
GAZİ ÜNİVERSİTESİ
Gazi
1926

YAZILIM TEMELLİ AĞLAR AÇISINDAN DAĞITILMIŞ HİZMET REDDİ DOĞRULTUSUNDAKİ SALDIRILARA YÖNELİK VERİ KATMANI TABANLI SİSTEM VE GERÇEK ZAMANLI SALDIRILARIN ÖNLENMESİ SİSTEMİ

Kenan ERDOĞAN

DOKTORA TEZİ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OCAK 2026

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.

Kenan ERDOĞAN
16/01/2026

YAZILIM TEMELLİ AĞLAR AÇISINDAN DAĞITILMIŞ HİZMET REDDİ DOĞRULTUSUNDAKİ SALDIRILARA YÖNELİK VERİ KATMANI TABANLI SİSTEM VE GERÇEK ZAMANLI SALDIRILARIN ÖNLENMESİ SİSTEMİ

(Doktora Tezi)

Kenan ERDOĞAN

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Ocak 2026

ÖZET

Yazılım temelli ağ mimarisi günümüzde yeni ortaya çıkan ve gelecek vadeden bir alan olarak görülmektedir. Bu yeni alan bilişim sektörü içerisinde birçok yazılım temelli problem için olası çözüm önerisi geliştirmekte ve uygulamaya sunmaktadır. Tüm avantajlarına rağmen yazılım temelli ağlar halen daha Dağıtılmış Hizmet Reddi saldırılarına karşı güçlü bir savunma sistemi geliştirememiştir. Saldırılar, yazılım temelli ağlarda yıkıcı etkiye sahip olmakla birlikte sistemde bütüncül bir kayıp yaşanmasına neden olarak kullanıcıyı zor duruma düşürmektedir. Bu tez çalışmamızda sistemin bu açıklığına yönelik olası bir çözüm modül örneği sunulmuştur. Bu modül yardımı ile olası saldırıların tespit edilip önlenebilmesi için çalışmalar gerçekleştirilmiştir. Veri katmanlı bir sistem izlemesi bağlamında oluşturduğumuz çözüm önerisinin yazılım temelli ağlarda Dağıtılmış Hizmet Reddi saldırılarına karşı bir savunma sistemi oluşturması hedeflenmiştir. Çalışma içerisinde oluşturulan güvenlik sistemine belirli testler uygulanarak güvenilirlik düzeyi ölçülmüş ve veriler ortaya koyulmuştur. Öte yandan çalışmamız bütüncül bir biçimde ele alındığında gerçek zamanlı saldırıların önlenebilmesi içinde bir savunma mekanizması örneği ortaya koymaktadır. Özellikle 5G teknolojisinin yaygın hale gelmesi sonrasında yazılım temelli ağlar için güvenlik sorunlarında artış meydana gelmiş ve bu noktada belirli savunma sistemlerinin geliştirilmesi zorunlu bir gereklilik halini almıştır. Çalışmamız Dağıtılmış Hizmet Reddi saldırıları ile birlikte Gerçek Zamanlı Saldırılar içinde bir savunma sistemi önermektedir. Her iki saldırı biçimi de yazılım temelli ağlar için risk oluşturması bağlamında dikkat edilmesi ve kullanıcının korunması gereken noktalar olarak öne çıkmaktadır.

Bilim Kodu : 93446
Anahtar Kelimeler : Dağıtılmış hizmet reddi saldırıları, 5G, yazılım temelli ağlar, savunma sistemleri
Sayfa Adedi : 139
Danışman : Prof. Dr. Nursel AKÇAM

DATA-PLANE-BASED SYSTEM AND REAL-TIME ATTACK PREVENTION FOR SOFTWARE-DEFINED NETWORKS DISTRIBUTED DENIAL-OF-SERVICE ATTACKS

(Ph. D. Thesis)

Kenan ERDOĞAN

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

January 2026

ABSTRACT

Nowadays software-based network architecture is seen as an emerging and promising field. This new field develops and implements possible solutions for many software-based problems in the IT sector. Despite all their advantages, software-based networks still have not developed a strong defense system against Distributed Denial of Service attacks. Although the attacks have a devastating effect on software-based networks, they cause a holistic loss in the system and put the user in a difficult situation. In this thesis study, a possible solution module example for this openness of the system is presented. With the help of this module, studies have been carried out to detect and prevent possible attacks. The solution proposal we created in the context of a data-layered system monitoring is aimed to create a defense system against Distributed Denial of Service attacks in software-based networks. By applying certain tests to the security system created in the study, the level of reliability was measured and the data were revealed. On the other hand, when our study is considered holistically, it presents an example of a defense mechanism to prevent real-time attacks. Especially after the widespread use of 5G technology, there has been an increase in security problems for software-based networks, and at this point, the development of certain defense systems has become a mandatory requirement. Our study proposes a defense system in Real-Time Attacks along with Distributed Denial of Service attacks. Both types of attacks stand out as points that need attention and protection of the user in the context of posing risks for software-based networks.

Science Code	:	93446
Key Words	:	Distributed Denial of service attacks, 5G, software based networks, defense systems
Page Number	:	139
Supervisor	:	Prof. Dr. Nursel AKÇAM

TEŞEKKÜR

Çalışmalarım boyunca verdiği desteklerden dolayı danışmanım Prof. Dr. Nursel AKÇAM hocama yine ve destekleriyle beni hiçbir zaman yalnız bırakmayan aileme teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET iv

ABSTRACT v

TEŞEKKÜR vi

İÇİNDEKİLER vii

ÇİZELGELERİN LİSTESİ ix

ŞEKİLLERİN LİSTESİ xi

SİMGELER VE KISALTMALAR xiii

1. GİRİŞ 1

2. YAZILIM TEMELLİ AĞLAR VE DAĞITILMIŞ HİZMET REDDİ SALDIRILARI KAPSAMINDA VERİ KATMANI TABANLI SİSTEM 5

2.1. Genel Bakış 5

2.1.1. Yazılım temelli ağlar (SDN) 5

2.1.2. Dağıtılmış hizmet reddi (DDoS) 8

2.1.3. Yazılım temelli ağlar ve dağıtılmış hizmet reddi arasındaki ilişkiler 10

2.1.4. Kontrol katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri 12

2.1.5. Hibrit katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri 14

2.1.6. Veri katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri 15

2.2. Sistem Tasarımı 16

2.2.1. Mimariye genel bakış 16

2.2.2. Sistem ihtiyaçları 19

2.2.3. Sistem yapısı 22

2.3. Tespit Algoritmaları 29

2.3.1. MiddleModule sistemi açısından tespit algoritmaları 30

2.3.2. Protokol istismar saldırısı tespit algoritmaları 33

Sayfa

2.3.3. IP Kandırma saldırısı tespit algoritmaları 35
2.3.4. Flood Saldırısı tespit algoritmaları 36
2.3.5. Amplifikasyon saldırı tespit algoritmaları 39
2.3.6. Yansıma saldırısı tespit algoritmaları 41
2.4. Değerlendirme 44
2.4.1. Değerlendirme araçları 45
2.4.2. Değerlendirme testleri ve test sonuçları 50
2.4.3. Karşılaştırmalar ve genel tartışmalar 79
3. GERÇEK ZAMANLI SALDIRILARIN ÖNLENMESİ SİSTEMİ 87
3.1. Genel Bakış 87
3.1.1. 5G ağları kapsamında yazılım temelli ağlar 87
3.1.2. Random Forest modeli 92
3.2. İlgili Çalışmalar 95
3.3. Önerilen Güvenlik Yaklaşımları 99
3.4. Yazılım Temelli Ağlar Veri Kümeleri 102
3.4.1. Test ortamına genel bakış 103
3.4.2. Yararlı trafik 104
3.4.3. Kötü amaçlı trafik 105
3.4.4. Akış toplama ve özellik oluşturma 109
3.4.5. Model değerlendirmesi 113
3.5. Güncel Çalışmalar 124
4. SONUÇ VE ÖNERİLER 127
KAYNAKLAR 133
ÖZGEÇMİŞ 139

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge 2.1. SYN flood tespit algoritması 34

Çizelge 2.2. IP Kandırma tespit algoritması 36

Çizelge 2.3. Flood saldırısı tespit algoritması 38

Çizelge 2.4. Amplifikasyon saldırı tespit algoritması 40

Çizelge 2.5. Yansıma saldırısı tespit algoritması 42

Çizelge 2.6. Bir savunma sisteminin olası sonuçları 47

Çizelge 2.7. Savunma gücü performans metrikleri 48

Çizelge 2.8. Trafik oluşturmaya yönelik modlar 51

Çizelge 2.9. OMNET++ açısından test aşamaları 53

Çizelge 2.10. OMNET++ kapsamında sistem performansı düşüşü sonucu 55

Çizelge 2.11. Simülasyon savunma gücü performans sonuçları 61

Çizelge 2.12. Yapılandırma değişkeni varyasyonu 63

Çizelge 2.13. Test senaryosu için optimum yapılandırma değişkenleri 65

Çizelge 2.14. VirtualBox özellikleri 66

Çizelge 2.15. MİNİNET kapsamında test aşamaları 66

Çizelge 2.16. MİNİNET üzerinde sistem performansı düşüşü sonucu 68

Çizelge 2.17. Temel test senaryosu ile tespit performansı sonuçları 72

Çizelge 2.18. MİNİNET kapsamında savunma gücü testi sonuçları 74

Çizelge 2.19. İdeal koşullar altında savunma gücü karşılaştırması 80

Çizelge 2.20. Paket alma zaman gecikmesi karşılaştırması 81

Çizelge 2.21. Trafik oluşturma oranları karşılaştırması 82

Çizelge 2.22. Kontrol uçağı temelli sistem ve hibrit sistem ile savunma gücü karşılaştırması 84

Çizelge 3.1. Uçtan uca işlem adımları 101

Çizelge 3.2. Yazılım temelli ağ veri setlerinin özellikleri 109

Çizelge **Sayfa**

Çizelge 3.3. İlk yazılım temelli ağ veri kümesindeki kayıtların dağılımı..... 111

Çizelge 3.4. İkinci yazılım temelli ağ veri kümesindeki kayıtların dağılımı..... 112

Çizelge 3.5. Karışıklık matrisi 114

Çizelge 3.6. Başlangıçta belirlenen özellikler..... 117

Çizelge 3.7. Başlangıçta seçilen 10 özelliği kullanan RF performansı..... 118

Çizelge 3.8. Çok sınıflı sınıflandırma için en önemli 10 özellik 118

Çizelge 3.9. Çok sınıflı sınıflandırma için RF modelinin hiperparametreleri 119

Çizelge 3.10. En iyi 10 özelliği kullanan RF performansı..... 119

Çizelge 3.11. İkili sınıflandırmada kullanılan 12 özellik..... 121

Çizelge 3.12. İkili sınıflandırma için RF modelinin hiperparametreleri..... 121

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Temel SDN işleyiş modeli 6
Şekil 2.2. Dağıtılmış hizmet reddi saldırı türleri 9
Şekil 2.3. Standart yazılım temelli ağ 17
Şekil 2.4. MiddleModule sistemi kapsamında yazılım temelli ağ 18
Şekil 2.5. MiddleModule blokları 23
Şekil 2.6. MiddleModule mesaj protokolü 27
Şekil 2.7. MiddleModule işlemlerine yönelik diyagram 29
Şekil 2.8. Ağ/Aktarım katmanındaki DDoS saldırıları tespit etme özellikleri 31
Şekil 2.9. OMNET++ kapsamında sistem performansı düşürme testi senaryosu 53
Şekil 2.10. OMNET++ için temel savunma gücü test senaryosu 58
Şekil 2.11. OMNET++'tan temel savunma gücü testi sonuçları 59
Şekil 2.12. OMNET++ için karmaşık savunma gücü test senaryosu 60
Şekil 2.13. Konfigürasyon değişkenleri testi ile savunma gücü değişimi senaryosu 62
Şekil 2.14. Konfigürasyon değişkenlerine göre savunma gücü varyasyon sonuçları 65
Şekil 2.15. MİNİNET açısından sistem performans düşüşü test senaryosu 68
Şekil 2.16. Temel savunma gücü testleri için test senaryosu 70
Şekil 2.17. IP Kandırma saldırısı tespit sonucu 71
Şekil 2.18. Güvenilir savunma gücü testleri için test senaryosu 73
Şekil 2.19. Veri katmanı cihazı MiddleModule uygulama önerisi 78
Şekil 3.1. SDN mimarisi ve saldırı yüzeyi 88
Şekil 3.2. Akış kuralı yapısı 89
Şekil 3.3. SDN'de paket akışı 89
Şekil 3.4. Yazılım tanımlı 5G ağ mimarisi 92
Şekil 3.5. Rastgele ormana göre sınıflandırma 94

Şekil **Sayfa**

Şekil 3.6. Bir ağ saldırı tespit görevi için rastgele ormandan örnek karar ağacı..... 94

Şekil 3.7. SDN tabanlı güvenlik çözümü mimarisi..... 99

Şekil 3.8. İlk veri seti için uygulanan test yatağı kurulumu..... 103

Şekil 3.9. 1000 akış girişine kadar özellik alma süresi 112

Şekil 3.10. 20000 akış girişine kadar özellik alma süresi 113

Şekil 3.11. Model karşılaştırması..... 116

Şekil 3.12. SDN'de DDoS saldırılarının kapsamlı sınıflandırması ve katmanlı güvenlik açıkları..... 124

Şekil 3.13. DDoS algılama modeli çerçeve diyagramı. 124

Şekil 3.14. DDoS saldırılarının sayısı (2023-2024). 125

Şekil 3.15. MSAAD Framework genel bakış..... 125

SİMGELER VE KISALTMALAR

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

Simgeler	**Açıklamalar**
Gb	Gigabayt
Gbps	Gigabit
Kb	Kilobayt
Kpps	Kilobit
Mb	Megabayt
Ms	Milisaniye
Sn	Saniye
Ms	Mikrosaniye

Kısaltmalar	**Açıklamalar**
5G	5. Generation
API	Application Programming Interface
CPU	Central Process Unit
DDoS	Distributed Denial of Service
DoS	Denial of Service
FPGA	Field Programmable Gate Array
http	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
ID	Identity (Number)
IDP	Identity Provider
IoT	Internet of Thinks
IP	İnternet Protokolü
LAN	Local Area Networks
MAC	Media Access Control
NFV	Network Function Virtualization
ONF	Open Networking Foundation

Kısaltmalar	Açıklamalar
QoS	Quality of Service
RAM	Random Access Memory
RF	Random Forest
SDN	Software Defined Network
SOM	Self-Organizing Maps
SVM	Support Vector Machines
SYN	Synchronous
TCAM	Ternary Content Addressable Memory
TCP	Transmission Control Protocol
VLAN	Virtual Local Area Networks

1. GİRİŞ

Yazılım Temelli Ağ (Software Defined Network - SDN), yeni bir ağ mimarisi ve paradigmasıdır. Son günlerde umut verici ilerlemeler sunmakla birlikte endüstri ve akademinin dikkatini çekmektedir. Fakat SDN, Dağıtılmış Hizmet Reddi (Distributed Denial of Service - DDoS) saldırılarına karşı savunmasız kalmaktadır (Yan, Yu, Gong ve Li, 2015). DDoS saldırıları, ağ oluşturma alanında her zaman büyük bir problem olarak görülmüştür. Ancak SDN açısından DDoS saldırılarının etkileri geleneksel ağlardan daha yıkıcı olmaktadır. Esas olarak, SDN bu saldırıların gerçek hedefi durumuna gelebilmekle beraber bu saldırılardan dolayı bütün bir SDN yasal kullanıcılar tarafından kullanılamaz hale gelebilmektedir (Yan vd., 2015). SDN kapsamında DDoS'a yönelik savunma yöntemleri hala çözülmemiş bir alandır ve bu probleme yönelik kabul edilen kesin bir çözüm yöntemi bulunmamaktadır (Kalkan, Gür ve Alagöz, 2017). Bu tezde, çeşitli ihtiyaçlara çözüm olarak veri katmanı tabanlı bir DDoS algılama ve azaltma metodunun, SDN'lere yönelik ortam açısından DDoS saldırılarına karşı etkili bir savunma mekanizması sağlayacağı önerilmektedir. Önerinin doğrulanması bakımından veri katmanı tabanlı bir algılama ve azaltma çerçevesi olan MiddleModule sistemi sunulmakla beraber birkaç algılama algoritması önerilmekte ve önerilen sistem kapsamlı bir şekilde değerlendirilmektedir.

Tezin amacı kapsamında SDN, ağ oluşturmaya birçok önemli nitelik getirmesine karşın, SDN'lerin DDoS saldırılarına karşı güvenlik açıklarından dolayı SDN'lerin kullanım durumu hala çok sınırlı kalmaktadır (Yan vd., 2015). Bu tezdeki amaç, DDoS saldırıları altında başarısızlığa yol açabilen SDN'lerin temel problemlerine açıklık getirmektir. Bunun yanı sıra, düşük sistem performansı ve yüksek savunma gücü performansı ile DDoS saldırılarını etkin bir şekilde tespit edebilen ölçeklenebilir ve kapsamlı bir savunma mekanizması sunulmaktadır. Zira günümüze kadar literatürde tavsiye edilen ve iyi bilinen savunma yöntemleri açıklanmakta ve sınıflandırılmaktadır.

Bu tezde, veri katmanı tabanlı bir DDoS algılama ve azaltma tekniği kullanılması önerilmektedir. İlk olarak, veri katmanı tabanlı bir yöntemin en iyi performansı ortaya koymasına yönelik karşılaması gereken birkaç gereksinim ifade edilmektedir. Bu gereksinimler göz önüne alınarak, bir SDN'nin her uç anahtarının istatistikleri topladığı ve gelen paketlere istatistiksel tabanlı algılama algoritmaları uyguladığı MiddleModule çerçevesi sağlanmaktadır. Herhangi bir kötü amaçlı etkinlik algılanırsa, ilgili uç anahtarı

tarafından uygun azaltma tekniği uygulanmaktadır. Bu işlevselliği etkili bir şekilde sağlamak için çoklu konum savunma yaklaşımı kullanılmakta, yani ağın farklı konumlarında farklı algılama işlevleri gerçekleştirilmektedir. Önerilen savunma sistemi ile kötü amaçlı trafik denetleyiciye gönderilmemektedir. Çünkü denetleyici aşırı yüklenmesini ve anahtar-denetleyici iletişim kanalı tıkanıklık sorunlarının önlenmesi amaçlanmaktadır. Kötü amaçlı trafiği, kaynağına en yakın anahtarda tespit edip tepki vermek önem teşkil etmektedir. Önerilen yöntem açısından işlevsel analiz ve performans değerlendirmeleri yapılmakta ve test sonuçları literatürde bulunan benzer araştırmalar ile kıyaslanmaktadır.

Tezin katkıları açısından, Ağ/Aktarım katmanında DDoS saldırılarının SDN'ler üzerindeki etkilerinin kapsamlı bir açıklaması sunulmaktadır ve hali hazırdaki savunma yöntemleri karşılaştırılmaktadır. SDN'ler açısından veri katmanı tabanlı bir DDoS savunma sisteminin karşılaması gereken, hafif algoritmalara sahip olmak ve çok konumlu savunma yaklaşımı kullanmak gibi bazı temel özellikler üzerinde durulmaktadır. Tavsiye edilen nitelikler kullanarak, uç anahtarlara izleme, algılama ve önleme yetenekleri getiren veri katmanı tabanlı bir DDoS savunma mekanizması olan MiddleModule çerçevesi açıklanmaktadır. Bunun yanı sıra, her bir ağ saldırı türünün en yaygın örneklerine karşı etkili olması beklenen birkaç DDoS algılama algoritması da sunulmakta ve bu algoritmalar MiddleModule çerçevesi içerisinde uygulanmaktadır. Tavsiye edilen metodun bir SDN katmanında DDoS saldırılarına karşı etkili bir şekilde koruyabildiğini göstermek için kapsamlı analizler ve değerlendirmeler gerçekleştirilmiştir. Önerilen metot kapsamında farklı test senaryoları altında detaylı bir fonksiyonel analiz ortamı için OMNET ortamı göz önünde bulundurulmaktadır. Simülasyona ek olarak, yöntemin performansını analiz etmek için önerilen yöntem MİNİNET ortamında da benzetilmektedir. Test sonuçları kullanılarak önerilen yöntem literatürdeki benzer çalışmalarla karşılaştırılarak avantaj ve dezavantajları tartışılmıştır.

Gerçek zamanlı saldırıların önlenmesi sistemi doğrultusunda mobil ağlar, büyük veri ve bulut bilişim alanlarındaki gelişmeler sayesinde bağlantılı cihazların sayısı ve Nesnelerin İnterneti (Internet of Thinks - IoT) kullanımı devamlı artış göstermektedir (Gökrem ve Bozuklu, 2016). Günlük yaşamı kritik derecede etkileyen ve kolaylaştıran Nesnelerin İnterneti kullanım örnekleri arasında akıllı evler, otonom arabalar, güvenlik sistemleri, akıllı şehirler ve uzaktan sağlık hizmetleri bulunmaktadır. Nesnelerin İnterneti tarafından üretilen büyük hacimli veriler göz önünde bulundurulduğunda bu çeşitli kullanım durumlarının

hizmet kalitesi gereksinimlerinin eski kablosuz ağlar tarafından karşılanmayacağı ifade edilmektedir. Kaynak yönetimi için SDN'ler ve Ağ İşlevi Sanallaştırmasına (Network Function Virtualization - NFV) dayanan 5G ve ötesi ağlar, geleceğin her yerde bulunan IoT kapsamında önem arz etmektedir.

Etkili model oluşturmaya yönelik gerçekçi ağ trafiği verilerinin önemine karşın, IoT saldırı tespitindeki mevcut araştırmaların çoğu, IoT trafiği olmayan eski ağlar için oluşturulmuş veri kümeleri kullanmaktadır. Bu çoğunlukla yakın zamanda yayınlanan Bot-IoT veri seti haricinde IoT trafiğini içeren halka açık veri setlerinin yer almamasından kaynaklanmaktadır (Koroniotis, Moustafa, Sitnikova ve Turnbull, 2019). Özellikle SDN'ler kapsamındaki IoT'ne dair ortamlar için halka açık herhangi bir veri kümesi bulunmamaktadır. IoT ve SDN'lerin ağ trafiği nitelikleri, eski ağlardan epeyce farklılık göstermektedir. Bundan dolayı eski ağ verileriyle desteklenen modellerin kullanılması yanlış sınıflandırma sonuçlarına sebebiyet verebilmektedir. Bunun yanı sıra, etkin saldırı tespiti ve hafifletme için izinsiz giriş tespit sistemlerinin özellikleri gerçek zamanlı olarak önem taşımaktadır.

SDN'ler katmanında IoT ağlarına yönelik hali hazırdaki güvenlik yaklaşımlarının eksiklikleri gidermek amacıyla, bu tezde gerçek zamanlı bir saldırı tespit ve önleme sistemi sunulmaktadır. SDN'ler için, denetleyicideki trafik akışlarını izleyen ve makine öğrenimine dayalı akıllı bir saldırı algılama modülü kullanarak trafiği sınıflandıran, akışı buna göre yapan uçtan uca bir saldırı tespit ve önleme sistemi mimarisi sunulmaktadır. Normal trafik ve farklı saldırı trafiği türlerini içeren bir IoT ağında yeni bir SDN'ye yönelik veri kümesi belirtilmektedir. Sunulan veri setinin performansı farklı makine öğrenme algoritmaları ile değerlendirilip en önemli özellikler belirlenmektedir. Bunun yanı sıra sunulan güvenlik sisteminin varlığında ve yokluğunda ağ performansı neticeleri belirtilmekle beraber sistemin gerçek zamanlı korumaya ulaşması hedeflenmektedir.

Bu tez çalışmasının ikinci bölümde SDN'ler ve DDoS saldırıları kapsamında veri katmanı tabanlı sistemler açıklanmıştır. SDN, DDoS ve aralarındaki ilişkiler açıklanmıştır. Ayrıca sistem tasarımı, tespit algoritmaları ve genel değerlendirmeler yapılmıştır. Üçüncü bölümde gerçek zamanlı saldırıların önlenmesi sistemi üzerine çalışmalar yapılmıştır. Son bölümde tez çalışmasının özeti verilmiş, genel bir değerlendirmesi yapılmış ve gelecekte yapılacak çalışmalar için performans iyileştirme amacı ile çeşitli önerilerde bulunulmuştur.

2. YAZILIM TEMELLİ AĞLAR VE DAĞITILMIŞ HİZMET REDDİ SALDIRILARI KAPSAMINDA VERİ KATMANI TABANLI SİSTEM

2.1. Genel Bakış

2.1.1. Yazılım temelli ağlar (SDN)

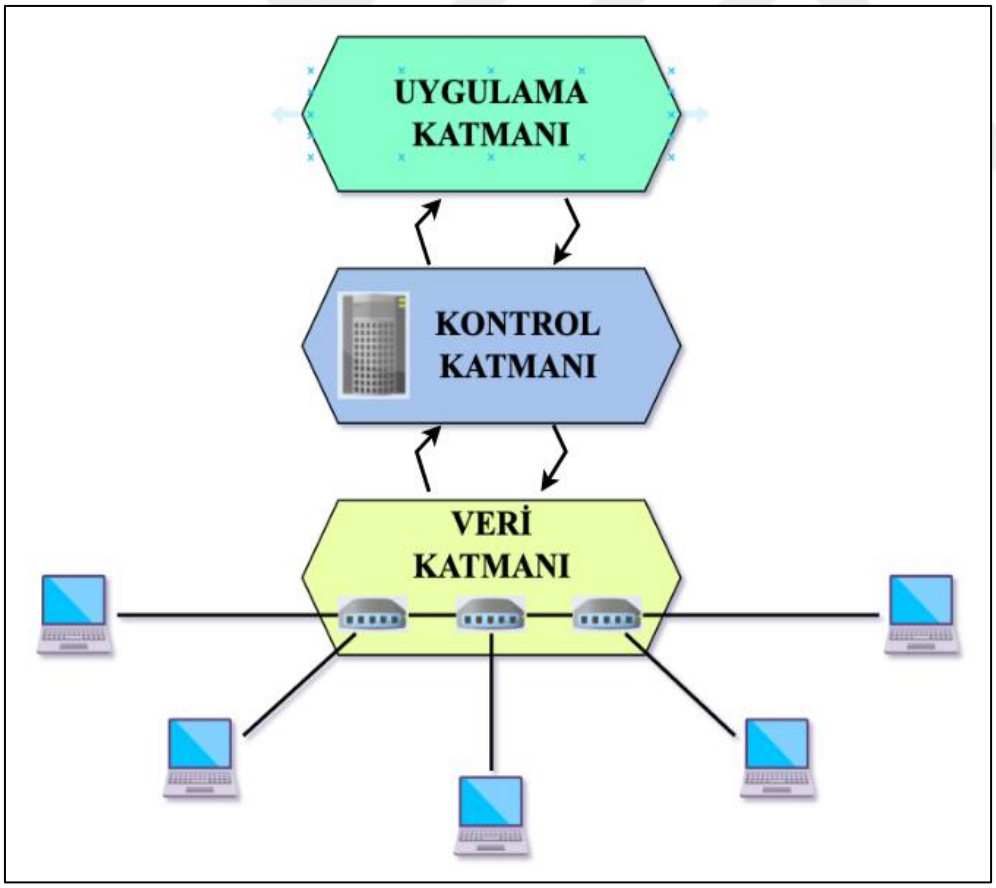
Yazılım Temelli Ağ, ağ oluşturma alanında yeni bir paradigma olarak görülmektedir. SDN, ağı farklı katmanlara; uygulama katmanına, kontrol katmanına ve veri katmanına ayırarak geleneksel ağdan farklı bir ağ mimarisi önermesi itibariyle yeni bir uygulama alanı olarak değerlendirilmektedir (Yan vd., 2015).

Geleneksel ağlarda, iki temel ağ oluşturma süreci bulunmaktadır. Bu ağ oluşturma süreçleri yönlendirme ve iletme olarak isimlendirilmekte ve yönlendiriciler tarafından ağlar meydana gelmektedir. Yönlendiriciler ağ boyunca dağıtılmakta ve birbirleriyle iletişim kurarak bağımsız bir şekilde çalışmaktadırlar. Ancak, Şekil 2.1'de gösterildiği gibi, SDN'de bu iki işlem, farklı ağ katmanlarında dağıtılan farklı cihazlarda işlenmektedir. Yönlendirme kararları, ağın kontrol katmanında dağıtılan ve denetleyici olarak adlandırılan, mantıksal olarak merkezileştirilmiş bir cihaz tarafından verilmektedir. Denetleyici, evrensel bir ağ görünümüne sahiptir ve uygulama katmanına ait diğer algoritmalarla birlikte gerekli yönlendirme algoritmalarını çalıştırmaktadır (Kreutz, Ramos, Verissimo, Rothenberg, Azodolmolky ve Uhlig, 2015).

Kontrol katmanı bir anlamda ağın zeka; akıl, kısmını temsil eder. Öte yandan, yönlendirme işlemi, ağın veri katmanında konuşlandırılan ve anahtar adı verilen mantıksal ve fiziksel olarak dağıtılmış cihazlar tarafından gerçekleştirilir. Denetleyiciden alınan yönlendirme kurallarına göre paketleri iletir; bu nedenle, veri katmanı bir anlamda ağın akıllı olmayan kısmı olarak ifade edilebilir (Yan vd., 2015). Her anahtar, denetleyici ile güvenli bir kanal üzerinden iletişim kurar ve iyi tanımlanmış bir protokol kullanarak bu kanal üzerinden yönlendirme kurallarını temin eder. Ağ iletişiminde ‘OpenFlow protokolü’ anahtar denetleyici iletişim protokolü olarak çok yaygın olarak kullanılmaktadır (Kalkan vd., 2017).

SDN bağlamında sistem, Şekil 2.1'de gösterildiği gibi, tüm ana bilgisayarlar doğrudan veri katmanı cihazlarına bağlanır ve ana bilgisayarlar arasında gerekli bağlantıyı sağlamak için

veri katmanı cihazları birbirine bağlı biçimde bulunmaktadır. Bir anahtar bir ana bilgisayardan bir paket aldığında, akış tablosunda alınan paket için eşleşen bir akış girişi olup olmadığını kontrol etmektedir. Eşleşme bulunması halinde anahtar ilgili yönlendirme kuralına göre paketi bağlantı noktalarından birine iletme görevini üstlenmektedir. Eşleşmenin bulunmadığı durumlarda, anahtar denetleyici iletişim kanalını kullanarak, paketin veri bölümü anahtar arabelleğinde depolanırken anahtar, alınan paketin başlık bölümüyle birlikte denetleyiciye bir geri bildirim mesajı (package-in) iletmektedir. SDN Denetleyicisi, 'package-in' sorgusunu kendi yönlendirme algoritmalarını kullanarak işler ve ilgili yönlendirme kuralını anahtara göndermektedir. Anahtar, yönlendirme kuralını yönlendirme tablosunda depolar, paketin saklanan veri kısmını kuyruktan çıkarır ve paketi karşılık gelen bağlantı noktasına ileti gerçekleştirir. Bu paket ilgili yönlendirme yolunda yer alan anahtardan anahtara iletilecek ve son olarak paket hedef ana bilgisayara ulaşır (Yan vd., 2015).



Şekil 2.1. Temel SDN işleyiş modeli

SDN'nin arkasındaki ana fikir olan veri katmanı ve kontrol katmanının birbirinden ayrılması, SDN mimarisinden daha uzun bir geçmişe sahip olan bir durum olarak öne çıkmaktadır. Bu alan içerisindeki ilk uygulamalar, telefon şebekelerini geliştirmek için, veri/kontrol katmanını ayırma girişimleri 1980'li ve 1990'lı yıllarda meydana gelmiştir. Ayrıca, aktif ağlar veya programlanabilir veri katmanı kavramları da ağ oluşturma alanında geçmişten günümüze gelen kavramlar olarak bilinmektedir. Programlanabilir anahtarlar 1990'larda tartışma konusu olarak öne çıkan ve bu kavramı gerçekleştirmek için bazı komut setlerinin anahtarlara indirilmesi ve anahtarların paket yönlendirme kurallarının değiştirilmesi gibi çeşitli yöntemler önerilen bir olgu olarak gösterilebilmektedir. SDN ise daha kompakt ve protokol tabanlı bir mimariyi temsil etmektedir. Temelde, SDN algoritmasının kökenleri, OpenFlow protokolü etrafındaki çalışmalar ve fikirler olarak kabul edilir (Kreutz vd., 2015).

SDN'nin merkezi yönetim sağlaması, ağ çapında trafiği değişen ihtiyaçlara göre ayarlaması, ağ kaynaklarını otomatik ve dinamik SDN programları aracılığıyla yönetmesi ve ağı satıcılardan bağımsız yönetmesi gibi birçok avantajı bulunmaktadır ve bu avantajları bağlamında kullanımı yaygınlaşmaktadır (Zargar, Joshi ve Tipper, 2013).

SDN mimarisinin arkasındaki gerçek motivasyon izolasyon ve sanallaştırma olarak ifade edilmektedir. SDN, ağ mühendislerinin farklı ağ yönlendirme kurallarını çalıştırmasına ve aynı anda farklı kullanıcılara ait farklı yalıtılmış trafikler için aynı ağ üzerinde farklı bağlantı araçları sağlamasına olanak tanıyan bir uygulamadır. SDN'nin akademik kullanım ve iş kullanımı için çok fazla avantajlı özelliği bulunmaktadır (McKeown, Anderson, Balakrishnan, Parulkar, Peterson, Rexford ve Turner, 2008).

Google, Akamai, Cisco ve Microsoft dahil olmak üzere birçok şirket, SDN'yi açık standartlar kullanarak ilerletmek amacıyla Açık Ağ Kuruluşu (Open Networking Foundation - ONF) üyesi olmaktadırlar. Son zamanlarda Google, dünya çapındaki veri merkezlerini birbirine bağlamak için SDN tabanlı bir ara bağlantı ağı kurmuştur. SDN, ağ iletişimi alanında dikkat çekmesine rağmen, endüstri uzmanları, SDN'nin güvenlik konularının hala olgunlaşmamış olduğunu ve SDN'nin güvenliği konusunda daha fazla araştırma yapılması gerektiğini öne sürerek bu alanın daha güvenli bir alan olması için çalışmalar yürütmektedirler (Kreutz vd., 2015).

2.1.2. Dağıtılmış hizmet reddi (DDoS)

Hizmet Reddi (Denial of Service - DoS) saldırıları, hatalı biçimlendirilmiş mesajlar göndererek saldırı yapılacak sunucunun erişilebilirliğini bozmayı amaçlar. Saldırganın bir sunucuya saldırması için finansal kazanç, intikam, ideolojik inanç, entelektüel meydan okuma veya siber savaş gibi birçok farklı neden sayılabilir (Zargar vd., 2013:2050).

Kötü amaçlı etkinlik birden fazla kötü niyetli kullanıcı tarafından desteklendiğinde, DoS saldırısı Dağıtılmış DoS (DDoS) haline gelir. DDoS saldırıları, DoS saldırılarından çok daha zordur, çünkü kötü niyetli kullanıcılar bir DDoS saldırısı kullanarak kolaylıkla fazla miktarda kötü yazılım içeren ağ trafiği meydana getirebilmektedirler. Bu tür bir saldırı altında, saldırıya uğrayan sunucu yasal kullanıcılar tarafından erişilemez hale gelir. Saldırı yapılan sunucuya ek olarak, altta yatan ağ da bu tür saldırılardan zarar görür. Altta yatan ağ çok fazla paketle tıkanmakta ve ağ cihazları, arabellek taşmaları ve aşırı paket düşüşleri nedeniyle yasal kullanıcılara bazı hizmetler sağlanamaz. DoS saldırısı kavramı sanal sistemler içerisinde uzun süredir varlığını sürdüren bir saldırı biçimi olarak öne çıkar. Bilinen ilk DoS saldırısına 1980'lerin başında rastlanmıştır. Bilgisayar ağlarının büyümesi ve internetin tüm dünyada yaygınlaşması ile DDoS saldırıları ortaya çıkmaya başlamış ve ilk DDoS saldırısı 1999 yılında kaydedilmiştir (Zargar vd., 2013:2052).

Ağlar genişledikçe DDoS saldırıları daha karmaşık hale gelmiş ve bu saldırıların uygulanma biçimleri gelişmiştir. İnternet teknolojileri artık daha gelişmiş olduğundan çeşitli saldırı araçları geliştirilmiştir, bu nedenle DDoS saldırısı yapmak çok daha kolay hale gelmiş ve DDoS saldırılarının boyutu sürekli olarak artmaktadır. Bu artış daha fazla trafiğe neden olmakta, sistem aksamalarını daha büyük boyutlu hale getirmektedir. 2010 yılında bildirilen en büyük DDoS saldırısının boyutu yaklaşık 100 Gbps iken, 2013 yılında yaklaşık 300 Gbps'dir (Yan vd., 2015: 614).

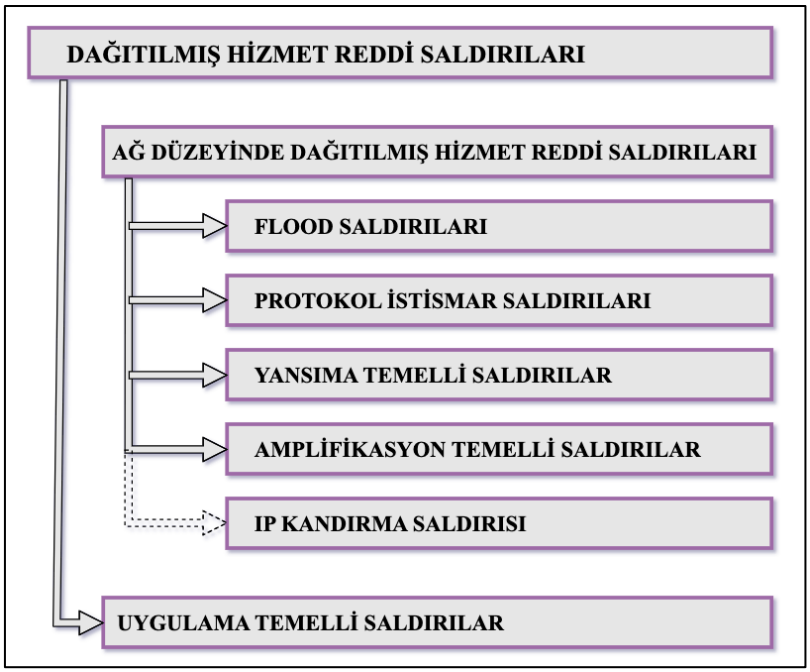
DDoS saldırıları Şekil 2.2’de gösterildiği gibi iki kategoriye ayrılır. Önemli başlıklar şu şekilde ifade edilmektedir (Yan vd., 2015: 615);

1. Flood Saldırıları: Temel Ağ/Aktarım katmanında DDoS saldırısı türüdür ve saldırı yapılan sunucunun ağ bant genişliğini tüketmeyi ve bu sunucuya yasal bağlantıyı bozmayı amaçlamaktadır.

2. Protokol İstismar Saldırıları: Bu saldırı türünde saldırgan, kurbanın bağlantı protokolünün bazı özelliklerini kullanarak veya uygulama hatalarından yararlanarak kurban sunucunun kaynaklarını tüketmeyi amaçlamaktadır.

3. Yansıma Temelli Saldırılar: Kötü amaçlı kullanıcılar, hatalı biçimlendirilmiş istekleri doğrudan kurbana göndermek yerine yansıtıcı kullanıcılara gönderir; bu durumda, yansıtıcı kullanıcı, ağda saldırgan ve kurban dışında üçüncü bir ana bilgisayardır. Hatalı isteklerde kaynak adres kısmı mağdurun adresi ile değiştirilir. Bu nedenle reflektörlerden (yansıtıcılardan) gelen yanıtlar kurbana gönderilir ve kurbanın bant genişliği ve kaynakları tüketilir. Smurf saldırısı gibi organize ve dağıtık bir flood saldırısı bu saldırı tipine örnek olarak gösterilebilir.

4. Amplifikasyon Temelli Saldırılar: Saldırganlar, saldırılarını kurbana doğru büyüterek saldırılarının etkilerini artırmayı amaçlayabilir. Bu saldırı genellikle Yansıma Temelli Saldırılarla birlikte kullanılır. Saldırgan, Amplifikasyon Tabanlı Saldırı kullanarak, böyle bir saldırı oluşturmak için kurban sunucunun kendi bant genişliğinden çok daha fazla bant genişliğini kullanır. En yaygın kullanımlardan biri, Smurf saldırısında olduğu gibi, reflektörlerin yayın mesajları üretmesini sağlamaktır.



Şekil 2.2. Dağıtılmış hizmet reddi saldırı türleri

2.1.3. Yazılım temelli ağlar ve dağıtılmış hizmet reddi arasındaki ilişkiler

DDoS saldırıları ile SDN'lerin mimarisi arasında çelişkili bir ilişki bulunmaktadır. SDN'ler, bir kurban sunucu üzerinden bir DDoS saldırısını tespit etmek ve önlemek için kullanılabilmektedir, ancak SDN'lerde bazı durumlarda saldırıya uğrayabilmektedir. Ana bilgisayarları DDoS saldırılarına karşı korumak için SDN'ler (Yan vd., 2015:616);

- Mantıksal olarak merkezileştirilmiş denetleyici ve küresel ağ görünümüne,
- Ağ öğelerinin programlanabilirliği özelliğine,
- Dinamik yönlendirme kurallarına,
- Ağ genelinde dağıtılmış düğümlere ve
- Bir savunma sistemini kolayca test etme gibi çeşitli özelliklere sahip olmaktadır.

Ana bilgisayarları DDoS saldırılarına karşı savunmak için kullanılabilen SDN'lerin sıralanan tüm olumlu özelliklerine rağmen SDN'ler saf ana bilgisayarlara hizmet etmek için tasarlandığından dolayı SDN'nin kendisi bu saldırıların hedefi haline gelebilmektedir. Kötü amaçlı bir DDoS etkinliği, SDN düğümlerinin hizmetlerini kesintiye uğratabilir ve bu da tüm ağın başarısız olmasına neden olabilecek durumlar meydana getirebilir. Sahte veya sahte olmayan, birkaç ölü ana bilgisayar veya bottan kaynaklanan çok sayıda paket ciddi sorunlara neden olabilmektedir. Bu sorunlardan bazıları şu şekilde açıklanmaktadır (Mousavi ve St-Hilaire, 2015:77):

Anahtar tamponu ve akış tablosu taşması

Bir OpenFlow anahtarı, akış kimliği anahtarın akış tablosunda olmayan bir paket aldığında; anahtar, paketin veri kısmını arabelleğinde saklar ve başlık kısmını bir 'package_in' istek mesajı ile denetleyiciye gönderir. Denetleyici gelen talebi işler ve anahtarın akış tablosunda saklanan uygun akış kuralını anahtara gönderir. Bir anahtar, akış kimlikleri bilinmeyen ana bilgisayarlardan çok fazla paket alırsa, anahtar arabelleği taşar ve akış kimliği bilinmeyen bir paket alındığında, anahtar veri bölümünü depolamak yerine paketin tamamını denetleyiciye gönderir. Bir DDoS saldırısı sırasında, anahtarların arabellekleri kolayca aşırı yüklenebilir. Ayrıca, denetleyiciden alınan yedekli akış kuralları da anahtar akış tablolarında saklanır. Her anahtarın sınırlı boyutlu bir akış tablosu olduğundan, çok fazla yedek akış

kuralı alındığında, geçerli olanlar akış tablosundan çıkarılabilir ve yasal trafiğin iletme hızı önemli ölçüde düşebilmektedir.

Anahtar denetleyici kanalı bant genişliği sıkışıklığı

Anahtarlar kısa bir zaman aralığında çok fazla ‘package_in’ mesajı gönderdiğinde, anahtar denetleyici iletişim kanalı tıkanır. Ayrıca, bir anahtar arabelleği taşmasını takiben, bu iletişim kanalı daha kolay tıkanır ve periyodik ve düzensiz mesajlar dahil olmak üzere tüm OpenFlow trafiğini anahtarlar ve denetleyiciler arasında taşıyan bu kanalda gecikmeler veya paket düşüşleri meydana gelir. Bu nedenle, böyle bir sorun oluştuğunda, böyle bir iletişim kanalını paylaşan tüm anahtarlar zarar görür ve ilgili anahtarların yönlendirme performansı önemli ölçüde düşer.

Denetleyici arabellek taşması

Denetleyici bir ‘package_in’ isteği aldığında, isteği işler ve uygun iletme kurallarını bulur. Her istek için bu tür eylemleri hesaplamak, bellek ve Merkezi İşlem Birimi (Central Process Unit - CPU) gibi denetleyici kaynaklarını tüketir. Denetleyici güçlü bir aygıt olmasına rağmen yine de sınırlı miktarda kaynağa sahiptir. Bir DDoS saldırısı sırasında, çok sayıda gereksiz ‘package_in’ isteği denetleyicinin kaynaklarını tüketebilir; bu nedenle, denetleyici yasal istekleri bırakmaya veya onları geciktirmeye başlar. Bu nedenle, Yazılım Temelli Ağlarda mantıksal olarak merkezi bir denetleyiciye sahip olduğu için bu tür saldırılarda tek nokta hatası sorunu ortaya çıkar. Denetleyici yasal ‘package_in’ isteklerini bıraktığında veya geciktirdiğinde, tüm ağ düzgün şekilde çalışamaz. Denetleyici, yasal kullanıcılar için erişilemez hale gelebilir. Bu sorun, bir yedek denetleyici olsa veya denetleyicinin mantıksal olarak merkezileştirilmesi nedeniyle denetleyici fiziksel olarak dağıtılsa bile oluşmaktadır.

Özetle, SDN mimarisi, SDN'yi DDoS saldırılarına karşı savunmasız hale getiren birkaç önemli soruna sahiptir ve bu sorunlar aşağıdaki üç ana özelliğe sahip olduğu için ortaya çıkmaktadır (Scott-Hayward, O’Callaghan ve Sezer, 2013:2);

- SDN bağlı tüm ana bilgisayarlara güvenmekte ve riskleri göz ardı etmektedir.
- Mantıksal olarak merkezileştirilmiş kontrol katmanı SDN'nin tek hata noktası olarak öne çıkmaktadır.

- SDN sınırlı boyutlu arabelleklere ve sınırlı boyutlu anahtar akış tablolarına sahiptir.

Bu nedenle, SDN için herhangi bir kapsamlı DDoS savunma mekanizması bunları dikkate almalıdır.

2.1.4. Kontrol katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri

Bir savunma yöntemi, ağ çapında bir savunma mekanizması sağlamak için merkezi denetleyicide veya başka herhangi bir merkezi cihazda çalışan DDoS algılama ve azaltma modülüne veya algoritmasına sahip olduğunda, buna Kontrol Katmanı Tabanlı DDoS Savunması adı verilir. Bu yöntemler, bir SDN ağında kolayca tasarlanıp uygulanabildikleri için yaygın olarak kullanılmaktadır. SDN merkezi denetleyici birçok araştırmacı tarafından incelenen bir konu olarak öne çıkmaktadır. Bu tür çalışmalar arasında veri izleme teknikleri veya algılama yaklaşımları farklılık gösterse de kullanılan çerçeve genellikle ortaktır. Bir araştırmada istatistik toplamak ve DDoS saldırılarını tespit etmek için denetleyicinin kullanılmasını önerilmektedir (Mousavi ve St-Hilaire, 2015:78). Bu amaçla, kontrol cihazı üzerinde her hedef IP adresinin (Internet Protocol adress) entropi değişimlerini kontrol eden entropi tabanlı bir uygulama çalışmaktadır. Ağdaki paketler homojen olarak tüm hedeflere gönderilmişse, her hedefin bir paket alma olasılığı birbirine yakın olur ve entropi yüksek olur. Öte yandan, paketler yalnızca bir hedefe yönlendirilmişse, bu hedefin bir paket alma olasılığı yüksek, diğerlerinin düşük olması ve entropi değeri daha küçük olmasıdır. Denetleyici entropiyi hesaplar ve bunu bir eşikle karşılaştırır. Entropi bir eşiğin altındaysa, denetleyici bir DDoS saldırısı olduğu sonucuna varır ve kurban sunucuya gönderilen tüm paketleri bırakmak için hafifletme kuralları uygular. Farklı araştırmalar içerisinde DDoS saldırılarını önlemek için çeşitli Makine Öğrenimi tabanlı algılama algoritmaları mevcuttur (Ashraf ve Latif, 2014:55; Phan ve Bao ve Park, 2016:350).

Uygulama katmanında Machine Learning algoritmaları uygulanmakta, ağdan girdi özellikleri toplanmakta ve bu özellikler kullanılarak kötü amaçlı faaliyetler tespit edilmektedir. Literatürde yer alan bazı çalışmalarda OpenFlow anahtarları tarafından toplanan istatistikleri kullanan Öz Örgütlemeli Öğrenme (Self-Organizing Maps - SOM) algılama algoritmasını kullanan üst katman bir DDoS algılama uygulaması önerilmiştir (Braga, Mota ve Passito, 2010:408; Xu ve Liu, 2016:45). Bu yöntemler, denetleyicinin ek DDoS algılama algoritması yükü ile aşırı yüklenmesini önler; ancak, bu cihazlar iletişim kanalı tıkanıklık probleminden muzdariptir ve ilgili cihaz kaynakları tükenebilir. Bu

nedenle, bu yöntemler organize bir DDoS saldırısı altında tek hata noktası sorunuyla karşı karşıyadır. Farklı istatistik toplama teknikleri önerilmiştir, bazıları bu amaç için üçüncü taraf bir cihazın kullanılmasını önerir (Dong, Du, Zhang ve Xu, 2016:12); diğerleri ise OpenFlow anahtarlama yeteneklerinin ve OpenFlow protokolünün kendisinin istatistik toplama için yeterli olduğuna işaret etmektedir (Kim, Firoozjaei, Jeong, Kim ve Park, 2015:526).

Bazı yöntemler, DDoS saldırılarının neden olduğu ana sorunlardan yalnızca birine odaklanır. Bu yöntemler ilgili spesifik soruna umut verici çözümler önerse de genellikle diğer sorunlar üzerinde olumsuz etkileri vardır. Bir araştırmada, OpenFlow anahtar akış tablosu aşırı yükleme sorununu önlemek için, anahtarlar arasında QoS farkında yük dengeleme stratejisi önerilmiştir (Yuan, Zou, Yu, Jin, Qiang ve Shen, 2016:1).

