS saldırılarının tespiti için SD-VANET_Guard olarak adlandırdığımız bir güvenlik sistemi önerilmektedir. Bu güvenlik sistemi, ağ kontrolcüsü içinde çalışmaktadır ve DDoS saldırılarının etkin bir şekilde tespit edilerek bu saldırılara karşı korunmaya yardımcı olacak şekilde tasarlanmıştır. Saldırı tespit metodolojisi Ryu kontrolcüsü ile veri toplama ve Çok Görevli Öğrenme Ağı (Multi-Task Learning Network-MT-LNet) ile veri işleme ve ağ trafik verilerinin sınıflandırılmasını içermektedir.

Tezin organizasyonu

Tezin organizasyonu Şekil 1.1' de verilmiş olup bölüm içerikleri aşağıda özetlenmiştir:



Şekil 1.1. Tezin organizasyon şeması

Tezin 2. bölümünde, araçsal ağların temelini oluşturan teknoloji ve ağ güvenliği konusundaki avantaj ve dezavantajları açıklanmıştır. Akıllı ulaşım sistemlerinin temelini oluşturan VANET ve yeni bir ağ kavramı olarak Yazılım Tanımlı Ağ yapısı detaylı bir şekilde incelenmiştir. Daha sonra bu iki mimarinin bileşiminden oluşan SD-VANET mimarisinin geleneksel VANET mimarisine göre avantaj ve dezavantajları açıklanmıştır. SD-VANET mimarisine yönelik gerçekleştirilen siber saldırıların en kapsamlı ve tehlikelisi olan DDoS saldırıları kapsamlı bir şekilde ele alınmıştır.

Tezin 3. bölümünde, Yazılım Tanımlı Ağlara yönelik gerçekleşen saldırılara karşı literatürde bulunan makine öğrenmesi ve derin öğrenme tabanlı saldırı önleme ile ilgili yapılan çalışmalar detaylı bir şekilde incelenmiştir.

Tezin 4. bölümünde, DDoS saldırılarını en etkin şekilde tespit eden modelin belirlenmesi için oluşturulan makine öğrenme ve derin öğrenme modelleri ile deneysel çalışmalar yapılmıştır. Ayrıca bu deneysel çalışmalardan elde edilen sonuçlar literatürde bulunan benzer çalışmalarla karşılaştırılmıştır.

Tezin 5. bölümünde, SD-VANET mimarisine yönelik gerçekleştirilen DDoS saldırılarını önlemeye yardımcı olması için önerilen yaklaşımın safhaları açıklanmıştır.

Tezin 6. bölümünde geliştirilen saldırı tespit ve önleme sisteminin test edilmesi için oluşturulan modeller ve deneysel senaryolar hakkında detaylar ve deneysel sonuçlar verilmiştir. Tezin son bölümünde, elde edilen deneysel sonuçlar yorumlanarak bu çalışmanın literatüre katkısı tartışılmıştır.

2. TEMEL KAVRAMLAR

Karayolları ve otoyollar üzerinden güvenli ve emniyetli yolculuklara olan talep son yıllarda hızla artmaktadır. Aynı zamanda, akıllı hareketlilik kavramını geliştirerek vatandaşların yaşam kalitesini artırmak için akıllı şehir paradigması ortaya çıkmıştır. Akıllı şehir kavramı ile ulaşım teknolojisinin gelişimine paralel olarak akıllı otomobillerin de daha güvenli, daha rahat ve daha verimli bir ulaşım sağlayacağı öngörülmektedir. Akıllı ulaşım servisleri için sürekli yeni protokoller ve mimariler geliştirilmektedir. VANET, mimarisi cazip güvenlik ve bilgi-eğlence hizmetleri sağlayarak bu kavramın gerçekleştirilmesinde önemli bir teknoloji olarak kabul edilmektedir. Bu nedenle VANET mimarisine son zamanlarda birçok yatırım yapılmıştır. Fakat VANET yapısı, zayıf bağlantı, ölçeklenebilirlik, yüksek düğüm hareketliliğinin yönetilebilirliği ve kullanıcı gizliliğininin güvence altına alınması gibi bir takım zorluklar barındırmaktadır. Mevcut sıkıntıları gidermek ve sistemi daha verimli bir hale getirmek için köklü bir dönüşüme ihtiyaç vardır. Son zamanlarda yeni bir ağ kavramı olarak ortaya çıkan Yazılım Tanımlı Ağ mimarisi, mevcut VANET mimarisinde yaşanan sıkıntılara esnek, ölçeklenebilir ve programlanabilir mimarisi ile çözüm sunmaya adaydır. Yazılım Tanımlı Ağ tabanlı olarak oluşturulan VANET yapısı (SD-VANET), ağ yönetiminde esneklik ve dinamizm sağlayan yeni bir yaklaşımdır. Bu yaklaşım, ağ yöneticilerinin ağ trafiğini daha iyi kontrol etmelerini sağlar. Ancak, SD-VANET yapısına gerçekleştirilebilecek DDoS saldırıları, ağın performansını ciddi şekilde etkileyebilir ve hatta ağın çökmesine neden olabilir. Bu tür saldırılar, ağa yüksek hacimli trafik göndererek ağ kaynaklarını tüketir veya ağdaki cihazların doğru çalışmasını engeller. Böylece hiç kimse uygulamaları/hizmetleri kullanamaz. Bu tür kullanılabilirliğe yönelik bir tehdit gerçekleşirse feci durumlar yaratabilir. Bu yüzden SD-VANET yapılarına gerçekleştirilen DDoS saldırılarının etkin bir şekilde tespit edilmesi ve buna göre hızlıca önlem alınması gerekmektedir. Bu çalışmada, DDoS saldırılarını en etkin şekilde tespit eden modelin belirlenmesi için oluşturulan makine öğrenme ve derin öğrenme modelleri ile çeşitli deneysel çalışmalar yapılmıştır.

Bu bölümde çalışmanın konusu içerinde yeralan VANET, Yazılım Tanımlı Ağ, bu iki mimarinin kombine edilmesi ile oluşturulan SD-VANET ve DDoS kavramlarının detayları verilmiştir.

2.1. Araçsal Tasarsız Ağlar

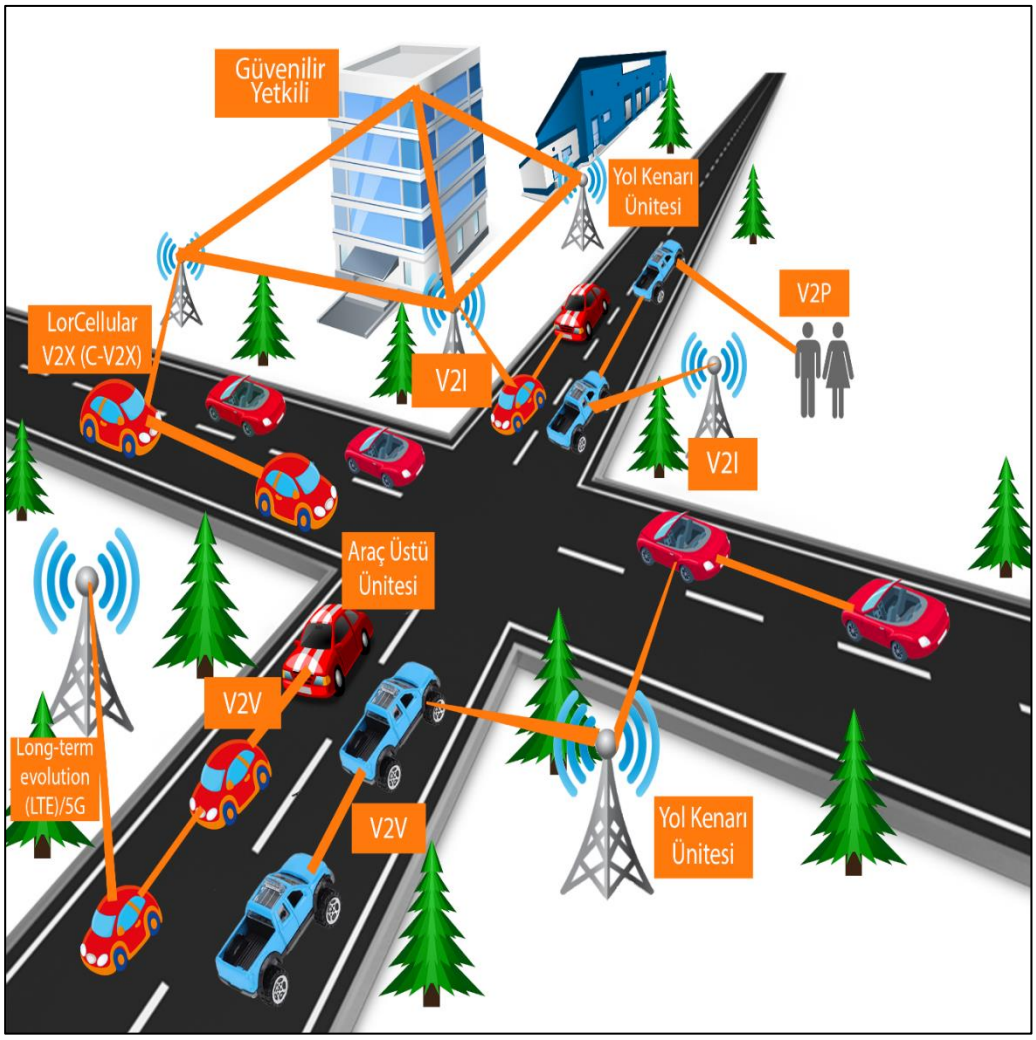
Tasarsız Ağlar (Ad Hoc Networks), merkezi bir erişim noktasına ihtiyaç duymadan birbirleriyle doğrudan iletişim kurabilen bir ağ türüdür. İlk tasarsız ağlar, askeri operasyonlarda kullanılmak

üzere geliştirildi. Bu ağlar, özellikle hareketli askeri birimler arasında iletişim kurmak için tasarlanmıştı. Tasarsız ağlar, hareketli birimler arasındaki iletişimi sürdürmek için fiziksel altyapıya ihtiyaç duymadan çalışabilen bir çözüm sunuyordu. Daha sonra MANET, kablosuz tasarsız ağların bir alt kümesi olarak ortaya çıktı. MANET, tasarsız ağların mobilite ve ölçeklenebilirlik açısından daha geliştirilmiş bir sürümüdür. MANET, mobil cihazlar arasında kablosuz iletişim kurulabilen bir ağ olarak tasarlandı. Bu, özellikle askeri operasyonlar, acil durum kurtarma çalışmaları ve diğer benzeri durumlarda büyük bir avantaj sağladı. MANET, merkezi bir erişim noktasına ihtiyaç duymadan çalışabilen bir ağ yapısına sahip olduğu için, hızlı kurulum, ölçeklenebilirlik ve bağımsızlık gibi avantajlara sahiptir.

Fakat MANET güvenlik, performans ve yönetim açısından bazı dezavantajlar içermektedir. MANET, diğer ağ türlerine kıyasla daha düşük bir güvenlik seviyesine sahiptir. Bu, güvenliği gerektiren durumlarda MANET kullanımını kısıtlayabilir. MANET'de, ağın yapısı düğümlerin hareketlerine ve iletişim kapasitelerine göre dinamik olarak değişebildiğinden ağın performansı düşürebilir. Ayrıca MANET, merkezi bir erişim noktasına sahip olmadığından, ağ yönetimi daha zor olabilir. Bu durum, organizasyonların daha iyi bir ağ yönetimi için daha fazla kaynak ayırması manasına gelir.

MANET yapısının dezavantajları, kablosuz ağ teknolojilerinin daha da gelişmesi ve akıllı araçların yaygınlaşması ile birlikte VANET fikri ortaya çıktı. Bu teknoloji, araçlar arasında kablosuz iletişim kurulabilmesine imkân tanıdı. İlk olarak, trafik yönetimi ve güvenliği amaçlarıyla düşünülen VANET, daha sonra gelişen teknolojiler sayesinde birçok uygulama alanına yayıldı. Özellikle, araçların birbirleriyle etkileşim halinde olabileceği, birbirleriyle bilgi alışverişinde bulunabileceği ve bu sayede sürücülerin güvenliğini arttırabileceği fikri, VANET teknolojisinin daha da gelişmesine katkıda bulundu. Bugün, VANET teknolojisi, araçların birbirleriyle ve çevreleriyle etkileşim halinde olduğu akıllı şehirler, trafik yönetimi, sürücü destek sistemleri ve otonom araçlar gibi birçok uygulama alanında kullanılmaktadır.

VANET teknolojisi, Yol Kenarı Ünitesi (YKÜ-Road Side Unit-RSU), Uygulama Ünitesi (UÜ-Application Unit-AU) ve Araç Üstü Ünitesi(AÜÜ-On Board Unit-OBU) olmak üzere 3 temel bileşenden oluşmaktadır (Şekil 2.1).



Şekil 2.1. Araçsal ağ genel mimarisi

*Yol Kenarı Ünitesi(YKÜ) :*Araçlarla iletişim kurmak için kullanılan yol kenarına, park alanı veya kavşak gibi belirli bir yere sabitlenmiş kablosuz haberleşme yapan düğümlerdir. Yol Kenarı Üniteleri, araçlarla iletişim kurarak trafik sistemlerinin yönetiminde önemli bir rol oynarlar. Yol Kenarı Üniteleri, normal araçlara göre daha yüksek iletim menzili ile daha geniş alanları kapsar ve bu nedenle menzilleri içinde bulunan çok sayıda araca erişim sağlar. Araçlar, tek-atlamalı komşularına periyodik olarak beacon mesajları yayınlar ve bu verileri ve diğer olay bilgilerini Yol Kenarı Üniteleri ile paylaşır. Yol Kenarı Üniteleri aldıkları bu verileri analiz etme ve farklı amaçla kullanma yeteneğine de sahiptir. Yol Kenarı Üniteleri, araçlarla kablosuz olarak iletişim kurmak için çeşitli teknolojiler kullanabilir. Bu teknolojilerin seçimi Yol Kenarı Ünitesinin konumu, trafik yönetimi uygulamalarının türü ve çevresel faktörlere bağlı olacaktır.

Araç Üstü Ünitesi(AÜÜ): Araç Üstü Ünitesi, çeşitli sensörler kullanarak çevresel ve araç verilerini toplayan, işleyen ve iletişim sağlayan cihazdır. Bu cihaz, araçların performansını izlemek, sürücü

davranışlarını analiz etmek ve trafik yönetiminde kullanılan verileri toplamak için kullanılır. Araç Üstü Ünitesi genellikle GPS sensörü, hız sensörü, jiroskopik sensörler, ivmeölçer, yol eğimi sensörü, yakıt seviyesi sensörü, araç bataryası sensörü, telematik sensörü gibi çeşitli sensörler kullanır [9].

GPS sensörü: Araç konumunu belirlemek için kullanılır.
Hız sensörü: Araç hızını ölçer.
Jiroskopik sensörler: Araç hareketini ölçmek için kullanılır.
İvmeölçerler: Araç ivmesini ölçmek için kullanılır.
Yol eğimi sensörü: Araçın yokuş yukarı ya da yokuş aşağı olup olmadığını belirlemek için kullanılır.
Yakıt seviyesi sensörü: Yakıt seviyesini ölçer.
Araç bataryası sensörü: Araç bataryasının durumunu ölçer.
Telematik sensörü: Araçla ilgili telematik verileri toplamak için kullanılır.

Araç Üstü Üniteleri, veri toplama, işleme ve iletişim konusunda oldukça esnek bir yapıya sahiptir. Farklı senaryolarda farklı iletişim yöntemleri kullanarak araç verilerinin doğru ve güvenli bir şekilde paylaşılmasını sağlar. Araç Üstü Ünitesileri diğer araçlar ve Yol Kenarı Üniteleri ile kablosuz haberleşme teknolojileri kullanarak iletişim kurarlar. İletişim yöntemleri, Araç Üstü Ünitelerinin amaçlarına ve kullanım senaryosuna bağlı olarak değişebilir. Araç Üstü Ünitesi ve Yol Kenarı Ünitesi arasındaki iletişim, genellikle aşağıdaki adımları içerir:

- Araç Üstü Ünitesi, yakındaki Yol Kenarı Ünitelerini tespit eder ve onlarla iletişim kurmaya hazırlanır.
- Yol Kenarı Üniteleri, Araç Üstü Ünitesinden gelen isteği alır ve iletişim için uygun olduğunu doğrular.
- Araç Üstü Ünitesi, Yol Kenarı Ünitesinden veri talebinde bulunur. Bu veri talebi, Araç Üstü Ünitesinin konumu, hızı, yönü vb. gibi çeşitli verileri içerebilir.
- Yol Kenarı Üniteleri, Araç Üstü Ünitesinin veri talebini işler ve gerekli verileri Araç Üstü Ünitesine gönderir.
- Araç Üstü Ünitesi, aldığı verileri işler ve sürücüye veya diğer sistemlere bildirir.

Araç Üstü Ünitesi, aracın yerel ağına da bağlanabilir. Bu, araç içindeki diğer sistemlerle veya cihazlarla veri paylaşımını sağlar. Örneğin, sürücü navigasyon cihazından bir rota alabilir ve bu bilgiyi Araç Üstü Üniteye ileterek trafik durumu hakkında daha doğru bir tahmin yapılabilir.

*Uygulama Ünitesi(UÜ):*Kablolu veya kablosuz bağlantı yoluyla Araç Üstü Ünitesinin iletişim yeteneklerini kullanan uygulama veya kullanıcı ara yüzü içeren aracın içine yerleştirilmiş bir cihazdır [10]. Uygulama Ünitesi iletişimi, Araç Üstü Ünitesi cihazlarının yardımıyla gerçekleştirir. Uygulama Ünitesi, kişisel bir cihaz veya kişisel bir dijital asistan olarak da kullanılabilir [11].

Bu VANET bileşenleri, veri aktarım aralığı ve hızı, gecikme süresi ve güvenlik gibi iletişimin çeşitli yönlerini belirleyecek olan kablosuz iletişim standartları ve protokollerini kullanarak iletişim kurar. Dinamik ve ağ topolojisindeki hızlı değişimler düğümlerin mahremiyetini artırır, sık sık bağlantı kesilmesine, değişkenliğe ve el sıkışmanın imkânsızlığına neden olur. Bu sık sinyal kesintilerinden dolayı cihazlar arasında veri iletiminde zorluklar yaşanır [10]. Kablosuz VANET mimarisinde araçtan araca, araçtan Yol Kenarı Ünitesine veya baz istasyonları arasında iletişimin kurulması için farklı kablosuz erişim teknolojileri vardır. Bu teknolojiler trafik yönetimi, servis kalitesi, iyileştirme ve sürücü ve yolcuların yol konforu, güvenlik, trafik vs. uygulamalarına erişimi için kullanılır. Bu erişim teknolojileri Çizelge 2.1'de gösterilmiştir.

Çizelge 2.1. Kablosuz erişim teknolojileri [10]

Kablosuz İletişim Teknolojileri	Standart	Menzil, Alan	Frekans bandı	Kanal Bant genişliği	Veri Aktarım Hızı	Uygulamalar
Hücresel teknolojiler	GSM/GPRS/EDGE (2G), UMTS/ HSPA (3G), LTE (4G)	35 km'ye kadar	900/1800/1900/ 2100 MHz	5–20 MHz (4 G)	2–100 Mbps	V2I ve araçtan mobil telefon iletişimi
WiMAX	802.16 a	5 km'ye kadar	2-11 GHz	20MHz	75 Mbps	VOIP, internet erişimi, geniş bant Kablosuz erişim
DSRC	802.11 p	300-1000 m	5,8-5,9 GHz	10 MHz	6-27 Mbps	V2V ve Yol Kenarı Ünitesinden araca
Wi-Fi	802.11 b ve 802.11 a/g	100-1000 m	20-25 MHz	2.4 GHz	108-600 Mbps	V2V ve V2I
Bluetooth	802.15.1	1-100 m	2,4–2,48 GHz	1 MHz	2-3 Mbps	Ses Uygulamaları
Zigbee	802.15.4	10-75 m	2,4 GHz	5 MHz	250 Kbps	Sensör ağları, Akıllı sayaçlar, Uzak Kontrol Ünitesi

Araçlar arasında geçici iletişimi kolaylaştırmak için araç ortamında kablosuz erişim (Wireless Access in Vehicular Environment- WAVE) IEEE P1609, özel kısa menzilli iletişim (Dedicated Short Range Communication-DSRC) IEEE802.11p, WiMAX IEEE 802.16, Bluetooth IEEE 802.15.1, MBWA IEEE 802.20, kızılötesi ve hücresel gibi çoklu ağ teknolojilerini kullanılır.

Araçların kendi aralarında veya sabit mobil düğümler olan Yol Kenarı Üniteleri ile iletişim kurması amacıyla VANET ağlar için araç ortamında kablosuz erişim WAVE, C2CNet ve CALM mimarileri geliştirilmiştir.

VANET ağlarda genel olarak diğer iki mimariye göre daha avantajlı olan IEEE 802.11p radyo frekans kanalı tabanlı WAVE mimarisi kullanılır. WAVE iletişimi, araç bilgilerini ve trafik akışını güncelleyerek yolcuların emniyetini, trafik verimliliği sağlar [11].

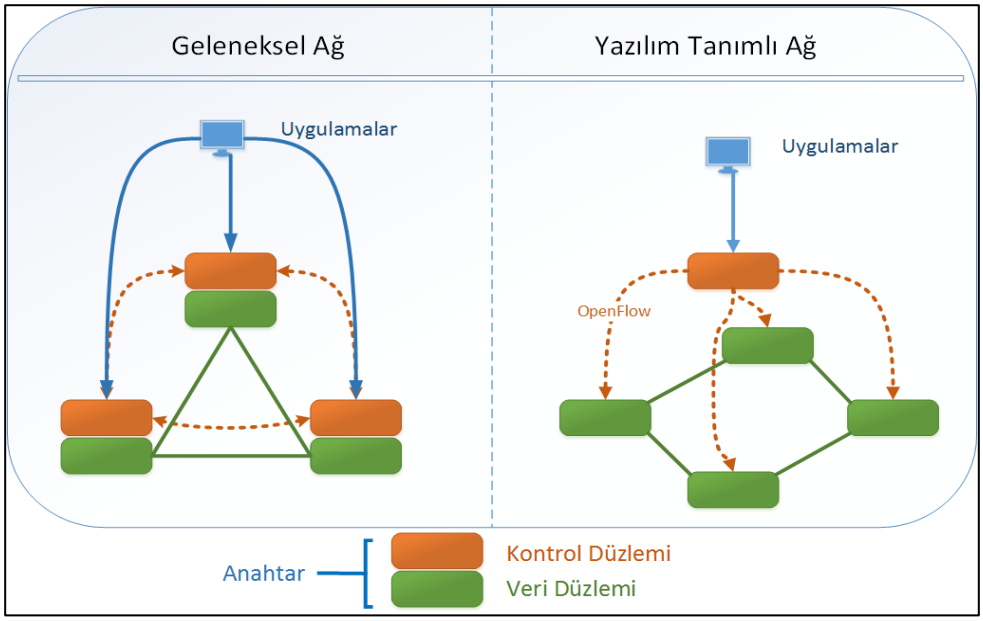
VANET mimarisinde araçlar ile Yol Kenarı Üniteleri, araçlar ile altyapı birimleri arasında ve alt yapı birimlerinin kendi arasındaki iletişim: Araçtan araca (Vehicle to Vehicle -V2V), 4G/Long Term Evolution (LTE) gibi mobil geniş bant aracılığıyla araçtan altyapıya (Vehicle-to-Infrastructure -V2I), altyapı içi (Intra-Infrastructure communication -I2I), araçtan sensöre (Vehicle-to-Sensor communication -V2S), araçtan kişisel cihaza (Vehicle-to-Personal Device communication -V2PD), araçtan hücresel ağ alt yapısına (Vehicle-to-Cellular Network Infrastructure Communication-V2CN) olmak üzere 5 şekilde gerçekleşir.

Akıllı şehirler, akıllı ulaşım ve mobilitenin öneminin artması, 5. nesil (5G) mobil ağlar gibi yeni teknolojilerin ortaya çıkmasına neden olmuştur. Bu teknolojiler araçlar, altyapı ve yayalar gibi farklı yol kullanıcılarını birbirine bağlayan, araçtan araç içi dâhil herhangi bir cihaza ve araçtan diğer yerleşik cihazlara bağlantısı olan araçtan her şeye iletişim (Vehicle to Everything communications V2X) yenilikçi özelliğini beraberinde getirmiştir [12].

2.2. Yazılım Tanımlı Ağlar

Yazılım Tanımlı Ağlar (Software Defined Networks-SDN) , ağ yönetimi ve yeni nesil ağların (5G hücresel, Wi-Fi 6, 5G, uç bilişim, nesnelerin interneti vs.) geleceği için umut verici bir teknolojidir. Yazılım Tanımlı Ağlar, akıllı bir yapılandırmaya, yenilikçi ağlara uyum sağlamak için daha iyi bir esnekliğe, uygun maliyetli, yüksek performanslı ve günümüz uygulamalarının yüksek bant genişliği ve dinamik doğası için ideal bir mimariye sahiptir. Yazılım Tanımlı Ağlar, geleneksel ağ

mimarisinde bir arada bulunan veri düzlemi ve kontrol düzlemini ayırarak ağ yönetiminin etkin ve kolay bir şekilde yapılmasını sağlamak için geliştirilmiş bir ağ paradigmasıdır (Şekil 2.2). Kontrol mekanizması soyutlanarak harici bir cihaza taşınır. Yazılım Tanımlı Ağ mimarisi içinde bulunan fiziksel cihazlar kontrol işlevleri veya herhangi bir yazılım olmaksızın yalnızca basit iletim cihazı haline gelir. Paket yönlendirme hedef tabanlı değil akış tabanlıdır. Yazılım Tanımlı Ağ paradigmasında bir akış, bir kaynak ile bir hedef arasındaki bir paket dizisidir.



Şekil 2.2. Gelensel ağ yapısının ve Yazılım Tanımlı Ağ yapısının karşılaştırılması

Yazılım Tanımlı Ağlar, Çizelge 2.2'de gösterildiği gibi çeşitli faktörlerde, geleneksel ağ paradigmalarından farklıdır.

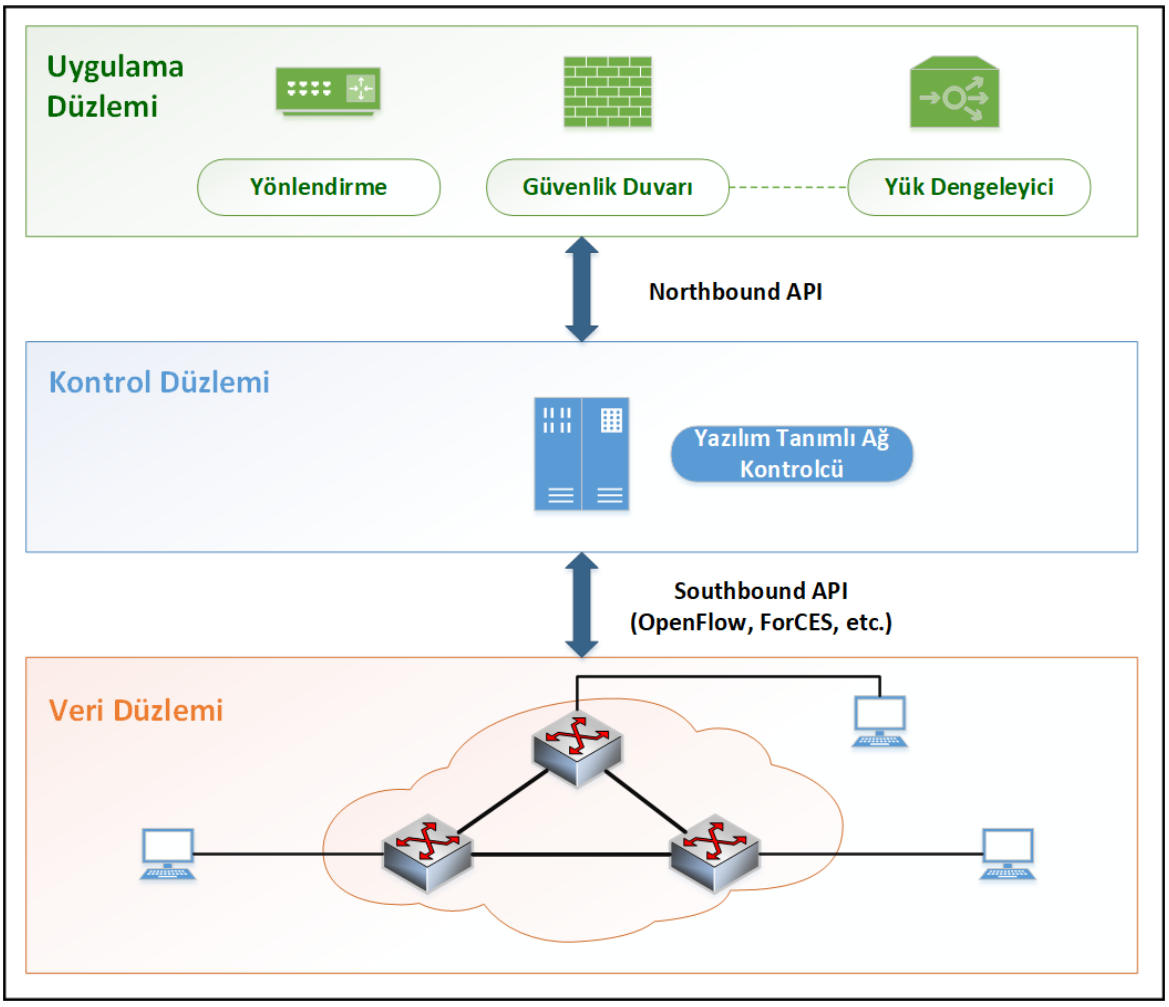
Çizelge 2.2. Yazılım tanımlı ağ ve geleneksel ağ özelliklerinin karşılaştırılması

	Yazılım Tanımlı Ağlar	Geleneksel Ağlar
Özellikler	Veri ve Kontrol Düzlemi birbirinden ayrılır. Programlanabilir. Ağdaki değişimlere ani tepki verir.	Çok cihazlı, komplex, dinamik ortam.
Mimari	Programlanabilir, merkezi	Dağıtık merkezi olmayan kontrol
Yapılandırma	Merkezileştirilmiş kontrol mantığı sayesinde otomatik yapılandırma	Ağa yeni hizmetler veya protokoller uygulandığında ağ içindeki her bir anahtarı güncellemesi gerekir. Bunun yanı sıra ağda herhangi bir değişiklik yapıldığında (örneğin, ağa tek bir cihaz eklemek veya çıkarmak), her cihaz için yapılandırmada (ACL'ler, VLAN'lar, QoS, 5G, QoE) yer alan birçok adım manuel olarak yapıldığından daha uzun bir zamana ihtiyaç duyulur. Ayrıca manuel yapılandırma hataya açıktır.

Çizelge 2.2. (devam) Yazılım tanımlı ağ ve geleneksel ağ özelliklerinin karşılaştırılması

Performans	Gelen ağ resmine sahip olmasından dolayı merkezi kontrol sağlar. Farklı ağ katmanları arasında bilgi alış verişi olur.	Sınırlı bilgi, statik yapılandırma
Yenilik Ekleme	Yazılım uygulamalarıyla ağa yenilik eklemek oldukça kolay. Yeniliklerin testi için uygun ortam sağlar.	Ağa, yenilik eklemek için donanım eklenmesi oldukça zordur. Yeniliklerin testi için uygun ortam yoktur.
Donamım ve yazılım desteği	Açık kaynak yazılım kullanılarak yapılandırma	Ağ cihazları
Güvenlik	Merkezi hale getirilmiş kontrol mekanizması sayesinde güvenlik politikaları tek yerden bütün ağa kolayca uygulanır. Merkezi kontrolcü anlık ağ trafik bilgisine sahip. Anormal trafik tespiti kolay.	Fiziksel ağın güvenliği için güvenlik duvarı, proxy sunucuları kullanılır. Yeni uygulamaların ağ geneline uygulanması için güvenlik cihazlarında yapılan yapılandırma zordur. Zaman ve enerji kaybı olur. Üst düzey uzmanlık gerektirir.
Uygun ortam	Yazılım Tanımlı Ağ ile adapte olabilecek her ortam.	İstemci-sunucu mimarisine uygun ortam
Ölçeklenebilirlik	Merkezi kontrol ölçeklemeyi kolaylaştırır.	Yüksek karmaşıklık ölçeklemeyi zorlaştırır.
Maliyet	Yönetim ve hizmet maliyetleri düşük	Yönetim ve hizmet maliyetleri yüksek

Yazılım Tanımlı Ağ mimarisi, veri, kontrol ve uygulama düzlemi olmak üzere 3 düzlemden olmaktadır (Şekil 2.3).



Şekil 2.3. Yazılım Tanımlı Ağ mimarisi

Veri düzlemi: Anahtar, yönlendirici, kablosuz cihazlar, araçlar, Yol Kenarı Üniteleri gibi cihazlar ile kablolu veya kablosuz iletişim bağlantılarından oluşur [5]

Veri düzleminde bulunan iletim cihazlarının iki görevi vardır. Birinci görev, ağdaki trafik verisini toplamak, geçici olarak depolamak ve ardından bilgileri periyodik olarak kontrolcüye göndermek. Bu ağ veri bilgileri, ağ topolojisi, trafik istatistikleri ve ağın kullanımları hakkında bilgileri içerir. İkinci görev, paketleri işlemekten sorumludur. Paketler, kontrolcünün belirlediği akış kurallarına göre işlenir.

Kontrol Düzlemi: Ağın beyni olan kontrolcü, kontrol düzleminde bulunur. Ağ cihazlarından soyutlanarak harici bir cihaza taşınan kontrolcü ağı izlemek, hareketlilik yönetimi, kaynak tahsisi, ağ işlevlerini sanallaştırma, akış kuralı oluşturma ve yönetmekten sorumludur. Ağ kaynaklarını optimize ederek ağın performansında (trafik kontrolü, servis kalitesi, sıkışıklık kontrolü, verimli iletişim vs.) büyük bir iyileştirme sağlar. Kontrolcü, OpenFlow protokolü

aracılığıyla güvenli bir kanal vasıtasıyla veri düzleminde bulunan iletim cihazlarına yeni üst düzey kurallar ekleyip var olan kuralları da değiştirebilir. En çok kullanılan bazı kontrolcü tipleri Çizelge 2.3'de gösterilmiştir.

Çizelge 2.3. Yaygın olarak kullanılan bazı kontrolcü türleri

Kontrolcü	Mimari	Programlama Dili	Açık kaynak
NOX, SNAC, RouteFlow	Merkezi	C++, Python	Evet
POX	Merkezi	Python	Evet
RYU	Merkezi ve çoklu iş parçacıklı	Python	Evet
Floodlight	Dağıtık	Java	Evet
ONOS	Dağıtık	Java	Evet
Beacon	Merkezi ve çoklu iş parçacıklı	Java	Evet
OpenDaylight	Dağıtık	Java	Evet
Trema	Merkezi ve çoklu iş parçacıklı	Ruby ve C	Evet

Çizelge 2.3'de verilen kontrolcü tipleri arasında literatürde en çok kullanılanlar NOX, Floodlight, Ryu ve POX kontrolcüleridir. OpenFlow protokolü temelinde 2008 yılında Yazılım Tanımlı Ağ daha yeni geliştirilmeye başlandığında kullanılan ilk kontrolcü C++ dilinde programlanan NOX kontrolcüsüdür. Daha sonra Python tabanlı açık kaynak kodlu POX kontrolcüsü, NOX temel alınarak geliştirilmiştir. Python programlama dilinin sağlamış olduğu avantajlar sayesinde POX kontrolcüye yeni özellikler eklemek kolaydır. Java tabanlı OpenFlow destekli bütün ağ cihazlarda çalışabilen Floodlight ise, kurulumu kolay oldukça hızlı bir kontrolcüdür. Ryu, Yazılım Tanımlı Ağ uygulamaları için bileşen tabanlı bir çerçeve sunar. Ağ yönetimi ve kontrol uygulamaları oluşturmayı kolaylaştıran iyi tanımlanmış API'ye sahip yazılım bileşenleri sağlar. Python tabanlı bir platformdur ve Apache 2.0 lisansında her kod serbestçe kullanılabilir.

Uygulama Düzlemi: Uygulama düzlemi, ağ hizmet ve uygulamalarından oluşur. Ağ hizmet, servis ve uygulamaları veri düzleminde bulunan cihazlarla kontrolcü vasıtasıyla iletişim kurar. Uygulama düzleminde bulunan servisler kontrolcü ile Northbound ara yüzü üzerinden iletişim kurar. Northbound ara yüzü, çeşitli ağ uygulamalarını desteklemek ve optimize etmek için gereklidir. Uygulama düzlemi Northbound ara yüzü üzerinden kontrolcü vasıtasıyla veri düzlemiyle ilgili ağ topolojisi, akış istatistikleri, bağlantı bant genişliği ve ağın durumu gibi bilgileri sorgular. Ayrıca uygulama düzlemi, ağa üst düzey politikalar uygulamak, topoloji

değişikliklerine ve trafik gereksinimlerine yanıt vermek, veri düzlemi kontrol etmek için de northbound ara yüzünü kullanır [13].

Kontrolcü, veri düzleminde bulunan iletim cihazlarıyla Southbound ara yüzü üzerinden iletişim kurar. Southbound ara yüzünde en çok kullanılan protokol OpenFlow protokolüdür. OpenFlow'un önemli özelliklerinden birisi de piyasada bulunan, hâlihazırda kullanılan yönlendirici ve anahtar gibi ağ donanımlarına konuşlandırılabilmesidir.

Kontrolcü, Southbound ve Northbound açık ara yüzleri sayesinde karmaşık ve heterojen bir yapıya sahip alan ağları kolaylıkla yapılandırabilir.

2.2.1. OpenFlow protokolü

Open Network Foundation (ONF) tarafından geliştirilen OpenFlow protokolü Southbound ara yüzünde en yaygın kullanılan ilk ve en önemli açık kaynak iletişim protokolüdür.

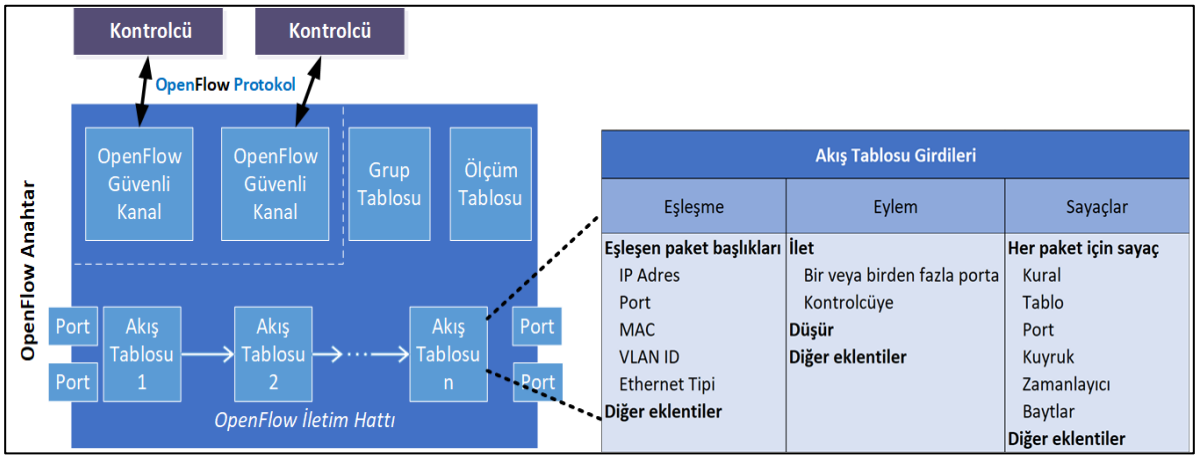
Yazılım Tanımlı Ağ mimarisinde kontrolcü ve veri düzleminde bulunan iletim cihazları arasındaki veri alış verişi güvenli bir kanal aracılığıyla OpenFlow protokolü vasıtasıyla gerçekleşir. OpenFlow, akış yönelimli bir protokoldür ve akışını kontrol etmek için anahtarlara ve bağlantı noktalarına sahiptir [14]. Protokol, Güvenli Soket Katmanı (Secure Sockets Layer-SSL) veya Taşıma Katmanı Güvenliği (Transport Layer Security-TLS) üzerinde uygulanarak güvenli bir iletim kanalı sağlar. Anahtar ve kontrolcü, özel bir anahtarla imzalanmış sertifikaları değiş tokuş ederek karşılıklı olarak kimlik doğrulaması yapar.

OpenFlow, Yazılım Tanımlı Ağ mantığının hem donanım hemde yazılım üzerinde uygulanmasına olanak tanır. Piyasada bulunan birçok firma artık OpenFlow özellikli yönlendirici ve anahtar üretmektedir.

Yazılım tanımlı ağ mimarisinde kullanılan OpenFlow gibi standartlaşmış protokoller farklı satıcılardan gelen cihazlar arasındaki iletişimi kolaylaştırır. Veri düzleminde bulunan cihazların yönetiminde ve ağın izlenmesinde kolaylık sağlar, programlanabilir ve esnek ağ mimarisi tasarımını kolaylaştırır [15].

OpenFlow yönlendirici ve anahtarları kontrol etmek için açık kaynak kod kullanılır. OpenVSwitch (OVS) en popüler, yazılım tabanlı OpenFlow anahtarlarından biridir. OpenFlow

mimarisinde, OpenFlow özellikli yönlendirici ve anahtar gibi cihazlar, bir veya daha fazla akış tablosu ve OpenFlow protokolü aracılığıyla bir kontrolcü ile güvenli bir şekilde iletişim kuran bir soyutlama katmanı içermektedir. Anahtar veya yönlendiriciler akış tablosu, grup tablosu ve ölçüm tablosu (meter) olmak üzere üç tür tablo içerir (Şekil 2.4).



Şekil 2.4. OpenFlow Anahtar

Akış tablosu her bir akışa ait paketlerin nasıl işleneceğini ve ileteceğini belirleyen akış girişlerinden oluşur. Anahtarda veya yönlendiricide ardışık düzende birden fazla akış tablosu olabilir. Bir akış tablosu, bir veya birden fazla akışı etkileyen farklı eylemleri tetikleyen grup tablosuna yönlendirilebilir. Ölçüm tablosu (meter table) bir akıştaki performans ile ilgili eylemleri tetikler. Her tablo ve bağlantı noktası, anahtarın maruz kaldığı olayları açıklayan çeşitli istatistikleri toplayabilen birçok sayaçla ilişkilendirilir. Kontrolcü tüm tabloları ve bunların girişlerini oluşturur; OpenFlow anahtarından geçen veri paketleri sayaçları günceller. Anahtara gelen her paket bir veya birden fazla akış tablosundan geçer [16]. Akış tabloları, her biri bir akışa ait paketlerin nasıl işleneceğini ve iletileceğini belirleyen yedi bileşenden oluşan girişlere sahiptir (ver. 1.5).

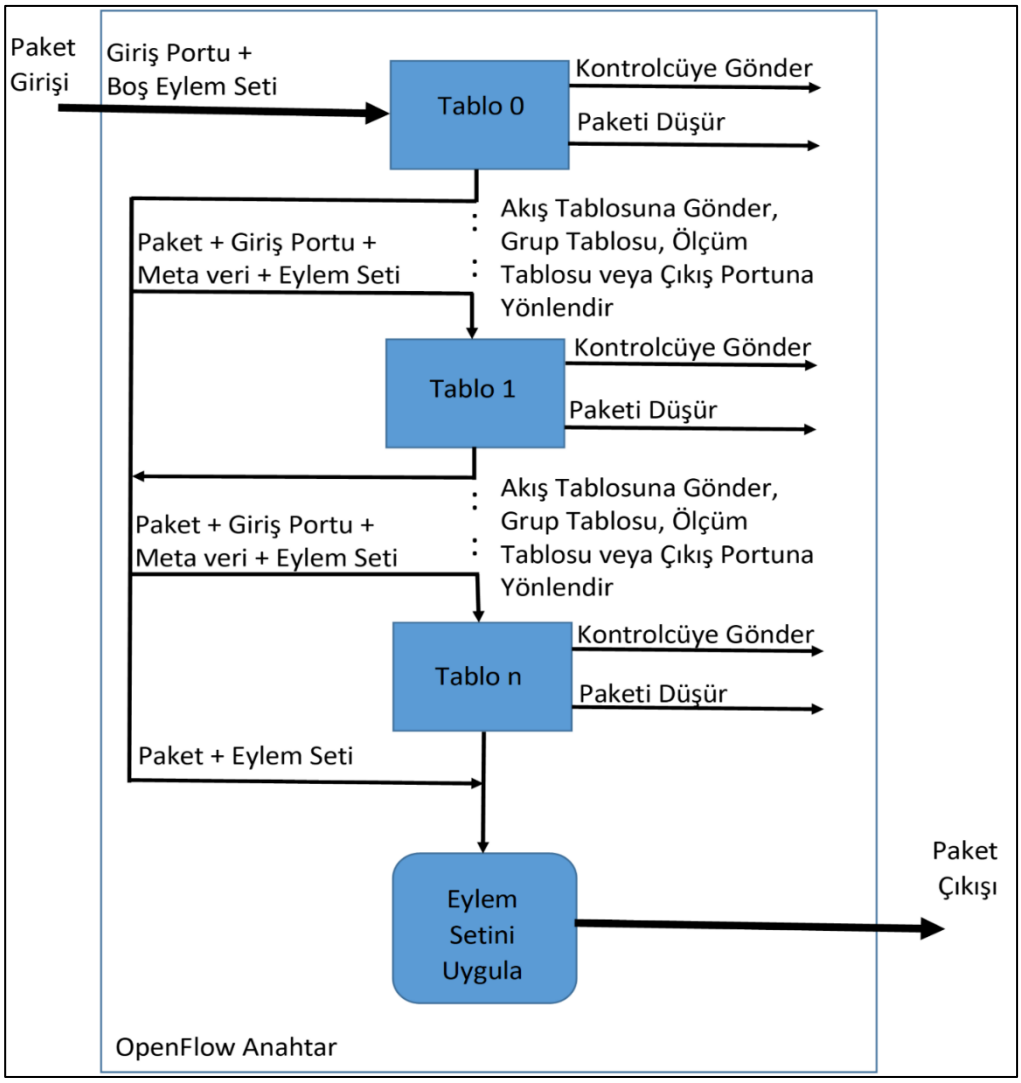
1. Eşleşme Alanı (Match Fields): Alanlardaki değerlerle eşleşen paketleri seçmek için kullanılır. Giriş portu, paket başlığı ve bir sonraki tabloya bilgi taşımak için kullanılan önceki tablodan kalmış ve maskelenmiş kayıt değerlerinden oluşur
2. Öncelik (Priority): Tablo girişlerinin göreli önceliği. Akış girdisinin eşleştirme önceliği ve sıralanmasını sağlar. Eylemi belirlemek için daha yüksek önceliğe sahip olan eşleştirme kullanılır.

3. Sayaçlar (Counters): Eşleşen paketler olduğunda güncellenir. Alınan paket sayısı, bayt sayısı ve akışın süresi gibi belirli bir akış için istatistikleri toplar. Bu veri ağdaki ani trafik değişimlerinin tespitinde büyük fayda sağlar.
4. Komutlar (Instructions): Gelen paketlerin nasıl işleneceği, eşleşme olduğunda pakete uygulanacak işlemleri (paketi kontrolcüye gönderme/hedef porta iletme/düşürme vs.) içerir.
5. Zaman aşımları (Timeouts): Anahtar tarafından bir akışın tablodan çıkarılmadan önceki maksimum boşta kalma süresi.
6. Çerez (Cookie): Kontrolcü tarafından seçilen opak veri değeri. Kontrolcü tarafından akış istatistiklerini, akış değiştirmeyi ve akış silmeyi filtrelemek için kullanılabilir.
7. Bayraklar (Flags): Akış girdilerinin yönetilme şekillerini belirler.

Akış tablosu iletim hattı

Şekil 2.5'de görüldüğü üzere bir anahtar, bir veya birden fazla akış tablosuna sahip olabilir. Akış şemasında Openflow anahtarından geçen paketlerin akışı gösterilmektedir (Şekil 2.5).

