
GAZİ ÜNİVERSİTESİ
Gazi
1926

PROGRAMLANABİLİR VERİ KATMANI YARDIMIYLA AĞ GÜVENLİK FONKSİYONLARININ GERÇEKLEŞTİRİLMESİ

Mehmet Emin ŞAHİN

DOKTORA TEZİ
BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

MART 2024

Mehmet Emin ŞAHİN tarafından hazırlanan "PROGRAMLANABİLİR VERİ KATMANI YARDIMIYLA AĞ GÜVENLİK FONKSİYONLARININ GERÇEKLEŞTİRİLMESİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilişim Sistemleri Ana Bilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. Mehmet DEMİRCİ

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……..

Başkan: Prof. Dr. Mahir DURSUN

Elektrik-Elektronik Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. …….

Üye: Prof. Dr. Ertan ONUR

Bilgisayar Mühendisliği Ana Bilim Dalı, Orta Doğu Teknik Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...…….……...

Üye: Doç. Dr. İbrahim Alper DOĞRU

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Üye: Dr. Öğr. Üyesi Serkan GÖNEN

Bilgisayar Mühendisliği Ana Bilim Dalı, İstanbul Gelişim Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum. ...……

Tez Savunma Tarihi: 29/03/2024

Jüri tarafından kabul edilen bu tezin Doktora Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Aslıhan TÜFEKCİ

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim 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.

Mehmet Emin ŞAHİN

29/03/2024

PROGRAMLANABİLİR VERİ KATMANI YARDIMIYLA AĞ GÜVENLİK FONKSİYONLARININ GERÇEKLEŞTİRİLMESİ

(Doktora Tezi)

Mehmet Emin ŞAHİN

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Mart 2024

ÖZET

Adres çözümleme amacıyla kullanılan DNS servisi onlarca yıldır internet altyapısının en temel yapı taşını oluşturmaktadır. DNS hizmeti veren altyapıların siber saldırılara karşı korunması hizmet kesintilerinin önlenmesi için büyük önem taşımaktadır. Bunun yanı sıra, SYN saldırı yöntemi, TCP bağlantısı kurulması esnasında kullanılan üçlü el sıkışma mekanizmasının sömürülmesi ile gerçekleştirilen ve yıllardır süregelen bir saldırı türüdür. Programlanabilir ağların ortaya çıkmasıyla birlikte ağ güvenliğine yönelik birçok yenilikçi yaklaşım ortaya çıkmıştır. Gerçekleştirilen bu çalışma kapsamında programlanabilir veri katmanlarında kullanılmak üzere alan-etkin çalışan, sabit sorgulama süresi sağlayan, yüksek doğruluklu, güncellenebilir bloom filtresi veri yapısı geliştirilerek önerilmiştir. Önerilen veri yapısı geleneksel Bloom filtresinin aksine, güncelliğini yitirmiş verilerin silinebilmesini ve zamansal veri güncellik kontrolü yapılabilmesini destekler. Önerilen veri yapısı bu çalışma kapsamında geliştirilen yeni güvenlik fonksiyonlarında ağ paketlerinin durumsal olarak takip edilmesinde kullanılmıştır. SYN saldırılarına karşı geliştirilen yöntem ile TCP bağlantı kurulumu arka uç sunucudan P4 anahtara aktarılmış ve bu sayede önerilen savunma fonksiyonunun herhangi bir ilave paket gönderimine sebep olmadan ve istemciler tarafından tespit edilmeden gerçekleştirilmesi sağlanmıştır. DNS sunucularına karşı yapılan yoğun DNS sorguları ve yükseltme saldırılarına karşı adaptif istek limitleme ve DNS güvenlik duvarı fonksiyonları gerçekleştirilmiştir. Aynı zamanda parçalanmış DNS yanıtlarının da durumsal takibi sağlanmıştır.

Bilim Kodu	:	92407
Anahtar Kelimeler	:	Programlanabilir Veri Katmanı, P4, Bloom Filtresi, SYN Savunma, DNS DDoS Savunma
Sayfa Adedi	:	101
Danışman	:	Doç. Dr. Mehmet DEMİRCİ

REALIZATION OF NETWORK SECURITY FUNCTIONS WITH THE HELP OF PROGRAMMABLE DATA PLANE

(Ph. D. Thesis)

Mehmet Emin ŞAHİN

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

March 2024

ABSTRACT

The DNS service, utilized for address resolution purposes, has served as a fundamental building block of internet infrastructure for decades. Safeguarding infrastructures that provide DNS services against cyber-attacks is crucial to prevent service interruptions. In addition, the SYN flood attack, a type of attack persisting for years, exploits the triple handshake mechanism during TCP connection establishment. With the advent of programmable networks, numerous innovative approaches to network security have emerged. In this study, a space-efficient, highly accurate, updatable bloom filter data structure has been developed and proposed for use in programmable data planes. Unlike traditional Bloom filters, the proposed data structure supports the deletion of outdated data and temporal data up-to-dateness control. Proposed data structure was employed to statefully track network packets for the new security functions developed within this study. As for the proposed SYN defense, TCP connection establishment is offloaded from the back-end server to the P4 switch, enabling the proposed defense function without causing additional packets or delay. This approach also makes proposed defense transparent to clients. Furthermore, adaptive DNS query limiting and DNS firewall functions are proposed against DNS query flood and DNS amplification attacks, where stateful tracking of fragmented DNS responses is also achieved.

Science Code	:	92407
Key Words	:	Programmable Data Plane, P4, Bloom Filter, SYN Defense, DNS DDoS Defense
Page Number	:	101
Supervisor	:	Assoc. Prof. Dr. Mehmet DEMİRCİ

TEŞEKKÜR

Akademik gelişimime büyük katkı sağlayarak doktora çalışmalarım süresince kıymetli görüş ve yardımlarıyla beni yönlendiren danışman hocam Sayın Doç. Dr. Mehmet DEMİRCİ'ye;

Doktora tez çalışmam süresince çok kıymetli değerlendirmelerde bulunarak tez çalışmamı yönlendiren tez izleme komitesi hocalarım Sayın Prof. Dr. Mahir DURSUN ve Sayın Dr. Öğr. Üyesi Serkan GÖNEN'e;

Tez savunma jürisinde yer alarak değerlendirmelerde bulunan ve görüşlerini paylaşarak tez çalışmama katkı sunan jüri üyesi hocalarım Sayın Prof. Dr. Ertan ONUR ve Sayın Doç. Dr. İbrahim Alper DOĞRU'ya;

Çalışanlarının gelişimine önem veren, doktora çalışmalarını teşvik eden ve bir çalışanı olmaktan mutluluk duyduğum çok değerli Türkiye Bilimsel ve Teknolojik Araştırma Kurumuna (TÜBİTAK);

Birlikte çalışmaktan her zaman keyif aldığım ve doktora eğitimim süresince her anlamda bilgi paylaşımında ve ortak değerlendirmelerde bulunduğum çalışma arkadaşlarım Koray YILMAZ, Harun BOLAT ve Oğuzhan AKYILDIZ'a;

Hayatım boyunca her anlamda desteklerini yanımda hissettiğim çok değerli anneme, azmi ve çalışkanlığı ile bana her zaman rol model olan rahmetli babama;

Doktora eğitimim süresince eş olarak verilebilecek azami desteği ve fedakârlığı sağlayan, her anlamda benim için alan yaratan, kıymetli yol arkadaşım Feride Hanım'a; sevgili çocuklarım Ömer Asaf ve Emir Talha'ya sonsuz teşekkürlerimi sunuyorum.