Merkezi denetleyici sürekli olarak her bir anahtarın durumunu kontrol eder ve bir anahtarın akış tablosu doluysa, o anahtara yönlendirilen trafik diğer anahtarlara yeniden dağıtılarak anahtarın akış tablosunda aşırı yükleme yaşamamasını sağlar. Bununla birlikte, bu yöntem yalnızca akış tablosu aşırı yükleme sorununu dikkate alır ve bu sorunu çözmek için, denetleyiciye ek bir işlem yükü yükler, bu da denetleyici kaynaklarını bir DDoS saldırısı altında daha da hızlı tüketebilir. Bir başka araştırmada, denetleyici aşırı yükleme probleminden kaçınmak için, denetleyici için bir işleme sırası çizelgeleme algoritması önerilmiştir (Lim, Yang, Kim, Yang ve Lim, 2015:1259).

Denetleyici, kuyruğunu mantıksal olarak farklı mantıksal sıralara ayırır, bunların her biri farklı anahtarlar için ayrılır ve denetleyici her bir mantıksal kuyruğa döngüsel bir şekilde veya başka bir şekilde hizmet verir. Bu yöntem yalnızca denetleyici aşırı yüklenmesini dikkate aldığından, bir anahtar bir DDoS saldırısı altındayken arabellek aşırı yükleme sorunlarıyla daha da hızlı karşılaşacaktır; ilgili anahtardan geçen yasal trafik, büyük gecikmeler yaşayacak veya hiç hizmet alamayacak. Kontrol katmanı tabanlı yöntemler her ne kadar kullanımı kolay görünse de SDN'nin ana sorunu olan DDoS saldırıları altında tek nokta arızası sorununa kapsamlı bir çözüm sunmamaktadır. Ek bir merkezi algılama cihazı eklemek de soruna bir çözüm değildir, çünkü bu cihazlar tek arıza noktası problemini denetleyiciden devralır. Ayrıca, kötü amaçlı paketler denetleyiciye gönderilmeye devam ettiğinden, anahtar-denetleyici kanal tıkanıklığı sorunu da çözülememiştir.

2.1.5. Hibrit katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri

Hibrit Katmanı Tabanlı Dağıtılmış Hizmet Reddi Savunma Mekanizmaları, DDoS tespit ve azaltma operasyonları sırasında hem veri katmanı cihazlarını hem de kontrol katmanı cihazlarını kullanır. Veri katmanı cihazlarında paketler izlenir ve veri katmanı cihazlarında bir şekilde şüpheli paketler bulunursa daha sonra tüm şüpheli paketler daha fazla analiz için denetleyiciye yönlendirilir. Ana DDoS algılama ve önleme süreçleri, denetleyici tarafından gerçekleştirilir. Konu bağlamında yapılan bir çalışmada SDN tabanlı bulut ağlarındaki DDoS saldırıları için bir savunma yöntemi önerilmiştir (Rengaraju, Ramanan ve Lung, 2017:217). Bu yöntemde SDN anahtarları, ilkel bir analizle şüpheli ve şüpheli olmayan paketleri ayırt eder ve ardından şüpheli paketleri daha fazla analiz için denetleyiciye iletir ve denetleyici saldırıları algılar ve uygun azaltma kurallarına karar verir.

Yapılan bir başka araştırmada saldırı monitörlerinin ve ilişkilendiricilerin ağ boyunca dağıtıldığı ve herhangi bir kötü amaçlı etkinliği tespit etmek için anahtarlara yerleştirildiği çok seviyeli bir DDoS tespit yöntemi önerilmiştir (Chin, Mountrouidou, Li ve Xiong, 2015:659). Monitörler, Snort tabanlı IDS çalıştırır ve herhangi bir anormal aktivite tespit edilirse, anormal paket, kaynak adresin sahte olup olmadığını kontrol eden ilişkilendiriciye iletilir. Böyleyse, bu paketler daha fazla analiz ve ayrıca bir önleme kuralının oluşturulması için denetleyiciye yönlendirilir. Bu nedenle, hacimsel bir DDoS saldırısı altında, denetleyici cihaz yine de aşırı yüklenir; bu nedenle, ağ tek hata noktası sorunuyla karşı karşıyadır.

Konu ile ilgili verilebilecek son örnekte, NFV ve SDN, DDoS algılama/azaltma amacıyla birleştirilir. NFV kullanılarak veri katmanına belirli izleme işlevleri eklenir ve bu sanallaştırılmış işlevleri koordine etmek için merkezi bir orkestratör kullanılır. İzlenen veriler denetleyicide analiz edilir ve gerekli azaltma işlevleri, veri katmanında sanal işlevler veya yönlendirme kuralları biçiminde uygulanır. Hibrit yöntemler, merkezi denetleyicilerin iş yükünü azaltır; ancak, bu yöntemlerde, tüm kötü amaçlı trafik, algılama ve azaltma amaçları için yine de denetleyiciye yönlendirilir. Bu yöntemler, SDN'nin performansını denetleyici aşırı yüklenmesi ve iletişim kanalı tıkanıklık sorunlarına karşı iyileştirse de yine de bu sorunlara yeterli bir çözüm getirememektedir (Machado, Granville ve Schaeffer-Filho, 2016:391).

2.1.6. Veri katmanı tabanlı dağıtılmış hizmet reddi savunma yöntemleri

Veri Katmanı Tabanlı DDoS savunma mekanizmaları, DDoS saldırılarının tüm sorunlarına kapsamlı bir çözüm sağlamak için diğerlerine göre daha uygun bir yapıya sahip olarak öne çıkmaktadır. Veri katmanı cihazlarına zeka eklemek, veri katmanındaki DDoS saldırılarının algılanıp önlenmesine ve kötü amaçlı akışların veri katmanında tutulmasına yardımcı olur. Bu nedenle, denetleyici, iletişim kanalı ve ağın geri kalanı kötü amaçlı trafiğe açık değildir. Ek olarak, veri katmanı tabanlı bir yöntemde yayılma ve kuyruklama gecikmesi olmadığından, kötü amaçlı bir etkinliğe doğru bir yanıt nispeten hızlı ve etkili bir şekilde üretilebilir. Veri katmanı tabanlı DDoS savunma yöntemleri ile şu soru gündeme gelebilir; "Veri katmanı cihazlarına bu tür bir zeka eklemek, SDN'nin ana paradigmasını tehlikeye atabilecek durumalar ortaya çıkarır mı?" SDN mimarisinin arkasındaki ana fikir, yönlendirme kararları oluşturmak için kontrol katmanında akıllı bir cihaz kullanmak ve bu kararlara göre paketleri iletmek için veri katmanında birkaç akıllı olmayan cihaz kullanmaktır (Mousavi ve St-Hilaire, 2015:79).

SDN'nin ortak anlayışı yukarıda ifade edilen biçimde olsa da sunulan orijinal SDN mimarisinde, standart bir OpenFlow anahtarı tarafından desteklenen yeteneklerin esnek ve genişletilebilir olduğuna işaret edilmektedir. Aslında, bugün, standart bir OpenFlow Anahtarı, paket iletme dışında birkaç ek işlevsellik sağlar (Wang, Jia ve Ju, 2015:310).

Tüm OpenFlow anahtarlarda işlemciler, birkaç bellek birimi ve üzerlerinde çalışan bazı yazılım programları bulunur. Bir OpenFlow Anahtarı, işlemcisini, gelen/giden paketleri analiz etmek, onlar hakkında bir dizi ayrıntılı istatistik çıkarmak ve denetleyiciyle iletişim kurmak için kullanır. Bu nedenle, bu cihazlar akıllı olmayan olarak bilinmesine rağmen, bir dereceye kadar zekaya sahiptirler, ancak zekalarını yönlendirme kararları oluşturmak için kullanmazlar (Shin, Yegneswaran, Porras ve Gu, 2013:413).

Yapılan bir çalışmada bir OpenFlow anahtarın en azından aşağıdaki parçaları içermesi gerektiği açıkça belirtilmektedir (Chin vd., 2015:616);

- Akış Tablosu,
- Güvenli İletişim Kanalı
- OpenFlow desteği.

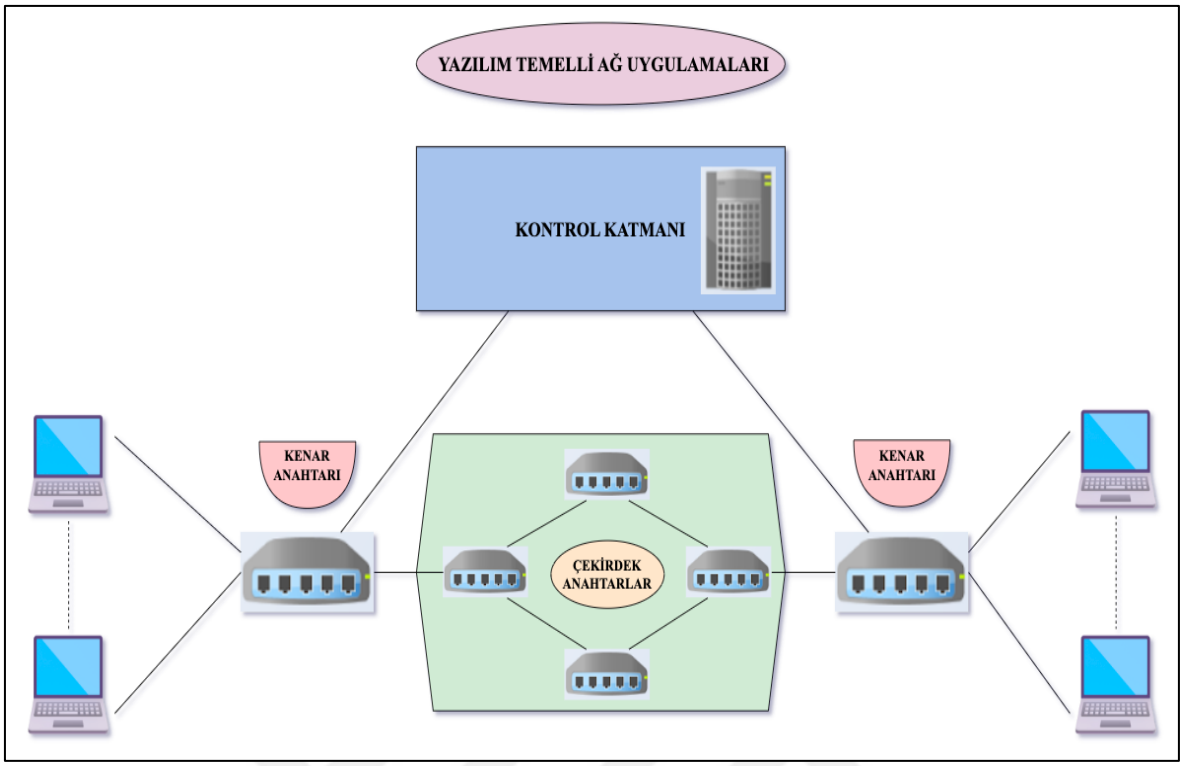
Ek olarak, OpenFlow Anahtarları için birkaç temel eylem seti sağlanır ve standart bir SDN mimarisi için yönlendirme/yönlendirme işlemi net bir şekilde açıklanır. Yönlendirme/yönlendirme özellikleri ve SDN'nin diğer temel özellikleri değiştirilmediği sürece, bir OpenFlow anahtarının yeteneklerini savunma amaçları için küçük zeka ekleyerek genişletmek, SDN'nin ana paradigmasını tehlikeye atmaz (Kalkan ve Gür ve Alagöz, 2017:673). Yapılan bazı çalışmalarda, DDoS saldırılarını tespit etmek ve azaltmak için bir anahtar içi işleme yöntemi önerilmektedir. İzlenen değerler, kötü amaçlı etkinliğin tespiti için anahtarlar üzerinde çalışan entropi tabanlı bir algoritma tarafından işlenir. Anahtarda bir saldırı algılanırsa, anahtar akışı keser ve denetleyiciye karşı önlemi belirlemesi için bildirimde bulunur. Bir paket bir anahtara ulaştığında, entropi tabanlı algoritma entropiyi hesaplar, eşiğin altında olup olmadığını kontrol eder ve eğer öyleyse bu paketi bir saldırı olarak etiketler. Ancak aynı anahtara bağlı birden fazla bilgisayar saldırı altındaysa bu algoritma saldırıyı doğru bir şekilde tespit edememektedir (Boite, Nardin, Rebecchi, Bouet ve Conan, 2017:12; Jantila ve Chaipah, 2016:437).

2.2. Sistem Tasarımı

Yazılım temelli ağ mimarisi açısından ağ katmanında DDoS saldırılarına karşı bir savunma sistemi sağlamak için, veri katmanı tabanlı bir DDoS algılama ve önleme mekanizması olan MiddleModule sistemi sunulmaktadır. Bunun yanı sıra, kapsamlı veri katmanı savunma mekanizmaları oluşturmaya yönelik çeşitli sistem ihtiyaçları ifade edilmektedir.

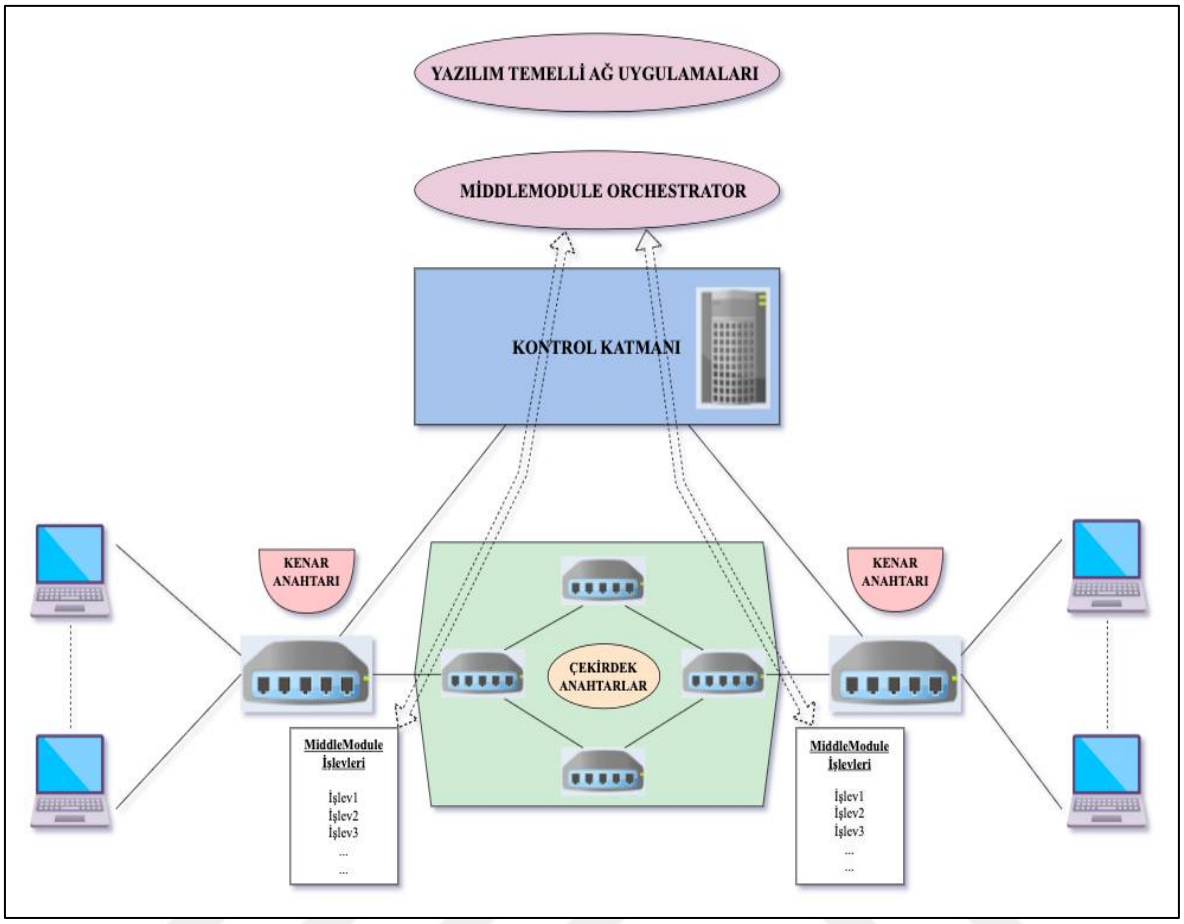
2.2.1. Mimariye genel bakış

Standart bir SDN’nin veri katmanı, ağ üzerinde gereken bağlantıyı sağlamak için birbiriyle ilişkili OpenFlow anahtarları içerir. Şekil 2.3 üzerinde gösterildiği gibi, bu anahtarlar kenar anahtarları ve çekirdek anahtarlar olarak kategorize edilebilir. Çekirdek anahtarlar ise veri katmanı ağının merkezinde yer almakla birlikte ağdaki diğer anahtarlara bağlılık gösterirler.



Şekil 2.3. Standart yazılım temelli ağ

MiddleModule sistemine yönelik olarak Şekil 2.4 üzerinde olduğu gibi, kenar anahtarlarda algılama ve azaltma işlemleri uygulanmaktayken, çekirdek anahtarlar herhangi bir değişiklik yapılmadan kullanılmaktadır. Ayrıca, denetleyici kapsamında kenar anahtarları içinde barındıran MiddleModule blokları ile denetleyicinin iletişimi açısından kullanılan bir Orchestrator Modülü uygulanmaktadır.



Şekil 2.4. MiddleModule sistemi kapsamında yazılım temelli ağ

Herhangi bir paket, MiddleModule kullanan bir SDN'ye girmesiyle beraber ağ geçidinde, yani kenar anahtarında analiz edilmektedir. Bundan dolayı, az miktarda kaynak ve basit algılama işlemleri kullanılarak, ağ geçidi noktalarında DDoS etkinlikleri hızlı ve doğru bir şekilde saptanabilmektedir. Bu yöntemin kullanılmasına yönelik olarak kötü amaçlı paketler diğer ağ bileşenlerinin kaynaklarını işgal etmemektedir.

MiddleModule sistemine dair iki önemli husus bulunur. Öncelikle bu sistemde denetleyici üzerinde çalışan bir Orchestrator Modülü olmasına karşın MiddleModule sistemi veri katmanı tabanlı bir savunma sistemi olmakla beraber Orchestrator Modülü algılama ve azaltma işlemleri için kullanılmamaktadır. Kenar anahtarlar üzerinde yer alan DDoS'un algılama işlemlerinin sorunsuz bir şekilde Orchestrator Modülünden gelen herhangi bir periyodik mesaja ya da komuta gereksinim duymamaktadır. MiddleModule kapsamındaki işlevler yalnız başına çalışmaktadır. Orchestrator Modülü, ağ operatörüne MiddleModule işlevlerini istenen şekilde yapılandırma ya da bunları etkinleştirme ve devre dışı bırakma

yeteneği sağlar. SDN kapsamında denetleyicinin paket bırakma kuralları da dahil olmak üzere ağdaki anahtarlar tarafından uygulanan herhangi bir iletme durumundan haberdar olması önem teşkil eder. Bundan dolayı, bir uç anahtarda yer alan MiddleModule işlevi, kötü amaçlı bir etkinlik algıladığında ve bir önleme kuralı kullanmaya karar verdiğinde, Orchestrator Modülünü kullanarak denetleyiciyi bilgilendirir. Denetleyici bu önleme kuralını bir şekilde değiştirmek isterse, Orchestrator Modülü aracılığıyla uygun mesajlar göndererek değiştirebilir. İkinci olarak, bu sistem merkezi dağıtılmış bir tespit/azaltma tekniği sağladığından, tespit düğümlerinin ağ boyunca geniş bir şekilde dağıtılması gerekir. Bu sebeple, bu dağılımı en üst katmana çıkarmak için MiddleModule işlevlerini kenar anahtarlarında uygulanması önerilir. Bu nedenle, en iyi performansı elde etmek ve sistemden en iyi şekilde faydalanmak için, ana bilgisayarlara doğrudan bağlı uç anahtarlarda MiddleModule işlevlerinin uygulanması gerekmektedir. Bu tez çalışmasında Şekil 2.4 üzerinde olduğu gibi uç anahtarlar ile tüm ana bilgisayarlar arasında doğrudan bağlantılar ele alınmış ve en iyi performansı elde etmek için MiddleModule bu şekilde uygulanmıştır.

2.2.2. Sistem ihtiyaçları

Sistem mimarisini ve çerçevesini kapsamlı şekilde açıklamadan önce; SDN mimarisine yönelik veri katmanı tabanlı DDoS savunma sistemi açısından sistem tasarlanırken göz önünde bulundurulması gereken noktalar; SDN mimarisinin ihtiyaçları, veri katmanı cihazlarının nitelikleri, DDoS saldırılarının özellikleri ve temel ağ güvenliği yaklaşımları şeklinde belirtilmektedir. Literatürde veri katmanı tabanlı DDoS savunma sistemleri için bu noktalar bir araya getirilip net bir şekilde belirtilmemiştir. Bu tezde bu noktaların altını çizilmekle beraber herhangi bir veri katmanı tabanlı DDoS savunma sisteminin aşağıda listelenen faktörler dikkate alınarak tasarlanması önerilmektedir:

Algoritmaların karmaşıklığı

Veri katmanı tabanlı bir savunma sistemi tasarlanırken tespit ve önleme algoritmalarının dikkatli bir şekilde uygulanması gerekir. Karmaşık algoritmalar yüksek savunma gücü sağlamasına ve çok sayıda saldırıyı önlemesine karşın bu tür algoritmalar, veri katmanı cihazları açısından ciddi sorunlara sebebiyet verebilmektedir. Veri katmanı cihazlarının sınırlı kaynakları bulunmaktadır. Bundan dolayı, karmaşık algoritmalar kullanılırsa CPU problem yaşayacaktır. Bir veri katmanı cihazına gelen trafik oranı yüksek olmakla beraber

veri katmanı cihazlarında karmaşık algoritmalar uygulanırsa, gelen trafiği analiz etmek için gereken işlem süresi fazla olabilmektedir. Bu durum yüksek paket iletme gecikmelerine ve hatta paket düşüşlerine sebep olabilmektedir. Bundan dolayı karmaşıklık düzeyi az olan algılama ve önleme algoritmalarının seçilmesi gerekmektedir.

Savunma sisteminin uygulama tarzı

SDN kapsamında veri katmanı tabanlı bir DDoS savunma sistemi, seri veya paralel bağlı anahtarda uygulanabilmektedir. Seri mod açısından uygulandığında, kötü amaçlı bir paket, iletilmeden önce kötü amaçlı olarak etiketlendiği anda bırakılmaktadır. Bu durum ise savunma sisteminin algılama oranını artırmaktadır. Ancak bütün paketlerin iletilmeden önce işlenmesi gerektiğinden, paket iletme gecikmesi önemli ölçüde artmaktadır. Savunma sistemi iletme devresine paralel olarak uygulandığında; alınan paket kötü amaçlı olarak etiketlense bile bu belirli paket yerine kötü amaçlı olarak etiketlenen paket zaten iletildiği için bir sonraki alınan paket bırakılabilir. Bu nedenle, algılama oranını biraz düşürerek savunma sistemi tarafından birkaç kötü amaçlı paket kaçırılsa da ek paket yönlendirme gecikmesi neredeyse hiç olmayacaktır.

Çok konumlu savunma yaklaşımı

Çok konumlu savunma yaklaşımı, ağdaki farklı konumlara farklı savunma algoritmalarının yerleştirilmesi veya çalıştırılması anlamına gelmektedir. Veri katmanı tabanlı bir DDoS savunma sistemi çerçevesinde bütün DDoS saldırı türlerini tespit etmek zor bir problem olabilmektedir. Bunun nedeni, birkaç saldırı türü vardır ve bazı saldırılar kaynak tarafından tespit edilebilirken, diğerleri yalnızca ağ içinde veya hedef tarafından doğru bir şekilde tespit edilebilmesidir. Örneğin kaynak tarafından bir protokol istismar saldırısı kolaylıkla tespit edilebilirken, bir yansıma saldırısı hiç tespit edilememektedir. Bir veri katmanı cihazı, kendi başına ağın genel bir görünümüne sahip değildir ve diğer anahtarlardan geçen trafiğin farkında değildir. Bu sebeple, DDoS saldırı türlerini tespit etmek için bilgilerin diğer anahtarlara gönderilmesi gerekmektedir. Bu durum ise sistemin iletişim yükünü ve sistemin karmaşıklığını artırmaktadır. Veri katmanı tabanlı bir DDoS algılama/azaltma sistemi tasarlanırken, çok konumlu savunma yaklaşımına duyulan ihtiyacın dikkate alınması gerekmektedir.

İzin verilen veri hızı

Günümüzde, tek bir ana bilgisayar tarafından üretilen trafik, gelişmiş ağ teknolojilerinden dolayı yoğunluk göstermektedir. Bunun yanı sıra DDoS saldırılarında kötü amaçlı bir ana bilgisayarın temel amaçlarından biri de olabildiğince çok paket üretmektir. Modern DDoS savunma sitemleri, önceki yöntemlerle karşılaştırıldığında çok daha yüksek veri hızlarını işleyebilmektedir. Bundan dolayı, bir savunma sistemi tasarlanırken izin verilen maksimum kötü amaçlı trafik oranı dikkate alınmalı ve bu savunma sistemleri yüksek veri hızları kullanılarak test edilmeli ve değerlendirme altına alınmalıdır.

Denetleyici ile iletişim

SDN kapsamında ağın en önemli kısımlarından birisi yönlendirme açısından denetleyicidir. Yönlendirme işlemine ilişkin herhangi bir işlem, denetleyici ile beraber ele alınmalıdır. Veri katmanı tabanlı DDoS açısından algılama ve önleme yöntemleri, bir veri katmanı cihazının kötü amaçlı etkinliği algıladığını ve kendi başına hafiflettiğini öne sürmektedir. Fakat veri katmanı cihazları kontrol cihazını tespit ve önleme kararları hakkında bilgilendirmelidir. Bunun yanı sıra, SDN çerçevesindeki bir ağ operatörü, denetleyici aracılığıyla tüm ağı kontrol edebilir ve yapılandırabilir. Bu doğrultuda, veri katmanı tabanlı bir savunma sistemi de denetleyici üzerinden yapılandırılabilir olmalıdır.

Hedefli dağıtılmış hizmet reddi saldırı türleri

Doğru hedef saldırı setinin seçilmesi, ağ güvenlik sistemleri açısından önem arz eder. Çok sayıda farklı saldırı türünü hedeflemek, özellikle veri katmanı tabanlı savunma sistemleri kapsamında iyi bir fikir değildir. Çünkü sistemin karmaşıklığını önemli ölçüde artırabilir veya etkinliği azaltabilir. Bunun yanı sıra, bir saldırı türünün sadece küçük bir alt kümesini hedeflemek de iyi bir fikir değildir. Bunun sebebi ise, ağ tasarımcıları genellikle belirli bir saldırı türünden kaçınmak için bir güvenlik sistemi kullanmaktadır. Kullanılan savunma sistemi, söz konusu saldırı türündeki bütün saldırıları kapsamıyorsa, ağ tasarımcısının başka bir savunma sistemi uygulaması gerekmektedir. Benzer savunma sistemlerini kullanmak, aynı cihazlarda çalışmak, aynı trafiği kontrol etmek ve benzer imzaları kontrol etmek genellikle düşük performansa sebebiyet verir. Veri katmanı tabanlı savunma sistemlerinin

bu şekilde kullanılması, veri katmanı cihazlarının kaynaklarını daha çok tüketmekte ve yüksek gecikmelere neden olmaktadır.

Kötü amaçlı etkinliğin kaynağını ve mağdurunu tespit etme

Etkili hafifletme için bir savunma sistemi, DDoS saldırısı algıladığında kötü amaçlı etkinliğin kaynağı ve mağduru hakkında bilgi sağlamalıdır. DDoS saldırısının türü de tespit edilmelidir. Bu bilgi, saldırıyı doğru bir şekilde azaltmak için değerlidir. Bir savunma sistemi bir saldırıyı algılar, ancak bu bilgiyi vermezse, saldırı trafiği doğru bir şekilde engellenememektedir.

MiddleModule sisteminde, sistemin karmaşıklığını en aza indirgemek ve tespit edilebilecek kötü amaçlı trafik oranını en üst düzeye çıkarmak için hafif algoritmalar kullanılarak algılama işlevselliği sağlanmaktadır. Paket işleme devresi, iletme gecikmesini en aza indirmek için iletme devresine paralel olarak uygulanır. Bir paket kenar anahtarına ulaştığında, orijinal paket hedefe doğru iletilirken algılama bloğuna yansıtılır.

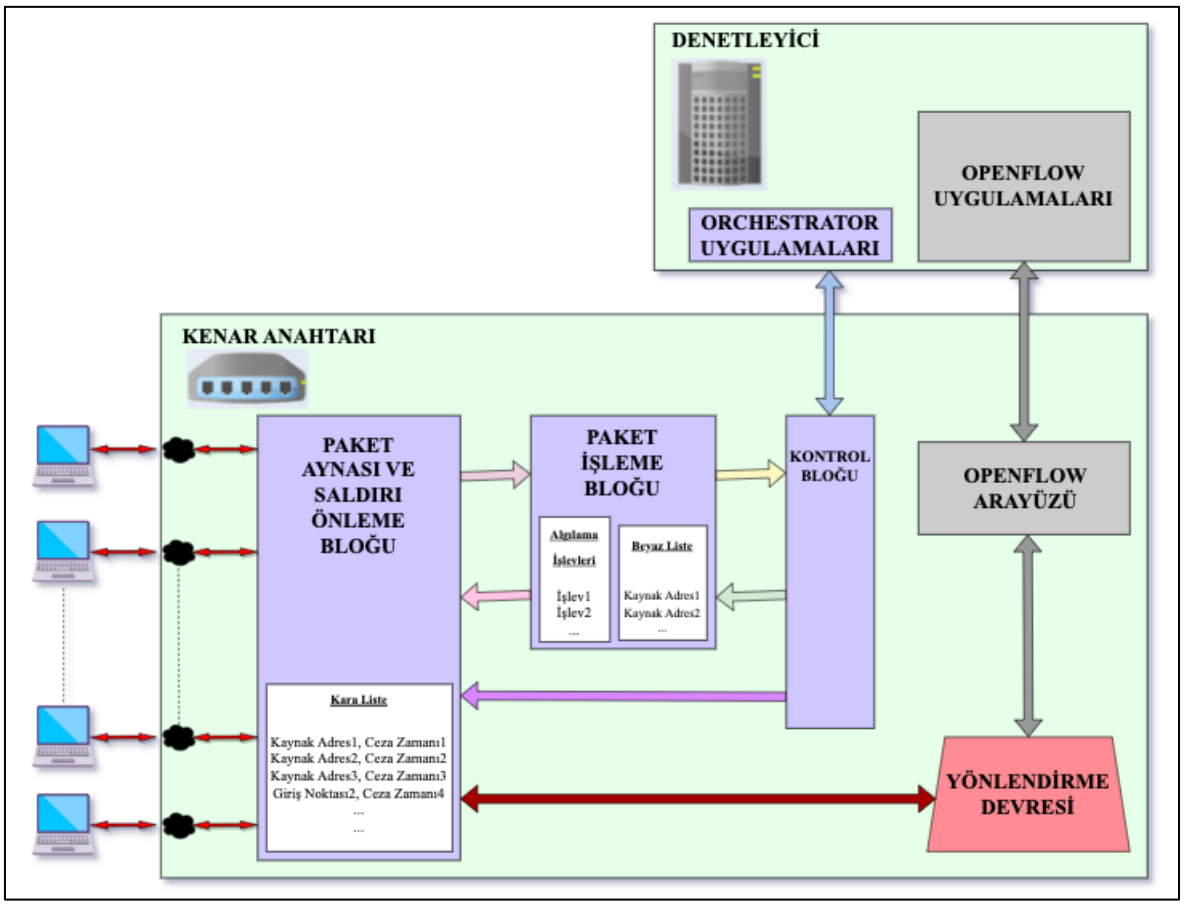
DDoS saldırısına karşı yüksek savunma gücü sağlamak için, MiddleModule sistemindeki çok konumlu savunma yaklaşımı kullanılmaktadır. Yansıma saldırıları hedef tarafından tespit edilirken, diğer saldırı türleri mesajların kaynağı tarafından tespit edilmektedir. Bu sistemde kenar anahtarları, gelen paketlere algılama işlevleri uygulamakla birlikte azaltma kurallarına karar vermektedir. Bunun yanı sıra kenar anahtarları, denetleyiciyi bu hafifletme kararları hakkında periyodik ya da düzensiz mesajlarla bilgilendirebilmektedir. Son olarak MiddleModule sistemi bir DDoS saldırısı tespit ettiğinde DDoS saldırı türünü ve söz konusu kötü amaçlı etkinliğin kaynağını ve mağdurunu bir çıktı olarak sunmaktadır.

2.2.3. Sistem yapısı

MiddleModule yapısı; izleme, tespit ve önleme amaçları için veri katmanında orkestrasyon ve konfigürasyon amaçları için kontrol katmanında bir blok gibi birkaç işlevsel bloğun uygulanmasını önermektedir.

Veri katmanında MiddleModule blokları

MiddleModule yapısı; izleme, algılama ve hafifletme yetenekleri sağlamak için uç anahtarlara “Paket Aynası ve Saldırı Önleme Bloğu” ve “Paket İşleme Bloğu” gibi birkaç blok eklenmesini önermektedir. Kenar anahtarların tespit ve önleme etkinliklerinde denetleyici ile iş birliği için kenar anahtarlara “Kontrol Bloğu” da eklenmektedir. Bundan dolayı her kenar anahtarına toplamda üç işlevsel blok eklenmektedir. Şekil 2.5 kapsamında bu blokların yerleşimi ve bağlantıları gösterilmektedir.



Şekil 2.5. MiddleModule blokları

Paket aynası ve saldırıyı önleme bloğu

“Paket Aynası ve Saldırıyı Önleme Bloğu”, gelen paketlerin “Paket İşleme Bloğu” kısmına yansıtılması kapsamında kaynak adresi ya da giriş bağlantı noktası kötü amaçlı listede bulunuyorsa gelen paketin bırakılması adına iki temel işlemden sorumludur. Bir paket bir kenar anahtarına girdiğinde doğrudan bu bloğa gider ve ilk olarak “Saldırı Önleme Alt

Bloğu" paketini gözden geçirmektedir. Bu paketin kaynak adresi kötü amaçlı listede bulunuyorsa, bu paket başka bir bloğa gönderilmeden bırakılmaktadır. Listede yoksa "Paket Aynası Alt Bloğu" paketini almaktadır. Bu alt blokta, orijinal paketin tamamı iletme devresine yönlendirilirken, paketin başlık kısmı "Paket İşleme Bloğu" aracılığıyla yansıtılmaktadır.

"Saldırı Önleme Alt Bloğu", kötü amaçlı yazılım tespit sonuçlarına göre uygun azaltma eyleminin gerçekleştirilmesinden sorumludur. Şekil 2.5'de görüldüğü gibi, MiddleModule sistemine yönelik olarak gelen paketi alan ilk blok "Saldırı Önleme Alt Bloğu" olmaktadır. Bu durumun sebebi, alınan paket zaten kayıtlı bir kötü niyetli kullanıcıdan geliyorsa, bu paketi tekrardan analiz edilmesine gerek duyulmamasıdır. Bu alt blok, önceden kaydedilmiş kötü amaçlı kullanıcıların kaynak adreslerini ya da giriş bağlantı noktalarını içeren bir kötü amaçlı listeye sahiptir.

"Paket İşleme Bloğu" herhangi bir zararlı etkinlik tespit ettiğinde, o etkinliğin kaynak adresini veya giriş bağlantı noktasını "Saldırı Önleme Alt Bloğu" tarafına göndermekle birlikte söz konusu alt blok onu kötü amaçlı listede saklamaktadır. Bunun yanı sıra, denetletici herhangi bir kötü amaçlı kaynak adresini bir kenar anahtarına gönderirse, "Kontrol Bloğu" bu bilgiyi alır ve "Saldırı Önleme Alt Bloğu" içerisine göndermektedir. Bu alt blok, kötü amaçlı adresi kötü amaçlı listesinde saklamaktadır.

"Paket Aynası Alt Bloğu", Şekil 2.5'de gösterildiği gibi bir kenar anahtarının giriş bağlantı noktalarının yakınında bulunmaktadır. Bir ana bilgisayardan veya bir çekirdek anahtarına yönelik herhangi bir giriş bağlantı noktasından bir kenar anahtarına bir paket geldiğinde ve "Saldırı Önleme Alt Bloğu" kapsamındaki paket düşmezse, "Paket Yansıtma Alt Bloğu" paketi alınmaktadır. Bu alt blokta, orijinal paketin tamamı iletme devresi aracılığıyla yönlendirilirken, paketin başlık kısmı "Paket İşleme Bloğu" aracılığıyla yansıtılır. "Paket İşleme Bloğu" içerisinde dağıtılan algılama algoritmaları, paketlerin sadece başlık kısımlarını analiz ederek kötü amaçlı etkinlikleri algılayabilmektedir. Bütün paketler yerine yalnızca başlık kısımlarını yansıtmak, anahtardaki iletişim ek yükünü en aza indirir ve "Paket İşleme Bloğu" açısından giriş sırası kullanımı en aza indirgenmektedir.

"Paket Aynası Alt Bloğu", genel MiddleModule sisteminin paket örnekleme bloğu olarak değerlendirilmektedir. Paket örnekleme için farklı metotlar bulunmakla beraber söz konusu

paketlerin bir olasılıkla örneklenmesi, paketlerin rastgele örneklenmesi veya gelen tüm paketlerin örneklenmesi gibi durumlar söz konusudur. Bu metotlarda örnekleme hızı arttıkça analiz edilen paket sayısı da artmaktadır. Sonuç olarak herhangi bir kötü amaçlı paketin kaçırılma olasılığı azalmaktadır. MiddleModule sistemindeki "Paket Aynası Alt Bloğu", kötü amaçlı algılama oranını en üst düzeye çıkarmak için gelen tüm paketleri örneklemektedir.

Paket işleme bloğu

"Paket İşleme Bloğu", uç anahtarlara algılama yeteneği sağlamaktadır. Bundan dolayı tespit işlemlerine yönelik olarak uç anahtarlara eklenen nitelik şeklinde ifade edilmektedir. Bu blok, "Paket Yansıtma Alt Bloğu" kısmından paket başlıklarını almakla birlikte birkaç Dağıtılmış Hizmet Reddi algılama algoritması kullanarak bunları işlemektedir. Bu nedenle, "Paket İşleme Bloğu", Giriş Kuyruğu ve Algılama Motoru gibi işleme ve depolama altyapısını içeren birkaç alt modül içermektedir. Herhangi bir paketin başlık kısmı, "Paket İşleme Bloğu" tarafından alındığı zaman Giriş Kuyruğu içerisinde depolanmaktadır. Bu, "ilk giren ilk çıkar" kuyruğu olarak tasarlanmıştır. Diğer çizelgeleme teknikleri kullanılabilse de kullanım kolaylığı nedeniyle bu tercih edilmektedir.

"Paket İşleme Bloğu" içerisinde akış tabanlı özellikler kaydedilmekte ve bu kayıtlar kötü amaçlı tespit algoritmalarında kullanılmaktadır. Akışlar, tespit algoritmasına bağlı olarak kaynak veya hedef adreslerine göre oluşturulmaktadır. Farklı algılama algoritmaları için farklı özellikler kaydedilmektedir.

Tespit işlemleri ana bilgisayarlara çok yakın yapıldığından, özellikler tablosunun boyutu çok küçük olabilir. Bunun birkaç nedeni vardır. İlk olarak, tespit algoritmaları özellikleri doğrudan ana bilgisayar bağlantılarına yönelik topladığından, doğru bir algılama yapmak için az sayıda özellik türü yeterli olmaktadır. İkincisi, toplanan özelliklerin uzunluğu veya pencere boyutu küçük olabilir ve mevcut kayıtların üzerine yeni kayıtlar yazılmaktadır. Bu kapsamda, algılama algoritmaları tarafından sadece yeni özellikler analiz edilmektedir. Ağ katmanında DDoS saldırılarının özellikleri nedeniyle küçük pencere boyutu, doğru algılama için yeterlidir. Herhangi bir ana bilgisayar böyle bir saldırı oluşturduğunda, son paket özelliklerinde değişiklikler yaratmakla birlikte bunlar daha sonra kolay bir şekilde tespit edilmektedir. "Paket İşleme Bloğu" kapsamında yanlış pozitif oranları en aza indirmek için

kötü amaçlı ve anormal olanı ayırt etmek önem taşımaktadır. MiddleModule sisteminde, mimari tedbirlerin yanı sıra ek bir yanlış pozitif önleme mekanizması kullanılmaktadır. Ağ oluşturmada, basit anormal etkinlikler kendilerini tekrar etme eğiliminde değilken, DDoS saldırıları uzun sürme eğilimini barındırmakta ve kendilerini tekrarlamaktadır.

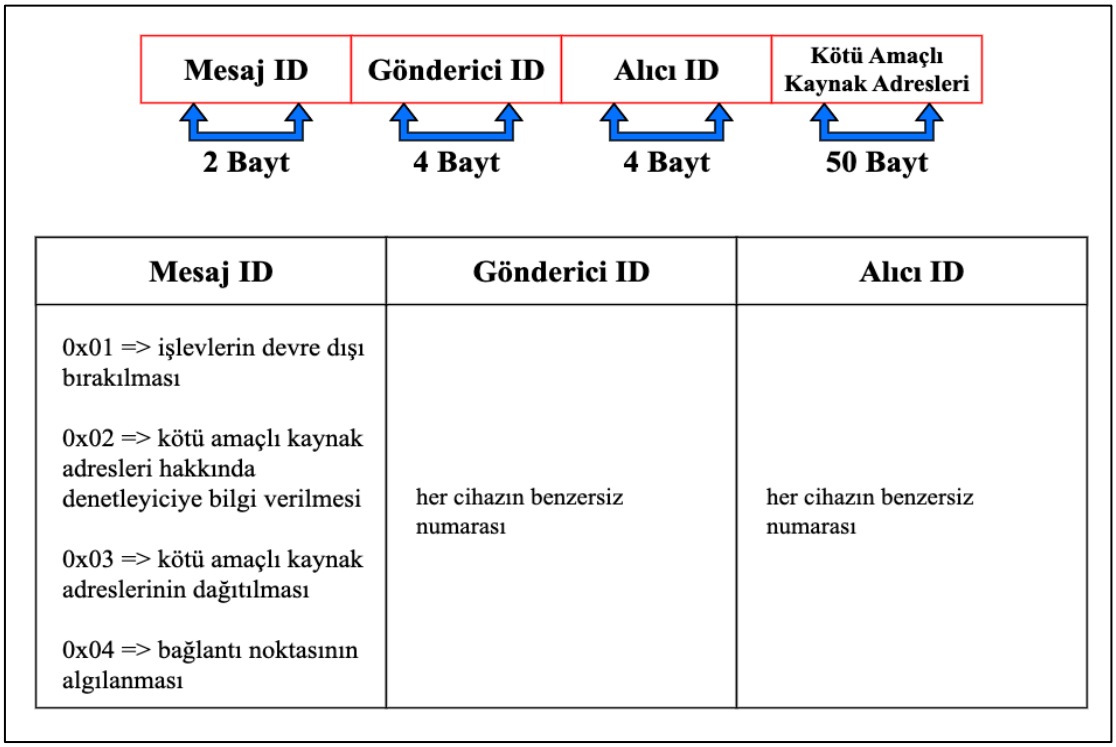
Kontrol bloğu

"Kontrol Bloğu", uç anahtarlarda çalışan MiddleModule blokları ile denetleyici üzerinde çalışan MiddleModule blokları arasındaki iletişim için kullanılmaktadır. Bu blok, MiddleModule sistemi için önem taşımaktadır. Çünkü sistem SDN'ler çerçevesinde tasarlanmıştır. Denetleyici, yönlendirme ile ilgili tüm kararlara karar vermektedir. Bir kenar anahtarı herhangi bir kötü amaçlı etkinlik algılayıp paketleri bırakmaya başladığında, denetleyicinin bu etkinliğin farkında olması gerekmektedir. Denetleyici herhangi bir kenar anahtarına bir beyaz liste veya kara liste tanımlayabilmelidir. Bunun yanı sıra denetleyici, ağdaki herhangi bir kenar anahtarının tüm MiddleModule işlemini etkinleştirebilmeli, devre dışı bırakabilmeli veya algılama algoritması değişkenlerini yapılandırabilmelidir.

"Kontrol Bloğu", denetleyiciye uygun mesajların gönderilmesinden ve denetleyiciden gelen mesajların alınmasından ve işlenmesinden sorumludur. "Kontrol Bloğu" nun önemli bir vizyonu bulunmaktadır. Bir kenar anahtarı tarafından bir yansıma saldırısı algılandığında, bu kenar anahtarının "Kontrol Bloğu", algılanan kötü amaçlı kaynak adreslerini diğer anahtarlara yayınlamak için kullanmaktadır. Bunun için kenar anahtar, bu blok üzerinden denetleyiciye bir mesaj gönderir ve denetleyici bu mesajı diğer kenar anahtarlara dağıtır. Böylece, kötü amaçlı ana bilgisayara çok yakın konumlanan uç anahtarlar, bu ana bilgisayarlardan gelen mesajları bırakmaya başlamakta ve kötü amaçlı trafik, ağa girmeden bırakılmaktadır.

Bağlantıyı sağlamak için, Şekil 2.6'da gösterildiği gibi, kenar anahtarları ile denetleyici arasında MiddleModule Mesaj Protokolü adı verilen bir iletişim protokolü tanımlanmaktadır. Mesajlar, Şekil 2.5 kapsamında görüldüğü gibi, Kontrol Bloğu ile denetleyici arasında özel bir iletişim kanalı üzerinden değiştirilmektedir. Mesaj türlerinin her birisi için önceden tanımlanan bir mesaj kimliği bulunmaktadır. Örneğin, bir anahtarın algılama/azaltma bloklarını devre dışı bırakmak için denetleyici, kimliği 0x01 olan bir mesaj göndermektedir. Bununla birlikte bağlantı noktası algılama mesajlarını oluşturmak ve

işlemek için de kullanılabilir. Bağlantı noktası algılama mesajları, kenar anahtarı bağlantı noktasına yönelik sınıflandırma için kullanılmaktadır. İstenen herhangi bir komut veya bilgi, Şekil 2.6'da görüldüğü gibi uygun mesaj kimliği ile alıcıya aktarılabilmektedir. MiddleModule mesajları, veri yükü boyutu 60 bayt olan İletişim Kontrol Protokolü (Transmission Control Protocol - TCP) mesajlarıdır. Bu protokol başka ihtiyaçlara göre değiştirilebilmekle birlikte mesajlaşmanın yükü göz ardı edilebilmektedir. MiddleModule mesaj iletişimi ek yükünün daha fazla analizi değerlendirme kısmında sağlanmaktadır.



Şekil 2.6. MiddleModule mesaj protokolü

Kontrol katmanında middlemodule blokları

Orchestrator bloğu

Orchestrator Bloğu, Şekil 2.6'da olduğu gibi uç anahtarların Kontrol Blokları ile iletişim kurması için kullanılır. Bu bloğun iletişim davranışı, Kontrol Bloğu bölümünde açıklanan durumlara benzerlik göstermektedir. Orchestrator Bloğu, denetleyicinin üzerinde çalışan bir uygulama olarak tasarlanmıştır. Bu blok, ağ operatörüne uç anahtarlarda çalışan MiddleModule savunma sistemini yapılandırma veya kontrol etme yeteneği sağlamaktadır. Bunun yanı sıra şebeke operatörü, MiddleModule sistemi tarafından verilen algılama

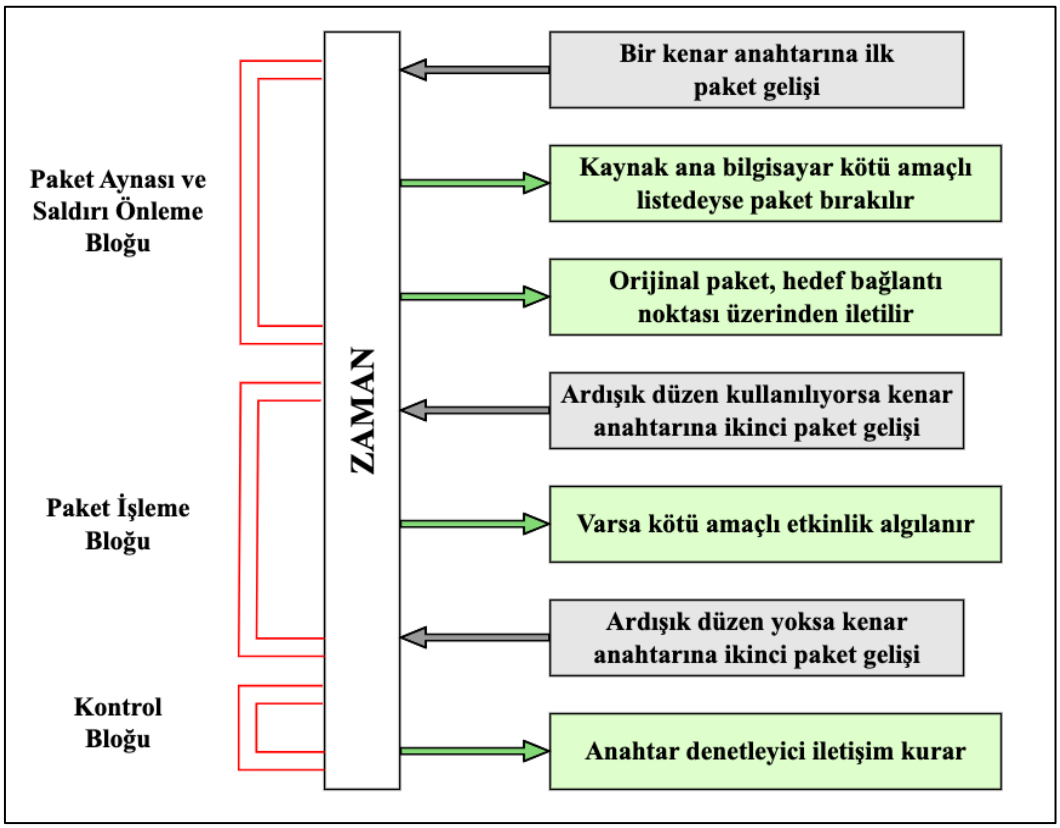
kararlarını gözlemleyebilmektedir. Ağ operatörü, sistemin performansını ayarlamak veya sistemin davranışını değiştirmek için algılama algoritmalarına eşik değerlerini yeniden atayabilir, ceza süresini değiştirebilir ve boşta kalma süresi ekleyebilir. Denetleyici bir şekilde aşırı yüklenirse, Orchestrator Bloğu sonlandırılabilir. Bu durum MiddleModule sisteminin genel performansını önemli ölçüde etkilememektedir.