Bir anahtarda birden fazla akış tablosu mevcut ise, eşleştirme ilk akış tablosuyla başlayıp iletim hattıyla boyunca birbirine ardışık bir düzende bağlı olan akış tabloları üzerinden artan sayılarla ilerler. Ağa veri girişi gerçekleştiğinde öncelikli olarak anahtarın ilk akış tablosu olan Tablo 0' a gelir. Bir anahtar, paket araması yapan ve paket iletimini gerçekleştiren birden fazla akış tablosuna sahip olabilir. Anahtarlardaki akış tabloları eşleşen paketlere uygulanacak kurallar, eşleşme alanları, sayaçlar gibi akış girdilerinden oluşur. Tablo 0'da meta veri ve eylem kümesi boştur. Bir paket iletim hattı boyunca şu şekilde ilerler:



Şekil 2.5. OpenFlow anahtardaki paket akış süreci

Ağa gelen paket için en yüksek önceliğe sahip akış girişi aranır. Akış için herhangi bir eşleşme ve "table miss" (bir paketin, bir akış tablosundaki bir akış girişiyle eşleşmemesi) akış girdisi yoksa paket düşer. "Table miss" girişinde eşleşme olması durumunda ise giriş için üç eylemden biri gerçekleşir:

- Paket kontrolcüye gönderilir. Kontrolcü bu paket için yeni akış kuralı oluşturabilir veya paketin düşmesini isteyebilir.
- Paket iletim hattında bulunan başka bir akış tablosuna yönlendirilebilir.
- Paket direk düşürülebilir.

Anahtar, yanlış akış girişi, eşleşmeyen akış, paket düşürme ve kontrolcüye gönderme gibi yapılandırmaları anahtardaki akış tablosunda bulunan kurallara göre gerçekleştirir. Ağa gelen paket

için ilk akış tablosunda eşleştirme alanları aranır. Eşleşme sağlanırsa kontrolcünün önceden belirlediği kuralara göre paket işlenir. Paket tabloyla eşleştirilirse, yalnızca en yüksek önceliğe sahip giriş seçilmelidir. Bundan sonra aşağıdaki talimatlar uygulanmalıdır:

- Girişle ilişkili tüm sayaçlar güncellenmeli.
- Girişle ilişkili eylem seti güncellenmeli. Meta veri değerinin güncellenmesi ve eylemlerin gerçekleştirilmesi gibi bütün talimatlar gerçekleştirilmeli.
- Paket iletim hattındaki bir akış tablosuna, grup tablosuna veya sayaç tablosuna iletilmeli veya bir çıkış bağlantı noktasına yönlendirilmeli.

Her paketle bir eylem seti ilişkilendirilir. Bu eylem seti kümesi varsayılan olarak boştur. Bir akış girişi, bir eylem yazma talimatı veya belirli bir eşleşmeyle ilişkili bir temizleme talimatı kullanarak eylem setini değiştirebilir. Eylem seti akış tabloları arasında taşınır. Bir akış girişinin komut seti tabloya git talimatı içermediğinde, işlem hattı boyunca ilerlemesi(paketin işleme alınması) durur ve paketin eylem setindeki eylemler yürütülür [16].

Ardışık düzendeki son tablo için başka bir akış tablosuna iletme bir seçenek değildir. Bir paket nihayet bir çıkış bağlantı noktasına yönlendirilirse ve yönlendirildiğinde, birikmiş eylem seti yürütülür ve ardından paket çıktı için kuyruğa alınır.

OpenFlow protokolü güvenli kanal vasıtasıyla veri düzleminde bulunan anahtar, yönlendirici gibi cihazlarla kontrolcü arasındaki iletişimi sağlar. Kontrolcü bu kanal aracılığıyla cihazlarda bulunan akış tablolarının girişlerine ekleme, silme ve güncelleme gibi işlemler yapar. Kontrolcü veri düzleminde bulunan cihazlar arasında 3 farklı mesaj türü ile akış sağlanır (Çizelge 2.4).

Kontrolcü ile veri düzleminde bulunan iletim cihazları (anahtar, yönlendirici vs.) arasındaki mesajlar: Kontrolcü tarafından başlatılır ve bazı durumlarda anahtarın yanıt vermesini gerektirir. Bu mesaj tipi, kontrolcünün anahtarları yapılandırması ve akış ve grup tablosu girişlerinin ayrıntıları dâhil olmak üzere anahtarın yönetilmesini için gerekli bütün bilgileri/kuralları barındırır.

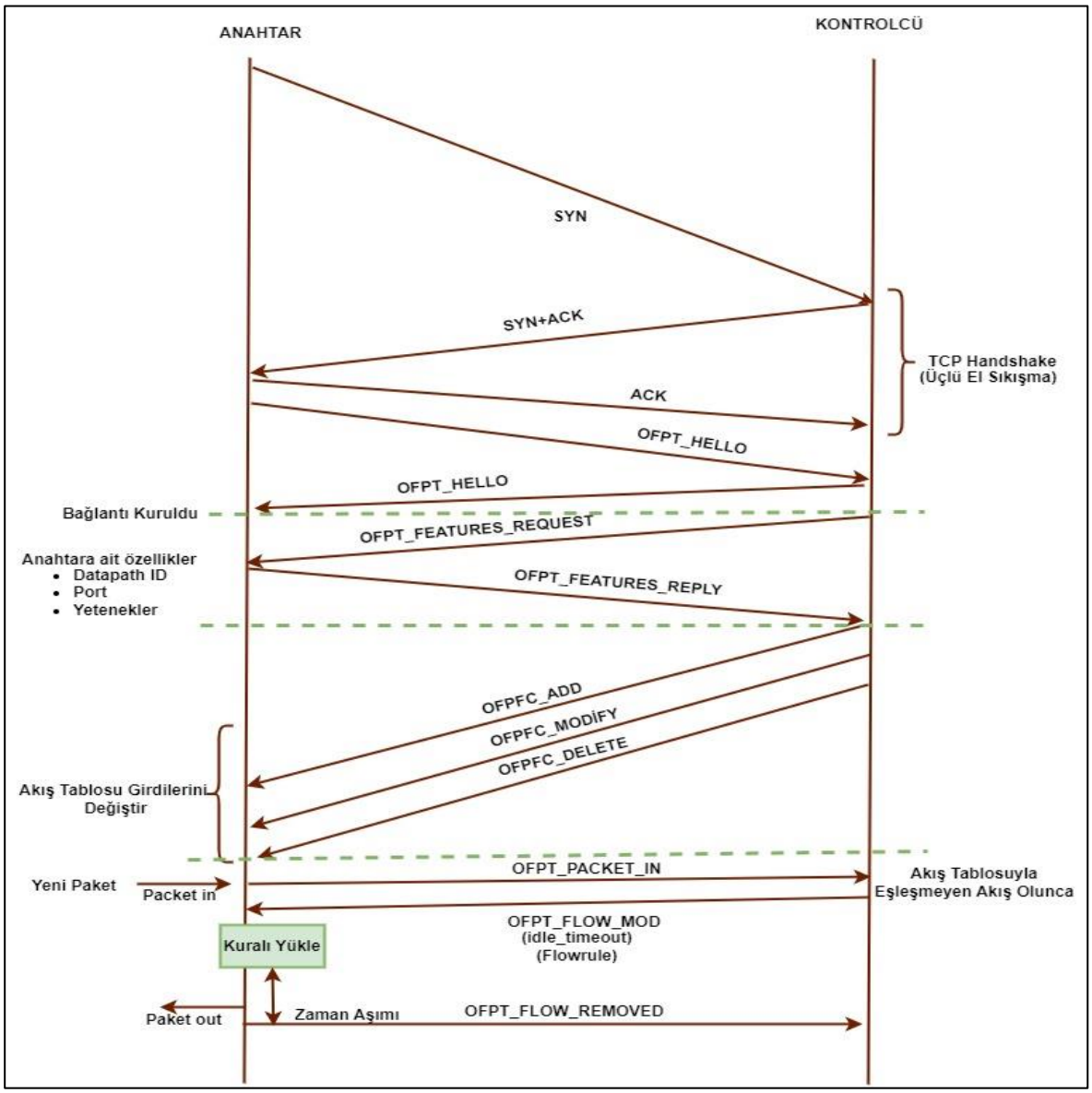
Simetrik mesajlar: Bu mesajlar kontrolcü veya anahtar tarafından talep edilmeksizin karşılıklı gönderilir.

Asenkron mesajlar: Bu mesaj kontrolcünün talebi olmaksızın gönderilir. Kontrolcüye gönderilen durum mesajlarının (eşleşmeyen akış girdisi, Packet_ in mesajı vs.) içerir.

Çizelge 2.4. OpenFlow mesajları

Kontrolcü ve veri düzleminde bulunan iletim cihazları arasında akan mesajlar	
Mesajın adı	Tanımı
Features	Kontrolcü bir anahtara ait anahtara kimliği ve özellikleri hakkında bilgi mesajı gönderir. Anahtar ise kimliği ve temel özelliği ile ilgili bilgileri mesajla kontrolcüye gönderir.
Configuration	Kontrolcü, anahtardaki yapılandırma parametrelerini ayarlar ve denetler. Anahtar, kontrolcüye parametre ayarlarıyla ilgili bilgi verir.
Modify-State	Anahtardaki akış ve grup girişlerini ekleyin, silin ve değiştirin ve anahtar bağlantı noktası(port ayarları) özelliklerini ayarlayın.
Read-State	Anahtardan geçerli/güncel yapılandırma, istatistikler ve kabiliyetleri hakkında bilgi istenir.
Packet-out	Kontrolcü, paketi anahtardaki belirli bir bağlantı noktasına(port) yönlendirir. Kontrolcü, packet_out mesajlarını kullanarak harici bağlantı noktalarına düzenli olarak sorgulama iletir. Bu sorgulama mesajları, kontrolörün ağın topolojisini oluşturmasını sağlayan packet_in mesajları şeklinde geri döner.
Barrier	Kontrolcü tarafından mesaj bağımlılıklarının karşılandığından emin olmak veya tamamlanan işlemler için bildirim almak için gönderilir.
Role-Request	OpenFlow kanalının rolünü ayarlar veya kontrolcünün kimliğini sorgular. Anahtar birden çok kontrolcüye bağlandığında daha kullanışlıdır. Gereksizlikler bu sayede ortadan kalkar.
Asynchronous Configuration	Eşzamansız iletilerde filtre ayarla veya bu filtreyi sorgula. Anahtar birden çok kontrolcüye bağlandığında daha kullanışlıdır.
Simetrik Mesajlar	
Mesajın adı	Tanımı
Hello	Bağlantı başlatıldığında anahtar ve kontrolcü birbirine gönderir.
Echo request/reply	Anahtardan veya kontrolcüden gönderilerek iki nokra arasındaki bağlantının olup olmadığı kontrol edilir. Bant genişliği ve gecikmenin ölçülmesi içinde kullanılır.
Experimenter	Ek işlevler için kullanılır. OpenFlow protokolüne yenilik eklendiğinde uygun olanak sağlar.
Asenkron mesajlar	
Mesajın adı	Tanımı
Packet-in	Akış girdileriyle eşleşmeyen paketler olduğu zaman yeni akış kuralı oluşturması için paket başlıkları bu mesajla kontrolcüye gönderilir.
Flow-Removed	Akış tablosunda bulunan bir akış girişi kaldırıldığında kontrolcüye bilgi verir.
Port-Status	Bir bağlantı noktasındaki (port) değişikliği veya yapılandırma bilgisini kontrolcüye gönderir.
Error	Kontrolcüye, anahtar tarafından hata veya sorun durumunda bilgilendirme amaçlı gönderilir.
Flow-monitor	Akış tablosu değişik meydana gelince kontrolcüyü bilgilendirir
Controller Status	İletim kanalının statüsü değişince kontrolcüyü bilgilendirir.

Her anahtar, kontrolcünün IP adresi ve TCP port numarasına göre yapılandırılır. Ağa dahil olan her anahtar, kontrolcü ile iletişimi başlatmak için üç yönlü bir el sıkışma (SYN, SYN/ACK, ACK) kullanarak bir TCP oturumu oluşturur (Şekil 2.6).



Şekil 2.6. OpenFlow anahtar ile Yazılım Tanımlı Ağ kontrolcü arasındaki bağlantı kurulum işlemleri

Daha sonra kontrolcü, anahtara bir "OFPT_FEATURES_REQUEST" mesajı göndererek anahtarın özelliklerini (MAC adresi, bağlantı noktaları ile ilgili bilgileri, eylem seti, akış tablosu vs.) talep eder. Anahtar istenen bilgileri içeren bir "OFPT_FEATURES_REPLY" mesajı ile kontrolcüye yanıt verir. Kontrolcü, anahtardan aldığı bu bilgileri topolojinin oluşturulması, anahtara yeni kural yazılması gibi işlemler için saklar veya kullanır. Kontrolcü ve anahtar bu şekilde birbirleriyle bağlantı sağlar.

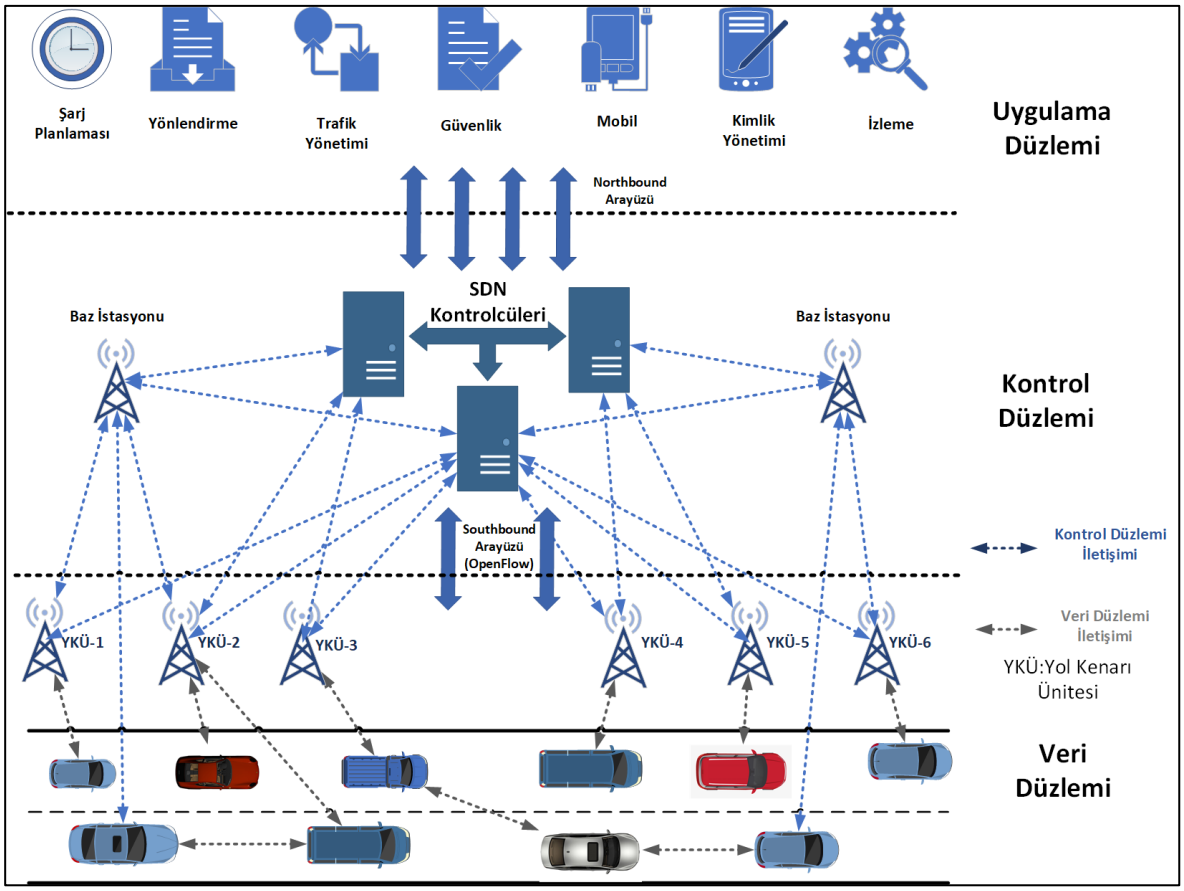
Gelen her paket için OpenFlow anahtar, paket bilgilerinin akış tablosundaki bir akış girişiyle eşleşip eşleşmediğini kontrol eder Akış tablosunda herhangi bir eşleme yok ise anahtar bu paketi kapsüller ve “OFPT_PACKET_IN” mesajıyla kontrolcüye gönderir. Bu mesaj giriş portu, paket başlığı ve paketin depolandığı Buffer_ID bilgilerini içerir. “OFPT_PACKET_IN” mesajına yanıt vermek için kontrolcü bir “OFPT_PACKET _OUT” mesajı gönderir. Bu mesaj, ilgili “OFPT_PACKET_IN” mesajının Buffer_ID'sini ve gerçekleştirilecek eylemleri (örneğin, belirli bir bağlantı noktasına iletme, bırakma vb.) içerir. Aynı akışın sonraki paketlerini işlemek için kontrolcü, yönlendirme eylemlerine karar verir ve gelen mesaja “OFPT_FLOW_MOD” (akış taplosunda akış girişleri değiştir/ekle) mesajıyla cevap verir.

“OFPT_FLOW_MOD” mesajını alan anahtar, akış tablosuna yeni bir giriş yükler ve bununla eşleşen paketleri hesaplamaya başlar. Ekleme mesajı, eşleşme alanlarını ve kuralın akış tablosunda ne kadar kalacağını belirleyen zaman aşımlarını (HARD_TIMEOUT ve IDLE_TIMEOUT) içerir. “HARD_TIMEOUT”, aktif akışlar durumunda bile bir girişin akış tablosunda ne kadar süre kalması gerektiğini belirler. “IDLE_TIMEOUT”, anahtara son eşleşen paket görüldükten sonra girişin akış tablosunda ne kadar süre kalması gerektiğini söyler [17].

2.3. Yazılım Tanımlı Araçsal Tasarsız Ağlar

Yazılım Tanımlı Araçsal Tasarsız Ağlar (Software Defined based Vehicular Ad Hoc Networks-SD-VANET), Yazılım Tanımlı Ağ, teknolojisinin VANET’e entegre edilmesiyle ortaya çıkmış gelişmekte olan bir paradigmadır. Aynı cihazda bulunan kontrol ve veri düzlemlerinin ayrılması, kontrol mantığının harici bir cihaza aktarılması VANET ağlarında esneklik ve programlanabilirlik sağlar. Yazılım Tanımlı Ağ ile entegre olan VANET ağı, basitleşir ve ağ kaynakları verimli bir şekilde yönetilir. SD-VANET mimarisi, kontrolcü, OpenFlow özellikli iletim cihazları (anahtar, Yol Kenarı Ünitesi, Access Point) ve kontrolcü denetimi altında olan araçlardan oluşur. Kontrolcü denetiminde olan veri düzlemindeki araçlar, kontrolcü tarafından oluşturulan akış kurallarına göre hareket eder [18].

SD-VANET ağ mimarisi, Araç Üstü Ünitesi donanımlı mobil düğümlerin (araçlar) ve OpenFlow özellikli sabit düğümlerin (Yol Kenarı Ünitesi, Access Point) bulunduğu veri düzlemi, ağ kontrolcüsünün bulunduğu kontrol düzlemi ve araçsal ağ uygulamalarının bulunduğu uygulama düzlemi olmak üzere 3 düzlemden oluşur (Şekil 2.7).



Şekil 2.7. Yazım tanımlı araçsal tasarsız ağ mimarisi

Veri düzlemi: Veri düzlemi, Yol Kenarı Üniteleri, Access Point (AP), anahtar gibi veri iletim cihazlarından oluşur. Yol Kenarı Üniteleri ve AP'ler, OpenFlow özellikli olup LTE iletişim standardı ile uyumlu bir ara yüze de sahiptirler [18].

Yol Kenarı Üniteleri, gerektiğinde verileri depolayan ve sağlayan bir erişim noktası, yönlendirici veya bir geçiş noktası olarak da görülebilir. Veri düzleminde bulunan iletim cihazları yaygın olarak kullanılan haberleşme protokolü olan OpenFlow protokolü ile Southbound ara yüzünü kullanarak kontrol düzleminde bulunan kontrolcüyle iletişim kurar. Kontrolcü, OpenFlow protokolü aracılığıyla güvenli bir kanal vasıtasıyla veri düzleminde bulunan iletim cihazları yönetilir. Ayrıca veri düzleminde bulunan düğümler üzerinde, kontrolcü ile bağlantının kopması durumunda iletişimin devam etmesi için bir "fallback" mekanizması mevcuttur. Bu mekanizma, aracın geleneksel VANET ağlarındaki iletişim özelliklerine göre iletişimini sürdürmesini sağlar.

Araçların ihtiyaç duyduğu ani durumlarda her yerden erişilebileceği gerçek zamanlı veri trafiği Yol Kenarı Ünitelerinde saklanır. Elde edilen gerçek zamanlı trafik verisi, trafik sıkışıklığı ve

tıkanıklığı gibi sorunları çözmede büyük fayda sağlar. Yol Kenarı Üniteleri, V2I iletişimi kullanarak, meydana gelen kazalar, çarpışmalar için de uyarılar üretir. Yol Kenarı Üniteleri, araçların konumunu, hızını içeren durumlarını kontrolcüye gönderir. Kontrolcü bu durumları korur, aldığı bu durumlar sayesinde verimli yönlendirme kararları alarak araçlara yardımcı olur.

Yol kenarlarına yerleştirilen sabit Yol Kenarı Üniteleri düğümlerine yerleştirilecek görüntüleme sistemleri sayesinde sürücülerin trafik ihlalleri denetlenerek, trafik suçları minimuma indirilebilir.

Kontrol Düzlemi: Ağın kontrol ve yönetiminden sorumlu olan, ağın beyni konumundaki kontrolcü bu düzlemde yer alır. Yol Kenarı Üniteleri ve kablosuz erişim noktaları aracılığıyla kontrolcü, veri düzleminde bulunan düğümlere veri akışının nasıl iletileceği ve veri akışının nasıl düzenleneceğine dair akış kuralları gönderir. Veri düzleminde bulunan düğümler ağdaki trafik akışı bilgilerini periyodik olarak kontrolcüye iletir. Periyodik olarak ağ trafiği ile ilgili bilgi alan kontrolcü, ağ trafiğindeki ani değişimlere tepki vererek ağdaki bütünlüğün bozulmaması için politikalar uygular. Ağı merkezi bir kontrolcü ile yönetmek; toplam yükü azaltmaya, güvenlik tehditlerini minimuma indirmeye ve sistem verimliliğini arttırmaya olanak sağlar. Ayrıca, karmaşık ağ fonksiyonlarını, servislerini ve uygulamalarını geliştirmeyi kolaylaştırır. Kablosuz ağlarda ve kablosuz iletişimde Yazılım Tanımlı Ağdaki kontrol mantığı, kaynakların optimizasyonu, paket yönlendirmesi ve iletimi, trafiğin etkili ve verimli bir şekilde hareketinde büyük fayda sağlar [19]. Birden çok kontrolcünün ağda kullanılmasıyla, kontrolcünün saldırı anında gerçek zamanlı yanıt verme hızı artırılarak saldırılara etkin bir şekilde yanıt verilebilir. Birden çok kontrolcü kullanmanın bir diğer avantajı da kontrolcünün Yol Kenarı Üniteleri ile veri alışverişi süresinin artmasıdır. Bu da, kontrolcünün Yol Kenarı Üniteleri üzerindeki kontrolünü ve veri aktarım hızını artırır [20]. Özellikle SD-VANET ağ yapılarında, birden çok kontrolcünün hiyerarşik bir yapıda kullanılması ve kontrolcünün ağdaki konumu, kontrolcünün veri katmanında yer alan ağ öğeleri üzerindeki denetim rolünü artırarak kaynak kullanımını optimize ederek gecikmeyi azaltır.

Uygulama Düzlemi: Uygulama düzleminde, SD-VANET mimarisinde haberleşme ve ağ servislerini kullanan yönlendirme, işletme uygulamaları, güvenlik, mobility, yük dengeleme, araç konumu tahmini, şüpheli ağ etkinliği tespiti, sanallaştırma, izleme gibi son kullanıcı uygulamaları bulunur. Uygulamalar, arabadan altyapıya uygulamaları, arabadan arabaya trafik

uygulanmaları, arabadan eve uygulamalar ve yönlendirme tabanlı uygulamalar olmak üzere 4 temel başlık altında sınıflandırılabilir.

Kontrolcü, Northbound ara yüzünü kullanarak uygulama düzleminde bulunan araçsal ağ ve Yazılım Tanımlı Ağ uygulama yazılımlarıyla iletişim kurar. Northbound ara yüzü, kontrolcünün verimli ve hızlı bir şekilde ağı düzenlenmesini, ağa uygulanacak yeni politikalarının belirlenmesini ve bu politikaların uygulanabilir hale getirilmesini sağlar.

SD-VANET kontrolcüsü merkezi kontrol modu, dağıtılmış kontrol modu ve hibrit kontrol modu olmak üzere 3 farklı kategoride sınıflandırılır.

Merkezi Kontrol Modu: Bu modda, OpenFlow özellikli cihazlar (Yol Kenarı Ünitesi, anahtar vs.) ve kablosuz düğümler Yazılım Tanımlı Ağ kontrolcüsü tarafından kontrol edilir. Ağdaki bütün veri akışını sağlayan kurallar Yazılım Tanımlı Ağ kontrolcüsü tarafından belirlenir. Veri düzleminde bulunan bütün iletim cihazları bu kurallara göre hareket eder.

Dağıtılmış Kontrol Modu: Bu modda, veri düzleminde bulunan iletim cihazları (Yol Kenarı Ünitesi, anahtar, araç vs.) Yazılım Tanımlı Ağ kontrolcüsünün denetiminde değildir. Yazılım Tanımlı Ağ özellikli olmayan GPRS yönlendirme protokolü altında çalışan, kendi kendini organize eden bir ağa benzer [21]. SD-VANET ağ yapısında kontrolcü, veri düzleminde bulunan yönlendirme cihazlarının yanı sıra hücresel baz istasyonlarını, radyo erişim ağını, Yol Kenarı Ünitelerini ve akıllı araçları yönetmek zorundadır. Ayrıca 5G, bulut, sis bilişim ve kenar unsurlar ağa entegre edilmektedir. Bilgi ve iletişim teknolojilerinin gereksinimlerine göre büyüyen SD-VANET ağ yapısında meydana gelebilecek zorlukların üstesinden gelmek için dağıtılmış Yazılım Tanımlı Ağ kontrolcü yapısının kullanılması gerekir.

Hibrit Kontrol Modu: Bu mod, merkezi ve dağıtılmış kontrol modlarının tüm özelliklerini içerir. Bu modda ağın bütün kontrolü kontrolcünün elinde değildir. Yazılım Tanımlı Ağ kontrolcü, ağ akış kurallarını oluşturarak veri düzleminde bulunan iletim cihazlarına gönderir. Veri düzleminde bulunan sabit düğümler (Yol Kenarı Ünitesi) ve kablosuz hareketli düğümler (araç) kontrolcünün gönderdiği akış kurallarını kendi zekâlarını kullanarak akış kurallarını uygulayıp yönlendirme kararı alır. Hibrit kontrol modu kablosuz ağlar için daha idealdir. SD-VANET, kablosuz yapısından kaynaklı olarak dinamik bir yapıya sahiptir. MANET dahil olmak üzere diğer birçok geleneksel ağın aksine, VANET düğümleri farklı hızlarda hareket

eder. Bu, belirli bir topolojinin oluşumunun sürekli değiştiği ve kalıcı komşuların olmadığı anlamına gelir [22]. Bu hareketlilik veri düzleminde düğümler ile kontrolcü arasındaki bağlantının kopmasına neden olabilir. Bu gibi durumlarda düğümler de bulunan akıllı sistem sayesinde düğümler geleneksel yönlendirme protokolleri altında çalışarak komşu düğümlerle haberleşir. Bu akıllı sistem sayesinde haberleşen düğümlerden alınan veriler daha sonra kontrolcüye iletilir. Kontrolcü bu bilgiler doğrultusunda ağ kurallarını daha efektif bir şekilde uygular. Bu mekanizma kontrolcü ile düğümler arasındaki iletişimin kopması durumunda ağ bütünlüğünün bozulmasının önüne geçer.

2.3.1. SD-VANET mimarisinin avantaj ve dezavantajları

Bilgi ve iletişim teknolojisinin gelişmesiyle birlikte, geleneksel VANET ağı mimarisi, akıllı araç sistemlerinin talep ettiği ölçekleme, esneklik, güvenlik, verimli mesaj iletimi gibi gereksinimleri karşılayamaz hale gelmiştir. Yazılım Tanımlı Ağ gibi esnek, programlanabilir ve dinamik mimariye sahip yeni teknolojilerin entegrasyonuna ihtiyaç duyulmuştur. Yazılım Tanımlı Ağ, yönetilebilirlik, ölçeklendirme ve performansa ilişkin birçok avantajına rağmen, güvenlik açısından bazı dezavantajları vardır. Yazılım Tanımlı Ağ tabanlı ağların geleneksel ağ yapısına göre bazı avantaj ve dezavantajları aşağıda özetlenmiştir:

SD-VANET ağının avantajları

- Geleneksel ağ yapısında kullanılan uygulamaların sürekli geliştiği ve yenilendiği bir ortamda altyapının da bu yenilikleri karşılaması gerekmektedir. Ancak mevcut ağ yapısında yeni teknolojilerin kullanılması birçok sorunu da beraberinde getirmektedir. Mevcut ağ yapısında inovasyonun önündeki en büyük engel, ağ altyapısında çeşitli satıcılar tarafından üretilen farklı arayüzlere sahip cihazların kullanılmasıdır [23]. Yazılım Tanımlı Ağ ile entegre olmuş bir geleneksel ağ yapısında, yönlendirme benzeri ağ uygulamalarının donanımdan izole edilmesi esneklik sağlar ve satıcı bağımlılığını ortadan kaldırır.
- Bilgi ve iletişim teknolojisinin hızlı bir gelişim göstermesinden dolayı geleneksel ağ alt yapısında birçok teknoloji birlikte çalışmaktadır. Farklı teknolojilerin bir arada kullanılması ağ performansını optimize edilmesini zorlaştırır [24]. Mevcut yapıdaki ağ performansını iyileştirmek için farklı yaklaşımlar denenmiş ancak başarılı olunamamıştır. Yazılım Tanımlı Ağda bulunan, ağ alt yapısına hâkim merkezi kontrolcü sayesinde ağ katmanları arasında bilgi alışverişi sağlanır [13]. Yazılım Tanımlı Ağ mimarisinin bu özelliğinden yararlanılarak veri trafiği planlaması, uçtan uca tıkanıklık kontrolü, yük dengeli paket

yönlendirme, verimli çalışma, farklı ara yüzlerin seçilmesi, hizmet kalitesi gibi optimizasyon sorunları Yazılım Tanımlı Ağ teknolojisinin geleneksel ağlara entegre edilmesiyle çözülebilir.

- Geleneksel ağ yapısının boyutunun artması heterojen ve karmaşık bir yapıya neden olmaktadır. Geleneksel yaklaşımlar, heterojen ve karmaşık bir sisteme entegre olan yeni teknolojilerde hata ayıklama, optimizasyon ve yapılandırma yapmayı zorlaştırır. Yazılım Tanımlı Ağ yapısının başlıca avantajlarından biri de, veri ve kontrol düzlemlerini ayırarak ağı yöneten merkezi kontrolcüdür. Bu özellik yapılandırmayı, performansı artırmayı ve ağa yenilik eklemeyi kolaylaştırır.
- Kontrolcü tarafında trafik log kayıtları kaydedilir. Bu kayıtlar ağ adli analizinde kullanılır. Bir siber saldırı durumunda saldırının nasıl yapıldığı, saldırganların cihaz yapılandırmaların da değişiklik yapıp yapmadığı, trafik akış ve verilerinden oluşan bu kayıtlar incelenerek tespit edilebilir. Yapılan ağ adli analizi sonucunda elde edilen veriler ile saldırı hakkında bilgi edinilir. Elde edilen bilgilerle siber saldıra karşı etkili bir savunma mekanizması geliştirilebilir.
- Yazılım Tanımlı Ağı geleneksel ağlardan ayıran önemli özelliklerinden biri de kendi kendini iyileştiren bir mekanizma içermesidir. Kurallar kontrolcü tarafından ağdaki bütün iletim cihazlarına tanımlanır. Kontrolcü tarafından oluşturulan kuralda belirlenen koşul sağlandığında kural aktif hale gelir. Örneğin, bir saldırı olması durumunda, bu kurallar iletim cihazına, kötü amaçlı olarak tanımlanan paketin düşürülmesi gibi saldırıyı önlemenin bir yolunu sunar [15]. Kendi kendini iyileştirme mekanizması sayesinde ağdaki kötücül trafik tespit edilerek, siber saldırıların sisteme zarar vermesi engellenir.
- Geleneksel VANET ağlarında güvenlikle ilgili ve güvenlikle ilgili olmayan uygulamaların farklı gecikme, yayılma ve bant genişliğindeki kanalların frekansının seçilmesi gibi gereksinimleri vardır. Trafikteki yoğunluk, öncelikli araç geçişi gibi durumlarda dinamik olarak kanal tahsisi için araçta ek uygulamalara ihtiyaç duyulmaktadır. Kontrolcü, veri düzleminde bulunan araçlar üzerindeki farklı arabirimlerden gelen akışlara göre, kanal ve frekans seçimini verimli, dinamik bir şekilde seçme yeteneğine sahiptir. Yazılım Tanımlı Ağ teknolojisinin, kablosuz heterojen ara yüzleri desteklemesi maliyetli, hesaplama, performans, depolama gibi konularda avantaj sağlar. Ayrıca trafik yoğunluğuna göre araçlardaki bu iletişim ara yüzlerinin gücü ayarlanıp iletişim aralığı değiştirilebilir [15]. Ağ global görüntüsüne sahip kontrolcü veri düzlemindeki iletim cihazlarından aldığı anlık verilere sayesinde tüm ağa hakim olur. Kontrolcü aldığı bu bilgiler sayesinde ağdaki

kullanıcılara kapsama alanı, bant genişliği ve maliyete göre en iyi kablosuz ara yüzü sunar. Kontrolcünün ağı yönetmesi kolaylaşır. Ayrıca ağ performansı artar.

- Geleneksel kablosuz ağlardaki heterojen ve dinamik yapıdan dolayı kullanılan yönlendirme mekanizmalarından yüksek verimlilik elde edilmez. Yönlendirme prosedürü, düğüm durumu alışverişi, rota seçimi, rota bakımı ve onarımından oluşur. Her düğüm, bir yönlendirme tablosu oluşturmak için kendi aralarında veri alış verişinde bulunur [17]. Hareketli bir yapıya sahip düğümlerden oluşan ağ yapısında sürekli topoloji değişir. Bu hareketli yapıdan kaynaklı araçlar ağın global yapısından haberdar olmaz. Araçlar arasındaki veri alış verişinde hareketli düğümlerden kaynaklı, veri aktaran kaynak araç hedef araca veri iletirken herhangi bir aktarma düğümünün yolunu değişip değişmediğini öğrenemez. Bu durum yüksek paket kaybına ve ağda tıkanıklığa neden olur. Ayrıca IP tabanlı yönlendirme protokolleri, ölçeklenebilirlik, bağlantı problemlerine yol açar. VANET ağında rota, araçlar veya Yol Kenarı Ünitesi tarafından üretilen ve veri düzleminde bulunan cihazlara yönlendirilen her periyodik uyarı mesajı için oluşturulur. Bu mekanizma yönlendirme ek yükü, bant genişliği israfı ve verimsiz kanal kullanımına neden olur. Yazılım Tanımlı Ağ mimarisinde kontrolcü ağ cihazlarından periyodik olarak aldığı anlık trafik verisi sayesinde veri düzleminde bulunan cihazlar ile ilgili detaylı bir bilgiye sahiptir. Veri düzlemindeki iletişim cihazlarından (araç, Yol Kenarı Ünitesi) gelen ilk uyarı mesajı ile kontrolcü yeni akış kuralı oluşturup hedef Yol Kenarı Ünitesine ve araçlara optimum yönlendirme yol bilgilerini gönderir. Araç ve Yol Kenarı Ünitesi kontrolcünün belirlediği kurala göre hareket eder. Bu yönlendirme mekanizması sayesinde mesaj gecikmesi, ağ bant genişliği tüketimi azalır. Kontrolcü, yönlendirme kararlarını veri trafiğine, trafikteki araç yoğunluğuna ve yol koşullarına göre güncelleyebilir.

SD-VANET mimarisinin dezavantajları

- Bilgisayar ağlarında bazı güvenlik tehditleri yaygın olsa da Yazılım Tanımlı Ağ mimarisinin kendine has güvenlik tehditleri vardır. Ağ bilgilerinin toplanması, ağın yapılandırılması, yönlendirmenin hesaplanması gibi birçok hayati fonksiyon, ağın beyni olan Yazılım Tanımlı Ağ kontrolcüsü tarafından yapılır. Bu durum kontrolcüyü saldırganların birincil hedefi haline getiriyor. Saldırgan, kontrolcüyü hedef alan başarılı bir DDoS saldırısı gerçekleştirirse, kontrolcünün kaynakları (bellek, işlemci, bant genişliği vb.) tükenerek ağı çalışamaz hale getirebilir. Yeni ve geçerli paketler kontrolcüye ulaşamaz ve yazılım tanımlı ağ mimarisi çöker. Yazılım Tanımlı Ağ yapısı ile entegre VANET ağ

yapısında meydana gelebilecek böyle bir saldırı insan hayatını tehlikeye atabilir ve ciddi mali kayıplara neden olabilir [25].

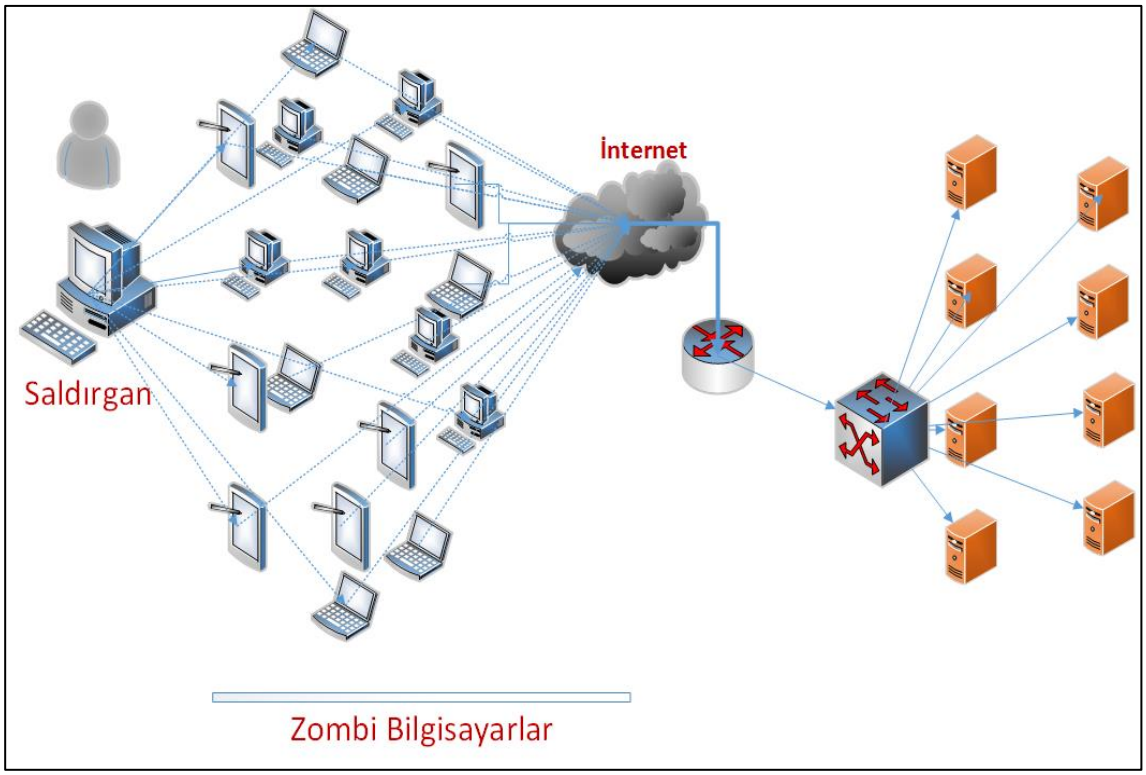
- Yazılım Tanımlı Ağ tabanlı ağların merkezi bir şekilde tek noktadan yönetilmesi ölçeklenebilirlik, gecikme süresi gibi sorunlara da neden olur. Özellikle büyük ölçekli ağların tek bir kontrolcü tarafından yönetilmesi kontrolcü üzerinde ek yüke, kaynak yönetiminin verimli bir şekilde yapılamamasına neden olur. Veri düzleminde bulunan iletim cihazlarının kendi aralarında oluşturdukları bağlantıya saldırganların kötücül yazılımlarla sızarak iletim cihazlarını ele geçirmesi veya cihazlar arasındaki akan veriyi değiştirmesini önlemek amaçlı kimlik doğrulama, erişim kontrolü gibi güvenlik önlemlerinin alınması, yetkisiz girişlerin engellenmesi gerekmektedir [18].
- VANET gibi kablosuz ağlar yüksek hareketliliğe ve dinamik değişken bir ağ yapısına sahiptir. Hareketli ve dinamik yapı ağ alt yapısındaki cihazların(araç, Yol Kenarı Ünitesi, AP, vs.) kendi arasındaki ve ağ alt yapısındaki cihazlar ile kontrolcü arasındaki iletişiminin anlık veya uzun süreli kopmasına neden olabilir. Bu durum kontrolcünün ağ üzerindeki denetimini olumsuz etkiler.
- Dinamik ve otomatik olarak değişen sınırlı bir topolojiye sahip olan VANET ağlarında karşılıklı bağlantı için önemli olan güven düzeyini hızlı bir şekilde belirlemek gerekir. Ağdaki herhangi bir cihazın, ağ cihazlarının güvenliliğini sağlaması için ağdan veri elde etmesi dinamik ve değişken yapıdan kaynaklı oldukça zordur. Bulut, sosyal, hücresel ağlar, sanallaştırma gibi teknolojilerin ağa dahil edilmesi yeni güven çözümleri gerektirir [13].
- Kontrolcü, veri düzleminde bulunan iletim cihazlarından anlık trafik verisi alır. Kontrolcü veri trafiği ile ilgili aldığı istatistikler doğrultusunda kötücül veri trafiği ve Yol Kenarı Ünitesi, araç gibi veri düzleminde bulunan kötücül davranış sergileyen cihazları da tespit eder. Örneğin, iletim cihazları (Yol Kenarı Ünitesi, AP, araç vs.) bazen gelen paketlerin düşmesine neden olur ve bu da bir kara delik oluşturur. Ağda birden fazla kötücül düğüm olması durumunda bu düğümlerin kontrolü zorlaşır. İşlem ve iletişim kapasitesi sınırlı olan kontrolcünün işlemcisinin aşırı yüklenmesine sebep olur.
- Nüfusun artmasıyla trafikteki araç yoğunluğu hızlı bir şekilde artmaktadır. Geleneksel VANET mimarisinde kullanılan yönlendirme algoritmaları yoğun trafik, otoyol çeşitliliği ve hava koşullarının değişkenlik gösterdiği durumlarda ölçeklenebilir değildir. SD-VANET mimarisi dinamik ve esnek yapısıyla ölçeklenebilirlik problemine çözüm sunar. Fakat yüksek hareket kabiliyetine sahip araç çeşitliliğin ve yoğunluğunun artması Yazılım Tanımlı Ağ kontrolcüsünün, zamanla veri düzlemindeki araçlara veriyi sorunsuz

göndermesinde sıkıntılar yaşamasına neden olabilir. Araç yoğunluğunun arması kontrolcüyü darboğaza sokabilir. Kontrolcünün darboğaza girmesi durumunda ağ performansında ciddi düşüşler yaşanabilir. Trafik yoğunluğunu tahmin edecek yapay zekâ tabanlı trafik tahmin algoritmaları geliştirilip ağa dâhil edilmelidir. Bu algoritmaları sayesinde ileride yaşanacak yoğunluk problemine karşı kontrolcü önceden aksiyon alabilir. Böylece kontrolcünün darboğaza girmesi engellenmiş olur. Bir kontrolcü veri düzleminde bulunan iletim cihazları tarafından gönderilen akış kuralı oluşturma taleplerine yanıt vermek için sınırlı bir kapasiteye sahiptir. SD-VANET ağının boyutu veya ağ trafiği miktarı büyüdükçe, tek bir kontrolcü akış kuralı oluşturma isteklerini karşılayamayabilir, bu da kontrolcü performansının düşmesine, SD-VANET ağının işlevsiz olmasına neden olabilir. Bir grup kontrolcünün büyük miktarda ağ trafiğini ve akış işleme isteklerini iş birliği içinde ele aldığı çoklu kontrolcünün kullanıldığı bir mimari oluşturularak da kontrolcünün darboğaza girmesi engellenebilir.

2.4. Dağıtılmış Hizmet Reddi Saldırıları

Hizmet reddi (Denial of Service-DoS) saldırısı, bir bilgisayarı veya internete hizmet veren sunucuları hizmet sağlayamaz hale getirmek için tasarlanmış bir saldırıdır. Saldırgan hedef sistemlerin işlemci, bellek, disk, bilgisayar ağlarının bant genişliği veya bağlana bilirliğini hedef alır. Saldırı bir kaynak tarafından gerçekleştirilen kötü niyetli bir eylem sonucunda, hedef sistemlere meşru kullanıcıların (sunucu, kişisel bilgisayar, vs.) erişimi kasıtlı olarak engellendiğinde veya bozulduğunda gerçekleşmiş sayılır. Sunucu veya bilgisayarlara saldırgan tarafından gönderilen yüksek miktardaki bağlantı talebi cihazların kaynaklarını (CPU, bellek, disk, internet bantgenişliği vs.) tüketir. Sunucular yaşanan bu yoğunluktan dolayı meşru kullanıcıların taleplerine cevap veremez hale gelir.

Bir kaynaktan bir hedefe doğru gerçekleştirilen DoS saldırılarının aksine, DDoS saldırısı birden fazla saldırgan sistem tarafından bir hedefe doğru gerçekleşir. Saldırı internete erişimi olan tüm cihazlar kullanılarak gerçekleştirilir. Saldırgan veya saldırganlar çeşitli yöntemlerle zombi haline getirdikleri bilgisayarlar sayesinde saldırıyı gerçekleştirir. Zararlı yazılım vasıtasıyla bilgisayarların zombi haline getirilmesi, saldırganlar tarafından en çok tercih edilen yöntemdir (Şekil 2.8).



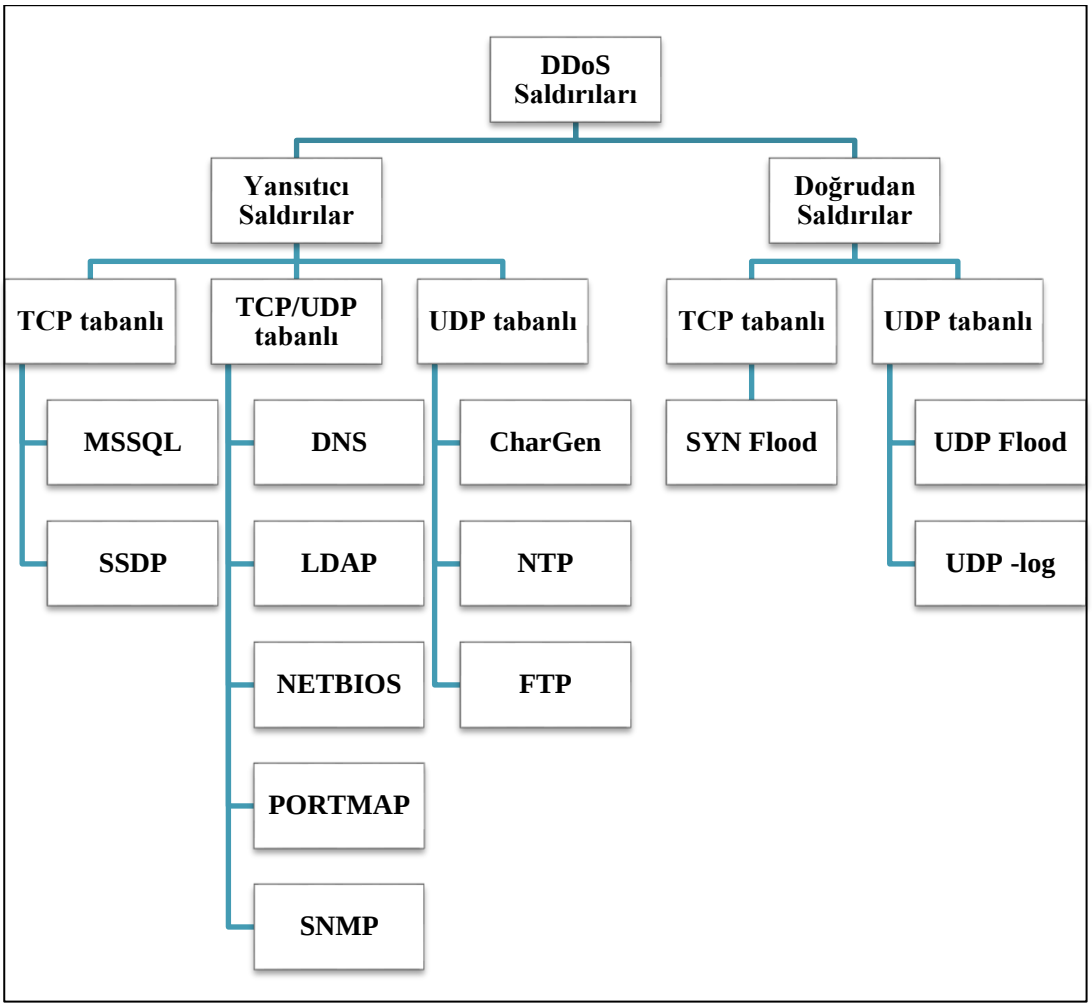
Şekil 2.8. Geleneksel DDoS saldırısı

DDoS, güvenliği ihlal edilmiş birçok ana bilgisayar kullanılarak oluşturulan koordineli bir saldırıdır. Saldırı kapsamlı ve başarılı bir şekilde gerçekleşir. Buda saldırganın kendini kolayca gizlemesine olanak tanır. DDoS saldırısının temel amacı meşru kullanıcılara herhangi bir hizmet sağlamayacak bir şekilde hizmet ve ağ alt yapısına zarar vermektir. Saldırgan kapsamlı bir DDoS saldırısı başlatmak için aşağıdaki adımları izler:

- Saldırgan, saldırıda kullanacağı savunmasız bilgisayarları bulamak amacıyla ağ taraması yapıp bilgi toplar.
- Ağ taraması sonucunda tespit ettiği savunmasız bilgisayarlara kötücül yazılım yükleyerek zombi haline getirir.
- Zombi haline getirdiği bilgisayarlara komut vererek yoğun bir şekilde saldırı paketlerini hedef sistemlere (kurban) yönlendirir.
- Son olarak saldırgan, saldırıya ait kayıt ve dosyaları bellekten silmesi için komut verir.