İÇİNDEKİLER

Sayfa

ÖZET iv

ABSTRACT v

TEŞEKKÜR vi

İÇİNDEKİLER vii

ÇİZELGELERİN LİSTESİ ix

ŞEKİLLERİN LİSTESİ x

SİMGELER VE KISALTMALAR xiii

1. GİRİŞ 1

2. PROGRAMLANABİLİR VERİ KATMANI İLE AĞ GÜVENLİĞİ 9

2.1. Programlanabilir Veri Katmanı ve P4 Programlama 9

2.2. SYN Saldırıları 13

2.2.1. SYN saldırılarına karşı P4 temelli savunma çalışmaları 14

2.3. DNS Servis Engelleme Saldırıları 16

2.3.1. DNS dağıtık servis engelleme saldırılarına karşı istek limitleme 17

2.3.2. DNS yükseltme saldırılarına karşı birebir eşleştirme temelli savunma 19

3. YÜKSEK DOĞRULUK SAĞLAYAN GÜNCELLENEBİLİR BLOOM FİLTRESİ TASARIMI 23

3.1. Bloom Filtresi Veri Yapısı 23

3.1.1. Programlanabilir veri katmanında Bloom filtresi kullanımı 27

3.2. Güncellenebilir Bloom Filtresi Yaklaşımı 29

3.2.1. Güncellenebilir Bloom filtresi için depolama gereksinimleri 34

4. SYN SALDIRILARINA KARŞI P4 TEMELLİ SAVUNMA 37

4.1. Bağlantı Havuzunun Programlanabilir Veri Katmanında Gerçekleştirilmesi 39

Sayfa

4.2. SYN Saldırılarına Karşı GBF Temelli Savunmanın Gerçekleştirilmesi 44

4.3. SYN Saldırılarına Karşı Savunma Performansının Değerlendirilmesi 46

4.3.1. Güncellenebilir Bloom filtresinin performansının değerlendirilmesi..... 47

4.3.2. Bağlantı havuzunun performansının değerlendirilmesi..... 52

4.3.3. SYN saldırılarına karşı savunma performansının değerlendirilmesi..... 54

5. DNS SALDIRILARINA KARŞI P4 TABANLI İSTEK LİMİTLEME UYGULAMASI VE GÜVENLİK DUVARI GELİŞTİRME 57

5.1.Aktif İstek Limitlemenin Güncellenebilir Bloom Filtresi ile Gerçekleştirilmesi 59

5.2. Saldırganlar için Zaman Bazlı Cezalandırma Tasarımı..... 64

5.3. DNS Güvenlik Duvarının Güncellenebilir Bloom Filtresi ile Gerçekleştirilmesi... 66

5.4. Parçalanmış DNS Yanıtlarının Durumsal Takibi..... 67

5.5. Önerilen DNS Fonksiyonlarının Performansının Değerlendirilmesi 69

5.5.1. DNS istek limitleme parametrelerinin etkisinin incelenmesi..... 70

5.5.2. Yoğun DNS sorgularına karşı istek limitleme performansı..... 77

5.5.3. DNS güvenlik duvarı performansının incelenmesi..... 79

6. SONUÇ VE ÖNERİLER..... 85

KAYNAKLAR..... 91

ÖZGEÇMİŞ..... 101

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge 2.1. SYN saldırılarına karşı programlanabilir veri katmanı kullanan çalışmalar 15

Çizelge 2.2. Birebir eşleştirme temelli çalışmalar 20

Çizelge 3.1. Programlanabilir veri katmanında Bloom filtresi kullanımı..... 28

Çizelge 4.1. Test ortamında tahsis edilen kaynaklar..... 47

Çizelge 4.2. TCP bağlantı takibinde GBF' nin karşılaştırılması 48

Çizelge 5.1. Test ortamında tahsis edilen kaynaklar..... 69

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Yazılım tanımlı ağ mimarisi 9

Şekil 2.2. Geleneksel mimariden programlanabilir veri katmanı ile SDN mimarisine geçiş 11

Şekil 2.3. P4 genel mimari modeli (PISA, Protocol Independent Switching Architecture) 12

Şekil 2.4. SYN saldırısı 13

Şekil 2.5. DNS servis engelleme saldırılarının görselleştirilmesi 16

Şekil 3.1. Örnek bir BF veri yapısı üzerinden veri ekleme ve sorgulamanın gösterimi . 24

Şekil 3.2. Güncellenebilir Bloom Filtresi Veri Yapısı 30

Şekil 3.3. Örnek 5 adet bağlantı isteği 30

Şekil 3.4. Önerilen güncellenebilir Bloom filtresi veri yapısı ve çalışmasının gösterimi 31

Şekil 3.5. Durum-1 için veri ekleme ve silme akış diyagramı 33

Şekil 3.6. Durum-2 için veri silme akış diyagramı 34

Şekil 4.1. Çerez temelli SYN vekili ve SYN doğrulama yaklaşımlarının görselleştirilmesi 38

Şekil 4.2. Önerilen GBF ve bağlantı havuzu temelli SYN savunma 38

Şekil 4.3. Programlanabilir veri katmanında bağlantı havuzu 40

Şekil 4.4. P4 yazmaçlarda oluşturulan bağlantı havuzu veri yapısı 40

Şekil 4.5. Bağlantı havuzunun hazırlanması 42

Şekil 4.6. Bağlantı havuzunun kullanılması 43

Şekil 4.7. SYN saldırılarına karşı bağlantı havuzu ve GBF temelli savunma 45

Şekil 4.8. SYN savunma test topolojisi 46

Şekil 4.9. GBF veri yapısının TCP bağlantı takibinde doğruluğu 49

Şekil 4.10. Çeşitli BF dizi genişlik değerleri için SYN takip doğruluğu 50

Şekil **Sayfa**

Şekil 4.11. Artırılmış devir süresinin 1Mpps için görselleştirilmesi 50

Şekil 4.12. Çeşitli boyutlarda GBF veri yapısının SYN takibinde devir başına doğruluğu..... 51

Şekil 4.13. Çeşitli boyutlarda GBF veri yapısının saniye başına paket oranları için doğruluğu 52

Şekil 4.14. Saniyede 1000 istek için sunucu ve istemci paket oranları 53

Şekil 4.15. SYN savunma için artan saldırı yoğunluğu ile bağlantı olasılığının değişimi 54

Şekil 5.1. DNS istek limitlemesi için değiştirilmiş jeton kovası algoritması 59

Şekil 5.2. Güncellenebilir Bloom filtresi ile DNS paketlerinin durumsal takibi 60

Şekil 5.3. Örnek eşzamanlı DNS sorgusu limitleme (esl = 3) 61

Şekil 5.4. Aktif istek limitleme için DNS sorgularının işlenmesi 63

Şekil 5.5. Aktif istek limitleme için DNS yanıt paketlerinin işlenmesi..... 63

Şekil 5.6. Güncellenebilir Bloom filtresinde ceza süresinin gerçeklenmesi..... 64

Şekil 5.7. Zaman temelli cezalandırma akış diyagramı 65

Şekil 5.8. DNS güvenlik duvarı durumsal sorgu takibi 66

Şekil 5.9. Parçalanmış DNS yanıtlarının durumsal takibi 68

Şekil 5.10. DNS istek limitleme test topolojisi..... 70

Şekil 5.11. Test ortamı DNS sorgusu yük testi..... 70

Şekil 5.12. Yetkili ad sunucusunda tek istemci için çeşitli esl değerlerinde DNS sorgusu limitleme..... 71

Şekil 5.13. Yetkili ad sunucusunda tek istemci için ortalama sorgu limiti değişimi 72

Şekil 5.14. Özyinelemeli ad sunucusunda tek istemci için çeşitli esl değerlerinde DNS sorgusu limitleme..... 73

Şekil 5.15. Özyinelemeli ad sunucusunda tek istemci için ortalama sorgu limiti değişimi..... 73

Şekil **Sayfa**

Şekil 5.16. Yetkili ad sunucusunda toplam eşzamanlı DNS sorgusu istek limitleme 74

Şekil 5.17. Yetkili ad sunucusunda toplam sorgu limiti ortalamasının değişimi..... 75

Şekil 5.18. Özyinelemeli ad sunucusunda çeşitli m değerlerinde toplam DNS sorgusu limitleme 75

Şekil 5.19. Özyinelemeli ad sunucusunda toplam sorgu limiti ortalamasının değişimi . 76

Şekil 5.20. Sorgular arası süre kullanılarak artan sorgu yoğunluğunun cezalandırılması..... 77

Şekil 5.21. Yetkili ad sunucuları için istek limitleme performansı..... 78

Şekil 5.22. Özyinelemeli ad sunucuları için istek limitleme performansı 79

Şekil 5.23. DNS Güvenlik Duvarı Test Topolojisi 80

Şekil 5.24. DNS yansıtma saldırısında durumsal takip performansı 81

Şekil 5.25. DNS yükseltme saldırısında durumsal takip performansı 82

Şekil 5.26. Çeşitli DNS yükseltme saldırısı boyutları için GBF doğruluk oranının değişimi..... 83