MiddleModule işlemleri

Bir paket bir kenar anahtarına ulaştığında, Şekil 2.7'de gösterildiği gibi doğrudan Paket Aynası ve Saldırı Önleme Bloğu kısmına aktarılır. Saldırı Önleme Alt Bloğu, paketi alır ve kaynak adresin kötü amaçlı listede olup olmadığını kontrol eder. Listede varsa, paket daha fazla analiz yapılmadan bırakılır. Bu, anahtar kaynaklarını, denetleyici kaynaklarını ve kurban ana bilgisayarı kötü amaçlı etkinliklerden koruyacaktır. Ayrıca, bu paketin daha ayrıntılı analizi için hiçbir MiddleModule kötü amaçlı tespit kaynağı kullanılmaz. Paketin kaynak ana bilgisayarı kötü amaçlı listede değilse, Paket Aynası Alt Bloğu paketi alır ve Paket İşleme Bloğu aracılığıyla paketin başlık kısmını yansıtır ve paketin tamamını Yönlendirme Devresi aracılığıyla gönderir. Sistemde organizasyon kullanılıyorsa, Paket Aynası ve Saldırı Önleme Bloğu boşta kaldığından, sistem ağdan yeni bir paket alabilir ve işleyebilir. Yönlendirme Devresi, anahtarın OpenFlow ve Yönlendirme bileşenlerinde MiddleModule sistemi herhangi bir değişiklik yapmadığı için standart bir OpenFlow anahtarında olduğu gibi çalışır.

Paketin başlık kısmı Paket İşleme Bloğuna ulaştığında, Giriş Kuyruğunda depolanır. Paket İşleme Bloğundaki Tespit Motoru her boşta kaldığında, Giriş Kuyruğundaki ilk giriş kuyruğundan çıkarılır ve Tespit Motoru girişi, başlık kısmını alır. Önce, Algılama Motoru giriş için karşılık gelen akışı bulur, ardından Özellikler Tablosunu kontrol eder ve alınan başlığın ilgili niteliklerini alır. Bu özellikler, Tespit Algoritmalarında kullanılan akış için alınan paketlerin önceki geçmişine dayanan listeler ve değerler içermektedir. Kaydedilen özelliklerle ilgili ayrıntılı açıklamalar, tezin Algılama Algoritmaları bölümünde bulunabilir. Algılama Motoru, birkaç hafif Algılama Algoritması çalıştırır ve bunlardan herhangi biri kötü amaçlı etkinlik tespit ederse, paketin kaynak adresini kara listeye kaydeder. Kaynak ana bilgisayarın adresi zaten kara liste tablosundaysa, Tespit Motoru bu adresi Saldırı Önleme alt bloğuna göndererek başka bir ceza süresi için Kötü Amaçlı Listede saklamasını sağlar ve bu adresi kara liste tablosundan kaldırır. Algılama Motorundaki Özellikler Tablosu

en son özelliklerle güncellenir. Kötü amaçlı bir etkinlik tespit edilirse, tespit kararı Kontrol Bloğuna da iletilir. Bu blok daha sonra bu kötü amaçlı etkinliğin kaynak adresini denetleyiciye gönderebilir ve denetleyici bunu tüm uç anahtarlara dağıtabilir.



Şekil 2.7. MiddleModule işlemlerine yönelik diyagram

2.3. Tespit Algoritmaları

MiddleModule sisteminde, Ağ/Aktarım katmanında DDoS saldırılarını tespit etmek için, uç anahtarların Paket İşleme Blokları içinde çeşitli tespit algoritmaları uygulanmaktadır. Başlıca beş tür Ağ/Aktarım katmanında DDoS Saldırısı vardır. Bu bölümde, aşağıdaki algılama algoritmalarının ayrıntılı bir açıklaması verilmektedir. Ağ/Aktarım katmanındaki DDoS saldırı türünün iyi bilinen örneklerine karşı aşağıdaki gibi algılama algoritmaları sağlanmaktadır;

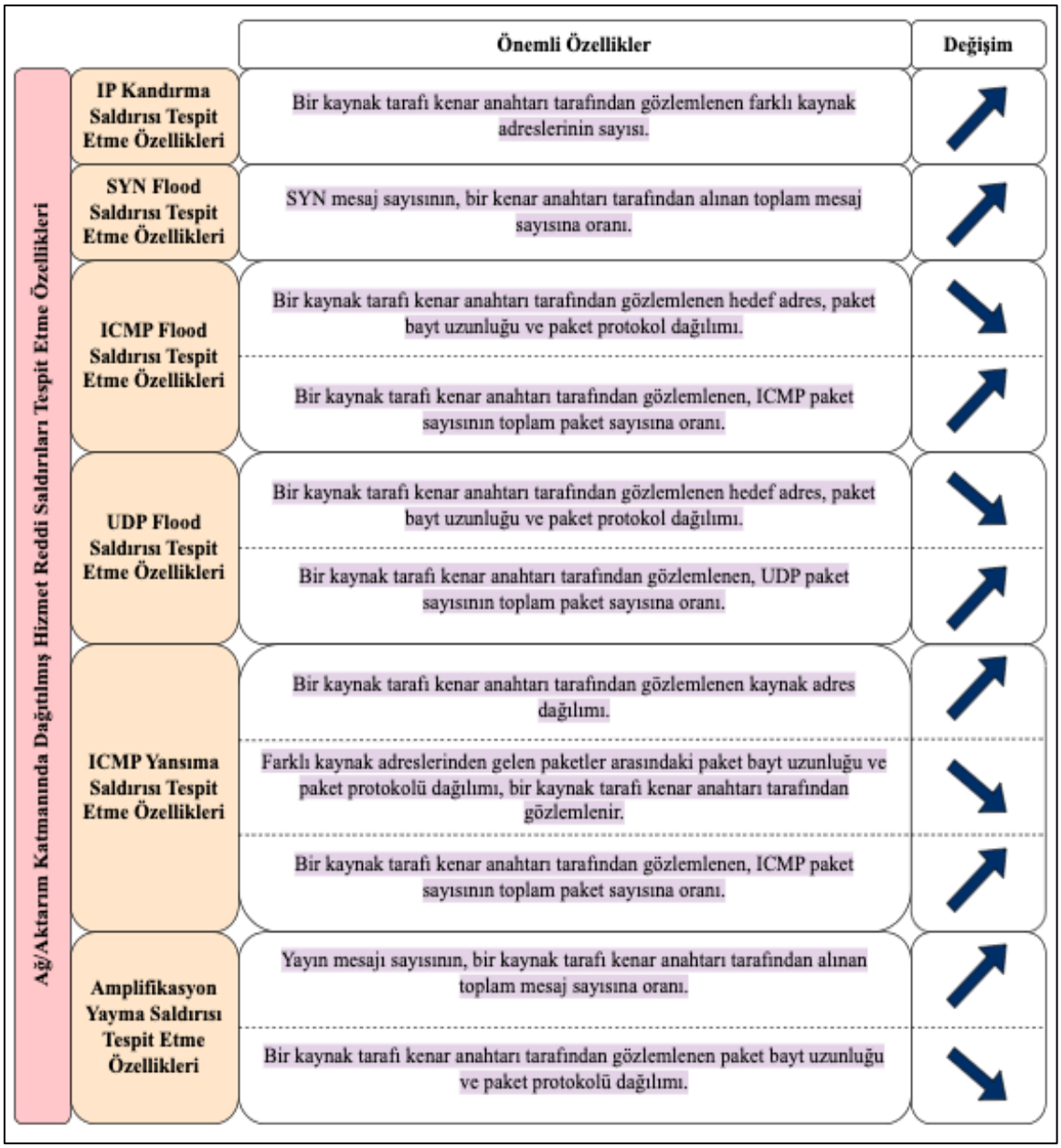
- IP Kandırma Saldırı Türü: Kaynak IP Kandırma Algılama algoritması,
- Protokol İstismar Saldırı Türü: SYN Flood Tespit Etme algoritması,
- Flood Saldırı Türü: ICMP Flood ve UDP Flood Tespit Etme algoritması,

- Yansıma Saldırı Türü: ICMP Yansıma Algılama algoritması,
- Amplifikasyon Saldırı Türü: Yayın Amplifikasyon Algılama algoritması.

2.3.1. MiddleModule sistemi açısından tespit algoritmaları

Ağ/Aktarım katmanındaki DDoS saldırıları, onları diğer ağ olaylarından ayıran iki kritik özelliğe sahiptir. İlk olarak, bu saldırılar büyük ağ trafiğine sebebiyet vermekle beraber trafik hacmini önemli ölçüde artırır. İkincisi, ağdaki paketlerin çoğu, bazı paket özellikleri bakımından birbirine çok benzer hale gelmektedir. Bu sebeple bazı paket özelliklerinin varyasyonu belirgin şekilde azalır veya artar. Organize ve kitlesel bir DDoS saldırısında bu değişiklikler tüm ağ boyunca tanınmaktayken, daha küçük bir saldırıda; en azından sorumlu kötü amaçlı ana bilgisayarın iletişim kanalında tespit edilmektedir. Bir saldırı, trafikte böyle bir değişikliğe neden olmayacak kadar küçükse, bu saldırı SDN'ye yönelik herhangi bir tehdit oluşturmaz. Dolayısıyla bu çalışmanın kapsamında değildir. Örneğin, bir Dağıtılmış Hizmet Reddi SYN Flood saldırısı esnasında kötü amaçlı ana bilgisayarın iletişim kanalında ölçülen SYN paket hacmi ve SYN paket sayısının tüm paket sayısına oranı gözle görülür şekilde artmaktadır. Bunun yanı sıra reflektörlerle yapılan organize bir DDoS İnternet Kontrol Mesajı Protokolü (Internet Control Message Protocol - ICMP) Yansıma saldırısı sırasında ağdaki ICMP paketlerinin hacmi artarken, ağdaki paketlerin hedef adreslerinin, paket bayt uzunluklarının ve paket protokollerinin varyasyonları azalmaktadır. Bu sebeple, Ağ/Aktarım katmanındaki DDoS algılama algoritmalarında, ağ üzerinde bir DDoS saldırısı olup olmadığı tespit edilmekle beraber varsa kaynak ve kurban ana bilgisayar için bazı niteliklerde değişiklik aranmaktadır. Bu durum Şekil 2.8'de gösterilmektedir.

MiddleModule sisteminde algılama algoritmaları belli başlı yaklaşımların göz önünde bulundurulmasıyla tasarlanmaktadır. Sistem kapsamında sağlanan algılama algoritmalarına yönelik olarak Şekil 2.8'de gösterildiği gibi basit özellik çıkarma yöntemleri ve temel eşik tabanlı karşılaştırma teknikleri kullanılmaktadır. Dolayısıyla bunlar hafif algoritmalar şeklinde değerlendirilmektedir. Hafif algoritmalar olsalar bile; veri katmanı cihazlarında uygulandıkları, DDoS saldırıları için karakteristik özelliklerde spesifik değişiklikler aradıkları ve bağlı ana bilgisayarlardan alınan her paketi analiz ettikleri için olağanüstü algılama performansları sağlamaktadır. Bunun yanı sıra, çok konumlu savunma yaklaşımı kullanmak, bu tür hafif algılama algoritmalarını kullanarak yüksek algılama performansı elde etmede kritik öneme sahiptir.



Şekil 2.8. Ağ/Aktarım katmanındaki DDoS saldırıları tespit etme özellikleri

Hafif teknikler açısından yanlış pozitifler elbette büyük bir sorun olabilir. Ancak, yanlış pozitif oranlarını en aza indirmek için, MiddleModule sistemine başka yetenekler de eklenmiştir. Kullanılan eşik değerleri hem anormal hem de normal trafik özellikleri dikkate alınarak seçilir. Aslında, yanlış pozitifler açısından güvenli tarafta olmak için uygulanan eşik değerler, hesaplanan eşik değerler üzerinde bir marjla seçilir. MiddleModule sistemi DDoS saldırılarını ihmal edilebilir yanlış pozitif oranlarla kolayca tespit etmektedir.

MiddleModule sisteminde, farklı Ağ/Aktarım katmanındaki DDoS saldırı türlerini tespit etmek için birkaç farklı saldırı tespit algoritması bulunmaktadır. Bu algoritmalar, kaynak tarafı algoritma seti ve hedef tarafı algoritma seti olmak üzere iki farklı kategoriye ayrılmaktadır. Bu setlerin her ikisi de tüm kenar anahtarlarında uygulansa da paketin kenar anahtarına gelen bağlantı noktasına göre, alınan bir pakete yalnızca bir set uygulanır. Bir anahtarın giriş bağlantı noktaları, ana bilgisayar bağlantı noktaları ve ağ bağlantı noktaları olarak sınıflama yapılmaktadır. Bir kenar anahtarının giriş bağlantı noktası doğrudan bir ana bilgisayara bağlıysa, bu bağlantı noktasına "ana bilgisayar bağlantı noktası" denir. Ayrıca, bir uç anahtarın giriş bağlantı noktası başka bir anahtara bağlıysa, bu bağlantı noktasına "ağ bağlantı noktası" denir. Bir ana bilgisayar bağlantı noktasından bir paket alındığında, kenar anahtarı kaynak tarafı algoritma setini uygulamaktadır. Bir ağ bağlantı noktasından bir paket alınırsa kenar anahtarı hedef tarafı algoritma setini uygulamaktadır. Yansıma Saldırısı Tespit Algoritmaları hedef tarafı algoritmalar setini, diğer saldırı tespit algoritmaları ise kaynak tarafı algoritmalar setini oluşturmaktadır. MiddleModule sisteminde, bir ana bilgisayar kötü amaçlı bir ana bilgisayarsa, bu ana bilgisayar tarafından üretilen trafiğin çoğunun o kötü amaçlı etkinliğe ait olduğu ifade edilmektedir. Ağ/Aktarım katmanında DDoS saldırılarının özellikleri dikkate alındığında geçerli bir ifadedir. Bildiğimiz kadarıyla, bu varsayım literatürde bulunan benzer çalışmaların çoğunda kullanılmaktadır (Boite vd., 2017; Chen, Junuthula, Siddhrau, Xu ve Chao, 2017).

Bir kenar anahtarının ana bilgisayar bağlantı noktalarını ve ağ bağlantı noktalarını algılamak için MiddleModule sistemine yönelik olarak MiddleModule Mesaj Protokolü kapsamında tanımlanan bağlantı noktası algılama mesajı kullanılmaktadır. Ağın başlatılması sırasında her kenar anahtarı, çekirdek anahtarlar üzerinden diğer kenar anahtarlarına iletilen bağlantı noktası algılama mesajı üretmektedir. Son olarak, tüm kenar anahtarları, ağ bağlantı noktaları üzerinden bu tür bağlantı noktası algılama mesajını almakta ve bu uç anahtarın geri kalan bağlantı noktaları ana bilgisayar bağlantı noktaları olarak etiketlenmektedir. MiddleModule sistemi, nispeten küçük ağlar üzerinde uygulandığı için tercih edilmektedir. Ağ büyükse, iletişim yükünü en aza indirmek için farklı yaklaşımlar kullanılabilir. Örneğin, bu bilgi denetleyici tarafından uç anahtarlara global ağ çapında görünüm kullanılarak sağlanabilir.

2.3.2. Protokol istismar saldırısı tespit algoritmaları

Protokol İstismar Saldırısı kapsamında kötü amaçlı bir ana bilgisayar, kurban ana bilgisayarın kaynaklarını tüketmek için iletişim protokollerinin bazı özelliklerini kötüye kullanan paketler üretmektedir. Bu açıdan, SYN Flood saldırısı, protokol istismar saldırı türünün bir örneği olarak ele alınarak Çizelge 2.1'de açıklandığı gibi, SYN Flood saldırılarına karşı bir tespit algoritması sağlanmıştır. SYN Flood saldırısını tespit etmek için, paketlerin kaynak tarafında farklı kaynak IP adreslerine sahip her bir ana bilgisayar için SYN paket sayısının toplam paket sayısına oranı hesaplanmaktadır. Dolayısıyla bu algoritma, mesaj tipindeki değişimi analiz etmektedir.

Normal TCP trafiği için, SYN mesajlarının sayısı ile toplam mesaj sayısının oranı bir eşikten küçük olmalıdır. Eşik, ağ anormalliklerini kapsayacak şekilde dikkatlice seçilirse, bu oran eşiği aştığında, bir ana bilgisayardan bir kurban sunucu aracılığıyla bir SYN Flooding etkinliği olduğu sonucuna varılabilir.

SYN Flood algılama algoritmasında, bir kenar anahtarına bir paket (p) ulaştığında, paketin bir ana bilgisayar bağlantı noktasından mı yoksa bir ağ bağlantı noktasından mı geldiğini görmek için paketin gelen bağlantı noktası (i) kontrol edilir. Bir ana bilgisayar bağlantı noktasından geliyorsa, p paketi, $lastEntry_s$ dizin numarasıyla $pcktWindow_s$ adlı bir listede depolanır. Farklı kaynak IP adresleri için farklı listeler kullanılmaktadır. Liste doluysa, yeni paketin üzerine en eski paket yazılır. Bu nedenle, liste her zaman algılama performansı için kritik olan en son paketleri içermektedir. Paket saklandıktan sonra, $pcktWindow_s$ listesindeki toplam paket sayısı olan t_s ve $pcktWindow_s$ listesindeki toplam SYN paket sayısı olan s_s hesaplanmaktadır. Ardından, kötü amaçlı etkinlik tespiti için birkaç karşılaştırma işlemi gerçekleştirilmektedir. İlk olarak, $pcktWindow_s$ listesindeki ve *windowSize* kapsamındaki toplam paket sayısının oranı hesaplanır ve gelişmemiş kararlardan kaçınmak için bir *saturationThreshold* ile karşılaştırılır. Yeterli sayıda paket varsa, SYN paketlerinin sayısının ve $pcktWindow_s$ listesindeki toplam paket sayısının oranı *synFloodThreshold* değeriyle karşılaştırılarak SYN paketlerinin sayısında çok büyük bir artış olup olmadığı kontrol edilmektedir. Bu durumda sorumlu kaynak IP adresine yönelik bir ceza süresi için kenar anahtarındaki bir kara liste tablosuna eklenir ve ilişkili ana bilgisayar için şimdiye kadar toplanan tüm istatistikler temizlenir.

Çizelge 2.1. SYN flood tespit algoritması

1	Yeni paket (p) girişi	
2	s <= Paketin IP adresi kaynağı	
3	i <= Paketin giriş anahtarı bağlantı noktası	
4	Eğer i bir ana bilgisayar bağlantı noktası ise,	
5		$pcktWindow_s[lastEntry_s]$ <= p
6	Eğer $lastEntry_s$ < *windowSize* ise,	
7		$lastEntry_s$ artar.
8	veya	
9		$lastEntry_s$ <= 0
10	t_s <= $pcktWindow_s$ listesindeki toplam paket sayısı	
11	s_s <= $pcktWindow_s$ listesindeki toplam SYN paket sayısı	
12	Eğer (s_s / t_s > *synFloodThreshold*) ve (t_s / *windowSize* < *saturationThreshold*)	
13		p paketi kötü amaçlıdır
14		t_s <= 0
15		s_s <= 0

Algoritmaya yönelik olarak *synFloodThreshold* değerinin atanması, algoritmanın algılama performansı açısından önemli bir konudur. Doğru bir değer atamak için hem standart trafik senaryoları hem de anormallik senaryoları dikkate alınmalıdır. 3 yollu anlaşma, veri alışverişi ve FIN-ACK kısımlarını içeren standart bir TCP bağlantısında, SYN paket sayısının toplam paket sayısına oranı 1/5'ten azdır. Hatta ağ üzerinde TCP dışında farklı mesaj protokolleri varsa bu oran daha da küçülür. Bazı durumlarda, ağ anormallikleri gün yüzüne çıkabilir ve bu durumda toplam paket sayısına göre SYN paketlerinin sayısı artırabilir. Ancak bu anormallikler sırasında bile bu oranın 1/2 gibi uzun süre bir eşikten büyük olmaması gerekir. Bu nedenle, beklenen normal trafik ve olası ağ anormallikleri dikkate alınarak ağ topolojilerine göre optimum bir *synFloodThreshold* değeri bulunabilir.

2.3.3. IP Kandırma saldırısı tespit algoritmaları

IP Kandırma Saldırısı açısından kötü amaçlı bir ana bilgisayar, sahte kaynak IP adreslerine sahip paketler oluşturur. Genel olarak, kaynak IP kaynak ayrı bir saldırı türü olarak sınıflandırılmamaktadır. Fakat bu çalışmada kaynak IP kandırma saldırısını ayrı bir Ağ/Aktarım katmanında saldırı türü olarak ele alınmaktadır. Çizelge 2.2'de gösterildiği gibi IP Kandırma Saldırılarına karşı bir algılama algoritması sunulmaktadır.

Paketlerin kaynak tarafında bu tür bir saldırıyı tespit etmek için gelen trafiğin kaynak IP adresleri izlenerek gelen paketlerin kaynak IP adreslerine yönelik varyasyondaki herhangi bir önemli artışı tespit eder. Tek bir ana bilgisayar bağlantı noktasının girişinden gelen farklı kaynak IP adreslerine sahip paket sayısı bir eşik değerinin üzerine çıkarsa, bu ana bilgisayar girişine bağlı IP Kandırma saldırganları kara listeye eklenmektedir.

IP Kandırma saldırısı algılama algoritmasında bir paket (p), bir kenar anahtarına ulaştığında, gelen bağlantı noktası (i) bir ana bilgisayar bağlantı noktası olup olmadığı kontrol edilir ve paket $lastEntry_s$ dizin numarasıyla $pcktWindow_i$ listesinde saklanır. Bu algoritmada her bir farklı anahtar girişi için farklı istatistikler ve farklı listeler saklanmaktadır. Bu nedenle, giriş bağlantı noktası katmanlı akışlar oluşturulur. Ardından, gelişmemiş kararlardan kaçınmak için $pcktWindow_i$ listesindeki ve *windowSize* üzerindeki toplam paket sayısının oranı *saturationThreshold* ile karşılaştırılır. Ayrıca $pcktWindow_i$ listesindeki paketlerin kaynak IP adresleri analiz edilir ve $pcktWindow_i$ listesinde saklanan farklı kaynak IP adreslerinin sayısı *spoofThreshold* ile karşılaştırılır.

IP adreslerinin sayısı *spoofThreshold* değerinden daha büyükse bu durum uç anahtarın tek bir giriş bağlantı noktasından gelen farklı kaynak IP adreslerine sahip çok fazla paket olduğu anlamına gelir. Bu algoritma, uç anahtarların ana bilgisayarlara çok yakın olduğu ve anahtarın bir giriş noktasına yalnızca bir ana bilgisayarın bağlı olduğu varsayılarak tasarlanmıştır. Dolayısıyla, her iki karşılaştırma da değerlerin eşikten daha büyük olduğunu gösteriyorsa, uç anahtarın bu giriş bağlantı noktasına bağlı ana bilgisayarın ağ üzerine IP Kandırma trafiği gönderdiği sonucuna varılır.

Kenar anahtarı, giriş bağlantı noktasını bir ceza süresi için kara liste tablosunda depolar, o bağlantı noktası için tutulan tüm istatistikleri siler ve izlemeyi yeniden başlatır. Bu ceza

süresi içinde aynı giriş portunda IP Kandırma faaliyeti tespit edilirse bu port kötü amaçlı listeye eklenir ve o girişten gelen paketler ceza süresi kadar düşürülür.

Çizelge 2.2. IP Kandırma tespit algoritması

1	Yeni paket (p) girişi
2	s <= Paketin IP adresi kaynağı
3	i <= Paketin giriş anahtarı bağlantı noktası
4	Eğer i bir ana bilgisayar bağlantı noktası ise,
5	$pcktWindow_i[lastEntry_i]$ <= p
6	Eğer $lastEntry_s$ < *windowSize* ise,
7	$lastEntry_i$ artar.
8	veya
9	$lastEntry_i$ <= 0
10	t_i <= $pcktWindow_i$ listesindeki toplam paket sayısı
11	$diffSrc_i$ <= $pcktWindow_s$ listesindeki toplam kaynak IP adresi paket sayısı
12	Eğer ($diffSrc_i$ > *spoofThreshold*) ve (t_i / *windowSize* > *saturationThreshold*)
13	p paketi kötü amaçlıdır
14	t_i <= 0
15	$açık\ (diffSrc_i)$

2.3.4. Flood Saldırısı tespit algoritmaları

Flood saldırı tipinde, kötü amaçlı bir ana bilgisayar, kaynaklarını ve iletişim bant genişliğini tüketmek için kurban ana bilgisayara doğru çok sayıda yedekli paket üretmekte ve göndermektedir. Çalışmamızda ICMP Flooding ve Kullanıcı Veri Bloğu Protokolü (User Datagram Protocol-UDP) Flooding saldırı tipi örnekleri ele alınmakta ve Çizelge 2.3'te açıklandığı gibi ICMP Flooding ve UDP Flooding saldırılarına karşı bir algılama algoritması ortaya koyulmaktadır.

Flood saldırısını tespit etmek için, tek bir kaynak ana bilgisayardan bir hedefe gönderilen paket sayısı kontrol edilmektedir. Bu kaynak ana bilgisayardan alınan paketlerin özellikleri de analiz edilmektedir. Bir Flood saldırısı sırasında, bir kaynak ana bilgisayar bir hedefe çok sayıda paket gönderir ve o kaynak ana bilgisayardan alınan paketlerin hedef IP adreslerindeki varyasyon önemli ölçüde azalmaktadır. Ayrıca, bu tür saldırılar sırasında, kaynak ana bilgisayardan alınan paketlerin çoğunun paket bayt uzunluğu ve paket protokolü aynı hale gelmektedir. Bu tür saldırılar sırasında paket bayt uzunluğu ve paket protokolü dışında başka benzerlikler gözlenebilse de kolaylık sağlamak için MiddleModule sisteminde diğer benzerlikler dikkate alınmamaktadır. Paket bayt uzunlukları ve paket protokolleri, bu algoritmada paket özellikleri olarak adlandırılmaktadır.

Flood saldırılarını tespit algoritmalarında bir p paketi, ana bilgisayar bağlantı noktasından bir kenar anahtarına ulaştığında, $pckWindow_s$ listesinde $lastEntry_s$ dizin numarasıyla depolanmaktadır. Listelemede farklı kaynak ana bilgisayarlar için farklı istatistikler ve listeler kullanılmaktadır. Paketi listede sakladıktan sonra, $lastEntry_s$ değeri, *windowSize* değerinden küçükse artırma uygulanmakta ve bununla birlikte $pckWindow_s$ listesindeki paket sayısının ve *windowSize*'ın oranı hesaplanır ve *saturationThreshold* ile karşılaştırma yapılmaktadır. Listede yeterli sayıda paket varsa, aynı hedef adrese sahip liste paketlerinin sayısı p $occDest_d$ olarak hesaplanmaktadır. Ek olarak, p ile aynı özelliklere sahip liste paketlerinin sayısı da $occProp_{props}$ şeklinde bulunmaktadır. Bu işlemlerin ardından $pckWindow_s$ listesindeki o$occDest_d$ ve toplam paket sayısına oranı hesaplanır. Hesaplanan oran, o kaynak konaktan gelen paketlerin hedef IP adreslerinin dağılımını kontrol etmek için *floodThreshold* değeri ile karşılaştırılır. Daha sonra $occProp_{props}$ değerinin $pckWindow_s$ listesindeki toplam paket sayısına oranı bulunarak o kaynak ana bilgisayardan alınan paketlerin özelliklerinin dağılımı hesaplanır. Bu oran *floodThreshold* değeri ile karşılaştırılır. Karşılaştırma sonuçları, o kaynak ana bilgisayardan taşma saldırısı olduğunu gösteriyorsa, ana bilgisayarın kaynak IP adresi kara liste tablosunda saklanır ve izlenen özellikler varsayılan değerlerine sıfırlanır ve izleme işlemi yeniden başlatılmaktadır.

Flood saldırısı tespit algoritmasında kritik bir detay bulunmaktadır. Bu detay kurban konak, gelen Flood paketlerine yanıt mesajları gönderirse, kötü amaçlı olarak da etiketlenebilmesi durumudur. Bu sorunu önlemek için, bir ana bilgisayar bağlantı noktasından ilk kez bir paket alındığında, algılama algoritması bunun ilk mesaj mı yoksa bir yanıt mesajı mı olduğunu

kontrol eder. Bunun için ana bilgisayar portundan bir paket alındığında, paketin hedef adresinin farklı hedef adres listesinde olup olmadığı, $occDest_d$ değerinin oluşturulup oluşturulmadığı araştırılır. Listede yoksa yansıma saldırısı algılama algoritmasından $diffSrc_s$ listesi, o paketin hedef adresinin listede olup olmadığına bakmak için kontrol edilir. Eğer öyleyse, paketin bir mesaja cevap olarak gönderildiği sonucuna varılır; bu nedenle taşma algılama algoritması pakete uygulanmaz.

Çizelge 2.3. Flood saldırısı tespit algoritması

1	Yeni paket (p) girişi
2	s <= Paketin IP adresi kaynağı
3	d <= Paketin IP adresi hedefi
4	i <= Paketin giriş anahtarı bağlantı noktası
5	*Prop* <= p özelliği
6	$diffDestAddrList_s$ <= farklı hedef IP adreslerinin listesi
7	Eğer i bir ana bilgisayar bağlantı noktası ise,
8	Eğer (d, $diffDestAddrList_s$ ve $diffSrc_s$ içerisinde) ise
9	depolanır.
10	$pcktWindow_s[lastEntry_s]$ <= p
11	Eğer $lastEntry_s$ < *windowSize* ise,
12	$lastEntry_s$ artar.
13	veya
14	$lastEntry_s$ <= 0
15	t_s <= $pcktWindow_s$ listesindeki toplam paket sayısı
16	$occProp_{props}$ <= $pcktWindow_s$ listesindeki toplam kaynak IP adresi paket sayısı
17	$occDest_d$<= $pcktWindow_s$ listesindeki hedefi d olan paket sayısı
18	Eğer (t_s / *windowSize* > *saturationThreshold*) biçimdeyse

Çizelge 2.3. (devam) Flood Saldırısı tespit algoritması

19	Eğer (o$occDest_d$ / t_s > *floodThreshold* ve $occProp_{props}$ / t_s > *floodThreshold) ise*
20	p paketi kötü amaçlıdır
21	t_s <= 0
22	o$occDest_d$ <= 0
23	$occProp_{props}$ <= 0

2.3.5. Amplifikasyon saldırı tespit algoritmaları

Amplifikasyon saldırı türünde, kötü amaçlı bir ana bilgisayar, kötü amaçlı trafik oluşturur ve bu trafikteki paketlerin bazı özelliklerini, kötü amaçlı etkinliğin etkilerini artırmak için değiştirir. Saldırgan bu saldırıyı kullanarak, saldırıyı oluşturmak için kurban ana bilgisayarın kendi bant genişliğinden daha fazla ağ bant genişliğini kullanır.

Çalışmamızda Broadcast Amplifikasyon Saldırıları, amplifikasyon tipi saldırıların bir örneği olarak ele alınmıştır. Kötü niyetli bir kullanıcı, kötü amaçlı mesajları kurban sunucuya yönlendirmek yerine ağ üzerinden yayınlar veya kötü niyetli bir kullanıcı, reflektörlere yansıma paketleri göndererek yanıtlarını yayınlamasını sağlar. Bu saldırının temel amacı, saldırının altta yatan ağ ve yasal kullanıcılar üzerindeki etkilerini artırmaktır. ICMP isteği gibi bir istek mesajı ağ üzerinden yayınlanırsa ve yasal kullanıcılar buna yanıt verirse, saldırının etkisi yıkıcı olmaktadır. MiddleModule sisteminde, yayın yükseltme saldırılarını önlemek için, Çizelge 2.4'te açıklandığı gibi hem yayın mesajlarının sayısını hem de paket özelliklerinin değişimini kontrol eden bir algoritma oluşturulmuştur.

Bu algılama algoritmasının çalışma biçimi şu biçimde ilerlemektedir; Bir paket ana bilgisayar bağlantı noktasından bir kenar anahtarına ulaştığında, $pckWindow_s$ listesinde $lastEntry_s$ dizin numarasıyla depolanır. $pckWindow_s$ Listesi ve *windowSize* paketlerinin toplam sayısının oranı hesaplanır ve bir *saturationThreshold* ile karşılaştırılır. Eğer $pckWindow_s$ listesinde doğru karar vermeye yetecek kadar paket varsa $pckWindow_s$listesindeki yayın mesajlarının sayısı bulunur. $pckWindow_s$ Listesindeki paket özellikleri, paket bayt uzunlukları ve paket protokolleri de kontrol edilir ve farklı paket

özelliklerinin sayısı hesaplanır. Daha sonra, yayın mesajlarının sayısı ve listedeki farklı paket özelliklerinin sayısı, listedeki toplam paket sayısıyla karşılaştırılır. Bu karşılaştırmalar bir yayın yükseltme saldırısı olduğunu gösteriyorsa, önce sorumlu kaynak ana bilgisayarın kaynak IP adresi kara liste tablosunda saklanır ve bu kaynağın tekrar kötü amaçlı olduğu tespit edilirse, kötü amaçlı listede saklanır. Bu algoritmada, kaynak tabanlı akışlar oluşturulur ve kaynak tabanlı her akış için paket izleme ve kötü amaçlı algılama ele alınır.

Çizelge 2.4. Amplifikasyon saldırı tespit algoritması

1	Yeni paket (*p*) girişi
2	*s* <= Paketin IP adresi kaynağı
3	*i* <= Paketin giriş anahtarı bağlantı noktası
4	*Prop* <= p özelliği
5	Eğer *i* bir ana bilgisayar bağlantı noktası ise,
6	$pcktWindow_s[lastEntry_s]$ <= p
7	Eğer $lastEntry_s$ < *windowSize* ise,
8	$lastEntry_s$ artar.
9	veya
10	$lastEntry_s$ <= 0
11	t_s <= $pcktWindow_s$ listesindeki toplam paket sayısı
12	b_s <= $pcktWindow_s$ listesindeki toplam yayın mesajı sayısı
13	$occProp_{props}$ <= $pcktWindow_s$ listesindeki prop özelliğine sahip paket sayısı
14	Eğer (t_s / *windowSize* > *saturationThreshold*) biçimdeyse
15	Eğer (b_s/ t_s> *amplificationThreshold* ve $occProp_{props}$ / t_s > *amplificationThreshold) ise*
16	p paketi kötü amaçlıdır
17	t_s<= 0
18	b_s <= 0

2.3.6. Yansıma saldırısı tespit algoritmaları

Yansıma Saldırısı kapsamında, kötü amaçlı bir ana bilgisayar, değiştirilmiş kaynak IP adresiyle reflektörlere birkaç istek mesajı göndermektedir. Kötü niyetli kullanıcı kendi IP adresini yazmak yerine, mesajların kaynak IP adresi kısmına kurban ana bilgisayarın IP adresini yazmakta ve buna bağlı olarak yansıtıcılardan gelen yanıtlar kurban ana bilgisayara gönderilir. Çalışmamızda ICMP Yansıma Saldırısı, yansıma saldırı tipinin bir örneği olarak ele alınmış ve Çizelge 2.5'te açıklandığı gibi ICMP Yansıma saldırısına karşı bir algılama algoritması sağlanmıştır.

Bu saldırı tipi, bir paketin kaynak tarafında tespit edilemez çünkü bu işbirlikçi bir saldırıdır ve birkaç ayrı sistem tarafından oluşturulmaktadır. MiddleModule sisteminde bu saldırı paketlerin hedef tarafında tespit edilir. Bir yansıma saldırısı sırasında, farklı kaynak adreslere sahip paketlerin sayısı kurban ana bilgisayarın yakınındaki kenar anahtarında artar, böylece kenar anahtarındaki kaynak IP adresi varyasyonu bir eşiğin üzerine çıkar. Ek olarak, yansıma saldırı trafiği aslında kötü amaçlı bir ana bilgisayardan alınan bir saldırı tetikleme mesajına yansıtıcı düğümlerden üretilen yanıt mesajlarıdır. Bu nedenle, yansıma saldırı paketleri, aynı paket bayt uzunluklarına ve paket protokollerine sahiptir. Bu mesajlarda başka birçok benzerlik olmasına rağmen, MiddleModule sisteminde basitlik için sadece bu iki özellik dikkate alınır ve bunlar paket özellikleri olarak adlandırılır. Bu nedenle, yansıma saldırısı sırasında, kurbanın ana bilgisayarının yanındaki kenar anahtarında, kaynak IP adresleri aynı olan paket özelliklerinin değişimi bir eşiğin altına düşmektedir.

Yansıma algılama algoritmasında, *p paketi* kenar anahtarına ulaştığında, gelen bağlantı noktası *i*'nin bir ağ bağlantı noktası olup olmadığı kontrol edilir ve eğer öyleyse, pakete yansıma saldırısı algılama algoritması uygulanır. Bu algoritmada, kaynak adres tabanlı akışların daha önce olduğu önceki algoritmalardan farklı olarak hedef adres tabanlı akışlar üretilir. Bir ağ portundan *p paketini* aldıktan sonra, paketi $pckWindow_d$ listesine kaydetmeden önce mevcut $pckWindow_d$ listesi kontrol edilir ve $pckWindow_d$listesindeki paketlerin farklı kaynak IP adresleri$diffSrc_d$ listesinde saklanır. $diffSrc_d$ Listesinde yer alan bu paketlerin özellikleri $diffSrcProp_d$ listesinde saklanır. *P paketi* daha sonra $LastEntry_d$ indeksi kullanılarak $pckWindow_d$ listesinde saklanır ve $LastEntry_d$, *windowSize*'dan küçükse artırılır. Daha sonra alınan paketin kaynak IP adresi olan s, $diffSrc_d$ listesinde bulunan IP adresleri ile karşılaştırılır ve bu listede yoksa $diffSrc_d$

listesine *s* eklenir ve alınan paketin özelliği $diffSrcProp_d$ listesine eklenir. $pckWindow_d$ Listesindeki toplam paket sayısının *windowSize*'a oranı *saturationThreshold* değeri ile karşılaştırılır. $diffSrcProp_d$ Listesinde yeterli paket varsa $diffSrc_d$ listesinin boyutunun $pckWindow_d$ listesindeki toplam paket sayısına oranı hesaplanır ve bu oran *refSrcThreshold* ile karşılaştırılır. Eşikten büyükse, hedef ana bilgisayara gönderilen farklı kaynak IP adreslerine sahip çok fazla paket olduğu anlamına gelir.

Son olarak, $diffSrcProp_d$ listesindeki paket özellikleri analiz edilir ve diğerleri ile aynı özelliklere sahip olan paketlerin sayısı bulunur; $diffSrcProp_d$ listesindeki aynı özelliklere sahip maksimum paket sayısı $maxNumbSameProp_d$ değerine atanır. $maxNumbSameProp_d$ Değerinin $diffSrc_d$ listesinin boyutuna oranı *refPropThreshold*'dan büyükse kaynak adresleri farklı ama özellikleri birbirine benzeyen çok fazla paket bulunduğu anlamına gelmektedir. Bu nedenle, bu koşulları sağlayan tüm paketler kötü amaçlı olarak kabul edilir ve bu paketlerin kaynak IP adresleri, denetleyici üzerinden diğer uç anahtarlara gönderilir. Bu nedenle, bu ana bilgisayarlardan gelen paketler kaynak tarafında hafifletilir.

Çizelge 2.5. Yansıma saldırısı tespit algoritması

1	Yeni paket (*p*) girişi
2	*i* <= Paketin giriş anahtarı bağlantı noktası
3	*d*<= Paketin hedef IP adresi
4	*s* <= Paketin IP adresi kaynağı
5	*Prop* <= p özelliği
6	Eğer *i* bir hedef bir port ise,
7	$diffSrc_d$<=$pcktWindow_d$ setindeki paketlerin farklı kaynak IP adresleri
8	$diffSrcProp_d$ <= $diffSrc_d$ setini oluşturan paketlerin özellikleri
9	$pcktWindow_d[lastEntry_d]$ <= *p*
10	Eğer $lastEntry_d$ < *windowSize* ise,
11	$lastEntry_d$ artar.

Çizelge 2.5.(devam) Yansıma saldırısı tespit algoritması

12	veya
13	$lastEntry_d$ <= 0
14	t_d <= $pcktWindow_s$ listesindeki toplam paket sayısı
15	Eğer s $diffSrc_d$ değil ise,
16	$diffSrc_d$ İçerisine *s* eklenir,
17	$diffSrcProp_d$ İçerisine prop eklenir.
18	$maxNumbSameProp_d$ <= $diffSrcProp_d$ setindeki aynı özelliklerin maksimum sayısı
19	$samePropSrc_d$ <= $maxNumbSameProp_d$ oluşturan paketlerin kaynak IP adresleri
20	Eğer t_d / *windowSize* > *saturationThreshold* ise
21	Eğer $diffSrc_d$ / t_d > *refSrcThreshold* ise
22	Eğer $maxNumbSameProp_d$/ $diffSrc_d$> *refPropThreshold* ise
23	$samePropSrc_d$ paketi kötü amaçlıdır.
24	$samePropSrc_d$ kontrol listesine gönderilmektedir.
25	t_d<= 0
26	$diffSrc_d$ seti temizlenmekte,
27	$diffSrcProp_d$ seti temizlenmekte,
28	$samePropSrc_d$ seti temizlenmekte,
29	$maxNumbSameProp_d$ <= 0

Örneğin, bir ICMP yansıma saldırısı altındaki bir kurban ana bilgisayarı düşündüğümüzde ve doğrudan kurban ana bilgisayara bağlı kenar anahtarının pencere boyutu 100'dür. Birkaç yansıtıcı düğüm, örneğin farklı kaynak IP adreslerine sahip 90 yansıtıcı, ana bilgisayara kötü amaçlı paketler gönderir. Kenar anahtarı, $pckWindow_d$ listesinde 100 paket sakladığında, 90 farklı yansıtıcıdan 95 paket alındığını ve iki farklı yasal ana bilgisayardan beş paket alındığını varsayalım. 95 zararlı paketin birbiriyle benzer paket özelliklerine sahip olduğunu,

5 normal paketten 2'sinin birbiriyle benzer paket özelliklerine sahip olduğunu ve kalan 3 paketin farklı paket özelliklerine sahip olduğunu varsayalım. Bu örnekte, $diffSrc_d$ listesi 92 farklı kaynak adresi içerecektir ve $diffSrcProp_d$ listesi, bu 92 farklı kaynaktan alınan en son paketlerin özellikleri olan 92 girişi içerecektir. Bu basit örnek için, *saturationThreshold* değerinin 0,75 ve *refPropThreshold* değerinin 0,8 olduğunu varsayalım. Öncelikle $pckWindow_d$ listesindeki paket sayısı ile *windowSize* değerinin oranı hesaplanır ve sonuç olarak 100/100 bulunur. Daha sonra bu bir oranı, 0,75 olan doygunluk Eşiği değeri ile karşılaştırılır. Listede yeterli sayıda paket olduğundan $diffSrc_d$ liste boyutunun $pckWindow_d$ listesindeki paket sayısına oranı artık 0,92 olarak hesaplanmış ve *refPropThreshold* değeri ile karşılaştırılmıştır. $diffSrcProp_d$ listesi aynı girişler için kontrol edilir; 95 adet giriş ile 2 adet girişin benzer paket özelliğinde olduğu tespit edilmiştir. Maksimum değer 95 olduğundan, $maxNumbSameProp_d$ 'a atanır. Daha sonra $maxNumbSameProp_d$ değerinin $diffSrc_d$ listesinin boyutuna oranı 1.03 (95/92) olarak hesaplanır ve bu oran *refPropThreshold* değeri ile de karşılaştırılır. Her iki karşılaştırma da bir Yansıma saldırısı olduğunu ima ettiğinden, $maxNumbSameProp_d$ değerini oluşturan paketlerin 90 kaynak IP adresi kötü amaçlı olarak etiketlenmektedir.

2.4. Değerlendirme

Başlık içerisinde MiddleModule sisteminin davranışını incelemek ve performansını analiz etmek için kapsamlı bir değerlendirme yapılmaktadır. Başlık bağlamında çalışmamızda yer alan alt başlıklarda performans ölçüm metrikleri ele alınmakta, değerlendirme senaryoları ayrıntılı olarak açıklanmakta ve elde edilen test sonuçları sunulmaktadır. Test sonuçları benzer önemli çalışmalarla karşılaştırılmış ve MiddleModule sisteminin avantaj ve dezavantajları açıkça belirtilmiştir.

MiddleModule sistemi OMNET++ 4.2 ve MİNİNET olmak üzere iki farklı platformda değerlendirilmektedir. OMNET++, ayrık bir olay simülasyon platformudur ve MİNİNET, süreç tabanlı sanallaştırma tekniğini kullanan bir ağ öykünücü platformudur. MiddleModule sistem tasarımı ve detaylı fonksiyonel analiz sırasında, esnek ve modüler yapısı nedeniyle OMNET++ 4.2 simülatörü kullanılmıştır. MiddleModule sisteminin performans analizi ve çalışma kanıtı için gerçekçi ve güvenilir yapısı nedeniyle MİNİNET ağ öykünücü platformu kullanılmaktadır.

MiddleModule sisteminin performansını değerlendirmek için savunma gücü ve sistem performansı düşüşü, farklı paket üretim hızları ve farklı test senaryoları kullanılarak ölçülmektedir. Elde edilen ölçüm sonuçları göz önünde bulundurularak, MiddleModule sisteminin ölçeklenebilirliği, uygulama karmaşıklığı ve uzlaşma yeteneği tartışılmaktadır.

2.4.1. Değerlendirme araçları

Değerlendirme araçları kendi içerisinde değerlendirme platformları ve trafik veri kümeleri olmak üzere iki farklı alana ayrılmaktadır. Aşağıda yer alan alt başlıklar içerisinde bu araçlar açıklanmaktadır.

Değerlendirme platformları

Bir ağ sistemi, bir simülatör platformunda veya bir donanım test yatağında veya bir emülatör platformunda değerlendirilebilir. Bu platformlar arasında pek çok fark vardır, bu farklar (Handigol, Heller, Jeyakumar, Lantz, Mckeown, 2012);

- Güvenilirlik,
- Gerçekçilik,
- Tekrarlanabilirlik,
- Esneklik ve
- Maliyet olarak sıralanmaktadır.

Çalışmamız içerisinde hem simülasyon hem de emülasyon konuları birlikte ele alınmaktadır. Simülatörler, tekrarlanabilir ve gerçekçi analizler sağlayabilir; esnek ve kolayca yönetilebilir ağ topolojileri ve düşük maliyetli değerlendirme sistemleri sağlarlar. Simülatörler kullanılarak özel ağ sistemleri kolayca oluşturulabilir ve özel topolojiler ve özel iletişim protokolleri ile değerlendirilebilir. Bu avantajlara rağmen, simülatörler gerçek ağlarda çalışan aynı kodları kullanmazlar ve trafik üretimi, iletişim protokolleri veya donanım cihazları için modelleri, güvenilirliklerini azaltan doğruluk endişelerini artırabilir (Handigol vd., 2012).

Emülatörler hem esnek hem de güvenilir değerlendirme platformları sağlar; dolayısıyla bu özellikler açısından simülatörler ile donanım test platformları arasında yer almaktadır. Emülatörler, sanal donanım oluşturarak etkileşimli ağ trafiği ile ağ uygulamaları, işletim sistemi çekirdeği vb. ile ilgili gerçek kodları çalıştırır; bu nedenle güvenilir ve doğru değerlendirme sonuçları sağlarlar. Öte yandan, simülatörler gibi, emülatörlerle özel topolojiler oluşturulabilir ve bunlar yine de düşük maliyetli analiz sistemleridir (Ringberg, Roughan ve Rexford, 2008).

Ek olarak önerildiği gibi, esnek ve yönetilebilir yapıları nedeniyle simülatörlerin kullanılması, ağ savunma sistemlerinin fonksiyonel analizi için faydalı olabilir. Bir ağ savunma sistemi tasarladıktan ve temel fonksiyonel analizleri yaptıktan sonra ağ emülatörleri, ağ savunma sistemlerinin detaylı performans analizi ve çalışma kanıtı için kullanılabilir. Emülatörler kullanılarak detaylı ve güvenilir analiz sonuçları toplanabilmekte ve önerilen sistemin performansı benzer çalışmalarla karşılaştırılabilmektedir. Çalışmamızda MiddleModule sisteminin değerlendirilmesi sırasında bu yaklaşım izlenmekte ve hem OMNET++ hem de MİNİNET kullanılmaktadır (Ringberg vd., 2008).

Trafik veri kümeleri

Ağ savunma sistemleri için güvenilir değerlendirme sonuçları elde etmek için trafik veri kümeleri önemli bir husus olarak öne çıkmaktadır. Ağ savunma sistemi oluşturulduktan sonra gerçekçi test senaryoları ile değerlendirilmesi gerekir ve ağ trafiği senaryonun önemli bir parçasıdır. MiddleModule sisteminin değerlendirilmesi sırasında farklı test platformları için farklı trafik veri setleri kullanılmaktadır. OMNET++ içeren simülasyon aşamasında, NSL-KDD veri seti yasal trafiği ve kötü amaçlı trafiğin bir kısmını oluşturmak için kullanılmaktadır (Tavallaee, Bagheri, Lu, Ghorbani, 2009).