DDoS saldırıları iki farklı şekilde oluşturulabilir: doğrudan saldırı ve yansıtıcı saldırı (Şekil 2.9). Doğrudan saldırıda, çok sayıda saldırı paketi doğrudan hedef sisteme gönderilir. Saldırgan kaynak IP adresini taklit eder, böylece yanıt yanlış yönlendirilir ve başka bir yere gider. Hedef(kurban) sistemler, saldırgan sistemlerin bağlantı taleplerine karşı göndermiş olduğu

yanıt paketlerine saldırgan tarafından cevap verilmediği için bağlantı sürekli açık kalır. Bu şekilde hedef sistemler sürekli meşgul edilir. Dolaylı bir saldırı durumunda, saldırgan doğrudan sistemi hedef almaz. Saldırı oluşturmak için yansıtıcı olarak bilinen birçok masum ara düğüm kullanılır. Saldırgan, yansıtıcılara yanıt gerektiren paketler gönderir ve paketlerin yazılı kaynak adresi hedef sistem adresi olarak ayarlanır. Saldırı paketleri TCP, UDP, ICMP protokolleri kullanılarak oluşturulabilir.



Şekil 2.9. DDoS saldırı türleri

Yansıtıcı Tabanlı DDoS Saldırıları (Reflection-based DDoS)

Yansıtıcı tabanlı DDoS saldırılarında saldırgan, bir saldırı oluşturmak için yansıtıcı olarak bilinen birçok masum ara düğüm kullanılır. Saldırganlar kurbanın IP adresini arzu edilen bir hedef IP kaynağı olarak belirler ve kurbanı yanıt paketleriyle alt etmek için paketleri yansıtıcı sunuculara aktarır. Yansıtıcı tabanlı DDoS, uygulama katmanı protokolleri aracılığıyla, taşıma katmanı protokolleri TCP, UDP veya her ikisini de kullanarak yapıldığı için tespit edilmesi zordur.

Yansıtıcı saldırıları, her iki yönde de aynı protokolü kullanan meydan okuma-yanıt kimlik doğrulama mekanizmasından yararlanır. Yansıtıcı saldırılarında ana fikir, hedefi saldırganın meydan okumasına yanıt vermesi için kandırmaktır. Yansıtıcı saldırılar için genel saldırı akışı aşağıdaki gibidir:

1) Saldırgan, hedef X ile bir bağlantı başlatır.

2) Hedef X, saldırganın kimliğini doğrulamak için bir istek gönderir.

3) Saldırgan, hedef Y ile başka bir bağlantı kurar ve hedef X'in kendi isteğini gönderir.

4) Hedef Y, hedef X'in isteğine yanıt verir.

5) Saldırgan bu yanıtı, hedef X'in isteğine yanıt olarak X hedefine gönderir.

Kimlik doğrulama mekanizması dikkatli bir şekilde tasarlanmazsa, hedef X yanıtı geçerli olarak kabul eder. Doğru tasarlanmayan kimlik doğrulama mekanizması, saldırganın kimliği tam olarak doğrulanmış bir bağlantı kanalına erişim sağlamasına sebep olur. Bazı saldırı türleri aşağıda açıklanmıştır:

MSSQL Tabanlı Saldırılar: Saldırganlar, Microsoft SQL Server Çözümleme Protokolünün (Microsoft SQL Server Resolution Protocol-MC-SQLR) çalışmasına müdahale ederler. Saldırganlar, DDoS saldırılarını gerçekleştirmek için Microsoft SQL Server'ı (MSSQL) kullanırlar. İstemciler, bir veri tabanı sunucusundaki veri tabanı örneklerini veya bir ağ üzerindeki veri tabanı örnekleri kümesini tanımlamak için MC-SQLR protokolünü kullanır. Sorgulanan sunucu mevcut veri tabanı örneklerinin bir listesiyle yanıt verir.

Basit Hizmet Algılama Protokolü (Simple Service Discovery Protocol-SSDP) Tabanlı Saldırılar: Evrensel tak ve çalıştır (Universal Plug and Play UPnP) cihazları ağ üzerindeki diğer cihazlara varlıklarını SSDP protokolü üzerinden yayınlarlar. SSDP, bilgisayarlar, kablosuz erişim noktaları, modemler, yönlendiriciler, web kameraları, akıllı TV'ler, tarayıcılar ve yazıcılar gibi cihazların çoğunda varsayılan olarak etkinleştirilmiştir. Protokol, cihazların ağ üzerinde iletişim ve veri aktarımı için birbirlerini sorunsuz bir şekilde keşfetmelerini sağlamak için kullanılır. SSDP sunucuları gibi ağ protokolü sunucuları, hacimsel DDoS saldırılarının iyi bilinen örnekleridir. Güçlendirilmiş yansıma tabanlı SSDP saldırısı, UPnP protokolünden yararlanarak büyük miktarda trafik oluşturur [26].

DNS Tabanlı Saldırılar: Alan Adı Sistemi (Domain Name System-DNS), adları IP adreslerine ve tersine çevirmek için kullanılan bir uygulama katmanı protokolüdür. Bir sunucu hiyerarşisine dayalı olarak çalışan dağıtılmış bir veri tabanı olarak işlev görür. İnternete acık olan DNS sunucuları kullanılır. Yansıtıcı saldırı türleri arasında en popüler olanıdır. DNS sunucusuna yönelik UDP flood kullanılarak gerçekleştirilen bir uygulama katmanı saldırısıdır. Saldırgan çok sayıda kaynak IP adresinden büyük miktarda sahte DNS istek paketi gönderir. DNS sunucusunun sahte bir DNS isteğini gerçeğinden ayırt etmesi zordur. Bir müddet sonra sahte istekler DNS sunucusunu boğarak ve hizmet dışı kalmasına neden olur.

Hafifletilmiş Dizin Erişim Protokolü (Lightweight Directory Access Protocol-LDAP) Tabanlı Saldırılar: Microsoft aktif dizininde depolanan hesap adı ve parola bilgilerine erişmek için bir istemci tarafından kullanılan bir protokoldür. Saldırı TCP ve UDP protokolleri vasıtasıyla gerçekleştirilir. Saldırı UDP protokolü kullanılarak yapıldığında Connection-less Lightweight Directory Access Protocol (CLDAP) protokolü kullanılır. Bir saldırgan, kurbanın IP'sini taklit ettikten sonra birkaç LDAP sunucusuna istek gönderir. LDAP sunucuları da kurbana güçlendirilmiş yanıtlar gönderir. Saldırı sonucunda kurbanın ağı önemli sayıda LDAP yanıt paketiyle işlevsiz hale gelir [27].

Ağ Zamanlama Protokolü (Network Time Protocol-NTP) Tabanlı Saldırılar: Ağ üzerindeki makinelerin saatlerini senkronize etmek için kullanılan basit bir UDP protokolüdür. Saldırgan kurbanın IP adresini taklit eder ve monlist adında bir komut verir. Komut, NTP sunucusuna yapılan son 600 bağlantı hakkında bilgi verir. Küçük bir istek muazzam bir yanıtla sonuçlanır.

CharGen Tabanlı Saldırılar: Saldırgan CharGen protokolünü destekleyen bir sunucuya bir TCP isteği gönderirse, sunucu bağlantı kapanana kadar sürekli olarak isteğe yanıt olarak rastgele karakterler göndermeye başlar. UDP sorguları gönderilmesi durumunda, sunucu her UDP datagramı aldığında rastgele seçilmiş karakterler içeren bir UDP datagaramı (0-512 bayt) geri göndererek yanıtlar. Saldırgan bu yapıyı kullanarak özellikle DNS sunucularına büyük zarar verebilir. CharGen çalıştıran cihazlara saldırgan tarafından sahte IP adresleriyle (kurban) gönderilerek başlatılır. Paketleri alan cihazlar 19 numaralı port üzerinden kurbana UDP paketleri gönderir. Kurban, cihazlardan gelen yanıtlardan dolayı işlevsiz hale gelir.

Portmap Tabanlı Saldırılar: Portmapper, RPC protokolünü temel alan ve bir programdaki bir prosedürün ağ üzerinden başka bir makinede çalıştırılabilmesi için çağrılmasını sağlayan TCP ve

UDP kullanılarak 111 numaralı port üzerinden çalışan bir hizmettir. Portmap hizmetini çalıştıran bir makineye bir istek göndererek, RPC hizmetini sağlayan mevcut programlar ve bunlara karşılık gelen port numaraları ile yanıt verir. Saldırganların portmap'i kötüye kullanmasına, hedefin IP adresini taklit etmesine ve internet üzerinden portmap hizmeti sağlayan birçok bilgisayar sunucusuna istek göndermesine olanak tanır. Port mapper TCP'den de faydalanabilir; ancak amplifikasyon saldırısı UDP kullanılarak başlatılır. Kurban böyle bir saldırıyı hafifletmekle yükümlü değildir. Bu tür saldırıları hafifletmek için sunucuların düzgün bir şekilde yapılandırılması gerekir ve bu da sunucu sahiplerinin sorumluluğundadır [28].

Doğrudan (Sömürü Tabanlı) DDoS Saldırıları

Protokollerin veya yazılım uygulamalarının zayıflıklarından yararlanılarak başlatılan, hedef sisteme doğrudan bağlantı isteğinde bulunan, istismara dayalı DDoS saldırılarıdır. Saldırı TCP veya UDP protokolleri vasıtalıyla gerçekleşir. Birçok sömürü tabanlı DDoS saldırısı türü bulunmaktadır. Bu saldırı türlerinin bazıları aşağıda açıklanmıştır.

TCP SYN Flood saldırıları: SYN flood en yaygın DDoS saldırısı türüdür. Saldırgan tarafından TCP bağlantılarının üçlü el sıkışma mekanizma süreci kötüye kullanılır. Saldırgan sahip olduğu botnet ağında bulunan zombi makineler vasıtasıyla kurban sunuculara çok sayıda TCP-SYN request isteği gönderir. Sunucular her SYN paketine SYN-ACK paketiyle cevap verir. Sunucu zamanlayıcı sona erene kadar istemci tarafından bir onay bekleyerek bağlantıyı açık tutar. Saldırgan asla kurbandan aldığı ACK paketlerine SYN-ACK paketiyle cevap vermez. Kurban gönderdiği ACK paketlerine cevap alana kadar SYN-ACK paketi göndermeye devam eder. Bu da ağda giderek artan sayıda yarı açık bağlantıya neden olur. Bu işlem sunucunun hafızasının ve işlem gücünün bir kısmını kullanır. Saldırgan tarafında gelen çoklu SYN istekleri zamanla sunucuların kaynaklarını tüketerek hizmet veremez hale getirir.

UDP Flood Saldırıları: Saldırgan çok sayıda sahte UDP paketini rastgele sunuculara veya belirli bir sunucuya gönderir. Saldırgan, kurbanın ağındaki mevcut tüm bant genişliğini tüketmeyi amaçlar.

UDP-Lag Saldırıları: UDP-Lag, istemci ve sunucu arasındaki bağlantıyı bozan saldırı türüdür.

2.5. SD-VANET Ağ Güvenliği

Sürücüsüz araçların trafikte kullanımı, akıllı ulaşım servisleri için sürekli yeni protokollerin ve mimarilerin geliştirilmesi gibi yeni teknolojilerin ve mimari bileşenlerin mevcut ağ yapısına entegrasyonundan önce güvenlik, sistem verimliliği ve güvenliği etkileyecek siber saldırılar için gerekli önlem ve alt yapılarının oluşturulması gerekmektedir [29].

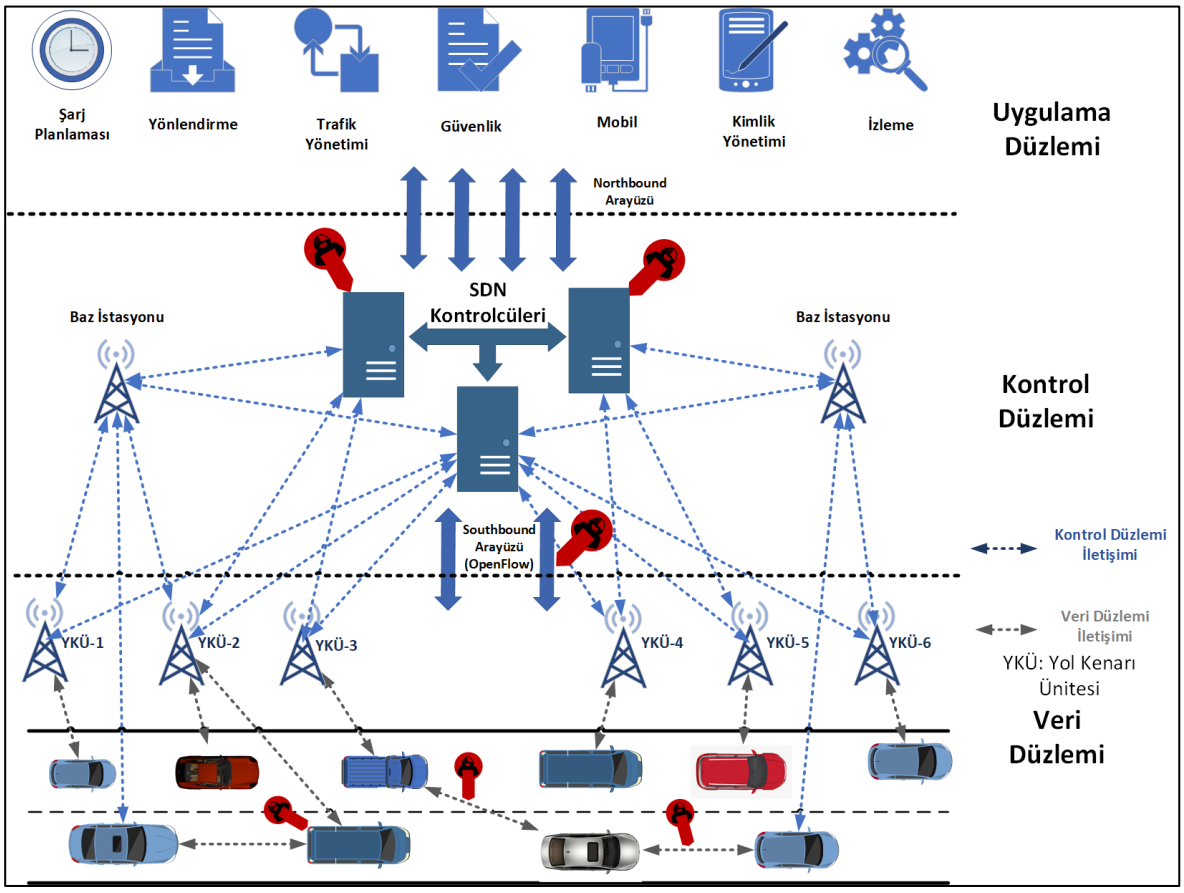
Bu önlemler alınmadan yeni teknolojilerin bir siber saldırıya maruz kalması durumunda alt yapının zarar görmesi, insan hayatının tehlikeye girmesi gibi birçok sonuç ortaya çıkabilir. SD-VANET ağını hedef alan kötü niyetli kullanıcılar tarafından gerçekleştirilen birçok saldırı türü vardır.

Bu saldırıların en yaygını ve tehlikeli olanı DDoS saldırılarıdır. Yazılım Tanımlı Ağa yönelik gerçekleştirilen DDoS saldırısının işlem adımları:

- Başlangıçta saldırgan, kaynak adreslerinin geçerli bir paket gibi görünecek şekilde taklit edildiği bir dizi sahte paket oluşturur.
- Paketler ağa giriş yaptıktan sonra anahtarlardan geçer. Anahtarlar bu paketlerin yasal/meşru olduğunu varsayar.
- Anahtara gelen paketler akış tablosu girdileriyle eşleşmediği için bu paketleri anahtar tarafından kontrolcüye gönderilerek yeni bir akış kuralı talep edilir. Sorun burada ortaya çıkar. Kontrolcü, sahte paketler için yeni akış kuralı oluşturup anahtara gönderir.
- Saldırgan, sürekli olarak ağa sahte paketler göndermeye devam ettiği için bir süre sonra kontrolcünün kaynakları (CPU, bant genişliği vs.) tükenmeye başlayacaktır. Ağa gelen meşru paketler kontrolör tarafından yönlendirilemez. Bir süre sonra yasal paketler düşmeye başlar.
- Bir süre sonra kontrolcünün kaynaklarını saldırı trafiğinden kaynaklı tükenecektir. Bu sahte paketler kontrolcüyü belirli bir süre için ulaşılamaz hale getirecektir. Ağın beyni konumunda olan kontrolörün kaybı Yazılım Tanımlı Ağdaki tüm ağ hizmetlerinin tehlikeye girmesine sebep olur.

DDoS saldırılarının iki ana amacı vardır. Birincisi, hedef hizmetin ve servislerin işlemci, bellek, bant genişliği gibi kaynakları tüketilerek çalışamaz hale getirmektir. İkincisi ise; hedef sisteme sızarak ağ alt yapısı ve sunucuların tüm güvenlik açıkları tespit ederek, bulunan açıktan

yararlanılarak hedef servis çalışamaz hale getirmektir. SD-VANET mimarisine yönelik yapılan DDoS saldırılarının 3 ana hedefi vardır (Şekil 2.10): Kontrol düzlemine, veri düzlemine ve iki düzlem arasındaki iletişim hattına yönelik yapılan saldırılar.



Şekil 2.10. SD-VANET ağına yönelik DDoS saldırılarının ana hedefleri

SD-VANET mimarisine yönelik yapılan DDoS saldırılarının en tehlikelisi, kontrol düzleminde bulunan ve ağın beyni olan kontrolcüye yapılan saldırılardır.

Saldırganın virüs bulaştırarak zombi haline getirdiği bilgisayarlar; anahtar, araçlar vs. sayesinde eşzamanlı olarak Yol Kenarı Ünitelerine çok sayıda istek gönderir. Gelen paketler Yol Kenarı Ünitelerinde bulunan akış tablosundaki kurallar ile eşleşmiyorsa (miss flow), tüm eşleşmeyen girdiler için, Yol Kenarı Üniteleri aracılığı ile OpenFlow protocol "OFPT _PACKET_IN" mesajıyla kontrolcüye yeni kural yazması için gönderilir [30]. Kontrolcü gelen akış kuralı isteklerine "OFPT_PACKET_MOD" mesajıyla cevap verir. Yeni akış kuralı istekleri artıkça, kontrolcünün işlem ve iletişim kapasitesi aşırı yüklenir. Sınırlı kaynağa (hesaplama, işlemci, hafıza ve bant genişliği gibi) sahip olan kontrolcünün kaynakları bir süre sonra tükenir. Kaynakları tükenen kontrolcü gelen yeni akış kuralı isteklerine cevap veremez hale gelir. SD-VANET ağının

düzlemleri arasındaki iletişim, kontrolcünün çökmesi sonucu kesilir. Böylece SD-VANET mimarisi çöker. SD-VANET ağında kontrolcüye yönelik meydana gelebilecek bu tür bir saldırıda ağır yol kayıplarına ve ciddi trafik sıkışıklığına neden olabilir. Bu tür saldırılara en iyi çözüm, trafik yoğunluğunun yüksek olduğu alanlarda birden fazla kontrolcü kullanılmasıdır.

Çoklu kontrolcü kullanımı ile saldırı anında kontrolcünün gerçek zamanlı tepkime hızı arttırılarak saldırılara karşı etkin bir şekilde cevap verilebilir. Çoklu kontrolcü kullanımının başka bir avantajı ise kontrolcünün Yol Kenarı Ünitesi ve anahtarlar gibi iletim cihazları ile olan iletişim süresi artar. Bu da kontrolcünün iletim cihazları üzerindeki denetimini ve veri iletim hızını arttırır. Özellikle SD-VANET gibi kablosuz ağ yapılarında hiyerarşik yapıda çoklu kontrolcünün kullanımı ve kontrolcünün ağdaki konumu, kontrolcünün veri düzleminde bulunan ağ elemanları üzerindeki denetimini arttırarak, kaynak kullanımını optimize ederek gecikmeyi azaltır.

Fakat çoklu kontrolcünün bir arada olduğu ağlarda, ölçeklenebilirlik, yük dengeleme, güvenlik gibi sıkıntılar mevcuttur. Bu sıkıntıların en önemlisi ve kritik olanı güvenlik konusudur. Çoklu kontrolcünün kullanıldığı bir ağda, ağın devamlılığı için kontrolcüler arasındaki veri akışının kesilmemesi gerekmektedir. DDoS benzeri saldırılarda saldırgan kontrolcüler arasında akan veri paketlerine zarar vererek kontrolcülerin birbirleriyle olan iletişimini kesmeyi hedefleyebilir. Ağın devamlılığı için kontrolcüler arasındaki bağlantının kopmaması gerekmektedir. Kontrolcüler arasındaki bağlantının kopması durumunda Yazılım Tanımlı Ağ mimarisi çökebilir [31].

SD-VANET ağında, veri düzlemimde bulunan iletim cihazları (Yol Kenarı Ünitesi, AP, anahtar, yönlendirici vs.) ve kontrolcü arasında OpenFlow protokolü vasıtasıyla ağ trafiği bütünlüğü sağlanır (Şekil 2.10). OpenFlow protokolü aracılığıyla kontrolcü, iletim cihazlarında bulunan mevcut akış tablosundaki akış girdilerini değiştirebilir, silebilir veya yeni akış girdisi ekleyebilir. Saldırgan OpenFlow protokolünü idare ederse, kontrolcü tarafından veri düzleminde bulunan iletim cihazlarına gönderilen verileri ele geçirerek veri düzlemini ele geçirebilir.

OpenFlow kanalı, veri düzleminde bulunan iletim cihazlarını kontrolcüye bağlayan arayüzdür. Kontrolcü, bu arabirimi kullanarak iletim cihazlarını (Yol Kenarı Ünitesi, AP, anahtar, yönlendirici vs.) yapılandırır ve yönetir. Ayrıca kontrolcü, bu kanal vasıtasıyla iletim cihazlardan akış istatistiklerini alır ve akış kurallarının bulunduğu kuralları içeren paketleri iletim cihazlarına gönderir. Bu kanal genellikle TLS kullanılarak şifrelenir, ancak doğrudan TCP üzerinden çalışabilir. OpenFlow protokolü, kontrolcüye bağlı anahtar veya Yol Kenarı Ünitesi cihazlarında

bulunan kötücül davranışı tespit ve engelleme mekanizmasına sahip değildir. Kötücül davranıştan kaynaklı sorunları direk olarak kontrol düzlemine taşır.

Saldırgan bu kanalı dinleyerek kontrolcü tarafından gönderilen iletim kurallarına kötücül kural ekleyerek veri paketlerinin doğru iletilmesini engelleyebilir. DDoS saldırılarında, OpenFlow ve kontrolcü arasındaki güvenli hat saldırı trafiği ile işgal edilip, bu hattın bant genişliği tüketilirse, kontrolcü ve veri düzleminde bulunan iletim cihazları (Yol Kenarı Ünitesi, AP, anahtar, yönlendirici vs.) arasındaki veri akışı olumsuz etkilenir. Veri trafiğinin devamlılığı ve bütünlüğü bu kanal vasıtasıyla sağlandığı düşünüldüğünde bu kanalın kopması durumunda kontrolcünün veri düzlemi ile olan bağlantısı kesilir.

Yol Kenarı Ünitesi ve Access Point gibi iletim cihazları sınırlı depolama kapasitesine sahip olduğu için tüm akışları kapsayan tüm kuralları depolaması mümkün değildir. Bilinmeyen bir adresten paketler geldiği zaman yeni kuralların Yol Kenarı Ünitesi/Access Point'in akış tablosuna eklenmesine neden olur. Saldırgan bilinmeyen bir adresten kısa süre içinde çok miktarda paket gönderdiği zaman, kontrolcü tarafından bu paketler için kural yazılıp Yol Kenarı Ünitesi/Access Point'in akış tablosuna gönderilir. Sınırlı depolama alanına sahip olan akış tablosu kısa sürede dolar. Akış tablosunda yeni kural için yer kalmaz. Bundan dolayı yasal trafiğin iletilmesi durur. Akış tablosunun haricinde iletim cihazlarında bulunan akış önbelleği de DDoS saldırıları için hedef konumundadır. Reaktif önbellek mekanizmasına sahip olan Yol Kenarı Ünitesi/Access Point, giriş portundan paketi alınca bu paketi akış arabelleğine gönderir. Gelen yeni paketler için akış tablosundaki akışlar ile eşleşme aranır. Eğer eşleşme sağlanırsa paket arabellekten çıkış portuna yönlendirilir. Eşleşme sağlanmaz ise, "OFPT_PACKET_IN" mesajıyla OpenFlow protokolü vasıtasıyla kontrolcüye gönderilir. Kontrolcü ise paket için gerekli kuralları ve kuralın ne kadar kalacağını belirleyen "HARD_TIMEOUT" ve "IDLE_TIMEOUT" atayarak "OFPT_FLOW_MOD" mesajıyla cevap verir. Yol Kenarı Ünitesi/Access Point kontrolcünün yazdığı kuralı alınca paket işleme alınır ve gelen paketlerin direk işleme alınması için kontrolcü tarafından yazılan kural akış tablosu önbelleğine alınır. Bu mekanizma Yol Kenarı Ünitesi/Access Point'i DDoS saldırıların karşı korumasız hale getirir. Yol Kenarı Ünitesi/Access Point'lerde bulunan kontrolcünün gönderdiği akış kurallarının tutulduğu akış tablosu ve önbellek, sınırlı depolama kapasitesine sahip olduğu için kısa zamanda dolar ve ağa gelen yasal paketler de saldırı anında düşmeye başlar.

3. LİTERATÜR ÖZETİ

SD-VANET mimarisine yönelik siber saldırıların tespiti ve engellenmesine yönelik yapılmış çalışmaların sayısı fazla değildir. Var olan çalışmaların çoğunluğu SD-VANET ağlardaki güvenlik gereksinimlerinin ve çerçevesinin belirlenmesini sağlamak amacıyla yapılmıştır. Ayrıca çalışmalarda geleneksel VANET mimarisinin, zayıf bağlantı, ölçeklenebilirlik, esneklik, yönetilebilirlik, güvenlik gibi zayıflıkları da ortaya konulmuştur. Geleneksel VANET mimarisinin akıllı ulaşım sistemler gibi yeni nesil teknolojiler için yeterli olmadığı, yaşanılabilecek problemlerin büyük bir kısmının SD-VANET mimarisi ile ortadan kalkacağı öngörülmektedir.

Literatürde, DDoS saldırılarına benzer sınıflandırma problemlerinin çözümü için genellikle makine öğrenimi yöntemleri kullanılmaktadır. Var olan çalışmalarda, genellikle Destek Vektör Makinaları (DVM-Support Vector Machine-SVM) , Naive Bayes, K-En Yakın Komşu (K-EYK-K-Nearest Neighbour-KNN), Doğrusal Diskriminant Analizi (Linear Discriminat Analysis), ve Karar Ağacı (Decision Tree-DT) gibi sınıflandırıcı yöntemleri kullanılmıştır. Buna ek olarak, az sayıdaki çalışmada, saldırıların sınıflandırılması için derin öğrenme tabanlı yöntemler kullanılmış, etkili öznitelikleri seçmek için de öznitelik seçme ve öznitelik azaltma işlemleri uygulanmıştır.

Yazılım Tanımlı Ağ mimarisine yönelik DDoS saldırılarının makine öğrenimi yöntemleri ile tespitine dayalı literatürde bulunan bazı çalışmalar aşağıda özetlenmektedir:

Alshamrani . ve arkadaşları yaptıkları çalışmada mevcut kullanılan DDoS saldırılarını engelleme mekanizmalarının saldırıyı engellemede çok başarılı olmadığını belirtmişlerdir. Misbehavior ve Newflow saldırılarının Yazılım Tanımlı Ağ mimarisi üzerindeki etki incelemiş. Veri düzleminde bulunan iletim cihazlarından periyodik olarak trafik verileri toplanarak makine öğrenimi sınıflandırma algoritmaları uygulamış. Yapılan çalışmada Yazılım Tanımlı Ağ mimarisinde yaşanacak ani trafik değişimlerine anında oluşturulacak sistemin cevap vermesi amaçlanmıştır. Sistem, makine öğrenme modülü, öznitelik ve makine öğrenimi modelinin eğitim, test ve tahmininden oluşuyor. 3 modülden oluşan bir sitem tasarlanmıştır. Çalışmada net olarak belirtilmemişse de saldırı anında kontrolcü ve iletim cihazları arasında akan Packet_In mesajları temel alınmıştır. Sınıflandırma algoritması olarak DVM, J48 ve Naive Bayes kullanılmıştır [32].

Latah ve Toker yaptıkları çalışmada DoS saldırısında saldırı anında gelen paket oranını ölçerek saldırıyı tespit edip, DVM ile sınıflandırmışlardır. Saldırı anında kontrolcüye gelen paketler önceden belirlenen eşiği geçtiği zaman gelen paketler incelenerek, DVM algoritması ile sınıflandırılmış [33].

Ye ve arkadaşları yaptıkları çalışmada kontrolcü vasıtasıyla veri düzleminde bulunan iletim cihazlarından ağ trafiği ile ilgili bilgi toplamıştır. Elde edilen trafik verileri kullanılarak, DVM algoritması ile DDoS saldırıları ile ilgili 6 karakteristik özellik çıkartılarak saldırı tespiti gerçekleştirilmiştir. Çalışmada saldırı tespitinde yüksek başarım oranı elde edildiği belirtilmiştir. ICMP protokollü tabanlı DDoS saldırısının tespitindeki doğruluk oranı diğer saldırı tiplerine göre daha düşük başarım oranına sahip olduğu vurgulanmıştır [34].

Nanda ve arkadaşları, güvenlik tehditlerini en aza indirmek için potansiyel kötü niyetli bağlantıları ve potansiyel hedefleri belirlemek için daha önce ağa yönelik gerçekleştirilen ağ saldırısı verileri üzerinde eğitilmiş makine öğrenimi algoritmalarının kullanılmasını önermiştir. Çalışmada C4.5, Bayes Ağı, Karar Ağacı ve Naive Bayes algoritmalarını kullanılmıştır. Makine öğrenimi algoritmaları kullanılarak yapılan tahminler sayesinde veri düzleminde bulunan kötü niyetli kullanıcıların tespit edilebileceği belirtilmiştir. Kullanıcı tanımlamasının sağlanması, DVM kontrolcünün, ağın verimliliği ve sürekliliği için önemli olan saldırıyı önlemek için hızlı ve etkili bir şekilde yeni kurallar yazmasını sağlayabilir [35].

Jankowski ve Amanowicz yaptıkları çalışmada veri düzleminde bulunan kötücül aktivitelerin tespiti ve izlenmesi için 4 modülden oluşan bir saldırı tespit sistemi önermiştir. Flow Bundle Modül, trafik istatistiklerini toplar. Integrator Collect and Processes ve Traffic Statistics modülleri trafik özelliğini tutar. Machine Learning Classifier Modülü ise kötücül trafiği tespit eder. Kendini Düzenleyen Haritalar (Self-Organizing Maps-SOM) ve Öğrenmeli Vektör Kuantalama (Learning Vector Quantization - LVQ) makine öğrenimi algoritmaları kullanılmıştır [36].

Mowla ve arkadaşları tarafından Yazılım Tanımlı Ağ güdümlü içerik dağıtım ağı'nda bilişsel anahtar tabanlı DDoS algılama ve azaltma yöntemi önerilmiştir. Trafik sınıflandırması için DVM ve Lojistik Regresyon (LR) algoritmaları kullanılmış ve bu sınıflandırma, yeni flooding saldırılarını önlemek ve olası tüm DDoS saldırılarını tespit etmek ve bunlara karşı savunma yapmak için OpenFlow anahtarlarına konuşlandırılmıştır [37].

Yu ve arkadaşları yaptıkları çalışmada araçsal ağlar için yazılım tabanlı bir platform önermişlerdir. Kontrolcü ve veri düzlemi arasındaki mesajlar(Packet_In) ve akış tablosu girdileri temel alınarak saldırı tespit platformu oluşturmuşlardır. DVM, sınıflandırıcı kullanılarak akış tablosu girdileri eğitilmiştir. Gerçek zamanlı saldırı tespiti için anahtar akış tablosu girdileri oluşturulan toplama modülünde toplanır. Toplanan akış girdilerinin özellik değeri çıkartılır. Akış girdilerinin iyi huylu veya kötü huylu olup olmadığının tespiti için protokol türlerine göre DVM eğitim modülüne gönderilir. Bu şekilde saldırı tespiti yapılır. Oluşturulan trigger modülü ile de Packet_In mesajı belli bir eşik değeri geçtiğinde modül vasıtasıyla kontrolcünün cevaplama süresi azaltılarak Packet_In mesajı sayısı azaltılır. [38].

Chuanhuang ve arkadaşları DDoS saldırılarının tespiti ve engellenmesi için SDN ağ yapısının bütün katmanlarını kapsayacak bir derin öğrenme modeli uygulamışlardır. Bu modelde Tekrarlayan Sinir Ağı (TSA-Recurrent Neural Network -RNN), Uzun Kısa Süreli Bellek (UKSB-Long Short-Term Memory-LSTM) ve Evrişimsel Sinir Ağı (ESA Convolutional Neural Network-CNN) kullanmışlardır. DDoS saldırılarının gerçek zamanlı tespiti ve engellenmesi için oluşturulan model, birden fazla kontrolcünün kullanıldığı büyük ağlarda, ağı yavaşlatıp kontrolcülerin birbirleriyle senkronize çalışmasını engelleyebilir [39].

Latah ve Toker yaptıkları çalışmada DoS saldırısında saldırı anında gelen paket oranını ölçerek saldırıyı tespit edip, DVM ile sınıflandırmışlardır. Saldırı anında kontrolcüye gelen paketler önceden belirlenen eşiği geçtiği zaman gelen paketler incelenerek, DVM algoritması ile sınıflandırmışlardır [40].

Mowla ve arkadaşları, Yazılım Tanımlı Ağ güdümlü içerik dağıtım ağında bilişsel anahtar tabanlı DDoS algılama ve azaltma yöntemini önermiştir. Trafik sınıflandırması için DVM ve LR algoritmaları kullanılmışlar. Bu trafik sınıflandırması ile güvenlik kuralları OpenFlow anahtarların üzerinde çalıştırılıp, tüm olası DDoS saldırıları tespiti ve engellenmesi amaçlanmıştır[41].

Niyaz ve arkadaşları yaptıkları çalışmada mevcut ağ yapısının, yönetimsel zorluklardan kaynaklı DDoS saldırılarına karşı korunaksız olduğunu vurgulamışlardır. Ağın tek elden yönetilmesini sağlayan Yazılım Tanımlı Ağ teknolojisiyle yönetimsel zorlukların aşılabileceğinden bahsedilmiştir. Ayrıca Yazılım Tanımlı Ağ teknolojisine Derin Öğrenme (Deep Learning) entegrasyonu ile DDoS saldırılarının tespiti ile ağda yaşanacak problemlerin

çözüme kavuşacağını savunmuşlardır. Yaptıkları çalışmada Yazılım Tanımlı Ağ ortamında Derin Öğrenme tabanlı çoklu vektör DDoS algılama sistemi önermişlerdir. Farklı senaryolar denenerek ağ trafik metrikleri alınıp, sistemin DDoS saldırısı anında performansı test edilmiştir [42].

Tang ve arkadaşları esnek ve programlanabilir alt yapısıyla Yazılım Tanımlı Ağ paradigmasının yakın zamanda geleneksel ağ yapısının yerini alabileceğini belirtmişlerdir. Bu kapsamda çalışmada akış tabanlı anomali tespiti için bir derin öğrenme modeli uygulanmıştır. Saldırı tespiti için Derin Sinir Ağı (DSA-Deep Neural Network-DNN) modeli oluşturulmuştur. Çalışmada elde edilen başarım oranlarının neticesinde, derin öğrenme yaklaşımının Yazılım Tanımlı Ağda akış tabanlı anomali tespiti için güçlü bir potansiyele sahip olduğunu belirtmişlerdir [43].

Hsider ve arkadaşları yaptıkları çalışmada, yeni teknolojilerin dijital çağı yeniden şekillendirdiği bir dünyada, siber saldırıların gelişmekte olan bilgi işlem ağları için büyük bir tehdit oluşturduğunu vurgulanmış. Bu saldırıların en tehlikelilerinden biri olan DDoS saldırılarına karşı merkezi kontrol mantığının temelini oluşturan Yazılım Tanımlı Ağ yapısının yapay zeka yöntemleri ile entegre edilmesi bu saldırıların tespit edilmesi ve önlenmesi konusunda büyük avantaj sağladığı belirtilmiştir. Çalışmada, akış tabanlı bir veri kümesi kullanarak saldırıyı tespit etmek ve doğrulamak için ESA tabanlı bir sistem önerilmiştir [44].

Tang ve arkadaşları, Yazılım Tanımlı Ağ mimarisinin esnek ve programlanabilir yapısıyla bilişim dünyasındaki birçok soruna çözüm sunsa da siber saldırılara karşı savunmasız olduğunu yaptıkları çalışmada belirtmişlerdir. Çalışmada derin öğrenme tabanlı bir saldırı tespit çerçevesi olan DeepIDS modeli önerilmiştir. Derin öğrenme modeli, tam bağlantılı DSA ve geçitli TSA kullanılarak NSL-KDD veri kümesiyle test edilmiştir. Çalışmada, DeepIDS modelinin akış tabanlı saldırılara karşı başarı sağlayabileceğini belirtilmiştir [45].

Dey ve arkadaşları geleneksel makine öğrenme algoritmaları ve derin öğrenme modelleri kullanarak saldırganların ana hedefi olan yazılım tanımlı ağ denetleyicisine yönelik saldırıları tespit etmeyi amaçlamışlardır. İlk olarak, NSL-KDD veri kümesine Rastgele Orman (Random Forest-RF) algoritması uygulanmıştır. Doğruluk oranını artırmak için bu veri kümesine öznitelik seçimi yapılmıştır. İkinci olarak Geçitli Tekrarlayan Birim (GTB-Gated Recurrent Unit-GRU)-UKSB tabanlı derin öğrenme tabanlı saldırı tespiti yapılmıştır. Her iki durumda

elde edilen sonuçlar karşılaştırıldığında, saldırı tanıma için derin öğrenmenin daha iyi bir seçenek olduğu vurgulanmıştır [46].

Ko ve arkadaşları 5G teknolojisinin gelişmesiyle DDoS saldırılarının sıklığının ve vektörel artışının artacağını vurgulamışlar. DDoS saldırılarının internet kullanıcıları üzerindeki etkilerini en aza indirmek için internet servis sağlayıcıları için 4 modülden oluşan Dinamik Öğrenme Sistemi (Dynamic Learning System) önermişlerdir. Dinamik Öğrenme Sistemi, ağ trafiğini sınıflandırmak için çekirdek öğreniciler olarak Tam Oto Kodlayıcı (Complete Autoencoder-CAE) kullanan denetimsiz bir topluluk modelidir. Derin oto kodlayıcıdan oluşan Tam Oto Kodlayıcı ile normal oto kodlayıcıdan oluşan Tam Oto Kodlayıcı arasındaki en büyük fark, Tam Oto Kodlayıcı'nın bir sınıf anahtarı aracılığıyla ikili bir sınıflandırma oluşturmak için saldırı verilerinin dengesiz özelliğini kullanmasıdır [47].

Ujjan ve arkadaşları yaptıkları çalışmada heterojen bir yapıya sahip olan IoT (Internet of Things) ağ altyapısının DDoS saldırılarına karşı korunmak hayati öneme sahip olduğunu vurgulanmış. Özellikle merkezi bir kontrolcüye sahip olan Yazılım Tanımlı Ağ yapısı, kendisine entegre IoT ağının maruz kalacağı böyle bir saldırının yüksek bellek tüketimi, düşük doğruluk, ağa üzerinde ek yük gibi ağın verimliliği ve sürekliliği için önemli sorunlara yol açabileceğini belirtilmiş. Çalışmada, DDoS saldırılarının etkilerinin üstesinden gelmek için IoT ağına yönelik DDoS saldırılarını azaltmaya yardımcı olan uyarlanabilir yoklama tabanlı örnekleme ve derin öğrenme tabanlı bir model ile sFlow ve Snort saldırı tespit sistemini önerilmiştir[48].

Ravi ve Shalinie sınırlı kaynaklara (işlemci, bant genişliği vb.) sahip Yazılım Tanımlı Ağ tabanlı IoT ağına karşı gerçekleştirilecek DDoS saldırılarında ağ servislerinin kesileceğini vurgulanmış. Çalışmada, kablosuz IoT ağları ve sunucuları tarafından tetiklenen DDoS saldırılarına karşı yarı denetimli bir makine öğrenme algoritması kullanarak saldırıyı algılayan ve azaltan bir Öğrenme Güdümlü Algılama Azaltma (Learning-Driven Detection Mitigation-LDDM) mekanizması önerilmiştir [49].

Ujjan ve arkadaşları yaptıkları çalışmada, DDoS saldırılarının IoT ağ altyapılarında gerçekleştirilen en yaygın saldırılardan biri olduğuna belirtmiştir. Yazılım Tanımlı Ağ tabanlı IoT ağına yapılacak bir DDoS saldırısı, aşırı bellek tüketimi, yüksek ek yük vb. gibi ağın kalıcılığını tehlikeye atan sorunlara neden olabilir. Bu nedenle, sFlow ve Snort saldırı tespit sistemi, bir IoT ağındaki yaygın

DDoS saldırılarını azaltmaya yardımcı olmak için önerilmiştir. İlk olarak, veri düzlemindeki anahtarların işleme kapasitesinin aşırı yüklenmesinin bir sonucu olarak ağda oluşacak ek yükü azaltmak için sFlow ve uyarlanabilir yoklama tabanlı örneklemeyi ayrı ayrı uygulanmıştır. İkinci olarak, kontrol düzleminde, algılama doğruluğunu optimize etmek için Yığınlanmış Oto Kodlayıcı (YOK-Stacked Autoencoder-SAE) tabanlı derin öğrenme modeliyle birlikte Snort kimliklerini uygulanmış [50].

De Assis ve arkadaşları tarafından yapılan çalışmada, kaynak uç ağda DDoS saldırılarını tespit eden ve bant genişliği, işlemci ve ağın beyni konumundaki Yazılım Tanımlı Ağ kontrolcüsünün kaynaklarını koruyan bir güvenlik sistemi önerilmiştir. DDoS tespiti, ESA tabanlı bir sitem oluşturularak gerçekleştirilmiştir [51].

Çalışmada, ağ paketlerinin akışını ve sistemin bant genişliğini kontrol etmek için Yazılım Tanımlı Ağ yapısı içinde önceliğe dayalı bir model önermiştir. Sistemdeki anormal girişleri tespit etmek için veri aktarım trafiği, akış tabanlı makine öğrenme modeli aracılığıyla sürekli olarak izlenmiş. Makine öğrenimi modeli, ayrılan bant genişliğiyle verimi sınırlamak ve ekstra bant genişliğinden yararlanmak için kullanılmış. Hem Makine Öğrenmesi hem de Derin Öğrenme yaklaşımları, tüm saldırı kategorilerindeki saldırılar için tanımlama yöntemleri olarak kullanılmış [52].

Çalışmada, internet üzerinden verilen hizmetlerin artması sonucunda ağ altyapısının yoğun bir şekilde DDoS saldırılarına maruz kaldığı ve bu saldırıların ardından ağ hizmetlerinin daha fazla kesintiye uğradığı vurgulanmıştır. Çalışmada, ağ trafiğini tespit etmek ve ayırmak için DSA ve ağ trafiğinden alınan örnek paketler ile DDoS saldırılarını tespit edebilen bir derin öğrenme modelinin kullanılması önerilmiştir. Oluşturulan DSA modelinde, öznitelik çıkarma ve sınıflandırma işlemlerini barındırdığından ve eğitildikçe kendini güncelleyen katmanlara sahip olduğundan, küçük örneklemlerde de olsa hızlı ve yüksek doğrulukla derlendiği kaydedildi [53].

Myint ve arkadaşı, saldırganların faaliyetlerini en aza indirmek için Yazılım Tanımlı Ağ mimarisinde DDoS saldırılarının tespiti için mevcut DVM'nin geliştirilmiş bir versiyonu olan Gelişmiş DVM (GDVM) tekniğini önermişlerdir. GDVM tekniği birden çok sınıftan oluşan bir sınıflandırma yöntemidir. GDVM yöntemi ile eğitim ve test süreleri kısaltılmıştır [54].

Çalışmada mininet simülasyon platformunda Yazılım Tanımlı Ağ mimarisi oluşturulmuştur. Veri düzleminde bulunan iletim cihazlarının akış tablosunda ve kontrolör ile veri düzlemi arasında yer alan

iletim hattında DDoS saldırısı gerçekleştirilmiştir. DDoS saldırılarını sınıflandırmak için dört makine öğrenme algoritması (DVM, Çok katmanlı Algılayıcı (Multiple Layer Perceptron-MLP), Karar Ağacı ve Rastgele Orman) kullanılmış [55].

Latah ve Toker yaptıkları çalışmada Yazılım Tanımlı Ağ mimarisinin getirdiği benzersiz yeniliklerden yararlanılarak geleneksel güvenlik sorunlarının çözülebileceğini belirtmişlerdir. Yazılım Tanımlı Ağ mimarisinde, veri düzleminde bulunan iletim cihazları, ağ trafiği istatistiklerini anlık olarak kontrolcüye gönderir. Yazılım Tanımlı Ağ mimarisinin bu özelliğinden yararlanarak ağ tabanlı bir saldırı tespit sistemi geliştirdiler. Çalışmada, sistemin bir bütün olarak doğruluğunda bir iyileşme sağlamak için akış istatistiklerine dayalı 5 seviyeli hibrit bir sınıflandırma sistemi önerilmiştir. Birinci seviye için K-EYK yaklaşımı, ikinci seviye için Aşırı Öğrenme Makinesi (Extreme Learning Machine-ELM) ve geri kalan seviyeler için hiyerarşik Aşırı Öğrenme Makinesi kullanılmıştır [56].

Sahoo ve arkadaşları yaptıkları çalışmada, Yazılım Tanımlı Ağ mimarisinde kontrolcüye yönelik DDoS saldırı trafiğini tespit etmeye çalışmışlardır. Tespitin doğruluğunu artırmak için DVM, Genetik Algoritma (GA) ve Çekirdek Ana Bileşen Analizi tarafından desteklenmiştir. Ayrıca önerilen model, olası saldırıları önlemek için güvenlik kurallarını tanımlayan kontrolcüye entegre edilmiştir [57].

Ujjan ve arkadaşları, önceden tanımlanmış DDoS veri kümeleri ile ağ hizmetleri ve meşru kullanıcı trafiği kalıpları için entropi tabanlı DDoS saldırı tespit ve engelleme sistemi önermiştir. Ayrıca, Shannon ve Renyi entropisi birleştirilerek, DDoS saldırı trafiğinin özellikleri, Yazılım Tanımlı Ağ kontrolcüsünün saldırının olumsuz etkilerinden daha az etkilenmesini sağlayan genelleştirilmiş entropi hesaplamasıyla belirlendi. Entropi tabanlı özellikler için imza tabanlı Snort algılama yoluyla veri düzleminden toplanan trafik verileri, derin öğrenme modellerinin algılama doğruluğunu geliştirmek için YOK ve ESA ile analiz edilmiştir [58].