SİMGELER VE KISALTMALAR

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

Simgeler	**Açıklamalar**
μ_{ys}	Ortalama yanıt süresi
σ_{ys}	Yanıt süresinin varyansı
ms	Mili saniye
µs	Mikro saniye
m	Bloom filtresi dizi uzunluğu
k	Bloom filtresi özetleme fonksiyonu sayısı
esl	Eşzamanlı sorgu limiti

Kısaltmalar	**Açıklamalar**
ACK	Acknowledgement (Onay)
ANS	Authoritative Name Server (Yetkili Ad Sunucusu)
BF	Bloom Filter (Bloom Filtresi)
CBF	Counting Bloom Filter (Sayan Bloom Filtresi)
DDoS	Distributed Denial of Service (Dağıtık Servis Engelleme)
DNS	Domain Name Service (Alan Adı Hizmeti)
DoS	Denial of Service (Servis Engelleme)
EDNS0	Extension Mechanisms for DNS (DNS için Eklenti Mekanizmaları)
GBF	Güncellenebilir Bloom Filtresi
HTTP	Hypertext Transfer Protocol (Üst Metin Aktarım Protokolü)
INT	In-Band Network Telemetry (Bant içi Ağ Telemetrisi)
ISS	Internet Servis Sağlayıcı
LPM	Longest Prefix Matching (En Uzun Önek Eşleşmesi)

PISA	Protokolden Bağımsız Anahtar Mimarisi (Protocol-Independent Switch Architecture)
PTP	Precision Time Protocol (Hassas Zaman Protokolü)
RNS	Recursive Name Server (Özyinelemeli Ad Sunucusu)
RRL	Response Rate Limiting (Yanıt Hızı Limitleme)
SDN	Software Defined Networking (Yazılım Tanımlı Ağ)
SRAM	Static Random Access Memory (Statik Rasgele Erişim Belleği)
SYN	Synchronize (Senkronize)
TBF	Ternary Bloom Filter (Üçlü Bloom Filtresi)
TCAM	Ternary Content-Addressable Memory (Üçlü Adreslenebilir İçerik Tabanlı Bellek)
TCP	Transmission Control Protocol (İletim Denetimi Protokolü)

1. GİRİŞ

P4 temelli programlanabilir veri katmanı, yazılım tanımlı ağ *(Software Defined Networking, SDN)* yaklaşımında yeni bir paradigmayı temsil eder ve hat hızında paket işleme, protokol bağımsız paket anahtarlama ve esneklik gibi çeşitli avantajlar sunar. Programlanabilir veri katmanının ortaya çıkması, öncesinde yalnızca çip üreticilerinin sahip olduğu ağ tasarımına yönelik fonksiyonların geliştirilmesindeki sınırları genişletmiş ve bu alanda yeniliklerin yapılabilmesinin önündeki bariyerleri ortadan kaldırmıştır [1]. Son yıllarda programlanabilir ağların ortaya çıkmasıyla birlikte ağ güvenliğine yönelik birçok yenilikçi yaklaşım ortaya çıkmıştır. Literatürdeki güncel çalışmalar kategorik olarak ele alınırsa, bunlar güvenlik duvarı yaklaşımları [2-4], dağıtık servis engelleme saldırılarına karşı savunma yaklaşımları [5-8], bağlantı güvenliği yaklaşımları [9-11], saldırı tespit ve derin paket inceleme yaklaşımları [12-14] olarak ifade edilebilir. Yazılım tanımlı ağ mimarisinde, herhangi bir saldırı esnasında kontrolcü ile veri katmanı arasındaki iletişimin yükü artabilir ve bu durum performans ve iletişim darboğazlarının oluşmasına neden olabilir [15]. P4 programlama sayesinde veri katmanı daha akıllı hale gelir ve bazı ağ fonksiyonlarını üstlenme yeteneği kazanır.

TCP/IP protokolünün geçmişine bakıldığı zaman temellerinin 1970'li yıllara dayandığı görülür ve günümüzde internete bağlı hemen hemen tüm uç sistemlerin bu protokolü kullandığını söylemek mümkündür. İki uç sistem arasında güvenli bir TCP (*Transmission Control Protocol*) bağlantısı kurulabilmesi amacıyla üçlü el sıkışma olarak adlandırılan bağlantı kurulum mekanizması kullanılır. Bu mekanizma bağlantı kurulumu için oldukça iyi çalışır fakat bağlantı kurulumu esnasında istemci doğrulamasına yönelik bir kontrol fonksiyonu içermemesi sebebiyle SYN (*Synchronize*) saldırılarına karşı savunmasızdır. Bağlantı kurulumu esnasında sunucu aldığı bir SYN bağlantı isteği sonucunda bu istek için bir port tahsisi gerçekleştirerek protokol tarafından belirlenen ön tanımlı zaman aşım süresince ilgili portu bu bağlantı için tahsis eder. Bu davranıştan hareketle, saldırı amacıyla gönderilen yüksek oranda SYN isteği neticesinde sunucu üzerinde bulunan bütün portlar tahsis edilerek devamında gelen istekler için sunucu yanıt veremez duruma gelir. SYN saldırıları bilgisayar ağlarında bilinen bir hizmet engelleme saldırısı türüdür. Bu saldırı türü aynı zamanda yarı-açık bağlantı saldırısı olarak da isimlendirilebilir [16] .Yakın zamanda yayınlanan bir raporda 2023 yılının son çeyreğinde yapılan dağıtık servis engelleme

saldırıları içerisinde SYN saldırıları tüm saldırıların %19'unu oluşturarak ikinci sırada yer aldığı ifade edilmiştir [17].

SYN saldırıları aslında TCP protokolünün yapısına istemci doğrulamasına yönelik bir fonksiyon eklenerek engellenebilir. Fakat günümüzde TCP protokolünü kullanan milyarlarca uç sistem olduğu göz önünde bulundurulduğunda artık standart hale gelen protokol yapısında değişiklikler yapmak neredeyse imkansızdır. Bu nedenle SYN saldırısı yöntemi yıllardır süregelen ve literatürde her zaman aktif bir çalışma alanı olarak karşımıza çıkan bir saldırı yöntemidir. Yeni teknolojiler ortaya çıktıkça, araştırmacılar tarafından çok çeşitli tespit, hafifletme ve önleme yaklaşımları geliştirilmekte ve önerilmektedir.

Adres çözümleme amacıyla kullanılan alan adı hizmeti (*Domain Name Service, DNS*) ise onlarca yıldır internet altyapısının en temel yapı taşını oluşturmaktadır. DNS hizmeti veren altyapıların siber saldırılara karşı korunması hizmet kesintilerinin önlenmesi açısından büyük önem taşımaktadır. Nitekim önde gelen bir DNS servis sağlayıcısı olarak hizmet veren Dyn şirketi, 2016 yılında son yılların en geniş ölçekli DNS saldırılarından biri olarak kaydedilen Mirai Botnet saldırısına maruz kalmış ve bu durum Twitter, Amazon, Github ve Netflix gibi yüksek trafik talebine sahip popüler çevrimiçi hizmetlerin birkaç saat süren bir siber saldırının kurbanı olmasına neden olmuştur. Gerçekleşen bu siber saldırıda 107 000 IP adresinin kullanıldığı sonrasında yapılan araştırmalarda ortaya çıkmıştır [18]. Benzer şekilde popüler bir bulut hizmet sağlayıcı olan Cloudflare şirketi 2021 yılının sonlarında, Mirai varyantlı bir botnet ağından gerçekleştirilen DNS sunucularına gerçekleştirilen dağıtık servis engelleme *(Distributed Denial of Service, DDoS)* saldırısına maruz kaldığını ve bu saldırının yoğun DNS sorguları ve DNS yükseltme ataklarının birleşiminden oluşarak 2 Tbps hacmine ulaştığını raporlamıştır [19]. Ayrıca, 2023'ün 4. çeyreği için hazırlanan bir DDoS tehdit raporunda, yoğun DNS sorgusu ve DNS yükseltme saldırılarının kaydedilen tüm saldırıların yaklaşık %53 'ünü oluşturarak üst üste üçüncü çeyrekte en yaygın saldırı kategorisi olmaya devam ettiği raporlanmıştır [17,20] .