Standart KDD veri setinin geliştirilmiş bir versiyonudur. Standart KDD veri seti, MIT Lincoln Laboratuvarlarında üretilir. Gerçek bir askeri LAN ortamını simüle eden bir yerel alan ağı oluşturuldu ve gerçek bir ağ ortamı varmış gibi değerlendirilerek çalıştırılmıştır. Ayrıca diğer saldırı türleri ile birlikte bazı DoS saldırıları da ağ üzerinde gerçekleştirilmiştir. Bu ağ trafiği izlenmiş ve her bağlantının özellikleri kaydedilerek KDD veri setinde sunulmuştur. NSL-KDD, gereksiz paketleri ve yinelenen kayıtları vb. kaldırarak orijinal KDD veri setini geliştirir. Öte yandan, MİNİNET platformunun kullanıldığı emülasyon

aşamasında, literatürde yaygın olarak kullanılan popüler bir veri seti olan MAWI Working Group Traffic Archive'den alınan gerçek bir trafik veri seti kullanılarak trafik oluşturulmuştur. 2002'den günümüze gerçek trafik veri setlerini paylaşan bu veri seti alan içerisinde yaygın biçimde kullanım alanı bulmaktadır (Tavallaee vd., 2009).

Performans ölçüleri

Bir DDoS savunma mekanizmasını tasarlarken ve test ederken göz önünde bulundurulması gereken bazı performans ölçüm metrikleri bulunmaktadır. Zargar ve diğerleri (2013), bir savunma sisteminin gücü, sistem performansının düşmesi, ölçeklenebilirliği, uygulama karmaşıklığı ve ödün verme yeteneği DDoS savunma sistemi açısından önem teşkil ettiğini belirtmektedir. Bunun yanı sıra Ağ/Aktarım katmanında bir DDoS savunma mekanizması, yüksek trafik enjeksiyon oranlarını işlemelidir. Bundan dolayı, değerlendirme esnasında izin verilen kötü amaçlı trafik enjeksiyon hızı da göz önünde bulundurulmalıdır.

Savunma gücü

Bir sistemin savunma gücü, onun kötü amaçlı etkinliklere yönelik tespit ve önleme performansını göstermektedir. Savunma sistemlerinin tepkilerine göre Çizelge 2.6'da gösterildiği gibi yanlış pozitif, yanlış negatif, doğru pozitif ve doğru negatif olmak üzere dört farklı sonuç elde edilebilir (Zargar vd., 2013).

Çizelge 2.6. Bir savunma sisteminin olası sonuçları

	İstenen Sonuç Pozitif	İstenen Sonuç Negatif
Sistem Tepkisi Pozitif	Doğru Pozitif (DP)	Yanlış Pozitif (YP)
Sistem Tepkisi Negatif	Yanlış Negatif (YN)	Doğru Negatif (DN)

Savunma sisteminin sonuçlarına dayalı olarak altı savunma gücü performans ölçütü hesaplanabilir ve bu ölçütler bir savunma sistemini değerlendirmek için kullanılabilir. Bu metrikler Çizelge 2.7. gösterildiği gibi hesaplanır.

Çizelge 2.7. Savunma gücü performans metrikleri

Metrik Türü	Açıklama	Ölçüm
Kesinlik	Bir savunma sisteminin tüm olaylara doğru tepki verme derecesini ölçer.	$= \frac{DP + DN}{YP + YN + DP + DN}$
Duyarlılık	Tüm olumlu olaylara ilişkin doğru olumlu yanıtların derecesini ölçer.	$= \frac{DP}{DP + YN}$
Özgünlük	Tüm olumsuz olaylar üzerindeki doğru olumsuz tepkilerin derecesini ölçer.	$= \frac{DN}{YP + DN}$
Hassasiyet	Tüm olumlu yanıtlar üzerindeki doğru olumlu yanıtların derecesini ölçer.	$= \frac{DP}{DP + YP}$
Yanlış Pozitif Oranı	Tüm olumlu yanıtlar üzerindeki yanlış olumlu yanıtların derecesini ölçer.	$= \frac{YP}{DP + YP}$
Yanlış Negatif Oranı	Tüm olumsuz yanıtlar üzerindeki yanlış olumsuz yanıtların derecesini ölçer.	$= \frac{YN}{YN + DN}$

MiddleModule sisteminin değerlendirilmesi esnasında Duyarlılık, Kesinlik ve Yanlış Pozitif Oranı değerleri hesaplanmakta ve analiz edilmektedir.

Ölçeklenebilirlik

Bir ağ sisteminden geçen trafik miktarı ve bir ağ sistemine bağlı düğümlerin sayısı artabileceği veya azalabileceği hususunda sistem performansının bu tür değişikliklerden önemli ölçüde etkilenmemesi gerektiğinden, ölçeklenebilirlik ağ oluşturmada önemli bir değerlendirme ölçütü olarak değerlendirilmektedir.

Sistem performansı üzerindeki azalma

Sistem performansı üzerindeki azalma metriği, savunma sisteminin kullanılmasının neden olduğu ağda yaşanan performans düşüşünü ölçmektedir. Gerçekleştirilen savunma sistemi, ağ düğümlerinin iş yükünü artırabilir veya düğümlerin iletme gecikmesini artırabilir. Bunlar genel sistem performansının düşmesine sebebiyet verebilmektedir. Bu, özellikle veri katmanı tabanlı bir DDoS savunma sistemi açısından önemli bir husustur. Çünkü savunma sistemi yönlendirme cihazları üzerinde uygulanmaktadır. Bundan dolayı, yasal ana

bilgisayarlarda ek savunma sisteminin neden olduğu iletme gecikmesi veya ortalama paket alma gecikmesi (saniye olarak) dikkatlice analiz edilmelidir. Bu amaçla ağ, önerilen sistemle test edilmeli ve yasal ana bilgisayarların paket alma süreleri (paket isteği gönderme ile yanıt alma arasındaki süre) ölçülmelidir (saniye olarak).

Uygulama kompleksi

Uygulama Kompleksi metriği, bir savunma sistemini uygulamak için gerekli ek kaynakları ve değiştirilmesi gereken gerekli sayıda cihazı göstermektedir. Uygulama karmaşıklığı arttığında, savunma sisteminin uygulanabilirliği azalmaktadır.

İzin verilen paket alma hızı

İzin verilen paket alma hızı, veri katmanı tabanlı Ağ/Aktarım katmanındaki DDoS savunma mekanizması bağlamında önemli bir özelliktir. Ağ/Aktarım katmanındaki DDoS saldırılarının en belirgin özelliklerinden birisi, saldırganların mümkün olan en yüksek paket üretim hızıyla ağa DDoS trafiği enjekte etmeleridir. Günümüzün gelişmiş ağ teknolojilerinde, ana bilgisayarların paket oluşturma hızı artmaktadır. Bu sebeple, bir savunma sisteminin izin verilen maksimum veri hızı (saniye başına paket olarak) kritik bir performans ölçütü haline gelmektedir. DDoS savunma sistemleri, daha yüksek paket üretim hızlarını kaldırabilecek şekilde tasarlanmalı ve bu sistemlerin savunma gücü performansları da bu koşullar altında test edilmelidir.

Ödün verme durumu

Tüm ağ savunma sistemleri, ağı bazı kötü amaçlı faaliyetlere karşı korumak için kullanılmaktadır. Fakat uygulanan ağ savunma sistemi kötü niyetli kullanıcılar tarafından bir şekilde istismar edilirse, herhangi bir ağ saldırısına maruz kalabilme durumu oluşmaktadır. Bu önemli bir konudur çünkü bir savunma sistemi istismar edildiğinde, o savunma sistemi için hesaplanan doğruluk değerleri veya sistem performansı düşüş değerleri önemsiz hale gelir. Bunun sebebi ise ağ artık bu durumda saldırılara karşı savunmasızdır. Bu nedenle, herhangi bir savunma sisteminin değerlendirilmesi sırasında, sistemin ödün verme durumu ayrıntılı olarak tartışılmalıdır.

2.4.2. Değerlendirme testleri ve test sonuçları

MiddleModule sisteminin değerlendirme sürecinde farklı test platformlarında sisteme çeşitli testler uygulanmıştır. Bu amaçla daha önce de açıklandığı gibi OMNET++ 4.2 simülasyon platformu ve MİNİNET emülasyon platformu kullanılmıştır.

İlk olarak, önerilen yöntemin sistem performans düşüşü OMNET++ platformunda test edilmiştir. Bu amaçla, MiddleModule sistemi olan ve olmayan bir normal kullanıcının ortalama paket alma süresi ölçülmüştür. Daha sonra özel test senaryoları ile OMNET++ platformunda savunma gücü değerlendirmesi yapılmıştır. Savunma gücü testlerinde kullanılan test senaryoları, MiddleModule sisteminin çalışma prensipleri dikkate alınarak tasarlanmıştır. MiddleModule sisteminin hedefe yönelik kötü amaçlı saldırı türlerine karşı düzgün çalıştığı gözlemlendikten sonra MİNİNET üzerinde detaylı performans analiz testleri başlatılmıştır. Bu platformda öncelikle sistem performans düşüşü ölçülmüştür. Ardından farklı test senaryoları ve farklı paket üretim hızları kullanılarak savunma gücü ölçülmüştür. MİNİNET platformundaki testler sırasında kullanılan test senaryoları, yasal trafik oluşturma veri setleri ve hedeflenen DDoS saldırı türleri, literatürde bulunan benzer çalışmalar dikkate alınarak seçilmiştir. Bu testlerde toplanan sonuçlar, MiddleModule sisteminin performansını mevcut çalışmalarla karşılaştırmak için kullanılmıştır.

Testlerin güvenilirliğini ve doğruluğunu artırmak için test prosedürleri aşağıdaki noktalar dikkate alınarak tasarlanmıştır:

- Gerçekçi yasal trafik oluşturma,
- Çoklu turlar açısından testleri uygulama,
- LFarklı veri hızlarıyla trafik oluşturma.

Gerçekçi yasal trafik oluşturulması açısından, OMNET++ 4.2 üzerindeki simülasyon testlerine yönelik olarak yasal trafik, NSL-KDD veri seti kullanılarak üretilmiştir. Emülasyon testlerinde ise MİNİNET üzerindeki MAWI Çalışma Grubu Trafik Arşivinden alınan gerçek bir trafik veri seti kullanılmıştır. Yasal trafik, bu veri kümeleri kullanılarak üretilmiştir. Kötü amaçlı trafik, hem bu veri kümelerindeki kötü amaçlı paketler hem de karşılık gelen saldırıların bariz özellikleri kullanılarak oluşturulmuştur.

Emülasyon sonuçlarının güvenilirliğini artırmak için MAWILab trafiği kullanılmıştır, çünkü SYN patlamaları, paket hataları, paketin yeniden iletimleri, üçlü ACK'ler gibi kötü amaçlı olmayan anormallik içermektedir. MİNİNET bağlamında yasal ve kötü amaçlı trafiği farklı veri hızlarıyla ağa enjekte etmek için Scapy ve packETH programları kullanılmıştır. Scapy, birçok yeteneğe sahip gelişmiş bir paket oluşturma aracıdır. Ancak *1 paket / 500 milisaniye* 'den daha yüksek oranlarda trafik enjekte edemez. Bunun yanı sıra packETH, hafif bir paket oluşturma aracıdır ve *1 paket / 1 mikrosaniye* 'ye kadar daha yüksek veri hızlarıyla trafiği enjekte edebilir. Bunlardan farklı olarak her iki test platformunda yasal trafiği dalgalandırmak için farklı yüzdelerde hem TCP paketleri hem de ICMP paketleri kullanılmıştır.

Çoklu turlar açısından testlerin uygulanması açısından, değerlendirme sırasında testlerin güvenirliğini artırmak için her testte bir tur yerine birden fazla tur uygulanmıştır. Her tur bir süre devam eder veya her ana bilgisayar kesin sayıda paket gönderene kadar sürer. Bu turlar için ortalama performans değerleri, ilgili testlerin sonucu olarak hesaplanır ve raporlanır. Her testte, yöntemin o test için nasıl uygulandığına ilişkin ayrıntılar açıklanmıştır.

Çizelge 2.8. Trafik oluşturmaya yönelik modlar

Trafik Oluşturma Hızına Dair Modlar	Paket Üretim Oranı	Aktarılan Veri Oranı (Ortalama paket boyutu 250 bayt)
Ultra düşük hız modu	100 paket / saniye, 1 paket / 10 milisaniye	200 kb / sn
Düşük hız modu	500 paket / saniye, 1 paket / 2 milisaniye	1 mb / sn
Normal hız modu	5000 paket / saniye, 1 paket / 200 mikrosaniye	10 mb / sn
Yüksek hız modu	100000 paket / saniye, 1 paket / 10 mikrosaniye	200 mb / sn
Ultra yüksek hız modu	500000 paket / saniye, 1 paket / 2 mikrosaniye	1 gb / sn

Farklı veri hızlarıyla trafik oluşturulması açısından, değerlendirme sırasında hem yasal hem de kötü amaçlı trafik için Çizelge 2.8'deki gibi ultra düşük hız, düşük hız, normal hız, yüksek hız ve ultra yüksek hız modu gibi farklı trafik oluşturma hızları kullanılmıştır.

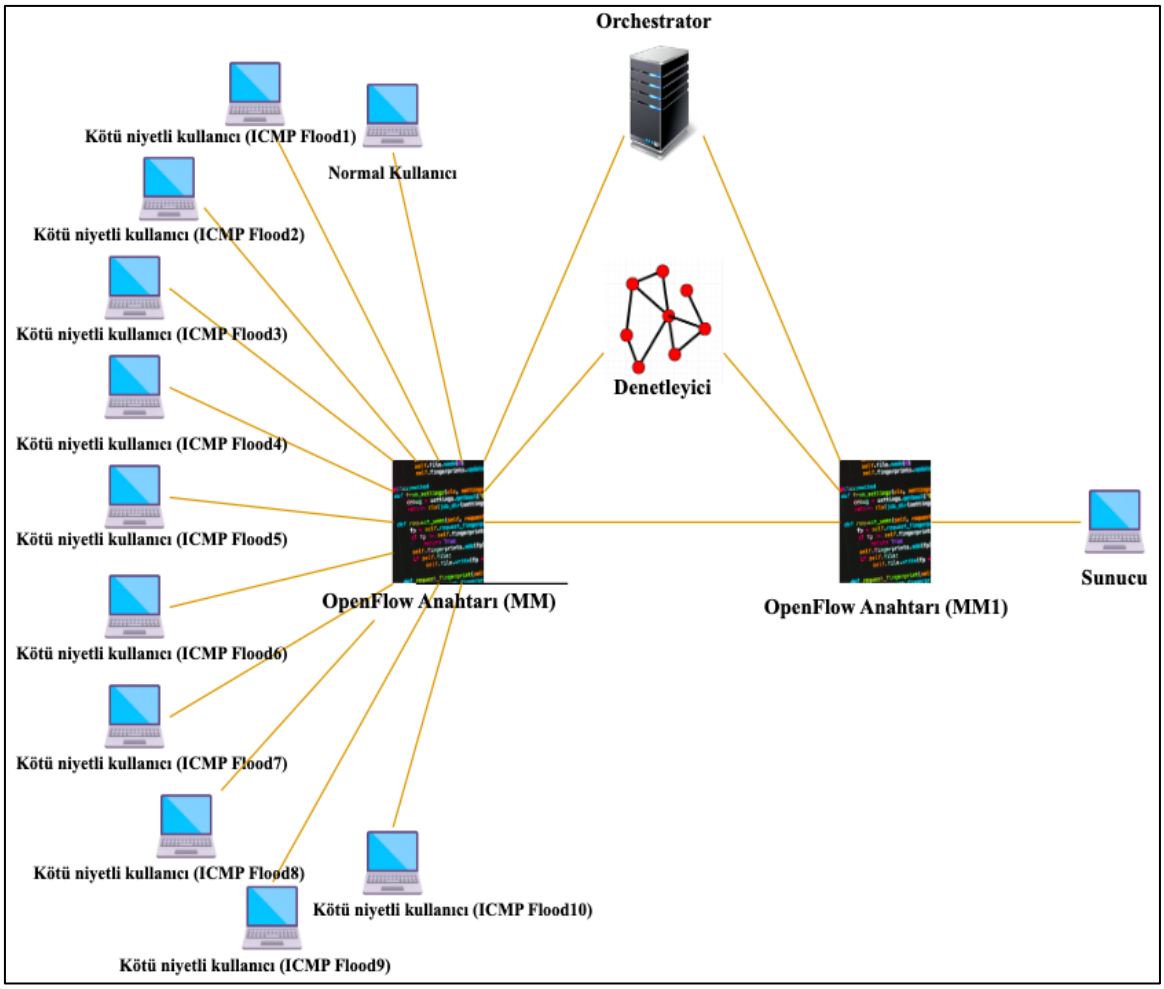
OMNET kapsamında değerlendirme

OMNET açısından sistem performansında bozulma testi ve savunma gücü testi durumlarına bakılmaktadır. Sistem performansında bozulma testi kapsamında MiddleModule sistemine yönelik olarak normal kullanıcının ortalama paket alma süresi üzerindeki etkisi ölçülmektedir. Test, MiddleModule sistemli-sistemsiz ile kötü amaçlı kullanıcı düğümleri tarafından oluşturulan kötü amaçlı trafik içeren-içermeyen dört farklı senaryo kullanılarak uygulanmaktadır. Ayrıca, OMNET++ 4.2 simülasyon platformunda, MiddleModule sistemine üç hedefi gerçekleştirmek için savunma gücü testleri uygulanmaktadır. Savunma gücü testinde, MiddleModule sisteminin kötü amaçlı etkinlikleri tespit edip etmediğini gözlemlemek için temel fonksiyonel analize yönelik testler uygulanmaktadır. Bunun yanı sıra, MiddleModule sisteminin daha kompleks test senaryoları bağlamında doğru çalışıp çalışmadığını gözlemlemek için daha fazla fonksiyonel analiz yapılmaktadır.

MiddleModule sistemi OMNET++ 4.2 adlı platformda değerlendirilmektedir. OMNET++, ayrık bir olay simülasyon platformudur. MiddleModule sistem tasarımı ve detaylı fonksiyonel analiz sırasında, esnek ve modüler yapısı nedeniyle OMNET++ 4.2 simülatörü kullanılmıştır. MiddleModule sisteminin performansını değerlendirmek için savunma gücü ve sistem performansı düşüşü, farklı paket üretim hızları ve farklı test senaryoları kullanılarak ölçülmektedir.

Sistem Performansında Bozulma Testi

Şekil 2.9'da gösterilen test senaryosu, OMNET++ 4.2 platformunda oluşturulmuştur. Bu testte MiddleModule sistemine yönelik olarak normal kullanıcının ortalama paket alma süresi üzerindeki etkisi ölçülür. Test, MiddleModule sistemli-sistemsiz ile kötü amaçlı kullanıcı düğümleri tarafından oluşturulan kötü amaçlı trafik içeren-içermeyen dört farklı senaryo kullanılarak uygulanmaktadır. İlk olarak normal kullanıcı, NSL-KDD veri setini kullanarak sunucuya doğru ve yasal TCP trafiği oluştururken, on tane kötü amaçlı kullanıcıya yönelik ana bilgisayarlar herhangi bir trafik oluşturmaz ve MiddleModule sistemi devre dışı bırakılır.



Şekil 2.9. OMNET++ kapsamında sistem performansı düşürme testi senaryosu

Bu test sırasında Çizelge 2.9'da açıklanan aşağıdaki adımlar kullanılır.

Çizelge 2.9. OMNET++ açısından test aşamaları

1.	'*test.ned*' dosyasında uygun ağ topolojisini oluşturun.
2.	*TCPBasicClientApp* kapsamında değiştirilmiş sürümünü kullanarak *Normal_User* (Normal Kullanıcı) düğümü için '*omnetpp.ini*' dosyasında aşağıdaki özellikleri ayarlayın. Bu nedenle *Normal_User* düğümü, TCP istemcisi olarak çalışacak ve NSL-KDD veri setine göre paketler üretecektir. a) "**.Normal_User.numTcpApps = 1 b) **.Normal_User.tcpApp[*].typename = TCPBasicClientApp c) **.Normal_User.tcpApp[*].kMeansMode = normal d) **.Normal_User.tcpApp[*].connectAddress = server (sunucu)"

Çizelge 2.9. (devam) OMNET++ açısından test aşamaları

3.	Her *Malicious_User* düğümü için '*omnetpp.ini*' dosyasında aşağıdaki özellikleri ayarlayın. a) "**.Malicious_User*. numPingApps = 1 b) **.Malicious_User*.pingApp[*].isIPSpoofing = true c) **.Malicious_User*.pingApp[*].typename = PingApp d) **.Malicious_User*.pingApp[*].destAddr = server e) **.Malicious_User*.pingApp[*].packetSize = 250B f) **.Malicious_User*.pingApp[*].count = 10000 g) **.Malicious_User*.pingApp[*].sendInterval = 10us"
4.	Her *open_flow_switch* düğümü için '*omnetpp.ini*' dosyasında aşağıdaki özellikleri ayarlayın. a) "**.open_flow_switch*.open_Flow_Processing.windowSize = 40 b) **.open_flow_switch*.open_Flow_Processing.saturationThreshold=0.15 c) **.open_flow_switch*.open_Flow_Processing.maliciousThreshold=0.8"
5.	*TCPEchoApp* açısından değiştirilmiş sürümü kullanarak her sunucu düğümü için '*omnetpp.ini*' dosyasında aşağıdaki özellikleri ayarlayın. Bu nedenle, sunucu düğümü hem TCP hem de ICMP web sunucusu olarak çalışacaktır. TCP web sunucusu, NSL-KDD veri setine göre yanıtlar üretir. a) "**.server*.numTcpApps = 1 b) **.server*.numPingApps = 1 c) **.server*.pingApp[*].typename = PingApp d) **.server*.tcpApp[*].typename = TCPEchoApp **.server*.tcpApp[0].localAddress = 0 e) **.server*.tcpApp[0].localPort = 1000 f) **.server*.tcpApp[0].echoFactor = 1.0 g) **.server*.tcpApp[0].echoDelay = 0"
Malicious_User düğümleri için girişleri yorumlayın, '*omnetpp.ini*' açısından aşağıdaki özelliği ayarlayın ve testi uygulayın. Bu nedenle *Malicious_User* düğümleri herhangi bir paket üretmez ve MiddleModule sistemi çalışmaz. a) "**.open_flow_switch*.open_Flow_Processing.isMMIDSOn = false"	
Bu koşullar altında *Normal_User* düğümü yeni TCP bağlantısı kurar ve ardından NSL-KDD kullanarak TCP paketleri üretir. *TCPBasicClientApp* kapsamında değiştirilmiş sürüm, SYN istek oluşturma süresini, FIN yanıt alma süresini ve bunların farklılıklarını simülasyon penceresinde milisaniye cinsinden yazar.	

Öncelikle 1'den 7'ye kadar olan aşamaları uygulayın. Bu testte, sunucu düğümünde bir web sunucusu yerleşiktir ve Normal Kullanıcı düğümü, sunucu düğümüne bir TCP bağlantısı oluşturur. Bağlantı kurulduktan sonra Normal Kullanıcı düğümü sunucu ile veri alışverişi yapar ve ardından TCP bağlantısını kapatır. Normal Kullanıcı düğümü tarafından bağlantı başlatma süresi ile FIN paketi alma süresi arasındaki fark, simülatör penceresinde milisaniye cinsinden gösterilir. Bu zaman farkı, Normal Kullanıcı düğümünün Dağıtılmış Hizmet Reddi saldırıları ve MiddleModule sistemi olmadan tek paket alma süresini gösterir. Bu testi, her turda 100 bağlantı içeren 10 tur boyunca uygulayın. Ortalama paket alma süresini milisaniye cinsinden hesaplayın.

İkinci testte 1'den 7'ye kadar olan aşamaları uygulayın ancak 6. adımda parametreyi aşağıda gösterildiği gibi *true* olarak ayarlayın.

"***.open_flow_switch*.open_Flow_Processing.isMMIDSOn = true*"

Bu testte Kötü Amaçlı Kullanıcı düğümleri, herhangi bir trafik oluşturmaz ancak MiddleModule sistemi etkinleştirilir. İlk testin test prosedürünü takip edin ve milisaniye cinsinden 10 tur için ortalama paket alma sürelerini hesaplayın.

Üçüncü ve dördüncü testlerde, önceki iki test tekrarlanırken, Kötü Amaçlı Kullanıcı düğümleri yüksek hızlı modda "IP Spoofed (Kandırma) SYN Flood Trafiği" oluşturur. Bu amaçla 5. adımda, Kötü Amaçlı Kullanıcı düğümleri için girişlerde yorum yapın. Bu koşullar altında toplam ortalama paket alma sürelerini milisaniye cinsinden hesaplayın. Açıkçası, paket alma süreleri gerçek zamanlı değil, simülasyon zamanındadır.

Çizelge 2.10. OMNET++ kapsamında sistem performansı düşüşü sonucu

	Dağıtılmış Hizmet Reddi (DDoS) trafiği yok	Dağıtılmış Hizmet Reddi (DDoS) trafiği var
MM yok	0,531 milisaniye	Sonsuz
MM var	0,532 milisaniye	0,534 milisaniye

Genel ortalama paket alma süreleri Çizelge 2.10'da karşılaştırılmıştır. Bu çizelgenin gösterdiği gibi, MiddleModule sistemi DDoS saldırısı olmadığında Normal Kullanıcı paketi alma süresinde gözle görülür bir bozulmaya neden olmaz. Kötü Amaçlı Kullanıcı düğümleri, daha önce bahsedilen anahtar arabelleği taşmaları ve anahtar akış tablosu taşmaları nedeniyle DDoS trafiği oluşturduğunda, Normal Kullanıcı, MiddleModule sistemi devre dışı bırakılırsa sunucudan herhangi bir yanıt alamaz. Normal Kullanıcının ortalama paket alma süresi sonsuza kadar gider. Öte yandan, DDoS saldırıları sırasında MiddleModule sistemi çalışıyorsa, Normal Kullanıcı herhangi bir paket kaybı olmadan sunucu ile iletişim kurabilir. Paket alma süresi, ağda DDoS saldırısı olmadığında ortalama paket alma süresine çok yakındır.

Savunma gücü testi

OMNET++ 4.2 simülasyon platformunda, MiddleModule sistemine üç hedefi gerçekleştirmek için savunma gücü testleri uygulanmaktadır. İlk olarak, MiddleModule sisteminin hedeflenen kötü niyetli faaliyetleri tespit edip etmediğini gözlemlemek için temel fonksiyonel analiz için testler uygulanmaktadır.

İkinci olarak, MiddleModule sisteminin daha karmaşık test senaryoları altında doğru çalışıp çalışmadığını gözlemlemek için daha fazla işlevsel analiz gerçekleştirilmektedir. İkinci hedefi gerçekleştirmek için, hedeflenen tüm Ağ/Aktarım katmanında DDoS saldırıları aynı anda kötü amaçlı kullanıcı düğümleri tarafından üretilir ve yasal trafik normal kullanıcı tarafından oluşturulmaktadır.

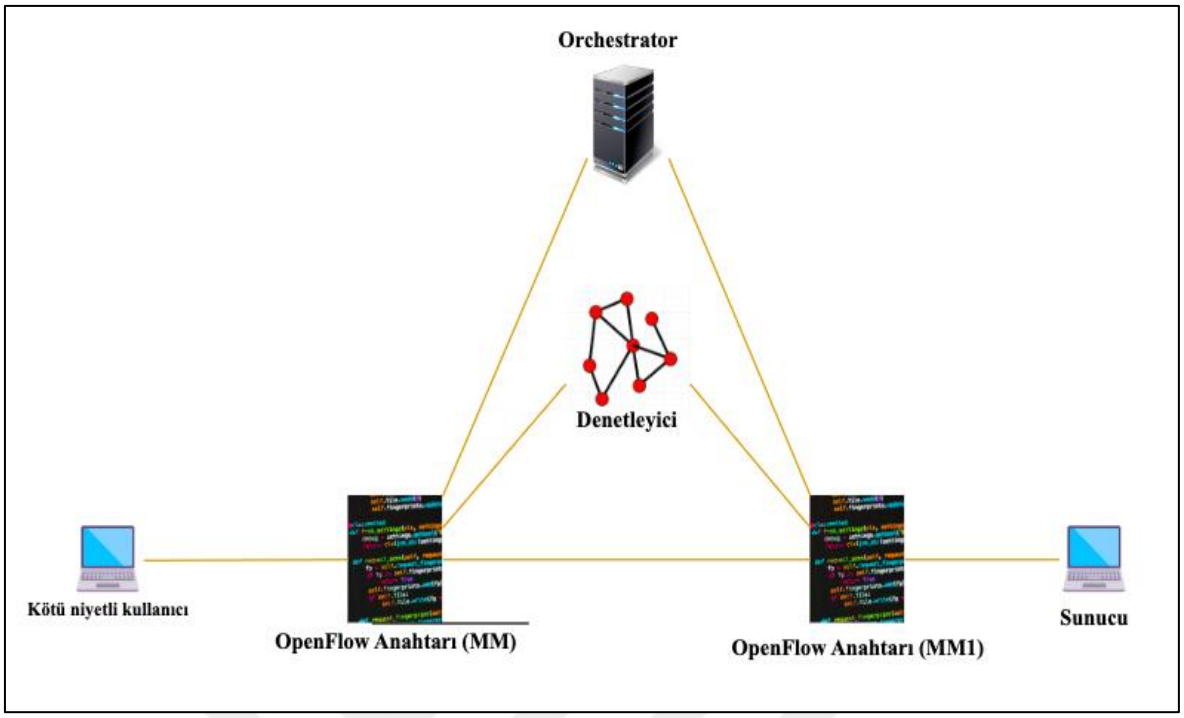
Bu koşullar altında, MiddleModule sisteminin algılama performansı değerlendirilmektedir. Son olarak, MiddleModule sisteminin yapılandırma değişkenleri, algılama algoritması eşikleri ve *windowSize* çeşitlendirilir ve MiddleModule sisteminin savunma gücü performansı ölçülmektedir. MiddleModule sisteminin savunma gücü performansı ile konfigürasyon değişkenleri arasındaki ilişki bu test ile değerlendirilmektedir. Bu testlerde IP spoofing saldırıları, SYN Flood saldırıları, ICMP Flood saldırıları, UDP Flood saldırıları, ICMP Reflection saldırıları ve Broadcast Amplification saldırıları hedeflenmektedir.

Testler esnasında Çizelge 2.9'da açıklanan 1'den 7'ye kadar olan adımlar kullanılmaktadır. Adım 3, uygulanan teste göre değişim göstermektedir. Örneğin UDP Flooding için

“*PingApp*” yerine “*UDPApp*” kullanılmaktadır. 5. adımda UDP sunucu uygulaması da çalıştırılır ve 6. adımda Kötü Amaçlı Kullanıcı düğümleri için girişlerde yorum yapılır ve parametreyi “*true*” olarak ayarlanır. MiddleModule sisteminin algılama algoritması parametre bağımlılık analizi sırasında, 4. adımdaki parametreler Çizelge 12’ye göre değiştirilmektedir. “*open_flow_switch*” kaynak kodundaki değişiklikler sayesinde algılama ve önleme kararı simülasyon penceresinde gösterilir.

“***.open_flow_switch*.open_Flow_Processing.isMMIDSOn = true*”

İlk olarak, Şekil2.10’da gösterilen test senaryosu, MiddleModule sisteminin temel işlevselliğini test etmek için OMNET++ platformunda oluşturulmaktadır. DDoS algılama işlevleri, “*open_flow_switch_withMM*” ve “*open_flow_switch_withMM1*” kenar anahtarlarında yerleşiktir ve denetleyicide bulunan önerilen Orkestrator Bloğunu simüle eden “*Myorchestrator modülüne*” bağlanır. “*open_flow_switch_withMM*” modülü herhangi bir kötü amaçlı aktivite tespit ederse, “*Myorchestrator*” modülüne haber verir. Beş farklı test uygulanmakta ve her birinde kötü amaçlı kullanıcı modülünden sunucu modülüne doğru farklı kötü amaçlı trafik (IP Spoofing trafiği, SYN Flood trafiği, ICMP Flood trafiği, UDP Flood trafiği ve Broadcast Amplification trafiği) üretilmektedir. Bu testlerde, kötü amaçlı kullanıcı yüksek hızlı modda trafik oluşturur ve yapılandırma değişkenleri Çizelge 2.13’e göre ayarlanmaktadır.

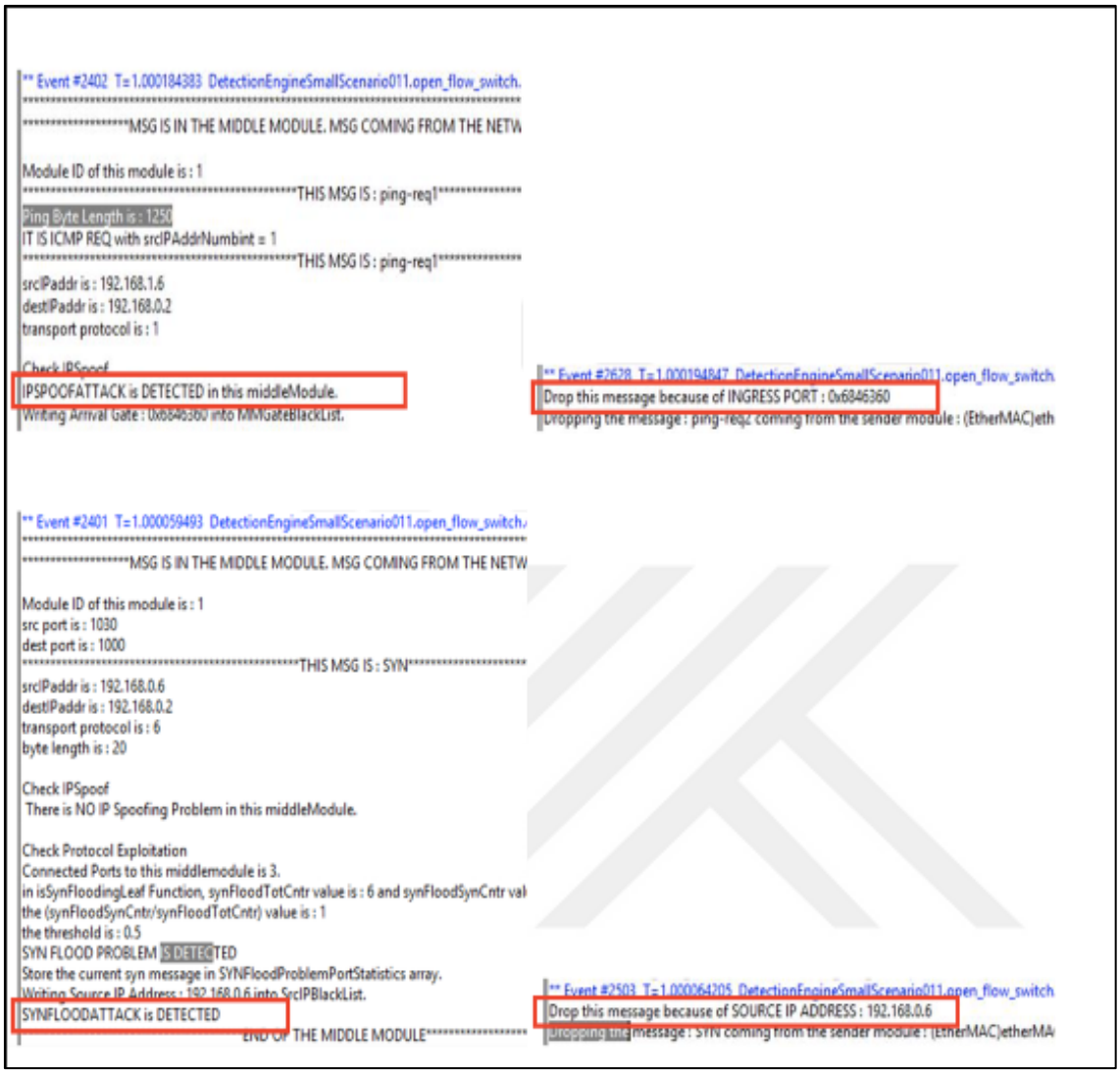


Şekil 2.10. OMNET++ için temel savunma gücü test senaryosu

Test sonuçları, kötü amaçlı paketlerin, kötü amaçlı koşulları sağladıklarında "*open_flow_switch_withMM*" modülü tarafından algılandığını göstermektedir. "*open_flow_switch_withMM*" modülü kötü amaçlı etkinliğin kaynağına daha yakın olduğundan, bu modül kötü amaçlı etkinliği algılar ve engeller. "*open_flow_switch_withMM*" düğümünün önleme kararları analiz edilir ve tespit oranı aşağıdaki gibi hesaplanır;

$$Algılama\ Oranı = \frac{\text{Engellenmiş kötü amaçlı paket sayısı}}{\text{Tüm kötü amaçlı paket sayısı}} \tag{2.1}$$

Bu koşullar altında, MiddleModule sisteminin algılama oranı %99'un üzerinde olarak ölçülmektedir. OMNET++ simülasyon platformundan toplanan tespit ve önleme operasyon çıktılarından bazıları Şekil 2.11'de gösterilmektedir.

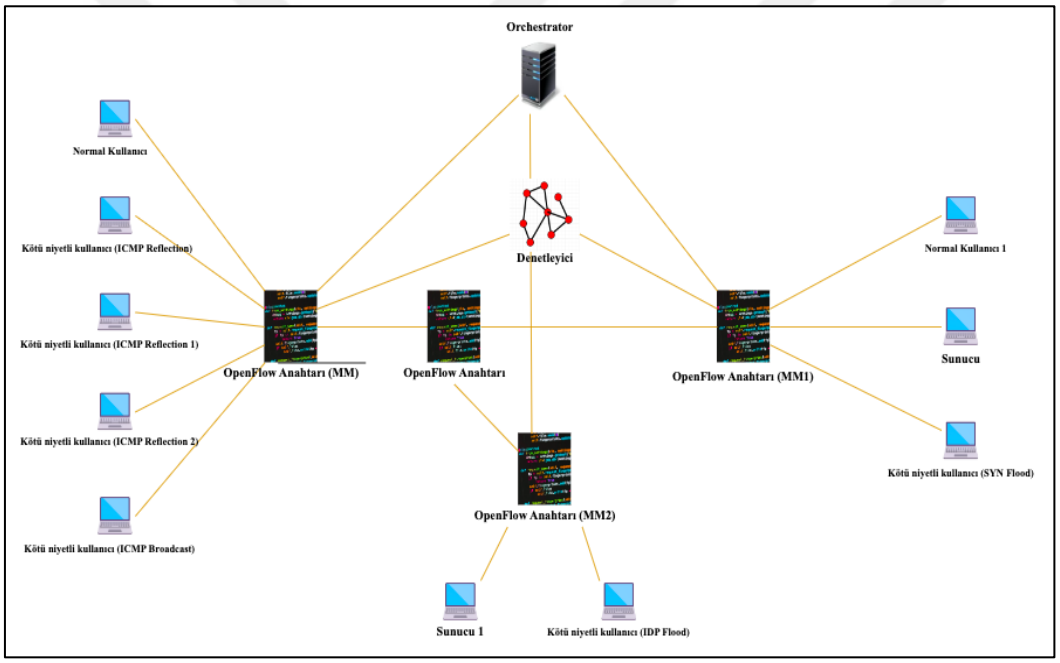


Şekil 2.11. OMNET++'tan temel savunma gücü testi sonuçları

OMNET++ simülasyonunda ikinci ama daha karmaşık bir test senaryosu oluşturulur ve bu senaryo kullanılarak savunma gücü testleri yeniden uygulanır. Bu test sırasında hem yasaltrafik hem de kötü amaçlı trafik aynı anda üretilir. Farklı kötü amaçlı kullanıcı düğümleri tarafından farklı saldırı türleri de üretilir. Bu test için, Şekil 2.12'de gösterilen senaryo OMNET++'ta oluşturulmuştur ve yasal trafik, "*normal kullanıcı*" ve "*normal kullanıcı1*" modülleri tarafından oluşturulur.

Testin güvenilirliğini artırmak için, yasal trafik, TCP ve TCP olmak üzere iki farklı iletişim protokolü kullanılarak oluşturulur. ICMP ve yasal trafiğin yüzde 90'ı TCP ile kalan yüzde 10'u ise ICMP ile oluşturulmuştur. TCP trafiği NSL-KDD veri seti kullanılarak üretilir ve ICMP trafiği normal kullanıcı modüllerinden sunucu ve sunucu1 modüllerine doğru ping

mesajları olarak üretilir. Şekilde gösterildiği gibi, üç kötü amaçlı kullanıcı düğümü, sunucu modülüne yönelik ICMP yansıması saldırısı oluşturur, bir kötü amaçlı kullanıcı düğümü, sunucu1 modülüne yönelik yayın yükseltme saldırısı oluşturur ve normal kullanıcı modülü, sunucu modülüne doğru yasal trafik oluştururken normal kullanıcı1 modülü, sunucu1 modülüne doğru yasal trafik oluşturur. Ayrıca, bir kötü amaçlı kullanıcı, sunucu modülüne yönelik UDP Flooding saldırısı oluştururken, bir kötü amaçlı kullanıcı, sunucu1 modülüne yönelik IP Spoof ile SYN Flooding saldırısı oluşturur. Uç anahtarlarda uygulanan algılama algoritmalarının yapılandırma değişkenleri Çizelge 2.13'e göre ayarlanır. 10 tur boyunca yüksek hızlı modda kötü amaçlı trafik oluşturulur ve her tur 20 ms simülasyon süresi sürmektedir.



Şekil 2.12. OMNET++ için karmaşık savunma gücü test senaryosu

"*open_flow_switch_withMM*" düğümünün önleme kararları analiz edilerek algılama oranı (hassasiyet) ve yanlış pozitif oranları aşağıdaki gibi hesaplanır;

$$Algılama\ Oranı = \frac{\text{Engellenmiş kötü amaçlı paket sayısı}}{\text{Tüm kötü amaçlı paket sayısı}} \tag{2.2}$$

$$Yanlış\ Pozitif\ Oran = \frac{\text{Engellenmiş normal paket sayısı}}{\text{Tüm normal paket sayısı}} \tag{2.3}$$

Simülasyon sonuçları, MiddleModule sisteminin kötü niyetli faaliyetleri doğru bir şekilde tespit ederken, yasal trafiği herhangi bir paket kaybı olmadan ilettiğini göstermektedir. Böylece DDoS saldırıları sırasında MiddleModule sistemi sayesinde normal kullanıcı düğümleri sunucu düğümleri ile başarılı bir şekilde haberleşebilmekte, kötü amaçlı kullanıcı düğümlerinden gelen trafik ise veri katmanı cihazlarında sunucu düğümlerine veya denetleyiciye gönderilmeden engellenmektedir.

Bu koşullar altında, MiddleModule sistem modelinin duyarlılığı, yanlış pozitif oranı ve doğruluğu ölçülmekte ve Çizelge 2.11'de gösterilmektedir. Duyarlılık sonuçları tüm saldırı türleri için %99,6'nın üzerinde olarak ölçülür ve doğruluk sonuçları %99,8'in üzerindedir; bu, hedeflenen tüm Ağ/Aktarım katmanında DDoS saldırı türlerinin MiddleModule sistemi tarafından doğru bir şekilde algılandığı anlamına gelmektedir. Öte yandan, yasal paketler MiddleModule sistemi tarafından bırakılmaz; bu nedenle yanlış pozitif oranı sıfırdır.

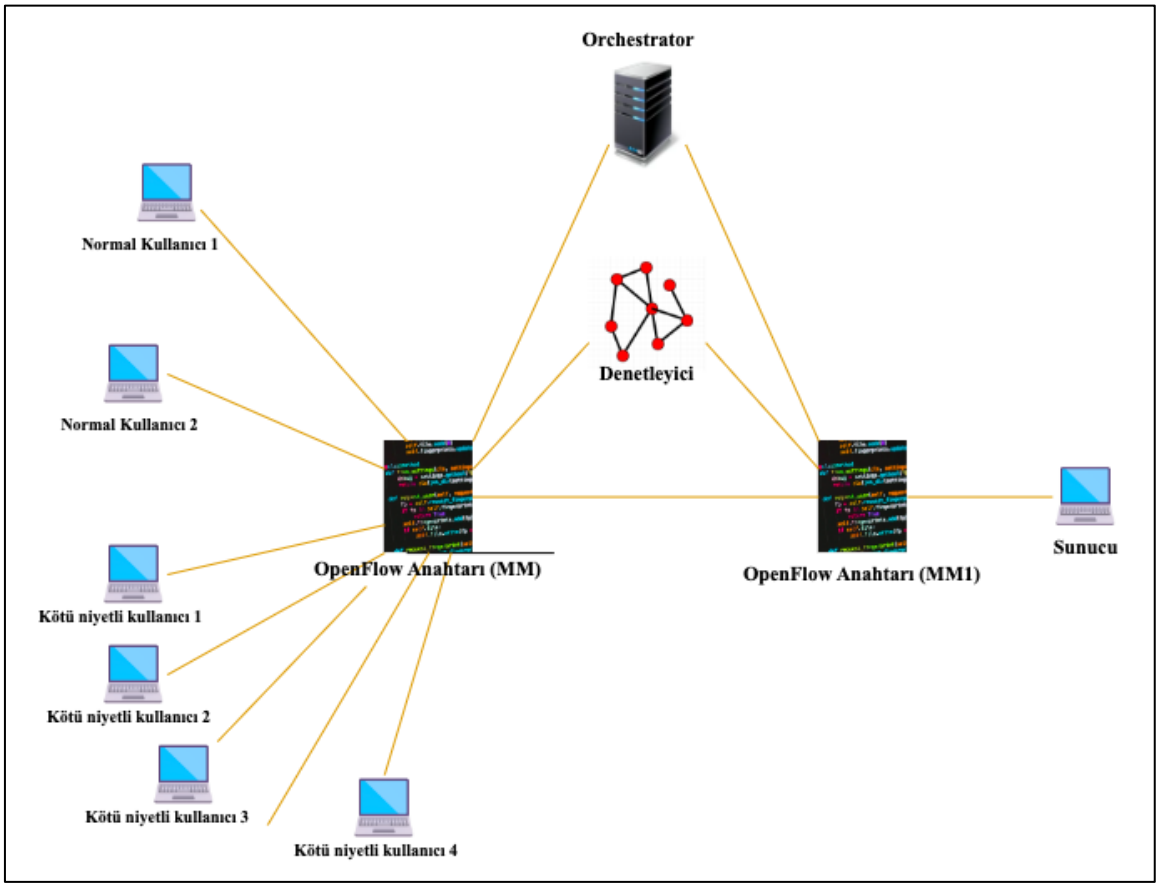
Çizelge 2.11. Simülasyon savunma gücü performans sonuçları

	Duyarlılık	Yanlış Pozitif Oran	Doğruluk Oranı
IP Kandırma Saldırısı	% 99,8	%0	%99,9
SYN Flood Saldırısı	%99,6	%0	%99,8
UDP Flood Saldırısı	%99,6	%0	%99,8
ICMP Yansıma Saldırısı	%99,7	%0	%99,8
Yayın Yükseltme Saldırısı	%99,7	%0	%99,8

MiddleModule sisteminin savunma gücü performansı varyasyonu, "*saturationThreshold*", "*maliciousThreshold*" ve "*windowSize*" olmak üzere üç yapılandırma değişkenine göre değerlendirilir. Bu amaçla aşağıdaki şekilde gösterilen test senaryosu OMNET++ ile oluşturulmuştur. "*SaturationThreshold*" ve "*maliciousThreshold*" değerlendirmeleri sırasında, kötü amaçlı kullanıcı düğümleri, IP Spoofing olmadan yalnızca SYN Flood trafiği oluşturur. Bu testlerde, windowSize'a beklenen optimum değer olan 20 atanır. Daha sonra

diğer parametreler optimum değerlerine ayarlanır ve *windowSize*'ın etkisi SYN Flood trafiği ile değerlendirilir.

Son olarak, kötü amaçlı kullanıcı düğümleri, ICMP Yansıma saldırısı oluşturur ve bu saldırı türü için de *windowSize*'ın etkisi değerlendirilir. Testler 10 tur olarak uygulanır ve her turda hem kötü amaçlı kullanıcı hem de normal kullanıcı olmak üzere her ana bilgisayar düğümü, sunucu düğümüne doğru 200 paket üretir.



Şekil 2.13. Konfigürasyon değişkenleri testi ile savunma gücü değişimi senaryosu

Yasal trafik, TCP protokolü kullanılarak oluşturulur. Farklı TCP bağlantıları; 3 yönlü el sıkışma, veri aktarımı ve FIN/ACK mesaj alışverişi dahil; NSL-KDD veri seti kullanılarak oluşturulur. İyi huylu bir bağlantı için veri kümesinde sıfır olmayan bir yük uzunluğu sağlanmışsa, söz konusu bağlantı için normal kullanıcı düğümü ile sunucu düğümü arasında bu tür bir veri aktarımı gerçekleşir. İyi huylu bir bağlantı için veri kümesinde sağlanan yük uzunluğu sıfırsa, o bağlantıdaki düğümler arasında veri aktarımı gerçekleşmez; yalnızca 3 yollu el sıkışma ve FIN/ACK mesaj alışverişi gerçekleşir.

Çizelge 2.12. Yapılandırma değişkeni varyasyonu

	SynFlood Threshold (maliciousThreshold)	saturation Threshold	windowSize	Malicious Traffic Type
1. Test	0,8	0,15	20	SYN Flood
2. Test	0,8	0,75	20	SYN Flood
3. Test	0,2	0,25	20	SYN Flood
4. Test	0,4	0,25	20	SYN Flood
5. Test	0,5	0,25	20	SYN Flood
6. Test	0,7	0,25	20	SYN Flood
7. Test	0,99	0,25	20	SYN Flood
8. Test	0,8	0,25	5	SYN Flood
9. Test	0,8	0,25	20	SYN Flood
10. Test	0,8	0,25	100	SYN Flood
11. Test	0,8	0,25	5	ICMP Reflection
12. Test	0,8	0,25	20	ICMP Reflection
13. Test	0,8	0,25	100	ICMP Reflection

Çizelge 2.12'de gösterilen her senaryo için 10 turluk test uygulanır. Savunma gücü sonuçları her tur için ölçülür ve ortalama doğruluk ve yanlış pozitif oranları hesaplanır. Çizelge 2.12'de gösterildiği gibi, ilişkili değişkenler için farklı aralıklardan farklı değerler seçilir. Bu test sonucunda MiddleModule sisteminin defans gücü ile konfigürasyon değişkenleri arasındaki ilişki değerlendirilir ve Şekil 2.14'de de ağ topolojisi için ideal konfigürasyon değişkeni değerleri bulunur.