Assis ve arkadaşları, merkezi olarak yönetilen bir Yazılım Tanımlı Ağ yapısına karşı gerçekleştirilebilecek siber saldırılarda ağın sürekliliğinin tehlikeye girebileceğini vurgulamıştır. Çalışmada IP akış kayıtlarının analizine dayalı bir savunma mekanizması önerilmiştir. DDoS saldırılarını tespit etmek için bir derin öğrenme yöntemi olan GTB kullanıldı. Bu yöntemde, IP akışları ayrı ayrı analiz edilir ve veriler normal veya anormal olarak sınıflandırılır. Akış kontrolü sağlandığı için saldırının Yazılım Tanımlı Ağ yapısına olumsuz etkileri en aza indirilir [59].

Gadze ve arkadaşları yaptıkları çalışmada, Yazılım Tanımlı Ağ yapısında ağı tek bir noktadan kontrol eden ve ağın beyni olarak kabul edilen kontrolcünün saldırıların ana hedefi olduğu vurgulamış. Çalışmada DDoS saldırılarını tespit etmek için UKSB ve ESA tabanlı derin öğrenme modellerini kullanılmış. Ayrıca, Derin Öğrenme yöntemlerinin performansı, geleneksel makine öğrenimi algoritmalarınınkiyle karşılaştırılmıştır. Derin Öğrenme modellerinin başarı oranlarının geleneksel makine öğrenme modellerinin algoritmalarına göre daha yüksek olduğu da çalışmada vurgulanmıştır [60]

4. DENEYSEL ÇALIŞMALAR VE BULGULAR

Bu bölümde Yazılım Tanımlı Ağ ve SD-VANET mimarisine gerçekleştirilen, DDoS saldırılarının etkin bir şekilde tespiti için makine öğrenimi ve derin öğrenmeye dayalı en iyi performansı sergileyecek modeli elde etme amacıyla yapılan deneysel çalışmalar verilmiştir.

4.1. Deneysel_Çalışma_1: DDoS Saldırılarının Klasik Makine Öğrenimi Teknikleri İle Tespiti

Bu deneysel çalışmada, Yazılım Tanımlı Ağ mimarisine karşı gerçekleştirilen DDoS saldırılarının klasik makine öğrenimi teknikleri ile tespiti için iyi performans gösteren bir model elde edilmeye gayret edilmiştir. Bu amaçla Mininet benzetim yazılımı ile bir ağ topolojisi oluşturuldu ve DDoS saldırısı gerçekleştirilerek veri toplama işlemi gerçekleştirilmiştir. Çalışma kapsamında oluşturulan veri kümesindeki özellikler, Yazılım Tanımlı Ağ mimarisinin devamlılığı için hayati önem taşıyan ana metriklerden elde edilmiştir. DDoS saldırıları sonucunda elde edilen saldırı verileri, değişik makine öğrenme teknikleri kullanılarak sınıflandırılmıştır. Veri kümesindeki öznitelik seçme yöntemleri kullanılarak en iyi özniteliklerin seçilmesi sağlanmış ve elde edilen yeni veri kümesinin sınıflandırıcıların performansları üzerindeki etkisi incelenmiştir. Makine öğrenme algoritmalarının eğitim ve test işlemleri Matlab ortamında yapılmıştır. Deneysel_çalışma_1'in son kısmında literatürdeki benzer çalışmalar özetlenerek, elde edilen sonuçlarla karşılaştırılmıştır.

4.1.1. Deneysel_çalışma_1 için veri kümesi oluşturma

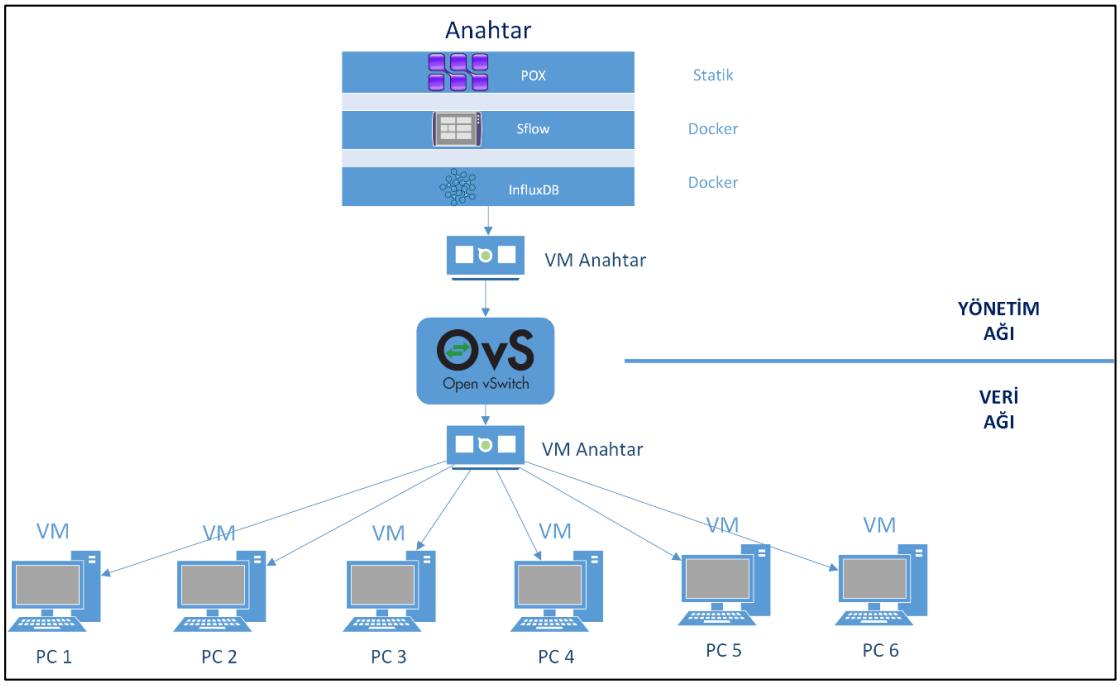
Deneysel_çalışma_1 kapsamında normal ve DDoS saldırı verilerinin toplanması için oluşturulan deneysel Yazılım Tanımlı Ağ topolojisi Şekil 4.1'de gösterilmiştir.

Topoloji, DDoS saldırısını uygulamak ve etkileri değerlendirmek için gereken minimum düğüm sayısından oluşur. Topolojide, VirtualBox-KVM üzerinde Ubuntu 18.04, 1 GB RAM, 1 CPU yapılandırmasına sahip altı (PC1-PC6) Virtual Machine/Sanal Makine (VM), iki VM anahtarımız, bir OpenvSwitch (OVS), sFlow ve InfluxDB docker imajı ve bir Phyton tabanlı açık kaynak OpenFlow/POX kontrolcüsü çalışmaktadır. Kullanıcıların bağlı olduğu VM anahtarı bridge(br0) ile OVS anahtarına bağlanmıştır. Aynı sanal makinede kurulan sFlow ve InfluxDB, bir VM anahtarı aracılığıyla OVS anahtarına bağlanmıştır. sFlow kollektörü ile saldırı anında ağ metrikleri OVS anahtarından işlenmesi için toplanmıştır

sFlowDocker imajı kullanılarak içerisinde seçilen metrikleri veri tabanına aktarılmasını sağlayan javascript kodları yazılarak, yeni bir imaj haline getirilip docker host üzerinde çalıştırılmıştır. sFlow kollektörü ile toplanan saldırı verileri, zaman tabanlı açık kaynak kodlu olan InfluxDB veri tabanı Docker imajı kullanılarak çalıştırılıp, veri tabanına toplanan veriler zaman damgası ile kaydedilmiştir.

Protokol tabanlı bir saldırı senaryosu uygulanmıştır. DDoS saldırısı "hping3" paket üretici aracı kullanılarak, saldırısı veri kümesi oluşturmak amaçlı Transmission Control Protocol (TCP), User Datagram Protocol (UDP), ve Internet Control Protocol (ICMP) flood saldırıları gerçekleştirilmiştir. Hping3 aracı ağda PC1 üzerine kurulmuş ve saldırgan olarak tanımlanmış, kurban olarak ise PC6 seçilmiştir. Saldırı altındaki 10.0.0.6 Internet Protocol (IP) adresine sahip PC6'ya, her protokol tabanlı flood saldırısında 512 bayt boyutundaki veriler saniyede 2000 paket olarak gönderilmiştir. Bu paket oranı bir kontrolcüden gelen akış değiştirme mesajlarının sayısının saniyede 1304'ten fazla olmasına yol açar, bu da akış tablosu girişlerinin sayısının 6996 olmasıyla sonuçlanır.

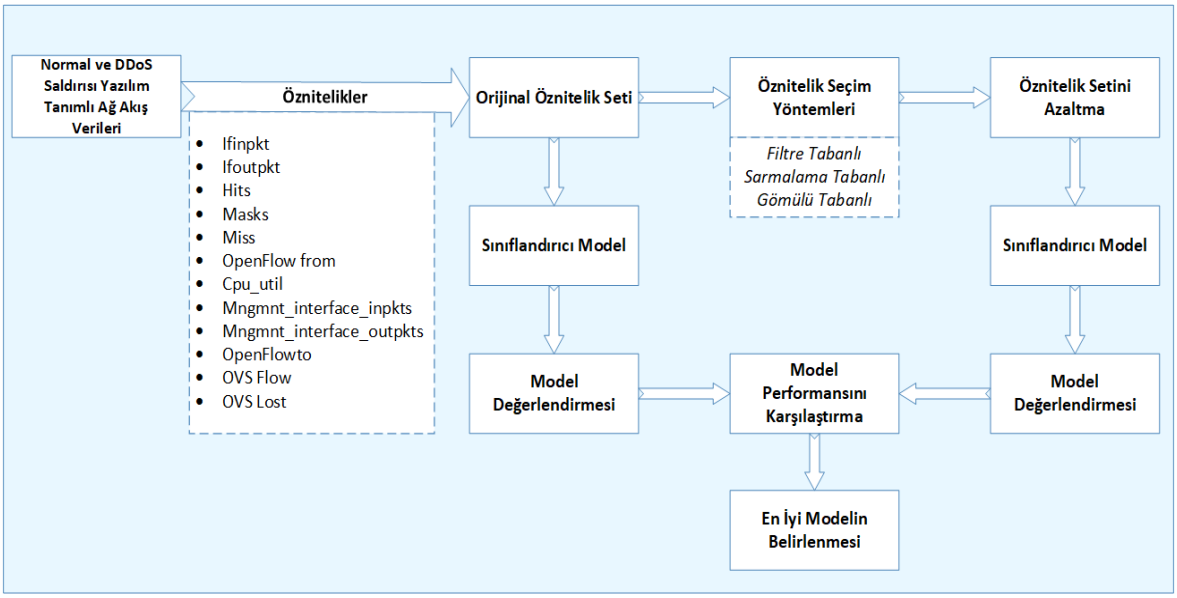
Simülasyon, TCP flood, UDP flood, ICMP flood saldırısı ve normal ağ trafiği paketlerinin her biri için 15 dakika gönderildiği 4 aşamadan oluşur. Bu dört aşama sonunda 12 öznitelik ve toplam 129000 adet örnekten oluşan bir veri kümesi elde edilmiştir. Örneklerden 64000 adeti normal ağ trafik akışına ait verilerden, geriye kalan 65000 adet veri ise DDoS saldırısı(TCP, UDP, ICMP flood) anındaki ağ trafik akışına ait verilerdir [61].



Şekil 4.1. Deneysel_çalışma _1 için oluşturulan ağ topolojisi

4.1.2. Deneysel_çalışma_1 için kullanılan öznitelik seçim yöntemleri ve makine öğrenme algoritmaları

Öznitelik seçme yöntemlerinin ve makine öğrenimi algoritmalarının uygulanmasına yönelik süreç adımları Şekil 4.2'de verilmiştir. Çalışma kapsamında oluşturulan veri kümesindeki öznitelikler, Yazılım Tanımlı Ağ mimarisinin devamlılığı için hayati önem taşıyan ana metriklerden elde edilmiştir. DDoS saldırıları ve normal trafik durumunda elde edilen veri kümesi, makine öğrenme teknikleri kullanılarak eğitilmiş ve test edilmiştir. Veri kümesinde öznitelik seçme yöntemleri kullanılarak en iyi öznitelikler seçilmiş ve elde edilen yeni veri kümesinin sınıflandırıcıların performansları üzerindeki etkisi incelenmiştir.



Şekil 4.2. Deneysel _çalışma _1 akış şeması

Çalışmada DDoS saldırılarını tespit etmek için öznitelik seçme yöntemleri ile desteklenen makine öğrenimi modelleri kullanılmıştır. Makine öğrenimi, matematiksel ve istatistiksel yöntemler kullanarak var olan verilerden çıkarımlar yapan ve bu çıkarımlarla bilinmeyenler hakkında tahminler yapan bir yöntemdir. Literatürde çeşitli makine öğrenme modelleri önerilmiştir. Modellerin taksonomisi, çekirdek tabanlı, mesafe tabanlı, sinir ağı tabanlı ve olasılık tabanlı özelliklerle özetlenebilir. Devam eden araştırmalar, sınıflandırma için evrensel iyi bir model olmadığını göstermektedir. Literatürde, DVM, K-EYK, Yapay Snir Ağı (YSA-Artificial Neural Network-ANN) ve Naive Bayes modellerinin sınıflandırma problemlerinin çözümünde iyi performans gösterdiği belirtilmiştir Çalışma kapsamında oluşturulan veri kümesi, öznitelik seçim yöntemleri kullanılarak indirgenmiş ve daha sonra bu veriler üzerinden

sınıflandırma yapılmıştır. Sınıflandırıcı olarak DVM, Naive Bayes, YSA ve K-EYK kullanılmıştır.

Öznitelik Seçimi

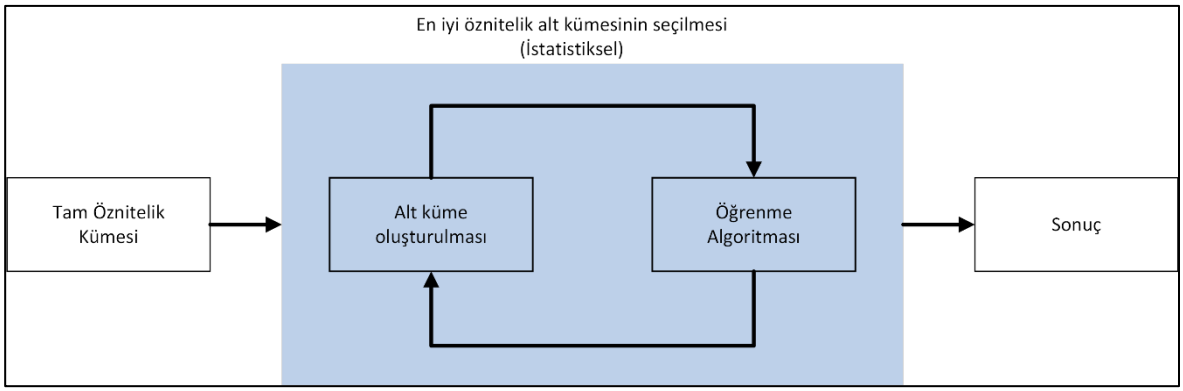
Makine öğrenimi modellerinin eğitim testinde kullanılan veri kümeleri çok sayıda öznitelik içerebilir. Bu özniteliklerden bazılarının sınıflandırma sonucu üzerinde yüksek derecede etkisi olmasına rağmen, bazı özniteliklerin sınıflandırma sonucu üzerinde çok az etkisi olabilir veya hiç etkisi olmayabilir. Sınıflandırma üzerinde çok az etkisi olan özelliklerin kullanılması, zaman ve işlem maliyetlerini artırabilir. Amaç, sınıflandırmada etkisi düşük olan öznitelikleri azaltmak ve etkinliği yüksek özniteliklerin kullanımını sağlamaktır.

Özellikle bilişim alanında oluşturulmuş veri setlerinde çok fazla sayıda öznitelik bulunmaktadır. Bu nedenle bu özniteliklerin indirgenerek yüksek değerlikteki özniteliklerin sınıflandırmada kullanılabilmesi amacıyla öznitelik seçim yöntemleri geliştirilmiştir. Bu öznitelikler algoritmaların uygulanışı bakımından filtreleme tabanlı yöntemler, sarmalama tabanlı yöntemler, gömülü tabanlı yöntemler olmak üzere 3 ana başlıkta incelenir.

Filtreleme tabanlı yöntemler öğrenme algoritmasından bağımsızdır. Sarmalama tabanlı yöntemler döngü içindeki özel öğrenme fonksiyonuyla veriyi optimize eder ve bir maliyet işlevine bağlı olarak farklı özellik alt kümelerinin performanslarını değerlendirir. Gömülü tabanlı yöntemler ise öğrenme algoritmasına yerleşik olarak çalışır ve en iyi özellik alt kümesinin bu öğrenme algoritmasının çalışması sırasında bulur.

Filtreleme Tabanlı Yöntemler

Filtre tabanlı öznitelik seçim yönteminin özniteliklerin sonuçlara ulaşılmasına katkıları istatistiksel yöntemler kullanılarak hesaplanmıştır [62]. Şekil 4.3'te verilen süreçte görüldüğü gibi derecelendirilen özniteliklerden en iyi öznitelik kümesi seçilerek indirgenmiş öznitelik kümesi oluşturulur [63].

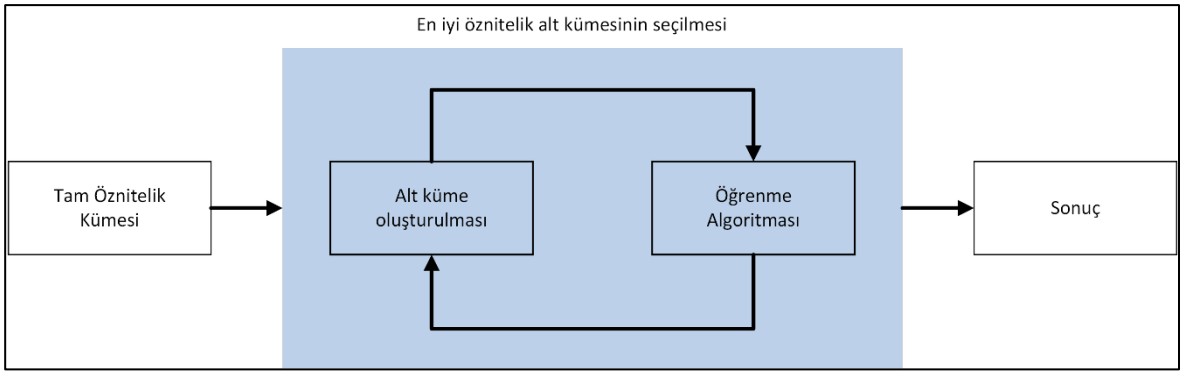


Şekil 4.3. Filtre tabanlı öznitelik seçme algoritması

Literatürde yaygın kullanılan filtreleme algoritmaları: Saklı Markov Filtreleme (Hidden Markov Filtering-SMF), Varyans Eşiği (Variance Threshold), Korelasyona Dayalı Öznitelik Seçme (Correlation based Feature Selection), Bilgi Kazancı (BK), Ki-kare, ReliefF ve Fisher Skoru şeklinde sıralanır.

Sarmalama Tabanlı Yöntemler

Sarmalama yöntemi, sınıflandırma algoritmasındaki tüm öznitelikleri deneyerek en ideal öznitelikleri bulmayı amaçlar. İdeal öznitelikler alt kümesine ulaşıldığında işlem tamamlanır ve azaltılmış bir veri kümesi oluşturulur (Şekil 4.4). Bu istatistiksel yöntemlerden daha başarılı olabilir, ancak sınıflandırma algoritmasına bağlı olarak hız açısından dezavantajlı olabilir [64-65].



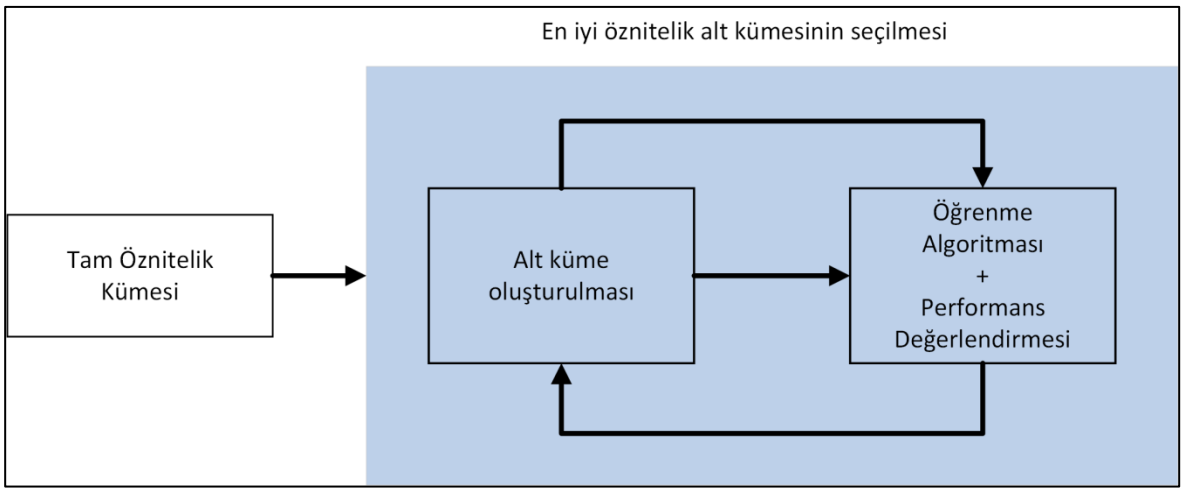
Şekil 4.4. Sarmalama tabanlı öznitelik seçme algoritması

Sarmal yöntemlerde, Sıralı İleri Seçim (Sequential Forward Selection), Sıralı Geri Seçim (Sequential Backward Selection), Sıralı İleri Kayan Seçim (Sequential Forward Floating

Selection), Sıralı Geri Kayan Seçim (Sequential Backward Floating Selection) gibi sıralı arama algoritmaları ve sezgisel arama algoritmaları kullanılır.

Gömülü Tabanlı Yöntemler

Gömülü tabanlı öznitelik seçim yöntemleri, sınıflandırma algoritmasında yer alan özellik seçim algoritmalarıdır. Bu algoritmalar öznitelik seçimlerini modelin doğruluğuna en iyi katkı sağlayacak öznitelikleri belirleyerek yaparlar (Şekil 4.5). Filtre ve sarmalama tabanlı yöntemlerin avantajlarını birleştirmek için geliştirilmiştir. Bu yöntemde, bir öğrenme algoritması değişken seçim sürecinden yararlanır ve eş zamanlı olarak özellik seçimi ve sınıflandırma gerçekleştirir [66].



Şekil 4.5. Gömülü tabanlı öznitelik seçme algoritması

Makine öğrenimi Sınıflandırıcıları

Deneysel_çalışma_1'de literatürde yaygın olarak kullanılan dört farklı makine öğrenimi algoritması kullanılmıştır: DVM, Naive Bayes, YSA ve K-EYK. Veri kümelerinin sınıflandırılması için evrensel bir en iyi sınıflandırma algoritması kesin olarak önerilemez. Sınıflandırıcılar, farklı veri kümelerinde farklı performanslar sergileyebilir. K-EYK, DVM düşük hiperparametreli, kararlı ve düşük performanslı algoritmalardır.

K-EYK algoritması, sınıflandırma ve regresyon problemlerini çözmek için kullanılan karmaşık olmayan, uygulaması kolay ve verimli bir denetimli öğrenme algoritmasıdır. Algoritma, gerçek zamanlı kullanım sırasında girişteki değişikliklere hızlı bir şekilde yanıt verir. Çok sınıflı bir

problem için uygulanması çok kolaydır. Ayrıca K-EYK modelini oluştururken farklı mesafe metriklerini seçme esnekliği sağlar. Algoritmaya yeni veri girildiğinde ilk önce en yakın K komşusunu kontrol ederek yeni verinin sınıfına karar verir. K-EYK sınıflandırma performansı, komşu sayısı, mesafe metriği ve mesafe ağırlığı gibi hiperparametrelerden büyük ölçüde etkilenir. K-EYK sınıflandırıcısının hiperparametrelerinin arama aralıkları Çizelge 4.1'de listelenmiştir [67].

Çizelge 4.1. K-EYK sınıflandırıcısı için hiperparametreler ve arama aralıkları

Hiperparametreler	Arama Aralıkları
Komşu Sayısı	1-8880
Mesafe Metriği	City block, Chebyshev, Correlation, Cosine, Euclidean, Hamming, Jaccard, Mahalanobis, Minkowski, Spearman
Mesafe Aralığı	Eşit, Ters, Kare Ters

DVM, istatistiksel öğrenme teorisine dayalı sınıflandırma ve regresyon problemleri için kullanılabilen başka bir denetimli makine öğrenme algoritmasıdır. Optimizasyon tabanlı bir algoritma olduğu için sınıflandırma performansı oldukça başarılıdır. Hem doğrusal hem de doğrusal olmayan şekilde ayrılabilen veriler için uygun bir algoritmadır. DVM'nin çalışma prensibi, lineer olarak ayrılabilir giriş vektöründe iki sınıfı birbirinden ayırabilen bir hiper düzlem bulma prensibine dayanır. Lineer olarak ayrılamayan veri setlerinde ise giriş vektörünü lineer olarak ayrılabilir hale getiren bir kernel fonksiyonu kullanılarak verilerin daha yüksek boyutlu bir özellik uzayına dönüştürülmesi prensibine dayanmaktadır [68]. DVM sınıflandırma performansı, çekirdek işlevi, çekirdek ölçeği ve kutu kısıtlama düzeyi gibi hiperparametrelerden etkilenir. DVM sınıflandırıcısının hiperparametrelerinin arama aralıkları Çizelge 4.2'de listelenmiştir [69].

Çizelge 4.2. DVM sınıflandırıcısı için hiperparametreler ve arama aralıkları

Hiperparametreler	Arama Aralıkları
Kutu kısıtlama düzeyi	0,001-1000
Çekirdek ölçeği	0,001-1000
Çekirdek işlevi	Gaussian, Linear, Quadratic, Cubic

Naive Bayes sınıflandırıcısı, olasılık ilkelerine göre tanımlanmış bir dizi hesaplama ile sisteme girilen verilerin sınıfını yani kategorisini tespit etmeyi amaçlar [70]. Naive Bayes sınıflandırıcısında belli bir miktar eğitim verisi sisteme girilir. Eğitim için girilen verilerin bir sınıfı olmalıdır. Olasılık işlemleri ile sisteme girilen test verileri eğitim verileri üzerinde

gerçekleştirilir. Bu işlem önceden elde edilen olasılık değerlerine göre yapılır ve ardından verilen test verilerinin sınıfı belirlenir. Eğitim verisi sayısı ne kadar fazlaysa, test verisinin gerçek kategorisini tespit etmek o kadar doğru olur.

YSA, yapay nöron modeline dayalı güçlü bir sınıflandırma aracıdır. En temel yapı taşı olarak nöronları kullanır. Yapay nöronlar, biyolojik nöronlara benzer şekilde davranmak üzere tasarlanmıştır. Bu çalışmada oluşturulan YSA mimarisi, tek bir gizli katmanda 12 girdi birimi ve 10 nöron içermektedir. Unutulmamalıdır ki, öznitelik seçiminde kullanılan filtre yöntemlerine göre tek bir gizli katmandaki girdi sayısı ve nöron sayısı değişmektedir.

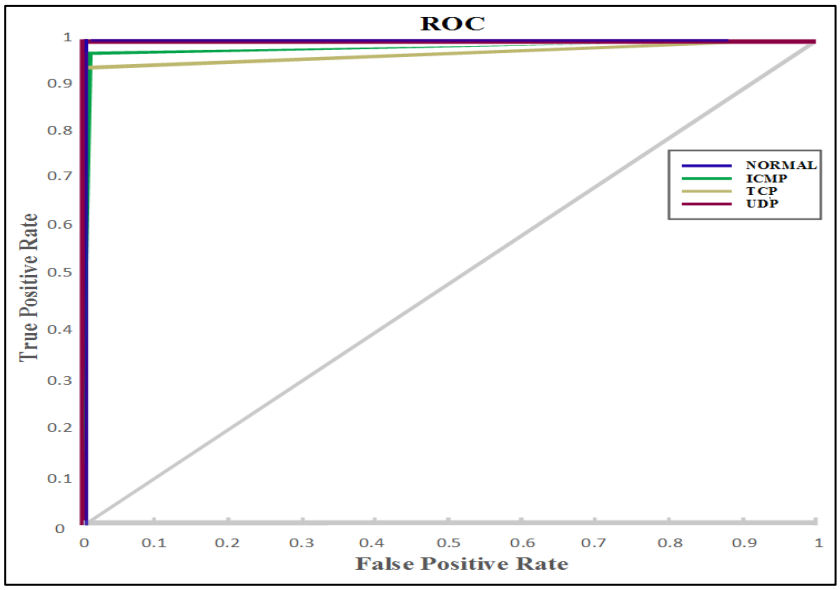
4.1.3. Deneysel_Çalışma_1 sonuçları

Çalışmanın bu aşamasında öznitelik seçim yöntemleri kullanılmamıştır. Elde edilen analiz sonuçları Çizelge 4.3'te sunulmuştur.

Çizelge 4.3. Özellik seçme yöntemi olmadan makine öğrenimi modellerinin performansı

Sınıflandırıcı	Doğruluk()	Duyarlılık(%)	Özgünlük(%)	Kesinlik(%)	F1_puanı(%)
DVM	92,11	88,71	96,93	91,42	89,91
K-EYK	95,67	93,87	98,01	97,05	95,30
YSA	91,07	87,27	96,58	89,89	88,45
Naive Bayes	94,48	91,77	98,29	92,94	91,79

Başarı oranları incelendiğinde K-EYK ve Naive Bayes, Yazılım Tanımlı Ağ verilerini analiz etmede daha iyi performans sergilemiştir. K-EYK algoritması ile gerçekleştirilen eğitim sonunda Receiver Operating Characteristic (ROC) eğrisi hazırlandı (Şekil 4.6). ROC eğrisi, test sonuçlarını değerlendirmek için kullanılan bir araçtır. Y ekseni olarak True Positive (TP) ve X ekseni olarak False Positive (FP) olan iki boyutlu bir grafikle tanımlanır. TP, pozitif çıktılar olarak doğru bir şekilde sınıflandırılan pozitif örnekleri gösterir. FP, yanlışlıkla pozitif çıktılar olarak sınıflandırılan negatif örnekleri gösterir. Karışıklık matrislerinde (confusion matrices) kullanılan başka bir ölçüm, yanlış bir şekilde negatif çıktı olarak sınıflandırılan pozitif örnekleri gösteren False Negative (FN)’dir. ROC eğrisinde TP değerinin %0,9'un üzerinde olduğu gözlendi.



Şekil 4.6. Öznitelik seçimi yapılmadan K-EYK için normal, ICMP, TCP ve UDP ROC eğrileri

Performans sonuçları incelendiğinde K-EYK sınıflandırıcısının saldırı verilerini analiz etmede daha başarılı olduğu görülmüştür. K-EYK sınıflandırıcısından elde edilen analiz sonuçlarına göre Çizelge 4.4'de gösterilen karışıklık matrisi hazırlanmıştır. Saldırı trafiği türünün (ICMP, TCP ve UDP) performans oranını etkilediğini gözlemledi. En yüksek performans oranı UDP flood saldırısı yapıldığında elde edilmiştir.

Çizelge 4.4. Öznitelik seçimi olmadan K-EYK sınıflandırıcısının karışıklık matrisi

		Tahmin Sınıfı					
		Normal(%)	ICMP(%)	TCP(%)	UDP(%)	TP(%)	FN(%)
Sınıf	Normal	99,61	0,07	0,11	0,18	99,6	0,4
	ICMP	7,31	90,99	1,68		91,0	9,0
	TCP	12,78	1,15	85,63	0,41	85,6	14,4
	UDP	0,44	0,33		99,21	99,2	0,8

Deneysel_çalışma_1'in bu aşamasında makine öğrenmesi modellerinin tahmininde en güçlü etkiye sahip öznitelikleri elde etmek için filtreleme, sarmalama ve gömülü tabanlı öznitelik seçimi yöntemleri uygulanmıştır. Öznitelik seçme yöntemleri ile elde edilen indirgenmiş veri kümesi her bir modele uygulanmış ve modelin doğruluğu test edilmiştir.

DVM, K-EYK, YSA ve Naive Bayes modellerine filtreleme yöntemi olarak Relief algoritması uygulanmıştır. Bu algoritma veri kümesindeki her örneğin kendi sınıfındaki diğer örneklere yakınlığı ve farklı sınıflara uzaklığı hesaplanmıştır. Bu hesaplama sonucunda bir model oluşturulmuş ve

öznitelik seçimi yapılmıştır. Bu öznitelikler, her sınıflandırma algoritmasıyla yeniden eğitildi. Sonuçlar Çizelge 4.5'te özetlenmiştir.

Çizelge 4.5. Makine öğrenimi modellerinin filtre tabanlı öznitelik seçim yöntemi ile performansı

Sınıflandırıcı	Doğruluk()	Duyarlılık(%)	Özgünlük(%)	Kesinlik(%)	F1_puanı(%)
DVM (10 Öznitelik)	92,46	89,13	97,02	92,02	90,43
K-EYK (6 Öznitelik)	97,15	95,88	98,68	98,10	96,92
YSA (6 Öznitelik)	92,28	89,02	96,99	91,62	90,20
Naive Bayes (8 Öznitelik)	95,70	93,49	98,65	95,07	93,60

Filtreleme yöntemi olarak Relief algoritması uygulanıp öznitelik seçimi yapıldığında en yüksek başarım oranı K-EYKsınıflandırıcısı ile elde edilmiştir (Çizelge 4.5). En yüksek performans oranına sahip sınıflandırıcının karışıklık matrisi Çizelge 4.6'da verilmiştir.

Çizelge 4.6. Filtre tabanlı özellik seçimine sahip K-EYK sınıflandırıcısının karışıklık matrisi

		Tahmin Sınıfı					
		Normal(%)	ICMP(%)	TCP(%)	UDP(%)	TP(%)	FN(%)
Sınıf	Normal	99,84	0,07	0,07		99,8	0,2
	ICMP	7,20	91,66	1,12		91,7	8,3
	TCP	6,60	0,94	92,45		92,5	7,5
	UDP	0,11	0,11	0,22	99,55	99,6	0,4

Sarmalama öznitelik seçim yöntemi ile DVM, K-EYK, YSA ve Naive Bayes modellerine Sıralı İleri Kayan Seçim algoritması uygulanmıştır. Bir özellik tek başına hatayı azaltmayabilir ancak bu özelliğin başka bir özellikle göre değerlendirilmesi sınıflandırma performansını iyileştirebilir. Bu amaçla Floating algoritmalar geliştirilmiştir. Bu algoritmalar kayan bir değişkene sahiptir ve her aşamada eklenen veya kaldırılan özelliklerin sayısı sabit değildir. Bu, çok sayıda özellik altkümesi kombinasyonu oluşturur [71]. Sıralı Özellik Seçimi, özelliği kullanıcı tanımlı bir sınıflandırıcı performans metriğine göre ekler veya çıkarır.

Sarmalama tabanlı öznitelik seçim algoritmasında, sıralı ileri kayan seçim algoritması ile öznitelikler indirgendi. İndirgenmiş veri kümesi, her sınıflandırma algoritması için yeniden düzenlendi. Sonuçlar Çizelge 4.7'de gösterilmiştir.

Çizelge 4.7. Makine öğrenimi modellerinin sarmalama tabanlı özellik seçim yöntemi performansı.

Sınıflandırıcı	Doğruluk(%)	Duyarlılık(%)	Özgünlük(%)	Kesinlik(%)	F1_puanı(%)
DVM (8 Öznitelik)	92,15	90,20	97,26	90,23	90,21
K-EYK (6 Öznitelik)	98,30	97,73	99,45	97,72	97,70
ANN (10 Öznitelik)	91,44	87,82	97,31	88,11	87,89
Naive Bayes (8 Öznitelik)	94,87	92,05	98,43	93,29	92,01

En yüksek başarım oranı, sıralı ileri kayan seçim algoritması algoritması kullanılarak sarmalama yöntemi uygulandığında ve öznitelik seçimi yapıldığında K-EYK sınıflandırıcısı ile elde edilmiştir (Çizelge 4.7). En yüksek performans oranına sahip sınıflandırıcının karışıklık matrisi Çizelge 4.8'de gösterilmektedir.

Çizelge 4.8. Sarmalama tabanlı özellik seçimine sahip K-EYK sınıflandırıcısının karışıklık matrisi

		Tahmin Sınıfı					
		Normal(%)	ICMP(%)	TCP(%)	UDP(%)	TP(%)	FN(%)
Sınıf	Normal	99,58	0,15	0,15	0,11	99,6	0,4
	ICMP	0,56	97,97	1,12	0,33	98,0	2,0
	TCP	0,52	5,45	94,02		94,0	6,0
	UDP	0,22	0,11	0,33	99,32	99,3	0,7

Son olarak gömülü tabanlı öznitelik seçim algoritmalarından olan Lasso algoritması kullanılarak öznitelik seçimi yapılmıştır. Kullanılan yöntem tek tek DVM, K-EYK, YSA ve Naive Bayes modellerine uygulanmıştır. Lasso, doğrusal regresyonun başka bir düzenlileştirilmiş çeşididir. Lasso regresyonunun önemli özelliği, en önemsiz özelliklerin ağırlıklarını ortadan kaldırmasıdır. Bu şekilde, doğal olarak öznitelik seçimi yapar ve azaltılmış bir öznitelik seti çıkarır. İndirgenmiş veri kümesi, her sınıflandırma algoritması için yeniden düzenlendi. Sonuçlar Çizelge 4.9'da gösterilmektedir.

Çizelge 4.9. Makine öğrenimi modellerinin gömülü özellik seçim yöntemi performansı

Sınıflandırıcı	Doğruluk(%)	Duyarlılık(%)	Özgünlük(%)	Kesinlik(%)	F1_puanı(%)
DVM (10 Öznitelik)	92,46	90,73	97,41	90,49	90,60
K-EYK (8 Öznitelik)	96,30	94,95	98,85	95,09	94,80
YSA (6 Öznitelik)	92,09	88,91	97,42	89,22	89,06
Naive Bayes (10 Öznitelik)	95,09	93,34	98,45	93,44	93,18

Gömülü yöntem olan Lasso uygulanıp öznitelik seçimi yapıldığında en yüksek başarım oranı K-EYK sınıflandırıcısı ile elde edilmiştir (Çizelge 4.9). En yüksek performans oranına sahip sınıflandırıcının karışıklık matrisi Çizelge 4.10'da gösterilmektedir.

Çizelge 4.10. Gömülü özellik seçimine sahip K-EYK sınıflandırıcısının karışıklık matrisi

		Tahmin Sınıfı					
		Normal(%)	ICMP(%)	TCP(%)	UDP(%)	TP(%)	FN(%)
Sınıf	Normal	99,39	0,15	0,33	0,11	99,4	0,6
	ICMP	0,78	97,40	1,35	0,45	97,4	2,6
	TCP	0,41	15,30	84,06	0,20	84,1	15,9
	UDP	0,33	0,22	0,44	98,98	99,0	1,0

Çalışma boyunca tüm veri kümesi öznitelikleri kullanılmış ve indirgeme için iki farklı yöntem kullanılmıştır (Çizelge 4.11). Tüm özniteliklerin kullanıldığı sınıflandırma sonuçları kabul edilebilir olmakla birlikte öznitelik seçme yöntemlerinin kullanılmasıyla performans sonuçları artmıştır.

Çizelge 4.11. Öznitelik seçme yöntemlerinin kullanımı ile makine öğrenimi modeli performanslarının karşılaştırılması

Sınıflandırıcı	Metot	Öznitelik sayısı	Doğruluk	Duyarlılık	Özgünlük	Kesinlik	F1_puanı
DVM	-	12	92,11	88,71	96,93	91,42	89,91
	Filtre	10	92,46	89,13	97,02	92,02	90,43
	Sarmalama	8	92,15	90,20	97,26	90,23	90,21
	Gömülü	10	92,46	90,73	97,41	90,49	90,60
K-EYK	-	12	95,67	93,87	98,01	97,05	95,30
	Filtre	6	97,15	95,88	98,68	98,10	96,92
	Sarmalama	6	98,30	97,73	99,45	97,72	97,70
	Gömülü	8	96,30	94,95	98,85	95,09	94,80
YSA	-	12	91,07	87,27	96,58	89,89	88,45
	Filtre	6	92,28	89,02	96,99	91,62	90,20
	Sarmalama	10	91,44	87,82	97,31	88,11	87,89
	Gömülü	6	92,09	88,91	97,42	89,22	89,06
Naive Bayes	-	12	94,48	91,77	98,29	92,94	91,79
	Filtre	8	95,70	93,49	98,65	95,07	93,60
	Sarmalama	10	94,87	92,05	98,43	93,29	92,01
	Gömülü	10	95,09	93,34	98,45	93,44	93,18

Çizelge 4.11'de gösterilen test verilerinin makine öğrenimi modeline uygulanmasıyla elde edilen en yüksek doğruluk oranı, sarmalama öznitelik seçimi yöntemi uygulandığında K-EYK sınıflandırıcısı (%98,3) ile elde edilmiştir. Bu doğruluk oranı, altı özellik üzerinde eğitim ve test yoluyla elde edildi. 12 özellik içeren bir veri kümesi üzerinden eğitim sınıflandırıcıları

tarafından elde edilen performans oranlarının, öznitelik seçim algoritmaları kullanılarak arttığını gözlemledik.

4.1.4. Deneysel_Çalışma_1 sonuçlarının literatürdeki benzer çalışmalarla karşılaştırılması

Deneysel çalışma sonunda elde ettiğimiz sonuçlar üçüncü bölümde özetlenen benzer literatür çalışmalarıyla karşılaştırılmıştır. Deneysel Sonuçlar ve özetlenen literatür çalışmalarına ait veriler Çizelge 4.12'de özetlenmiştir.

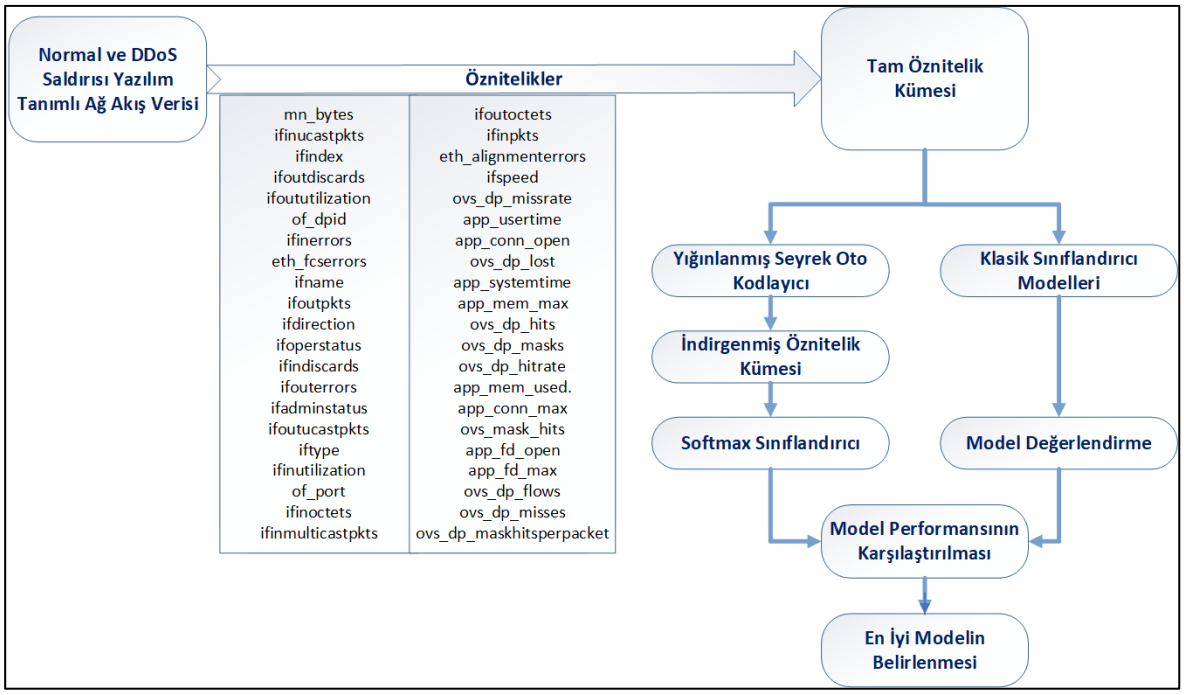
Çizelge 4.12. Deneysel sonuçların literatürdeki benzer çalışmalarla karşılaştırılması

Veri kümesi	Öznitelik Seçim Algoritmaları	Sınıflandırma Algoritmaları	Doğruluk (%)	Ref.
NSL-KDD VERİSETİ(2009)	Öznitelik Seçimi Yok	SMO	97,31	[32]
		J48	98,78	
		Naive Bayes	95,11	
	Genetic Alg.+ CFS	SMO	96,85	
		J48	98,33	
		Naive Bayes	82,62	
	Greedy Alg. + CFS	SMO	95,71	
		J48	92,74	
		Naive Bayes	83,92	
Kendi veri kümleri	Öznitelik Seçimi Yok	DVM	96,25	[33]
Kendi veri kümleri	Öznitelik Seçimi Yok	DVM	Ortalama 95,24	[34]
LongTail	Öznitelik Seçimi Yok	C4.5 Bayes Ağı Karar Ağacı Naive Bayes	Ortalama 91,68	[35]
Kendi veri kümleri	Öznitelik Seçimi Yok	SOM LVQ	Ortalama 94	[36]
CAIDA DDoS 2007	Öznitelik Seçimi Yok	DVM	92,9	[37]
Veri kümemiz	Filtre	DVM	92,46	
		K-EYK	97,15	
		YSA	92,28	
		Naive Bayes	95,70	
	Sarmalama	DVM	92,15	
		K-EYK	98,30	
		YSA	91,44	
		Naive Bayes	94,87	
	Gömülü	DVM	92,46	
		K-EYK	96,30	
		YSA	92,09	
		Naive Bayes	95,09	

Çizelge 4.12'de, öznitelik seçme yaklaşımlarıyla önerilen makine öğrenimi modelinin [32-37] sonuçlarıyla karşılaştırıldığında daha iyi performans sağladığı görülebilir. Literatürdeki benzer çalışmaların farklı veri kümeleri ve farklı modeller kullandığı unutulmamalıdır. Bu nedenle, karşılaştırma sonuçlarının gerekçelendirilmesi zordur. Sonuçlar, Yazılım Tanımlı Ağ yapısının DDoS saldırılarını makine öğrenme teknikleri ile tespit etme konusunda başarılı olduğunu göstermektedir. Yazılım Tanımlı Ağ mimarisi üzerinde planlanacak yaklaşımlar ile ağ üzerinde güvenli ve verimli bir mekanizma geliştirilebilir. Büyük ağlardaki kontrolcülerin performansı, trafikteki paketlerin yakalanması ve paketin önceden öğrenilen akış verileriyle karşılaştırılmasıyla arttırılabilir. Diğer bir yaklaşım ise veri kümesine yönelik DDoS saldırılarının tespitinde iki aşamalı bir sistemin kullanılması olabilir. Gelen akış verileri kontrolcü tarafından işlenirken başka bir modüle kopyalanarak bu modül tarafından analiz edilir. Ağ üzerinde herhangi bir çıktının olması için modülün gelen verilerde bir sorun olup olmadığını kontrolcüye bildirmesi gerekir. Böylece kontrolcü sadece ağ işlemlerini gerçekleştirirken, modül saldırı tespit sistemi haline gelir.

4.2. Deneysel _Çalışma _2: Derin Ağ Yaklaşımı ile DDoS Saldırılarının Tespiti

Deneysel_çalışma_2'de, SD-VANET'leri hedef alan DDoS saldırılarını tespit etmek için Yığınlanmış Seyrek Oto Kodlayıcı (YSOK-Stacked Sparse Autoencoder-SSAE)+Softmax sınıflandırıcılı derin ağ modeli oluşturulmuştur. İlk olarak Yazılım Tanımlı Ağ tabanlı VANET'ten normal ağ trafiğini ve saldırı trafiğini içeren bir veri kümesi oluşturulmuştur. Veri kümesindeki öznitelikler, SD-VANET mimarisinin sürekliliği için hayati öneme sahip ana metriklerden elde edilmiştir. Oluşturulan derin ağ modelinde Softmax sınıflandırıcı katmanının önünde bir YSOK yapısı kullanılmıştır. YSOK'un temel amacı, n-boyutlu bir özellik vektörünü minimum kayıpla daha küçük boyutlu bir özellik vektörüne indirgemektir. Klasik yöntemlerle öznitelik boyutunun küçültülmesi ve önemli özniteliklerin elde edilmesi son derece zordur. Bu işlemi otomatik olarak gerçekleştiren bir YSOK yapısı daha etkin sonuçlar verebilir. YSOK ile elde edilen düşük boyutlu ve yüksek öneme sahip öznitelikler Softmax sınıflandırıcıya girdi olarak verilmiş, ardından normal trafik ve DDoS saldırı trafiği sınıflandırılmıştır. Çalışmanın genel akış şeması Şekil 4.7'de ayrıntılı olarak sunulmaktadır. Çalışmada önerilen modelin teorik alt yapısı da bu bölümde detaylandırılmıştır.