DNS hizmetini dağıtık hizmet engelleme saldırılarına açık hale getiren iki temel faktörden söz edilebilir. Bunlardan ilki, DNS tarafından veri iletim katmanı protokolü olarak durumsuz çalışan UDP *(User Datagram Protocol)* kullanılmasıdır. TCP protokolünde istemci ve sunucu arasında bağlantı kurulumu ve sonlandırmasında üçlü el sıkışma mekanizmasının kullanımı aynı zamanda sahte IP adresleri kullanımının önüne geçer. Nitekim sahte IP adresi

kullanılarak yapılan bir bağlantı kurulum isteği sunucuya ulaşsa dahi üçlü el sıkışma tamamlanamayacağı için bağlantı kurulumu tamamlanamaz. Fakat bezer şekilde bağlantı kurulumu veya sonlandırma aşamaları UDP protokolü için söz konusu değildir. Bunun neticesinde UDP protokolü kullanılarak yapılan DNS sorgularında istemcilerin IP adresleri için herhangi bir kontrol veya doğrulama yapılmaz. Bu durum DNS sorgularının sahte IP adresleri kullanılarak yapılabilmesine olanak tanır. Ayrıca küresel internet ağı düşünüldüğünde, sahte IP adreslerinin kullanılmasının önlenmesi amacıyla BCP38 standardı oluşturulmuştur [21]. Fakat bu standardın internet servis sağlayıcıları tarafından yaygın olarak benimsenmemesi sebebiyle sahte IP adresleri internet ağı üzerinde engellenmemektedir [22]. İkinci faktör ise, birçok DNS sunucusunun internet ağı içerisinde farklı konumlardaki istemcilerden herhangi bir kısıtlama olmadan sorgulama yapılmasına olanak tanıyan açık özyinelemeli DNS (*Open Recursive DNS*) servisi olarak hizmet vermesidir. Bu durum, açık özyinelemeli DNS sunucularını, bunları yükseltme ve yansıma saldırılarında kullanan saldırganlar için cazip bir araç haline getirmektedir.

Yoğun DNS sorgusu saldırıları, bir DNS sunucusunun kaynaklarına ve bant genişliğine odaklanan, sunucunun kapasitesini aşmayı ve yüksek yoğunluklu istekler altında sunucuyu yanıt veremez hale getirmeyi amaçlayan bir servis engelleme saldırısı türüdür. Öte yandan, DNS yükseltme saldırılarında, yükseltilmiş DNS yanıtları oluşturulur ve bu yanıtların yansıtılması için açık özyinelemeli DNS sunucuları araç olarak kullanılır. Herhangi bir DNS yanıtının boyutu ve buna karşılık gelen sorgunun boyutu arasındaki oran, yükseltme faktörü olarak bilinir ve bu oran 4670 gibi yüksek bir değere ulaşabilir [23]. Saldırı, hedef sistemin IP adresinin taklit edilmesi yoluyla gerçekleştirilir ve saldırı sonucunda oluşturulan DNS yanıtları ilgili sorguların boyutunu önemli ölçüde aşar. Her iki saldırı türünde de saldırganlar tarafından dağıtık ve kapsamlı bir saldırı gerçekleştirmek ve servis kesintisi oluşturmak amacıyla, kontrol edilen ve yüksek oranda istemci barındıran botnet ağları üzerinden saldırılar gerçekleştirir.

Araştırmanın amacı

Gerçekleştirilen bu tez çalışması kapsamında programlanabilir veri katmanında akışların durumsal takibinde kullanılmak üzere Bloom filtresi (*Bloom Filter, BF*) veri yapısının doğruluğunun düşmesi sebebiyle yeni bir BF temeli veri yapısı oluşturulması amaçlanmış ve bu veri yapısının gerçeklenerek önerilecek güvenlik fonksiyonlarında kullanılması

hedeflenmiştir. Gerçekleştirilen çalışmada BF veri yapısının doğruluğunun düşmesinin esas sebepleri irdelenmiş ve programlanabilir veri katmanlarında kullanılmak üzere yüksek doğruluklu yeni bir veri yapısının önerilmesi amaçlanmıştır.

Gerçekleştirilecek ağ güvenlik fonksiyonları için ilk olarak SYN saldırılarına karşı güncel çalışmalar incelenerek aşağıdaki soruların yanıtlanması ve bu doğrultuda yeni yöntemler geliştirilerek önerilmesi amaçlanmıştır:

1. TCP bağlantı kurulumu aşamasındaki üçlü el sıkışma mekanizmasının çalışması bir görev olarak değerlendirilirse bu görev arka uç sunucudan programlanabilir veri katmanına aktarılabilir mi?

2. TCP bağlantı kurulumu sırasındaki isteklerin durumsal takibi programlanabilir veri katmanında etkin ve yüksek doğrulukla başarılabilir mi?

3. Bağlantı takibinde BF veri yapısından veriler silinemez olarak kullanmak yerine veri yapısını güncellenebilir hale getirmek mümkün müdür?

4. Veri katmanında bağlantı havuzunu kullanılarak SYN doğrulaması sonrasında ek paket gönderimi ihtiyacı ortadan kaldırılabilir mi?

Devamında DNS saldırılarına karşı kullanılan DNS yanıt hızı limitleme *(Response Rate Limiting, RRL)* yöntemi ile ilgili olarak aşağıdaki soruların yanıtlanmasına yönelik araştırmalar gerçekleştirilerek önerilen yöntemler geliştirilmiştir.

5. Yoğun DNS sorgusu saldırıları hedef DNS sunuculara ulaşmadan önce proaktif olarak ağ katmanında engellemek için programlanabilir veri katmanı üzerinde DNS istek limitleme gerçekleştirilebilir mi?

6. Hem yetkili ad sunucular (*Authoritative Name Server, ANS*), hem de özyinelemeli ad sunucularına (*Recursive Name Server, RNS*) uygun ortak bir DNS istek limitleme yaklaşımını oraya koymak mümkün müdür?

7. Dağıtık servis engelleme saldırısının ölçeğine bakılmaksızın, programlanabilir veri katmanında alan-etkin bir istek limitleme algoritması uygulamak için önerilen güncellenebilir Bloom filtresi veri yapısı kullanılabilir mi?

Ayrıca DNS yükseltme saldırılarına karşı birebir eşleştirme yaklaşımı göz önünde bulundurulduğunda aşağıdaki soruların yanıtları araştırılarak yeni bir DNS güvenlik duvarı yaklaşımının programlanabilir veri katmanında geliştirilmesi amaçlanmıştır.

8. DNS paketlerinin durumsal takibinde sadece IP ve port numaralarını kullanmak yerine uygulama seviyesinde başlık alanlarından DNS işlem numarası parametresi de kullanılabilir mi?

9. Parçalanmış DNS yanıtları durumsal takip yaklaşımına dahil edilerek saldırı paketlerinin tespiti ve engellenmesi sağlanabilir mi?

10. DNS paketlerinin durumsal takibinde yüksek doğrulukla ve alan-etkin bir algoritma geliştirebilmek için güncellenebilir Bloom filtresi veri yapısı kullanılabilir mi?

Tezin katkıları

Öncelikle bu tez çalışması kapsamında programlanabilir veri katmanlarında akışların takibinde kullanılmak üzere yeni bir yaklaşım olarak, yüksek doğruluklu olarak çalışan, zamanla doğruluğu düşmeyen, zamana dayalı veri güncellik kontrolü yapabilen, bu amaçla zaman damgası kullanan, Güncellenebilir Bloom Filtresi (GBF) olarak isimlendirilen yeni bir veri yapısı modeli geliştirilerek önerilmiştir. Önerilen GBF veri yapısı çeşitli ağ güvenlik fonksiyonlarında kullanılarak etkinliği incelenmiştir.

GBF veri yapısı ve bağlantı havuzu kullanılarak gerçekleştirilen SYN savunma yaklaşımının katkıları şu şekilde listelenebilir:

- Önerilen yaklaşımda herhangi bir protokol ihlali, bağlantı yeniden kurulma isteği veya paket düşürme gibi bir durum söz konusu olmadığından istemciler tarafından önerilen savunma yönteminin tespit edilmesi söz konusu değildir.

- Doğrulanmış bir IP adresinin izin verilenler listesine eklenmesi gibi bir durumda söz konusu değildir. Belirli bir IP adresinden yapılan her bir istek eşsiz olarak değerlendirilir ve ayrı ayrı doğrulanır.