Şekil 2.14'te gösterildiği gibi, MiddleModule sisteminin savunma gücü konfigürasyon değişkenlerine göre değişir. Konfigürasyon değişkenleri hakkında aşağıdaki üç sonuç çıkarılabilir:

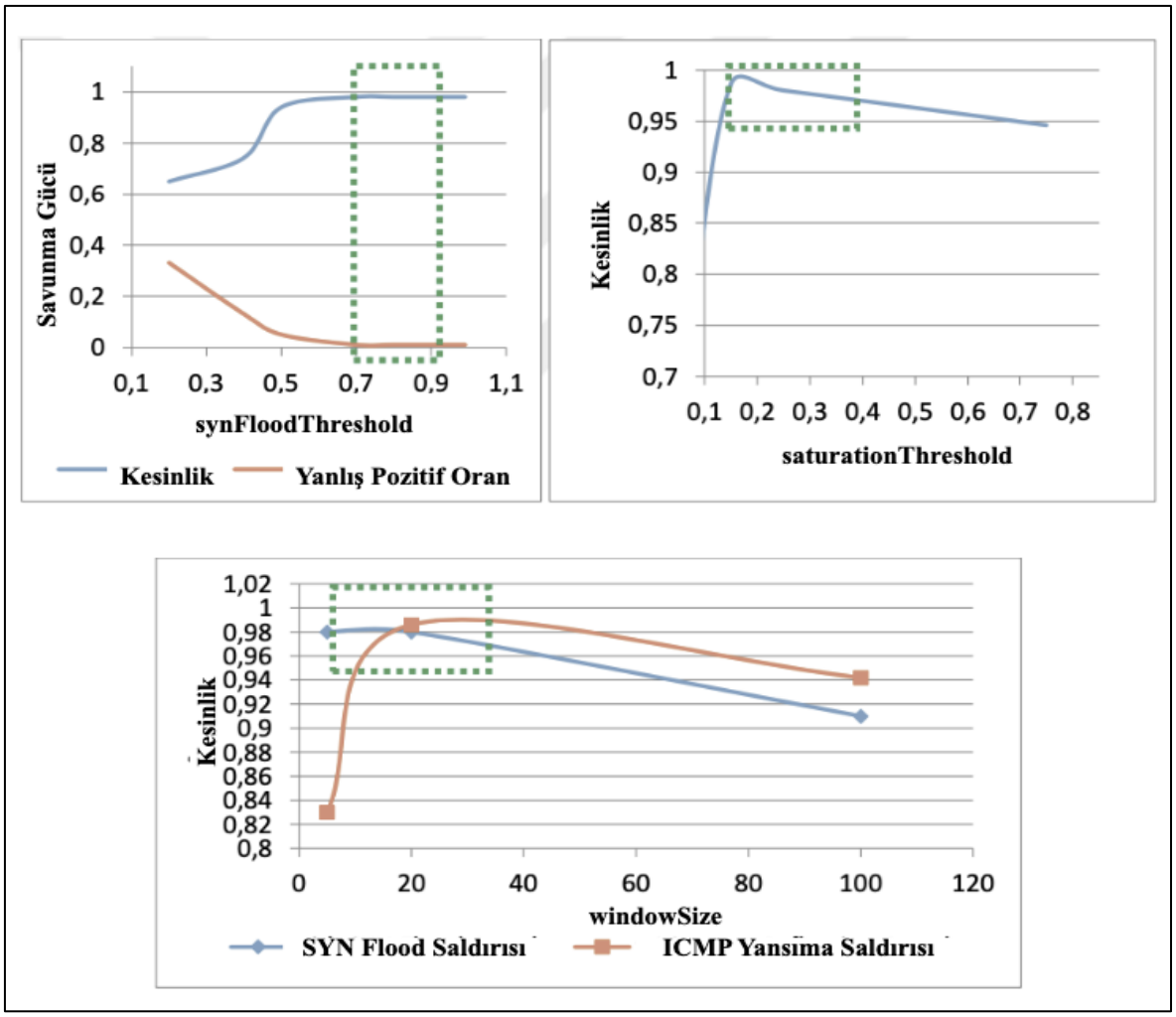
İlk olarak, yüksek yanlış pozitif oranlarından kaçınmak için "*synFloodThreshold'a*" düşük bir değer atanmamalıdır. Bu değişken, örneğin 1.0 gibi yüksek bir değere ayarlanırsa, kötü

amaçlı trafikte beklenmeyen herhangi bir değişiklik meydana gelirse sistemin algılama oranı düşebilir. Test sonuçları bu noktayı vurgulamasa da bu değişkenin çok yüksek ayarlanması da tespit algoritmalarının yapısı gereğince önerilmez. Bu nedenle, "*synFloodThreshold*" değişkeni orta ila yüksek aralık arasında bir değere ayarlanabilir. İkinci olarak, olgunlaşmamış kararları önlemek için doygunluk eşiği kullanılır; bu nedenle çok düşük ve çok yüksek ayarlanmamalıdır.

Algılama algoritmaları, herhangi bir karar vermeden önce doygunluk koşulunu sağlamak için yeterli sayıda paket alana kadar bekler. Bu değişken çok yüksek ayarlanırsa, MiddleModule sistemi birçok kötü amaçlı paketi kaçırır. Bu nedenle, orta ila düşük aralık arasında bir değere ayarlanmalıdır.

Son olarak, ağ topolojisine göre "*windowSize*" değişkeni seçilmelidir. Belirli bir topoloji için çok düşük ayarlanırsa, algılama algoritmalarının doğruluğu azalır. Çok yükseğe ayarlanırsa algılama algoritmaları en son paketleri analiz edemez ve bu da sistemin algılama performansını düşürür. Bu sonuçlara ek olarak, testlerimizden çıkarılan iki önemli sonuç vardır; doğruluk ve yanlış pozitif performanslar, optimum noktaların yakınında sürekli olarak değişir; bu nedenle, kesin optimum değerleri kullanmak hayati önem taşımaz.

Konfigürasyon değişkenlerini optimum noktalara yakın ayarlamak, yüksek performanslı sonuçlar almak için yeterlidir. İkinci olarak, savunma gücü konfigürasyon değişkenleriyle tahmin edilebilir bir şekilde değişir. Dolayısıyla, bu değişkenler için optimum değerler ağ topolojisi dikkate alınarak tahmin edilebilir. Bu testte, verilen ağ topolojisi için optimum değerler Şekil 2.14'de yeşil noktalarla gösterildiği gibi bulunur.



Şekil 2.14. Konfigürasyon değişkenlerine göre savunma gücü varyasyon sonuçları

Çizelge 2.13. Test senaryosu için optimum yapılandırma değişkenleri

windowSize	20-40
saturationThreshold	0,15-0,30
spoofThreshold	2
synFloodThreshold	0,70-0,90
floodThreshold	0,70-0,90

Öykünme testlerinde en iyi performansı elde etmek için yapılandırma değişkenleri Çizelge 2.13'te gösterildiği gibi ayarlanır.

MİNİNET kapsamında değerlendirme

OMNET++ gerçekçi test sonuçları sağlasa da daha önce de belirtildiği gibi, simülatörlerin güvenilirlik açısından bazı sınırlamaları vardır. Daha güvenilir ve daha doğru sonuçlar sağlamak için performans testlerinin bir kısmı MİNİNET platformunda tekrarlanır. Bu testler, farklı paket üretim hızları kullanarak MiddleModule sisteminin sistem performans düşüşünü ve savunma gücünü tespit etmek için uygulanır.

Sistemin ölçeklenebilirliği de ayrı bir sorun olmakla birlikte ayrıca MİNİNET'te toplanan sonuçlar, önerilen sistemin benzer çalışmalarla performans karşılaştırması için kullanılmaktadır. Testlerimizde Mininnet'Te sağlanan MİNİNET VM görüntüsünü kullanarak VirtualBox üzerine kurulmuştur. MİNİNET'i çalıştıran sanal makine Çizelge 2.14'te gösterilen özelliklere sahiptir. Bu testlerde MİNİNET sürüm 2.2.2 ve OpenFlow sürüm 1.0 kullanılmıştır. Test senaryolarında, OpenFlow uyumlu ağ düğümleri olarak Open vSwitch 2.0.2 ve OVS-Controller 2.0.2 kullanılmıştır.

Çizelge 2.14. VirtualBox özellikleri

VirtualBox Version	5.2.6 r120293
Operating System	Ubuntu – 64bit
Number of Cores	1
Installed Memory	6 GB
Upper Limit of Operation	%100

Bu testler Çizelge 2.15.'de açıklanan adımlar izlenerek uygulanır.

Çizelge 2.15. MİNİNET kapsamında test aşamaları

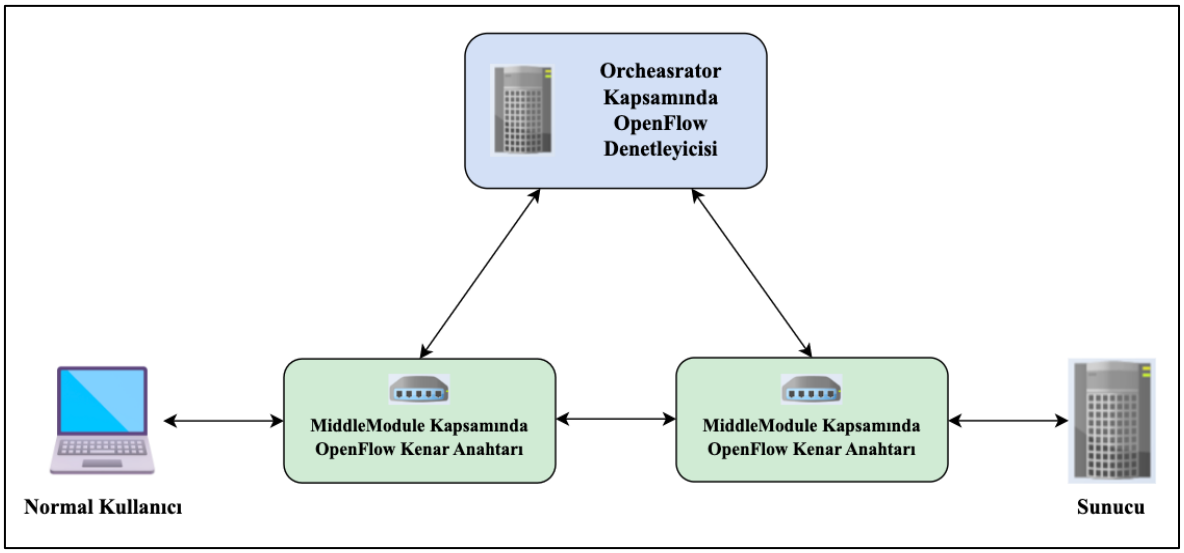
1.	MİNİNET-VM'yi (Mininet Sanal Makinesi) VirtualBox'ta çalıştırın, ardından komut satırına "*startx"* yazarak MİNİNET GUI/X11'i çalıştırın.
2.	Sonrasında, "*~/mininet/custom*" klasörüne gidin ve test için "*myTopology.py*" adıyla gerekli ağ topolojisini oluşturun.

Çizelge 2.15. (devam) MİNİNET kapsamında test aşamaları

3.	Terminal komut satırına "*sudo mn --custom custom/myTopology.py --topo myTopo --link tc -x*" yazın. Bu, "*myTopology.py*" dosyasında tanımlanan ağ için MİNİNET emülasyonunu başlatacaktır. Ayrıca her ağ düğümü için *xterm* komut satırları açılacaktır.
4.	Anahtar düğümünde (switch node) *tcpdump* komutu ile gelen trafiği okuyun ve ardından alınan paketler üzerinde MiddleModule fonksiyonlarını çalıştırın.
5.	Bağlantı noktası numarası 80 olan basit bir HTTP sunucusu oluşturmak için sunucu düğümünün komut satırına "*python -m SimpleHTTPServer 80 &*" yazın.
6.	HTTP sunucu düğümünden paket almak için Normal Kullanıcı düğümünün komut satırına "*wget -0 -destIPAddr &*" yazın.
7.	Yasal TCP trafiği oluşturmak için *scapy* kullanın. TCP trafik veri setinin "**.pcap*" dosyasını *rdpcap* komutuyla okuyun, ardından her paketin kaynak ve hedef başlık bilgisini test ağına göre değiştirin. Daha sonra *sendp* komutu ile paketi gönderin. Yasal ICMP trafiği oluşturmak için yine *rdpcap* ve *sendp* komutları kullanılabilir.
8.	Yasal TCP paketlerine yanıt mesajları oluşturmak için, *scapy*'yi karşılık gelen '*.pcap' dosyasıyla tekrar kullanın.
9.	Kötü Amaçlı Kullanıcı düğümünde kötü amaçlı trafik oluşturmak için paket oluşturma hızı *1 paket/500 ms*'den az ise *scapy* kullanın. Kötü amaçlı trafik veri setini *rdpcap* ile okuyun, paket başlıklarını uygun şekilde değiştirin ve ardından paketleri *sendp* ile gönderin. Üretim hızı *1 paket/500 ms*'den yüksek ise *packETH* kullanın. Kötü Amaçlı Kullanıcı düğümünün komut satırına "*./packETHcli -i eth0 -m 2 -d delayBetweenPacketInus -n numberOfPackets -f malwareTraffic.pcap*" yazın.
10.	Tüm düğümlerdeki paket üretiminde veya paket alımında, bu eylemin zamanını milisaniye cinsinden okumak için *strftime* işlevini de kullanın.
11.	Sunucu düğümünde, gelen trafiği gözlemlemek için *tcpdump* komutunu çalıştırın.

Sistem performansında bozulma testi

Bu test sırasında Çizelge 2.15'de açıklanan 1, 2, 3, 4, 5, 6, 10 adımlarının izlenmesi gerekmektedir. MiddleModule sistemini devre dışı bırakmak için 4. adımda herhangi bir MiddleModule işlevinin kullanılması tavsiye edilmemektedir. Adım 10'da alınan süreleri, Normal Kullanıcı düğümündeki paket istek oluşturma ve paket yanıt alma sürelerinin kıyaslanması gerekmektedir. Bunun yanı sıra paket alma sürelerinin hesaplanması önem taşımaktadır.



Şekil 2.15. MİNİNET açısından sistem performans düşüşü test senaryosu

MİNİNET platformunda Şekil 2.15'de gösterilen test senaryosu oluşturulmuş ve Sunucu düğümünde bir Hiper Metin Transfer Protokolü (Hyper Text Transfer Protocol-http) web sunucusu uygulaması çalıştırılmıştır. Öncelikle MiddleModule devre dışı bırakılır. Bu durumda Normal Kullanıcı, Sunucuya bağlanarak HTTP web sunucusundan paket alır ve ortalama paket alma süresi ölçülür. Bu test, her biri 100 paket geri getirme içeren 10 tur için uygulanmakta ve her turda testin tamamı için ortalama paket alma süresi hesaplanmaktadır. Ardından MiddleModule etkinleştirilir ve Normal Kullanıcı Sunucuya bağlanır. Paket alma süreleri yine 10 tur kapsamında her tur için 100 paket alımı için ölçülür. Normal Kullanıcının ortalama paket alma süresi, bu koşullar altında her tur için hesaplanır.

Saniye cinsinden ortalama paket alma sürelerinin birbirine çok yakın olduğu ve MiddleModule sisteminin neden olduğu paket alma süresi ek yükünün %0,2 olduğu Çizelge 2.16'da gösterilmektedir. Bu nedenle, paket alma süresi açısından MiddleModule sisteminin sistem performansı düşüşü ihmal edilebilir düzeydedir.

Çizelge 2.16. MİNİNET üzerinde sistem performansı düşüşü sonucu

	MiddleModule yok (saniye)	MiddleModule var (saniye)
Tur 1	0,02530	0,02532
Tur 2	0,02516	0,02531

Çizelge 2.16. (devam) MİNİNET üzerinde sistem performansı düşüşü sonucu

	MiddleModule yok (saniye)	MiddleModule var (saniye)
Tur 3	0,02531	0,02526
Tur 4	0,02542	0,02544
Tur 5	0,02534	0,02532
Tur 6	0,02542	0,02528
Tur 7	0,02528	0,02556
Tur 8	0,02501	0,02547
Tur 9	0,02533	0,02503
Tur 10	0,02556	0,02520
Ortalama	0,025313	0,025319
Genel	-	0,2 %

Savunma Gücü Testi

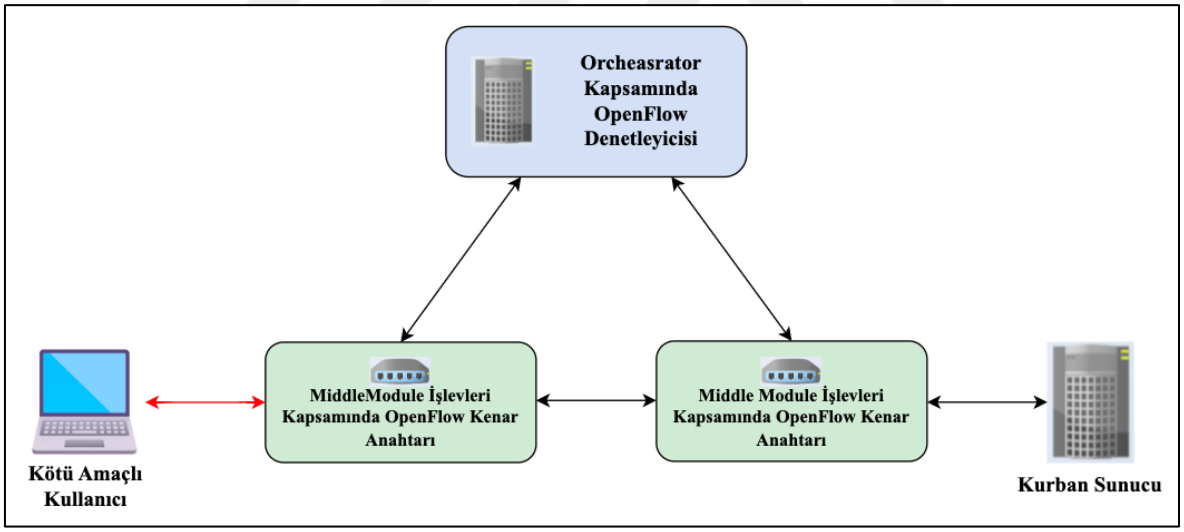
MİNİNET kapsamındaki MiddleModule sisteminin doğruluğu ve yanlış pozitif oranı; IP Kandırma saldırısı, IP spoofed/non-spoofed SYN Flood saldırısı, IP spoofed/non-spoofed

ICMP Flood saldırısı ve IP spoofed/non-spoofed UDP Flood saldırısına karşı ölçülmektedir. Ölçüm sonuçları, MiddleModule sisteminin literatürde bulunan benzer çalışmalarla karşılaştırılması için kullanılmaktadır. Bu çalışmalarda sağlanan yansıma saldırıları ve yayın yükseltme saldırıları için herhangi bir sonuç bulunmadığından, MİNİNET'teki savunma gücü testlerinde bu saldırı türleri dikkate alınmaz. Bu testlerde, senaryolar diğer çalışmalarda kullanılanlara benzer olacak şekilde seçilir. Konfigürasyon değişkenlerine yönelik olarak ideal sonuçlar elde etmek için Çizelge 2.13. üzerindeki Optimum Yapılandırma Değişkenleri kapsamında seçilmektedir. Bunun yanı sıra, *penaltyTime* (ceza süresi) 5 saniye olarak ayarlanmıştır. Testler 10 tur olarak uygulanır ve her raunt 100 saniye sürer. Doğruluk ve yanlış pozitif oranları her turda ölçülmekte ve ölçüm sonuçlarının ortalaması 10 tur

kullanılarak hesaplanmaktadır. Bu testlerde, daha önce açıklandığı gibi, yasal trafik MAWILab veri seti kullanılarak üretilmektedir.

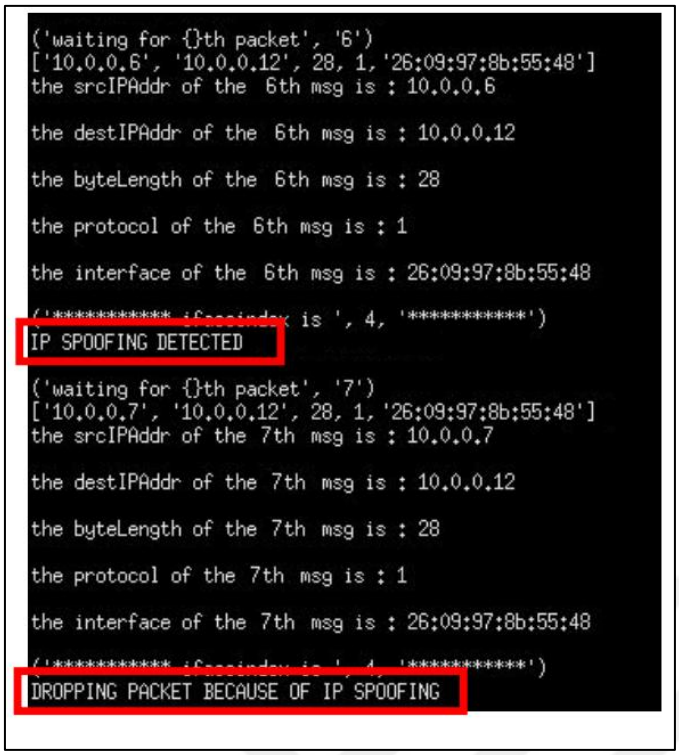
Şekil 2.16'da temel savunma gücü testleri için test senaryosu görülmektedir. Bu test sırasında Çizelge 2.15.'de açıklanan 1, 2, 3, 4, 7, 8, 9, 10, 11 adımlar izlenir. Savunma gücü ölçümü için, 11. adımda Kötü Amaçlı Kullanıcı düğümleri ve Normal Kullanıcı düğümleri tarafından oluşturulan paketleri ve ayrıca sunucu düğümü tarafından alınan paketleri gözlemlendi. Bu gözlem neticesinde doğru pozitif, doğru negatif, yanlış pozitif ve yanlış negatif değerlerinin ölçülmesi gerekmektedir.

MiddleModule sisteminin kötü amaçlı trafiği doğru bir şekilde tespit edip engellediğini gözlemlemek için öncelikle "Referans kaynağı bulunamadı" kısmında gösterilen test senaryosu oluşturulur ve farklı testler uygulanır. Her testte Kötü Amaçlı Kullanıcı tarafından farklı bir DDoS saldırı türü üretilir ve MiddleModule sisteminin doğruluğu ölçülür.



Şekil 2.16. Temel savunma gücü testleri için test senaryosu

İlk testte, Kötü Amaçlı Kullanıcıdan Kurban Sunucusuna doğru IP Kandırma ping mesajları (Flooding olmadan) üretilir. Kötü amaçlı trafik, ultra düşük hız modunda (1 paket / 10 ms) üretilir ve Şekil 2.17'de gösterildiği gibi IP Kandırma saldırısı, alınan paket sayısındaki *saturationThreshold* miktarını aştığında algılanır. Daha sonra kötü amaçlı ana bilgisayardan gelen paketler doğru bir şekilde engellenir.



Şekil 2.17. IP Kandırma saldırısı tespit sonucu

Test senaryosu (Şekil 2.16) kullanılarak; IP Kandırma ile birlikte SYN Flooding saldırısı, IP Kandırma ile birlikte ICMP Flooding saldırısı ve IP Kandırma ile birlikte UDP Flooding saldırısı, Kötü Amaçlı Kullanıcıdan ultra düşük hız modunda (1 paket / 10 ms) oluşturulur, MiddleModule sistemi bu saldırıları IP Kandırma olarak başarılı bir şekilde algılar. Saldırı tespit edildikten ve bu mesajların giriş portu zararlı olarak etiketlendikten sonra, Şekil 2.17'de görüldüğü gibi, o giriş portundan alınan paketler düşürülür. Son olarak, kalan saldırı türleri, IP Kandırma olmadan SYN Flooding saldırısı, IP Kandırma olmadan ICMP Flooding saldırısı ve IP Kandırma olmadan UDP Flooding saldırısı ultra düşük hız modunda uygulanır. MiddleModule sistemi ise saldırıları doğru bir şekilde tespit etmektedir. Bu testler diğer trafik üretim hızları kullanılarak uygulanmakta ve test koşulları değiştirilse bile sistemin algılama performansının değişmediği görülmektedir.

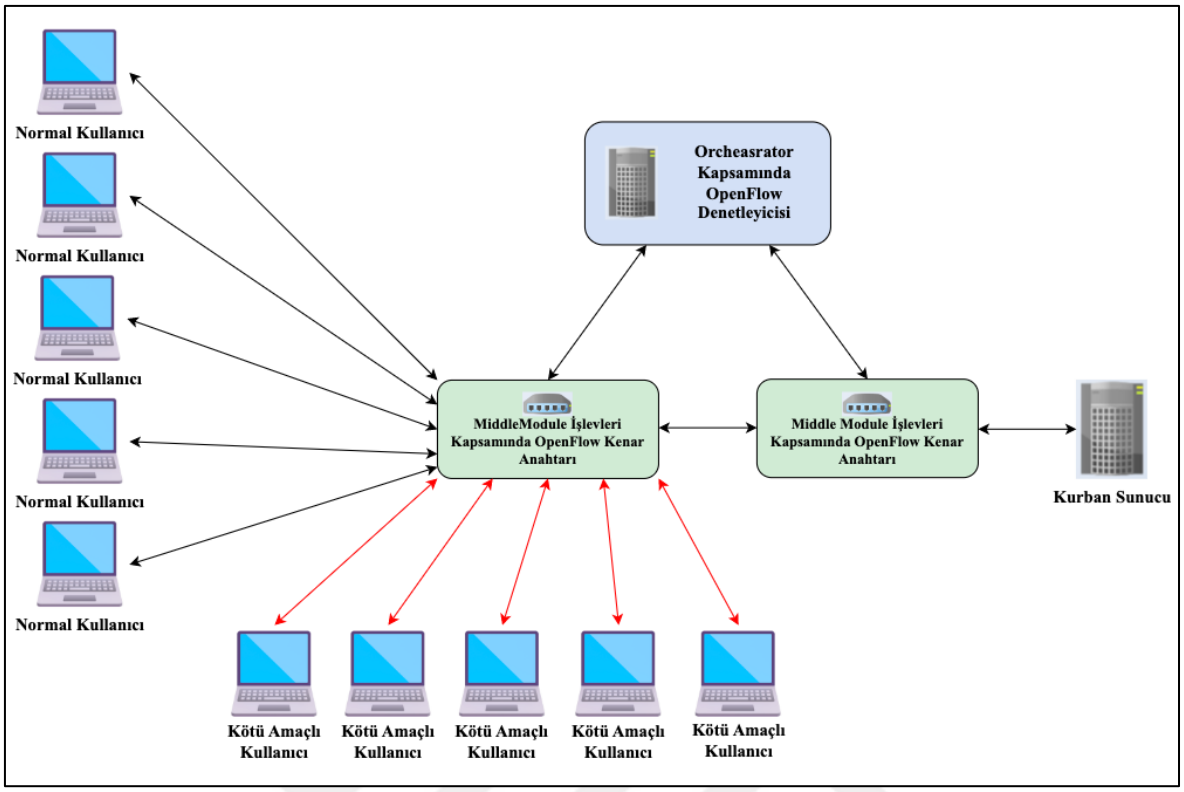
Tespit sonuçları ve tespit oranı değerleri Çizelge 2.17.'de ölçülerek gösterilmektedir. penaltyTime değeri 5sn olarak ayarlandığından her test süresi 5sn olarak kabul edilir. Böylece 100 sn süren her turda 20 farklı alt tur yapılır. Paket üretim hızı arttıkça yanlış negatif sayıları da artmaktadır. Çünkü test araçları algılama sistemine belirli bir gecikme ekler ve bu süre zarfında kötü amaçlı paketlerin bir kısmı gözden kaçmaktadır. Bu sistem uygun bir donanım platformunda uygulanırsa, bu algılama gecikmesine yönelik değerler

önemli miktarda küçülmektedir. Ancak, bu tür gecikme etkileriyle bile hassasiyet değeri %99,5 üzerinde kalarak bu oran oldukça yüksektir.

Çizelge 2.17. Temel test senaryosu ile tespit performansı sonuçları

	Duyarlılık	YN	DP
Ultra yüksek hız modu (1 paket / 2 μs)	%99,6	8114	2 491 886
Yüksek hız modu (1 paket / 10 μs)	%99,6	1714	498 286
Normal hız modu (1 paket / 200 μs)	%99,6	100	24 900
Düşük hız modu (1 paket / 2 ms)			2 489

MiddleModule sisteminin temel tespit oranına yönelik analizin ardından, daha güvenilir ve gerçekçi test senaryoları için ileri savunma gücü analizi yapılmaktadır. Şekil 2.18.'de gösterilen birkaç Normal Kullanıcı düğümü ve birkaç Kötü Amaçlı Kullanıcı düğümü içeren daha gerçekçi bir test senaryosu MİNİNET'te oluşturulmuştur. Bu testler sırasında yasal trafik; TCP paketleri ve ICMP paketleri kullanılarak oluşturulur. Bu paket türlerinin genel paketlere oranı, yasal trafiği dalgalandırmak için testler sırasında değiştirilir. TCP paket oranı %80 ile %100 arasında, ICMP paket oranı ise %20 ile %0 arasında değişir. Yasal TCP paketleri, daha önce açıklandığı gibi MAWILab veri kümesi kullanılarak üretilir. Paketler ağa enjekte edilir ve trafik hızına bağlı olarak Scapy veya packETH araçları ile Kurban Sunucusuna yönlendirilir. Bu testler sırasında, yanlış pozitif oranları en aza indirmek için *windowSize* değişkeni 100'e ve *saturationThreshold* değişkeni 0,25'e ayarlanır. *spoofThreshold* 2 olarak ayarlandı, *synFloodThreshold* ve *floodThreshold* 0,80 olarak ayarlandı ve *penaltyTime* 5 saniye olarak ayarlandı.



Şekil 2.18. Güvenilir savunma gücü testleri için test senaryosu

Bu testlerde sırasıyla şu saldırı tipleri üretilir; IP Kandırma saldırı trafiği, IP Kandırma ile SYN Flooding saldırı trafiği, IP Kandırma olmadan SYN Flooding saldırı trafiği, IP Kandırma ile ICMP Flooding saldırı trafiği, IP Kandırma olmadan ICMP Flooding saldırı trafiği, IP Kandırma ile UDP Flooding saldırı trafiği, UDP IP Kandırma olmadan Flooding saldırısı trafiği. Bir testte, tüm Kötü Amaçlı Kullanıcı düğümlerinden yalnızca bir saldırı türü oluşturulur. Kötü Amaçlı Kullanıcı düğümleri bu trafiği oluştururken Normal Kullanıcı düğümleri, daha önce bahsedildiği gibi yasal trafik üretir. MiddleModule sisteminin hassasiyeti, doğruluğu ve yanlış pozitif oranı, her saldırı türüne göre, her turun 100 saniye sürdüğü 10 tur için ölçülür. Bu 10 turun ortalama sonuçları hesaplanır. Ayrıca, sistemin paket üretim hızına yönelik bağımlılığını ölçmek için her bir test beş farklı paket üretim hızı için uygulanmaktadır. Paket oluşturma hızlarına dair modlar kapsamında doğru ve yanlış pozitif oranlar kaydedilir. *penaltyTime* değeri 5sn olarak ayarlandığından her test süresi 5sn olarak kabul edilir. Böylece 100 sn süren her turda 20 farklı alt tur yapılır. Farklı saldırı türleri için test sonuçları birbirine benzerlik göstermektedir ve tüm saldırı türleri için ortalama sonuçlar Çizelge 2.18'de belirtilmektedir.

Çizelge 2.18. MİNİNET kapsamında savunma gücü testi sonuçları

	Kesinlik	Duyarlılık	DP	YP	YN	DN
Ultra yüksek hız modu (1 paket / 2 μs)	%99,5	%98,9	7 500 000	0	78252	7 422 748
Yüksek hız modu (1 paket / 10 μs)	%99,6	%99,3	2 5000 00	0	16948	2 483 051
Normal hız modu (1 paket / 200 μs)	%99,6	%99,3	125 000	0	857	124 143
Düşük hız modu (1 paket / 2 ms)	%99,4	%98,8	7 500	0	91	7409
Ultra düşük hız modu (1 paket / 10 ms)	%98,2	%96,4	500	0	18	482

Çizelge 2.18.'de görüldüğü gibi, MiddleModule sistemi her bir trafik hızı için kötü amaçlı trafiği başarılı bir şekilde tespit edip engellediği halde herhangi bir yasal paket düşmesine neden olmaz. Yasal bir paket düşüşü olmadığından, bu test koşulları altında MiddleModule sisteminin yanlış pozitif (YP) oranı sıfır olur. Öte yandan, bazı yanlış negatifler gözlemlenir ve daha önce bahsedilen test araçlarının neden olduğu belirli gecikmeler nedeniyle paket üretim hızı arttıkça yanlış negatiflerin (YN) sayısı artar. Çizelge 2.18'de gösterildiği gibi, tüm koşullar için kesinlik %99'a yakın veya daha yüksektir. Bu test sonucu iki önemli noktaya işaret etmektedir. Öncelikle, MiddleModule sisteminin gerçek trafik veri seti ve ilgili test senaryosu ile savunma gücü, hedeflenen Ağ/Aktarım katmanında DDoS Saldırı türleri için dikkat çekicidir. İkincisi, bu sistemin savunma gücü, paket üretim hızındaki değişiklikle önemli ölçüde değişmemektedir. Bu sistem, saniyede 0,5 milyon paket veri hızında bile saldırıları başarıyla tespit eder.

İletişim ek yükü ölçümü

MiddleModule sisteminde her veri katmanına yönelik MiddleModule düğümü, kendi başına çalışır. Düğümlerin düzgün çalışması için herhangi bir komuta veya diğer düğümlerden gelen herhangi bir ek bilgiye ihtiyaç yoktur. Ancak, denetleyiciyi önleme kuralları hakkında bilgilendirmek ve en iyi azaltma performansını elde etmek için veri katmanına yönelik

MiddleModule düğümleri iki farklı mesaj gönderir. İlk olarak, bir veri katmanına yönelik MiddleModule düğümü, herhangi bir kötü amaçlı etkinlik algıladığında, denetleyiciye bu kötü amaçlı etkinlik hakkında bilgi vermek için denetleyiciye kötü amaçlı bilgi mesajı gönderir. Bu mesaj, kötü amaçlı etkinlik ilk kez algılandığında gönderilir. İkinci olarak, bir mesajın hedef tarafında kötü amaçlı bir aktivite tespit edildiğinde, bu veri katmanına yönelik MiddleModule düğümü, denetleyiciye kötü amaçlı dağıtım mesajı gönderir. Denetleyici bu mesajı aldığında, bu mesajı diğer veri katmanına yönelik MiddleModule düğümlerine dağıtır. Kötü amaçlı dağıtım mesajları kullanılarak, önceden etiketlenmiş kötü amaçlı bir ana bilgisayardan bir paket alındığında, bu paket, kötü amaçlı etkinliğin kaynağına en yakın kenar anahtarında azaltılır. Böylece kötü amaçlı paketler ağa girmeden engellenir. Her iki mesaj da başlık kısmı için 64 bayt ve veri yükü kısmı için 60 bayt olmak üzere, tek bir mesaj için 124 bayt TCP mesajlarıdır. Ayrıca, her ikisi de kötü amaçlı etkinliğin ilk kez algılanmasıyla gönderilir ve bunlar periyodik mesajlar değildir.

MiddleModule sisteminin tüm ağa iletişim yükü, Şekil 2.18'de gösterilen test senaryosuna göre hesaplanır. Bu test senaryosunda, 100 saniye süren, 5 saniye ceza süresi olan tek bir turda, bir uç anahtarı 5 farklı kötü amaçlı ana bilgisayardan DDoS mesajları alır. Bu nedenle, ağdaki kötü amaçlı bilgi mesajının toplam boyutu 124 bayt x 20 x 5 = 12,1 kb olmaktadır. Ayrıca kötü niyetli ana bilgisayarlar Yansıma saldırısı oluşturduğunda, hedef taraftaki kenar anahtarı, toplam boyutu 124 bayt x 20 = 2,42 kb olan kötü amaçlı dağıtım mesajlarını denetleyiciye gönderir. Bu mesajlar, denetleyici tarafından diğer uç anahtarına dağıtılır. Bu da toplam 12,1 kb kapasitedeki kötü amaçlı dağıtım mesajı yüküne yol açar. Bundan dolayı, bu test senaryosu için MiddleModule sisteminin toplam iletişim yükü 24,2 kb iken, ağ üzerinden geçen toplam trafik GB mertebesindedir. Bu nedenle, MiddleModule sisteminin iletişim yükü %0,1'den azdır ve bu ihmal edilebilir düzeydedir.

MiddleModule sistemi, ağ operatörünün çalışması sırasında; algılama değişkenlerini değiştirme, veri katmanına yönelik MiddleModule düğümlerinden periyodik durum mesajları alma, veri katmanına yönelik MiddleModule düğümlerine önleme komutları gönderme gibi ek yetenekler sağlar. Bu yeteneklerin kullanılması isteğe bağlı olduğundan ve isabetli savunma hareketleri için gerekli olmadığından, bu hareketlerin neden olduğu ek yük bu tezde değerlendirilmemiştir.

Ölçeklenebilirlik analizi

MiddleModule sisteminin ölçeklenebilirliği, algılama performansı ölçeklenebilirliği açısından değerlendirilir. Bu, farklı sayıda ana bilgisayarla ve farklı trafik enjeksiyon hızlarıyla değerlendirilir. Ayrıca gerekli donanımın ölçeklenebilirliği de tartışılmaktadır.

MiddleModule sisteminin algılama oranı performansı, tek bir kenar anahtarına bağlı ana bilgisayarların sayısına ve bu ana bilgisayarların trafik enjeksiyon hızına göre ölçeklenir. Tüm paketler aynı algılama algoritması seti ile değerlendirildiğinden, bu beklenen bir değerdir. Ayrıca, MiddleModule sistemi gelen tüm paketleri işlemek için tek bir Paket İşleme Bloğu kullanılmasını önerdiğinden, gelen paket sayısı arttıkça yanlış negatif oranı değişkenlik gösterebilir. Ancak, duyarlılık oranı veya yanlış pozitif oranı bu tür koşullar altında kritik derecede değişmez. Değerlendirme testlerinde önerildiği gibi, Şekil 16. ve Şekil 18.'de gösterilen test senaryoları ile uygulanan; bağlı ana bilgisayar sayısı birden ona çıktıkça ve gelen trafik hızı 1 paket / 10 ms'den 1 paket / 1 μs'a değiştirildiğinden, MiddleModule sisteminin kesinlik değeri %99'un üzerinde kalırken yanlış pozitif oranı %0'a yakındır. Ayrıca kenar anahtarlarının kendi kendine çalışması durumunda ağdaki kenar anahtarı sayısı artmasına yönelik olarak her bir kenar anahtarının performansı değişmez.

Bir kenar anahtarına MiddleModule işlevselliği sağlamak için gereken donanım, o kenar anahtarının giriş bağlantı noktası sayısına göre değişir. Paket Aynası ve Saldırı Önleme Bloğunun boyutu, giriş port numarasına göre, Paket İşleme Bloğunda ise ayrılan kaynaklara göre artar. Tahsis edilen donanımın ölçeklenebilirliği sistem tasarımı ile garanti altına alınmıştır.

Uygulama kompleksi analizi

MiddleModule sistemi, SDN'lerin veri katmanı cihazlarına izleme, tespit ve azaltma bloklarının yerleştirilmesini önerir. Bu sistem, çekirdek anahtarların herhangi bir değişiklik yapılmadan kullanılmasıyla ağdaki kenar anahtarların değiştirilmesini önerir. Bu tez sayesinde ağdaki tüm paketler savunma sistemi tarafından işlenirken, veri katmanı cihazlarının sadece bir kısmı üzerinde değişiklik yapılmıştır. Değiştirilen anahtar sayısının tüm anahtar sayısına oranı ağ topolojisine bağlıdır. Bu oran 1 veya 0,5'e yakın olabilir.

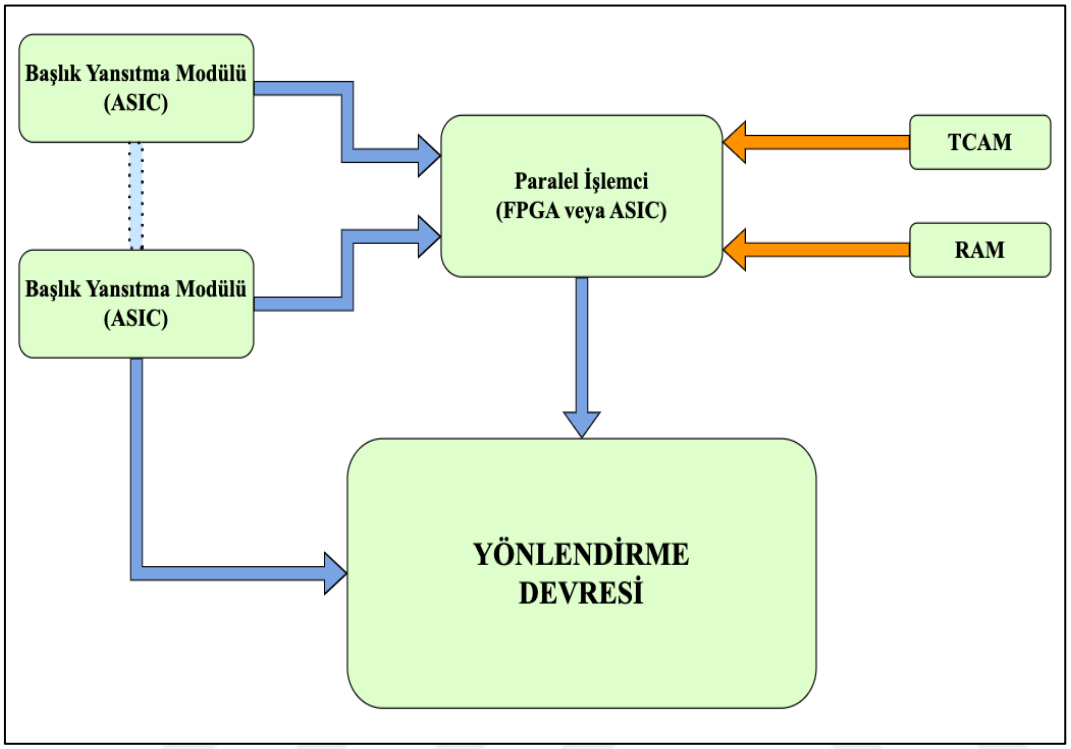
Kenar anahtarlarındaki değişiklik, karmaşık veya kapsamlı bir değişiklik değildir. MiddleModule sistemi çok hafif algılama algoritmaları kullandığından, bu algoritmaları çalıştırmak için gereken donanım kaynakları ve işlem gücü fazla olmamaktadır. Algılama algoritmaları tarafından gelen paketlerin yalnızca başlık kısmı işlendiğinden, gerekli bellek genişliği ve bant genişliği küçüktür. Bunun yanı sıra, algılama işlemleri sorumlu ana bilgisayarlara çok yakın olarak gerçekleştirilir. Kenar anahtarlar, yalnızca en son paketleri analiz ederek bu ana bilgisayarların ürettiği trafikteki herhangi bir değişikliği algılayabilir. Bu nedenle, her ana bilgisayara yönelik olarak izleme sonuçlarını depolamak için gereken tablo boyutları da küçüktür.

Orchestrator uygulaması, veri katmanına yönelik MiddleModule düğümlerinden gelen mesajları analiz etmekten ve bu düğümlere yanıtlar oluşturmaktan sorumludur. Bu mesajlar yüksek frekansta üretilmez. Yük bölümlerinde önceden tanımlanmış bazı bilgiler sağlayan standart TCP mesajlarıdır. Bu nedenle, hafif bir Orchestrator uygulaması bu mesajları doğru bir şekilde işleyebilir ve uygun yanıtları oluşturabilir.

Kenar anahtarlarına MiddleModule işlevselliğini sağlamak için, Şekil 2.19'da gösterilen değişikliğin tüm kenar anahtarlarına uygulanması önerilir. Paket izleme amacıyla, işleme bloğu tarafından gelen paketlerin başlık kısımlarını yansıtmak için ASIC tabanlı bir modül kullanılabilir. Başlık yansıtma modülü, sistem performansı düşüşünü en aza indirmek için yüksek hızlarda çalışmalıdır. Paket işleme algoritmaları da o kadar karmaşık değildir ve yüksek saat hızları gerektirmez. İşleme bloğu, aynı anda farklı başlık yansıtma modüllerinden farklı paketler alabilir.

İşlem bloğunun güç tüketimi minimum düzeyde olmalıdır. Bu nedenle, Alanda Programlanabilir Kapı Dizileri (Field Programmable Gate Array - FPGA) gibi bir paralel işlemci birimi, her bir kenar anahtar için bir paralel işlem bloğu olarak önerilmiştir. İşlem bloğunun bellek birimi, Rastgele Erişimli Bellek (Random Access Memory - RAM) veya Üç İçerik Adreslenebilir Bellek (Ternary Content Addressable Memory - TCAM) kullanılarak oluşturulabilir. Küçük bellek blokları, bu amaç için yeterli olacaktır. Çünkü her bağlı ana bilgisayar için tutulan tabloların boyutu sınırlıdır ve algılama algoritmalarının gerekli parametre sayısı küçüktür. Saldırı önleme işlevi, OpenFlow komutları kullanılarak uygulanabilir. MİNİNET üzerindeki değerlendirmelerde bu teknikten faydalanılmaktadır.

Paket İşleme Bloğu, saldırı önleme kurallarını uygulamak için OpenFlow komutları göndererek Yönlendirme Devresinin akış tablosunu değiştirebilir.



Şekil 2.19. Veri katmanı cihazı MiddleModule uygulama önerisi

Uzlaşma ve yetenek tartışması

MiddleModule sistemi, ağ genelinde dağıtılan ve bağımsız bir şekilde çalışan veri katmanı izleme, algılama ve önleme düğümlerini kullanır. Bu nedenle MiddleModule sisteminde, denetleyici tabanlı savunma sistemlerinden farklı olarak herhangi bir hata noktası yoktur. Bir savunma düğümü herhangi bir nedenle arızalanırsa, diğer düğümler sorunsuz bir şekilde çalışmalarına devam edebilir. Ayrıca, daha önce belirtildiği ve test edildiği gibi, MiddleModule sistemi farklı protokoller ve trafik oranları ile çalışabilir. Dolayısıyla dalgalanan trafik veya yüksek paket üretim hızı, MiddleModule sisteminde herhangi bir arızaya sebebiyet vermez. Ayrıca savunma düğümleri kötü amaçlı bilgi mesajları gönderebildiği ve istenirse denetleyiciye periyodik durum mesajları da gönderebildiği için; denetleyici, her bir veri katmanı düğümünün kararlarını analiz edebilir. Herhangi bir savunma düğümünün güvenliği kötü amaçlı bir kullanıcı tarafından ele geçirilirse ve düğüm arızalanmaya başlarsa, denetleyici arızalı olan bu düğümü algılayabilir. Denetleyici daha

sonra sorunu çözebilir. Bunun yanı sıra, algılama algoritması parametrelerini değiştirerek, beyaz liste ve kara listeleri tanımlayarak düğümü tamamen devre dışı bırakabilir. Veri katmanı düğümlerinin yapılandırma kanalları, yani veri katmanı düğümleri ile denetleyici arasındaki kanal, düğümlerin zayıf noktası olabilir. Bununla birlikte, MiddleModule sisteminde kenar anahtarlar, normal giriş bağlantı noktalarından farklı olan özel yapılandırma kanalı bağlantı noktalarına sahiptir. Gerekirse, bu yapılandırma kanalındaki iletişim; şifreleme veya başka bir yöntem kullanılarak güvence altına alınabilir.

2.4.3. Karşılaştırmalar ve genel tartışmalar

Bu bölümde MiddleModule sistemi, daha önce bahsedilen performans ölçütleri kullanılarak literatürde bulunan benzer çalışmalarla karşılaştırılmıştır. Bu tezde karşılaştırma yapılırken aşağıdaki veri katmanı tabanlı DDoS savunma sistemleri ele alınmıştır; AVANT-GUARD (Shin vd., 2013), CIDS (Chen ve Yu, 2016), StateSec (Boite vd., 2017) ve SDNScore (Kalkan vd., 2017).

MiddleModule sistemi ve tüm çalışmalar DDoS saldırılarına karşı önerilmiş olsa da yukarıdaki çalışmalarda hedeflenen saldırı türleri birbirinden biraz farklıdır. AVANT-GUARD sistemi sadece SYN Flood DDoS saldırı türünü dikkate alır, CIDS sistemi SYN Flood saldırılarını ve Flooding saldırılarını (ICMP Flooding ve UDP Flooding) dikkate alır, StateSec sistemi ise sadece Flooding saldırılarına karşı test edilir. SDNScore, SYN Flood saldırısı ve Flooding saldırılarına odaklanır. MiddleModule sistemi ise IP Kandırma saldırısı, SYN Flood saldırısı, kandırma Flooding saldırısı, kandırma olmadan Flooding saldırısı, Yansıma saldırısı ve Broadcast Amplification saldırılarına karşı tasarlanmış ve test edilmiştir. Bu nedenle, MiddleModule sistemi daha geniş bir Ağ/Aktarım katmanında DDoS saldırı setini kapsamaktadır.

Çalışmaların savunma güçleri önerdiğimiz yöntemle karşılaştırılmıştır. Bu amaçla hem ideal hem de ideal olmayan koşullar altında kıyaslama yapılmaktadır. İdeal koşul, saldırıların iyi tanımlanmış paket özniteliklerine (paket uzunlukları, protokoller vb.) sahip olduğu ve normal hızda üretildiği anlamına gelir.