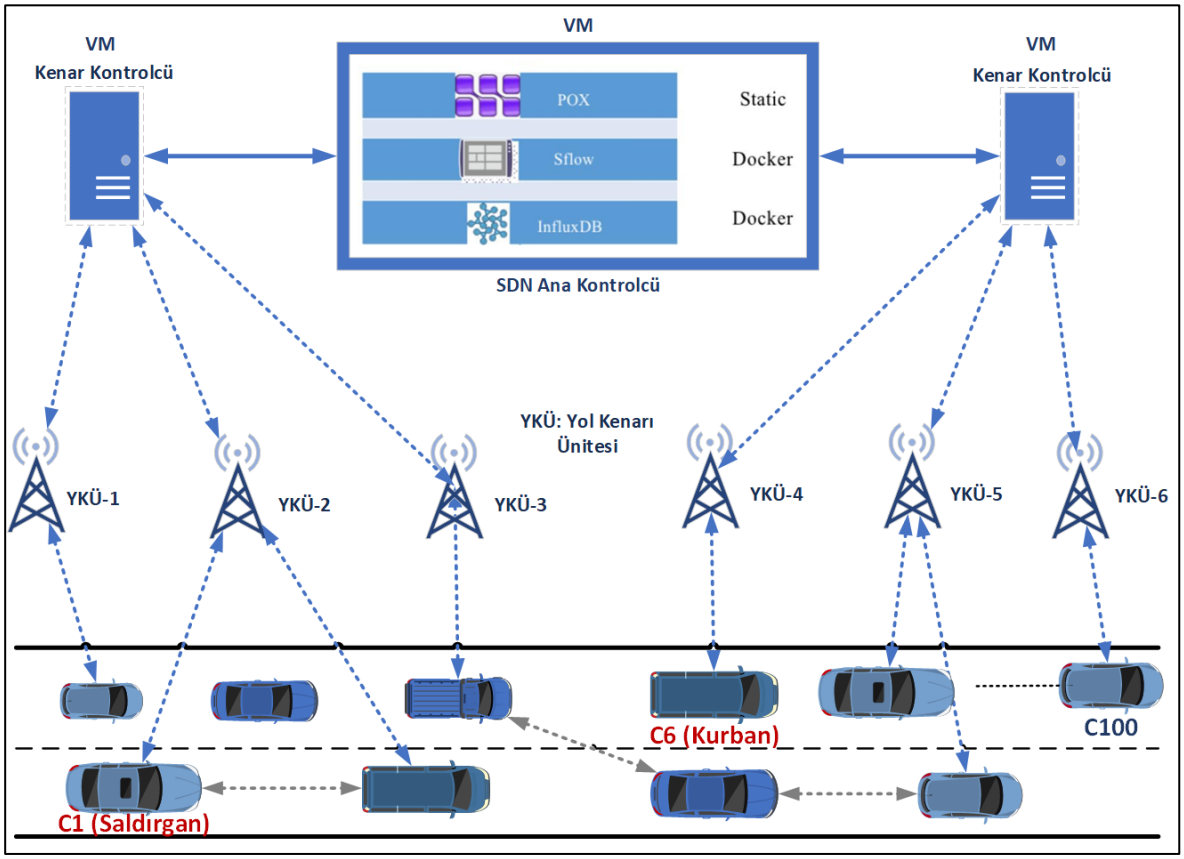


Şekil 4.7. Deneysel_çalışma_2 genel akış diyagramı

4.2.1. Deneysel_çalışma_2 için oluşturulan veri kümesi

Normal ağ trafiği verilerini ve DDoS saldırı verilerini toplamak için kullanılan deneysel SD-VANET topolojisi, SUMO simülatörü üzerinde araç trafiği üretilerek geliştirilmiştir. Yollara ilişkin bilgiler, 1000m × 1000m'lik bir alan içindeki trafik ışıkları, hız limitleri, yol yönleri ve kavşaklara ilişkin verileri kapsar ve OpenStreetMap'den (OSM) küçültülür. SD-VANET topolojisi Şekil 4.8'te sunulmuştur. Yollarla ilgili tüm bilgileri içeren .osm dosyası daha sonra işlenerek, SUMO simülatörü kullanılarak izleme dosyaları elde edilmiştir. Bu dosyalar daha sonra Mininet-WiFi'ye entegre edilmiştir. Topoloji 6 adet Yol Kenarı Ünitesi, 100 (C1, C2..., C100) araç, iki adet kenar kontrolcü ve bir ana kontrolörden oluşmaktadır. POX kontrolcüsü, sFlow ve InfluxDB aynı sanal makineye (Virtual Machine-VM) (Ubuntu 18.04 1GB RAM 1 CPU) yüklendi ve VM kenar kontrolcüler aracılığıyla Yol Kenarı Ünitelerine bağlandı. sFlow kollektör ile, ağ metrikleri saldırı anında Yol Kenarı Üniteleri aracılığıyla toplandı. sFlow kollektör ile toplanan saldırı ve normal trafik verileri, açık kaynaklı InfluxDB veritabanı kullanılarak bir zaman damgasıyla kaydedildi. Çalışmada çoklu kullanıcı kullanımının temel amacı, Yol Kenarı Üniteleri üzerindeki kontrolcünün kontrol süresini ve veri iletim hızını artırmaktır. SD-VANET ağlarında birden fazla kenar kontrolcünün kullanılması kaçınılmazdır. Çalışma, birden çok

kontrolcünün kullanıldığı ağlara yönelik gerçekleşen DDoS saldırılarının ağda meydana getirdiği sorunları göstermeyi amaçlamaktadır [72].



Şekil 4.8. Deneysel_çalışma_2 için oluşturulan SD-VANET topolojisi

Bu deneysel çalışma, protokol tabanlı DDoS saldırıları üretilerek gerçekleştirilmiştir. Değerlendirme aşamasında oluşturulan veri kümesi, normal ağ trafiği akışına ilişkin 6808 veri ve DDoS saldırısı anındaki ağ trafiği akışına ilişkin 10971 veriden oluşmaktadır. Hping3 paket üretici aracı kullanılarak, TCP, UDP ve ICMP flood saldırıları gerçekleştirilmiştir. Bu araç deneysel çalışmada saniyede 2000 paket hızında DDoS saldırı trafiği üretmektedir.

Şekil 4.8'de görüldüğü gibi, TCP, UDP, ICMP flood saldırısı, "hping3" paket üreteci ile saldırgan C1 aracılığıyla, 10.0.0.6 IP adresine sahip kurban C6 aracına yönelik gerçekleştirilmiştir. 1 saniyede (1000 ms) 512 bayt boyutunda 2000 paket saldırgan tarafından kurbana gönderilmiştir. Bu saldırı ağda akış kaybına ve kontrolcüye gönderilen Packet_In mesaj sayısında büyük artışa sebep olur.

Simülasyon 4 senaryodan oluşmaktadır.

Senaryo 1: Saldırgan C1'den kurban C6'ya 30 dakika boyunca saniyede 512 bayt boytunda 2000 paket gönderilerek TCP flood saldırısı gerçekleştirilmiştir.

Senaryo 2: Saldırgan C1'den kurban C6'ya 30 dakika boyunca saniyede 512 bayt boytunda 2000 paket gönderilerek UDP flood saldırısı gerçekleştirilmiştir.

Senaryo 3: Saldırgan C1'den kurban C6'ya 30 dakika boyunca saniyede 512 bayt boytunda 2000 paket gönderilerek ICMP flood saldırısı gerçekleştirilmiştir.

Senaryo 4: C1 kullanıcısı son senaryoda yasal kullanıcı olarak kullanılmıştır. C1 kullanıcından sırayla TCP, UDP ve ICMP protokolleri kullanılarak normal ağ trafiği C6 kullanıcısına doğru oluşturulmuştur. Her protokol için simülasyon 30 dakika çalıştırılmıştır.

Gerçekleştirilen senaryolar kontrolcüye gönderilen akış değiştirme mesaj sayısının saniyede 3357'den fazla olmasına neden olmuştur. Kontrolcüye gönderilen akış değiştirme mesaj sayısındaki artış nedeniyle akış tablosu girdi sayısının saniyede18010'fazla olmasına neden olur. Senaryolar sonucunda elde edilen veri kümesi 42 öznitelik ve toplam 17779 örnekten oluşmaktadır. Örneklerin 6808'i normal ağ trafiği akışının verileridir. Kalan 10971 örnek, DDoS saldırısı sırasında ağ trafiği akışından alınan verilerdir [72].

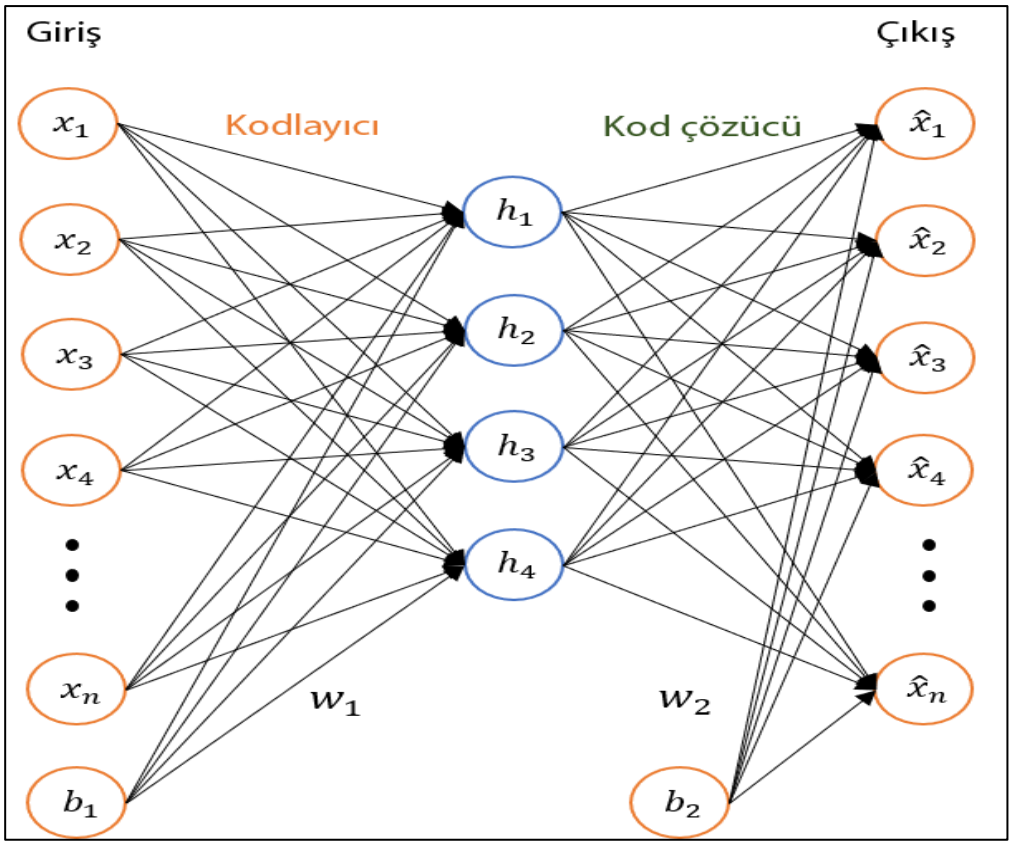
4.2.2. Deneysel_çalışma_2 için kullanılan makine öğrenme algoritmaları

Deneysel_çalışma _2 kapsamında, SD-VANET mimarisini hedef alan DDoS saldırılarını tespit etmek Yığınlanmış Seyrek Oto Kodlayıcı (YSOK-Stacked Sparse Autoencoder-SSAE)+Softmax sınıflandırıcı derin ağ modeli modeli önermiştir. Önerilen modelde kullanılan yöntemlerin detayları alt bölümlerde yer almaktadır.

Yığınlanmış Seyrek Oto Kodlayıcı (Stacked Sparse Autoencoder)

Oto Kodlayıcılar (Autoencoders) , daha az önemli özellikleri ortadan kaldırarak yüksek boyutlu özellikleri düşük bir boyuta indirgemek için diğer özellik boyut küçültme

tekniklerinin yerine giderek daha fazla kullanılmaktadır. Oto kodlayıcılar kullanılarak veri tabanlı öğrenme yoluyla elde edilen özellikler, verileri daha başarılı bir şekilde temsil edebilir. Oto kodlayıcı, yapay sinir ağı modeline dayalı denetimsiz bir öğrenme yöntemidir [73,74]. Bir oto kodlayıcı, üç katmanlı ileri beslemeli bir sinir ağından oluşur: giriş katmanı, gizli katman ve çıkış katmanı (Şekil 4.9). Bir oto kodlayıcının giriş katmanındaki ve çıkış katmanındaki nöron sayıları eşittir. Ayrıca gizli katmanın boyutu, girdi ve çıktı katmanlarının boyutundan daha küçüktür. Oto kodlayıcıya girdi olarak sağlanan veriler, çıktı katmanında yeniden oluşturulur. Oto kodlayıcı iki bölüm içerir: kodlayıcı ve kod çözücü. Bu iki bölüm tek bir mimari olarak birlikte eğitilir. Eğitim bittikten sonra kodlayıcı ve kod çözücü modelleri ayrı ayrı kullanılabilir.



Şekil 4.9. Üç katmanlı oto kodlayıcı yapısı

Bir kodlayıcı ve bir kod çözücü bölümünden oluşan üç katmanlı temel bir oto kodlayıcı yapısı Şekil 4.9'da gösterilmektedir. Burada , $x_i, h_{1-4}, \hat{x}_i$ *and* $b_{1,2}$ değerleri sırasıyla giriş katmanını, gizli katmanı (özellikleri), bir çıkış katmanını ve bias değerlerini temsil etmektedir.

Kodlama aşamasında Denklem (4.1)'e göre giriş x_i değerinden öznitelikler çıkarılır.

$$h_i = s_1(W_1 x_i + b_1), \quad (4.1)$$

Burada W_1 kodlama ağırlık matrisini temsil eder, s_1 gizli katman nöronları için aktivasyon fonksiyonudur, b_1 ise kodlayıcı yanlılık vektörüdür. Bu çalışmada kodlayıcıda lojistik sigmoid aktivasyon fonksiyonu kullanılmıştır. Kod çözme aşamasında, orijinal giriş x_i değerinin bir tahmini, çıkarılan özniteliklere göre oluşturulur (Denklem 4.2).

$$\hat{x}_i = s_2(W_2 h_i + b_2), \quad (4.2)$$

W_2 şifre çözme ağırlık matrisini temsil ederken, s_2 çıkış katmanı nöronları için aktivasyon fonksiyonu, b_2 şifre çözücü yanlılık vektörüdür. Çalışmada şifre çözücü lojistik sigmoid aktivasyon fonksiyonu kullanılmıştır.

Oto kodlayıcının eğitimi, orijinal giriş x_i değerleri ile bunların yeniden oluşturulması arasındaki bir hata ölçüsünü en aza indirmeyi amaçlar. Yeniden oluşturulmuş $\hat{x}_i$ orjinal girişler x_i değerleri arasındaki hata, denklem 4.3 ile optimize edilir. Optimizasyon problemi, stokastik gradyan iniş [75,76] kullanılarak çözülebilir.

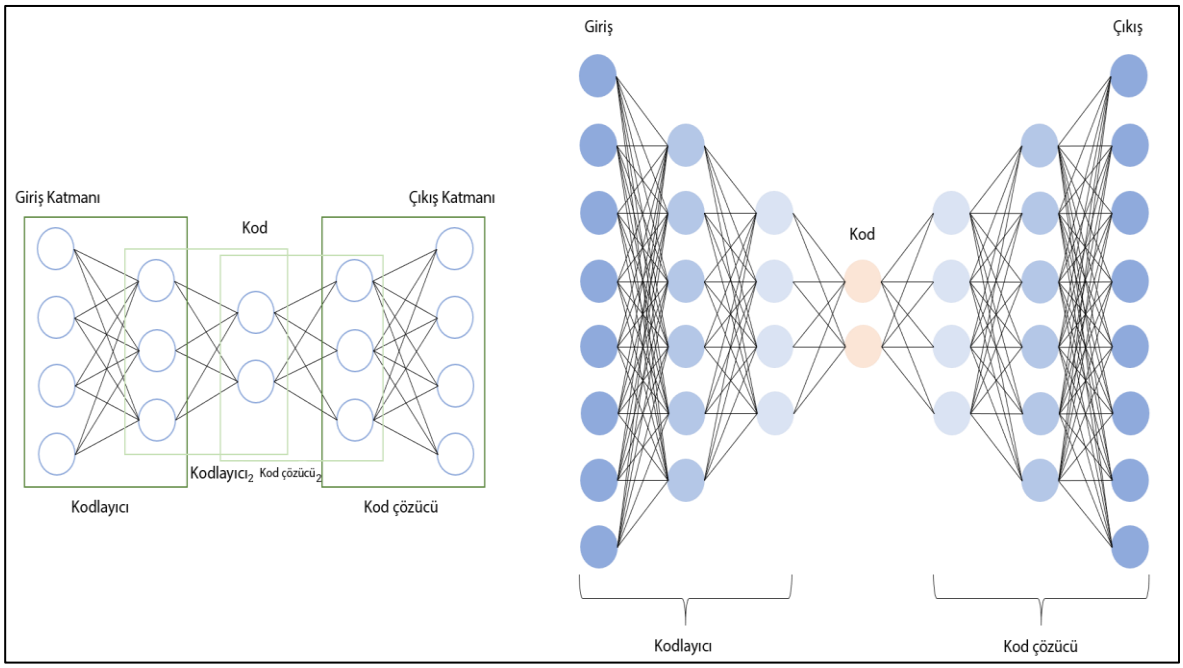
$$\hat{\theta} = arg\ min_\theta = \sum_{i=1}^{N} (\hat{x}_i - x_i)^2 \quad (4.3)$$

θ = {W1, W2, b1, b2} oto kodlayıcının parametreleridir. N, giriş örneklerinin sayısını temsil eder. Gizli katman nöronları, giriş katmanı nöronlarından daha az olduğu sürece, otomatik kodlayıcı, giriş özelliklerinin daha düşük boyutlu bir temsilini öğrenir ve bu, otomatik kodlayıcının özellik boyut küçültme için kullanılmasına izin verir.

Oto kodlayıcı, seyreklik kısıtlaması uygulanarak nöronların yalnızca bir bölümü aktif nöronlar olarak kullanılacak şekilde düzenlenebilir. Sadece bazı nöronların aktif olması için kayıp fonksiyonuna bir ceza terimi eklenir. Bu, otomatik kodlayıcıyı her girişi az sayıda nöronun bir kombinasyonu olarak temsil etmeye zorlar ve verileri temsil edebilecek önemli özellikleri keşfeder. Bu yöntem seyrek oto kodlayıcı olarak bilinir. Seyrek oto kodlayıcı, aktif nöronların sayısını kontrol etmek için gizli katmandaki nöronlara seyrek kısıtlamalar uygulayan bir oto kodlayıcı sürümüdür. Seyrek oto kodlayıcı, gizli katman boyutu giriş boyutundan daha büyük olsa bile çalışır çünkü herhangi bir zamanda gizli katmandaki nöronların yalnızca küçük bir alt kümesi aktif

olacaktır. Seyrek oto kodlayıcı, aktif nöron sayısını azaltarak sistem karmaşıklığını ve parametrelerini azaltır. Böylece daha önemli özelliklerin öğrenilmesi sağlanmaktadır [73,77].

Yığınlanmış Oto Kodlayıcı (YOK-Stacked Autoencoder-SAE) giriş katmanının boyutu yüksekse, tüm eğitim verilerini temsil etmek için yalnızca bir gizli katman kullanmak yeterli değildir. Bu zorluğun üstesinden gelmek için yığılmış oto kodlayıcı ilkesi kullanılır. Tüm eğitim verilerini daha iyi temsil etmek için oto kodlayıcıda n sayıda gizli katman kullanılabilir. Her gizli katmanın çıktısı, ardışık gizli katmanların girdisine bağlıdır. Her gizli katmanın boyutu, her bir gizli katman bir kod boyutuna ulaşana kadar simetrik olarak küçültülür. Daha sonra kod boyutundan gizli katmanlar simetrik olarak çıktı boyutuna genişletilir. Kod çözücü, kodlayıcının simetrisidir. Bu şekilde, kodlayıcı ve kod çözücü katmanları simetrik olarak istiflendiğinde, YSOK mimarisi oluşturulabilir (Şekil 4.10). Kodlayıcı ile kod çözücü arasındaki kod boyutunu temsil eden gizli katmandaki nöron sayısı azaltılmak istenen öznitelik sayısını belirlemektedir.



Şekil 4.10. Yığınlanmış seyrek oto kodlayıcı yapısı

Bir YSOK eğitimi, maliyet fonksiyonunun ayarlanmasından oluşur [76,78]:

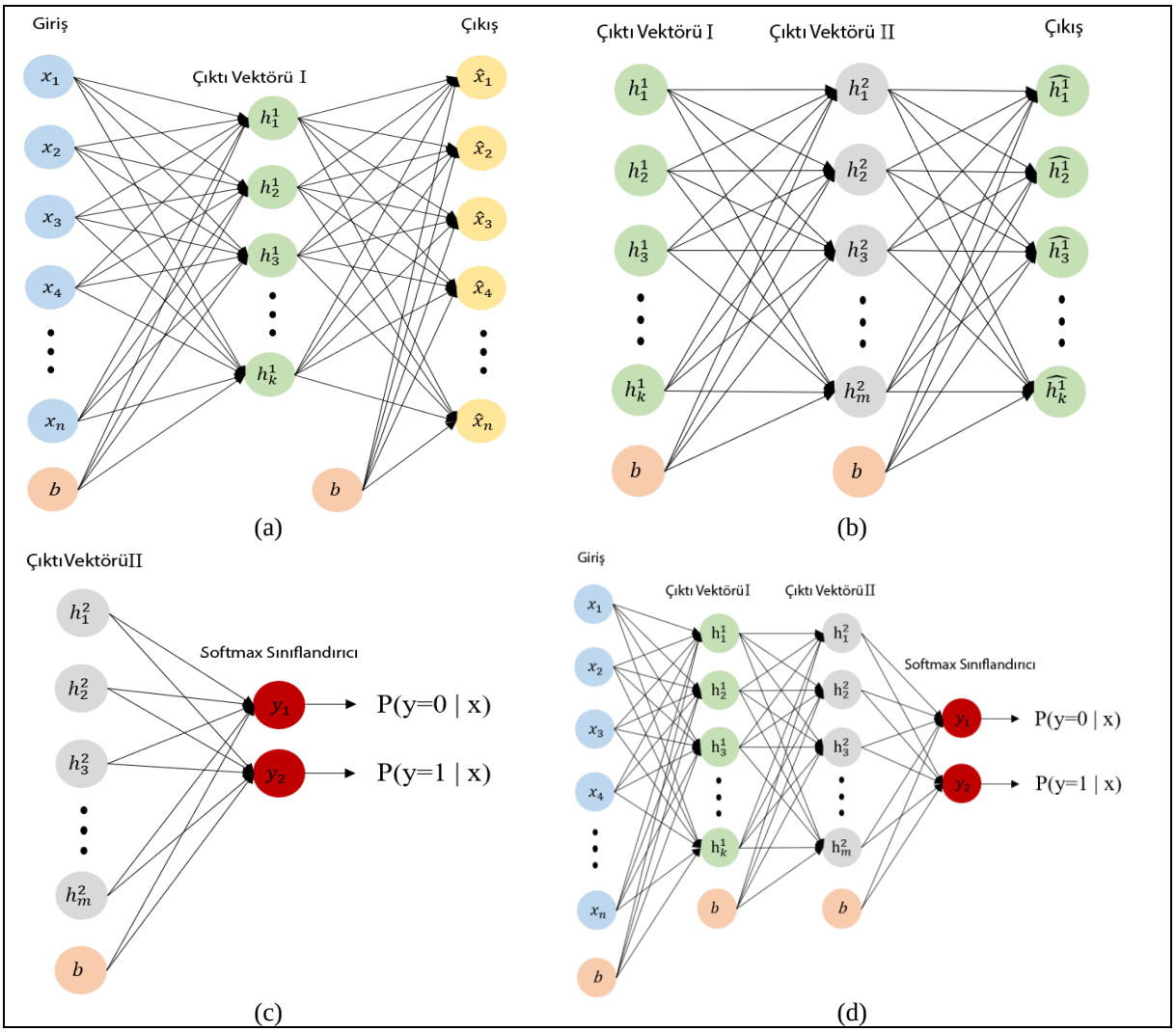
$$E = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}(x_{kn} - \hat{x}_{kn})^2 + \lambda * \Omega_{weights} + \beta * \Omega_{sparsity} \quad (4.4)$$

N ve K girdi örneklerinin sayısını ve bir örnekteki özniteliklerin sayısını temsil eder. Ayrıca $\Omega_{weights}$, L2 düzenlemesini temsil ederken, $\Omega_{sparsity}$ seyreklik düzenlemesini temsil eder. SSAE'nin eğitimi için stokastik gradyan iniş algoritması kullanılır [79,80].

YSOK+Softmax Sınıflandırıcı Derin Ağ Modeli

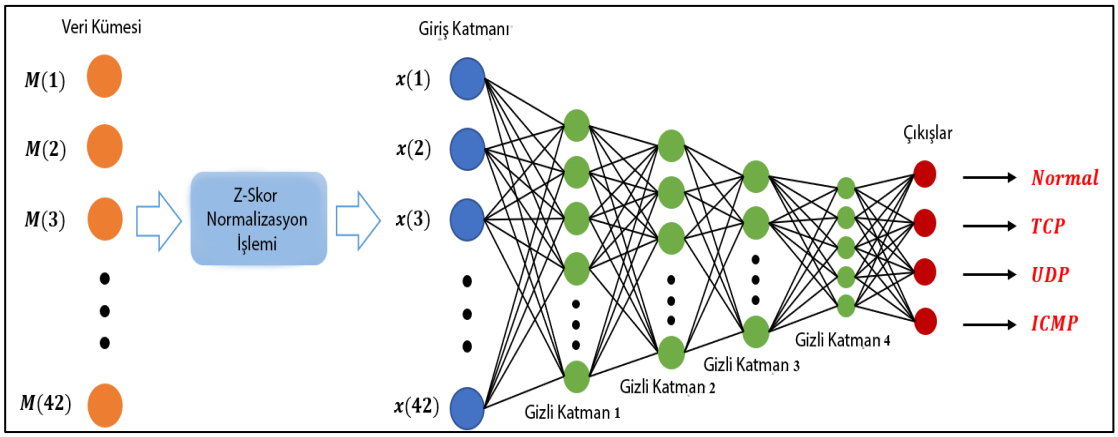
Bir YSOK+Softmax sınıflandırıcı derin ağ yapısı, YSOK yapısının kodlayıcı bölümünü ve Softmax sınıflandırıcısını içerir. Bu derin ağın eğitimi, ön öğrenme ve hassas öğrenmeden oluşur. Ön öğrenme sayesinde, tüm yapıların eğitim süreci ayrı ayrı gerçekleştirilir. Duyarlı öğrenme tüm bu yapılarla birleştirilerek eğitim süreci yeniden gerçekleştirilir [76, 81-83]. Şekil 4.11, iki oto kodlayıcı ile YOK modeline dayalı derin ağ sınıflandırıcısını göstermektedir. Bu modele atanan girdi vektörü $x_1, x_2, x_3, \dots, x_n$ olsun ve karşılık gelen çıktı sınıfı değişkenleri $y_1 \; ve \; y_2$ olsun. YSOK+Softmax sınıflandırıcı derin ağının eğitim aşamaları şu şekildedir:

1. Şekil 4.11(a)'da görüldüğü gibi, birinci oto kodlayıcı katmanı, ham (orjinal) giriş vektörleri kullanılarak eğitilir. Ayrıca, giriş vektörü ile hedef vektör aynıdır. Bu katman, özellikleri çıkararak giriş özelliklerini $h_1^1, h_2^1, h_3^1, \dots, h_k^1$ olarak yeniden yapılandırır.
2. Şekil 4.11(b)'de görüldüğü gibi, birinci oto kodlayıcı katmanının çıktı vektörü, ikinci oto kodlayıcı katmanının girişine verilir ve çıktı vektörü $h_1^2, h_2^2, h_3^2, \dots, h_m^2$ olarak üretilir. Daha sonra, ikinci oto kodlayıcı katmanı girdiyi $\widehat{h_1^1}, \widehat{h_2^1}, \widehat{h_3^1}, \dots, \widehat{h_k^1}$ olarak yeniden yapılandırır.
3. Şekil 4.11(c)'de görüldüğü gibi, ikinci oto kodlaycı katmanından elde edilen çıktı vektörü softmax katmanının girişine verilir ve eğitilir. Bu katman ağırlıkları, gerçek çıktı ile tahmin edilen çıktı arasındaki fark minumum olana kadar yenilenir.
4. Şekil 4.11(d)'de görüldüğü gibi, son olarak, tüm oto kodlayıcı katmanları ile softmax sınıflandırıcısı birleştirilerek YSOK+softmax yapısını oluşturur. Derin sinir ağının sınıflandırma performansını artırmak için, fine-tuning olarak adlandırılan backpropagation işlemi kullanılarak derin ağ eğitim verileriyle yeniden eğitilir.



Şekil 4.11. Softmax sınıflandırıcılı yığınlı seyrek oto kodlayıcı

Bu çalışmada sırasıyla tek, iki, üç ve dört katmandan oluşan dört farklı YSOK modeli geliştirilmiştir. İlk olarak 42 özellik içeren 17779 adet girişe z-skor normalizasyon işlemi uygulanmıştır. Daha sonra YSOK modellerinin girişlerine uygun düşük boyutlu özniteliklerin elde edilmesi için tüm öznitelikler gönderilmiştir. Ardından, normal trafik ve DDoS saldırı trafiği, Softmax sınıflandırıcısı kullanılarak sınıflandırıldı. Dört katmanlı YSOK modelinin genel yapısı, Şekil 4.12'de sunulmaktadır.



Şekil 4.12. Dört katmanlı YSOK yapısı

Dört katmanlı YSOK modeli dışında tek, iki ve üç katmanlı modeller geliştirilmiştir. Bu dört farklı YSOK modelinin her bir gizli katmanı için kullanılan nöron sayıları Çizelge 4.13'de verilmiştir.

Çizelge 4.13. YSOK modellerinin gizli katmanları ve her katmandaki nöron sayısı

Model	Gizli Katman	Nöron Sayısı
Tek katmanlı YSOK	Gizli Katman 1	15
İki katmanlı YSOK	Gizli Katman 1	25
	Gizli Katman 2	10
Üç katmanlı YSOK	Gizli Katman 1	40
	Gizli Katman 2	20
	Gizli Katman 3	5
Dört katmanlı YSOK	Gizli Katman 1	50
	Gizli Katman 2	25
	Gizli Katman 3	10
	Gizli Katman 4	5

Çizelge 4.13'de gösterilen dört farklı YSOK modelinin gizli katmanlarındaki nöron sayıları için [5-50] aralığındaki değerler kullanılmıştır. Değerler için kapsamlı deneysel çalışmalar yapılmış ve her bir model için en yüksek performansı veren nöron sayıları elde edilmiştir.

4.2.3. Deneysel_çalışma_2 sonuçları

Çalışmada önerilen YSOK+Softmax sınıflandırıcı derin ağ modelleri MATLAB yazılımı kullanılarak oluşturulmuştur. Deneysel çalışmalarda i7 2.8 GHz işlemci, GTX 950m 8GB GPU kart ve 16 GB RAM'e sahip bilgisayar kullanılmıştır. Ayrıca deneyler sırasında veri kümesinin %70'i eğitim süreci için, kalan %30'luk kısım ise önerilen ağ modellerinin test süreci için seçilmiştir. Deneysel sonuçlar

aşağıdaki alt bölümlerde verilmiştir. Birinci alt bölümde önerilen YSOK+Softmax sınıflandırıcı derin ağ modellerinden elde edilen deneysel sonuçlar sunulmaktadır. İkinci alt bölümde geleneksel DVM, K-EYK ve Karar Ağacı sınıflandırıcıları kullanılmış ve sonuçlar önerilen SSAE+Softmax sınıflandırıcı derin ağ modelleri ile karşılaştırılmıştır.

Çalışmada, SD-VANET'leri hedef alan DDoS saldırılarını tespit etmek için dört farklı YSOK+Softmax sınıflandırıcı derin ağ modeli önerilmiştir. Bu modeller ayrı ayrı tek, iki, üç ve dört gizli katman içerir. İlk olarak 42 özniteliğe z-skoru normalizasyon işlemi uygulanmıştır. Daha sonra bu öznitelikler YSOK+Softmax sınıflandırıcı derin ağ modellerine girdi olarak verilmiştir. Böylece SD-VANET için normal ve DDoS saldırı trafikleri belirlenmeye çalışılmıştır. Önerilen modelleri oluşturan katmanların eğitim parametreleri, değişken değerlere dayalı kapsamlı deneysel çalışmalar yapılarak belirlenmiştir. Bu parametreler, önerilen her model için en yüksek performans gösteren değerlerdir. Bu bağlamda, önerilen derin ağ modellerinin ayrıntılı konfigürasyonu Çizelge 4.14'de verilmiştir.

Çizelge 4.14. Önerilen derin ağ modellerinin eğitim parametreleri

Modeller		L2 Ağırlık Düzenlileştirme	Seyreklik Düzenlileştirme	Seyreklik Oranı	Maks. Eğitim Adımı
Tek katmanlı YSOK+Softmax	Katman 1	0,0004	4	0.6	100
	Softmax Sınıflandırıcı	-	-	-	3000
İki katmanlı YSOK+Softmax	Katman 1	0,0004	4	0.6	100
	Katman 2	0,0001	4	0.4	100
	Softmax Sınıflandırıcı	-	-	-	3000
Üç katmanlı YSOK+Softmax	Katman 1	0,0004	4	0.4	100
	Katman 2	0,0001	4	0.4	100
	Katman 3	0,0001	4	0.4	100
	Softmax Sınıflandırıcı	-	-	-	3000
Dört katmanlı YSOK+Softmax	Katman 1	0,0001	4	0.6	100
	Katman 2	0,0001	4	0.4	100
	Katman 3	0,0001	4	0.4	100
	Katman 4	0,0001	4	0.4	100
	Softmax Sınıflandırıcı	-	-	-	3000

Önerilen tek, iki, üç ve dört katmanlı YSOK+Softmax sınıflandırıcı derin ağ modellerinin doğruluk, duyarlılık, kesinlik, özgünlük ve F1 puanı gibi performans ölçütleri Çizelge 4.15'de sunulmaktadır.

Çizelge 4.15. Önerilen YSOK+Softmax sınıflandırıcı derin ağ modellerinin performans metrikleri

Model	Performans Ölçütleri (%)				
	Doğruluk	Duyarlılık	Özgünlük	Kesinlik	F1_puanı
Tek katmanlı YSOK+Softmax	93,73	92,29	98,03	92,42	92,34
İki katmanlı YSOK+Softmax	96,2	95,3	98,81	95,30	95,30
Üç katmanlı YSOK+Softmax	96,41	95,64	98,87	95,64	95,63
Dört katmanlı YSOK+Softmax	96,9	96,17	99,01	96,22	96,19

Çizelge 4.15'de görüldüğü gibi önerilen dört katmanlı YSOK+Softmax sınıflandırıcı derin ağ modeli ile en iyi doğruluk puanı %96,9 olarak elde edilmiştir. İkinci en iyi puan daha sonra üç katmanlı YSOK+Softmax sınıflandırıcı derin ağ modeli ile elde edildi. İki katmanlı YSOK+Softmax sınıflandırıcı ve tek katmanlı YSOK+Softmax sınıflandırıcı modelleri için sırasıyla %96,2 ve %93,73 doğruluk puanı elde edilmiştir. Önerilen dört derin ağ modeli için karışıklık matrisi (confusion matrix) ve ROC diyagramları sırasıyla Şekil 4.13 ve 4.14'de gösterilmektedir.

a) Tek katmanlı YSOK+Softmax için karışıklık(confusion Matrix) matrisi

b) İki katmanlı YSOK+Softmax için karışıklık(confusion Matrix) matrisi

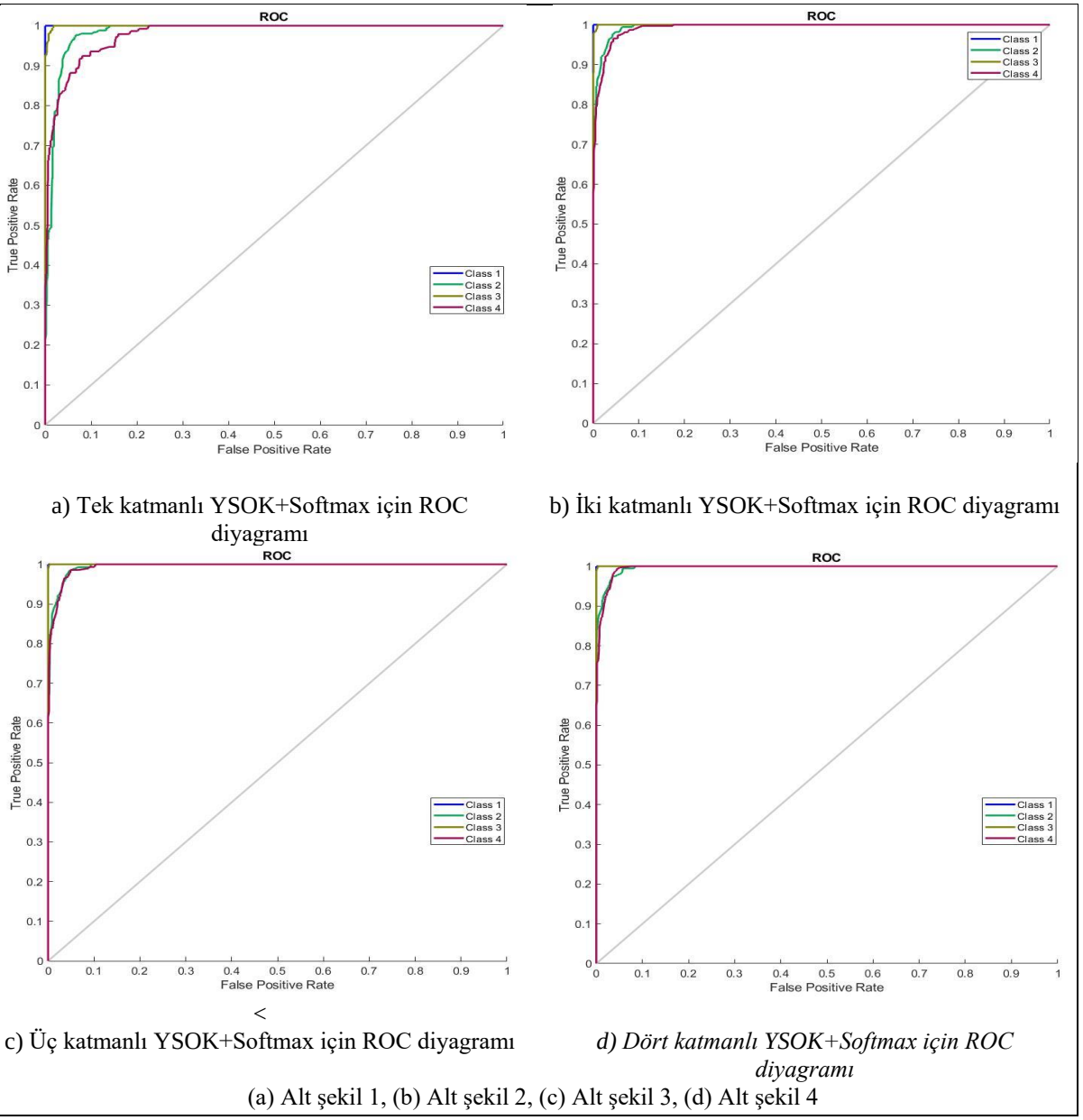
c) Üç katmanlı YSOK+Softmax için karışıklık(confusion Matrix) matrisi

d) Dört katmanlı YSOK+Softmax için karışıklık(confusion Matrix) matrisi

(a) Alt şekil 1, (b) Alt şekil 2, (c) Alt şekil 3, (d) Alt şekil 4 *(Sembol: 1) Normal, 2) TCP, 3) UDP, 4) ICMP)*

Şekil 4.13. Önerilen modellerin karışıklık matrisleri

Şekil 4.14'de önerilen derin ağ modelleri için karışıklık matrisleri verilmiştir. Bu sonuçlara göre, dört derin ağ modeli kullanılarak DDoS saldırısı olmayan durum (normal) sınıfının tespiti için doğruluk oranı %100 olarak elde edilmiştir. En yüksek doğruluk puanına sahip dört katmanlı YSOK + Softmax sınıflandırıcı derin ağ modeli kullanılarak TCP, UDP ve ICMP saldırılarının tespiti için sırasıyla %94,5, %98,5 ve %91,7 doğruluk oranları elde edilmiştir.



a) Tek katmanlı YSOK+Softmax için ROC diyagramı

b) İki katmanlı YSOK+Softmax için ROC diyagramı

c) Üç katmanlı YSOK+Softmax için ROC diyagramı

d) Dört katmanlı YSOK+Softmax için ROC diyagramı

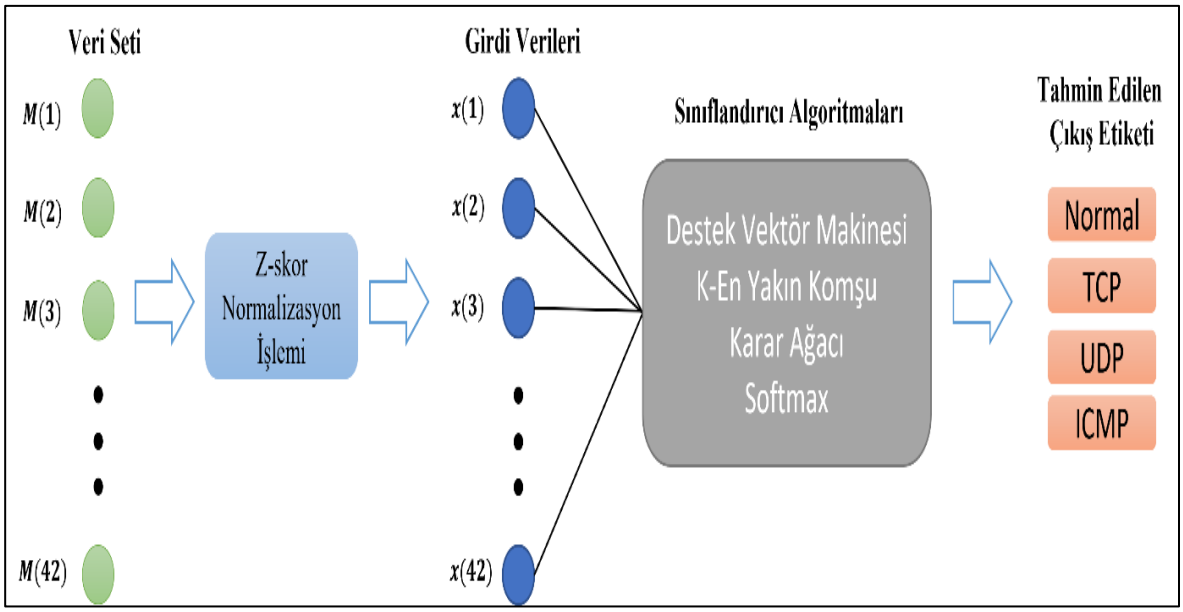
(a) Alt şekil 1, (b) Alt şekil 2, (c) Alt şekil 3, (d) Alt şekil 4

Şekil 4.14. Önerilen derin ağ modellerinin ROC diyagramları

Klasik makine öğrenme yöntemlerin önerilen modellerle karşılaştırılması

Ayrıca, bu çalışmada önerilen YSOK+Softmax sınıflandırıcı derin ağ modellerinin sınıflandırma performansları, çeşitli klasik makine öğrenimi sınıflandırıcılarının performansları ile karşılaştırıldı (Şekil 4.15). Bu amaçla veri kümesi herhangi bir öznitelik boyut küçültmesi yapılmadan DVM, K-EYK ve Karar Ağacı gibi klasik makine öğrenimi sınıflandırıcılarının girdilerine verildi. Ayrıca, oto kodlayıcının derin ağ modelinin performansı üzerindeki etkisini belirlemek için softmax sınıflandırıcının performansı test edilmiştir. DVM, K-EYK ve Karar Ağacı sınıflandırma algoritmalarının kullanımındaki parametreler şunlardır:

- DVM algoritması: bire bir yaklaşım ve Fine Gaussian DVM sınıflandırıcı türü kullanıldı.
- K-EYK algoritması: komşuluk değeri 10 ve Öklid mesafe fonksiyonu kullanılmıştır.
- Karar Ağacı algoritması: maksimum karar bölme sayısı 20 ve Medium Tree, sınıflandırıcı türü olarak tanımlandı.



Şekil 4.15. Oto kodlayıcı kullanmadan makine öğrenimi algoritmalarıyla sınıflandırma

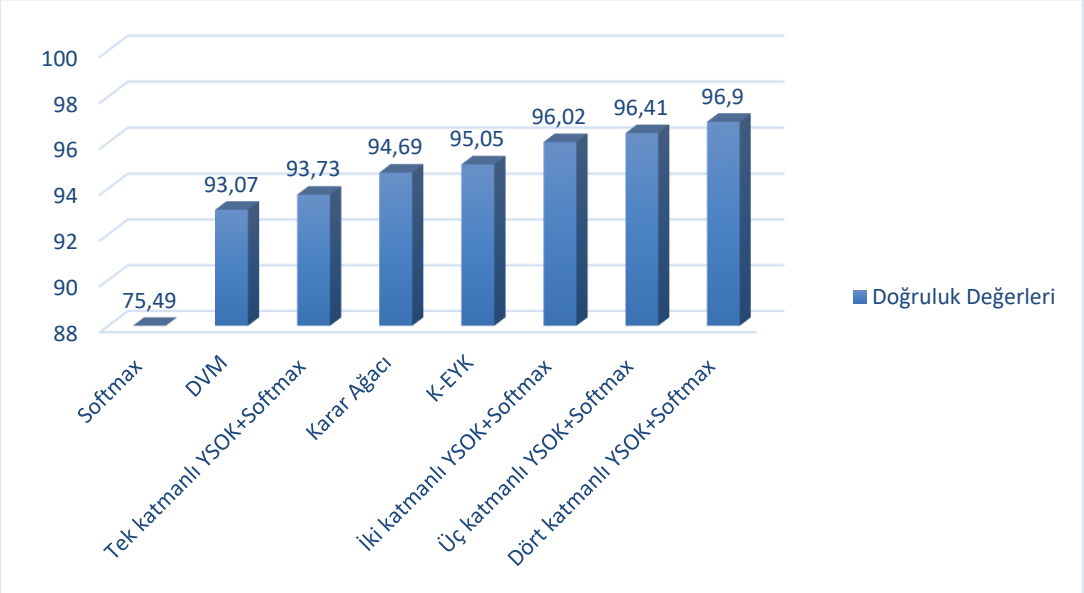
SD-VANET'leri hedef alan DDoS saldırılarını tespit etmek amaçlı oluşturulan veri kümesinin DVM, K-EYK, Karar Ağacı ve Softmax sınıflandırıcılarına uygulanmasıyla elde edilen performans metrikleri Çizelge 4.16'da sunulmaktadır.

Çizelge 4.16. DVM, K-EYK, Karar Ağacı ve Softmax sınıflandırıcılarının performans metrikleri

Sınıflandırıcılar	Performans Metrikleri (%)				
	Doğruluk	Duyarlılık	Özgünlük	Kesinlik	F1-skoru
DVM	93,07	91,63	97,82	91,52	91,56
K-EYK	95,06	94,21	98,44	94,18	94,19
Karar Ağacı	94,69	93,66	98,33	94,48	93,54
Softmax	75,49	69,56	92,32	68,88	69,18

Bu sınıflandırıcılar arasında en yüksek doğruluk %95,06 ile K-EYK algoritmasında elde edilirken, DVM ve Karar Ağacı yöntemlerinde sırasıyla %93,07 ve %94,69 doğruluk elde

edilmiştir. Ayrıca en düşük doğruluk %75,49 ile Softmax sınıflandırıcı ile elde edilmiştir (Şekil 4.16).