- Kullanılan GBF veri yapısı sayesinde düşük depolama alanı kullanılarak yüksek oranda SYN paketleri için zaman bazlı güncellik kontrolü yapılarak SYN isteklerinin doğrulanması sağlanmıştır.

- Kullanılan GBF veri yapısı olasılıksal bir veri yapısı olmasına karşın hedef seti kapsama *(target-set coverage)* saldırılarına karşı tasarımı gereği güçlüdür.

Ayrıca önerilen DNS istek limitleme ve DNS güvenlik duvarı yaklaşımları için katkılar şu şekilde listelenebilir.

- DNS servis engelleme saldırılarına karşı programlanabilir veri katmanı üzerinde DNS istek limitleme ve DNS güvenlik duvarı olmak üzere iki yeni güvenlik fonksiyonu geliştirilerek önerilmiştir. Her iki yaklaşımda bu tez kapsamında geliştirilen GBF veri yapısını kullanarak programlanabilir veri katmanında yüksek doğrulukla, depolama alanı açısından verimli ve uygulanabilir algoritmaların geliştirilmesi sağlamıştır.

- Önerilen DNS güvenlik duvarı, DNS protokolünün birebir eşleştirme davranışından yararlanılarak programlanabilir veri katmanında uygulama seviyesinde gerçekleştirilmiştir.

- Parçalanmış DNS yanıt paketlerinin durumsal takibe dahil edilmesine yönelik yeni bir yöntem önerilerek geliştirilmiştir. Bunun neticesinde DNS yükseltme saldırılarının bir parçası olmayan yalnızca meşru DNS yanıt paketlerinin iletilmesi sağlanarak saldırı paketleri yüksek doğrulukla engellenmiştir.

- DNS istek limitleme yaklaşımında zamana dayalı istek limitleme yaklaşımı yerine, istemci başına eşzamanlı ve aktif istekler takip edilerek DNS sunucusunun henüz yanıt vermediği isteklerin sayısı üzerinden limitleme gerçekleştirilmiştir. Bu yaklaşım önerilen yöntemin aktif istek oranına adaptif çalışmasını sağlayarak hem

yetkili ad sunucularında hem de özyinelemeli ad sunucularında kullanılabilmesini mümkün hale getirmiştir.

- Önerilen adaptif istek limitleme yaklaşımı aynı zamanda yoğun DNS sorguları sebebiyle DNS sunucunun aşırı kaynak tüketimini ve buna bağlı olarak yanıt veremez duruma gelmesini engellerken sunucu kaynaklarının istemciler arasında daha adil dağılımını sağlamaktadır.

Tezin organizasyonu

Gerçekleştirilen bu tez çalışması giriş bölümü dahil 6 bölümden oluşmaktadır. Tezin giriş bölümünde bu tez kapsamında araştırmaya konu problemlerin tespiti, araştırmanın amacı ve motivasyonu ile beraber bu tezin sunduğu katkılar sunulmuştur.

Tezin ikinci bölümünde bu tez çalışmasında kullanılan programlanabilir veri katmanı ve P4 programlama dili ile ilgili temel bilgilere yer verilmiştir. Bunun yanı sıra SYN saldırısının yapısı ve bu saldırılara karşı literatürdeki güncel programlanabilir veri katmanı temelli savunma yaklaşımlarına yer verilmiştir. Devamında DNS servis engelleme saldırılarından yoğun DNS sorgusu ve DNS yükseltme saldırılarının yapısı ile beraber bu saldırılara karşı literatürdeki güncel savunma yaklaşımları ele alınmıştır.

Tezin üçüncü bölümünde önerilen güncellenebilir Bloom filtresi veri yapısının tasarımı yer almaktadır. İlk olarak Bloom filtresi veri yapısının temel bileşenleri ve özelliklerine yer verilmiş ve farklı amaçlar için literatürde önerilen çeşitli Bloom filtresi yapılarının çalışma yapıları ve sağladığı üstünlüklere yer verilmiştir. Devamında literatürde programlanabilir veri katmanında Bloom filtresi kullanan farklı çalışmalar incelenerek detaylı olarak ele alınmıştır. Devamında güncellenebilir Bloom filtresi veri yapısı detaylı olarak anlatılmış ve önerilen modelin depolama gereksinimlerinin nasıl belirlendiğine yer verilmiştir.

Tezin dördüncü bölümünde SYN saldırılarına karşı önerilen GBF ve bağlantı havuzu temelli savunma yaklaşımın gerçekleştirilmesindeki tasarımsal detaylara ve gerçekleştirilmesinde kullanılan algoritma detaylarına yer verilmiştir. Devamında önerilen GBF veri yapısının ve bağlantı havuzunun performansı ayrı ayrı değerlendirilmiş ve son olarak SYN saldırılarına karşı birlikte kullanılmasındaki performansı irdelenmiştir.

Tezin beşinci bölümünde DNS servisine yapılan hizmet engelleme saldırılarına karşı önerilen DNS istek limitleme ve DNS güvenlik duvarı fonksiyonlarının gerçekleştirilmesindeki tasarımsal detaylar ayrıntılı olarak ele alınmıştır. Devamında GBF veri yapısı kullanılarak tespit edilen saldırganlar için önerilen zaman bazlı cezalandırma yaklaşımının tasarımı, sorgular arası zamanın GBF veri yapısıyla ölçülebilmesindeki aşamalar ve parçalanmış DNS yanıtlarının durumsal takibinin gerçekleştirilmesindeki aşamalar detaylı olarak anlatılmıştır. Sonrasında bu fonksiyonların performansının değerlendirilmesi amacıyla öncelikle DNS istek limitleme yaklaşımındaki çeşitli parametrelerin performansa olan etkisi ayrı ayrı detaylı olarak incelenmiştir. Önerilen DNS güvenlik duvarı yaklaşımının performansı IP parçalanmaya uğrayan ve uğramayan DNS yanıtları için ayrı ayrı değerlendirilmiştir. Son olarak önerilen GBF veri yapısının yükseltilmiş DNS yanıtları için sağladığı doğruluğun değişimine yer verilmiştir.

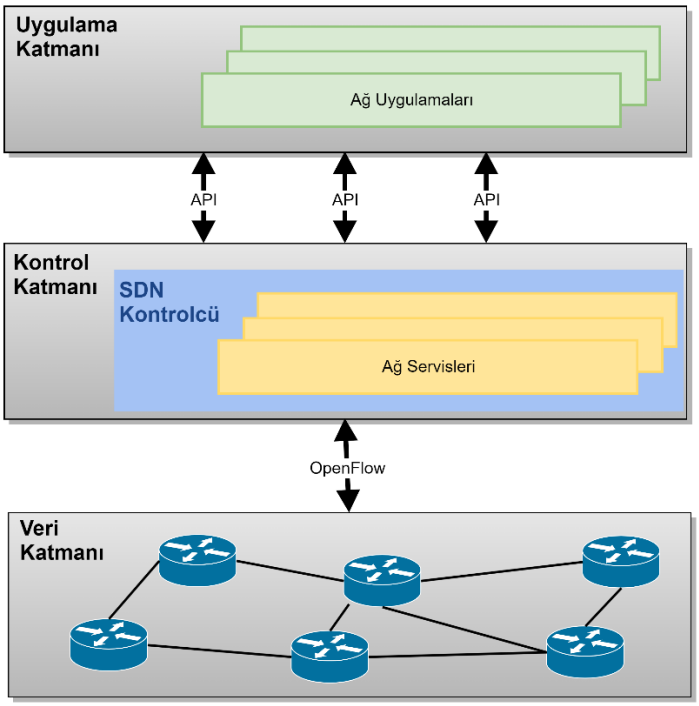
Tezin son bölümü olan altıncı bölüm sonuç ve öneriler bölümüdür. Bu bölümde tezin amacı ve gerçekleştirilen yaklaşımlar özetlenmiş ve tezin sağlamış olduğu katkılara özet olarak yer verilmiştir. Ayrıca gelecekte yapılabilecek çalışmalara yön göstermesi amacıyla yapılabilecek çalışmalara değinilmiştir.