Veri katmanı tabanlı savunma yaklaşımı, Ağ katmanında DDoS saldırılarına karşı etkili bir yöntem olduğundan, Çizelge 2.19'da gösterildiği gibi, karşılaştırılan tüm yöntemler ve MiddleModule sistemi ideal koşullar altında savunma gücü değerlerine sahiptir.

Çizelge 2.19. İdeal koşullar altında savunma gücü karşılaştırması

	ICMP veya UDP Flooding Saldırıları		SYN Flood Saldırıları	
	Yanlış Pozitif Oran	Duyarlılık	Yanlış Pozitif Oran	Duyarlılık
MiddleModule	%0	%99,3	%0	%99,3
SDNScore	%1	%100	%0,02	%100
StateSec	%0	%100	-	-
CIDS	%3.4	%96,3	%2,3	%94,8

AVANT-GUARD sistemi için verilen sayısal savunma gücü değerleri olmadığı için savunma gücü karşılaştırmasında kullanılmamaktadır. Çizelge 2.19.'da görüldüğü gibi, MiddleModule sistemi bu saldırılara karşı diğer çalışmalarla kıyaslandığında benzer savunma gücü performansı sağlamaktadır.

Ancak ideal olmayan koşullarda, MiddleModule sisteminin savunma gücü performansı diğer çalışmalara göre belirgin şekilde yüksektir. Kötü amaçlı trafik oluşturma oranı düşük olduğunda, saldırı yoğunluğu azaldığından duyarlılık sonuçları değişir. StateSec, CIDS ve MiddleModule sistemleri bu koşullar altında test edilir. StateSec 1pckt/10ms ile test edilir. CIDS, 1pckt/10ms ile 1 pckt/2ms arasında bir değerle test edilir. MiddleModule, 1pckt/10ms paket üretim hızlarıyla test edilir. StateSec ve CIDS sistemlerinin algılama oranı yaklaşık %80'e düşerken, MiddleModule sisteminin algılama oranının ideal koşullarda toplanan sonuçlara yakın olan %96'nın üzerinde kalmaktadır. SDNScore, düşük üretim hızlarında test edilmez. Ancak hem iyi tanımlanmış hem de rastgele paket özniteliklerine (protokol tipi, hedef port numarası, paket bayt uzunluğu, vb.) sahip saldırılar kullanılarak test edilir. Rastgele niteliklere sahip saldırılar, SDNScore çalışmasında jenerik saldırılar olarak adlandırılır. Nominal paket üretim hızlarında jenerik saldırılar kullanılarak savunma sistemi test edildiğinde SDNScore'un algılama oranı önemli ölçüde düşerek yaklaşık %75,0'e düşer.

Bunun yanı sıra, MiddleModule algılama algoritmaları, bu özelliklerin değerlerinden bağımsız olarak, paket özelliklerinin hacmini ve varyasyonunu analiz eder. Bu nedenle, iyi bilinen değerlere veya genel değerlere sahip paket özniteliklerine sahip kötü amaçlı bir paket oluşturulduğunda, aynı performans sonuçları elde edilir. MiddleModule sisteminin emülasyonu sırasında, gerçek trafik kullanılarak zararlı paketler üretilir ve Dağıtılmış Hizmet Reddi paket öznitelikleri rastgele oluşturulur. Dolayısıyla, genel kötü amaçlı paket özniteliklerine sahip MiddleModule sisteminin algılama oranı %99,3'tür.

MiddleModule sisteminin ideal ve ideal olmayan koşullarda oldukça iyi performans gösterdiği, ideal olmayan koşullarda ise diğerlerinin savunma gücünün önemli ölçüde azaldığı gözlemlenmektedir. Sistem performans düşüşü, özellikle veri katmanı tabanlı savunma sistemleri için diğer önemli bir husustur çünkü bu tür sistemlerde ağdaki tüm paketler, savunma sisteminin neden olduğu performans düşüşünü yaşamaktadır. Bir savunma sistemi, kötü amaçlı etkinlikleri doğru bir şekilde algılamalı ancak aynı zamanda nominal kullanıcının bağlantısında minimum gecikmeye neden olmalıdır. Nominal bir kullanıcının paket alma süresinde MiddleModule sistemi ve AVANT-GUARD sisteminin neden olduğu gecikmeler Çizelge 2.20'de karşılaştırılmıştır.

Çizelge 2.20. Paket alma zaman gecikmesi karşılaştırması

	MiddleModule	AVANT-GUARD
Paket alma süresinde gecikme oranı	%0,2	% 1,86

Savunma sistemi olan ve olmayan nominal bir kullanıcının paket alma süreleri kaydedilir ve savunma sisteminin neden olduğu paket alma gecikmesi hesaplanır. AVANT-GUARD sistemi için bu test uygulandığında paket alma süresi %1,86 daha uzun olarak ölçülmektedir. MiddleModule sistemi içerisinde ise nominal bir kullanıcının paket alma süresi %0,2 daha uzun olarak ölçülmektedir. Bu nedenle, MiddleModule nominal bir kullanıcının paket alma süresinde AVANTGUARD sistemi ile karşılaştırıldığında dikkate değer bir gecikmeye neden olmaz. Bu metrik kritik bir ölçü olmasına rağmen diğer savunma sistemleri bu tür test sonuçları vermemektedir.

Bu sistemlerin sistem performansı düşüşünü değerlendirmek için, yalnızca savunma sisteminin neden olduğu iletişim ek yükünü sağlarlar. Diğer sistemlerin iletişim ek yükü değerleri %1'den azdır; Öte yandan, MiddleModule sisteminin iletişim ek yükünün % 0,01'den az olması da diğer çalışmalarla karşılaştırıldığında dikkat çekicidir. Aslında, MiddleModule sistemindeki savunma düğümleri, hafif algılama algoritmaları ile bağımsız bir şekilde çalıştığından, MiddleModule sistemi için bu tür sistem performans düşüşü sonuçları beklenmektedir.

Ağ/Aktarım temelli DDoS saldırılarının en önemli özelliklerinden biri normal koşullardan daha yüksek oranlarda üretilmeleridir. Genel olarak, bir ana bilgisayar ağa Ağ/Aktarım temelli DDoS trafiği oluşturuyorsa mümkün olan en yüksek üretim hızına sahip trafiği üretmektedir. Bu nedenle, yüksek trafik üretim oranları kullanan önerilen savunma sistemlerini değerlendirmek kritik öneme sahiptir. Bir savunma sistemi, yalnızca bugünün ihtiyaçlarından daha düşük paket hızlarında test edilirse, o sistemin savunma gücü, gerçek ağ oluşturma koşullarında anlamsız hale gelebilir. Aşağıdaki çizelgede önerilen savunma sistemlerinde bildirilen minimum ve maksimum paket üretim hızları kullanılarak oluşturulmuştur. Çizelge 2.21’de kullanılan bazı paket üretim hızları, CIDS'de olduğu gibi ilgili çalışmada doğrudan sağlanmaz, ancak bu değerler bu çalışmalarda sağlanan bilgilerden çıkarılır.

Çizelge 2.21. Trafik oluşturma oranları karşılaştırması

	En az paket oluşturma hızı	En fazla paket oluşturma hızı
StateSec	10 ms (100 adet/saniye)	1200 μs (833 adet/saniye)
MiddleModule	10 ms(100 adet/saniye)	2 μs (5000000 adet/saniye)
CIDS	1 ms (100 adet/saniye)	200 μs (5000 adet/saniye)
AVANT-GUARD	10 ms (100 adet/saniye)	1 ms (1000 adet/saniye)

Çizelge 2.21’de de görüldüğü üzere, minimum paket üretim hızları birbirine benzer ancak maksimum hızlar farklıdır. MiddleModule sisteminde değerlendirme sırasında kullanılan maksimum paket üretim hızı diğer çalışmalara göre yüz kat daha fazladır. Ortalama paket uzunluğunun 250 bayt olduğu varsayılırsa, 5 kpps sadece 10 Mbps'ye, 500 kpps ise 1

Gpbs'ye karşılık gelir ki bu günümüzün gelişmiş ağ teknolojilerinde daha gerçekçidir. Bu nedenle, MiddleModule sistemi daha yüksek paket üretim hızlarında çalışacak şekilde tasarlanmış ve ayrıca daha gerçekçi koşullar altında test edilmiştir. MiddleModule sistemi ile diğer çalışmaların karşılaştırılması sırasında başka bazı hususlar da dikkate alınır. Yüksek performanslı bir savunma sistemi sağlamak önemli olmakla birlikte, DDoS saldırıları, SDN ve önleme sistemlerinin kritik özelliklerini karşılayan bir sistem sağlamak da önemlidir.

AVANT-GUARD sistemi, yalnızca sahte IP adresine sahip SYN Flooding saldırılarını algılar. Ancak, daha önce açıklandığı gibi, diğer bazı DDoS saldırı türleri SDN için tehlikeli olabilir. Diğer Flooding saldırılarını önlemek için ağda ek bir algılama sistemi uygulanmalıdır. CIDS sistemi sadece DDoS olup olmadığı bilgisini verir, ancak saldırının kaynak ana makinelerini ve kurban ana bilgisayarlarını sağlayamaz. Ancak bu bilgi olmadan bir saldırı etkili bir şekilde önlenemez. Ağdaki DDoS trafiği nominal trafikten çok daha güçlüyse, StateSec sistemi düzgün çalışamaz. StateSec sisteminin algılama performansı bu koşul altında önemli ölçüde düşer. Ancak genel olarak bir DDoS saldırısı sırasında DDoS trafiği çok daha güçlü hale gelir. Ayrıca saldırgan sayısı belirli bir değeri aşarsa kötü amaçlı faaliyetlerin kaynağını tespit edemez.

Kötü amaçlı paketler iyi tanımlanmış özniteliklere sahip değilse, SDNScore sistemi doğru şekilde çalışamaz, ancak kötü amaçlı paketler de rasgele özniteliklere sahip olabilir. Ayrıca, oldukça karmaşık bir algılama mekanizması önerir, ancak bu algılama mekanizması sisteminin neden olduğu bağlantı gecikmeleri hakkında herhangi bir test sonucu sağlamaz. Ayrıca, daha yüksek paket üretim hızlarında algılama performansında değişiklik sağlamaz. Bu yönler, DDoS savunma sistemleri için kritik öneme sahiptir ve bu yönler dikkate alındığında kapsamlı bir savunma sistemi iyi performans göstermelidir.

MiddleModule sistemi bu çalışmalarla bu yönleriyle karşılaştırıldığında dikkat çekici özelliklere sahiptir. MiddleModule sistemi, birkaç Ağ/Aktarım temelli saldırı türü algılar. Kötü amaçlı bir etkinliğin hem kaynak hem de kurban ana bilgisayarlarını sağlar. MiddleModule, DDoS trafiği yasal trafikten çok daha güçlüyse, herhangi bir fark edilebilir performans kaybı olmaksızın çalışabilir. İyi tanımlanmış kalıplara veya rastgele kalıplara sahip saldırıları doğru bir şekilde tespit eder. Son olarak, MiddleModule sistemi hafif algılama algoritmaları kullanır; dolayısıyla MiddleModule, gözle görülür bir sistem performansı düşüşüne neden olmaz ve yüksek hızlar kullanılarak üretilse bile kötü amaçlı

etkinlikleri doğru bir şekilde algılayabilir. MiddleModule sisteminde tespit algoritma parametrelerinin ayarlanması gibi dikkat edilmesi gereken bazı noktalar bulunsa da MiddleModule sistemi tüm yönleriyle düşünüldüğünde diğer veri katmanı tabanlı savunma sistemlerinden üstündür.

MiddleModule sistemi, gelecek vaat eden bir kontrol uçağı temelli savunma sistemi ve bir hibrit sistem ile karşılaştırılmıştır. Kokila, Selvi ve Govindarajan (2014) saldırı tespiti amacıyla Destek Vektör Makinesi (Support Vector Machines - SVM) kullanılmasını önerir. Bu yöntemde, denetleyici üzerinde SVM tabanlı algılama algoritması çalışır. Ek olarak, bu çalışmada SVM diğer bazı sınıflandırıcılarla karşılaştırılmıştır. Rengaraju ve diğerleri (2017) OpenFlow anahtarlarının basit anormallik algılama algoritmaları çalıştırdığı hibrit bir sistem önerir. Anahtar herhangi bir anormal trafik algılarsa veya bilinmeyen kaynaklardan paket alırsa, bu paketleri denetleyiciye yönlendirir. Denetleyici, herhangi bir imza uyuşmazlığını algılamak için bu paketlere ID algoritmaları uygular.

Rengaraju, bir saldırgan (ICMP ve SYN flooding), bir yasal kullanıcı ve bir kurban sunucu içeren bir test ağı için tespit oranı değerleri sağlar. Kötü amaçlı trafik, ağa 10 kpps enjeksiyon hızında enjekte edilir. Öte yandan Kokila, SVM tabanlı merkezi savunma sistemini farklı ağ topolojileri ile yasal kullanıcılar, kötü niyetli kullanıcılar ve tek bir kurban sunucu ile test ediyor. Bu çalışmanın değerlendirilmesi sırasında tespit algoritması parametreleri optimum değerlerine getirildikten sonra savunma sistemi 120 saniye eğitilmiştir. Ardından savunma sisteminin isabet oranı %95,1 olarak ölçülüyor.

MiddleModule sistemi bu sistemlerle karşılaştırıldığında ise Çizelge 2.22'de görüldüğü gibi MiddleModule sistemi diğer çalışmalara göre üstün savunma gücü performansı sağlamaktadır. Rengaraju'nun tespit oranı %90'ın altındadır ve Kokila'nın tespit oranı %95,1'dir. MiddleModule sistemi ise %99'un üzerinde tespit oranı sağlamaktadır.

Çizelge 2.22. Kontrol uçağı temelli sistem ve hibrit sistem ile savunma gücü karşılaştırması

	Rengaraju	MiddleModule	Kokila	MiddleModule
Flood Atakları	%89,4	%99,3	%95,1	%99,6
	Algılama Hızı		Kesinlik	

Ayrıca, MiddleModule sisteminin ana avantajının, tek nokta arıza problemini önlemesi olduğu belirtilmelidir. Diğer sistemler paketleri kontrol katmanı seviyesinde analiz eder ve saldırı boyutu arttıkça denetleyicinin iş yükü de artar. Bir noktadan sonra, daha önce açıklandığı gibi denetleyici kaynakları tükenirdi. Bu kritik bir konu olmasına rağmen, bu merkezi yöntemler bu sorunla ilgili herhangi bir test sonucu sağlamaz, sadece savunma gücü performansını analiz ederler. Sonuç olarak, MiddleModule sistemi hem veri temelli yöntemlere hem de merkezi yöntemlere kıyasla her açıdan üstün sonuçlar sağlamaktadır.

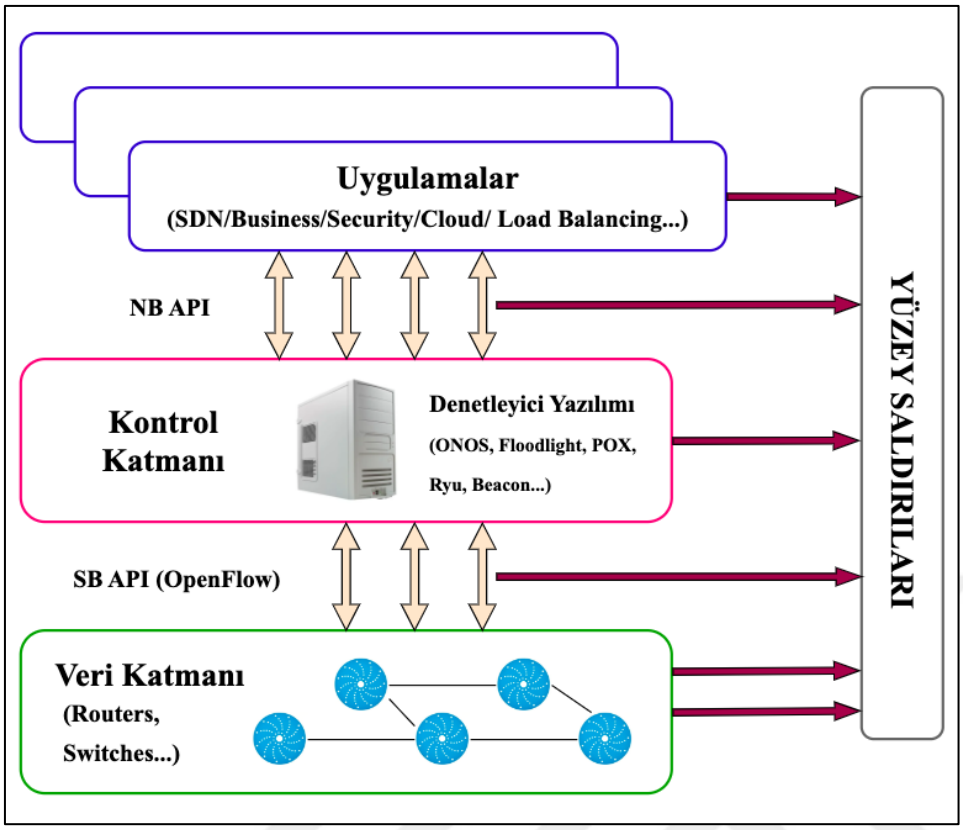
3. GERÇEK ZAMANLI SALDIRILARIN ÖNLENMESİ SİSTEMİ

3.1. Genel Bakış

3.1.1. 5G ağları kapsamında yazılım temelli ağlar

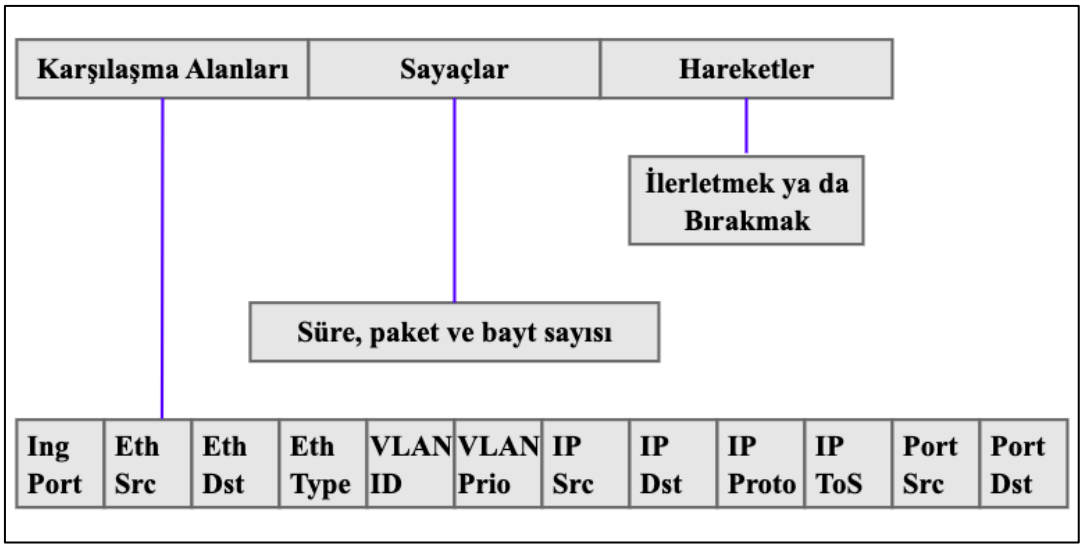
Yazılım temelli ağ oluşturma, son on yılda yeni bir ağ oluşturma paradigması olarak ortaya çıktı ve yaygın kullanım ve çeşitli IoT kullanım durumlarını barındırmak için ihtiyaç duyulan yeni teknolojilerle operasyonel maliyetleri keskin bir şekilde artan ağları programlı olarak yönetme ihtiyacına katkı sağladığı için yaygınlık kazanmıştır. SDN, veri ve kontrol katmanlarını birbirinden ayırdığından dolayı diğer ağ biçimlerinden farklı olarak görülmektedir; burada yönlendiriciler/anahtarlar, artık iletme işlevinden sorumludur ve yönlendirme kararları denetleyici (kontrol katmanı) tarafından alınmaktadır.

SDN mimarisi temel olarak üç katmandan oluşur. Bu katmanlar Şekil 3.1'de görüldüğü gibi uygulamalar, kontrol ve altyapı katmanı olarak sıralanmaktadır. Yük dengeleme ve izinsiz giriş tespit sistemleri gibi tüm uygulamalar, uygulama katmanında çalışır ve denetleyici ile iletişim, kuzeye bağlı Uygulama Programlama Arayüzü (Application Programming Interface - API) üzerinden gerçekleşir. Denetleyici ve anahtarlar arasındaki iletişim, çoğunlukla OpenFlow protokolü kullanılarak güneye bağlı API aracılığıyla gerçekleşir. Mantıksal olarak merkezileştirilmiş denetleyici, ağı yönetmekten sorumludur. Denetleyici, ağın genel bir görünümünü korur ve verdiği yönlendirme kararlarına dayalı olarak ilgili anahtarlara akış kuralları adı verilen yönlendirme kuralları yükler. Anahtarlar, akış kurallarını akış tablolarında depolar ve mevcut akış kurallarıyla eşleşmelere dayalı olarak ağ paketlerini iletir (McKeown vd., 2008:80).

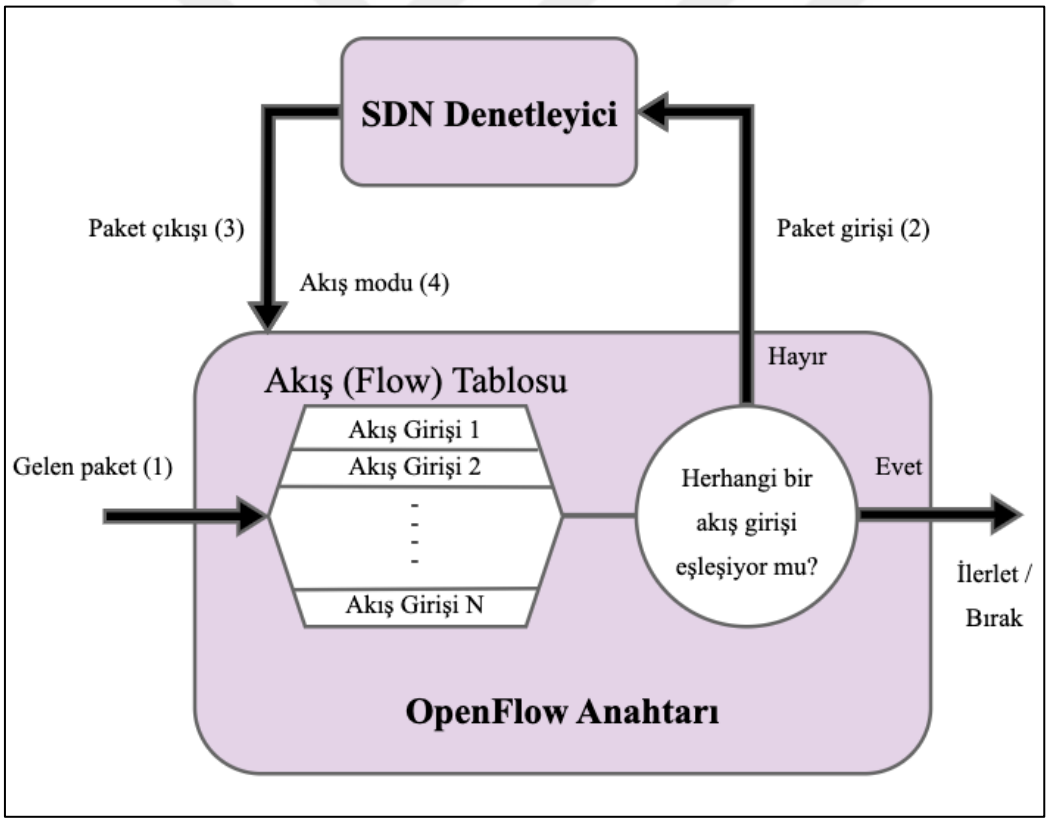


Şekil 3.1. SDN mimarisi ve saldırı yüzeyi

Şekil 3.2 bir akış kuralının yapısını göstermektedir. Bu yapı temel olarak üç bölümden oluşmaktadır. Yönlendirmeyi hedef adreslere göre gerçekleştiren geleneksel ağların aksine, karşılaşma (eşleşme) alanları yönlendirme uygulamasının yapılandırmasıyla belirlenir ve giriş bağlantı noktası, Sanal Yerel Ağ Alanları (Virtual Local Area Networks - VLAN) Kimliği, kaynak ve/veya hedef Ortam Erişim Kontrolü (Media Access Control - MAC) adresleri, IP adresleri ve/veya bağlantı noktası numaraları olabilir. Sayaçlar, akışın süresini ve akışla eşleşen bayt ve paket sayımlarını takip eder. Eylem, diğerlerinin yanı sıra, paketi belirtilen bağlantı noktasına iletmek veya bırakmak olabilir.



Şekil 3.2. Akış kuralı yapısı



Şekil 3.3. SDN'de paket akışı

Yukarıda belirtilen Şekil 3.3, SDN'de gelen paketler üzerinde gerçekleştirilen işlemleri göstermektedir. Gelen paketlerin başlık alanları, anahtardaki akış kurallarının eşleşme alanlarıyla karşılaştırılır. Paketin gerekli tüm başlık alanlarının, hemen iletilmesi için akış tablosundaki bir kuralın eşleşme alanlarıyla eşleşmesi gerekir. Aksi takdirde, anahtar paketi

arabelleğe alır ve paketin başlık alanlarını içeren denetleyiciye ‘package_in’ mesajı olarak adlandırılan mesajı gönderir. Denetleyici daha sonra ‘package_in’ mesajını inceler ve paket için bir ‘route_out’ mesajında anahtara geri gönderilen bir yönlendirme kararı üretir. Paket için gerekli işlem yapılır ve ‘flow_mod’ mesajları gönderilerek ilgili akış kuralları anahtarlara kurulur. Denetleyici ilgili anahtarlara akış kuralları yüklemeye karar verse bile, her zaman ‘flow_mod’ mesajından önce bir ‘package_out’ mesajı gönderilir. Bu nedenle, geleneksel ağlardan farklı olarak, akış kuralı kurulumunu tetikleyen ilk paketin istatistikleri kurulu akış kuralında görülemez. TCP bağlantıları için, sırasıyla kaynaktan hedefe ve hedeften kaynağa gönderilen ilk paketler oldukları için SYN ve SYN ACK paketlerinin istatistikleri kaybolur.

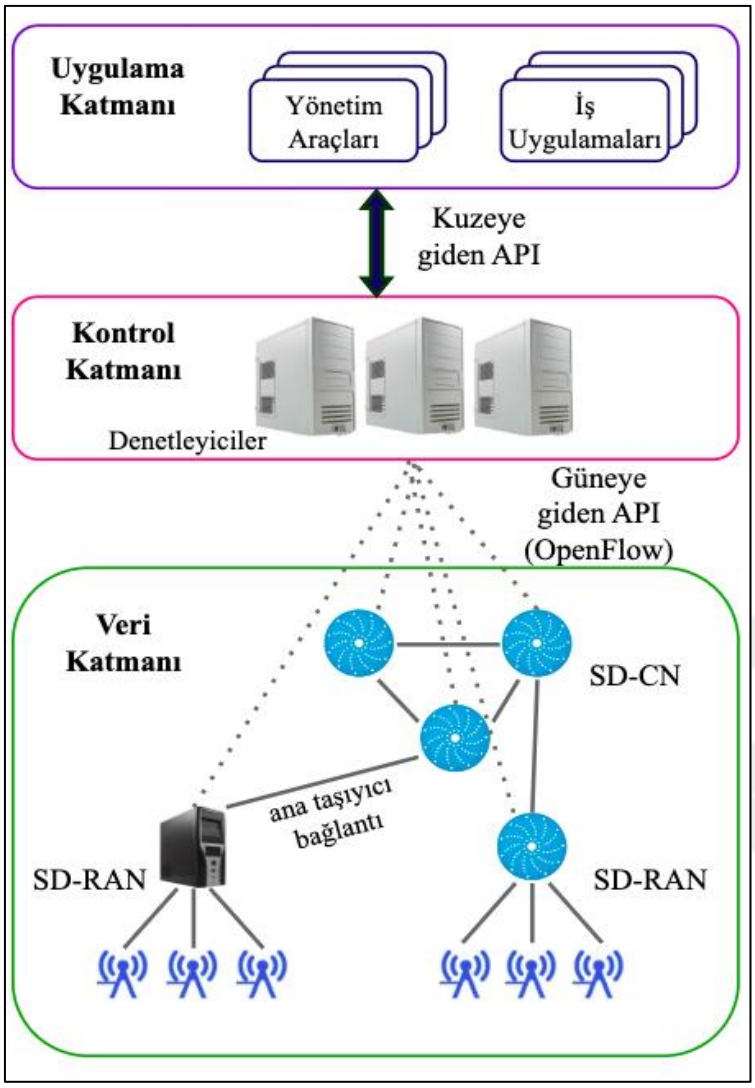
SDN'nin paket eşleştirme ve akış kuralı yükleme davranışları nedeniyle, DoS ve DDoS saldırıları SDN denetleyicisini de etkiler. Sahte adreslerle yapılan DoS ve DDoS saldırılarında, gelen paketlerin tamamı farklı kaynak adreslere sahip olabilir. Bu nedenle, saldırgandan gelen tüm paketler yeni bir akış kuralı yüklemesini tetikleyebilir. DoS ve DDoS saldırılarının neden olduğu ‘Packet_in’ istekleri, denetleyici ile anahtar arasındaki kullanılabilir bant genişliğini azaltır. Bu ‘package_in’ istekleri ayrıca SDN denetleyicisinin kaynaklarını, yani CPU, belleği tüketmektedir.

SDN'nin erken benimsenmesi çoğunlukla kablolu kurumsal ağlarda yer alırken, esnekliği, programlanabilirliği, hızı ve maliyet avantajları, son zamanlarda onu kablosuz sensör ağları ve yeni nesil kablosuz ağ altyapıları dahil olmak üzere diğer ağlar için umut verici bir araç haline getirmiştir. SDN, ağları uzaktan ve dinamik olarak yönetmek için gereken ağ sanallaştırma yeteneklerini sağlayarak 5G ve ötesi ağların en büyük etkinleştiricilerinden biri olacaktır. SDN uygulamalarıyla elde edilecek hızlı yük devretme ve otonom yönetim yetenekleri, 5G ağlarının yüksek bant genişliği ve düşük gecikme gereksinimlerini sağlayarak, çeşitli IoT kullanım durumlarını desteklemelerini sağlayacaktır (Mostafaei ve Menth, 2018:42-56).

SDN’ler, NFV ile birlikte, özellikle 5G çekirdek ve radyo erişim ağlarında ağ dilimlemenin temelini oluşturacak ve bu, operatörlerin gerekli hizmet kalitesini sağlamak için altyapılarını verimli bir şekilde kullanmalarına önemli bir olanak sağlayacak ve müşterilerine güvenlik garantisi sunmak için gerekli işlevleri yerine getirmektedir (Barakabitze, Ahmad, Mijumbi ve Hines, 2020).

5G ağları için bir dizi SDN tabanlı mimari önerilmektedir (Long, Chen, Zhang ve Lei, 2019). İlk öneriler arasında, Akyıldız ve diğerleri tarafından yapılan SoftAir önerisi bulunmaktadır (Akyıldız, Wang ve Lin, 2015:10). Bu öneride veri katmanı, yazılım temelli çekirdek ağ (SD-CN) ve yazılım temelli radyo erişim ağından (SD-RAN) oluşan programlanabilir bir ağ iletme altyapısıdır. SD-RAN, geleneksel makro hücrelere ek olarak küçük hücreler (mikro hücreler, femto hücreler ve piko hücreler) dahil olmak üzere yazılım tanımlı baz istasyonları içerirken, aşağıda yer alan Şekil 3.4'de görüldüğü gibi SD-CN, 5G çekirdek ağını oluşturan yazılım temelli anahtarlar içerir. Kullanıcı ekipmanı ve diğer cihazlar, ana taşıyıcı bağlantıları aracılığıyla yazılım temelli çekirdek ağa bağlanan yazılım temelli baz istasyonlarına veya kablosuz erişim noktalarına bağlanır. SoftAir'de önerildiği gibi, SD-CN ve SD-RAN'ın her ikisi de güneye giden API olarak OpenFlow'u kullanabilir; bu, ana ağdaki en uygun yollardan baz istasyonlarından gelen trafiği yönlendiren denetleyiciyle tekdüze bir arabirim sağlar. Bu mimari, kablolu SDN'den SDN tabanlı 5G ağlarına kadar ağ kontrolü açısından aynı ilkelerin çoğunun uygulanmasını sağlamaktadır.

SDN'nin tanıtılmasının en büyük vaatlerinden ve nedenlerinden biri, küresel görünürlük yoluyla ağda gelişmiş güvenliğin sağlanması ve akış kurallarının hızlı otomatik yeniden yapılandırılmasıdır. Bu, ağdaki kötü amaçlı trafiğin gerçek zamanlı olarak algılanmasını ve azaltılmasını sağlayacaktır. Şekil 3.4'de görüldüğü gibi, bir SDN'ye çeşitli yüzeylerde saldırılabilir (saldırı yüzeyi, SDN'de saldırıya uğrayabilecek cihazlardan, uygulamalardan veya ara yüzlerden işaret eden kırmızı oklarla gösterilmiştir). Bu saldırılar sadece veri katmanı cihazlarını değil, denetleyici ve uygulamaları da hedefleyerek ağ işleyişinde aksamalara neden olabilir.



Şekil 3.4. Yazılım tanımlı 5G ağ mimarisi

3.1.2. Random Forest modeli

Random Forest (RF), her bir karar ağacının bağımsız ve aynı şekilde dağıtılmış bir rasgele vektör kullanılarak yapılandırılacağı şekilde, orman adı verilen bir karar ağaçları topluluğu oluşturan bir makine öğrenme modeli olarak öne çıkmaktadır (Breiman, 2001:27).

Belirli bir veri örneğini sınıflandırmak için rastgele bir orman, çoğunluk kararını seçmek için ormandaki tüm ağaçların çıktılarını kullanır. Birden çok ağacın çıktılarının kullanılması, sınıflandırıcıyı, çoğu durumda fazla uydurma probleminden mustarip olan karar ağaçlarından daha sağlam hale getirir.

Yüksek düzeyde, RF algoritması aşağıdaki gibi çalışır:

- Bir C sınıfı kümesinden $\{c_i$, i = 1, ..., n$\}$ sınıf etiketli n veri örneğinden oluşan tam eğitim kümesi δ, önyükleme örneklemesi kullanılarak K rasgele alt kümeye bölünür:

$$\delta = \delta_1, \delta_2, \dots, \delta_K \tag{3.1}$$

- Rastgele bir öznitelik vektörü θ_k oluşturulur ve her δ_k'den bir karar ağacı oluşturmak için kullanılır. Tüm $\{\theta_k$, k=1, 2, 3, ..., $\theta_k\}$ bağımsızdır ve aynı şekilde dağılmıştır.

- Tüm gruplar (δ_k, θ_k), R ortamını oluşturmak için genişletilmektedir.
- Bir test veri örneğinin x sınıflandırması şu şekilde hesaplanır:

$$H(\mathrm{x}) = max_{c_j} \sum_{i=1}^{k} (I(h_i(x) = C_j)) \tag{3.2}$$

- Burada I, gösterge işlevidir ve h_i(x) tarafından yapılan sınıflandırmanın sonucudur.

$$r(S_i, \theta_i). \tag{3.3}$$

Şekil 3.5 RF'ye göre sınıflandırmanın basitleştirilmiş bir görünümünü göstermektedir. Karar ağaçlarındaki çeşitli düğümler için bölme kriterlerine karar vermek için yaygın olarak kullanılan bir ölçü bilgi kazancıdır. Rastgele bir değişkene (a) bağlı olarak bir S düğümünün bölünmesinden elde edilen bilgi şu şekilde hesaplanır:

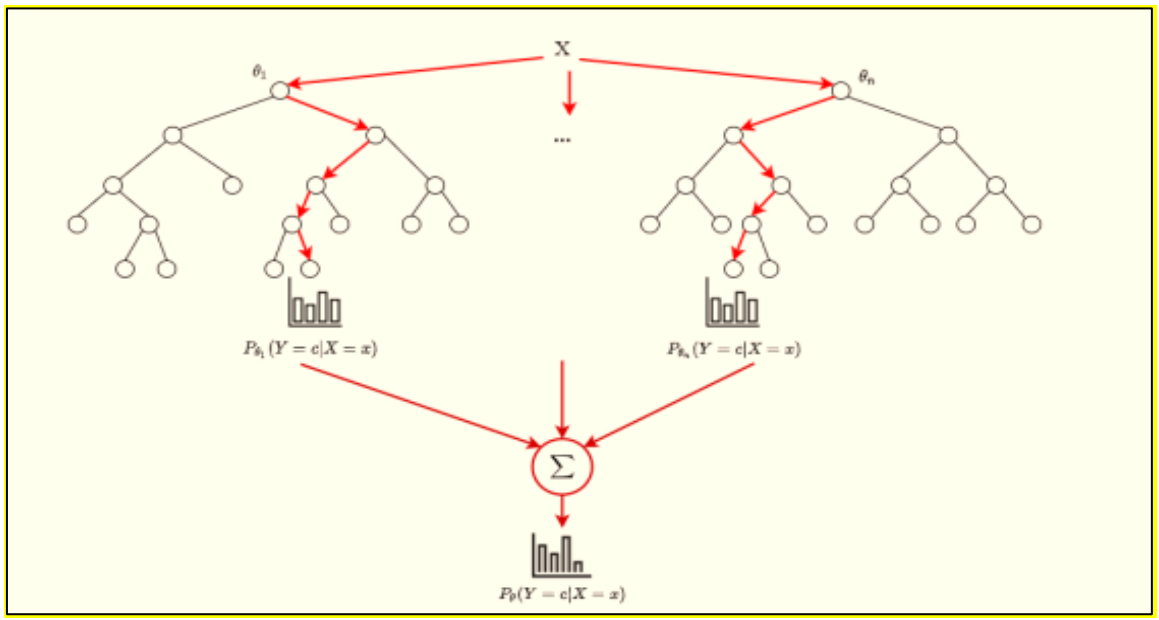
$$IG(S, a) = E(S) - E(\mathrm{S|a}) \tag{3.4}$$

Burada $E(S)$, ana düğümün bölünmeden önceki entropisini ve E(S|a), bölünmeden sonraki alt düğümlerin entropilerinin ağırlıklı ortalamasıdır. $E(S)$ şu şekilde hesaplanır:

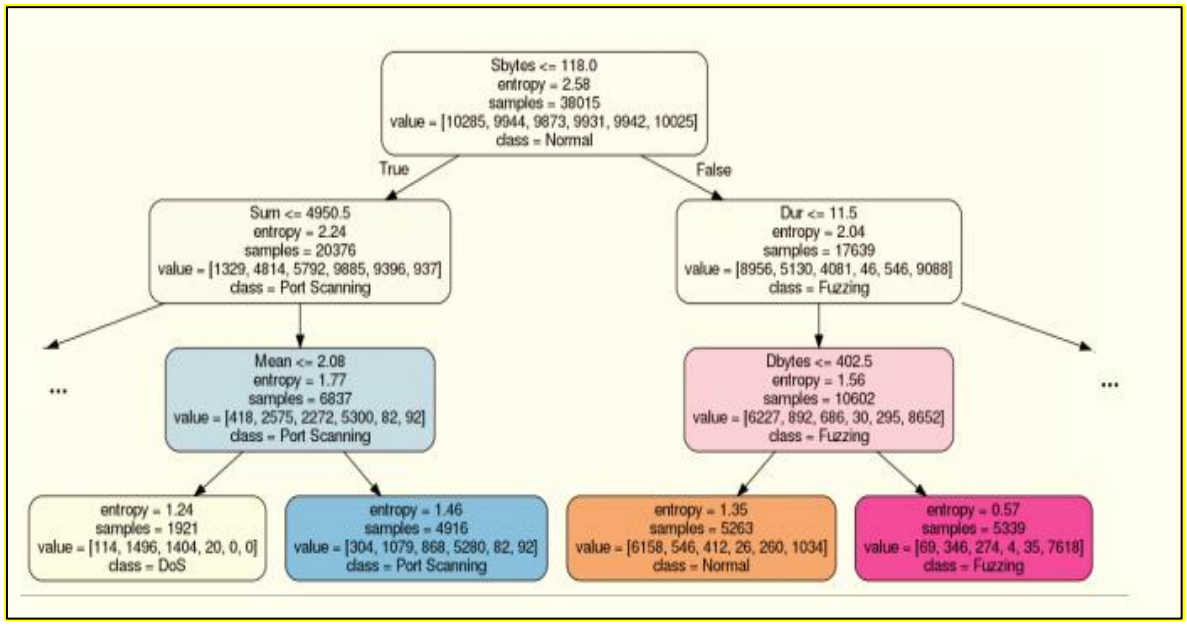
$$E(S) = -\sum_{i=1}^{C} p(c_i) \log p(c_i) \tag{3.5}$$

Burada $p(c_i), c_i$ sınıf etiketine sahip S düğümündeki veri örneğinin olasılığıdır.

Şekil 3.6, oluşturduğumuz IoT ağ veri kümesinde örnek bir ağ saldırı tespit görevi için oluşturulan RF2den bir karar ağacının kısmi bir görünümünü göstermektedir. Şekilde görüldüğü gibi, ağaçta daha yüksekteki düğümler, algoritma tarafından keşfedilen özellik değerlerinin belirli bir eşiğine göre bölündüğünden, yapraklara yaklaştıkça düğümlerin entropisi azalır. RF, her biri tüm eğitim verilerinin farklı, rastgele örneklenmiş bir alt kümesinden oluşturulmuş çok sayıda bu tür karar ağacı içerir.



Şekil 3.5. Rastgele ormana göre sınıflandırma



Şekil 3.6. Bir ağ saldırı tespit görevi için rastgele ormandan örnek karar ağacı

RF, insan etki alanı uzmanları tarafından kolayca okunabilen, özellik değerleri boyunca eğitim verilerinin bölünmesine dayalı modeller oluşturan karar ağaçlarına bağlı olması nedeniyle en yüksek açıklanabilirlik / yorumlanabilirlik derecesine sahip makine öğrenimi algoritmaları arasında yer almaktadır. Çeşitli sınıflandırma görevleri için RF'nin etkinliği birçok çalışmada yer almaktadır (Delgado ve Cernadas ve Barro ve Amorim, 2014:3138).

3.2. İlgili Çalışmalar

Ağlarda saldırı tespiti ve hafifletilmesi, IoT tarafından oluşturulan büyük saldırı yüzeylerinin neden olduğu artan siber güvenlik olayları ile her zamankinden daha önemli bir araştırma konusu haline gelmiştir. Son on yılda SDN temelli ağ mimarilerinin giderek daha fazla benimsenmesiyle birlikte SDN güvenliği, siber güvenlik araştırma topluluğu için ilgi merkezlerinden biri haline gelmiştir.

Rathore ve Park çalışmalarında (2018:79), IoT ağlarında dağıtılmış saldırı tespiti için sis tabanlı yarı denetimli bir öğrenme yaklaşımı önermektedir. Araştırmacılar, NSL-KDD veri setini kullanmışlardır ve dağıtık yaklaşımlarının algılama süresi ve doğruluğu açısından merkezi çözümden daha iyi performans gösterdiğini saptamışlardır.

Evmorfos, Vlachodimitropoulos, Bakalos ve Gelenbe (2020), yürüttükleri bir araştırmada IoT ağlarında SYN taşma saldırılarını algılamak için Rastgele Sinir Ağı ve LSTM kullanan bir yapı önermesinde bulunmuşlardır. Araştırmacılar sanal bir ağ oluşturarak veri setlerini meydana getirmekte ve trafiği ‘pcap’ dosyalarına kaydetmişlerdir.

Soe, Feng, Santosa, Hartanto ve Sakurai (2020:4372) yaptıkları bir çalışmada IoT ağları için üç makine öğrenimi modeli kullanan sıralı bir saldırı tespit yapısı önermektedirler. Araştırmacılar, N-BaIoT veri setini kullanmakta ve %99 doğruluk elde etmektedirler.

Alqahtani, Mathkour ve İsmail (2020:6336) nesnelerin interneti ağlarındaki özellikleri seçmek için Fisher skorunu kullanan, genetik tabanlı bir aşırı gradyan artırma (GXGBoost) modeli önermişlerdir. Araştırmacılar ayrıca N-BaIoT veri setini kullanmışlar ve %99,96 doğruluk elde etmişlerdir. Bu yaklaşımlar IoT'ye yönelik saldırıları başarılı bir şekilde tespit etse de geleneksel ağlarda çalışan uygulamalar olarak görülmektedirler.

Şimdiye kadar SDN tabanlı ağlar için önerilen çözümlerin çoğu DoS ve DDoS saldırılarının tespit edilmesi ve hafifletilmesine yönelik tekniklere odaklanmaktadırlar. Ravi ve Shalinie (2020:3562) tarafından yapılan bir çalışmada SDN tabanlı IoT ağlarındaki DDoS saldırılarını tespit etmek için UNB-ISCX veri kümesi üzerinde yarı denetimli bir model kullanılmıştır. Model, UNB-ISCX veri kümesinde ve yalnızca UDP sel saldırılarını içeren araştırmacıların kendi veri kümesinde %96'nın üzerinde doğruluk elde etmişlerdir.

Galeano-Brajones, Carmona-Murillo, Valenzuela-Valdés ve LunaValero (2020:816) tarafından yapılan bir çalışmada yazılım temelli IoT ağlarında DoS ve DDoS saldırılarının tespiti ve hafifletilmesi için entropi tabanlı bir çözüm önerilmektedir. Yaklaşım, Bot-IoT veri kümesinde ve araştırmacıların TCP SYN sel saldırılarını içeren veri kümesinde yüksek doğruluk elde etmişlerdir.

Ahmed ve Kim (2017:271) tarafından yürütülen bir çalışmada DDoS saldırılarını azaltmak için farklı alanlardaki anahtarlardan toplanan istatistikleri kullanan bir alanlar arası bilgi alışverişi yaklaşımı önerilmektedir.

Bhunia ve Gurusamy (2017:5) tarafından yapılan bir araştırmada SDN tabanlı IoT ağlarında DoS saldırılarını tespit etmek ve azaltmak için SVM kullanılmıştır.

Li ve diğerleri (2019:2095) tarafından yapılan bir çalışmada özellik seçimi için BAT algoritmasını kullanılması önerilmiş ve ardından KDD CUP’99 veri setinde RF algoritmasını kullanarak %96 doğruluk elde edilmiştir.

Amangele, Reed, Al-Naday, Thomos ve Nowak (2019:3) tarafından yapılan bir çalışmada SDN kullanan IoT ağlarındaki anormallikleri tespit etmek için CART karar ağacı algoritması önerilmektedir. Araştırmacılar CICIDS'2017 veri setini kullanmış ve %99 tespit oranı elde etmişlerdir.

Derhab, Guerroumi, Gumaei, Maglaras, Ferrag, Mukherjee ve Khan (2019:3119) tarafından yapılmış bir çalışmada endüstriyel kontrol süreçlerini hedef alan saldırılara karşı Rastgele Altuzay Öğrenme, K-En Yakın Komşu ve blockchain kullanan bir izinsiz giriş tespit sistemi önerilmektedir. Araştırmacılar, Endüstriyel Kontrol Sistemi Siber saldırı veri setini

kullanmakta ve çözümlerinin yüksek doğruluk sağladığını ortaya koymaktadırlar. Açıklanabilir saldırı tespit sistemlerindeki çalışmalar şu ana kadar oldukça sınırlı kalmıştır.

Wang, Zheng, Yang ve Wang (2020) tarafından yapılmış bir araştırmada Shapley Ek Açıklamalarına dayalı saldırı tespit sistemleri için açıklanabilir bir makine öğrenimi çerçevesi önerilmiştir. Çerçeve, NSL-KDD veri kümesi üzerinde değerlendirilmektedir.

Bazı saldırı türleri, SDN denetleyicisini ve anahtarlarını da etkiler. Alharbi, Layeghy ve Portmann (2017:5) tarafından yürütülen bir araştırmada SDN'de DoS saldırılarının etkilerini gösteren bir çalışma yürütmüşlerdir. Araştırmacılar, DoS saldırısı altında paket dağıtım oranını, denetleyici işlemci kullanımını ve anahtar işlemci kullanımını saptamışlardır.

Şimşek, Bostan, Sarıca, Sarıkaya, Keleş, Angin, Alemdar ve Onur (2020) tarafından yapılmış bir çalışmada farklı tipteki DoS saldırılarının SDN üzerindeki etkilerini göstermek maksadıyla bir araştırma yapmışlar ve P4 anahtarları kullanarak veri katmanında DoS saldırılarını azaltmak için bir yaklaşım ortaya koymuşlardır.