Şekil 4.16. YSOK+Softmax sınıflandırıcı derin ağ modelleri ile diğer klasik makine öğrenimi modellerinin karşılaştırılması

Şekil 4.16'da, önerilen YSOK+Softmax sınıflandırıcı derin ağ modellerinin ve diğer klasik makine öğrenimi sınıflandırıcılarının doğrulukları karşılaştırılmıştır. Bu sonuçlara göre en iyi doğruluk puanı dört katmanlı YSOK+Softmax sınıflandırıcı derin ağ modeli ile elde edilirken, ikinci en yüksek doğruluk puanı üç katmanlı YSOK+Softmax sınıflandırıcı derin ağ modeli ile elde edilmektedir. K-EYK ve Karar Ağacı sınıflandırıcılarının, tek katmanlı YSOK+Softmax sınıflandırıcı derin ağ modeline göre daha başarılı olduğu görülmektedir. Son olarak oto kodlayıcı+Softmax (dört katmanlı YSOK+Softmax) modeli Softmax sınıflandırıcı ile karşılaştırıldığında yaklaşık %22 gibi önemli bir performans artışı elde edilmiştir.

4.2.4. Deneysel_çalışma_2 sonuçlarının literatürdeki benzer çalışmalarla karşılaştırılması

Deneysel çalışma sonunda elde ettiğimiz sonuçlar üçüncü bölümde özetlenen benzer literatür çalışmalarıyla karşılaştırılmıştır. Önerilen YSOK+Softmax sınıflandırıcı derin ağ modelinin performans sonuçları üçüncü bölümde özetlenen literatür çalışmaları ile karşılaştırılarak Çizelge 4.17'de verilmiştir.

Çizelge 4.17. Literatürdeki benzer çalışmaların karşılaştırılması

Kullanılan Veri kümesi	Makine öğrenimi Algoritması		Doğruluk (%)	Ref.
DARPA (1999-2000) CAIDA DDoS (2007)	DVM		Ortalama 95,4	[38]
ISCX, ISCX2012	TSA UKSB ESA		98	[39]
Kendi veri kümeleri	DVM		Ortalama 96,25	[40]
CAIDA DDoS 2007	DVM		92,9	[41]
Kendi veri kümeleri	YOK		95,65	[42]
NSL-KDD	DSA		75,75	[43]
CICIDS2017	ESA		99,45	[44]
NSL-KDD	DNN		90	[45]
NSL-KDD	Rastgele Orman GTB-UKSB		82 88	[46]
NSL-KDD	CAE		-	[47]
Kendi veri kümeleri	YOK		95	[48]
Kendi veri kümeleri	LDDM		96,28	[49]
Kendi veri kümemiz	DVM K-EYK Karar Ağacı Softmax		93,07 95,06 94,69 75,49	
	Önerilen YSOK+Softmax sınıflandırıcılı Derin Ağ Modeli	Tek Katmanlı İki Katmanlı Üç Katmanlı Dört Katmanlı	93,73 96,2 96,41 96,9	

Çizelge 4.17, önerilen YSOK+Softmax sınıflandırıcı derin ağ modelinin önceki çalışmalara göre başarılı bir performans verdiğini göstermektedir. Ayrıca önceki çalışmaların büyük çoğunluğunda geleneksel makine öğrenimi sınıflandırıcı yöntemleri kullanılmıştır. Literatürde yer alan bu çalışmalarda farklı veri kümelerinin kullanıldığı unutulmamalıdır. Bu nedenle, karşılaştırmanın sonuçlarını doğrulamak zordur.

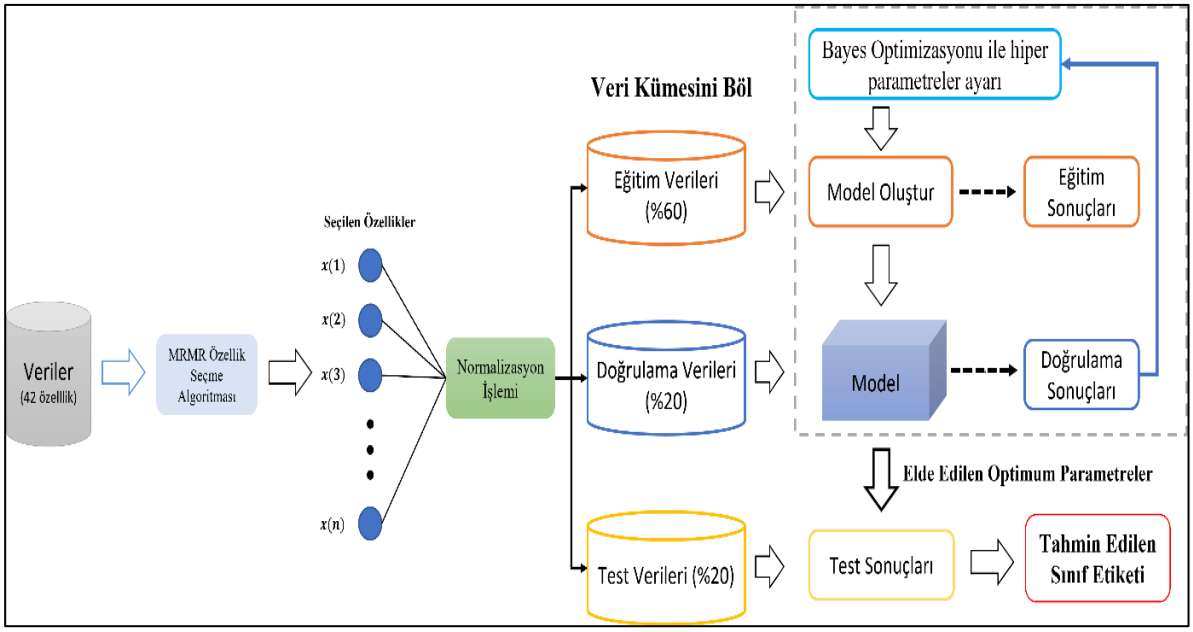
4.3. Deneysel_Çalışma_3: Hiperparametre Optimizasyonu ve Özellik Seçimi Kombinasyonu Tabanlı Yöntemle DDoS Saldırısı Tespiti

Deneysel_çalışma_3'de, SD-VANET'leri hedefleyen DDoS saldırılarının tespiti için veri kümesinden öznitelik seçimi ile desteklenen makine öğrenimi sınıflandırıcıları ve sınıflandırıcı modellerinin optimum hiperparametreleri kullanılmıştır. Veri kümesinin en verimli ve ayırt edici öznitelikleri En Küçük Artıklık En Büyük İlgililik (Minimum Redundancy Maximum Relevance-MRMR) yöntemi kullanılarak seçilmiştir. Makine öğrenmeli sınıflandırıcıların hiperparametre optimizasyonu işlemi için Bayes Optimizasyon (BO) yöntemi kullanılmıştır.

Çalışmanın süreç adımları, veri kümesi elde etme, öznitelik seçimi, veri normalleştirme, veri kümesinin bölünmesi, eğitim/hiperparametre optimizasyonu ve test etme aşamalarını içermektedir. Bu aşamalar şu şekilde açıklanmaktadır:

1. İlk adım ham veri toplamaktır. Veri kümesi, hem DDoS saldırıları hem de düzenli ağ trafiği verilerinden oluşan özel tasarlanmış SD-VANET topolojisinden elde edilmiştir.
2. MRMR algoritması kullanılarak ham veri kümesinden verimli ve ayırt edici özellikler seçildi.
3. Veri kümesine Max-Min normalizasyonu uygulandı.
4. Makine öğrenimi sınıflandırıcı mimarileri ve hiperparametreleri belirlendi.
5. Veri kümesi "eğitim", "doğrulama" ve "test" setleri (üç yönlü veri bölmeleri) olarak üç alt kümeye ayrılmıştır.
6. Eğitim seti kullanılarak sınıflandırıcı modeller oluşturulmuştur (eğitim seti, veri kümesinin %60'ıdır).
7. Doğrulama seti kullanılarak, sınıflandırıcı modellerin hiperparametreleri ayarlandı. (doğrulama seti, veri kümesinin %20'sidir). Hiperparametre optimizasyonu, Bayes optimizasyonuna dayalı olarak gerçekleştirildi.
8. Sınıflandırıcı modellerin performansı, test seti kullanılarak değerlendirildi (test seti, veri kümesinin %20'sidir).

Çalışmanın süreç adımlarının genel işleyişi Şekil 4.17'de sunulmaktadır.



Şekil 4.17. Deneysel_çalışma_3 genel akış diyagramı

Bu çalışma, en iyi veri kümesi özelliklerini elde ederek SD-VANET'leri hedef alan DDoS saldırılarını tespit etmek için en iyi sınıflandırıcı modeli belirlemeyi ve MRMR algoritması ve Bayes Optimizasyonu kullanarak model hiperparametre optimizasyonunu amaçlamaktadır. DDoS saldırıları ve izin verilen ağ trafiği verilerini içeren veri kümesinin elde edilmesi, veri kümesinden en ayırt edici özelliklerin seçilmesi ve makine öğrenimi modelleri için hiperparametre optimizasyonu gibi bu çalışmada kullanılan materyal ve yöntemlere ilişkin detaylar aşağıdaki alt bölümlerde açıklanmıştır [84].

4.3.1. Deneysel_çalışma_3 için oluşturulan veri kümesi

Deneysel_çalışma_2'de kullanılmak üzere SD-VANET mimarisi oluşturularak elde edilen veri kümesi deneysel_çalışma_3'de de kullanılmıştır.

Öznitelik Seçim Yöntemleri ve Makine Öğrenme Modeli

Bazı durumlarda, makine öğrenimi sınıflandırıcıları tarafından kullanılan veri kümesi birçok özellik içerebilir. Bu özelliklerin sınıflandırma performansını iyileştirmede rastgele yüksek etkileri olabilirken, bazı özelliklerin sınıflandırma performansı üzerinde hiçbir etkisi olmayabilir. Sınıflandırma performansı üzerinde daha zayıf bir etkiye sahip olan özelliklerin kullanılması süreç ve zaman maliyetlerini artırabilir. Öznitelik seçiminin amacı, sınıflandırma performansı üzerinde etkisi daha zayıf olan öznitelikleri azaltmak ve sınıflandırma performansına güçlü katkısı olan özniteliklerin daha fazla kullanılmasını sağlamaktır.

En Küçük Artıklık En Büyük İlgililik (Minimum Redundancy Maximum Relevance)

MRMR öznitelik seçim algoritması kullanılarak veri kümesinin en verimli ve ayırt edici özellikleri seçilmiştir. MRMR öznitelik seçme yönteminin tercih edilmesinin nedeni, entropiye dayalı çalıştığı için yüksek performanslı ve hızlı bir algoritma olmasıdır. MRMR algoritması, sınıf etiketlerine sahip en ilgili öznitelikleri seçerken, eş zamanlı olarak seçilen öznitelikler arasındaki fazlalığı en aza indirmeyi amaçlayan bir filtreleme yöntemidir.

En başarılı makine öğrenimi filtreleme tekniklerinden biri olan MRMR yöntemi, Peng ve arkadaşları tarafından geliştirilmiştir [85]. MRMR yöntemi, değişkenlerin bir kombinasyonunun mutlaka iyi bir sınıflandırma/tahmin performansı sağlamayacağının kabulüne dayanmaktadır [86]. Her adımda, en üst sıradaki değişkenlerin hedef değişken

üzerindeki ortak bağımlılığını en üst düzeye çıkarmak için bir girişimde bulunulur. Eşzamanlı olarak seçilen değişkenler arasındaki fazlalığı en aza indirmek için ayarlanan değişkenleri bir sonrakine genişletir [87].

Belirli bir sınıflandırma görevi için bir dizi özniteliğin önemini sıralamak için MRMR kullanılır. Böylece, öznitelikler hedefle olan ilişkilerine göre sıralanır. Buradaki temel amaç, karşılıklı bilgileri kullanarak bir dizi özniteliği ifade eden S ve c sınıfı arasındaki maksimum bağımlılığı bulmaktır [88]. Bir öznitelik çifti arasındaki karşılıklı bilgi, denklem 4.5 ile tanımlanır. X ve Y değişkenlerinin karşılıklı bilgisi, I ile temsil edilir. I, ortak olasılık dağılımları p(x, y) ve ilişkili marjinal olasılıklar p(x) ve p(y). X ve Y rastgele değişkenlerinin marjinal olasılık dağılım fonksiyonları, sırasıyla p(x) ve p(y) ile temsil edilir.

$$I(X,Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y) log\left(\frac{p(x,y)}{\mathrm{p}(x)\mathrm{p}(y)}\right) \quad (4.5)$$

Daha yüksek boyutlu uzaylarda çözmek için maksimum bağımlılık kriterini uygulamak kolay bir iş değildir. Yani, örnek sayısı çoğu zaman yetersizdir ve genellikle çok değişkenli yoğunluğu tahmin etmek için hesaplama maliyetini artırır. Maksimum bağımlılık kriterinin uygulanması zor olduğundan alternatif, kriter olarak maksimum alaka düzeyini kullanmaktır. Maksimum alaka düzeyi koşulu, en yüksek hedef değişkenin seçilen bireysel değişkenlere bağımlılığı, denklem 4.6'yı kullanarak tatmin edici değişkenler arar [87].

$$\mathrm{MAX}\, D(\mathrm{S}, c);\ D = \frac{1}{|\mathrm{S}|}\sum_{\mathrm{S_i}\in \mathrm{S}} I(S_i; c) \quad (4.6)$$

Özniteliklerin maksimum alaka düzeyi kriterlerine göre seçilmesi, büyük miktarda veri fazlalığını beraberinde getirebilir. Ayrıca, öznitelikler arasındaki bağımlılığın büyük olmasına neden olabilir. İki öznitelik birbirine oldukça bağımlı olduğunda açıklayıcı değişkenler gereksiz olabilir. Gereksiz değişkenlerden herhangi birinin atlanması durumunda, tahmin performansı düşmeyecektir. İkame değişkenler, denklem 4.7'de gösterildiği gibi minimum fazlalık kriteri eklenerek azaltılabilir:

$$\mathrm{MİN}\, R(\mathrm{S}); R = \frac{1}{|\mathrm{S}|}\sum_{\mathrm{s_i},\mathrm{s_j}\in \mathrm{S}} I(S_i, S_j) \quad (4.7)$$

Yukarıdaki koşulların her ikisini de entegre eden kriter MRMR'dir. Denklem 4.8, alaka düzeyini D ve fazlalık R'yi aynı anda optimize etmek için kullanılır.

$$\text{MAX}\Phi(\text{D},\text{R}),\ \Phi = \text{D} - \text{R},\ \text{respectively}\ \Phi = \frac{D}{R} \tag{4.8}$$

Makine Öğrenimi Sınıflandırıcıları

Deneysel_Çalışma_3'te literatürde yaygın olarak kullanılan üç farklı makine öğrenmesi algoritması kullanılmıştır: K-EYK, DVM ve Karar Ağacı. Veri kümelerinin sınıflandırılması için evrensel bir en iyi sınıflandırma algoritması kesin olarak önerilemez. Sınıflandırıcılar, farklı veri kümelerinde farklı performanslar sergileyebilir. K-EYK, DVM ve Karar Ağacı düşük hiperparametreli, kararlı ve düşük performanslı algoritmalardır. Çalışmada bu algoritmaların SD-VANET'leri hedef alan DDoS saldırılarını tespit etme performansları incelenmiştir.

Karar Ağacı, sınıflandırma problemlerinde kullanılan denetimli öğrenme algoritmasıdır. Karar Ağacı öğrenimi sırasında, öğrenilen bilgi bir ağaç üzerinde modellenir. Karar Ağacında her bir düğüm bir özelliği, ağacın yaprak düğümleri seviyesindeki sınıf etiketleri ve bu yapraklara giden ve baştan ayrılan dallar, özellikler üzerindeki işlemleri ifade eder. Karar Ağacı öğrenmede, veri kümesi bir dizi karar/kural uygulanarak daha küçük kümelere bölünür. Bu işlem yinelemeli olarak tekrarlanır ve tekrarlama işlemi sınıflandırmayı etkilemeyene kadar devam eder. Karar Ağacı sınıflandırma performansı, maksimum bölme sayısı ve bölme kriteri gibi hiperparametrelerden etkilenir. Karar Ağacı sınıflandırıcısının hiperparametrelerinin arama aralıkları Çizelge 4.18'de listelenmiştir [89].

Çizelge 4.18. Karar Ağacı sınıflandırıcısı için hiperparametreler ve arama aralıkları

Hiperparametreler	Arama Aralıkları
Maksimum Bölme Sayısı	1-17758
Bölme Kriteri	Gini'nin çeşitlilik indeksi, Twoing kuralı, Maksimum sapma azaltma

Bayes Hiperparametre Optimizasyonu

Hiperparametreler, makine öğrenme tekniğinde model mimarisini tanımlayan özelliklerdir ve hiperparametrelerin en iyi kombinasyonunu bulma işlemine hiperparametre optimizasyonu (ayarlama) denir. Hiperparametre optimizasyonu, bir makine öğrenimi modelinin

performansını büyük ölçüde etkileyebilir. Bir sınıflandırıcı modelinin hiperparametreleri, rastgele deneyler kullanılarak dikkatlice ayarlanmalıdır. Bu deneyler uzun zaman alabilir, bu nedenle bir dizi hiperparametre mümkün olduğunca az deneyle optimize edilmelidir. Bayes Optimizasyonu, Grid Search ve Random Search gibi hiperparametre optimizasyon yaklaşımlarına kıyasla daha az deneme ile hiperparametrelerin optimize edilmesine olanak tanır [90].

Bayes Optimizasyonu, sıralı model tabanlı optimizasyon (Sequential Model Based Optimization) olarak da adlandırılır. Bu optimizasyon, olasılığa dayalı bir yaklaşımdır ve iki ana işlem aşamasından oluşur: amaç işlevini modellemek için vekil işlev ve bir sonraki veri noktasını örneklemek için edinme işlevi. Gauss süreci olan vekil işlev, x aday noktasında f(x) amaç fonksiyonu için potansiyel değerleri tanımlayan bir bayes arka olasılık dağılımı sağlar. Edinme fonksiyonu, amaç fonksiyonu f üzerindeki mevcut sonsal dağılıma dayalı olarak amaç fonksiyonunu yeni bir x noktasında değerlendirerek üretilecek değeri hesaplar [91].

Edinme fonksiyonu, sadece amaç fonksiyonunun sorgulanacağı noktaları belirlediği için Bayes Optimizasyonu algoritmasının merkezinde yer alır. Bayes Optimizasyonu, edinme işlevini en üst düzeye çıkararak değerlendirilecek bir sonraki veri noktasını yinelemeli olarak örnekler [92].

4.2.3. Deneysel_çalışma_3 sonuçları

Deneysel_çalışma_3'te öncelikle veri kümesi, MRMR öznitelik seçim yöntemi kullanılmadan K-EYK, DVM ve Karar Ağacı makine öğrenmesi algoritmaları kullanılarak sınıflandırılmıştır. Deneyler sırasında veri kümesinin %60'ı eğitim için, %20'si doğrulama için ve geri kalan %20'si sınıflandırıcı modellerin test edilmesi için kullanılmıştır. Bu aşamada, sınıflandırıcılardan en yüksek performansı alabilmek için Bayes Optimizasyonu yardımıyla sınıflandırıcı model hiperparametre optimizasyon işlemi gerçekleştirilmiştir. Öznitelik seçimi kullanılmadan elde edilen sonuçlar Çizelge 4.19'da gösterilmektedir**.**

Çizelge 4.19**.** MRMR ile öznitelik seçimi yapılmadan makine öğrenimi yöntemlerinin performans sonuçları

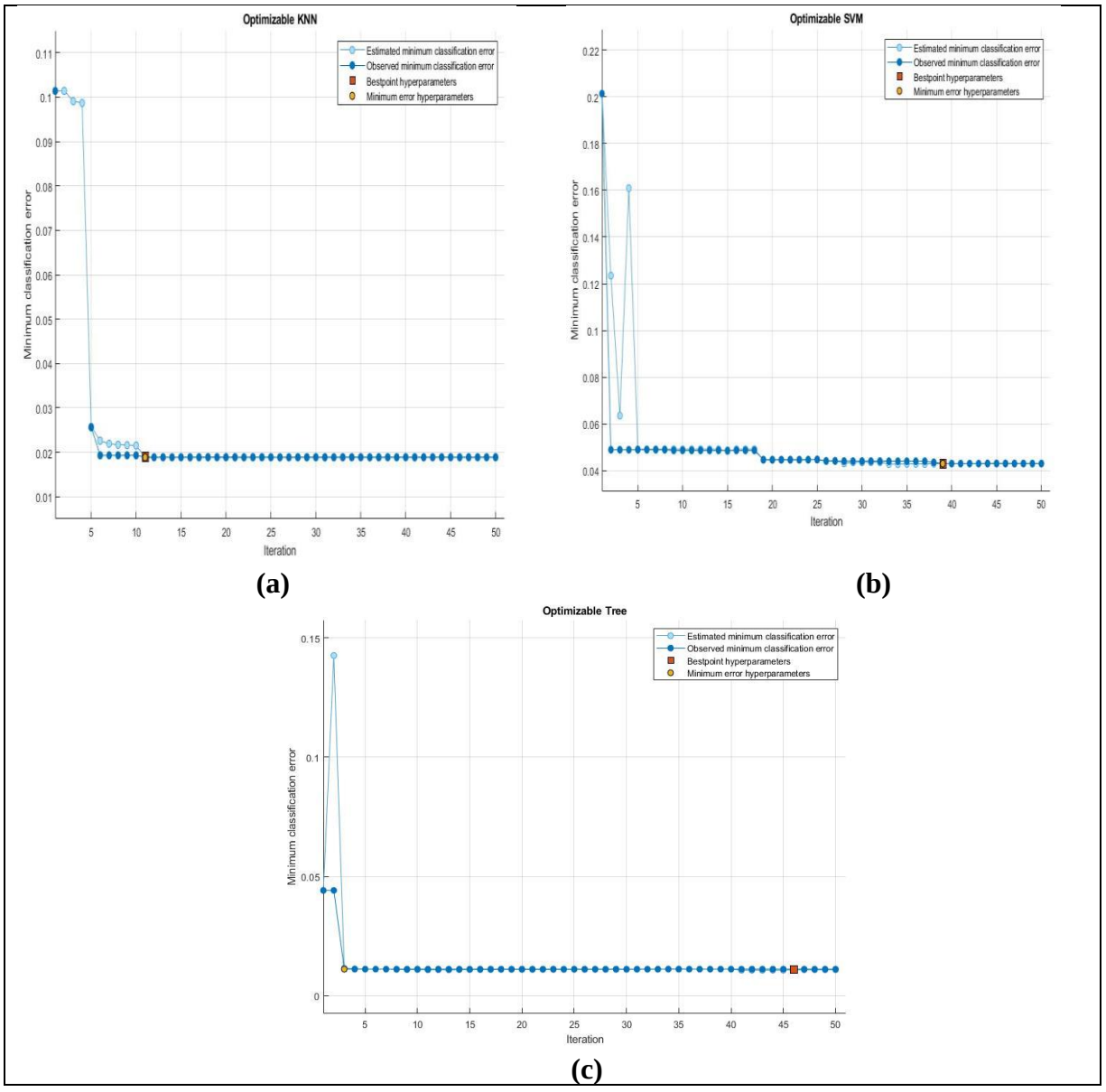
	Doğruluk(%)	Duyarlılık(%)	Özgünlük(%)	F1-Skoru(%)
K-EYK	98,06	97,66	99,39	97,64
DVM	95,27	94,29	98,40	94,59
Karar Ağacı	98,25	97,89	99,45	97,87

Çizelge 4.19, üç farklı sınıflandırıcının doğruluk, özgünlük, duyarlılık ve F1-Skor değerlerini vermektedir. Bu sonuçlarla uyumlu olarak, sınıflandırıcılar arasında en yüksek doğruluk puanı %98,25 ile Karar Ağacı sınıflandırıcısına aittir. Ayrıca, her bir sınıflandırıcı için elde edilen en yüksek performanslı hiperparametreler aşağıdaki gibidir:

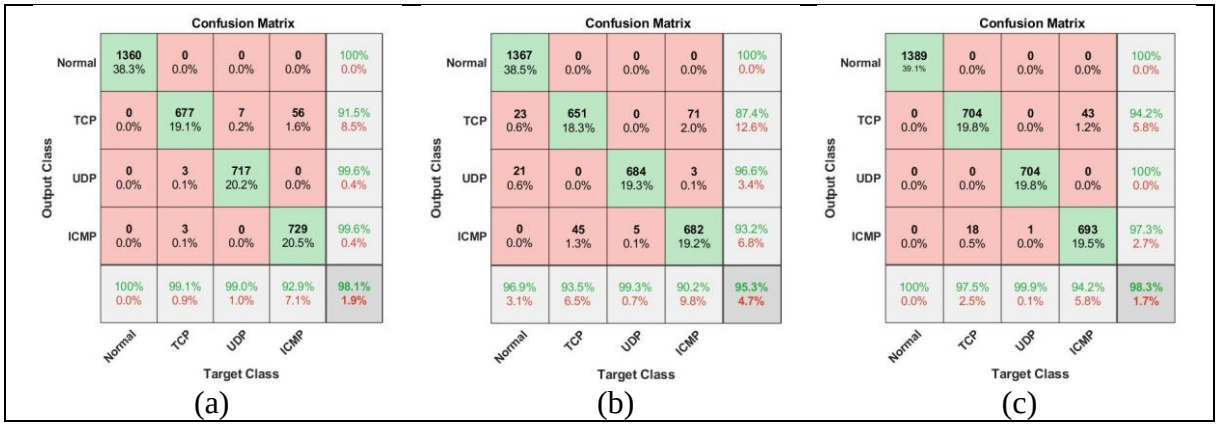
- K-EYK sınıflandırıcısı için Bayes Optimizasyonu yöntemi kullanılarak komşu sayısı (42), uzaklık metriği (öklid) ve mesafe ağırlığı (kare ters) hiperparametrelerinde en yüksek gerçekleşme sağlanmıştır.
- DVM sınıflandırıcısı için Bayes Optimizasyonu yöntemi kullanılarak, çekirdek fonksiyonu (Gaussian), çekirdek ölçeği (0,019661), kutu kısıtlama seviyesi (545.663) ve çoklu sınıf yöntemi (bire karşı bir) hiperparametrelerinde en yüksek gerçekleşme elde edilmiştir.
- Karar Ağacı sınıflandırıcısı üzerinde Bayes Optimizasyonu yöntemi uygulanarak, maksimum bölme sayısı (927) ve bölme kriteri (maksimum sapma azaltma) hiperparametreleri ile en yüksek etki elde edilmiştir.

Üç farklı sınıflandırıcıya dayalı optimizasyon sürecinin minimum sınıflandırma hatası grafiği Şekil 4.18'de gösterilmektedir.

Şekil 4.18'de görüldüğü gibi üç farklı sınıflandırıcı üzerinde Bayes Optimizasyonu işlemi için 50 iterasyon uygulanmıştır. En iyi nokta hiperparametreleri, Karar Ağacı sınıflandırıcısı ile 46. iterasyonun sonunda elde edilmiştir. Ayrıca K-EYK ve DVM sınıflandırıcılarının en iyi hiperparametreleri sırasıyla 11. ve 39. iterasyon sonunda elde edilmiştir. Üç Bayes Optimizasyonu tabanlı sınıflandırıcı için karışıklık matrisleri, Şekil 4.19'de verilmiştir.



Şekil 4.18. Bayes Optimizasyonu minimum sınıflandırma hatası grafiği, a) K-EYK (K-Nearest Neighbour-KNN) b) DVM (Support Vector Machine-SVM), c) Karar Ağacı (Decision Tree)



Şekil 4.19. a) K-EYK, b) DVM, c) Karar Ağacı modelleri için karışıklık matrisleri

Şekil 4.19'da üç farklı sınıflandırıcı kullanılarak elde edilen performansların karışıklık matrisleri sunulmaktadır. Bu sonuçlara göre bütün sınıflandırıcılar DDoS saldırısının olmadığı durumun tespitinde %100 başarı oranı elde etmiştir.

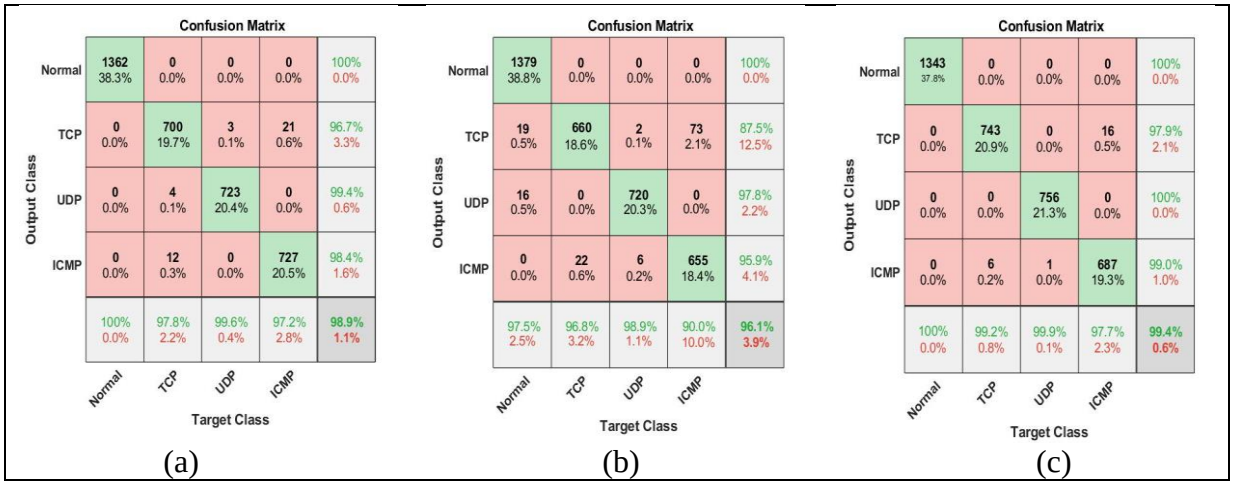
MRMR Öznitelik Seçim Algoritmasının Etkileri

Deneysel_çalışma_3'te ikinci olarak, veri kümesindeki her bir saldırı tipine ait 42 özelliğe MRMR öznitelik seçim yöntemi uygulanmıştır. Bu işlemin amacı, veri kümesinden daha sağlam ve verimli öznitelikler seçmektir. Yeni seçilen indirgenmiş özniteliklere sahip veri kümesi, bayes hiperparametre optimizasyonuna dayalı K-EYK, DVM ve Karar Ağacı sınıflandırma algoritmaları kullanılarak sınıflandırıldı. Sınıflandırıcı doğruluk değerleri Çizelge 3.20'de verilmiştir.

Çizelge 4.20. MRMR öznitelik seçimi ile makine öğrenimi yöntemlerinin performans sonuçları

		5 seçilmiş öznitelik	10 seçilmiş öznitelik	15 seçilmiş öznitelik	20 seçilmiş öznitelik	25 seçilmiş öznitelik	30 seçilmiş öznitelik
K-EYK	Doğruluk	88,71	94,12	98,17	98,62	98,87	98,42
	Duyarlılık	86,47	93,06	97,76	98,37	98,63	98,16
	Özgünlük	96,44	98,16	99,42	99,56	99,65	99,50
	F1-Skoru	86,55	92,86	97,77	98,35	98,63	98,14
DVM	Doğruluk	86,15	86,94	95,72	96,11	95,97	95,78
	Duyarlılık	83,43	84,09	94,76	95,31	95,19	95,02
	Özgünlük	95,61	95,90	98,57	98,69	98,65	98,59
	F1-Skoru	83,56	84,09	94,96	95,47	95,38	95,16
Karar Ağacı	Doğruluk	88,32	96,23	98,79	99,13	99,35	98,93
	Duyarlılık	86,14	95,45	98,53	98,92	99,22	98,75
	Özgünlük	96,29	98,82	99,62	99,73	99,80	99,67
	F1-Skoru	86,31	95,36	98,52	98,91	99,21	98,70

Çizelge 4.20'de hem MRMR özellik seçme algoritmasını hem de bayes hiperparametre optimizasyonunu kullanan makine öğrenimi sınıflandırıcı modellerinin performans sonuçlarını göstermektedir. Bu sonuçlardan da anlaşılacağı üzere en iyi doğruluk değerine K-EYK ve Karar Ağacı sınıflandırıcıları için seçilen 25 öznitelik ve DVM sınıflandırıcısı için seçilen 20 öznitelik ile ulaşılmıştır. Sonuçlar, veri kümesinden öznitelik seçimi ve sınıflandırıcı modellerin hiperparametre optimizasyonu kullanılarak SD-VANET'te DDoS saldırılarının tespitinde daha iyi sınıflandırma performansı elde edilebileceğini göstermektedir. MRMR öznitelik seçim yöntemine sahip üç Bayes Optimizasyonu tabanlı sınıflandırıcı için en iyi doğruluk değerlerinin karışıklık matrisleri Şekil 4.20'de gösterilmektedir.



Şekil 4.20. a) K-EYK, b) DVM, c) Karar Ağacı modelleri için karışıklık matrisleri

Deneysel_Çalışma_3 Kapsamına Önerilen Yaklaşımın Zaman Analizi

Gerçek zamanlı uygulamalarda yürütme süresi ve sınıflandırma performansı önemli bir rol oynar. Bu nedenle öznitelik seçimi, veri normalleştirme ve test etme aşamalarının zaman analizleri yapılmıştır. Yürütme süresi Çizelge 4.21'de verilmiştir.

Çizelge 4.21. Önerilen sistemde yürütme süresi (dk/sn)

	İşlem Adımları	K-EYK	DVM	Karar Ağacı
Önerilen yaklaşım	Öznitelik seçimi (20 öznitelik)	0,0117		
	Normalizasyon(benzer)	0,0002		
	Test	0,0816	0,0865	0,0008
	Toplam	0,0935	0,0984	0,0127
42 öznitelik (ham veri)		0,1860	0,1537	0,0017

Önerilen sistemin her işlem adımı için yürütme süreleri Çizelge 4.21'de gösterilmiştir. Bu sonuçlara göre, Karar Ağacı sınıflandırıcısı (0,0127 sn), üç model arasında en düşük toplam yürütme süresini vermiştir. K-EYK ve DVM sınıflandırıcılarının toplam yürütme süreleri sırasıyla 0,0935 sn. ve 0,0984 sn'dir. Ayrıca Çizelge 4.21'deki verilere göre K-EYK ve DVM sınıflandırıcıları için önerilen yaklaşımın uygulama süresinin ham verilere göre daha düşük olduğu belirlenmiştir. Bu sonuçlarla uyumlu olarak öznitelik sayısı azaltıldığında daha hızlı bir karar destek sisteminin elde edileceği sonucuna varılmıştır.

Bu çalışmada makine öğrenmesi sınıflandırıcı modelleri için hiperparametre optimizasyonu, Bayes Optimizasyonu yöntemi kullanılarak gerçekleştirilmiştir. Bu işlemler için geçen süre hesaplanarak Çizelge 4.22'de verilmiştir.

Çizelge 4.22. Bayes hiperparametre optimizasyon tabanlı sınıflandırıcılarda yürütme süresi (dak/sn)

	K-EYK	DVM	Karar Ağacı
Bayes hiperparametre optimizasyon	68,632	592,97	25,851

Çizelge 4.22'de görüldüğü gibi Karar Ağacı sınıflandırıcısı (25,851 sn) üç model arasında en düşük toplam yürütme süresini verirken, K-EYK ve DVM sınıflandırıcılarının yürütme süreleri sırasıyla 68,632 sn ve 592,97 sn olarak gerçekleşmiştir.

4.3.3. Deneysel_çalışma_3 sonuçlarının literatürdeki benzer çalışmalarla karşılaştırılması

Deneysel çalışma sonunda elde ettiğimiz sonuçlar üçüncü bölümde özetlenen benzer literatür çalışmalarıyla karşılaştırılmıştır. MRMR yöntemiyle veri kümesinden öznitelik seçimi ve Bayes Optimizasyonu ile sınıflandırıcı hiperparametreleri kullanılarak elde edilen sınıflandırma sonuçları, literatürde daha önce yapılmış çalışmalarla karşılaştırıldı ve sonuçlar Çizelge 4.23'de sunuldu. Literatürde, SD-VANET'lere karşı gerçekleştirilen DDoS saldırılarının, makine öğrenmesi yöntemleri kullanılarak tespitine yönelik yapılmış çok fazla çalışma bulunmamaktadır. Çizelge 4.23'de karşılaştırılan çalışmaların çoğu YazılımTanımlı Ağ yapısının kablolu kısmına yönelik yapılan saldırıların tespitinde kullanılan makine öğrenimi yöntemleri kullanılarak yapılan çalışmaları kapsamaktadır.

Çizelge 4.23. Literatürdeki çalışmaların karşılaştırılması

Veri kümesi	Öznitelik Seçimi	Makine Öğrenimi Alg.	Doğruluk (%)	Ref
Kendi veri kümeleri	Öznitelik seçimi yok	YOK deep learning model.	95	50
Kendi veri kümeleri CICDDoS 2019	Öznitelik seçimi yok	ESA model	Ortalama 97,45	51
NSL-KDD	Gain Ratio	Rastgele Orman	82,28	52
CICDDoS 2019	Öznitelik seçimi yok	DSA	94,57	53
Kendi veri kümeleri	Öznitelik seçimi yok	GDVM	97	54
Kendi veri kümeleri	Öznitelik seçimi yok	MLP DVM Rastgele Orman Karar Ağacı	Ortalama 92,5 Ortalama 92,5 Ortalama 99 Ortalama 99	55
NSL-KDD	Öznitelik seçimi yok	K-EYK ELM HELM	84,29	56
NSL-KDD	PCA	GA+DVM	98,907	57
Kendi veri kümeleri	Entropy tabanlı seçim	ESA YOK	Ortalama 94 Ortalama 93	58
CICDDoS 2019 CICIDS 2018	Öznitelik seçimi yok	GTB	Ortalama 97,1	59
Kendi veri kümeleri	Öznitelik seçimi yok	UKSB, ESA	99,4	60
Kendi veri kümemiz	MRMR	DVM, K-EYK, ve Bayes Optimizasyonu tabanlı Karar Ağacı	99,35	

5. SD-VANET MİMARİSİNE YÖNELİK DDoS SALDIRILARININ GERÇEK ZAMANLI TESPİTİ

SD-VANET mimarisi içerisinde yeralan kontrolcüyü hedef alan DDoS saldırılarının tespiti için SD-VANET_Guard olarak adlandırdığımız bir güvenlik sistemi önerilmiştir. Bu sistem, VANET ağlarının Yazılım Tanımlı Ağ kontrolcüsü içinde çalışacak şekilde ve DDoS saldırılarına karşı korunmasına yardımcı olmak için tasarlanmıştır. Tespit metodolojimiz Ryu kontrolcü ile veri toplama, Çok Görevli Öğrenme Ağı (Multi-Task Learning Network -MT-LNet) ile veri işleme ve trafik sınıflandırmasını içermektedir.

5.1. Saldırı Tespit Modülü

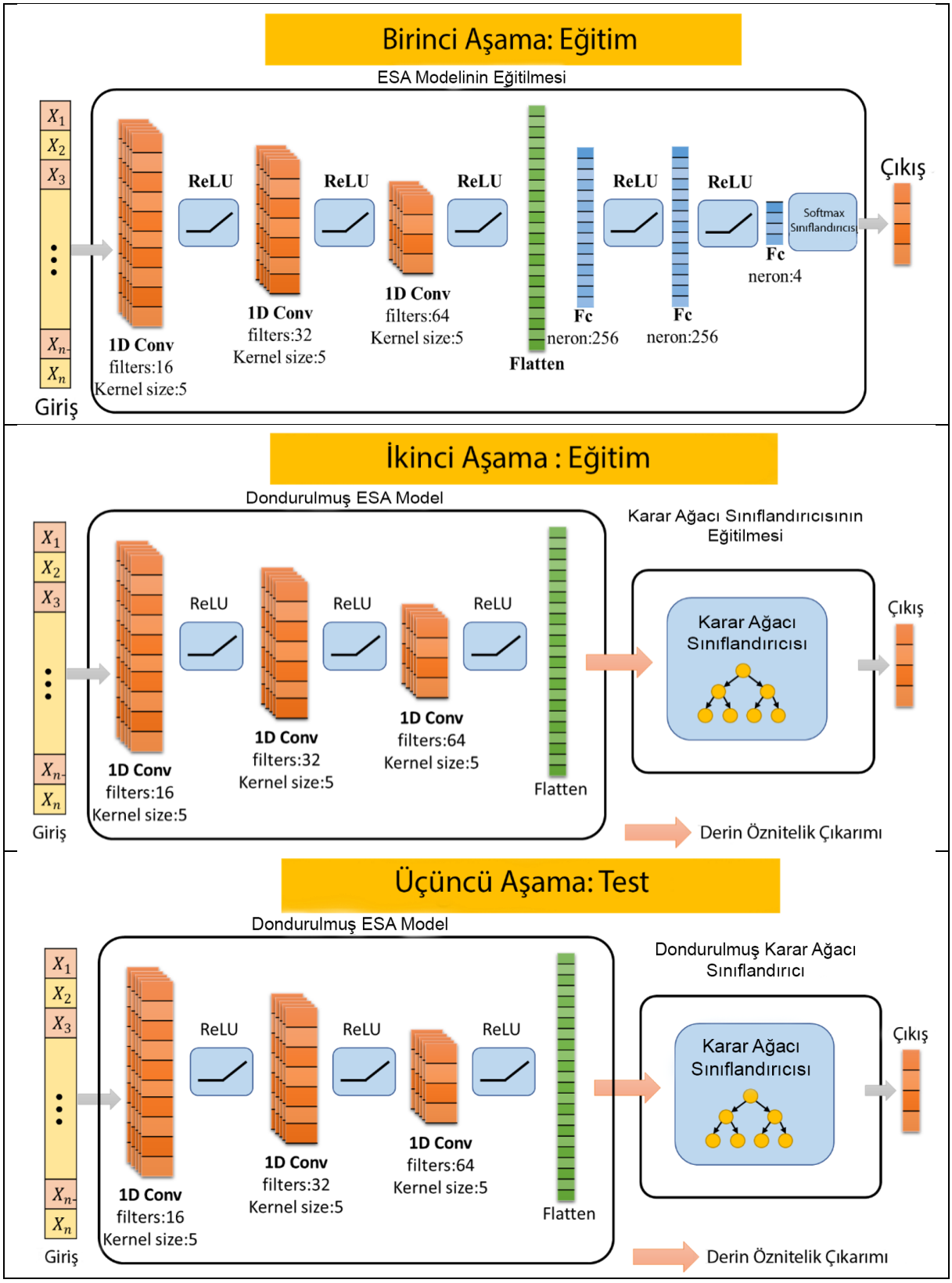
Tespit metodolojimiz Ryu kontrolcü ile veri toplama Multi-Task Learning Network (MT-LNet) olarak adlandırılan hibrit model ile veri işleme ve trafik sınıflandırmasını içermektedir. SD-VANET_Guard güvenlik sisteminde, MT-LNet kontrolcüye yerleştirir. DDoS saldırısını tespit etmek için MT-LNet kullanılır. Normal trafik tarafından oluşturulan iyi huylu akış girişlerini ve DDoS saldırı trafiği tarafından oluşturulan kötü niyetli akış girişlerini tanımlayabilir. Saldırı tespiti şu şekilde gerçekleşir.

Saldırı tespiti iki aşamaya ayrılmıştır: MT-LNet modelinin eğitim aşaması ve gerçek zamanlı tespit aşaması. Eğitim aşamasında, bir dizi kötü niyetli saldırı trafiği, iyi huylu trafik özelliği ve bunlara karşılık gelen hedef değerleri (yani iyi huylu trafik için “0” ve kötü niyetli saldırı trafiği için “1”) eğitim veri kümesi olarak kullanılır. MT-LNet 'nin giriş parametreleri olarak çıkarılan özellik parametreleri ve çıkış parametreleri olarak hedef değerler ile mevcut bir MT-LNet modeli oluşturulmuştur.

Gerçek zamanlı saldırı tespit aşamasında, akış tablosu girişi toplama modülü öncelikle her bir anahtarın tüm akış tablosu girişlerini elde eder ve ardından akış istatistikleri mesajındaki akış girişlerini tek tek işleyerek akış girişlerinin özellik değerlerini çıkarır. Özellik değerleri, akış girişinin iyi huylu veya kötü huylu olup olmadığını belirlemek için farklı protokol türlerine göre MT-LNet eğitim modeline aktarılır. Kötü niyetli trafik olduğu tespit edilince bir DDoS saldırı uyarısı oluşturulur.

Bu çalışmada, DDoS saldıları tespiti için derin öğrenme ve geleneksel makine öğrenme modelini bir arada kullanıldığı hibrit bir model önerilmiştir. MT-LNet olarak adlandırılan bu

model 3 aşamadan oluşmaktadır. Bu aşamaları içeren MT-LNet modelin genel akışı Şekil 5.1'de verilmiştir.

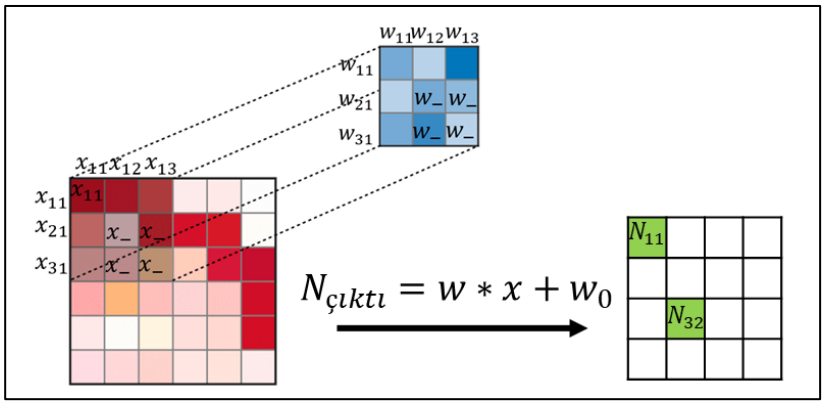


Şekil 5.1. Saldırı tespiti için önerilen MT-LNet sistem mimarisi

Şekil 5.1'de görüleceği üzere, önerilen MT-LNet modelinin birinci aşamasında önerilen ESA mimarisi eğitilmektedir. Bu Evrişimsel Sinir Ağı (ESA - Convolutional Neural Network-CNN) mimarisi şekil 5.1'de gösterildiği gibi 3 evrişim ve 2 tam bağlı katmandan oluşmaktadır. Önerilen modelin ikinci aşamasında öncelikle eğitilmiş ESA mimarinsin parametreleri kullanılarak derin öznitelikler elde edilir. Daha sonra bu derin özellikler ile Karar Ağacı sınıflandırıcısı eğitilir. Önerilen modelin son aşamasında ise test veri kümesindeki örnekler kullanılarak önerilen modelin doğruluğu hesaplanılır. Bölüm devamında önerilen modelin aşamaları ayrıntılı olarak ele alınmıştır.

Birinci aşama: Önerilen ESA modeli

Derin öğrenmenin gelişmesiyle birçok alanda aktif olarak uygulanmaya başlamıştır. Geleneksel makine öğrenme modellerin aksine derin öğrenme mimarilerinde uygun öznitelikleri çıkarmak için öğrenme tabanlı bir yaklaşım kullanmaktadır. Bu yaklaşımda giriş özniteliklerinden öznitelikleri çıkartmak için evrişim katmanları kullanılmaktadır. Evrişim katmanı temel olarak bir filtreleme işlemidir. Bu filtreleme işlemi ile derin öznitelikler elde edilmektedir. ESA'ların en önemli avantajı filtreleme işleminde kullanılan sayısız filtrenin parametreleri eğitim boyunca güncellenir. Şekil 5.2'de evrişim işlemi için örnek verilmiştir. Burada w filtre ağırlıkları (eğitilmesi gereken parametre), x girdi, w_0 bias ve $N_{çıktı}$ evrişim çıktısını temsil etmiştir.



Şekil 5.2. Evrişim katmanında evrişim süreci.

Bu çalışmada DDoS saldırılırının tespiti için önerilen MT-LNet modelinde derin öznitelikler kullanılmıştır. Derin özniteliklerin elde edilmesi için Çizelge 5.1'de verildiği gibi bir ESA mimarisi kullanılmıştır.

Çizelge 5.1. Önerilen tek buyutlu (1D) evrişimli sinir ağı mimarisi, fc: Tam bağlı katman (Fully connected layer), Conv1D: Tek boyutlu evrişim (1D convolution)

Katman	Parametreler	Çıktı Boyutu
Input	Input Model	89×1
Conv1D	filters:16, kernals:5	85×16
ReLU		86×16
Conv1D	filters:32, kernals:5	81×32
ReLU		82×32
Conv1D	filters:64, kernals:5	77×64
ReLU		78×64
Flatten		4928
Dense	nerons:256	256
ReLU		256
Dense	nerons:256	256
ReLU		256
Dense	nerons:4	4
Softmax	Output Model	4

Çizelge 5.1'de gösterildiği gibi önerilen tek buyutlu ESA mimarisinde 3 evrişim katmanı ve 3 tam bağlı katman kullanılmıştır. Her evrişim ve tam bağlı katmanın sonunda ReLU aktivasyon fonksiyonu kullanılmıştır. Ağ mimarinin son katmanında softmax sınıflandırıcı kullanılmıştır.