2. PROGRAMLANABİLİR VERİ KATMANI İLE AĞ GÜVENLİĞİ

Bu bölümde programlanabilir veri katmanının yapısı ve bu tez kapsamında geliştirilen çeşitli ağ güvenliği fonksiyonları için güncel literatür çalışmaları yer almaktadır.

2.1. Programlanabilir Veri Katmanı ve P4 Programlama

Geleneksel ağ donanımları ve mimarileri, değişen ve gelişen ihtiyaçların karşılanmasında yetersiz kaldığından ağ teknolojilerinin daha etkin, esnek, hızlı, güvenli, yönetilebilir ve programlanabilir yapılara dönüştürülebilmesi amacıyla son dönemde ağ teknolojilerinde yeni yaklaşımlarla yoğun çalışmalar yürütülmekte ve bu yaklaşımları kullanan teknolojiler hızla hayata geçirilmektedir. Yazılım tanımlı ağ yaklaşımının bunların başında geldiği söylenebilir.



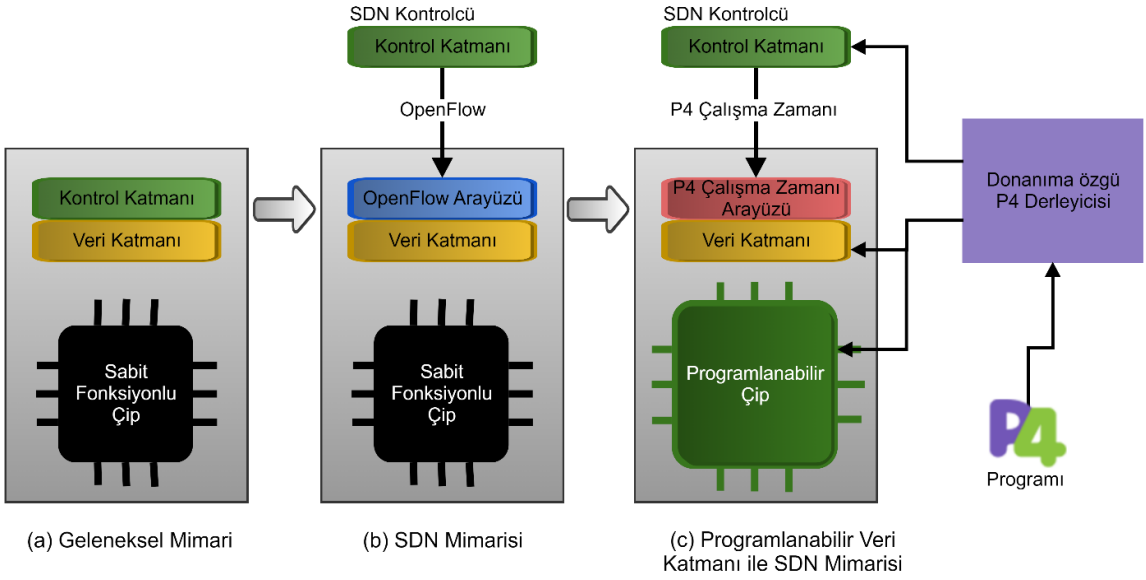
Şekil 2.1. Yazılım tanımlı ağ mimarisi

Yazılım tanımlı ağ mimarisinde veri katmanı ve kontrol katmanı Şekil 2.1' de gösterildiği gibi ayrıştırılmıştır. Ay yönetimi mantıksal olarak merkezileştirilerek esas paket anahtarlamasını gerçekleştiren veri katmanı ağ uygulamalarından soyutlanmıştır [47]. Bu sayede donanım üreticilerinden bağımsız olarak ağ yönetiminin sunulan arayüzler

vasıtasıyla merkezileştirilmesi sağlanır. Ayrıca ağ yönetiminde otomasyon uygulamalarının gerçekleştirilebilmesine ve değişen ihtiyaçlara kolayca uyum sağlayabilen yüksek düzeyde ölçeklenebilir, esnek ağ yapıları oluşmasına olanak tanınmıştır.

OpenFlow protokolü [48] SDN mimarisinde veri katmanı ve SDN kontrolcü arasında iletişimde standart hale gelen ilk arayüzdür. Veri katmanında yer alan anahtarlar gelen paketleri yüksek hızda anahtarlayabilen özelleşmiş donanımlardır ve sadece programlanabilir akış tablolarına göre paket iletimini gerçekleştir. Akış tabloları ise doğrudan kontrol katmanında merkezi konumda yer alan SDN kontrolcü tarafından OpenFlow protokolü kullanılarak yönetilir. Veri katmanına gelen herhangi bir paketin akış tablosu ile eşleşmemesi durumunda paketler düşürülebilir veya karar verilmesi amacıyla SDN kontrolcüsüne gönderilebilir.

OpenFlow protokolü sayesinde paket başlık bilgileri üzerinden paket işleme veri katmanında gerçekleştirilir. Buradaki desteklenen başlık sayısı birkaç yıl içerisinde 12'den 44'e çıkarılmıştır. Bu aşamaya kadar, birçok iyileştirme ve uygulama OpenFlow protokolü ile gerçekleştirilmiş olmasına rağmen, yeni fonksiyonlar veya gereksinimler için OpenFlow versiyonlarının güncellenerek yeni başlık alanlarının eklenmesi ve bu alanların veri katmanındaki donanımlar tarafından da desteklenmesi zorunluluğu söz konusudur. Zira OpenFlow destekleyen veri katmanı anahtarları, yalnızca üreticileri tarafından geliştirilen sabit fonksiyonları desteklemektedir. Bu sebeple esneklik anlamında OpenFlow protokolünün zamanla yetersiz kaldığı anlaşılmıştır. Bununla beraber özellikle ağdaki akışların durum bilgisinin tutulması ve bu akışların durumlarına göre karar verilmesi gereken uygulamalar için ilgili bütün akış bilgilerinin veri katmanından kontrolcüye iletilmesi gerekliliği söz konusudur. Bu durum hem aradaki iletişim kanalının yükünü artırmakta hem de ağ da gecikmelere sebep olduğu için performans sorunları yaratmaktadır. Bahsedilen bu problemlerin çözümü için P4 (*Programming Protocol-Independent Packet Processors*) programlama dili ile veri katmanı programlama yaklaşımı son yıllarda ortaya çıkmış ve benimsenmiştir [15,49]. Aşağıdaki Şekil 2.2' de paket anahtarlama donanımlarında geleneksel mimariden programlanabilir veri katmanı ile geçekleştirilen SDN mimarisine geçiş görülmektedir.



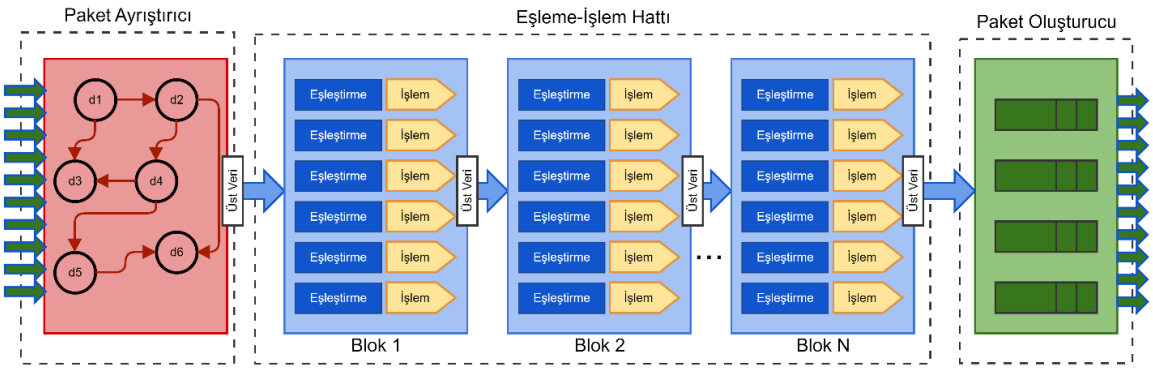
Şekil 2.2. Geleneksel mimariden programlanabilir veri katmanı ile SDN mimarisine geçiş [15]

Programlanabilir veri katmanı anahtarları üzerinde başlangıçta üreticisi tarafından geliştirilen fonksiyonların yanı sıra ilave fonksiyonların ve paket işleme yaklaşımlarının P4 programlama dili kullanılarak gerçekleştirilmesi mümkün hale gelir. P4 programlama dili kullanılarak paket başlıklarında yeni veri yapıları oluşturularak bunlara uygun paket anahtarlama yaklaşımları gerçekleştirilebilir.