Makine öğrenimi tekniklerini kullanan ağ saldırı tespiti, günümüzde araştırmacıların bir dizi kapsamlı ağ izleme veri seti oluşturduğu popüler bir ağ güvenliği yaklaşımı alanı olarak öne çıkmaktadır. Bu veri kümeleri, eski olsa bile, bugün hala güvenlik araştırmacıları tarafından kıyaslama olarak kullanılmaktadır. Mevcut halka açık ağ saldırı tespit veri kümeleri şu şekilde sıralanmaktadır;

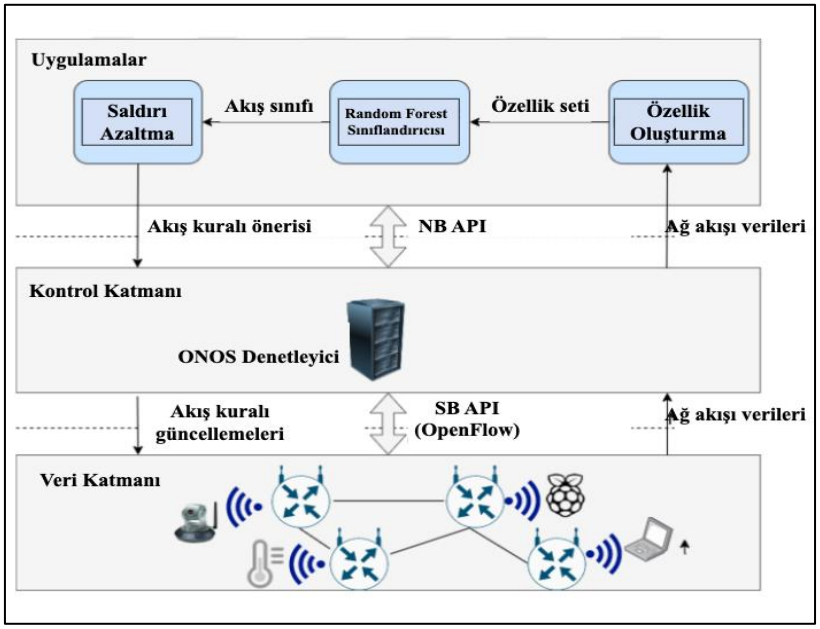
- KDD CUP’99 Veri Seti: 1999 yılında bir ABD Hava Kuvvetleri LAN'ını simüle eden DARPA98 veri setinden belirli özelliklerin çıkartılması sonrasında oluşturulmuş bir veri setidir. KDD CUP’99, 41 özelliğe ve 4 saldırı kategorisine sahiptir. Eski bir veri seti olmasına rağmen birçok araştırmacı halen bu veri setini kullanmaktadır ancak bu veri setinin belirli avantajsız durumları bulunmaktadır. İlk olarak, eğitim ve test setlerindeki kayıtların dağılımı oldukça farklıdır, çünkü test seti eğitim setinde olmayan bazı saldırı türlerini içermektedir. İkinci olarak, eğitim ve test setlerindeki verilerin yaklaşık %75'i yinelenen verilerdir ve bu da taraflı sınıflandırma modellerine yol açabilmektedir. En önemlisi, veri seti bir IoT ortamında üretilmemiştir ve SDN'ye özgü özellikler içermemektedir.

- NSL-KDD Veri Seti: KDD CUP’99 veri setine yeni özellikler eklenerek ve iyileştirilerek oluşturulmuş bir veri setidir. Bu veri setinin amacı KDD CUP’99 veri setindeki problemleri çözmektir. Sistem tekrar eden kayıtları ortadan kaldırmış ve kayıt sayısında azalmaya gitmiştir. Yine de bu veri kümesi mevcut ağların davranışını temsil etmemektedir (Tavallaee, Bagheri, Lu ve Ghorbani, 2009).
- UNB-ISCX Veri Seti: Kanada Siber Güvenlik Enstitüsü tarafından 2012 yılında oluşturulmuştur. Gerçekçi profiller oluşturmak için gerçek ağ izleri analiz edilmiştir. Veri seti, DDoS, Kaba Kuvvet SSH ve Ağa İçeriden Sızma Saldırıları olarak adlandırılan üç saldırı türünü barındırmakta ve 7 günlük ağ trafiğinden oluşmaktadır (Shivari, Shivari, Tavallaee ve Ghorbani, 2012).
- CAIDA Veri Seti: Anonimleştirilmiş ağ izlerini içermektedir. Paketlerin yükleri kaldırılarak ve başlık anonimleştirilerek kayıtlar oluşturulmaktadır. Bu veri kümesi yalnızca DoS saldırılarını içerir ve özellikler başlık alanlarıdır. Başlık alanını kullanan ek özellikler oluşturulmamaktadır.
- UNSW-NB15 Veri Seti: 2015 yılında oluşturulan bu veri seti ağ trafiğini oluşturmak için IXIA aracı kullanmaktadır. UNSW-NB15 veri seti 49 özelliğe sahiptir ve bunlardan ikisi ikili ve çok sınıflı sınıflandırma için etiketlerdir. Veri seti, normal trafik ve DoS, DDoS, fuzzing, arka kapı, analiz, yararlanma, jenerik, solucan ve kabuk kodu olmak üzere 9 tür saldırı trafiğinden oluşmaktadır. Veri setinin temel sorunu, bazı saldırı türleri için yeterli sayıda örneğin olmamasıdır (Moustafa ve Slay, 2015).
- CICIDS2017 Veri Seti: Kanada Siber Güvenlik Enstitüsü tarafından oluşturulan başka bir veri setidir. B-Profile sistemleri kullanılarak gerçekçi zararsız trafik oluşturulmuştur. Veri seti normal trafiği ve DoS, botnet, port tarama, kaba kuvvet, sızma ve web saldırısı olmak üzere 6 tür saldırı trafiği içermektedir (Sharafaldi, Lashkari ve Ghorbani, 2018).
- BOT-IOT Veri Seti: 2018 yılında kullanıma sunulan bu veri setinin en önemli kısmı, mevcut saldırı tespit veri setlerinin çoğundan farklı olarak IoT trafiğini içermesidir. Veri seti 46 özelliğe sahiptir ve bunlardan ikisi ikili ve çok sınıflı sınıflandırma için etiketlerdir. Veri seti normal trafik ve DoS, DDoS, hizmet tarama, işletim sistemi parmak izi, veri hırsızlığı ve keylogging olmak üzere 6 farklı saldırı türünden oluşmaktadır. Veri setinin temel sorunu, bazı saldırı türleri için yeterli sayıda örneğin olmamasıdır. Normal trafik için kayıt sayısı da düşüktür (Koroniotis, Moustafa, Sitnikova ve Turnbull, 2018).

IoT ve SDN ortamları için izinsiz giriş tespit sistemlerine ilişkin mevcut çalışmaların çoğu, yukarıda belirtilen veri kümelerini kullanmaktadır, ancak bu veri kümelerinin hiçbiri bir SDN ortamında oluşturulmamıştır. Bu sebeple SDN ağ davranışlarını yansıtmazlar. Ayrıca bu veri kümeleri BoT-IoT veri kümesi dışında IoT trafiği içermemektedir. Mevcut veri kümelerinin çoğu, *pcap* dosyalarının farklı araçlarla kaydedilmesi ve işlenmesiyle oluşturulmuştur. Bu nedenle, bir SDN denetleyicisi tüm özellikleri gerçek zamanlı olarak ulaşmayabilmektedir. Bildiğimiz kadarıyla, IoT trafiğini içeren, halka açık başka bir SDN veri kümesi bulunmamaktadır.

3.3. Önerilen Güvenlik Yaklaşımları

Genel işleyişi Şekil 3.7'de yer alan önerilen izinsiz girişi tespit etme ve azaltma yaklaşımı, ağ akışlarının otomatikleştirilmiş, akıllı analizi ve ardından izinsiz giriş tespit bileşeninin kararına uygun olarak alınan hafifletme eylemleriyle SDN tabanlı ağlarda güvenlik sağlamaktadır. Uçtan uca saldırı tespiti ve azaltma işlemi, uygulama katmanında bir SDN uygulamasında çalışan üç ana modüle, yani Özellik Oluşturma, RF sınıflandırıcı ve Saldırı Azaltmaya dayanmaktadır.



Şekil 3.7. SDN tabanlı güvenlik çözümü mimarisi

Özellik Oluşturma Modülü, anahtarlardan düzenli aralıklarla ağ akışlarını toplar ve her akış için RF sınıflandırıcının gerektirdiği özellikleri hesaplar. RF sınıflandırıcı, önceden oluşturulmuş izinsiz giriş algılama modelini akış örneğine uygular ve sonucu Saldırı Azaltma Modülüne iletir. Saldırı Azaltma Modülü daha sonra sınıflandırma sonucuna göre yapılacak eylemi belirler ve gerekirse saldırıyı azaltmak için ilgili anahtarlara akış kuralları yükler. Çizelge 3.1'deki Algoritma, önerilen güvenlik çözümünün uçtan uca çalışmasını özetlemek maksadı ile verilmiştir.

Denetleyici, düzenli aralıklarla Özellik Oluşturma Modülü tarafından alınan anahtarlardan ağ akış girişlerini düzenli olarak toplar. Alındıktan sonra, her akış için özellikler oluşturulur. Her akış için ortak özellikler, anahtardaki her akış girişi üzerinden döngü halinde oluşturulur. Örneğin; ortalama akış süresi ve işlemdeki toplam paket sayısı, ilk geçiş akışı girişleriyle oluşturulur. Ardından, akışa özgü özellikler, örneğin akış süresi ve kaynaktan hedefe paket sayısı, tüm akış girişlerinin üzerinden geçilerek alınır. Akış girişleri üzerinde döngü yaparken, bir akış için oluşturulan özellik vektörü, diğer akış girişleri için özellik oluşturmayı bitirmeyi beklemeden hemen RF sınıflandırıcıya gönderilir. Özellik Oluşturma Modülü, kaynak IP ve MAC adresleri gibi akış eşleştirme alanlarını ve paketin geldiği anahtarın fiziksel bağlantı noktasını da alır. Bu özellikler Saldırı Azaltma Modülü tarafından kullanılır.

RF sınıflandırıcı, Özellik Oluşturma Modülünden öznitelik vektörlerini birer birer alır ve bunları önceden kullanarak sınıflandırır. Sınıflandırmanın sonucu herhangi bir saldırı türüyse, algılanan saldırı türü ve akışın kaynak tanımlayıcıları, yani paketin geldiği kaynak IP, kaynak MAC adresi ve fiziksel anahtar bağlantı noktası Saldırı Azaltma Modülüne gönderilir. Kullanılan makine öğrenimi modeli, mevcut saldırı türleri için yeni eğitim verileri dahil edilerek veya keşfedildikçe yeni saldırı türleri eklenerek dinamik olarak güncellenmelidir. Oluşturulan RF modeli, normal trafiğe ek olarak çeşitli saldırı türlerinden oluşan eğitim verileri kullanılarak oluşturulan çok sınıflı sınıflandırma modelidir. İlki, gerçekleştirilecek eylem/yüklenecek belirli akış kuralı açısından daha bilinçli karar verme yeteneği sağladığından, ikili sınıflandırma/anormallik tespiti yerine çok sınıflı saldırı sınıflandırmasının kullanılması gerekmektedir.

Çizelge 3.1. Uçtan uca işlem adımları

1	L_S← Bağlı olan SDN anahtarları alınır.	
2	M←RF sınıflandırıcı modeli yüklenir.	
3	L_B← Kara Liste	
4	Doğru ise,	
5		S ve L_S için
6		L_E←S için akış girişleri çekilir.
7		L_E=Özellik_Oluşturma
8	Son.	
9	Bir süre beklenir.	
10	Son.	
11	Özellik Oluşturma İşlevi:	
12		F_1← L_E ile ortak özellikleri belirlenir.
13		E ve L_E için
14		F_2← E için akışa özgü özellikler hesaplanır.
15		F←(F_1. F_2)
16		Saldırı Tespiti (F)
17	Son.	
18		Saldırı Tespiti İşlevi (Özellik Vektörü F):
19		C← F ve M sınıflandırmak için kullanılır.
20		Eğer C saldırı ise,
21		I ← Kaynak tanımlayıcılarını F den almaktadır.
22		Azaltma (C. I)
23	Son.	
24	Azaltma İşlevi (C $saldırı$ $tipi$, I $tanımlayıcı$) :	
25		Eğer L_B içerisinde I bulunmuyorsa,

Çizelge 3.1. (devam) Uçtan uca işlem adımları

26		$E \leftarrow I'nın$ engellenmesi ya da yeniden yönlendirilmesi için akış girişi oluşturulmaktadır.
27		$E, S'ye$ takılır.
28		$I,\ L_B$ içerisine eklenir.
29	Son.	

Daha önce tartışıldığı gibi, RF sınıflandırıcısı, sonuçları kolayca yorumlanamayan kara kutu makine öğrenimi modellerinin aksine, insan uzmanlar tarafından oldukça açıklanabilir sonuçlar oluşturur. Örneğin, belirli bir akış bir DoS saldırısı olarak sınıflandırıldığında, RF ortamındaki DoS olarak oy kullanan ağaçları ve hangi özellik değerlerinin bu kararı vermelerine neden olduğunu izlemek mümkündür. Bu, bir insan ağı uzmanına modelin kalitesini değerlendirme, önerilerde bulunma, modeli güncelleme veya gerekirse ek önlemler alma yeteneği sağlamaktadır.

Saldırı Azaltma Modülü, saldırı tespit edildiğinde RF sınıflandırıcı tarafından bilgilendirilir. Bu modül, saldırı türüne bağlı olarak bir akış kuralı güncellemesi oluşturur. Oluşturulan akış kuralı güncellemesi, akış girişlerini ilgili anahtarlara yükleyen denetleyiciye gönderilir. Kurulan akış girişleri, anahtardaki normal akış girişlerinden daha yüksek önceliğe sahiptir. Paket engelleme ve yeniden yönlendirme, kaynak MAC adresine, kaynak IP'ye veya fiziksel anahtar bağlantı noktasına dayalı olabilmektedir.

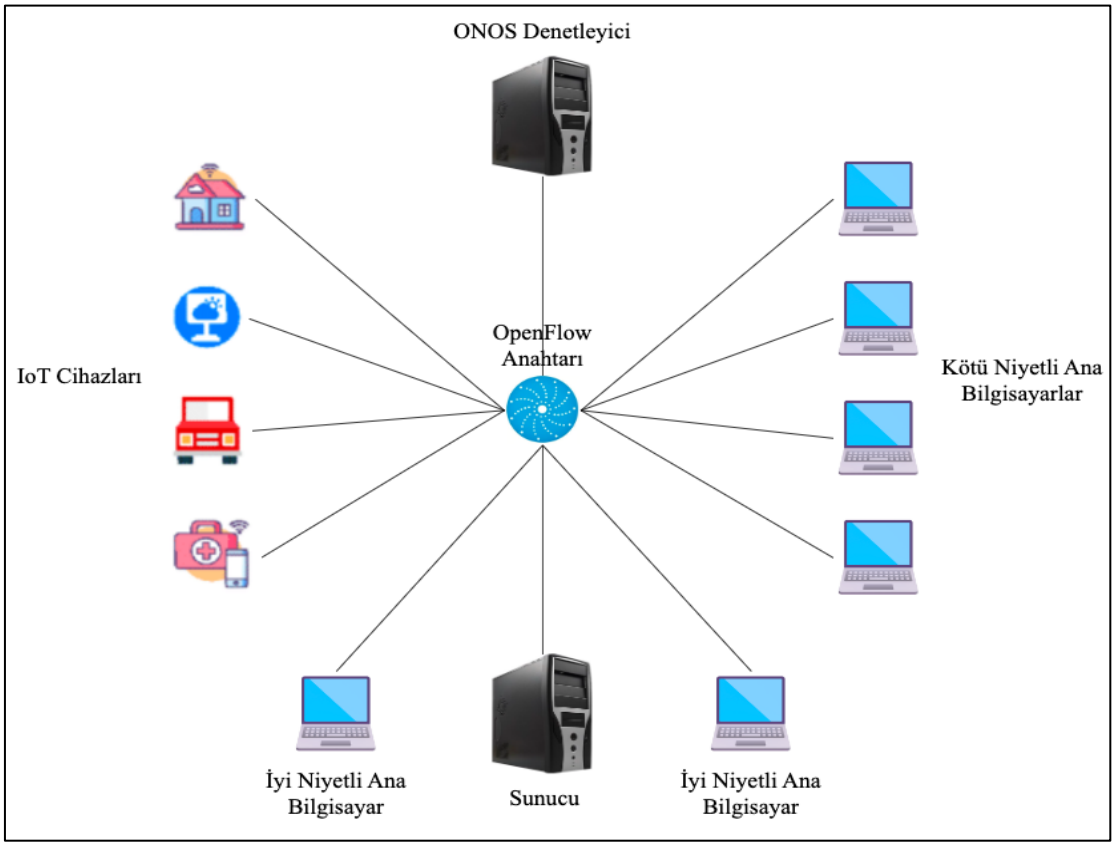
3.4. Yazılım Temelli Ağlar Veri Kümeleri

Başlık içerisinde ve alt başlıklar dahilinde SDN temelli bir ağ simülasyonunda oluşturulan veri kümelerimizin ayrıntıları açıklanmıştır. Normal trafiğin paket gönderme hızları ve paket boyutları, gerçek bir test yatağında oluşturulan bir veri kümesinden alınmıştır. Veri kümelerimiz normal trafik ve DoS, DDoS, port tarama, işletim sistemi parmak izi ve fuzzing olmak üzere 5 farklı saldırı türü içermektedir. Normal trafik hem IoT hem de normal cihaz trafiğini içermektedir. Tüm özelliklerimiz SDN'ye özeldir ve bir SDN uygulaması kullanılarak gerçek zamanlı olarak alınmıştır. RF sınıflandırıcının doğruluğu, oluşturduğumuz halka açık iki SDN veri seti ile değerlendirilmiş ve son teknoloji makine

öğrenimi algoritmalarının doğruluklarıyla karşılaştırılmıştır. Özellik seçimi, SDN tabanlı ağlardaki saldırıları tespit etmek için önemli özellikleri belirlemek için kullanılmıştır. Modelin performansı ağ değişiklikleri için değerlendirilmiştir.

3.4.1. Test ortamına genel bakış

Bot-IoT veri setine benzer bir ağ topolojisi kullanılmıştır ve Şekil 3.8'de ilk veri setimizin test yatağı kurulumu görülmektedir. Ağımızı sanallaştırmak için Mininet (Lantz, Heller ve McKeown, 2010) kullanılmış ve ağımızı bir ONOS denetleyicisi yönetmiştir (ONOS, 2023). Denetleyiciyi ve simüle edilmiş cihazları bağlamak için bir Open vSwitch (Open vSwitch, 2023) kullanılmıştır. Veri üretimi sırasında, sunucu sistemdeki diğer iki ana bilgisayarla sürekli olarak büyük miktarda veri alışverişi yapmıştır. İlk veri setinde, ağdaki beş ana bilgisayar, IoT hizmetlerini simüle etti ve sunucuya periyodik olarak küçük miktarlarda veri göndermiştir. İkinci veri setinde, başlangıçta on IoT cihazı simüle edilmiştir. Bunlardan ikisinin bağlantısı önceden belirlenmiş bir süre sonra kesilmiştir. Dört saldırgan ana bilgisayar, sunucuyu ve IoT cihazlarını hedef almıştır.



Şekil 3.8. İlk veri seti için uygulanan test yatağı kurulumu

SDN uygulamamız, belirli aralıklarla anahtardan akış girişlerini toplamış ve trafik oluşturma sırasında özelliklerimizi oluşturmuştur. Veri kümelerimizdeki tüm özellikler, bir SDN uygulaması kullanılarak elde edilmiştir ve gerçek zamanlı olarak ağ performansını etkilemeden algılama modelleri kullanılmıştır.

3.4.2. Yararlı trafik

Bot-IoT veri setindeki normal kayıtları incelenerek benzer paket gönderme hızları, paket boyutları ve taşıma katmanı protokollerini kullanarak normal trafik oluşturulmuştur. Bot-IoT veri setinde normal trafik için sadece 9543 kayıt vardır, bu nedenle normal trafiği analiz etmek ve çoğaltmak için kısa bir süre yeterli olmaktadır.

Hem saldırı trafiği olan hem de olmayan zararsız trafik oluşturulmuştur. Saldırı senaryoları olmadan 20 kez normal trafik kaydedilmiştir ve bu kayıtlar farklı csv dosyalarına kaydedilmiştir. Her kayıt periyodu 30 ila 35 dakika sürmüştür. İlk 15 kayıtta hem arka plan trafiğini hem de IoT trafiği oluşturulmuştur. Son 5 kayıtta sadece IoT trafiği oluşturulmuştur. Saldırı senaryoları sırasında hem arka plan trafiği hem de IoT trafiği oluşturulmuş ve kaydedilmiştir. Her senaryonun farklı bir csv dosyasına kaydedilmesindeki amaç, tren ve test setleri oluşturulurken her bir senaryodan eşit sayıda kayıt almak ve önyargısız ağ trafiğinin farklı varyasyonlarını oluşturmaktır.

Arka plan trafiği

Arka plan trafiği için, ağdaki iki ana bilgisayar sürekli olarak sunucuya paketler göndermiştir. Ana bilgisayarlardan biri TCP, diğeri UDP kullanmıştır. TCP trafiği için 3 farklı paket gönderme hızı ve 4 farklı paket boyutunun kombinasyonlarını kullanılmıştır. UDP trafiği için 2 farklı paket gönderme hızı ve 2 farklı paket boyutu kombinasyonu kullanılmıştır. Paket gönderme hızları ve paket boyutları, Bot-IoT veri setindeki gözlem sayısıyla orantılı olasılıkla başlangıçta paket oluşturma betiğimiz tarafından belirlenmiştir. Örneğin, TCP paketleri için saniyede 80 paketlik bir gönderme hızı ve 1005 baytlık bir paket boyutu seçme olasılığı %15,8 olarak hesaplanmıştır.

IoT trafiği

Bot-IoT veri setinde, 9543 normal kaydın yaklaşık 6000'i UDP kullanmış ve 100 bayttan az olan yalnızca 1 veya 2 paket gönderilmiştir. IoT cihazları Nodered aracı kullanılarak simüle edilmiş ve simüle edilen hizmetler meteoroloji istasyonu, akıllı buzdolabı, hareketle etkinleştirilen ışıklar, uzaktan etkinleştirilen garaj kapısı ve akıllı termostat olarak belirlenmiştir. Tüm bu hizmetler periyodik olarak küçük miktarlarda veri göndermektedir.

Ağımızdaki IoT cihazlarını simüle ederken bu bayt sayıları paket boyutları olarak da kullanılmıştır. 6 farklı paket boyutu kullanılmış ve her seferinde bir veya iki paket gönderilmiştir. İki paket gönderilmesi durumunda, paketler arka arkaya gönderilmiştir. IoT simülasyon betiğimiz, başlangıçta Bot-IoT veri kümesindeki gözlem sayılarıyla orantılı bir olasılıkla bayt ve paket sayılarını seçmiştir. Örneğin, 86 bayt boyutunda bir paket gönderme olasılığı %16,7 ve 88 bayt boyutunda iki paket gönderme olasılığı %50 olarak hesaplanmaktadır.

Simüle edilmiş IoT cihazları, paketleri sunucuya periyodik olarak göndermiştir. ONOS'un varsayılan akış kuralı zaman aşımı 10 saniyedir. Bir akış girişi, 10 saniye boyunca herhangi bir gelen paketle eşleşmezse, akış girişi silinir. Paket gönderme periyotları da betiğimiz tarafından 12 saniye ile 15 saniye arasında rastgele seçilmiş ve bu gelen her IoT paketinin yeni akış kuralı kurulumunu tetiklemesine neden olmuştur.

Bekleme aralıklarının rastgele seçilmesi nedeniyle, anahtardaki IoT cihazlarına ait akış girişlerinin sayısı dinamik olarak değişmektedir. Sanal IoT ana bilgisayarları için UDP yerine TCP kullanılmıştır. Bunun nedeni, akış kuralı kurulumunu tetikleyen paket bir ‘package_out’ mesajı tarafından iletildikten sonra akış kurallarının kurulmasıdır.

3.4.3. Kötü amaçlı trafik

Ağdaki dört saldırgan ana bilgisayar, sunucuyu veya simüle edilmiş IoT cihazlarını hedef alan siber saldırılar başlatmaktadır. 5 farklı saldırı türü gerçekleştirilmiştir: DoS, DDoS, port tarama, OS parmak izi ve fuzzing. Her saldırı tipi, 10 ve 12 kez farklı konfigürasyonlarda gerçekleştirilmiştir ve kayıt süresi 30 ve 45 dakika olmaktadır. Her yapılandırma farklı csv dosyalarına kaydedilmiştir. Saldırı türleri ve araçları şu şekilde açıklanmaktadır:

DoS

Hizmet Reddi Saldırıları (Moore, Shannon, Brown, Voelker ve Savage, 2006), hedef ana bilgisayarın veya ağın kaynaklarını tüketmeyi ve normal işleyişini bozmayı amaçlar. Bu saldırının iki türü vardır: mantık ve kaynak saldırıları. Mantık saldırıları, yazılım açıklarını bulmaya, sunucuyu çökertmeye veya performansını düşürmeye çalışır. Kaynak saldırıları, sunucunun veya ağın CPU, bellek ve bant genişliği gibi kaynaklarını tüketmeye çalışır. Saldırgan, hedefe çok sayıda sahte paket gönderir ve sunucuyu yasal kullanıcılara yanıt vermemeye zorlamaktadır. Saldırı paketlerini yasal paketlerden ayırt etmek ve saldırının kaynağını bulmak her zaman kolay değildir. Saldırganlar, IP adreslerini, MAC adreslerini ve bağlantı noktası numaralarını kolayca taklit edebilir. Bu kapsamda hedef sunucuyla birçok yeni bağlantı açabilir. Sahte adreslerle yapılan Hizmet Reddi Saldırıları, SDN'lerin denetleyicisi ve anahtarları için sorunlara sebebiyet verebilmektedir. Sahte eşleşme alanı (örneğin IP adresi ve kaynak bağlantı noktası numarası) içeren her paket, yeni bir akış kuralının kurulumunu tetikler. SDN'lerin denetleyicisi tüm bu istekleri işlemeli ve akış kurallarını anahtarlara yüklemelidir. Sahte eşleşme alanlarına sahip DoS, anahtar ile denetleyici arasındaki kanalın bant genişliğini, denetleyicinin CPU'yu ve belleğini tüketir. Bunun yanı sıra anahtarın akış tablolarını taşımaktadır.

Ağ içerisinde, hping3 aracı kullanılarak sunucuyu veya bir IoT cihazını hedef alan Hizmet Reddi Saldırılarını gerçekleştirmek için bir saldırgan ana bilgisayar kullanılmıştır. Hizmet Reddi Saldırılarına yönelik trafik 12 kez kaydedilmiştir. Her kayıt 30 ve 45 dakika sürmüştür. İki kayıtta sunucu, sahte IP adresleri ve kaynak bağlantı noktası numaraları kullanılarak bir TCP SYN saldırısıyla hedef alınmıştır. Diğer iki kayıtta sunucu, sahte IP adresleri ve kaynak bağlantı noktası numaraları kullanılarak bir UDP Flooding saldırısıyla hedef alınmıştır. Kayıtlardan birinde sunucu, sızdırma yapılmadan TCP SYN saldırısı ile hedef alınmıştır. Kayıtlardan birinde sunucu, sahtecilik yapılmadan UDP Flooding saldırısı ile hedef alınmıştır. Kandırma (Spoofing) yapılmayan saldırılarda saldırgandan gelen paketlerin tamamı aynı akış girişi üzerinden geçmekte ve yeni akış kuralı oluşturma durumu tetiklenmemektedir. Kalan 6 kayıtta IoT cihazlarından biri hedef alınarak aynı saldırı senaryoları uygulanmıştır.

Dört farklı paket gönderme hızı kullanılmıştır: saniyede 4000, 6000, 8000 ve 10000 paket. Ayrıca dört farklı paket yük boyutu kullanılmıştır: 0, 100, 500 ve 1000. Bu paket gönderme hızlarının ve yük boyutlarının tüm kombinasyonları kullanılmıştır.

DDoS

Dağıtılmış Hizmet Reddi saldırısı (Chang, 2002), saldırganın birden fazla ana bilgisayarı tehlikeye attığı ve aynı kurbanı hedeflemek için güvenliği ihlal edilmiş makineleri kullandığı bir hizmet reddi saldırısı biçimidir. DDoS saldırısı, ek saldırgan cihazlar nedeniyle DoS saldırısından daha fazla kaynak tüketebilir. Kötü amaçlı paketler farklı ana bilgisayarlardan ve yönlerden geldiği için DDoS saldırılarını önlemek ve kötü amaçlı paketleri normal paketlerden ayırmak da daha zordur. Güvenliği ihlal edilmiş ana bilgisayarlar farklı saldırı stratejileri kullanabilir.

Hping3 aracını kullanarak sunucuyu veya bir IoT cihazını hedef alan DDoS saldırısı gerçekleştirmek için dört saldırgan ana bilgisayar kullanılmıştır. DoS saldırılarıyla aynı senaryolarla 12 kez DDoS saldırısının trafiği kaydedilmiştir. Her kayıt 30 ve 45 dakika sürmüştür. DoS saldırılarıyla aynı paket gönderme hızları ve paket boyutları da kullanılmıştır.

Port tarama

Port tarama (Kaushik, Pilli ve Joshi, 2010), hedefteki açık portları belirleme işlemidir. Saldırgan, açık bağlantı noktalarını kullanarak çalışan güvenlik açıklarına sahip hizmetleri belirlemek amacıyla, belirtilen bağlantı noktası numarası aralığına paketler gönderir.

Port tarama kapsamında nmap aracı kullanılarak sunucuyu veya bir IoT cihazını hedef alan bağlantı noktası tarama saldırısı gerçekleştirmek için bir saldırgan ana bilgisayar kullanılmıştır. nmap'in port taraması için iki mod vardır. Taranması gereken port numaraları belirtilmezse, nmap ilk 1024 port numarasını taramaktadır. Bu port numaralarına iyi bilinen port numaraları denir ve en sık kullanılan servisler için ayrılmıştır. Taranması istenen bağlantı noktası aralığı da belirlenebilmiştir. 10 kez port tarama trafiği kaydedilmiştir. Her kayıt 30 ve 45 dakika sürmüştür. İki kayıtta, sunucunun tüm portunu taramak için nmap kullanılmıştır. Diğer iki kayıtta bir IoT cihazının tüm portları taranmmıştır. Diğer üç kayıtta

sunucunun ilk 1024 port numarası taranmıştır. Son üç kayıtta bir IoT cihazının ilk 1024 port numarası taranmıştır.

OS parmak izi

OS parmak izi (Owens ve Wang, 2011), hedefin işletim sistemi türünü ve sürümünü belirleme işlemidir. Saldırgan, önce hedef üzerindeki açık portları keşfeder ve ardından paketleri kurbana gönderir. Saldırgan, yanıt paketlerinin içeriğini ve gecikmesini analiz eder ve işletim sistemi türünü ve sürümünü belirlemeye çalışır. OS parmak izi genellikle kurban hakkında bilgi toplamak ve zayıf yönlerini belirlemek için kullanılır.
Bir saldırgan ana bilgisayar, nmap aracını kullanarak yalnızca sunucuyu hedefleyen OS parmak izi gerçekleştirmek için kullanılmıştır. IoT cihazları hedef alınmamıştır. Çünkü bu cihazlarda açık port olmamaktadır. Dolayısıyla onları hedeflemek sadece bir port tarama saldırısı ile sonuçlanacaktı. OS parmak izi trafiği 10 kez kaydedilmiştir. Her kayıt 30 ve 45 dakika sürmüştür.

Fuzzing

Fuzzing (Felderer, Büchler, Johns, Brucker, Breu ve Pretschner, 2016); potansiyel güvenlik açıklarını, hataları ve zayıflıkları bulmak için rastgele oluşturulmuş girdilerin hedefe gönderildiği bir test yöntemi ve saldırı tekniğidir. Bulanıklaştırma bir kara kutu yöntemidir, dolayısıyla saldırganın hedef sistemin iç işleyişini bilmesine gerek yoktur. Saldırgan, hedefte çökmeye neden olan girişi inceler ve sistemin güvenlik açıklarını tespit eder. Ardından, zayıflıklardan yararlanmak için özel girdiler oluşturabilir.

Fuzzing saldırıları için boofuzz kullanılmıştır. Boofuzz, belirsiz komut dosyaları oluşturmak için kullanılan bir Python kitaplığı olarak kurulur. HTTP ve FTP fuzzer betikleri uygulanmıştır. Sunucuda HTTP ve FTP sunucu kodu çalıştırılmıştır. Saldırgan ana bilgisayarlardan biri, sunucuya karşı HTTP ve FTP bulanıklaştırma saldırıları gerçekleştirmek için kullanılmıştır. 10 kez fuzzer saldırıları kaydedilmiştir. İlk beş kayıtta FTP fuzzing kullanılmış ve son beş kayıtta HTTP fuzzing kullanılmıştır.

3.4.4. Akış toplama ve özellik oluşturma

Mevcut veri kümelerinin çoğu, trafik oluşturulduktan sonra pcap dosyaları işlenerek oluşturulmuştur. Amaç, SDN'ye özel bir veri seti oluşturmak ve bu veri setini gerçek zamanlı olarak SDN denetleyicisi kullanarak saldırıları önlemek için kullanmaktı. Bu nedenle, her 1 saniyede bir akış girişlerini anahtardan almak ve her akış için özellikler oluşturmak için bir SDN uygulaması geliştirilmiştir. Uygulama, ana bilgisayarların anahtara bağlı olduğu portları kullanarak her akışı etiketlenmiş ve özellikleri csv dosyalarına yazılmıştır. Her saldırı türünün kayıt oturumları farklı csv dosyalarına kaydedilmiştir.

Bot-IoT veri setindeki tüm özellikler oluşturulmaya çalışılmıştır. Bot-IoT veri setinde özellikleri çıkarmak için Argus aracı kullanılmıştır. SDN uygulaması yalnızca anahtardan çekilen akış giriş alanlarını kullanılmıştır. Dolayısıyla veri kümesindeki tüm özellikler SDN'ye özeldir ve bir SDN uygulaması kullanılarak hesaplanabilmektedir. Bot-IoT'de SDN ortamlarına özgü olmayan özellikler (örneğin argus sıra numarası) oluşturulamamaktadır.

Veri kümeleri 33 özellik içermektedir. Özellik adları ve açıklamalar Çizelge 3.2'de açıklanmıştır. Her akışa özgü özellikler mevcuttur (örneğin süre, kaynaktan hedefe paket ve bayt sayısı). Ayrıca her akışa özgü ancak akışın tersi hakkında istatistikler içeren özellikler de mevcuttur (örneğin, hedeften kaynağa saniye başına paket ve paket ve bayt sayısı). Bunun yanı sıra, tüm akış girişleri kullanılarak hesaplanan anahtardaki her akış için ortak özellikler mevcuttur (örneğin, akış sürelerinin standart sapması, kaynak ve hedef IP başına gelen bağlantı sayısı ve kaynak–hedef IP başına toplam paket sayısı). Özelliklerden ikisi etiketdir: saldırı ve kategori. Saldırı, akış girişinin bir saldırı senaryosuna ait olup olmadığını gösterir. Saldırı etiketi, ikili sınıflandırma için kullanılmıştır. Kategori, akış girişlerinin hangi trafik kategorisine ait olduğunu gösterir: normal, DoS, DDoS, port tarama, işletim sistemi parmak izi ve bulanıklaştırma. Kategori etiketi, çok sınıflı sınıflandırma için kullanılmıştır.

Çizelge 3.2. Yazılım temelli ağ veri setlerinin özellikleri

Özellik	Tanımlama
N_IN_Conn_P_DstIP	Hedef IP başına gelen bağlantı sayısı
N_IN_Conn_P_SrcIP	Kaynak IP başına gelen bağlantı sayısı

Çizelge 3.2. (devam) Yazılım temelli ağ veri setlerinin özellikleri

Özellik	Tanımlama
TnP_Per_Dport	Hedef bağlantı noktası başına toplam paket sayısı
TnP_PerProto	Protokol başına toplam paket sayısı
TnP_PDstIP	Hedef IP başına toplam paket sayısı
TnP_PSrcIP	Kaynak IP başına toplam paket sayısı
TnBPDstIP	Hedef IP başına toplam bayt sayısı
TnBPSrcIP	Kaynak IP başına toplam bayt sayısı
Sum	Birleştirilmiş kayıtların toplam süresi
Drate	Saniyede hedeften kaynağa paket sayısı
Dbytes	Hedeften kaynağa bayt sayısı
Sbytes	Kaynaktan hedefe bayt sayısı
Dpkts	Hedeften kaynağa paket sayısı
Spkts	Kaynaktan hedefe paket sayısı
Bytes	İşlemdeki toplam bayt sayısı
Pkts	İşlemdeki toplam paket sayısı
Max	Birleştirilmiş kayıtların maksimum süresi
Min	Birleştirilmiş kayıtların minimum süresi
Stddev	Toplu kayıtların standart sapması
Mean	Birleştirilmiş kayıtların ortalama süresi
Dur	Toplam sürenin kaydı
proto_number	Özellik protokolünün sayısal gösterimi
Protocol	İşlem protokolünün dokusal gösterimi
last_seen	Son kayıt
dstPort	Hedef bağlantı noktası numarası
srcPort	Kaynak bağlantı noktası numarası

Çizelge 3.2. (devam) Yazılım temelli ağ veri setlerinin özellikleri

Özellik	Tanımlama
dstIP	Hedef IP adresi
srcIP	Kaynak IP Adresi
dstMac	Hedef MAC adresi
srcMac	Kaynak MAC adresi

Normal trafik saldırı senaryoları sırasında ve saldırı senaryosu olmadan kaydedilmiştir. Sahte IP adresleriyle yapılan Hizmet Reddi ve DDoS Saldırıları çok sayıda yinelenen akış girişi oluşturmaktadır. Bu nedenle, DoS ve DDoS Saldırıları için akış girişleri anahtardan her çekildiğinde kaydedilen kayıt sayısı 100 ile sınırlandırılmıştır. Bağlantı noktası tarama ve OS parmak izi saldırıları da çok sayıda akış girişi oluşturmuştur, ancak oluşturulan paketler kopya olmadığı için bunlar sınırlandırılmamıştır. İlk SDN veri seti 27,9 milyon giriş içermektedir. Çizelge 3.3, her kategori için katılım sayılarını ve yüzdelerini göstermektedir.

Çizelge 3.3. İlk yazılım temelli ağ veri kümesindeki kayıtların dağılımı

Kategori	Büyüklük	%
Fuzzing	1.18M	4,24
OS ve Servis Tespiti	3.39M	12,15
Port Tarama	20.68M	74,08
DDoS	192K	0,67
DoS	793K	2,84
Normal	1.67M	5,99

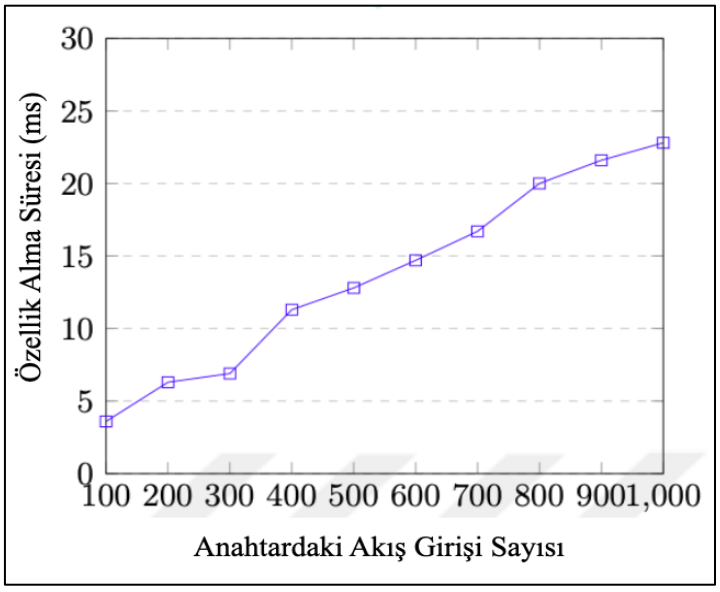
İkinci SDN veri seti de aynı senaryolar kullanılarak oluşturulmuştur. Tek fark, daha önce açıklandığı gibi IoT cihazlarının sayısının fazla olmasıdır. İkinci SDN veri seti 30,2 milyon giriş içermektedir. Çizelge 3.4, her kategori için katılım sayılarını ve yüzdelerini göstermektedir.

Özellik alma süresi çok önemlidir. Saldırılar bittikten veya ciddi hasara yol açtıktan sonra tespit etmenin bir anlamı yoktur. Etkili saldırı tespiti ve önleme için özellikler hızlı bir şekilde alınmıştır. Ayrıca, özellik alma işlemi çok fazla denetleyici kaynağı tüketmemelidir, aksi takdirde ağ performansı olumsuz etkilenmektedir.

Çizelge 3.4. İkinci yazılım temelli ağ veri kümesindeki kayıtların dağılımı

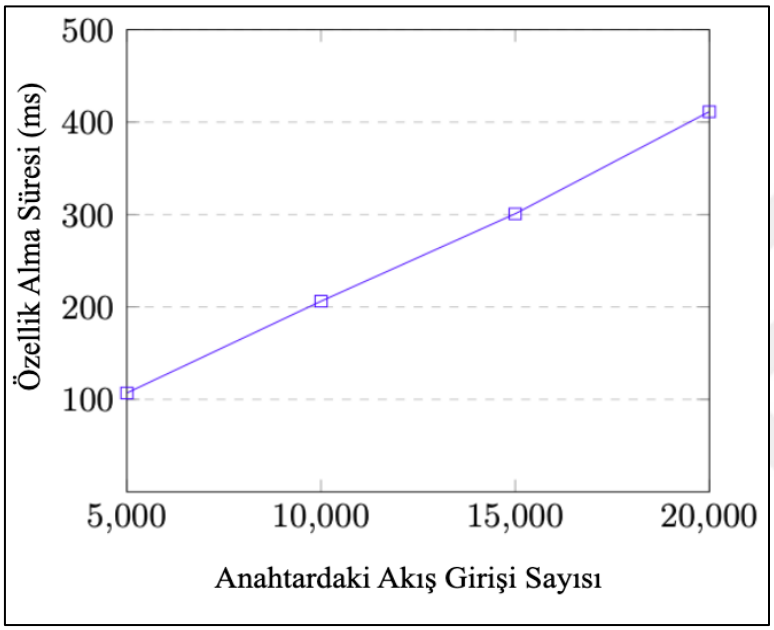
Kategori	Büyüklük	%
Fuzzing	1,05M	3,48
OS ve Servis Tespiti	3,39M	11,20
Port Tarama	22,44M	74,23
DDoS	182K	0,60
DoS	495K	1,67
Normal	2,67M	8,84

Şekil 3.9, normal trafiğe karşılık gelen, anahtarda 1000 akış girişine kadar akış girişi toplama ve özellik oluşturma süresini göstermektedir. Anahtarda 1000 akış girişi olduğunda, tüm akışlar için akış toplama ve özellik oluşturma süresi 22,8 milisaniye olmuştur ve bu oldukça düşük bir değer olarak görülmektedir.



Şekil 3.9. 1000 akış girişine kadar özellik alma süresi

Şekil 3.10, saldırı trafiğine karşılık gelen, anahtardaki 20000 akış girişine kadar akış girişi toplama ve özellik oluşturma süresini göstermektedir. Anahtarda 20000 akış girişi olmasına rağmen, SDN uygulaması akış girişlerini topladı ve tüm akışlar için 411.3 milisaniyede özellikler oluşturmuştur. Bu da denetleyici için çok fazla ek yüke sebebiyet vermemektedir. Özellik alma süresi, anahtardaki akış girişlerinin sayısıyla doğrusal olarak artmış ve algoritma O(n) olarak belirtilmiştir.



Şekil 3.10. 20000 akış girişine kadar özellik alma süresi

3.4.5. Model değerlendirmesi

Bu bölümde, oluşturulan SDN veri kümelerinden tren ve test veri kümelerinin nasıl elde edildiği ve farklı makine öğrenimi algoritmalarının sonuçları değerlendirilmiştir.

Ön işleme

Veri setleri milyonlarca kayıt içermektedir ve her kategoriye ait kayıt sayısı daha önce Çizelge 3.3 ve Çizelge 3.4'de gösterildiği gibi değildir. Milyonlarca veri kaydının işlenmesi mümkün değildir ve fazla uydurmaya yol açabilir. Ayrıca dengesiz veri kümeleri, önyargılı modellere neden olabilir. Kayıtların yaklaşık %74'ü port tarama saldırısına aittir. Bu nedenle model eğitimi için her kategoriden eşit sayıda kayıt almak gerekmiştir. Ayrıca bir

kategorinin her kaydından eşit sayıda kayıt almak önem taşımaktadır. Bunun nedeni ise yapılandırmaya ve hedefe bağlı olarak kayıt sayılarının çok farklı olmasıdır. Örneğin, kandırma olmadan DoS Saldırılarından biri 3251 ile en düşük kayıt sayısına sahipken, kandırma ile Dos Saldırıları 137000'e kadar kayıt yapmaktadır. DoS Saldırıları trafiği 12 kez kaydedilmektedir. Yani alınabilen maksimum kayıt sayısı 39012 olmaktadır. Bu nedenle, saldırı tipinin her senaryosuna yönelik eşit olarak alınan her saldırı kategorisinden 35000 kayıt alınmış ve toplamda 175000 saldırı kaydı elde edilmiştir.

Çok kümeli sınıflandırma için 35000 normal kayıt alınmıştır. DoS, DDoS, port tarama, OS parmak izi, fuzzing ve saldırısız normal trafik dosyalarından her biri 5834 olarak normal kayıtlar alınmıştır. Oluşturulan veri setinde her kategoriden toplam 210.000 kayıt olmak üzere 35.000 kayıt bulunmaktadır. Bu veri seti eğitim ve test setlerine ayrılmıştır. Eğitim setinde her kategoriden 25000 kayıt olmak üzere toplam 150000 kayıt bulunmaktadır. Test setinde her kategoriden 10000 kayıt olmak üzere toplam 60000 kayıt yer almaktadır.

İkili sınıflandırma için yukarıda açıklandığı gibi toplam saldırı kaydı sayısına eşit 175000 normal kayıt alınmıştır Oluşturulan veri setinde toplamda 350000 kayıt bulunmaktadır. Bu veri seti eğitim ve test setlerine ayrılmıştır. Eğitim setinde 250000 kayıt ve test setinde 100000 kayıt bulunmaktadır. Her iki veri seti için de aynı prosedür izlenerek her ikisi için de eğitim ve test veri setleri oluşturulmuştur.

Performans metrikleri

Sınıflandırma için literatürde en sık kullanılan metrikler doğruluk, kesinlik, hatırlama ve F1 puanıdır. Çizelge 3.5, karışıklık matrisini (hata matrisi) ve metrik hesaplamalar için gerekli tanımları göstermektedir.

Çizelge 3.5. Karışıklık matrisi

		Tahmini Sınıf	
		Pozitif	Negatif
Gerçek Sınıf	Pozitif	Doğru Pozitif (DP)	Yanlış Pozitif (YP)
	Negatif	Yanlış Negatif (YN)	Doğru Negatif (DN)

Kesinlik

Doğru sınıflandırılmış pozitif ve negatif kayıtların toplam kayıt sayısına oranı.

$$Kesinlik = \frac{DP+DN}{YP+YN+DP+DN} \quad (3.6)$$

Hassasiyet

Doğru olarak sınıflandırılan pozitif kayıtların, pozitif olarak sınıflandırılan toplam kayıtlara oranı.

$$Hassasiyet = \frac{DP}{DP+YP} \quad (3.7)$$

Duyarlılık

Doğru sınıflandırılmış pozitif kayıtların toplam pozitif kayıtlara oranı.

$$Duyarlılık = \frac{DP}{DP+YN} \quad (3.8)$$

F1 Puanı

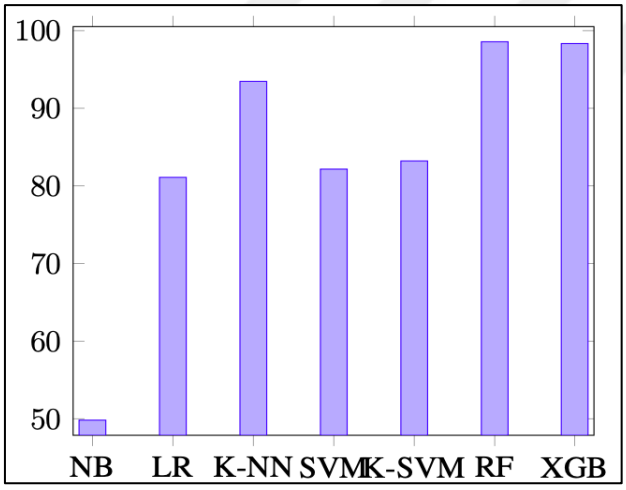
Hassasiyet ve duyarlılık arasında dengeli ölçüm.

$$F1 = 2\,x\,\frac{Duyarlılık\ x\ Hassasiyet}{Duyarlılık+\ Hassasiyet} \quad (3.9)$$

Performans ölçümleri, hedeflere ve sistem gereksinimlerimize göre seçilmiştir. Yanlış pozitif miktarı azaltılmak istendiğinde duyarlılık iyi bir ölçümdür. Bunun yanı sıra, yanlış negatif miktarını azaltılmak istendiğinde duyarlılık iyi bir performans sergilemektedir. F1 puanı hassasiyet ve duyarlılık arasında bir denge sağlamaktadır. Bu nedenle, deneylerde değerlendirme ölçütünün F1 puanı seçilmiştir.

Çok sınıflı sınıflandırma

Oluşturulan eğitim ve test setleri, farklı makine öğrenmesi algoritmalarının performanslarını değerlendirmek için kullanılmıştır. İlk SDN veri setinin eğitim ve test setleri kullanılarak farklı makine öğrenimi algoritmaları eğitilmiş ve test edilmiştir. Farklı algoritmaların çok sınıflı sınıflandırmasının sonuçları Şekil 3.11'de gösterilmektedir: Naive Bayes (NB), Lojistik Regresyon (LR), Nearest Neighbor (K-NN), Destek Vektör Makineleri (SVM), Çekirdek Destek Vektör Makineleri (K- SVM), Random Forest (RF) ve XGBoost (XGB). RF ve XGB, diğer algoritmalardan daha iyi performans ortaya koymuştur. İki veri seti oluşturmaktaki amacımız, ilk veri seti ile eğitilen modelleri kullanarak ikinci veri seti üzerinde test etmek ve sistemin ağ değişikliklerinden nasıl etkileneceğini görmektir. Eğitilmiş Random Forest ve XGBoost modellerini kullanarak ikinci veri setinin test setindeki modelleri değerlendirdiğimizde, her iki model de ilk veri setindeki performanstan çok daha düşük olan yaklaşık %80 F1 puanı elde edilmiştir.



Şekil 3.11. Model karşılaştırması

Özellik önem özelliği, eğitim setindeki her bir özelliğin kirlilik tabanlı özellik önemini ortaya çıkarmıştır. Özellik önem değerlerinin ortalamasından daha yüksek özellik önemine sahip özellikler kullanılmıştır. XGBoost modelinin ikinci veri setindeki performansı düşmüş ve Random Forest modelinin performansı artmıştır. Bu nedenle Random Forest modeli daha önemli hale gelmiştir. Önemi ortalamaya yakın bir özellik daha ekledik ve 10 özellik elde ettik. Seçilen özellikler ve açıklamaları Çizelge 3.6'da gösterilmektedir. Seçilen öznitelikler ile eğitilen RF modelinin ilk veri setindeki genel F1 puanı %97,86 olmuştur. Hem eğitim

verilerini hem de özellik sayısını yarıdan fazla azaltmamıza rağmen performans, tüm eğitim verileri ve 24 özellikle eğitilen modele yakın olarak ölçülmüştür. Daha az özellik kullanmak, SDN uygulamamızın özellikleri çok daha hızlı almasına da olanak tanımaktadır. İlk veri kümesindeki tüm sınıflar için performans ölçümleri Çizelge 3.7'de gösterilmektedir.