Önerilen MT-LNet modelinin birinci aşamasında DDoS eğitim veri veri kümesi ile ESA mimarisi eğitilmiştir. Eğitim için İkili Çapraz Entropi (Binary Cross-Entropy) kayıp fonksiyonu kullanılmıştır. İkili Çapraz Entropi kayıp fonksiyonu Denklem 5.1'deki gibi tanımlanmaktadır.

$$L = -\sum_{i,j}^{M} y_{i\times j} log\left(P_{i\times j}\right) + \left(1 - y_{i\times j}\right) log\left(1 - P_{i\times j}\right) \tag{5.1}$$

Burada $y_{i\times j}$ ve $P_{i\times j}$, sırası ile $i \times j$ konumundaki pikselin gerçek değerini ve tahmin değerini temsil etmektedir. L değeri ise ortalama hata değerini göstermektedir.

İkinci Aşama: Derin öznitelikler ile Karar Ağacı Sınıflandırıcısının eğitilmesi

Karar Ağacı, sınıflandırma problemlerinde kullanılan denetimli öğrenme algoritmasıdır. Karar Ağacı öğrenimi sırasında, öğrenilen bilgi bir ağaç üzerinde modellenir. Karar Ağacında her bir düğüm bir özelliği, ağacın yaprak düğümleri seviyesindeki sınıf etiketleri ve bu yapraklara giden ve baştan ayrılan dallar, özellikler üzerindeki işlemleri ifade eder. Karar Ağacı öğrenmede, veri kümesi bir dizi karar/kural uygulanarak daha küçük kümelere bölünür. Bu işlem yinelemeli olarak tekrarlanır ve tekrarlama işlemi sınıflandırmayı etkileyene kadar devam eder. Karar Ağacı, insanların karar verme modelinden yola çıkarak, bir dizi kural kullanan bir denetimli makine öğrenimi algoritmasıdır. Karar Ağacı hem sınıflandırma hem de Regresyon problemlerinde kullanılabilen denetimli bir öğrenme tekniğidir. Bununla birlikte sınıflandırma problemlerinin çözümünde daha çok tercih edilmektedir. İç düğümlerin bir veri kümesinin özelliklerini temsil ettiği, dalların karar kurallarını temsil ettiği ve her yaprak düğümün sonucu temsil ettiği ağaç yapılı bir sınıflandırıcıdır. Bir Karar Ağacında, karar düğümü ve yaprak düğümü olmak üzere iki düğüm vardır. Karar düğümleri herhangi bir karar vermek için kullanılır ve birden fazla dala sahiptir, Yaprak düğümleri ise bu kararların çıktısıdır ve başka dallar içermez.

Karar Ağacında, verilen veri kümesinin sınıfını tahmin etmek için algoritma ağacın kök düğümünden başlar. Daha sonra girdi değerleri kök özeliği kullanılarak karşılaştırmaya göre dalı takip eder ve bir sonraki düğüme atlar. Bir sonraki düğüm için, algoritma yine öznitelik değerini diğer alt düğümlerle karşılaştırır ve daha ileri gider. Ağacın yaprak düğümüne ulaşana kadar işleme devam eder. Tüm süreç, aşağıdaki algoritma kullanılarak tekrarlanır:

- Adım-1: Ağaca, tüm veri setini içeren kök düğümle başlanır.
- Adım-2: Öznitelik seçimi kullanarak veri kümesindeki en iyi öznitelik bulunur.
- Adım-3: En iyi öznitelik için örneklerin olsalıksal dağılımları elde edilir. Bu olasılıksal dağılımlar ile alt kümeler seçilir.
- Adım-4: En iyi özelliği içeren Karar Ağacı düğümleri oluşturulur.
- Adım-5: Maksimum ağaç derinliğine ulaşana kadar Adım 3 ve 4 tekrarlanır.

Karar Ağacında öznitelik seçimi için Gini Index seçilmiştir. Bununla birlikte ağıcın maksimum derinliği 12 olarak ayarlanmıştır.

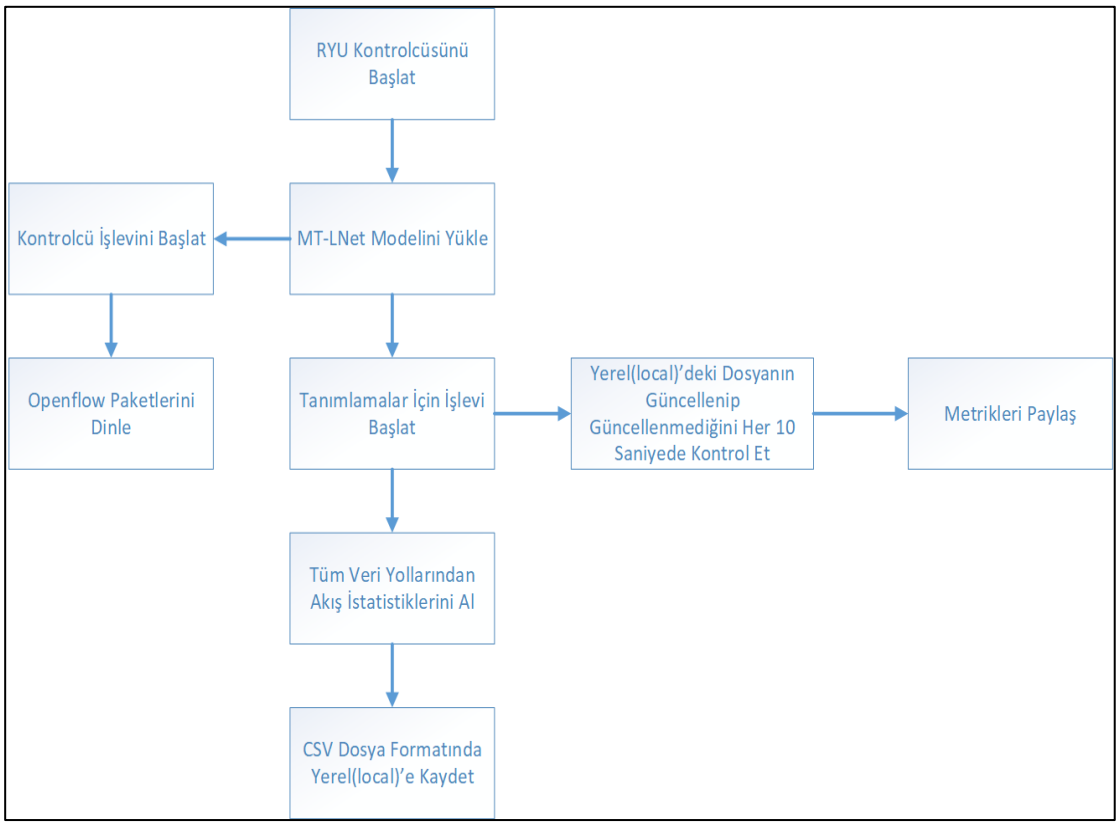
Üçüncü aşama: Test aşaması

Önerilen MT-LNet modelinin son aşamasında oluşturulan modelin performans analizi yapılmıştır. Test aşamasında MT-LNet modelin tüm parametreleri dondurulur. Daha sonra modelin daha önce hiç görmediği örnekler için MT-LNet modeli kullanılarak örneklerin tahminleri elde edilir. Son olarak elde edilen tahminlerin gerçek değerleri ile karşılaştırılarak performans değerleri elde edilir.

5.2. Akış Tablosu Giriş Toplama Modülü

Tespit metodolojimiz Ryu kontrolcü ile veri toplama MT-LNet ile veri işleme ve trafik sınıflandırması içermektedir. Önerilen DDoS tespit metodolojimizin genel görünümü, uygulama ayrıntılarını bu bölümde detaylandırılmaktadır.

Test yatağında, Ryu modüler uygulaması kullanılarak veri kümesi toplanmıştır. Ryu kontrolcüsünde, her 10 saniyede bir anahtarların istatistiklerini okuyan ve bunları bir dosyada saklayan bir modül tasarlanmıştır (Şekil 5.3).



Şekil 5.3. SD-VANET_Guard DDoS saldırı tespit sistemi akış diyagramı

OpenFlow tabanlı yaklaşımda, ağ trafiği özelliklerinin toplanması için, kontrol düzlemi (SDN kontrolcüsü) veri düzlemi cihazlarına (OpenFlow anahtarları) bir durum isteği (OFPC_FLOW_STATS_REQUEST) gönderir. Veri düzleminde bulunan iletim cihazları ağ trafiği özelliklerini göndererek bir veya daha fazla yanıt mesajıyla (OFPC_FLOW_STATS_REPLY) yanıt verir. OpenFlow anahtarlardan "OFPC_FLOW_STATS_REPLY" mesajı gelince kontrolcüde bulunan akış istatistikleri toplama modülüne gelir. Toplama modülüne gelen akış istatistikleri lokal dosyaya csv formatinda kaydedilir. Dosya, timestamp, datapath_id, flow_id, ip_src, tp_src, ip_dst, tp_dst, ip_proto, icmp_code, icmp_type, flow_duration_sec, flow_duration_nsec, idle_timeout, hard_timeout, flags, packet_count, byte_count, packet_count_per_second, packet_count_per_nsecond, byte_count_per_second, byte_count_per_nsecond gibi çeşitli trafik özniteliklerinden oluşmaktadır.

Ryu kontrolcü içinde birden fazla thread (iş parçacığı) çalışmaktadır. Kontrolcü çalışmaya başlayınca kontrolcüde kayıtlı olan bir iş parçacığı birden fazla iş parçacığı oluşturuyor. Oluşturulan her iş parçacığı farklı görevler yerine getirmektedir.

- Veri düzleminde bulunan iletim cihazları tarafından gönderilen "OFPT_PACKET_IN" mesajlarını dinleyip akış kuralı yazan iş parçacıkları
- "OFPC_FLOW_STATS_ REQUEST" mesajıyla veri düzleminde buluna cihazlardan durum isteği, akış kuralı istatistikleri talep eden iş parçacıkları.
- Veri düzleminde bulunan iletim cihazları "OFPC_FLOW_STATS_ REPLY" mesajlarıyla ağ tafiği ile ilgili bütün bilgileri kontrolcüye gönderir. Veri düzleminde bulunan iletim cihazlarından ağ trafiği ile bilgilerin alınmasından sorumlu iş parçacığı aldığı akış istatistiklerini local dosyaya csv formatında kaydeder.
- Local de bulunan dosyayı okuyan iş parçacığı. Bu iş parçacığı 10 saniyede bir dosyayı kontrol eder. Eğer dosyada akış kuralı isteği var ise tespit modülünü çalıştırır. Akış kuralı isteği yok ise 10 saniye boyunca bekler. Bu süreç bir döngü içinde devam etmektedir. Tespit gerçekleştiği zaman tespit modülü tarafından sonuçlar influx'a gönderilmektedir.
- Kontrolcü, anahtara bir "OFPT_FEATURES_REQUEST" mesajı göndererek anahtarın özelliklerini (MAC adresi, bağlantı noktaları ile ilgili bilgileri, eylem seti, akış tablosu vs.) talep eder. Ardından, anahtar istenen bilgileri içeren bir "OFPT_FEATURES_REPLY" mesajı ile kontrolcüye yanıt verir. Kontrolcü, anahtardan aldığı bu bilgileri topolojinin oluşturulması, anahtara yeni kural yazılması gibi işlemler için saklar veya kullanır.

Kontrolcü ve anahtar bu şekilde birbirleriyle bağlantı sağlar. Bu işlemin gerçekleşmesi de iş parçacığı sayesinde sağlanır.

Deneysel amaçlı oluşturulan topolojimizde sflow ayri bir sunucu icerisinde (sFlow-RT) yer almaktadır.

Deneyde, sanal anahtar yazılımı OpenvSwitch (OVS) kullanılmıştır. Üst uygulamalara API'ler sağlamak için toplayıcı olarak sFlow-RT seçilmiştir. sFlow-RT gerekli akış verilerini toplamak üzere yapılandırılmıştır. Sanal bir makinede yapılandırılan sFlow-RT çalıştırılarak mininet emülatörü üzerinde çalışan bütün anahtarlara erişim sağlar. Sflow-RT erşim sağladığı anahtarlardan her 10 saniyede bir Plug-in (eklenti) kullanarak verileri http endpoint üzerinden erişilebilir hale getirir. Prometheus, her 10 saniyede bir sflow tarafından elde edilen verileri alarak zaman damgalı olarak kaydeder. Daha sonra Grafana, Prometheus sunucusundan anlaşılabilir olan metrikleri ve uyarıları alarak görselleştirir.

6. SD-VANET_GUARD DDOS SALDIRI TESPİT SİSTEMİ PERFORMANS DEĞERLENDİRMESİ

Bu bölümde, SD-VANET_Guard DDoS saldırı tespit sisteminin performans değerlendirilmesi yapılmaktadır. Geliştirilen SD-VANET_Guard saldırı tespit sisteminin performanısı iki faklı Yazılım Tanımlı Ağ tabanlı ağ yapısı oluşturularak test edilmiştir. Birinci senaryoda Yazılım Tanımlı Ağ tabanlı kablolu ağ yapısı oluşturularak modelin performansı test edilmiştir. İkinci senaryada ise SD-VANET ağ yapısı oluşturularak kablosuz ortamda önerilen modelin performansı test edilmiştir.

6.1. Simülasyonda Kullanılan Araçlar

Yazılım Tanımlı Ağ tabanlı VANET kontrolcüsünü hedef alan DDoS saldırılarının tespiti için geliştirilen SD-VANET_Guard DDoS saldırı tespit sisteminin performansını test etmek amacıyla kullanılan araçlar ile ilgili bu bölümde bilgi verilmiştir.

SFLOW

sFlow, InMon Inc. tarafından geliştirilmiştir. sFlow, ağ verilerini toplamak ve analiz etmek için kullanılan örnekleme tabanlı bir araçtır. Ayrıca, sFlow aracı ağ performansını izlemek ve ağlardaki potansiyel darboğazları veya sorunları belirlemek için kullanılabilir. sFlow izleme sistemi, birden fazla dağıtılmış sFlow agent ve bir sFlow toplayıcısından oluşan merkezi bir kontrol mimarisidir. Tüm arayüzlerde aynı anda kablo hızında uygulama düzeyinde trafik akışlarını sürekli olarak izleme olanağı sağlar. sFlow kullanılarak, açık sanal anahtarlar (OVS), fiziksel anahtarlar, ana bilgisayarlar vb. gibi çok çeşitli ağ cihazlarından trafik örnekleri toplanabilir. sFlow sistem mimarisi 3 temel bileşenden oluşur.

sFlow Agent: Arayüz sayaçlarını ve akış örneklerini sFlow datagramlarında birleştiren ve bunları UDP aracılığıyla hemen sFlow toplayıcılarına gönderen bir yazılım işlemidir.

sFlow Toplayıcısı(Collector): sFlow datagramlarının toplandığı ve depolandığı sunucu olarak görev yapar.

sFlow Analizörü (Analyzer): sFlow toplayıcısı tarafından depolan sFlow datagramları analiz eder ve ağ trafik akışına gerçek zamanlı veya geçmişe yönelik genel bakış sağlar. Ağ

parametreleri hakkında derinlemesine bilgi sağlar. Siber saldırılar sonucu ağda oluşan düzensizlikleri tespit etmek için kullanılabilir.

INFLUXDB

InfluxDB, InfluxData tarafından geliştirilen isteğe bağlı kapalı kaynaklı bileşenlere sahip açık kaynaklı, SQL benzeri bir sorgu dili sunan, go dilinde yazılmış şemasız bir zaman serisi veritabanıdır. InfluxDB, operasyon izleme, nesnelerin interneti sensör verileri, uygulama metrikleri ve ayrıca gerçek zamanlı analitik gibi alanlarda büyük depolama alanı kullanılabilirliği ve bir dizi verinin hızlı bir şekilde alınması için kaliteli sorgular yazmak üzere tasarlanmıştır.

Prometheus

Prometheus, açık kaynaklı bir sistem izleme ve uyarı araç setidir. Prometheus çok boyutlu bir veri modelini kolaylaştırır, çeşitli veri kaynaklarından gelen tüm verileri zaman damgalı değerler akışı olarak toplar ve metrik adı ve anahtar/değer çiftleri ile tanımlanabilen zaman serisi verilerini depolar. Prometheus kendine özgü sorgu dili PromQL kullanır. PromQL, HTTP üzerinden çekme modelini kullanan ve kullanıcının zaman serisi veritabanından gerçek zamanlı metrikleri seçmesine ve toplamasına olanak tanır. PromQL, güçlü ve esnek bir sorgulama ve gerçek zamanlı uyarı sağlar ve yüksek performanslı zaman serisi veri toplamayı kolaylaştırır. Ayrıca, sorgulanan veriler HTTP API kullanılarak tarayıcı üzerinden grafik veya tablo olarak görselleştirilebilir.

Grafana

Grafana, zaman serilerini ve metrikleri sorgulamak ve görselleştirmek için kullanılan açık kaynaklı bir analitik ve etkileşimli görselleştirme web uygulamasıdır. Verileri grafiksel bir gösterimde oluşturmak, paylaşmak, keşfetmek ve anlamak için birçok yol sağlar.

Mininet

Mininet binlerce düğüme kadar çok büyük ölçekli topolojiler oluşturmaya ve bunlar üzerinde çok kolay bir şekilde test yapmaya izin veren açık kaynak kodlu bir emülasyon aracıdır. Tek bir sistemin eksiksiz bir ağ gibi davranmasını sağlamak için konteyner tabanlı sanallaştırma

kullanır. OpenFlow tabanlı uygulamaları geliştirmek ve test etmek için basit, sağlam ve ucuz bir ağ aracıdır. Yazılım Tanımlı Ağ tabanlı ağların kolayca oluşturmasına, özelleştirmesine, prototiplemesine, hata ayıklama ve test edilmesine imkân tanır. Yazılım Tanımlı Ağ tabanlı ağlar oluşturma ve test etme için basit ve genişletilebilir Python API desteği sunar.

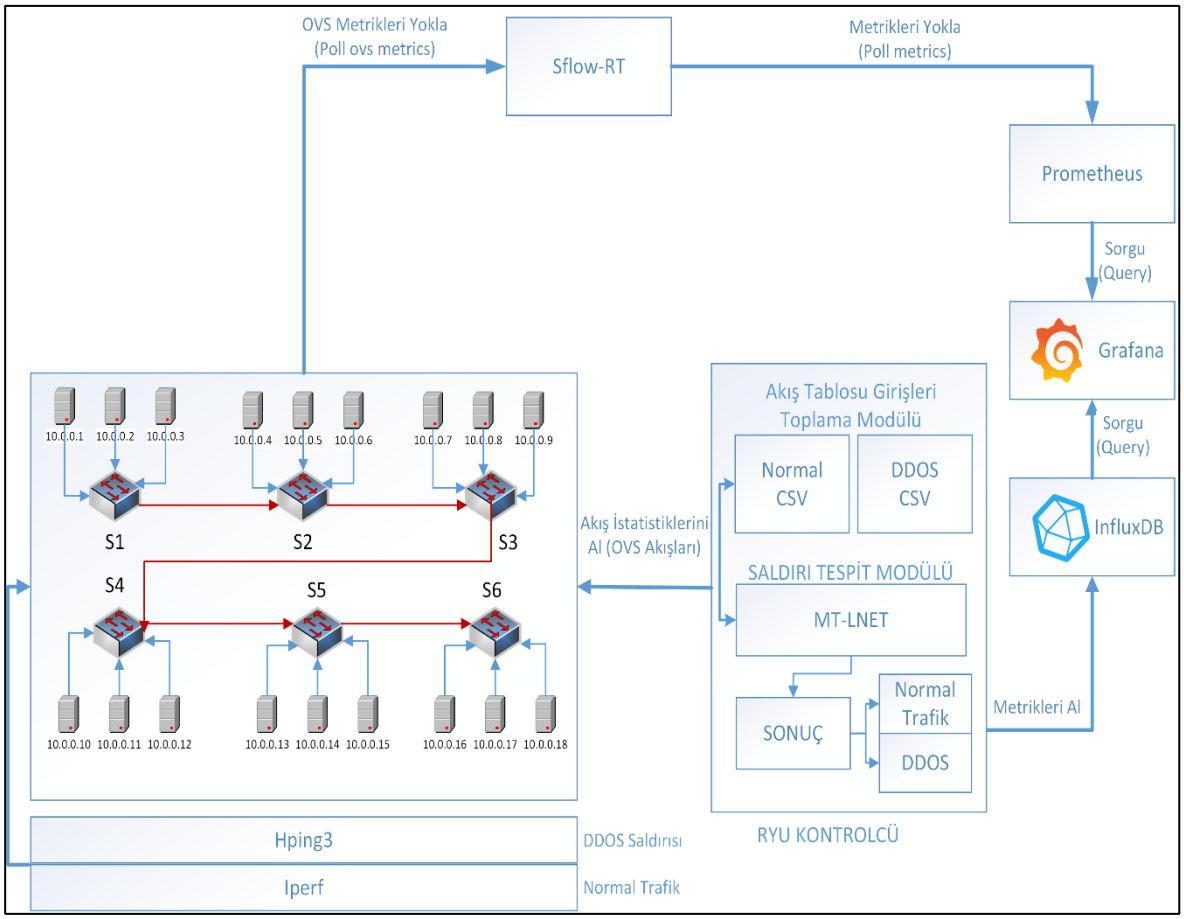
6.2. Senaryo_1: SD-VANET_Guard DDoS Saldırı Tespit Sistemi Performans Değerlendirmesi

Tez kapsamında geliştirilen SD-VANET_Guard DDoS saldırı tespit sisteminin performansının sağlıklı bir şekilde belirlenebilmesi için Yazılım Tanımlı Ağ yapısı oluşturulup model performansı test edilmiştir. Simülasyon ortamı aşağıdaki gibi oluşturulmuştur:

6.2.1. Senaryo_1 için oluşturulan deneysel yazılım tanımlı ağ topolojisi

DDoS saldırı verilerini ve normal ağ trafiği verilerini toplamak için oluşturulan deneysel topoloji ile SD-VANET_Guard saldırı tespit sisteminin entegrasyonu Şekil 6.1'de gösterilmektedir.

Deneysel Yazılım Tanımlı Ağ topolojisi 6 anahtar ve 18 kullanıcıdan (kullanıcı IP aralığı: 10.0.0.1-10.0.0.18) oluşmaktadır. Ryu kontrolcü ve Mininet aynı sanal makinde, sFlow ise farklı bir sanal makinede çalışmaktadır. Mininet ve Ryu kontrolcü, 4G RAM ve 2 çekirdek CPU ile 20-04-1 Ubuntu'ya kurulurken, sFlow 4G RAM ve 1 çekirdek CPU ile 20-04-1 Ubuntu'ya kurulmuştur. Çalışmada kullanılan algoritmalar aracılığıyla en iyi ağ parametrelerini elde etmek için topoloji, Ubuntu 20.04 LTS işletim sistemi üzerinde çalışan 32 Gb RAM, Intel i7-1165 g7 ana bilgisayar üzerinde çalıştırılmıştır.



Şekil 6.1. Senaryo_1 deneysel Yazılım Tanımlı Ağ topolojisinin SD-VANET_Guard saldırı tespit sistemi ile entegrasyonu

Normal trafik senaryosu

Ağı yüklemek ve kapasitesini değerlendirmek için Iperf aracı kullanılmıştır. IPerf, istemci-sunucu modeli kullanan ve sunucu ile istemci düğümleri arasında veri akışları şeklinde trafik oluşturan bir araçtır. IPerf, rastgele trafik üretir, yani bir ana bilgisayar sanal ağdaki diğer herhangi bir ana bilgisayara eşit olasılıkla paketler gönderir. Ağı yüklemek için TCP veya UDP akışları oluşturabilir. TCP modunda, bir IPerf istemcisi sunucuya bir TCP akışı aracılığıyla sonsuz miktarda veri iletir. Kullanıcı, bir süreden sonra, IPerf aracının TCP bağlantısını iptal eder ve ekrana farklı istatistikleri ve doğru şekilde iletmeyi başardığı toplam veri miktarını yazdırır.

Mininet Flow Generator ile IPerf kullanılarak kullanıcılar arasında rastgele akışlar oluşturulmuştur. Her akış için, ağdan bir kullanıcı sunucu olarak bir kullanıcı ise istemci olarak rastgele seçilir. Rastgele seçilen kullanıcılar rastgele seçilen kullacılara IPerf ile TCP ve UDP bağlantıları kurmaktadır.

ApacheBench, bir web sunucusunu HTTP istekleriyle doldurup gecikme ve başarı ölçümlerini kaydederek performansını http benchmark yaparak ölçer. ApacheBench ile büyük miktarda http isteği atılır. Simülasyondaki bütün kullanıcılarda, python HTTP server ve IPerf server çalıştırılarak TCP ve UDP trafik akışı oluşturulmaktadır.

Ağda normal trafik verisi aktığında, Ryu kontrolcü OpenFlow anahtarlardan "OFPC_FLOW_STATS_REPLY" mesajı gelince akış istatistiklerini lokal dosyaya csv formatinda kaydeder. Kaydedilen özniteliklere trafik sınıfı eklenerek öznitelik sayısı arttırılır. Bu sınıfta normal trafik "0" olarak etiketlenir.

Saldırı Senaryosu

Hping3 paket üreteci ile rastgele seçilen kullanıcılardan rastgele seçilen hedeflere hping3 -2 -V -d 120 --rand-source komutu ile UDP (120 bayt+protokol başlığı) flood saldırı gerçekleştirilmiştir.

Hping3 paket üreteci ile rastgele seçilen kullanıcılardan rastgele seçilen hedeflere hping3 -p 80 -d 120 -w 200 --rand-source komutu ile TCP (120 bayt+protokol başlığı) flood saldırı gerçekleştirilmiştir. TCP pencere boyutu 200 bayt olarak ayarlanmıştır. Deneysel çalışma kapsamında oluşturulan normal trafik saldırı anında kesilmeden devam ettirilmiştir.

Ağda saldırı trafik verisi aktığında, Ryu kontrolcü OpenFlow anahtarlardan "OFPC_FLOW_STATS_REPLY" mesajı gelince flow istatistiklerini lokal dosyaya csv formatında kaydeder. Trafik sınıfında saldırı trafiği "1" olarak etiketlenir.

Veri Kümesi

DDoS saldırıları verilerini ve normal ağ trafiği verilerini toplamak için 10 dakika boyunca deneysel bir simülasyon gerçekleştirilmiştir Simülasyon sırasında, TCP, UDP paketleri başlangıçta normal paketler olarak ve daha sonra DDoS saldırı paketleri olarak gönderilmiştir. Bu paketlerin her biri 512 baytlık bir yüke sahiptir. Elde edilen veri kümesi 698767 örnek, 21 öznitelik ve "Normal", "DDoS Saldırıları" olmak üzere iki sınıf içermektedir. Örneklerin 309.602'si normal ağ trafiği akış verisi, 389.165'i ise DDoS saldırı trafiği akış verisidir. Veri kümesindeki tüm öznitelikler Yazılım Tanımlı Ağ mimarisine özgü trafik akışı istatistiklerinden oluşmaktadır. Veri kümesindeki öznitelikler Çizelge 6.1'de sunulmuştur.

Çizelge 6.1. Veri kümesindeki öznitelikler

timestamp	datapath_id	flow_id
ip_src	tp_src	ip_dst
tp_dst	ip_proto	icmp_code
icmp_type	flow_duration_sec	flow_duration_nsec
idle_timeout	hard_timeout	flags
packet_count	byte_count	packet_count_per_second
packet_count_per_nsecond	byte_count_per_second	byte_count_per_nsecond
Sınıf: Trafik sınıfı. Bu çalışmada kullanılan veri kümesinde yer alan veriler etiketlenmiş verilerdir. Bu iki sınıf; "0", "1" şeklindedir. "0" normal trafik akışı anındaki verileri, "1" ise DDoS saldırısı anında alınan verileri temsil eder.		

Normal ağ trafik verileri ve saldırı trafiği verileri kontrolcüde etiketlenerek kaydedildikten sonra, MT-LNet eğitim kodu çalıştırılarak model oluşturulmuştur. Elde edilen verinin tümü eğitim aşamasında kullanılmıştır. Eğitim aşamasından sonra elde edilen model kontrolcüde bulunan tespit modülüne kaydedilmiştir.

6.2.2. Senaryo_1 deneysel sonuçlar

Bu bölümde güvenlik sistemi için önerilen tespit modülüne ilişkin performans sonuçları incelenmektedir. SD-VANET_Guard saldırı tespit sistemi senaryo_1 üzerinden test edilmiş ve sonuçlar özetlenmiştir:

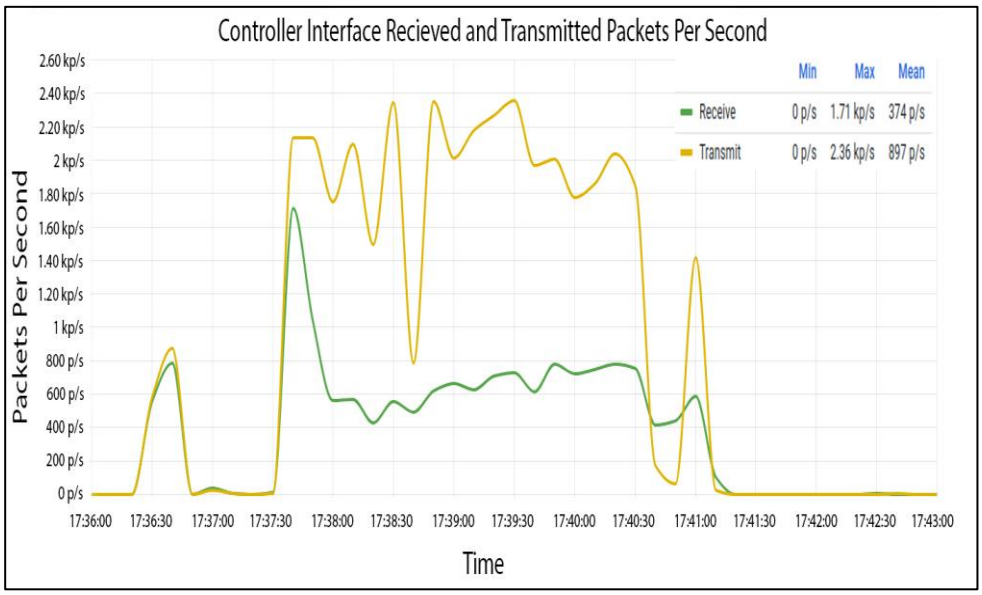
Senaryo_1'de, SD-VANET_Guard saldırı tespit sisteminin DDoS saldırısını tespit etme etkinliğini göstermek için uygulanmıştır. Sistem çalışırken TCP/SYN flood, ve UDP flood içeren karışık DDoS flood saldırıları başlatılmıştır.

Çizelge 6.2. Simülasyonda her bir modülün zaman içindeki eylemleri

DURUM	ETKİNLİK	ZAMAN
İlk Durum	Uygulamaya başlama	17:36:00
	Ryu kontrolcü (akış istatistiği toplama modülü) anahtarlara akış istatistiği isteği gönderir ve cevaplar local dosyaya yazılır.	17:36:00
Tespit Durumu	Tespit modülünün aktif hale gelmesi/başlatılması. (Her 10 saniyede bir yerel dosyayı oku ve tahmini çalıştır, ardından sonucu influxdb'de yayınla)	17:36:00
	Saldırı tespiti	17:37:30
Akış Oluşturma Durumu	Ryu kontrolcü akış kuralı oluşturma/değiştirme (veri düzleminde bulunan cihazlarından akış isteklerine kontrolcü cevap verir)	17:36:00

Çizelge 6.2, senaryo_1'deki her bir modülün başlangıç zamanının ayrıntılı açıklaması verilmiştir. İlk durumda, önceden oluşturulan eğitim veri kümesi kullanılarak eğitililen MT-LNet hibrit modeli çalıştırılmıştır. Bu süreç 17:36:00'de başlamıştır. 17:36:00'da saldırı tespit modülü için bir başlatma komutu üretmiştir. 17:36:00'da tespit modülü başlatılmış ve bu modül 17:37:30'da saldırıyı tespit etmiştir. 17:36:00'da Ryu kontrolcü veri düzleminde bulunan iletim cihazlarından gelen akış isteklerine akış kuralı oluşturarak cevap verir.

Yazılım Tanımlı Ağ yapısına yeni gelen paketler için anahtarlardaki akış tablosunda bulunan akış girdilerin de eşleşme aranır. Paket, akış girdileriyle eşleşmiyorsa paket başlıkları encapsule edilerek yeni akış kuralı tanımlanması için anahtar tarafından kontrolcüye gönderilir. DDoS saldırısında saldırgan büyük miktarda veriyi veri düzlemindeki anahtarlara gönderir. Saldırı anında akış tablosu girdileri ile eşleşmeyen büyük miktardaki paket yeni kural talebi için kontrolcüye gönderilir. Şekil 6.2'de görüldüğü gibi 17:37:30'da DDoS saldırısının başlamasıyla kontrolcüye gönderilen paketler artmaktadır. Bu paketler anahtarlar tarafından kontrolcüye gönderilen "OFPT_PACKET _IN" gibi OpenFlow mesajları, kontrolcü tarafından anahtarlara gönderilen "OFPT_PACKET _OUT", akış modifikasyonu gibi OpenFlow mesajlarını içermektedir. DDoS saldırısı nedeniyle anahtaralar tarafından kontrolcüye gönderilen büyük miktardaki akış talebi kontrolcünün verimliliğini ve etkinliğini olumsuz etkiler.



Şekil 6.2. Senaryo_1 deneysel Yazılım Tanımlı Ağ topolojisi kontrolcünün arayüzünden alınan ve verilen paket miktarı

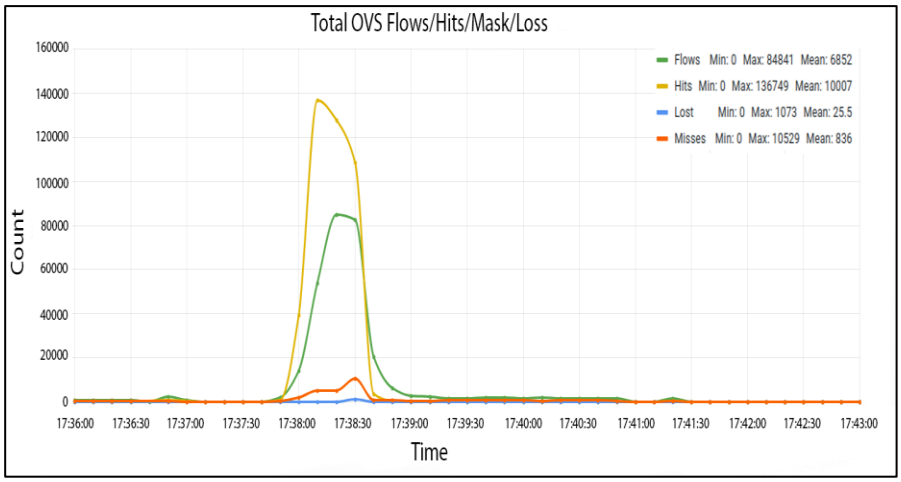
Ağa yeni bir paket geldiğinde, akış tablosunda akış kuralı girişi eşleştirilir. Eşleşmeyen tüm girişler için, anahtar kontrolcüden akış için yeni akış kuralı oluşturmasını talep eder. Anahtar yeni akış kuralı için paketin bir kısmını kontrolcüye gönderir ve paketin diğer bir kısmını ise anahtarda bulunan ara bellekte tutar. Ara bellek dolarsa, paketin tamamını kontrolcüye gönderir. Arabelleğe alınan paketler kontrolcüden gelen "OFPT_PACKET _OUT" veya "OFPT_FLOW_MOD" mesajı aracılığıyla işlenir veya bir süre sonra otomatik olarak düşer. Saldırgan, Yazılım Tanımlı Ağ yapısının bu özelliğinden faydalanarak anahtara sahte akışlar gönderebilir. Anahtar akış tablosu girdileriyle eşleşmeyen akışı her aldığında, kontrolcüye yeni akış kuralı talebinde bulunur. Şekil 6.3'de görüldüğü gibi DDoS saldırısının başladığı 17:37:30 'da kontrolcüye giden yeni akış kuralı isteklerine kısa zamanda kontrolcü büyük miktarda akış kuralı oluşturarak cevap vermiştir. Saldırgan kolayca yeterli sayıda sahte akış üretebilir ve bu da kontrolcünün işlem kapasitesini tüketebilir.

Hits: Kontrolcünün daha önceden oluşturup, OVS anahtarın akış tablosunda tutulan kurallarla eşleşen paket sayısını ifade eder.

Miss: Veri düzleminde bulunan iletim cihazlarına (anahtar, yönlendirici vs.) yeni bir paket geldiğinden akış tablosunda eşleşme aranır. Gelen paket herhangi bir akış girdisiyle eşleşmiyorsa (flow miss) yeni akış kuralı tanımlanması için kontrolcüye gönderilir.

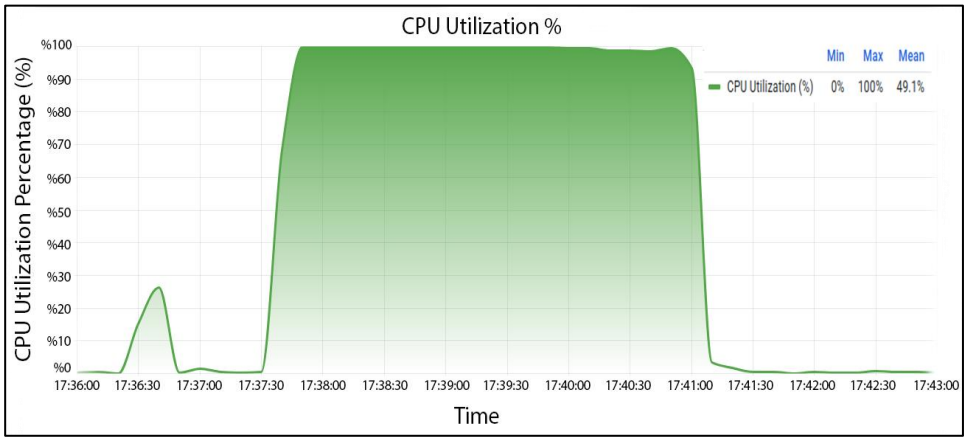
OVS Flow: Datapath içerisindeki flow sayısıdır.

OVS Lost: İşlenmesi için gönderilen ancak drop edilen paket sayısıdır.



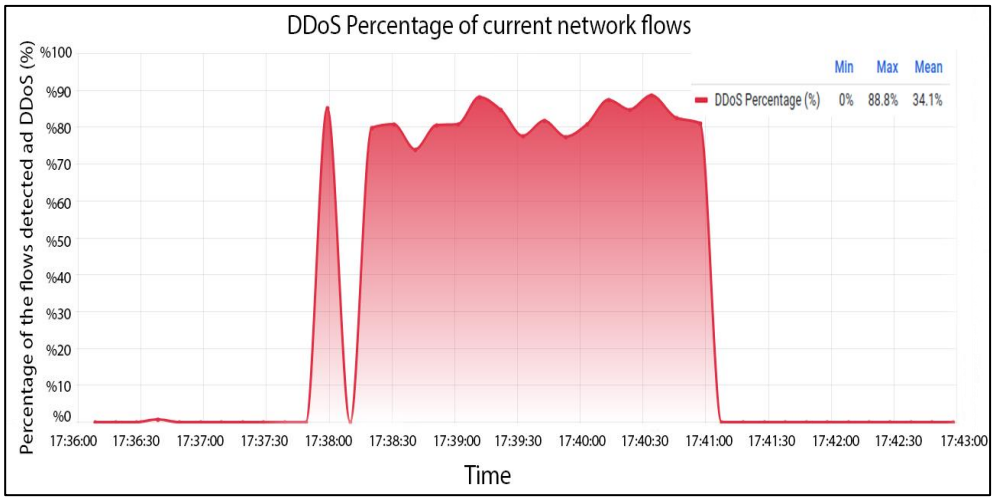
Şekil 6.3. Senaryo_1 deneysel Yazılım Tanımlı Ağ topolojisi toplam OVS akış metrikleri

Bir cihazın merkezi işlem birimi, gelen verileri matematiksel işlemler yardımıyla işler ve hesaplamalar yapar. Merkezi işlem biriminin kullanım yüzdesi bilgisayarın işlem yapma kapasitesine göre değişir. Merkezi işlem biriminin kullanım yüzdesinin yüksek bir değerde olması bilgisayarınızın maksimum seviyede veya çalışan uygulamaların sayısı için normal seviyenin üzerinde çalıştığı anlamına gelir. Saldırgan anahtarlara sahte paketler gönderdiğinde ve anahtarlar tüm talepleri kontrolcüye gönderdiğinde, kontrolcü sahte talepleri karşılamakla meşgul olur. Şekil 6.4'de görüldüğü gibi 17:37:30' da DDoS saldırısının başlamasıyla sahte paketlerin işlenmesi, kontrolcünün verimini ve işlem gücünü tüketmiştir.



Şekil 6.4. Senaryo_1 deneysel Yazılım Tanımlı Ağ topolojisi kontrolcünün işlemci kullanımı

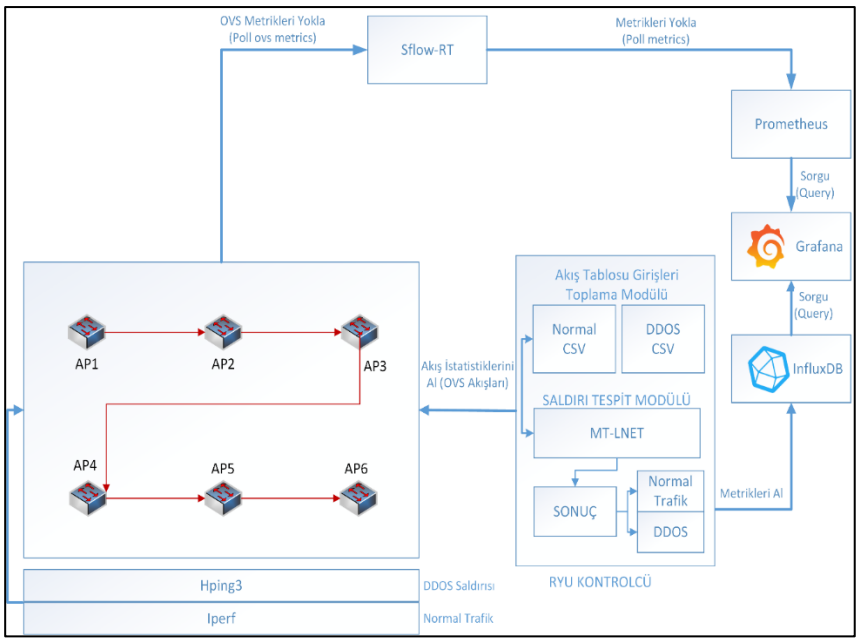
Geliştirilen, Yazılım Tanımlı Ağ açık kaynak Ryu kontrolcüsüne dayalı SD-VANET_Guard saldırı tespit sistemi, kontrolcünün oluşturduğu akış tablosu girişlerine dayalı bir tespit mekanizmasına sahiptir. DDoS saldırısında saldırgan büyük miktarda saldırı trafiğini veri düzleminde bulunan cihazla yönlendirir. Gelen paketler için OpenFlow anahtarda bulunan, akış tablosunda bulunan akış girdileriyle eşleşme aranır, eşleşme yok ise (miss flow), paketler "OFPT_PACKET _IN" mesajıyla kontrolcüye yeni kural yazması için gönderilir. Kısa zaman içinde kontrolcüye büyük miktarda "OFPT_PACKET _IN" mesajı gelir. Gelen kural istekleri için kontrolcü kural yazıp anahtarın akış tablosuna gönderir. Bu mekanizma temel alınarak geliştirilen saldırı tespit mekanizması Şekil 6.5'de görüldüğü gibi 17:37:30 da Yazılım Tanımlı Ağdaki DDoS saldırısını tespit etmiştir. Ağada bulunan mevcut akışların yaklaşık %90'ı DDoS trafiğinin oluşturduğu akışlardan oluşmaktadır. Tespit mekanizması Yazılım Tanımlı Ağdaki saldırı trafiği ve normal ağ trafiğini ayırt etme özelliğine sahiptir.



Şekil 6.5. Senaryo_1 deneysel Yazılım Tanımlı Ağ topolojisi mevcut ağ akışındaki DDoS yüzdesi

6.3. Senaryo_2: SD-VANET_Guard DDoS Saldırı Tespit Sistemi Performans Değerlendirmesi

DDoS saldırı verilerini ve normal ağ trafiği verilerini toplamak için oluşturulan deneysel Yazılım Tanımlı Ağ tabanlı VANET topolojiyisi ile SD-VANET_Guard saldırı tespit sisteminin entegrasyonu Şekil 6.6'de gösterilmektedir.



Şekil 6.6 Senaryo_2 deneysel SD-VANET topolojisi ile SD-VANET_Guard saldırı tespit sistemi entegrasyonu

Simülasyon ortamı aşağıdaki gibi oluşturulmuştur:

6.3.1. Senaryo_2 için oluşturulan deneysel SD-VANET topolojisi

DDoS saldırı verilerini ve normal ağ trafiği verilerini toplamak için oluşturulan deneysel topoloji Şekil 6.6'da gösterilmektedir.

Deneysel SD-VANET topolojisi, WPA2 şifrelemeyi destekleyen 20 araç (araç IP aralığı: 10.0.0.1-10.0.0.20) ve 6 Access Point (AP)'den oluşmuştur. AP'ler, veri düzleminde bulunan araçlara kablosuz bağlantı sağlayan yol kenarlarına yerleştirilmiştir. Araçlar, WAVE olarak da bilinen araç iletişimi için IEEE 802.11p kablosuz standardını simüle eden Akıllı Ulaşım Sistemleri (Intelligent Transportation Systems-ITS) bağlantıları kullanılarak AP'lere bağlanır. Ryu kontrolcü, OpenFlow protokolünü kullanarak AP'ler aracılığıyla veri düzleminde bulunan iletişim cihazları ile bağlantı kurmaktadır. Simülasyon, araçların önceden tanımlanmış bir yol ağındaki hareketini oluşturan SUMO emülatörü ile entegre edilmiştir.

sFlow, ağ trafiğinin gerçek zamanlı olarak izlenmesi için ağa entegre edilmiştir. sFlow, deneysel SD-VANET ağından veri trafiğini toplar. sFlow başlatıldığında, otomatik olarak SD-VANET ağındaki cihazlarda tanımlı olan sFlow agent'a bağlanır ve IP adresleri, örnekleme hızı ve yoklama aralığı dâhil olmak üzere veri trafiğini toplamaya başlar.

Ryu kontrolcü ve Mininet-WIFI aynı sanal makinde sFlow ise farklı bir sanal makinede çalışmaktadır. Mininet-Wifi ve Ryu-SDN, 4G RAM ve 2 çekirdek CPU ile 20-04-1 Ubuntu'ya kurulurken, sFlow 4G RAM ve 1 çekirdek CPU ile 20-04-1 Ubuntu'ya kurulmuştur.

Normal Trafik Senaryosu

Ağı yüklemek ve kapasitesini değerlendirmek için Iperf aracı kullanılmıştır. IPerf, istemci-sunucu modeli kullanan ve sunucu ile istemci düğümleri arasında veri akışları şeklinde trafik oluşturan bir araçtır. Iperf, rastgele trafik üretir, yani bir ana bilgisayar sanal ağlarda ağdaki diğer herhangi bir ana bilgisayara eşit olasılıkla paketler gönderir. Ağı yüklemek için TCP veya UDP akışları oluşturabilir. TCP modunda, bir iperf istemcisi sunucuya bir TCP akışı aracılığıyla sonsuz miktarda veri iletir. Kullanıcı, bir süreden sonra, iperf aracı TCP bağlantısını iptal eder ve ekrana farklı istatistikleri ve doğru şekilde iletmeyi başardığı toplam veri miktarını yazdırır.

Mininet Flow Generator ile IPerf kullanılarak kullanıcılar arasında rastgele akışlar oluşturulmuştur. Her akış için, ağdan bir kullanıcı sunucu olarak bir kullanıcı ise istemci olarak rastgele seçilmektedir.

Rastgele seçilen kullanıcılar rastgele seçilen kullacılara IPerf ile TCP ve UDP bağlantıları kurmaktadır.