P4 programlama dili, paketlerin veri katmanında nasıl işlendiğini programlamak için tasarlanmış üst düzey ve alana özgü bir dildir [49,50]. Veri katmanında kullanılan algoritmalar ağ üzerinden geçen bütün paketlerin işlenmesinden sorumlu olduğu için bu alanda yapılacak yenilikler ağ performansı ve ölçeklenebilirlik ile doğrudan ilişkilidir. P4 programlama dili aynı zamanda kullanıldığı donanımdan bağımsızdır. Günümüzde fiziksel ağ anahtarlama cihazlarında, sanal anahtarlarda, FPGA kartlarında ve ağ arayüz kartlarında P4 programlama dilini kullanmak mümkün hale gelmiştir [15].

P4 genel mimari modeli için paket işleme blokları Şekil 2.3' de yer almaktadır. Bu mimari modeli kullanan bir donanım için paket ayrıştırıcı ve paket oluşturucu blokları ile beraber bunların arasında eşleme-işlem hattı bloklarını donanımın desteklemesi söz konusudur. Paket ayrıştırıcı alanı paket başlık alanlarının durumsal olarak işlendiği aşamadır. Bu aşamada ayrıştırılan alanlara göre farklı durumlu işlemler gerçekleştirilmesi söz konusudur. Şekil 2.3'te örnek olarak altı adet farklı durum görselleştirilmiştir. Eşleme-işlem hattı

üzerinde ise gelen paketlerin belirli kontrol ve eşleştirme durumları neticesinde çeşitli işlemlerin gerçekleştirildiği alandır. Bu aşamada birden fazla eşleştirme ve işlem blokları geliştiriciler tarafından tanımlanabilmektedir. Bu sayede P4 destekleyen donanımlar üzerinde donanımdan bağımsız olmanın sağladığı esneklik ile uygulamalar geliştirilerek hayata geçirilmektedir. Son olarak paket oluşturucu alanda ise paket alanlarının tekrar birleştirilerek paketin kuyruğa alınıp gönderildiği kısımdır.

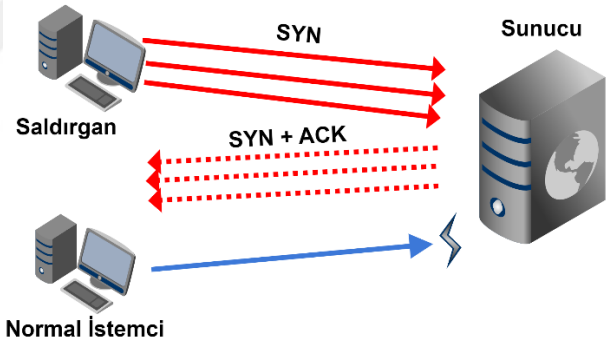


Şekil 2.3. P4 genel mimari modeli (*PISA, Protocol Independent Switching Architecture*) [51]

Programlanabilir veri katmanı yaklaşımı ile beraber ortaya çıkan P4 programlama dilinin yanı sıra önerilen yapı içerisindeki farklı donanım, arayüz ve amaçlar için farklı standartlar da ortaya çıkmıştır. P4 çalışma zamanı arayüzü (*P4 Runtime API*) P4 destekleyen bir anahtarın yönetimi için SDN kontrolcü arayüzünün standartlarını belirlemektedir [52]. Taşınabilir ağ arayüz kartı mimarisi (*Portable NIC Architecture, PNA),* bir veya daha fazla ağ arayüz kartı ile beraber fiziksel bilgisayar arasında paketleri işleyen ağ arayüz kartlarının ortak yeteneklerini tanımlayan hedef bir mimariyi ortaya koyar [53]. Benzer şekilde taşınabilir anahtar mimarisi (*The Portable Switch Architecture, PSA*), ağ anahtar cihazlarının paketleri işlemek ve iletmek için ortak özelliklerini tanımlayan bir hedef mimaridir [54] . Ayrıca ağ durumu ile ilgili veri toplanması ve raporlanmasına yönelik ortak bir çerçeve oluşturulması amacıyla bant içi ağ telemetrisi yaklaşımı da (*In-Band Network Telemetry, INT*) ortaya çıkmıştır [55,56] .

2.2. SYN Saldırıları

TCP protokolünün en temel mekanizmalarından biri bağlantı kurulumu esnasında kullanılan üçlü el sıkışma mekanizmasıdır. Bir servis engelleme saldırısı türü olan ve SYN saldırısı olarak bilinen saldırı yöntemi bu temel mekanizmanın istemci doğrulamaya yönelik bir kontrol fonksiyonu olmaması durumundan yararlanılarak gerçekleştirilir. Nitekim TCP bağlantı kurulumu esnasında istemci tarafından gönderilen bir SYN paketine sunucu tarafında herhangi bir doğrulama yapılmadan port açılarak kaynak tahsis edilir ve protokol tarafından belirlenen zaman aşım süresince açılan ilgili port belirtilen istek için tahsis edilir. Benzer şekilde aynı sunucuya belirli bir yoğunluk oranında SYN paketleri gönderilmesi durumunda ise sunucu üzerindeki sınırlı sayıda olan bütün portlar tahsis edilmiş olur ve ön tanımlı zaman aşım süresince sunucu yeni gelen SYN isteklerine yanıt veremez duruma geçer. Bunun neticesinde ilgili hizmet için servis kesintisi durumu oluşur. Aşağıda görülen Şekil 2.4'te SYN saldırısı olarak bilinen saldırı türü görülmektedir.



Şekil 2.4. SYN saldırısı

Esasen SYN saldırı türü TCP protokolünün yapısına istemci doğrulamasına yönelik bir fonksiyon kazandırılarak çözülebilir. Fakat gününüzde TCP protokolünü kullanan milyarlarca uç sistem olduğu göz önünde bulundurulduğunda protokol yapısında radikal değişiklikler yapmak neredeyse imkansızdır. Bu nedenle SYN saldırısı yöntemi yıllardır süregelen ve literatürde her zaman aktif bir çalışma alanı olarak karşımıza çıkan bir saldırı yöntemidir. Yeni teknolojiler ortaya çıktıkça, araştırmacılar tarafından çok çeşitli tespit, hafifletme ve önleme yaklaşımları geliştirilmekte ve önerilmektedir. Bu bölümde yeni bir teknoloji olarak ortaya çıkan programlanabilir veri katmanı kullanımı ile gerçekleştirilen güncel SYN savunma yaklaşımları ele alınmıştır.

2.2.1. SYN saldırılarına karşı P4 temelli savunma çalışmaları

Programlanabilir veri katmanı, hat hızında paketlerin işlenmesini sağlarken paketlerin başlık verilerinin değiştirilmesi, ağ üzerinde paketlerin anahtarlanmasındaki yöntemlerin değiştirilmesi, istemci ve sunucu arasında vekil olarak veri katmanının kullanılabilmesi gibi çok çeşitli yaklaşımların geliştirilmesini mümkün hale getirmektedir. Bu nedenle SYN saldırılarına karşı savunma tekniklerinde programlanabilir veri katmanının kullanılması cazip hale gelmiştir. Programlanabilir veri katmanı kullanılarak SYN saldırılarına karşı savunma tekniklerinin önerildiği literatürdeki güncel çalışmalar Çizelge 2.1 'de yer almaktadır.