Çizelge 3.6. Başlangıçta belirlenen özellikler

Özellik	Tanım
Sum	Birleştirilmiş kayıtların toplam süresi
Mean	Birleştirilmiş kayıtların ortalama süresi
Dbytes	Hedeften kaynağa bayt sayısı
Dur	Toplam sürenin kaydedilmesi
N_IN_Conn_P_SrcIP	Kaynak IP başına gelen bağlantı sayısı
Spkts	Kaynaktan hedefe paket sayısı
N_IN_Conn_P_DstIP	Hedef IP başına gelen bağlantı sayısı
TnP_Per_Dport	Hedef bağlantı noktası başına toplam paket sayısı
Sbytes	Kaynaktan hedefe bayt sayısı
TnP_PSrcIP	Kaynaktan IP başına toplam paket sayısı

Normal trafiğin en düşük F1 puanına sahip olması, normal paketleri kötü niyetli paketler olarak sınıflandırmak ve meşru trafiği engellemek istemediğimiz için olumsuz bir sonuç olarak öne çıkmıştır. Test setimizdeki normal kayıtların altıda beşi, saldırı senaryoları sırasında normal trafiğe aittir. Bir saldırı sırasında normal trafiği saldırı trafiğinden ayırmak zor bir iş olarak öne çıkmıştır. Bu nedenle, bir saldırı tespit ettikten sonra harekete geçmeden önce emin olunması gerekmektedir. Anlamlı bir müdahale için bu zorunlu bir gereklilik olarak görülmüştür.

Çizelge 3.7. Başlangıçta seçilen 10 özelliği kullanan RF performansı

Sınıf	Duyarlılık	F1 Puanı	Kesinlik
DDoS	98,87	98,03	97,21
DoS	98,66	98,37	98,08
OS and service detection	98,37	98,36	98,35
Fuzzing	99,41	98,86	98,32
Normal	92,75	94,69	96,71
Port scanning	99,19	98,86	98,54

Herhangi bir saldırı trafiği olmadığında, modelimizin normal trafiği tespit etme doğruluğu oldukça yüksek olan %99,67 olarak tespit edilmiştir. İkinci veri setindeki genel F1 puanı, başlangıçta seçilen 10 özellik kullanılarak %84,48 olarak hesaplanmıştır. Çizelge 3.8'de listelenen özellikler ve Çizelge 3.9'da listelenen Hiperparametreler kullanılırken iki özellik değiştirilmiş ve ikinci veri setindeki genel performans %91,01'e yükselmiştir. Modelimizin ikinci veri setindeki normal trafik performansı, ilk veri setindeki performansa benzerlik gösterse de genel performans ilk veri kümesinden daha düşüktü çünkü modelimiz, ikinci veri kümesindeki IoT cihazlarının sayısının artması nedeniyle beklendiği gibi ikinci veri kümesindeki bazı DoS saldırılarını DDoS saldırıları olarak sınıflandırılmıştır. Alınan azaltma eylemi aynı olduğundan, ağ performansı etkilenmemiştir. Özelliklerin değiştirilmesi, modelin performansını ilk veri setinde çok fazla etkilememiş ve genel F1 puanının %97,82 olarak ölçülmüştür.

Çizelge 3.8. Çok sınıflı sınıflandırma için en önemli 10 özellik

Özellik	Tanım
Sum	Birleştirilmiş kayıtların toplam süresi
Mean	Birleştirilmiş kayıtların ortalama süresi
Dbytes	Hedeften kaynağa bayt sayısı
Dur	Toplam sürenin kaydedilmesi

Çizelge 3.8. (devam) Çok sınıflı sınıflandırma için en önemli 10 özellik

Özellik	Tanım
Spkts	Kaynaktan hedefe paket sayısı
Sbytes	Kaynaktan hedefe bayt sayısı
TnP_PSrcIP	Kaynaktan IP başına toplam paket sayısı
Stddev	Birleştirilmiş kayıtların standart sapması
TTnP_PDstIP	Hedef IP başına toplam paket sayısı
TnP_Per_Dpor	Hedef bağlantı noktası başına toplam paket sayısı

İlk veri setindeki her sınıf için F1 puanları Çizelge 3.10'da yer almaktadır. Sonuçlar başlangıçta seçilen özelliklere benzemektedir.

Çizelge 3.9. Çok sınıflı sınıflandırma için RF modelinin hiperparametreleri

Hiperparametreler	Değer
max_depth	40
min_samples_split	6
max_features	auto
n_estimators	100
Bootstrap	Doğru
Criterion	entropy
min_samples_leaf	1
random_state	0

Çizelge 3.10. En iyi 10 özelliği kullanan RF performansı

Sınıf	Duyarlılık	F1 Puanı	Kesinlik
DDoS	98,58	97,93	97,29
DoS	99,12	98,44	97,77

Çizelge 3.10. (devam) En iyi 10 özelliği kullanan RF performansı

Sınıf	Duyarlılık	F1 Puanı	Kesinlik
OS and service detection	97,76	98,08	98,40
Fuzzing	99,32	98,76	98,20
Normal	92,92	94,80	96,75
Port scanning	99,27	98,89	98,52

İkili sınıflandırma

İkili sınıflandırma için çok sınıflı sınıflandırmada kullanılan aynı eğitim ve test veri setleri kullanılmıştır. Eğitim setindeki 25000 normal kaydın tamamı alınmış ve 5 saldırı kategorisinin her birinden 5000 kayıtla birleştirilerek ikili sınıflandırma için dengeli bir eğitim seti haline getirilmiştir. RF sınıflandırıcısı kullanılmış ve aynı öznitelik seçimi ve hiperparametre seçimi işlemleri gerçekleştirilmiştir. İkili sınıflandırma modelimizde Çizelge 3.11'de gösterilen 12 özelliği kullanmıştır. Bu 12 özellikten 7 tanesi çok sınıflı modelde ortaktır. İkili sınıflandırma modeli, yalnızca 4 tanesini (Dur, Spkts, Sbytes, Dbytes) kullanan çok sınıflı sınıflandırma modelinin aksine, akışa özgü tüm özellikleri (Dur, Spkts, Dpkts, Sbytes, Dbytes, Srate, Drate) kullanmıştır. Her iki model de Dur, Mean ve Stddev özelliklerini kullanılmış, modellerin kullandığı kalan özellikler ise farklıdır.

Çizelge 3.12'de ikili sınıflandırma modelimiz için en iyi hiperparametreler yer almaktadır. İkili sınıflandırma modelinin en iyi hiperparametreleri, çok sınıflı sınıflandırma modelinin en iyi hiperparametrelerine benzerdir. Yalnızca *max_features* ve *min_samples_leaf* hiperparametrelerinin değerleri farklıdır. İkili sınıflandırma modeli, çok sınıflı sınıflandırma modelinin performansından %0,2 daha düşük olan genel olarak %97,64 F1 puanı elde etmiştir ancak ikili sınıflandırma modeli, saldırı trafiğinin olmadığı normal trafik için %100 doğruluk elde etmiştir.

Ayrıca ikili sınıflandırma modelimizi, ikinci veri setimizde test ettiğimizde genel F1 puanının %97,01'ini elde ettik. Çok sınıflı sınıflandırma modelinin ikinci veri setindeki performansı %91,01 olarak ölçülmüştür. Modeller arasındaki ikinci veri setinde bu

performans farkını bekliyorduk, çünkü çok sınıflı sınıflandırma modeli bazı DoS kayıtlarını DDoS olarak yanlış sınıflandırmaktaydı. İkili sınıflandırmada, kötü amaçlı trafiğin tamamı saldırı olarak sınıflandırılmıştır.

Çizelge 3.11. İkili sınıflandırmada kullanılan 12 özellik

Özellik	Tanım
Sum	Birleştirilmiş kayıtların toplam süresi
Mean	Birleştirilmiş kayıtların ortalama süresi
Dbytes	Hedeften kaynağa bayt sayısı
Dur	Toplam sürenin kaydedilmesi
Spkts	Kaynaktan hedefe paket sayısı
Sbytes	Kaynaktan hedefe bayt sayısı
N_IN_Conn_P_SrcIP	Hedef IP başına gelen bağlantı sayısı
Stddev	Birleştirilmiş kayıtların standart sapması
N_IN_Conn_P_DstIP	Kaynak IP başına gelen bağlantı sayısı
TnP_Per_Dport	Hedef bağlantı noktası başına toplam paket sayısı
Dpkts	Hedeften kaynağa paket sayısı
Srate	Saniyede kaynaktan hedefe paket sayısı
Drate	Saniyede hedeften kaynağa paket sayısı

Çizelge 3.12. İkili sınıflandırma için RF modelinin hiperparametreleri

Hiperparametreler	Değer
max_depth	40
min_samples_split	6
max_features	log2
n_estimators	100

Çizelge 3.12. (devam) İkili sınıflandırma için RF modelinin hiperparametreleri

Hiperparametreler	Değer
Bootstrap	True
Criterion	Entropy
min_samples_leaf	6
random_state	0

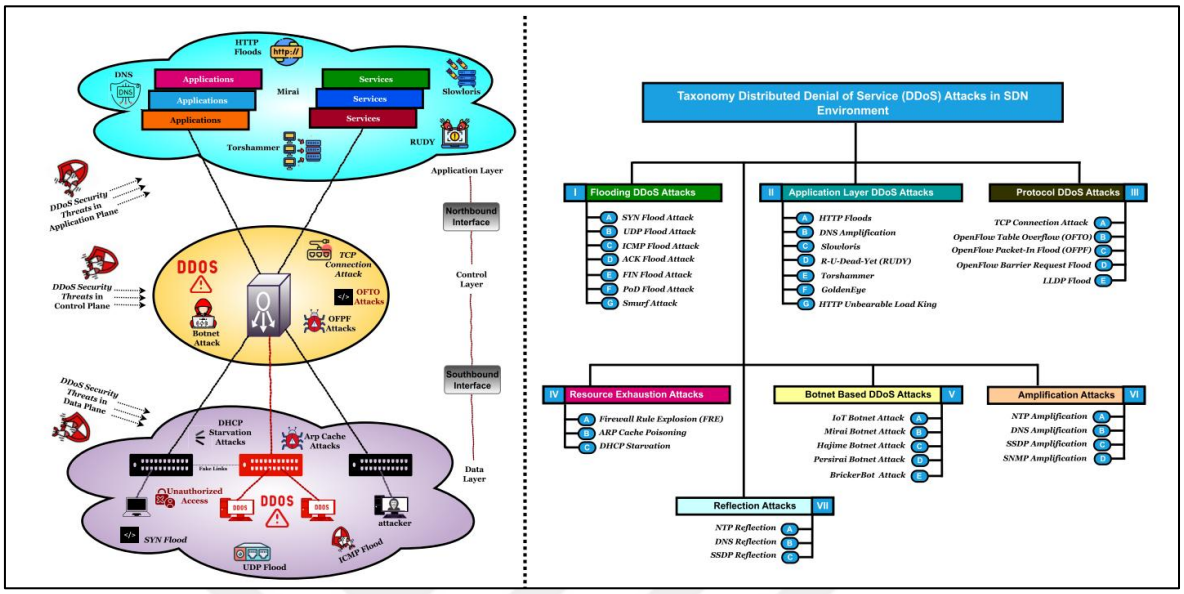
Seçili özellikler

Çok sınıflı ve ikili sınıflandırma için seçilen özellikler açıklanmaktadır. Bazı özellikler, anahtardaki tüm akış girişleri için ortaktır. Ortak özellikler, saldırıların varlığını tespit etmek için kullanılmış ve geri kalan özellikler, iyi huylu akışları kötü amaçlı olanlardan ayırmak için kullanılmıştır:

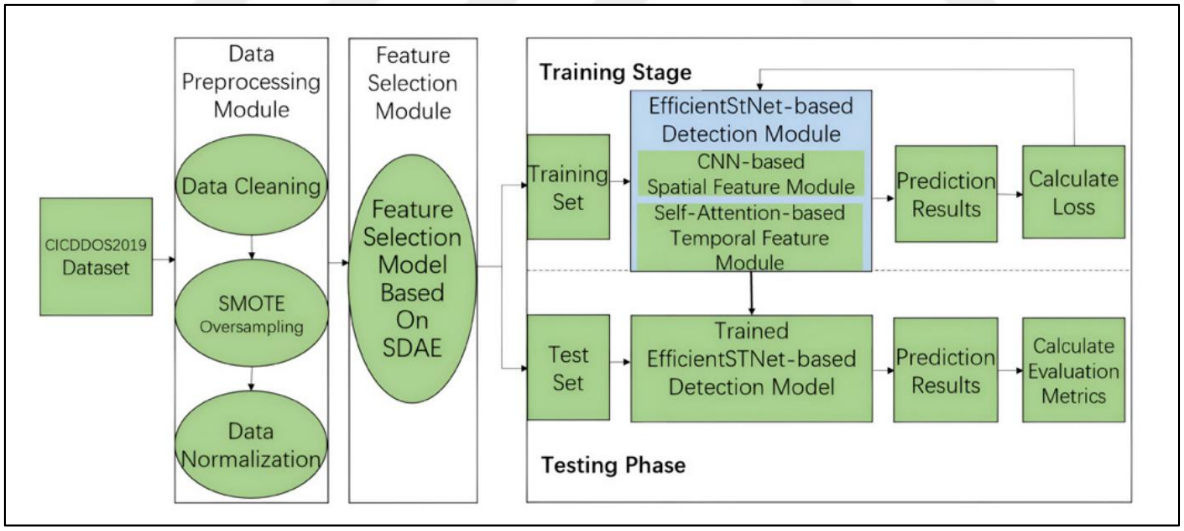
- Dur, Mean, Stddev ve Sum: Switch'teki akış girişlerinin süresi ile ilgili istatistikler, saldırıları tespit etmek için kullanılmıştır. Saldırılar sırasında, anahtarlara çok sayıda yeni akış girişi yüklenmiştir. Yeni yüklenen akış girişleri, akış kurallarının süresinin ortalamasını ve stddev'ini azaltmıştır. Bu nedenle, bu özellikler ağdaki saldırının varlığını tespit etmek için önemlidir. Öte yandan süre, saldırıların varlığında ve yokluğunda kötü niyetli ve zararsız akışların belirlenmesine yardımcı olur. Saldırı akışlarının çoğu için Spkts, Srate ve Sbytes Kaynaktan hedefe paket ve bayt sayıları düşüktür. Çoğu zaman saldırganlar, saldırının kaynağının tespit edilmesini zorlaştırmak için kimlik sahtekarlığı kullanır. Sahte adreslerle gelen her saldırı, yeni bir akış girişi kurulumunu tetikler ve saldırı türüne bağlı olarak bu akış girişlerinden yalnızca 1 veya 2 paket geçer. Ayrıca, hedefe daha fazla paket göndermek için bu paketlerin baytları genellikle düşüktür. Spoofing yapılmayan DoS ve DDoS saldırılarında, saldırı paketlerinin tamamı aynı akış girişi üzerinden geçer ve kaynaktan hedefe paket sayısı, paket hızı ve bayt sayıları önemli ölçüde artar.
- Dpkts, Drate ve Dbytes**:** Ters akış istatistikleri de önemlidir. Saldırı akışları için ters akış değerleri genellikle iyi huylu akışlardan daha düşüktür. Çoğu zaman denetleyici, sahte adres ağda bulunmadığından, paketin saldırganın sahte adresine nasıl yönlendirileceğini bilmez.

- TnP_Per_Dport: Hedef port başına toplam paket sayısı, saldırıları tespit etmek için bir diğer önemli özelliktir. DDoS ve DDoS saldırısının hedefi bir web sunucusu ise tüm paketler aynı porta gönderilir. Bu nedenle, hedef port başına paket sayısı artar. Öte yandan, port tarama saldırısı sırasında paketlerin çoğu hedefin farklı portlarına gönderilir ve hedef port başına düşen paket sayısı azalır.
- N_IN_Conn_P_DstIP: Saldırganlar sahte kaynak adresleri kullanıyorsa hedef IP başına gelen bağlantı sayısı çok artar. Saldırganlardan gelen tüm paketler, aynı hedef IP ile yeni bir akış girişi kurulumunu tetikler.
- N_IN_Conn_P_SrcIP: Kaynak IP başına gelen bağlantı sayısı, sahte kaynak adresleriyle yapılan saldırıları tespit etmek için de kullanışlıdır. Sahte IP adresleri muhtemelen ağda yoktur, bu nedenle denetleyici çoğu zaman saldırı paketleri için ters akışlar kuramaz. Sahte kaynak adresleriyle yapılan saldırılarda kaynak IP başına gelen bağlantı sayısı azalır.
- TnP_PSrcIP: N_IN_Conn_P_SrcIP'ye benzer şekilde, kaynak IP başına toplam paket sayısı, saldırı tespiti için bir başka önemli özelliktir. Değeri sahtekarlığa bağlıdır. Saldırganlar sızdırma kullanmazlarsa bu özelliğin değeri çok artar. Kimlik sahtekarlığında, artan akış giriş sayısı nedeniyle değer daha düşüktür.
- TnP_PDstIP: Hedef IP başına toplam paket sayısı, kimlik sahtekarlığı olan ve olmayan saldırıları tespit edebilen kullanışlı bir özelliktir. Saldırganlar hedef IP adresine çok sayıda paket gönderir ve hedef IP başına paket sayısı hızla artar.

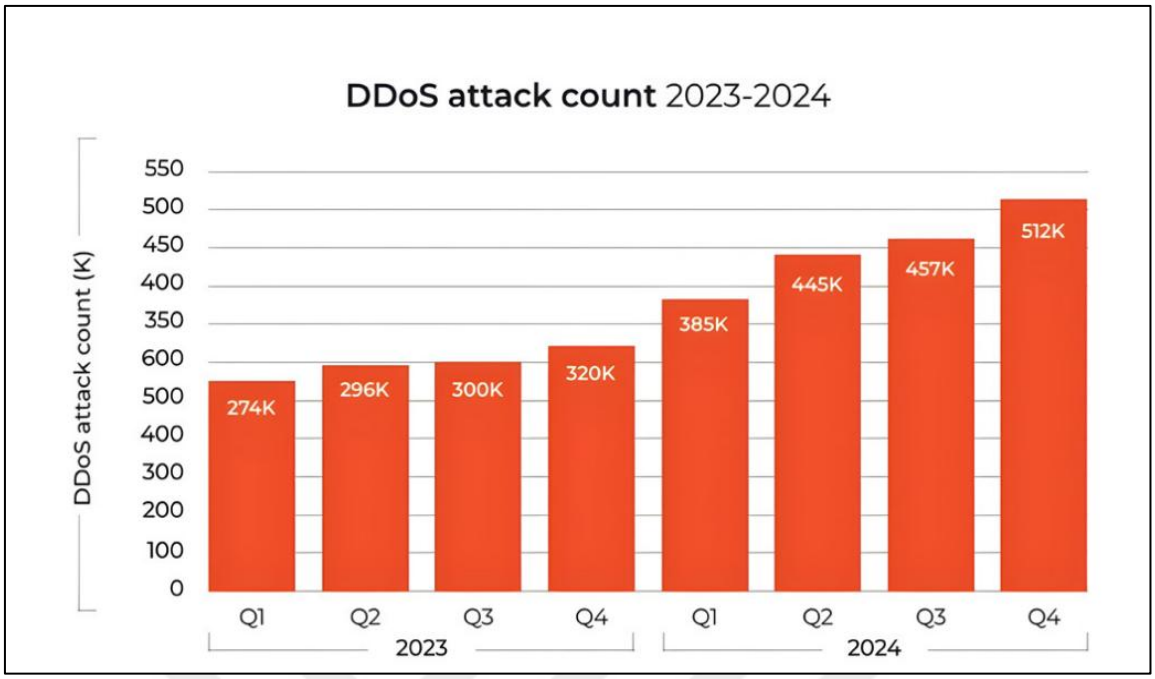
3.5. Güncel Çalışmalar



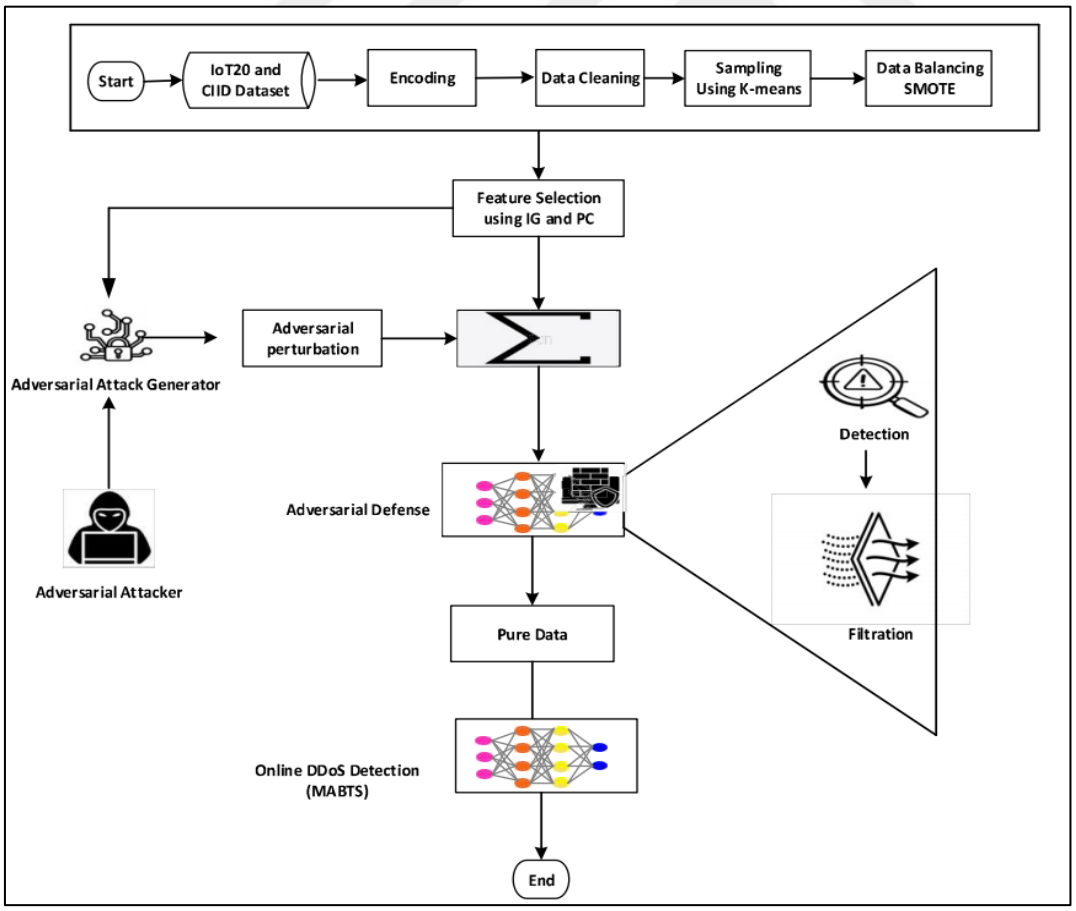
Şekil 3.12. SDN'de DDoS saldırılarının kapsamlı sınıflandırması ve katmanlı güvenlik açıkları. (Abdınasır Hırsı 2025)



Şekil 3.13. DDoS algılama modeli çerçeve diyagramı. (Lei Zhang 2025)



Şekil 3.14. DDoS saldırılarının sayısı (2023-2024). (Leı Zhang 2025)



Şekil 3.15. MSAAD Framework genel bakış (Besnah 2025)

Önerilen Multi-Stage Adversarial Attack Defence(MSAAD) framework çerçevesi üç temel aşama üzerinden işler:

Dayanıklı Düşman Tespiti ve Temizleme (RADP),

Çoklu sınıflandırıcılar,

Topluluk savunması. (Besnah 2025)

4. SONUÇ VE ÖNERİLER

DDoS saldırıları, ağ oluşturma alanında her zaman büyük bir problem olarak görülmüştür. Ancak SDN'ler açısından DDoS saldırılarının etkileri geleneksel ağlardan daha yıkıcı olmaktadır. Esas olarak, SDN'ler bu saldırıların gerçek hedefi durumuna gelebilmekle beraber bu saldırılardan dolayı bütün bir SDN yasal kullanıcılar tarafından kullanılamaz hale gelebilmektedir. Bu tezde, çeşitli ihtiyaçlara çözüm olarak veri katmanı tabanlı bir DDoS algılama ve azaltma metodunun, SDN'lere yönelik ortam açısından DDoS saldırılarına karşı etkili bir savunma mekanizması sağlayacağı önerilmektedir. Önerinin doğrulanması bakımından veri katmanı tabanlı bir algılama ve azaltma çerçevesi olan MiddleModule sistemi sunulmuş ve birkaç algılama algoritması önerilmiştir. Bu tezdeki amaç, DDoS saldırıları altında başarısızlığa yol açabilen SDN'lerin temel problemlerine açıklık getirmektir. Bunun yanı sıra, düşük sistem performansı ve yüksek savunma gücü performansı ile DDoS saldırılarını etkin bir şekilde tespit edebilen ölçeklenebilir ve kapsamlı bir savunma mekanizması sunulmaktır.

Bir SDN'nin her uç anahtarının istatistikleri topladığı ve gelen paketlere istatistiksel tabanlı algılama algoritmaları uyguladığı MiddleModule çerçevesi sağlanarak herhangi bir kötü amaçlı etkinlik algılandığında ise, ilgili uç anahtarı tarafından uygun azaltma tekniği uygulanmıştır. Burada önerilen savunma sistemi ile kötü amaçlı trafik denetleyiciye gönderilmemektedir. Çünkü denetleyicinin aşırı yüklenmesini ve anahtar-denetleyicinin iletişim kanalı tıkanıklık sorunlarının önlenmesi amaçlanmıştır. Kötü amaçlı trafiği, kaynağına en yakın anahtarda tespit edip tepki vermek önemlidir. Önerilen yöntem açısından işlevsel analiz ve performans değerlendirmeleri yapılmış ve test sonuçları literatürde bulunan benzer araştırmalar ile karşılaştırılmıştır.

SDN'ler açısından veri katmanı tabanlı bir DDoS savunma sisteminin karşılaması gereken, hafif algoritmalara sahip olmak ve çok konumlu savunma yaklaşımı kullanmak gibi bazı temel özellikler üzerinde durulmuştur. Tavsiye edilen nitelikler kullanarak, uç anahtarlara izleme, algılama ve önleme yetenekleri getiren veri katmanı tabanlı bir DDoS savunma mekanizması olan MiddleModule çerçevesi açıklanmıştır. Bunun yanı sıra, her bir ağ saldırı türünün en yaygın örneklerine karşı etkili olması beklenen birkaç DDoS algılama algoritması da sunulmuş ve bu algoritmalar MiddleModule çerçevesi içerisinde uygulanmıştır. Önerilen

metot kapsamında farklı test senaryoları altında detaylı bir fonksiyonel analiz ortamı için OMNET ortamı göz önünde bulundurulmuştur. Simülasyona ek olarak, yöntemin performansını analiz etmek için önerilen yöntem MİNİNET ortamında da benzetilmiştir.

Gerçek zamanlı saldırıların önlenmesi sistemi doğrultusunda mobil ağlar, büyük veri ve bulut bilişim alanlarındaki gelişmeler sayesinde bağlantılı cihazların sayısı ve IoT kullanımı devamlı artış göstermektedir. Kaynak yönetimi için SDN'ler ve NFV dayanan 5G ve ötesi ağlar, geleceğin her yerde bulunan IoT kapsamında önem arz etmektedir. SDN'ler katmanında IoT ağlarına yönelik hali hazırdaki güvenlik yaklaşımlarının eksiklikleri gidermek amacıyla, bu tezde gerçek zamanlı bir saldırı tespit ve önleme sistemi sunulmuştur.

SDN'ler için, denetleyicideki trafik akışlarını izleyen ve makine öğrenimine dayalı akıllı bir saldırı algılama modülü kullanarak trafiği sınıflandıran, akışı buna göre yapan uçtan uca bir saldırı tespit ve önleme sistemi mimarisi gerçekleştirilmiştir. Normal trafik ve farklı saldırı trafiği türlerini içeren bir Nesnelerin İnterneti ağında yeni bir SDN'ye yönelik veri kümesi belirtilmiştir. Sunulan veri setinin performansı farklı makine öğrenme algoritmaları ile değerlendirilmiş en önemli özellikler belirlenmiştir. Bunun yanı sıra sunulan güvenlik sisteminin varlığında ve yokluğunda ağ performansı neticeleri belirtilerek sistemin gerçek zamanlı korumaya ulaşması hedeflenmiştir.

Ağ/Aktarım katmanında DDoS Saldırıları, SDN mimarisi için önemli bir sorundur ve bu saldırılar, SDN'ler için başarısız olmasına yol açabilecek yıkıcı sorunlara neden olur. Literatürde SDN'ler için Ağ/Aktarım katmanında DDoS Saldırılarına karşı genel kabul görmüş bir çözüm yoktur. Ayrıca bu alanda çözülmemiş birçok sorun bulunmaktadır. Mevcut yöntemler, savunma sistemlerini kontrol katmanında veya veri katmanında konuşlandırmayı önermektedir. Bazı kontrol katmanı tabanlı sistemler umut verici savunma gücü değerleri sağlasa da denetleyicinin hesaplama yükünü arttırırlar. Bunun yanı sıra, doğru kararlar için anahtarlar ve denetleyici arasında mesaj alışverişine ihtiyaç duyarlar, bu da organize bir DDoS saldırısı altında iletişim kanalını daha da sıkışık hale getirir. Ayrıca, tek hata noktası sorunları vardır ve gelişmiş bir DDoS saldırısı altında etkisiz hale getirilebilirler. Bununla birlikte, veri katmanı tabanlı yaklaşımlar daha sağlam savunma sistemleri sağlar. Literatürde sağlanan birkaç veri katmanı tabanlı DDoS savunma mekanizması bulunmakla beraber çalışma içerisinde bahsedildiği gibi, değerlendirme ölçütlerine ilişkin bazı sakıncaları bulunmaktadır.

Bu çalışmada, SDN'leri, Ağ/Aktarım katmanında DDoS saldırılarına karşı korumak için MiddleModule tespit ve önleme sistemi önerilmektedir. MiddleModule, veri katmanı düğümlerinde izleme, tespit ve önleme yeteneklerini kullanan, veri katmanı tabanlı bir savunma sistemidir. Bu sistem tasarlanmadan önce, veri katmanı tabanlı bir sistemin karşılaması gereken gereksinimler analiz edilerek açıklanmıştır. MiddleModule sistemi bu gereksinimler dikkate alınarak tasarlanmıştır. Bu sistemde, ağın kenar anahtarları, gelen paketlerin bazı istatistiklerini izleyerek ve bu istatistikleri kullanarak basit eşik tabanlı algılama fonksiyonlarını çalıştırmıştır. Tespit fonksiyonlarında, bazı paket özniteliklerinin DDoS saldırı türlerine göre değişimi analiz edilmiştir. Denetleyici ayrıca tespit kararlarına göre bilgilendirilmiş ve ilgili uç anahtar üzerinde uygun önleme yöntemi uygulanmıştır. MiddleModule sistemi kapsamlı ve doğru performans sonuçları elde etmek için farklı değerlendirme platformlarında test edilmiştir. Değerlendirme sonuçları, MiddleModule sisteminin farklı test senaryoları altında gerektiği gibi çalıştığını göstermiştir. MiddleModule sistemi hem ideal hem de ideal olmayan koşullar altında Ağ/Aktarım katmanında DDoS saldırılarına karşı göz ardı edilebilir sistem performansı düşüşü ve dikkate değer savunma gücü performansları sağlamıştır. Test sonuçları, MiddleModule sistemine yönelik olarak kötü niyetli ana bilgisayarların hem yüksek hem de düşük üretim hızlarında paketler oluşturduğunda doğru şekilde çalışabileceğini göstermiştir. Test sonuçları, daha yeni ve gelecek vaat eden dört veri katmanı tabanlı savunma sistemi ile karşılaştırılmıştır. Karşılaştırma sonuçları, MiddleModule sisteminin, değerlendirme metriklerinin literatürdeki mevcut çalışmalardan üstün ve diğer birçok açıdan karşılaştırılabilir olduğunu göstermektedir.

Bu çalışmada değerlendirmeler açısından hem OMNET++ hem de MİNİNET platformları kullanılmıştır. Değerlendirme sonuçları analiz edilirken simülasyon ve öykünme ortamları karşılaştırılmıştır. Hem OMNET++ hem de MİNİNET tabanlı testler, özellikle fonksiyonel analiz aşamasında benzer sonuçlar vermiştir. OMNET++ daha esnek ve tekrarlanabilir testler sağlamaktadır. Bu nedenle, genel olarak sistem tasarımında bu simülasyon aracının kullanılmasını önermek mümkündür. MİNİNET ise daha gerçekçi bir test ortamında daha detaylı performans analizi sonuçları sunmaktadır. Bundan dolayı, çalışma kanıtı aşamalarında bu aracın kullanılması önerilmektedir.

Eksiklikler ve ileride yapılabilecek çalışmalar incelendiğinde üç husus göze çarpmaktadır. İlk olarak, mevcut MiddleModule sisteminde, algılama algoritması parametrelerinin

değerleri manuel olarak atanır. Uygun şekilde atanmazlarsa, MiddleModule sisteminin doğruluğu azalır. Bu nedenle, bu tür sorunlardan kaçınmak için bu değişkenler, yani *saturationThreshold, windowSize* ve *maliciousThreshold,* gelecekteki çalışmalar olarak otomatik olarak atanmalıdır. Bu amaçla denetleyici üzerinde çalışan MiddleModule Orchestrator modülü, anahtarların algılama kararlarını izleyerek ve ağ topolojisini dikkate alarak bu parametreler için uygun değerler üretebilir. Ardından, bu değerler her bir veri düzlemi algılama düğümüne dağıtılabilir. Ek olarak, her uç anahtarı tarafından uygulanan kötü amaçlı liste güncelleme tekniği, denetleyiciden bir onay aldıktan sonra herhangi bir girişin bu listede saklanacağı şekilde değiştirilebilir. İkincisi, bu tez kapsamında sadece Ağ/Aktarım katmanında DDoS saldırıları ele alınmıştır. MiddleModule sistemi, gelecekteki çalışmalar açısından uygulama düzeyinde DDoS saldırılarının yanı sıra daha karmaşık saldırılara karşı bir SDN'yi savunacak şekilde geliştirilebilir. Hacimsel saldırılar, daha ayrıntılı algılama algoritmaları ile daha işbirlikçi bir yaklaşım kullanılarak veri düzlemi cihazlarında aranabilir. Bu amaçla başka bazı teknikler kullanılabilse de NFV kullanmak belirli avantajlar sağlayabilir. Ağ İşlevi Sanallaştırmanın düzenleyicisi, veri düzlemi algılama düğümleri arasında iş birliği sağlamak için kullanılabilir. Her veri düzlemi algılama düğümünün performansını kolayca analiz edebilir, bunları otomatik olarak etkinleştirebilir, devre dışı bırakabilir, parametrelerini değiştirebilir veya algılama listelerini hızlı bir şekilde güncelleyebilir. Ayrıca, gelişmiş algılama algoritmaları, merkezi ve güçlü bir cihaz tarafından sanal işlevler olarak üretilebilir ve gerekli algoritmalar, NFV'nın orkestratörü kullanılarak gerekli veri düzlemi cihazlarına koordineli olarak dağıtılabilir. Ayrıca, hacimsel olmayan saldırıları kapsamak için, bu çalışmada önerilen veri katmanı tabanlı yaklaşımla uyumlu olarak merkezi bir algılama yöntemi çalıştırılabilir. Son olarak, MiddleModule sistemi dağıtılmış algılama düğümlerinden yararlanmaktadır. Aslında, en iyi performansı elde etmek için algılama düğümleri, ana bilgisayarlara yakın yerleştirilecek şekilde maksimum performansta ağa dağıtılmalıdır. Bu durum, sistemin uygulama karmaşıklığını artırmaktadır. Gelecekteki yapılabilecek çalışmalara yönelik olarak işbirlikçi bir yaklaşım kullanmak, bu tür uygulama karmaşıklığını azaltmak için yararlı olabilir.

Gerçek zamanlı saldırılar açısından bu çalışmada, 5G çağının IoT ağlarında açıklanabilir güvenlik sağlamayı amaçlayan SDN'ler için otomatik, akıllı bir saldırı tespiti ve azaltma yaklaşımı sunulmaktadır. Bu yaklaşım, çeşitli saldırı sınıflarını tespit etmek ve veri katmanında yüksek önceliğe sahip yeni akış kurallarının yüklenmesi yoluyla düzeltici önlem almak için SDN uygulama katmanında rastgele bir orman sınıflandırıcısı tarafından ağ

akışlarının otomatikleştirilmiş akış özelliği çıkarımına ve son derece hassas bir şekilde sınıflandırılmıştır. SDN'lere özgü veri kümesi modellemesi kapsamında normal trafiğin yanı sıra yaygın ağ saldırıları için akış verilerini içeren gerçekçi bir Ne ortamı sunulmuş ve önerilen güvenliğin varlığı ile yokluğunda izinsiz giriş tespitinin doğruluğu ve performans sonuçları hakkında sonuçlar değerlendirilmiştir.

Gerçek zamanlı saldırıların önlenmesi sistemine yönelik güvenlik yaklaşımı, 5G ve sonrasında yaygın olarak kullanılacak olan SDN'ler tarafından yönetilen ağlardaki saldırıların gerçek zamanlı, yüksek doğrulukta tespitini ve hafifletilmesini sağlamayı vaat ediyor. Oluşturulan veri setinin, SDN'ler tarafından yönetilen IoT ağlarında makine öğrenimi tabanlı izinsiz giriş tespitinde daha fazla araştırma yapmak için de yararlı bir kaynak olacağına inanılmaktadır.

KAYNAKLAR

Ahmed, M. E., and Kim, H. (6-09 April 2017). *DDoS attack mitigation in Internet of Things using software defined networking.* In 2017 IEEE third international conference on big data computing service and applications (BigDataService). Redwood City, CA, USA, 271-276

Akyıldız, I. F., Wang, P., and Lin, S. C. (2015). SoftAir: A software defined networking architecture for 5G wireless systems. *Computer Networks*, 85, 1-18.

Alharbi, T., Layeghy, S., and Portmann, M. (November 2017*). Experimental evaluation of the impact of DoS attacks in SDN*. In 2017 27th International Telecommunication Networks and Applications Conference (ITNAC). Melbourne, Australia ,1-6.

Alqahtani, M., Mathkour, H., and Ben Ismail, M. M. (2020). IoT botnet attack detection based on optimized extreme gradient boosting and feature selection. *Sensors*, 20(21), 6336.

Amangele, P., Reed, M. J., Al-Naday, M., Thomos, N., and Nowak, M. (19-21 November 2019). *Hierarchical machine learning for IoT anomaly detection in SDN*. In 2019 International Conference on Information Technologies (InfoTech) ,NewYork , 1-4.

Ashraf, J., and Latif, S. (11 November 2014). *Handling intrusion and ddos attacks in software defined Networks Using Machine Learning Techniques*. In *2014* National software engineering conference.

Barakabitze, A. A., Ahmad, A., Mijumbi, R., and Hines, A. (2020). 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Computer Networks*, 167, 106984.

Besnah, Y. K. (2025). Multi-Stage Adversarial Defense for Online DDoS Attack detection System in 10T, Digital Object Identifier 10.1109/ACCESS.2025.3560186

Bhunia, S. S., and Gurusamy, M. (2017, 22-24 November). Dynamic attack detection and mitigation in IoT using SDN. In 2017 27th International telecommunication networks and applications conference (ITNAC). IEEE. 1-6.

Boite, J., Nardin, P. A., Rebecchi, F., Bouet, M., and Conan, V. (2017, 3-7 July). *Statesec: Stateful Monitoring For Ddos Protection İn Software Defined Networks*. In 2017 IEEE Conference on Network Softwarization (NetSoft), 1-9, IEEE.

Braga, R., Mota, E., and Passito, A. (2010, 11-14 October). *Lightweight Ddos Flooding Attack Detection -Using NOX/Openflow*. In IEEE Local Computer Network Conference, 408-415, IEEE.

Breiman, L. (2001). Random forests article, *Machine Learning*,45, 5–32.

Chang, R. K. (2002). Defending against flooding-based distributed denial-of-service attacks: A tutorial. *IEEE Communications Magazine*, 40(10), 42-51.

Chen, K. Y., Junuthula, A. R., Siddhrau, I. K., Xu, Y., Chao, H. J. (2016, 7-10 November). *SDNShield: Towards more comprehensive defense against DDoS attacks on SDN control plane*. In 2016 IEEE conference on communications and network security (CNS), 28-36.

Chen, X., and Yu, S. (2016). A collaborative intrusion detection system against DDoS for SDN. *IEICE TRANSACTIONS on Information and Systems*, 99(9), 2395-2399.

Chin, T., Mountrouidou, X., Li, X., Xiong, K. (2015). An SDN-Supported Collaborative Approach For Ddos Flooding Detection And Containment. In *MILCOM 2015-2015 IEEE Military Communications Conference*, 659-664, IEEE.

Derhab, A., Guerroumi, M., Gumaei, A., Maglaras, L., Ferrag, M. A., Mukherjee, M., and Khan, F. A. (2019). Blockchain and random subspace learning-based IDS for SDN-enabled industrial IoT security. *Sensors*, 19(14), 3119.

Dong, P., Du, X., Zhang, H., Xu, T. (2016). A Detection Method For A Novel Ddos Attack Against SDN Controllers By Vast New Low-Traffic Flows. In *2016 IEEE International Conference on Communications (ICC)*, 1-6, IEEE.

Evmorfos, S., Vlachodimitropoulos, G., Bakalos, N., Gelenbe, E. (2020). Neural network architectures for the detection of SYN flood attacks in IoT systems. In *Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments*,1-4.

Felderer, M., Büchler, M., Johns, M., Brucker, A. D., Breu, R., and Pretschner, A. (2016). Security testing: A survey. *Advances in Computers*, Volume 101, 1-51.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems?. *The Journal of Machine Learning Research*, 15(1), 3133-3181.

Galeano-Brajones, J., Carmona-Murillo, J., Valenzuela-Valdés, J. F., and Luna-Valero, F. (2020). Detection and mitigation of DoS and DDoS attacks in IoT-based stateful SDN: An experimental approach. *Sensors*, 20(3), 816.

Gökrem, L., ve Bozuklu, M. (2016). Nesnelerin interneti: yapılan çalışmalar ve ülkemizdeki mevcut durum. *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, 13, 47-68.

Handigol, N. Heller, B. Jeyakumar, V. Lantz, B., and McKeown, N. (2012). Reproducible network experiments using container-based emulation, Proc. 8th International Conferenceon emerging Networking EXperiments and Technologies (CoNEXT)'12.

Hırsı, A. (2025). Comprehensive Analysis of DDoS Anomaly Detection in Software-Defined Networks, *Digital Object Identifier 10.1109/ACCESS.2025.3535943*

Jantila, S., and Chaipah, K. (2016). A security analysis of a hybrid mechanism to defend ddos attacks in SDN. *Procedia Computer Science*, 86, 437-440.

Kalkan, K., Gür, G., ve Alagöz, F. (2017). SDNScore: A statistical defense mechanism against DDoS attacks in SDN environment. *IEEE symposium on computers and communications (ISCC)*, 669-675.

Kaushik, A. K., Pilli, E. S., and Joshi, R. C. (2010). Network forensic system for port scanning attack. In *2010 IEEE 2nd International Advance Computing Conference (IACC)*, 310-315).

Kim, J., Firoozjaei, M. D., Jeong, J. P., Kim, H., and Park, J. S. (2015, October). SDN-based security services using interface to network security functions. In *2015 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 526-529). IEEE.

Kokila, R. T., Selvi, S. T., and Govindarajan, K. (2014). DDoS detection and analysis in SDN-based environment using support vector machine classifier. In *2014 sixth international conference on advanced computing (ICoAC)* ,205-210. IEEE.

Koroniotis, N., Moustafa, N., Sitnikova, E., and Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100, 779-796.

Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S. and Uhlig, S. (2014). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1), 14-76.

Lantz, B., Heller, B., and McKeown, N. (2010). A Network İn A Laptop: Rapid Prototyping For Software-Defined Networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks* pp. 1-6.

Lim, S., Yang, S., Kim, Y., Yang, S., and Kim, H. (2015). Controller scheduling for continued SDN operation under ddos attacks. *Electronics Letters*, 51(16), 1259-1261.

Long, Q., Chen, Y., Zhang, H., and Lei, X. (2019). Software defined 5G and 6G networks: a survey. *Mobile networks and applications*, 27,issue 5, 1792-1812.

Machado, C. C., Granville, L. Z., and Schaeffer-Filho, A. (2016). Answer: Combining NFV and SDN features for network resilience strategies. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, 391-396, IEEE.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J. and Turner, J. (2008). OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM computer communication review*, 38(2), 69-74.

Moore, D., Shannon, C., Brown, D. J., Voelker, G. M., and Savage, S. (2006). Inferring internet denial-of-service activity. *ACM Transactions on Computer Systems (TOCS)*, 24(2), 115-139.

Mostafaei, H. and Menth, M. (2018). Software-defined wireless sensor networks: A survey. *Journal of Network and Computer Applications*, 119, 42-56.

Mousavi, S. M., and St-Hilaire, M. (2015). Early detection of ddos attacks against SDN controllers. In *2015 international conference on computing, networking and communications (ICNC)*.

Moustafa, N., and Slay, J. (2015). UNSW-NB15: A Comprehensive Data Set For Network İntrusion Detection Systems (UNSW-NB15 Network Data Set). In *2015 military communications and information systems conference (MilCIS)*,1-6. IEEE.

ONOS, (2023) Erişim Adresi: https://www.opennetworking.org/onos Son Erişim Tarihi: 01.01.2026

Open vSwitch (2023) Erişim Adresi: https://www.openvswitch.org Erişim Tarihi: 01.01.2026.

Owens, R., and Wang, W. (2011). Non-interactive OS fingerprinting through memory de-duplication technique in virtual machines. In *30th IEEE international performance computing and communications conference*, 1-8).

Phan, T. V., Bao, N. K., and Park, M. (2016). A Novel Hybrid Flow-Based Handler With Ddos Attacks İn Software-Defined Networking. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, 350-357, IEEE.

Rathore, S., and Park, J. H. (2018). Semi-supervised learning based distributed attack detection framework for IoT. *Applied Soft Computing*, 72, 79-89.

Ravi, N., and 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.

Rengaraju, P., Ramanan, V. R., and Lung, C. H. (2017). Detection And Prevention Of Dos Attacks İn Software-Defined Cloud Networks. In *2017 IEEE Conference on Dependable and Secure Computing*, 217-223, IEEE.

Ringberg, H. Roughan, M. and Rexford, J. (2008). The need for simulation in evaluating anomaly detectors, *ACM SIGCOMM Comput*. Commun. Volume 38, 1.

Scott-Hayward, S., O'Callaghan, G., Sezer, S. (2013). SDN security: A survey. In *2013 IEEE SDN For Future Networks and Services (SDN4FNS)*.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward Generating A New İntrusion Detection Dataset And İntrusion Traffic Characterization. *ICISSp*, *1*, 108-116.

Shin, S., Yegneswaran, V., Porras, P., and Gu, G. (2013). Avant-guard: Scalable and vigilant switch flow management in software-defined networks. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 413-424.

Shiravi, A., Shiravi, H., Tavallaee, M., and Ghorbani, A. A. (2012). Toward Developing A Systematic Approach To Generate Benchmark Datasets For İntrusion Detection. *Computers & Security*, 31(3), 357-374.

Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., and Sakurai, K. (2020). Machine learning-based IoT-botnet attack detection with sequential architecture. *Sensors*, 20(16), 4372.

Şimşek, G., Bostan, H., Sarıca, A. K., Sarıkaya, E., Keleş, A., Angin, P. ve Onur, E. (2020). Dropppp: a P4 approach to mitigating dos attacks in SDN. In *Information Security Applications: 20th International Conference, WISA 2019, Jeju Island, South Korea, August 21–24, 2019, Revised Selected Papers 20*, 55-66. Springer International Publishing.

Tavallaee, M. Bagheri, E. Lu, W and Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set, in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*.

Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. (2009). A Detailed Analysis Of The KDD CUP 99 Data Set. In *2009 IEEE Symposium On Computational İntelligence For Security And Defense Applications*,1-6. IEEE.

Wang, M., Zheng, K., Yang, Y., and Wang, X. (2020). An explainable machine learning framework for intrusion detection systems. *IEEE Access*, *8*, 73127-73141.

Wang, R., Jia, Z., and Ju, L. (2015). An Entropy-Based Distributed Ddos Detection Mechanism İn Software-Defined Networking. In *2015 IEEE Trustcom/BigDataSE/ISPA*, 310-317, IEEE.

Xu, Y., and Liu, Y. (2016). Ddos Attack Detection Under SDN Context. In *IEEE INFOCOM 2016-the 35th annual IEEE international conference on computer communications*, 1-9, IEEE.

Yan, Q., Yu, F. R., Gong, Q., and Li, J. (2015). Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE communications surveys & tutorials*, 18(1), 602-622.

Yuan, B., Zou, D., Yu, S., Jin, H., Qiang, W., and Shen, J. (2016). Defending Against Flow Table Overloading Attack İn Software-Defined Networks. *IEEE Transactions on Services Computing*, 12(2), 231-246.

Zargar, S. T., Joshi, J. and Tipper, D. (2013). A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. *IEEE communications surveys & tutorials*, 15(4), 2046-2069.

Zhang, L. (2025). EfficientSTNet: A Deep Learning Approach for Multi-Class DDoS Detection, *Digital Object Identifier 10.1109/ACCESS.2025.3606644.*



Gazili olmak ayrıcalıktır