ApacheBench, bir web sunucusunu HTTP istekleriyle doldurup gecikme ve başarı ölçümlerini kaydederek performansını http benchmark yaparak ölçer. ApacheBench ile büyük miktarda http isteği atılır. Simülasyondaki bütün kullanıcılarda, python HTTP server ve IPerf server çalıştırılarak TCP ve UDP trafik akışı oluşturulmaktadır.

Ağda normal trafik verisi aktığında, Ryu kontrolcü OpenFlow anahtarlardan "OFPC_FLOW_STATS_REPLY" mesajı gelince akış istatistiklerini lokal dosyaya csv formatinda kaydeder. Kaydedilen özniteliklere trafik sınıfı eklenerek öznitelik sayısı arttırılır. Bu sınıfta normal trafik "0" olarak etiketlenmektedir.

Saldırı Senaryosu

Hping3 paket üreteci ile saldırgan araç 5 ve 4'ten kurban araç 8'e hping3 -a 1.1.1.1 -p 80 -w 200 --rand-source 10.0.0.8 komutu ile TCP flood saldırı gerçekleştirilmiştir. TCP pencere boyutu 200 bayt olarak ayarlanmıştır. Deneysel çalışma kapsamında oluşturulan normal trafik saldırı anında kesilmeden devam ettirilmiştir.

Ağda saldırı trafik verisi aktığında, Ryu kontrolcü OpenFlow anahtarlardan "OFPC_FLOW_STATS_REPLY" mesajı gelince flow istatistiklerini lokal dosyaya csv formatinda kaydedilmektedir. Trafik sınıfında saldırı trafiği "1" olarak etiketlenir.

Veri Kümesi

DDoS saldırıları verilerini ve normal ağ trafiği verilerini toplamak için 10 dakika boyunca deneysel bir simülasyon gerçekleştirilmiştir. Simülasyon sırasında, TCP, UDP paketleri başlangıçta normal paketler olarak ve daha sonra DDoS saldırı paketleri olarak gönderilmiştir. Bu paketlerin her biri 512 baytlık bir yüke sahiptir. Elde edilen veri kümesi 698.767 örnek, 21 öznitelik ve "Normal", "DDoS Saldırıları" olmak üzere iki sınıf içermektedir. Örneklerin 309.602'si normal ağ trafiği akış verisi, 389.165'i ise DDoS saldırı trafiği akış verisidir. Veri kümesindeki tüm öznitelikler SD-VANET mimarisine özgü trafik akışı istatistiklerinden oluşmaktadır. Veri kümesindeki öznitelikler Çizelge 6.3'de sunulmuştur.

Çizelge 6.3. Veri kümesindeki öznitelikler

timestamp	datapath_id	flow_id
ip_src	tp_src	ip_dst
tp_dst	ip_proto	icmp_code
icmp_type	flow_duration_sec	flow_duration_nsec
idle_timeout	hard_timeout	flags
packet_count	byte_count	packet_count_per_second
packet_count_per_nsecond	byte_count_per_second	byte_count_per_nsecond
Sınıf: Trafik sınıfı. Bu çalışmada kullanılan veri kümesinde yer alan veriler etiketlenmiş verilerdir. Bu iki sınıf; "0", "1" şeklindedir. "0" normal trafik akışı anındaki verileri, "1" ise DDoS saldırısı anında alınan verileri temsil eder.		

Normal ağ trafik verileri ve saldırı trafiği verileri kontrolcüde etiketlenerek kaydedildikten sonra, MT-LNet eğitim kodu çalıştırılarak model oluşturulmuştur. Elde edilen verinin tümü eğitim aşamasında kullanılmıştır. Eğitim aşamasından sonra elde edilen model kontrolcüde bulunan tespit modülüne kaydedilmiştir.

6.3.2. Senaryo_2 deneysel sonuçlar

Bu bölümde güvenlik sistemi için önerilen tespit modülüne ilişkin performans sonuçları incelenmektedir. SD-VANET_Guard saldırı tespit sistemi senaryo_2 üzerinden test edilmiş ve sonuçlar özetlenmiştir:

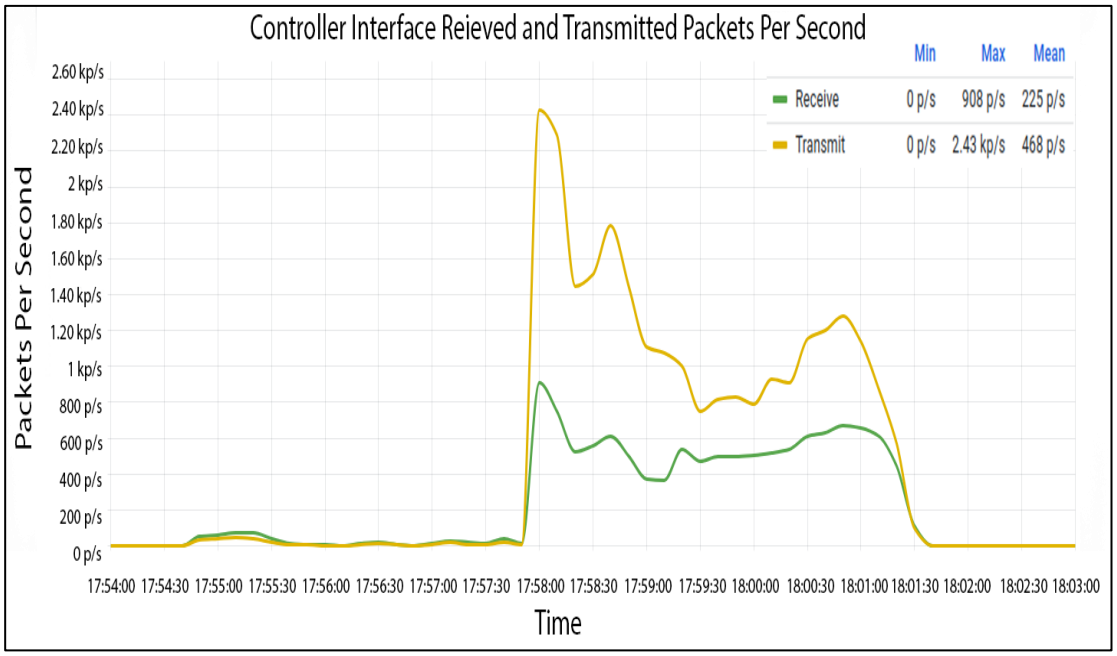
Senaryo_2'de, SD-VANET_Guard saldırı tespit sisteminin DDoS saldırısını tespit etme etkinliğini göstermek için uygulanmıştır. Sistem çalışırken TCP/SYN flood, ve UDP flood içeren karışık DDoS flood saldırıları başlatılmıştır.

Çizelge 6.4. Simülasyonda her bir modülün zaman içindeki eylemleri.

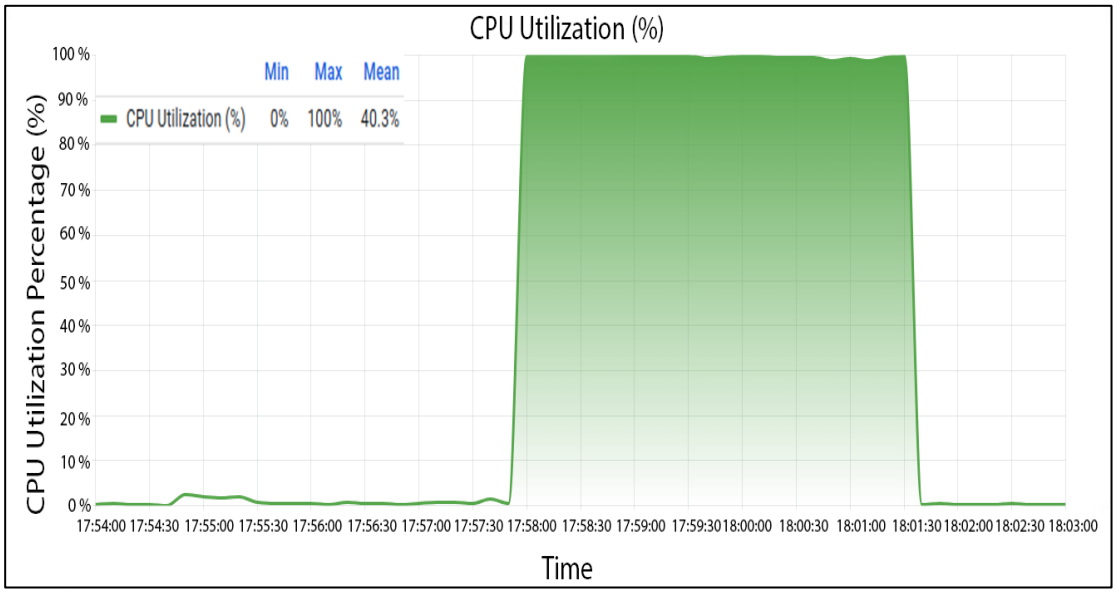
DURUM	ETKİNLİK	ZAMAN
İlk Durum	Uygulamaya başlama	17:54:00
	Ryu kontrolcü (akış istatistiği toplama modülü) anahtarlara akış istatistiği isteği gönderir ve cevaplar local dosyaya yazılır.	17:54:00
Tespit Durumu	Tespit modülünün aktif hale gelmesi/başlatılması. (Her 10 saniyede bir yerel dosyayı oku ve tahmini çalıştır, ardından sonucu influxdb'de yayınla)	17:54:00
	Saldırı tespiti	17:58:00
Akış Oluşturma Durumu	Ryu kontrolcü akış kuralı oluşturma/değiştirme (veri düzleminde bulunan cihazlarından akış isteklerine kontrolcü cevap verir)	17:54:00

Çizelge 6.4’de, senaryo_2'deki her bir modülün başlangıç zamanının ayrıntılı açıklaması verilmiştir. İlk durumda, önceden oluşturulan eğitim veri kümesi kullanılarak eğitililen MT-LNet hibrit modeli çalıştırılmıştır. Bu süreç 17:54:00'de başlatılmıştır. 17:54:00’de saldırı tespit modülü için bir başlatma komutu üretmiştir. 17:54:00'de tespit modülü başlatılmış ve bu modül 17:58:00'de saldırıyı tespit etmiştir. 17:54:00’de Ryu kontrolcü veri düzleminde bulunan iletim cihazlarından gelen akış isteklerine akış kuralı oluşturarak cevap verir. DDoS saldırısı saat 17:57:30’da başlatılmıştır. VANET mimari yapısının hareketli olmasından kaynaklı kontrolcünün veri düzleminde bulunan cihazlarla bazen iletişimi kopmaktadır. Hareketli ağ yapısından dolayı, veri düzleminde bulunan cihazlardan gönderilen akış oluşturma taleplerin kontrolcüye eşiminde gecikmeler yaşanır. Verinin kontrolcüye geç gelmesinden kaynaklı saldırı tespiti 17:58:00’da gerçekleşmiştir.

DDoS saldırısında saldırgan zombiye dönüştürdüğü bilgisayarlar vasıtasıyla, AP'lere araçlarla numarasız istekler gönderir. Gelen paketler AP veya Yol Kenarı Ünitesinde bulunan akış tablosunda yer alan kurallarla eşleşmezse (miss flow), eşleşmeyen tüm girişler için OpenFlow protokolü Yol Kenarı Ünitesi veya AP'ler üzerinden "OFPT PACKET_IN" mesajı ile kontrolöre yeni bir kural yazması için gönderilir. DDoS saldırısı kaynaklı kısa süre içinde veri düzleminde bulunan cihazlar tarafından kontrolcüye büyük miktarda paket gönderilir (Şekil 6.7). Şekilde görüldüğü gibi 17:57:30’da DDoS saldırısının başlamasıyla veri düzleminden büyük miktarda kontrolcüye doğru paket akışı olmuştur. Kontrolcü gelen akış kuralı taleplerine "OFPT PACKET_MOD" mesajıyla ile yanıt verir. Bu durumda kontrolcünün kaynakları (bellek, işlemci, bant genişliği vb.) yetersiz kalır ve ağ çalışmaz hale gelir. Şekil 6.8’de görüldüğü gibi saat 17:57:30 ‘da yeni akış kuralı talepleri arttıkça, işleme ve kontrolcünün iletişim kapasitesi aşırı yüklenir. DDoS saldırısından kaynaklı, CPU kullanımı %100’e yükselmiştir.



Şekil 6.7. Senaryo_2 deneysel SD-VANET topolojisi kontrolcünün arayüzünden alınan ve verilen paket miktarı



Şekil 6.8. Senaryo_2 deneysel SD-VANET topolojisi kontrolcünün işlemci kullanımı

Ağa yeni bir paket geldiğinde, akış tablosunda akış kuralı girişi eşleştirilir. Eşleşmeyen tüm girişler için, anahtar kontrolcüden akış için yeni akış kuralı oluşturmasını talep eder. Anahtar yeni akış kuralı için paketin bir kısmını kontrolcüye gönderir ve paketin diğer bir kısmını ise anahtarda bulunan ara bellekte tutar. Ara bellek dolarsa, paketin tamamını kontrolcüye gönderir. Arabelleğe alınan paketler kontrolcüden gelen “OFPT_PACKET _OUT” veya “OFPT_FLOW_MOD” mesajı aracılığıyla işlenir veya bir süre sonra otomatik olarak düşer.

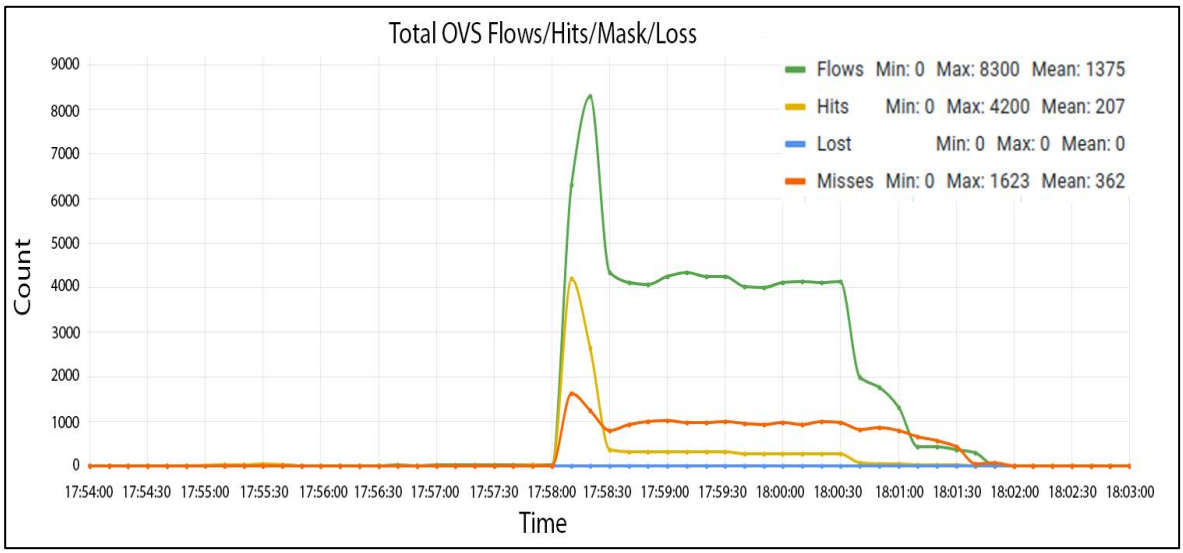
Saldırgan, Yazılım Tanımlı Ağ yapısının bu özelliğinden faydalanarak anahtara sahte akışlar gönderebilir. Anahtar akış tablosu girdileriyle eşleşmeyen akışı her aldığında, kontrolcüye yeni akış kuralı talebinde bulunur. Şekil 6.9'da görüldüğü gibi DDoS saldırısının başladığı 17:57:30 'da kontrolcüye giden yeni akış kuralı isteklerine kısa zamanda kontrolcü büyük miktarda akış kuralı oluşturarak cevap vermiştir. Saldırgan kolayca yeterli sayıda sahte akış üretebilir ve bu da kontrolcünün işlem kapasitesini tüketebilir.

Hits: Kontrolcünün daha önceden oluşturup, OVS anahtarın akış tablosunda tutulan kurallarla eşleşen paket sayısını ifade eder.

Miss: Veri düzleminde bulunan iletim cihazlarına (anahtar, yönlendirici vs.) yeni bir paket geldiğinden akış tablosunda eşleşme aranır. Gelen paket herhangi bir akış girdisiyle eşleşmiyorsa (flow miss) yeni akış kuralı tanımlanması için kontrolcüye gönderilir.

OVS Flow: Datapath içerisindeki flow sayısıdır.

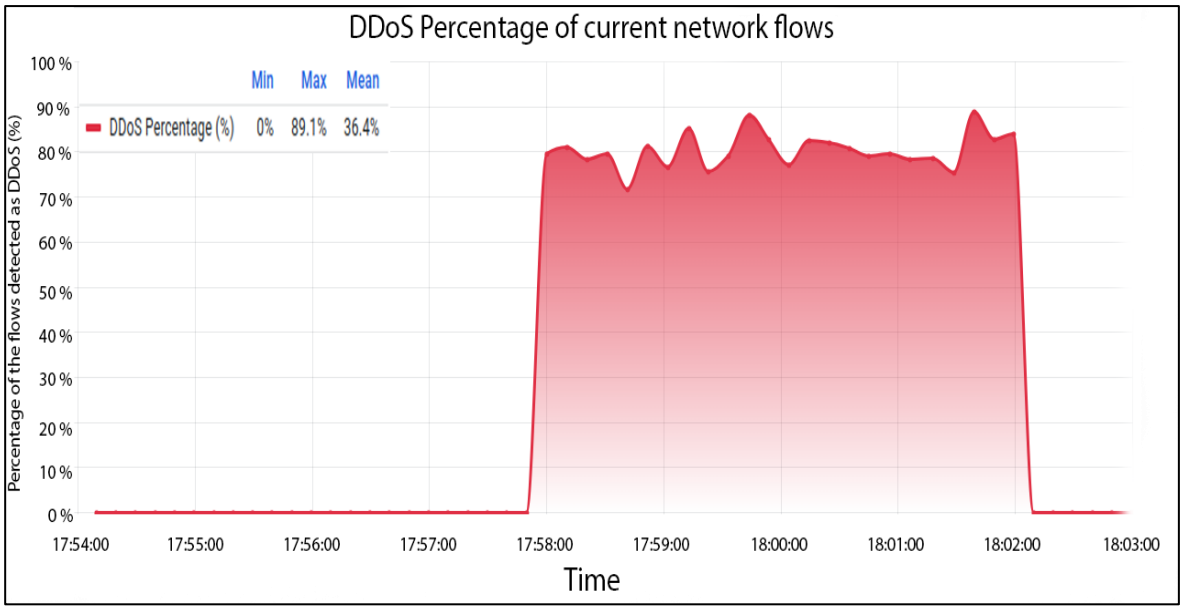
OVS Lost: İşlenmesi için gönderilen ancak drop edilen paket sayısıdır.



Şekil 6.9. Senaryo_2 deneysel SD-VANET topolojisi OVS metrikleri

Senaryo_2 kapsamında Hping3 paket üretici aracı ile oluşturulan DDoS saldırısında saldırgan saniyede 4.55 Mb/s büyük saldırı trafiğini veri düzleminde bulunan cihazla yönlendirir. Gelen paketler için OpenFlow anahtarda bulunan, akış tablosunda bulunan akış girdileriyle eşleşme aranır, eşleşme yok ise (miss flow), paketler "OFPT_PACKET _IN" mesajıyla kontrolcüye

yeni kural yazması için gönderilir. Kısa zaman içinde kontrolcüye büyük miktarda "OFPT_PACKET _IN" mesajı gelir. Gelen kural istekleri için kontrolcü kural yazıp anahtarın akış tablosuna gönderir. Bu mekanizma temel alınarak geliştirilen saldırı tespit mekanizması Şekil 6.10'da görüldüğü gibi 17:57:30 da SD-VANET ağındaki DDoS saldırısı tespit etmiştir. Ağada bulunan mevcut akışların yaklaşık % 90'ı DDoS trafiğinin oluşturduğu akışlardan oluşmaktadır. Tespit mekanizması SD-VANET ağındaki saldırı trafiği ve normal ağ trafiğini ayırt etme özelliğine sahiptir. Senaryo_2 test sonuçlarından anlaşıldığı üzere MT-LNet hibrit modelinin iyi sınıflandırma sonuçlarını elde ettiği ve sunulan SD-VANET_Guard savunma sisteminin tespit modülün de etkili bir DDoS tanımlayıcısı olarak çalıştığı sonucuna varmak mümkündür.



Şekil 6.10. Senaryo_2 deneysel SD-VANET topolojisi mevcut ağ akışındaki DDoS yüzdesi

7. SONUÇ VE ÖNERİLER

Dünya genelinde insan nüfusunun artmasıyla birlikte trafikte kullanılan araç sayısı da sürekli yükselmektedir. Araç sayısındaki artış, trafik kazalarının ve yaralanmaların da artmasına neden olmaktadır. Meydana gelen kazalar göz önünde bulundurulduğunda yol ve yolcu güvenliği büyük bir endişe kaynağıdır. Bu durum, modern dünyanın başa çıkması gereken önemli bir sorun haline gelmiştir. Ayrıca, akıllı şehirlere olan ilgi, akıllı ulaşım sistemlerinde fikirlerin ortaya çıkmasına neden olmuştur. VANET teknolojisinin geliştirilmesi, verilerin araçtan araca ve araçtan yol kenarı bilgi sistemlerine aktarılmasına olanak sağlamıştır. Bu teknoloji, yolculara daha güvenli ve daha konforlu bir sürüş sağlamayı vaat etmektedir.

Ancak VANET teknolojisinin de yaygın olarak kullanılabilmesi için bazı sorunlarının çözülmesi gerekmektedir. Ele alınması gereken VANET konuları; ağ trafiğinin sürekliliğinin sağlanması, çok sayıda araç ve yol bilgi sisteminin oluşturduğu dağınık yapının yönetilmesi, akıllı bilgi sistemlerinden gelen yoğun verilerin işlenmesi ve gerçek zamanlı bilgi akışının sağlanması (yol tıkanıklığı, kaza raporları, mevcut park yerleri, en yakın hastane, benzin istasyonları vb) gibi bazı vektörlerdir. Ayrıca mevcut VANET mimarisi, 5G teknolojisinin talep ettiği park yeri bildirimi, acil durum uyarısı, multimedya iletimi gibi içerik tabanlı bilgi alışverişi için uygun değildir. Yazılım Tanımlı Ağ yaklaşımı, VANET'in mevcut sorunlarına çözüm sunmaya adaydır. Yazılım Tanımlı Ağlar, ağ yönetimini merkezi bir kontrolcü aracılığıyla sağlayan ve geleneksel ağlardan farklı olarak programlanabilir ve esnek bir yapıya sahip olan ağlardır. Yazılım Tanımlı Ağlar, ağ yönetimini daha kolay hale getiren ve ağın performansını artıran birçok avantaj sağlar. Yazılım Tanımlı Ağların VANET ile entegrasyonu kısaca SD-VANET olarak adlandırılmaktadır.

SD-VANET teknolojisi, ağ yönetiminin kolaylaşması, esnek mimari ve ölçeklenebilirlik gibi birçok avantaj sağlamasına rağmen güvenlik konusunda hala bazı endişeler giderilememiştir. Sürücüsüz araçların trafikte kullanımı, akıllı ulaşım servisleri için sürekli yeni protokollerin ve mimarilerin geliştirilmesi gibi yeni teknolojilerin hayatımıza girmesinden önce, sistem verimliliği ve güvenliği etkileyecek siber saldırılara karşı gerekli önlemlerin oluşturulması gerekmektedir. Bu önlemler alınmadan SD-VANET teknolojisinin siber saldırılara maruz kalması durumunda alt yapının zarar görmesi, insan hayatının tehlikeye girmesi gibi birçok sonuçları olabilir.

Özellikle DDoS gibi hacimsel saldırılar SD-VANET performansını ciddi şekilde etkileyen ve hatta ağın tamamen çökmesine neden olabilen önemli bir tehdittir. DDoS saldırıları, ağa aşırı yüklenen ve kaynakları tüketen saldırılar olarak tanımlanır. Bu saldırılar genellikle çok sayıdaki botnetler aracılığıyla gerçekleştirilir. DDoS saldırılarının tespiti ve önlenmesi, ağ güvenliğinin önemli bir parçasıdır. Normalde Yazılım Tanımlı Ağ mimarisine yönelik yüksek yoğunluklu DDoS saldırılarına karşı statik bir eşiğe (threshold) bağlı olarak sınırlı düzeyde bir koruma sağlanabilir. Fakat bu güvenlik düzeyi, kritik işlem verilerini taşıyan hassas bir SD-VANET mimarisi için yeterli değildir. Güvenlik konusunda daha dinamik, etkin ve akıllı çözümler sunacak yaklaşımlara gereksinim duyulmaktadır. Makine öğrenmesi ve derin öğrenme gibi yapay öğrenme modelleri, ağlardaki anomali tespiti ve sınıflandırma gibi birçok alanda başarılı sonuçlar vermiştir. Bu nedenle, DDoS saldırıları için de bu yapay öğrenme modellerinin kullanımı önerilmektedir. DDoS saldırıları genellikle ağ trafiğindeki belirli özelliklerin (örneğin, paket boyutu, kaynak IP adresi, hedef port numarası vb.) anomali oluşumuna neden olur. Bu anomali durumları, yapay öğrenme modelleri ile tespit edilebilir. Bu modeller, ağ trafiğindeki bu özelliklerin normal ve anormal davranışını öğrenerek sınıflandırabilirler ve DDoS saldırılarını tespit edebilirler. SD-VANET mimarisinde akan trafik verilerinin, yapay öğrenme yöntemleri kullanılarak eğitilmesine dayalı, akıllı davranışlar sergileyen modellerin oluşturulması DDoS gibi siber saldırıların daha etkili tespitini sağlayabilmektedir. Yapay öğrenmeye dayalı bu modeller, ağ yöneticilerine saldırıların gerçek zamanlı tespiti ve hızlı yanıt verme imkânı sağlar.

Bu çalışmada, SD-VANET mimarisine yönelik gerçekleştirilen, DDoS saldırılarının etkin bir şekilde tespiti için yapay öğrenmeye dayalı en iyi performansı sergileyecek modeli elde etme amacıyla çeşitli deneysel çalışmalar gerçekleştirilmiştir. Bu deneysel çalışmalar sonucunda SD-VANET mimarisi içerisinde yeralan kontrolcüyü hedefleyen DDoS saldırılarının tespiti için SD-VANET_Guard olarak adlandırdığımız bir güvenlik sistemi önerilmiştir. Bu sistem, Ryu kontrolcü ile veri toplama, Çok Görevli Öğrenme Ağı (Multi-Task Learning Network -MT-LNet) olarak adlandırılan hibrit model ile veri işleme ve ağ trafik sınıflandırmasını içermektedir. MT-LNet, birden fazla görevi aynı anda çözmek için kullanılan bir derin öğrenme ağıdır. Bu ağ, farklı ama ilişkili görevleri aynı zamanda çözmek için tasarlanmıştır ve farklı veri kümeleri üzerinde eğitilebilir. MT-LNet, farklı görevleri tek bir modelde birleştirerek, verimli bir öğrenme ve tahminleme süreci sağlar. Bu model, farklı görevler arasındaki ortaklıkları kullanarak, bir görevdeki öğrenilen bilginin diğer görevlerde de kullanılmasını sağlar. Bu da, her görev için ayrı ayrı model eğitmek yerine, tüm görevlerin aynı modele dahil

edilerek eğitilmesi anlamına gelir. MT-LNet, her bir görev için ayrı bir çıkış katmanı kullanarak farklı görevlerin eş zamanlı çözümünü sağlar. Bu katmanlar, her bir görev için özelleştirilmiştir ve ayrı ayrı eğitilir. Ancak, ağın girdi katmanları ve ara katmanları tüm görevler için ortak olarak kullanılır. Bu ortak katmanlar, farklı görevler arasındaki ortak özellikleri yakalayarak, verimli bir öğrenme süreci sağlar.

SD-VANET_Guard güvenlik sisteminde, MT-LNet hibrit modeli yazılım tanımlı ağ kontrolcüsünün içine gömülmüştür. MT-LNet modeli normal ağ trafiği tarafından oluşturulan legal akış girişlerini ve DDoS saldırı trafiği tarafından oluşturulan illegal akış girişlerini tanıyabilmektedir. SD-VANET_Guard saldırı tespit sisteminin performansı iki faklı senaryo oluşturularak test edilmiştir. İlk senaryoda Yazılım Tanımlı Ağ tabanlı kablolu bir ağ topolojisi oluşturularak modelin performansı test edilmiştir. İkinci senaryoda ise kablosuz bir ağ topolojisi oluşturularak SD-VANET mimarisi temelinde modelin performansı test edilmiştir. Her iki senaryoda da SD-VANET_Guard saldırı tespit sistemi DDoS saldırı trafiğinin oluşturduğu akışları tespit etmede oldukça yüksek başarı sağlamıştır. Elde edilen sonuçlar neticesinde SD-VANET_Guard saldırı tespit sisteminin, SD-VANET mimarisi üzerinde etkili bir DDoS tespit sistemi olarak çalıştığı gözlenmiştir.

Bu çalışmada mevcutta tek bir Yazılım Tanımlı Ağ kontrolcüsü kullanılmıştır. Bu çalışmanın ilerleyen safhaları için birden fazla kontrolcü kullanılarak deneyler gerçekleştirilebilir. SD-VANET gibi hareketli ağ yapılarında çoklu kontrolcü kullanımı ile saldırı anında kontrolcünün gerçek zamanlı tepkime hızı arttırılarak saldırılara karşı daha etkin bir şekilde cevap verilebileceği düşünülmektedir. Çoklu kontrolcü mimarisi, ağın daha büyük ölçeklerde çalışmasını sağlar. Kontrolcüler arasında görevleri ve kaynakları paylaştırarak, ağ yönetimindeki yükü paylaştırır. Çoklu kontrolcü mimarisi, bir kontrolcünün çökmesi durumunda diğer kontrolcülerin yönetimi devralmasına izin verir. Bu sayede ağın kesintiye uğraması önlenir ve ağın sürekli olarak kullanılabilir olması sağlanır. Ayrıca çoklu kontrolcü mimarisi, ağın farklı bölümlerinin farklı kontrolcüler tarafından yönetilmesine olanak tanır. Bu sayede, ağın farklı kısımlarına özelleştirilmiş çözümler uygulayarak esneklik sağlanır. Özellikle Yazılım tanımlı kablosuz ağ yapılarında hiyerarşik yapıda çoklu kontrolcünün kullanımı ve kontrolcünün ağdaki konumu, kontrolcünün veri düzleminde bulunan ağ elemanları üzerindeki denetimini arttırarak, kaynak kullanımını optimize ederek gecikmeyi azaltır.

KAYNAKLAR

1. Balta, M., Özçelik, İ. (2020). A 3-stage fuzzy-decision tree model for traffic signal optimization in urban city via a SDN based VANET architecture. *Future Generation Computer Systems*, 104, 142-158.

2. Bhatia, J., Dave, R., Bhayani, H., Tanwar, S. and Nayyar, A. (2020). SDN-based real-time urban traffic analysis in VANET environment. *Computer Communications, 149*, 162-175.

3. Boucetta, S. I., Johanyák, C. Z. and Pokorádi, L. K. (2017). Survey on software defined vanets. *Gradus*, 4(1), 272-283.

4. Jiacheng, C., Haibo, Z. H. O. U., Ning, Z., Peng, Y., Lin, G. and Xuemin, S. (2016). Software defined Internet of vehicles: Architecture, challenges and solutions. *Journal of Communications and İnformation Networks*, 1(1), 14-26.

5. Jaballah, W. B., Conti, M. and Lal, C. (2019). A survey on software-defined VANETs: benefits, challenges, and future directions. *arXiv preprint arXiv:1904.04577*.

6. Singh, J., Behal, S. (2020). Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions. *Computer Science Review*, 37, 100279.

7. Wang, L., Liu, Y. (2020, June). *A DDoS attack detection method based on information entropy and deep learning in SDN.* In 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 1, 1084-1088, Chongqing, China.

8. Janarthanam, S., Prakash, N. and Shanthakumar, M. (2020). Adaptive learning method for DDoS attacks on software defined network function virtualization. *EAI Endorsed Transactions on Cloud Systems*, 6(18), e6-e6.

9. Anwer, M. S., Guy, C. (2014). A survey of VANET technologies. *Journal of Emerging Trends in Computing and Information Sciences*, 5(9), 661-671.

10. Chahal, M., Harit, S., Mishra, K. K., Sangaiah, A. K. and Zheng, Z. (2017). A survey on software-defined networking in vehicular ad hoc networks: Challenges, applications and use cases. *Sustainable Cities and Society*, 35, 830-840.

11. Sheikh, M. S., & Liang, J. (2019). A comprehensive survey on VANET security services in traffic management system. *Wireless Communications and Mobile Computing*, 1, 1-23.

12. Lozano Dominguez, J. M. and Mateo Sanguino, T. J. (2019). Review on V2X, I2X, and P2X communications and their applications: a comprehensive analysis over time. *Sensors*, 19(12), 2756.

13. Wazirali, R., Ahmad, R. and Alhiyari, S. (2021). SDN-openflow topology discovery: an overview of performance issues. *Applied Sciences*, 11(15), 6999.

14. Nunes, B. A. A., Mendonca, M., Nguyen, X. N., Obraczka, K. and Turletti, T. (2014). A survey of software-defined networking: Past, present, and future of programmable networks. *Communications Surveys & Tutorials*, 16(3), 1617-1634.

15. Sultana, R., Grover, J. and Tripathi, M. (2021). Security of SDN-based vehicular ad hoc networks: State-of-the-art and challenges. *Vehicular Communications*, 27, 100284.

16. Hu, F., Hao, Q. and Bao, K. (2014). A survey on software-defined network and openflow: From concept to implementation. *IEEE Communications Surveys & Tutorials*, 16(4), 2181-2206.

17. Latif, Z., Sharif, K., Li, F., Karim, M. M., Biswas, S. and Wang, Y. (2020). A comprehensive survey of interface protocols for software defined networks. *Journal of Network and Computer Applications*, 156, 102563.

18. Alrehan, A. M., & Alhaidari, F. A. (2019, May). *Machine learning techniques to detect DDoS attacks on VANET system: a survey.* In 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), 1-6, Riyadh, Saudi Arabia.

19. Shafiq, H., Rehman, R. A. and Kim, B. S. (2018). Services and security threats in sdn based vanets: A survey. *Wireless Communications and Mobile Computing*, 2018.

20. Jaballah, W. B., Conti, M. and Lal, C. (2020). Security and design requirements for software-defined VANETs. *Computer Networks*, 169, 107099.

21. Islam, M. M., Khan, M. T. R., Saad, M. M. and Kim, D. (2021). Software-defined vehicular network (SDVN): A survey on architecture and routing. *Journal of Systems Architecture*, 114, 101961.

22. Xiao, X., Kui, X. (2015). The characterizes of communication contacts between vehicles and intersections for software-defined vehicular networks. *Mobile Networks and Applications*, 20(1), 98-104.

23. Abbas, M. T., Muhammad, A. and Song, W. C. (2020). SD-IoV: SDN enabled routing for internet of vehicles in road-aware approach. *Journal of Ambient Intelligence and Humanized Computing*, 11, 1265-1280.

24. Hussain, R., Lee, J. and Zeadally, S. (2020). Trust in VANET: A survey of current solutions and future research opportunities.*Transactions on Intelligent Transportation Systems*, 22(5), 2553-2571.

25. Cui, Y., Yan, L., Li, S., Xing, H., Pan, W., Zhu, J. and Zheng, X. (2016). SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks. *Journal of Network and Computer Applications*, 68, 65–79.

26. Gondim, J. J., de Oliveira Albuquerque, R. and Orozco, A. L. S. (2020). Mirror saturation in amplified reflection Distributed Denial of Service: A case of study using SNMP, SSDP, NTP and DNS protocols. *Future Generation Computer Systems*, 108, 68-81.

27. Marvi, M., Arfeen, A. and Uddin, R. (2021). A generalized machine learning-based model for the detection of DDoS attacks. *International Journal of Network Management*, 31(6), e2152.

28. Najafimehr, M., Zarifzadeh, S. and Mostafavi, S. (2022). A hybrid machine learning approach for detecting unprecedented DDoS attacks. *The Journal of Supercomputing*, 78(6), 8106-8136.

29. Germano Da Silva, E., Dias Knob, L. A., Wickboldt, J. A., Gaspary, L. P., Granville, L. Z. and Schaeffer-Filho, A. (2015). *Capitalizing on SDN-based SCADA systems: An anti-eavesdropping case-study*. In Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, Ottawa, Canada.

30. Xia, W., Wen, Y., Foh, C. H., Niyato, D. and Xie, H. (2015). A Survey on Software-Defined Networking. *IEEE Communications Surveys and Tutorials*, 17(1), 27–51.

31. Dabbagh, M., Hamdaoui, B., Guizani, M., Rayes, A. (2015). Software-defined networking security: pros and cons. *Communications Magazine*, 53(6), 73-79.

32. Alshamrani, A., Chowdhary, A., Pisharody, S., Lu, D. and Huang, D. (2017, November). *A defense system for defeating DDoS attacks in SDN based networks.* In Proceedings of the 15th ACM international symposium on mobility management and wireless access, 83-92, Miami.

33. Latah, M., Toker, L. (2018). A novel intelligent approach for detecting DoS flooding attacks in software-defined networks. *International Journal of Advances in Intelligent Informatics*, 4(1), 11-20.

34. Ye, J., Cheng, X., Zhu, J., Feng, L. and Song, L. (2018). A DDoS attack detection method based on SVM in software defined network. *Security and Communication Networks*, 2018.

35. Nanda, S., Zafari, F., DeCusatis, C., Wedaa, E. and Yang, B. (2016, November). *Predicting network attack patterns in SDN using machine learning approach.* In 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks, Palo Alto, CA, USA.

36. Jankowski, D., Amanowicz, M. (2016). On efficiency of selected machine learning algorithms for intrusion detection in software defined networks. *International Journal of Electronics and Telecommunications*, 62(3), 247-252.

37. Mowla, N. I., Doh, I. and Chae, K. (2018). CSDSM: Cognitive switch-based DDoS sensing and mitigation in SDN-driven CDNi word. *Computer Science and Information Systems*, 15(1), 163-185.

38. Yu, Y., Guo, L., Liu, Y., Zheng, J. and Zong, Y. U. E. (2018). An efficient SDN-based DDoS attack detection and rapid response platform in vehicular networks. *IEEE Access*, 6, 44570-44579.

39. Li, C., Wu, Y., Yuan, X., Sun, Z., Wang, W., Li, X. and Gong, L. (2018). Detection and defense of DDoS attack–based on deep learning in OpenFlow-based SDN. *International Journal of Communication Systems*, 31(5), e3497.

40. Latah, M., Toker, L. (2018). A novel intelligent approach for detecting DoS flooding attacks in software-defined networks. *International Journal of Advances in Intelligent Informatics*, 4(1), 11-20

41. Mowla, N. I., Doh, I. and Chae, K. (2018). CSDSM: Cognitive switch-based DDoS sensing and mitigation in SDN-driven CDNi word. *Computer Science and Information Systems*, 15(1), 163-185.

42. Niyaz, Q., Sun, W. and Javaid, A. Y. (2016). A deep learning based DDoS detection system in software-defined networking (SDN). *arXiv preprint arXiv:1611.07400*.

43. Tang, T. A., Mhamdi, L., McLernon, D., Sar, Z. and Ghogho, M. (2016, October). Deep learning approach for network intrusion detection in software defined networking. In *2016 international conference on wireless networks and mobile communications (WINCOM)* (pp. 258-263). IEEE.

44. Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R. and Iqbal, J. (2020). A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks. *Ieee Access*, 8, 53972-53983.

45. Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., Ghogho, M. and El Moussa, F. (2020). DeepIDS: Deep learning approach for intrusion detection in software defined networking. *Electronics*, 9(9), 1533.

46. Dey, S. K., Rahman, M. M. (2019). Effects of machine learning approach in flow-based anomaly detection on software-defined networking. *Symmetry*, 12(1), 7.

47. Ko, I., Chambers, D. and Barrett, E. (2020). Adaptable feature-selecting and threshold-moving complete autoencoder for DDoS flood attack mitigation. *Journal of Information Security and Applications*, 55, 102647.

48. Ujjan, R. M. A., Pervez, Z., Dahal, K., Bashir, A. K., Mumtaz, R. and González, J. (2020). Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. *Future Generation Computer Systems*, 111, 763-779.

49. Ravi, N., Shalinie, S. M. (2020). Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture. *IEEE Internet of Things Journal*, 7(4), 3559-3570.

50. Ujjan, R. M. A., Pervez, Z., Dahal, K., Khan, W. A., Khattak, A. M. and Hayat, B. (2021). Entropy based features distribution for anti-ddos model in sdn. *Sustainability*, 13(3), 1522.

51. de Assis, M. V., Carvalho, L. F., Rodrigues, J. J., Lloret, J. and Proença Jr, M. L. (2020). Near real-time security system applied to SDN environments in IoT networks using convolutional neural network. *Computers & Electrical Engineering*, 86, 106738.

52. Satheesh, N., Rathnamma, M. V., Rajeshkumar, G., Sagar, P. V., Dadheech, P., Dogiwal, S. R. and Sengan, S. (2020). Flow-based anomaly intrusion detection using machine learning model with software defined networking for OpenFlow network. *Microprocessors and Microsystems*, 79, 103285.

53. Cil, A. E., Yildiz, K. and Buldu, A. (2021). Detection of DDoS attacks with feed forward based deep neural network model. *Expert Systems with Applications*, 169, 114520.

54. Myint Oo, M., Kamolphiwong, S., Kamolphiwong, T. and Vasupongayya, S. (2019). Advanced support vector machine-(ASVM-) based detection for distributed denial of service (DDoS) attack on software defined networking (SDN). *Journal of Computer Networks and Communications*, 2019.

55. Santos, R., Souza, D., Santo, W., Ribeiro, A. and Moreno, E. (2020). Machine learning algorithms to detect DDoS attacks in SDN. *Concurrency and Computation: Practice and Experience*, 32(16), e5402.

56. Latah, M., Toker, L. (2020). An efficient flow-based multi-level hybrid intrusion detection system for software-defined networks. *CCF Transactions on Networking*, 3(3-4), 261-271.

57. Sahoo, K. S., Tripathy, B. K., Naik, K., Ramasubbareddy, S., Balusamy, B., Khari, M. and Burgos, D. (2020). An evolutionary SVM model for DDOS attack detection in software defined networks. *IEEE Access*, 8, 132502-132513.

58. Ujjan, R. M. A., Pervez, Z., Dahal, K., Khan, W. A., Khattak, A. M. and Hayat, B. (2021). Entropy based features distribution for anti-ddos model in sdn. *Sustainability*, 13(3), 1522.

59. Assis, M. V., Carvalho, L. F., Lloret, J., & Proença Jr, M. L. (2021). A GRU deep learning system against attacks in software defined networks. *Journal of Network and Computer Applications*, 177, 102942.

60. Gadze, J. D., Bamfo-Asante, A. A., Agyemang, J. O., Nunoo-Mensah, H. and Opare, K. A. B. (2021). An investigation into the application of deep learning in the detection and mitigation of DDOS attack on SDN controllers. *Technologies*, 9(1), 14.

61. Polat, H., Polat, O. and Cetin, A. (2020). Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models. *Sustainability*, 12(3), 1035.

62. Hira, Z. M., Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics*, 2015.

63. Venkatesh, B., Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and İnformation Technologies*, 19(1), 3-26.

64. Hira, Z. M., Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in bioinformatics, 2015*.

65. Venkatesh, B., Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and İnformation Technologies*, 19(1), 3-26

66. Hira, Z. M., Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics*, 2015.

67. Hu, J., Peng, H., Wang, J. and Yu, W. (2020). kNN-P: A kNN classifier optimized by P systems. *Theoretical Computer Science*, 817, 55-65.

68. Jankowski, D., Amanowicz, M. (2016). On efficiency of selected machine learning algorithms for intrusion detection in software defined networks. *International Journal of Electronics and Telecommunications*, 62(3), 247-252.

69. Ye, J., Cheng, X., Zhu, J., Feng, L. and Song, L. (2018). A DDoS attack detection method based on SVM in software defined network. *Security and Communication Networks*, 2018.

70. Nitze, I., Schulthess, U. and Asche, H. (2012). *Comparison of machine learning algorithms random forest, artificial neural network and support vector machine to maximum likelihood for supervised crop type classification.* Proceedings of the 4th GEOBIA, Rio de Janeiro, Brazil, 79, 3540.

71. Somol, P., Novovicová, J., Pudil, P. and CZ37701, J. H. (2010, March). *Improving sequential feature selection methods performance by means of hybridization.* In Proc. 6th IASTED Int. Conf. on Advances in Computer Science and Engrg. ACTA Press, 2010.

72. Polat, H., Turkoglu, M. and Polat, O. (2020). Deep network approach with stacked sparse autoencoders in detection of DDoS attacks on SDN-based VANET. *IET Communications*, 14(22), 4089-4100.

73. Le, Q. V. (2015). A tutorial on deep learning part 2: Autoencoders, convolutional neural networks and recurrent neural networks. *Google Brain*, *20*, 1-20.

74. Le, Q. V., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B. and Ng, A. Y. (2011, June). *On optimization methods for deep learning.* In Proceedings of the 28th International Conference on International Conference on Machine Learning, 265-272, Washington, USA.

75. Lv, F., Han, M. and Qiu, T. (2017). Remote sensing image classification based on ensemble extreme learning machine with stacked autoencoder. *IEEE Access*, 5, 9021-9031.

76. Ali, A., Yangyu, F. and Liu, S. (2017). Automatic modulation classification of digital modulation signals with stacked autoencoders. *Digital Signal Processing*, 71, 108-116.

77. Zia ur Rehman, M., Gilani, S. O., Waris, A., Niazi, I. K., Slabaugh, G., Farina, D. and Kamavuako, E. N. (2018). Stacked sparse autoencoders for EMG-based classification of hand motions: A comparative multi day analyses between surface and intramuscular EMG. *Applied Sciences*, 8(7), 1126.

78. Bengio, Y., Lamblin, P., Popovici, D. and Larochelle, H. (2006). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19.

79. Zia ur Rehman, M., Gilani, S. O., Waris, A., Niazi, I. K., Slabaugh, G., Farina, D. and Kamavuako, E. N. (2018). Stacked sparse autoencoders for EMG-based classification of hand motions: A comparative multi day analyses between surface and intramuscular EMG. *Applied Sciences*, 8(7), 1126.

80. Ibrahim, M. F. I., Al-Jumaily, A. A. (2018). Auto-encoder based deep learning for surface electromyography signal processing. *Advances in Science, Technology and Engineering Systems*, 3(1), 94 - 102

81. Ng, A., Ngiam, J., Foo, C. Y., Mai, Y. and Suen, C. (2012). UFLDL tutorial. *Chapters available at* http://deeplearningstanford. edu/wiki/index. php/UFLDL_Tutorial.

82. Caliskan, A., Badem, H., Basturk, A. and Yuksel, M. E. (2016, December). *A comparative study on classification by deep learning.* In 2016 National Conference on Electrical, Electronics and Biomedical Engineering (ELECO), 503-506, Bursa, Turkey.

83. Kannadasan, K., Edla, D. R. and Kuppili, V. (2019). Type 2 diabetes data classification using stacked autoencoders in deep neural networks. *Clinical Epidemiology and Global Health*, 7(4), 530-535.

84. Türkoğlu, M., Polat, H., Koçak, C. and Polat, O. (2022). Recognition of DDoS attacks on SD-VANET based on combination of hyperparameter optimization and feature selection. *Expert Systems with Applications*, 203, 117500.

85. Peng, H., Long, F. and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine İntelligence*, 27(8), 1226-1238.

86. Sakar, C. O., Kursun, O. and Gurgen, F. (2012). A feature selection method based on kernel canonical correlation analysis and the minimum Redundancy–Maximum Relevance filter method. *Expert Systems with Applications*, 39(3), 3432-3437.

87. Ramírez-Gallego, S., Lastra, I., Martínez-Rego, D., Bolón-Canedo, V., Benítez, J. M., Herrera, F. and Alonso-Betanzos, A. (2017). Fast-mRMR: Fast minimum redundancy maximum relevance algorithm for high-dimensional big data. *International Journal of Intelligent Systems*, 32(2), 134-152.

88. Bugata, P., Drotar, P. (2020). On some aspects of minimum redundancy maximum relevance feature selection. *Science China Information Sciences*, 63, 1-15.

89. Xie, J., Yu, F. R., Huang, T., Xie, R., Liu, J., Wang, C. and Liu, Y. (2018). A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(1), 393-430.

90. Bergstra, J., Bardenet, R., Bengio, Y. and Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24

91. Frazier, P. I. (2018). A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

92. Blanchard, A., Sapsis, T. (2021). Bayesian optimization with output-weighted optimal sampling. *Journal of Computational Physics*, 425, 109901.

Gazili olmak ayrıcalıktır