Paolucci ve arkadaşları tarafından [57] P4 programlama dili kullanılarak programlanabilir veri katmanı üzerinde hedef ve kaynak IP adresleri başına SYN istekleri sayılarak önceden belirlenen eşik değerini aşan paketlerin düşürülmesini sağlanmıştır. Sholz ve arkadaşları tarafından [33,34] SYN çerezi ve SYN doğrulaması teknikleri gerçekleştirilmiştir. SYN çerezlerin doğrulanmasında her bir istek başına durumların veri katmanında tutulması yerine SYN-ACK ve ACK paketlerinde sırasıyla onay numarasının (*Acknowledgement Number*) oluşturulması ve doğrulanması için kriptografik bir özetleme fonksiyonu kullanmışlardır. Doğrulanan bir istek sonrasında istek yapan IP adresi veya bağlantı izin verilenler listesine alınarak devamında yapacağı bütün istekler meşru kabul edilmiştir. Benzer bir yaklaşımla SYN isteklerinin doğrulanmasını önceden oluşturulan çerez numaralarının kullanılması ile gerçekleştirilmesi [58] çalışmasında gerçekleştirilmiştir. Shen ve arkadaşları [59] sayan Bloom filtresi kullanarak kaynak IP adresi başına SYN ve SYN-ACK paketlerinin programlanabilir veri katmanı üzerinde sayılmasını gerçekleştirmiş ve bunu SYN saldırılarına karşı SYN doğrulaması ile birleştirerek kullanmıştır. Benzer şekilde Lin ve arkadaşları [60] saldırının hedefindeki sunucuya en yakın P4 anahtar üzerinde SYN-ACK ve FIN-ACK paketlerini sayarak elde edilen oransal değerler üzerinden saldırı tespitinin yapılmasını ve kontrolcünün bilgilendirilmesini sağlamışlardır. Friday ve arkadaşları [30] botnet ağları üzerinden yapılan SYN saldırılarında benzer SYN imzalarının üretildiğini iddia ederek saldırının tespiti ve hafifletilmesi için P4 anahtar üzerinde SYN imzalarının sayılmasına dayalı bir yaklaşım benimsemişlerdir. Musumeci ve arkadaşları [61,62] ise paket başlık verilerinden hat işleme hızında özellik çıkarılmasının sağlanması için P4 anahtar kullanmışlardır. Başka bir deyişle makine öğrenmesi algoritmasında kullanılmak üzere veriden özellik çıkarım görevini programlanabilir veri katmanına devretmeyi

başararak bunun sonucunda saldırı isteklerinin makine öğrenmesi temelli sınıflandırmasını gerçekleştirmişlerdir. Pegah ve arkadaşları tarafından [63] gelen SYN istekleri P4 anahtar üzerinde düşürülerek zaman aşımına dayalı yeninden iletim davranışı incelemiş ve TCP protokolü gereği artan bekleme sürelerinin oluşup oluşmama durumuna bakılarak istemcilerin saldırgan olup olmadığı tespit edilmiştir. SmartCookie [64] çalışmasında SYN saldırılarının tespiti ve engellenmesinde programlanabilir veri katmanı ve arka uç sunucu arasında görev paylaşımı yaklaşımı benimsenmiştir. Veri katmanı üzerinde güvenli kriptografik fonksiyon kullanılarak SYN isteklerinin onay numarasının doğrulanması ile saldırgan istekleri tespit edilerek engellenirken doğrulanan istekler için TCP sıra numarasının dinamik olarak takibi ve güncellenmesi görevi arka uç sunucu üzerinde eBPF [65] kullanılarak gerçekleştirilmiştir.

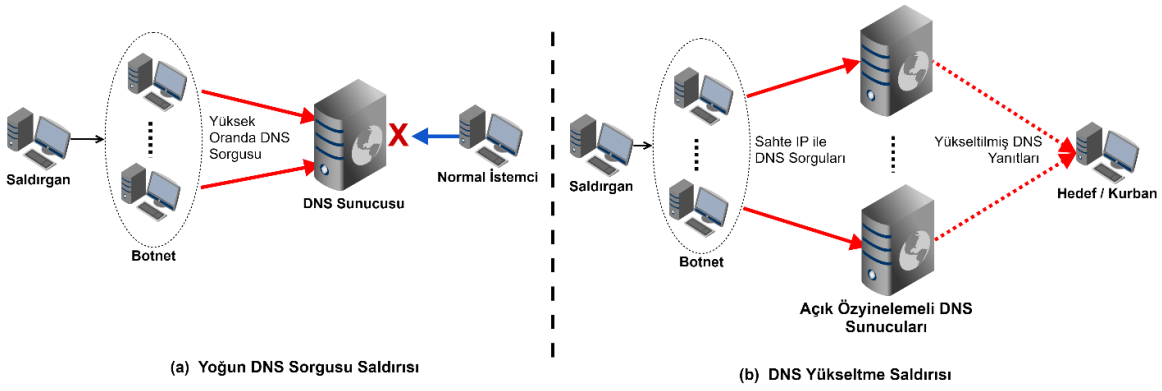
Çizelge 2.1. SYN saldırılarına karşı programlanabilir veri katmanı kullanan çalışmalar

Referans	**Yıl**	**Yöntem**	**Araçlar**
Paolucci [57]	2019	Hedef ve kaynak IP adresi başına paket sayımı	bmv2, NetFPGASume
Scholz [33,34]	2020	SYN çerezleri ve SYN doğrulaması	bmv2, libmoon(DPDK), Smart NIC
Mahrach [58]	2020	Önceden oluşturulmuş çerezler kullanılarak SYN doğrulaması	bmv2
Shen [59]	2021	SYN doğrulaması ve sayan BF temelli anomali tespiti	bmv2
Lin [60]	2020	SYN/ACK ve FIN/ACK oranlarının sayılmasına dayalı anomali tespiti	bmv2
Friday [30]	2020	SYN isteklerinde imza temelli takip	bmv2
Musumeci [61,62]	2022	Makine öğrenmesi için P4 temelli özellik çıkarımı	bmv2
Pegah [63]	2022	Zaman aşımına dayalı yeniden iletim davranışı temelli tespit	-
SmartCookie [64]	2024	Veri katmanı ve arka uç sunucu arasında görev paylaşımı	Tofino, eBPF

2.3. DNS Servis Engelleme Saldırıları

DNS sunucuları günümüzde internet altyapısının en temel ve en kritik servisi olarak kabul edilebilir. DNS servisinde yaşanabilecek kesintiler adres çözümlemelerin yapılamamasına sebep olduğundan doğrudan internet üzerinden sunulan servisler için kesintiye sebep olmaktadır. Günümüzde DNS sunucuların servis engelleme saldırılarına karşı açık olmasının iki temel sebebi olduğundan söz edilebilir. Bunlardan birincisi, DNS servisinin UDP protokolünü veri iletiminde kullanmasıdır. UDP protokolü ise durumsuz çalışır ve TCP protokolünün bağlantı kurulumunda kullandığı üçlü el sıkışmaya benzer herhangi bir mekanizmaya sahip değildir. Bu durum sahte IP adresleri üzerinden DNS sorgularının yapılabilmesine olanak tanır. İkincisi ise çok sayıda DNS sunucusunun internet üzerinden açık özyinelemeli DNS sunucusu olarak hizmet vermesidir. Bu sunuculara farklı lokasyonlardaki istemciler herhangi bir kısıtlama olmadan DNS sorgusu yapabilmektedir. Bu sayede açık DNS sunucuları saldırganlar tarafından DNS yükseltme ve yansıtma saldırılarında yaygın olarak kullanılmaktadır.

Şekil 2.5'te yoğun DNS sorgu ve DNS yükseltme saldırılarının görselleştirilmesi yer almaktadır. Yoğun DNS sorgusu saldırıları DNS sunucusunun kaynaklarını veya bant genişliğini hedef alarak DNS sunucunun normal istemci isteklerine yanıt veremez hale gelmesini amaçlar.



Şekil 2.5. DNS servis engelleme saldırılarının görselleştirilmesi

DNS yükseltme veya yansıtılmış yükseltme saldırısı olarak bilinen yöntemde ise saldırganlar tarafından açık özyinelemeli DNS sunucuları DNS isteklerinin yükseltilmesinde kullanılır. Bu yöntemde kurban olarak belirlenen hedef sistemin IP adresi kaynak IP adresi olarak

kullanılarak açık özyinelemeli DNS sunuculara yüksek oranda DNS sorgusu yapılır. Bunun neticesinde oluşan DNS yanıtları hedef sisteme yönlendirilir ve hedef sistemin bant genişliği ve kaynakları hedef alınarak hizmet kesintisi oluşması amaçlanır. UDP protokolünün durumsuz çalışması sahte IP adresleri kullanılabilmesinin önünü açmaktadır. Ayrıca yapılan saldırının yükseltilmiş olarak nitelendirilmesinin nedeni yapılan DNS sorgu paketlerinin düşük boyutlu olmasına rağmen oluşan DNS yanıt paketlerinin çok daha yüksek boyutlarda oluşabilmesinden kaynaklanmaktadır. Herhangi bir DNS sorgusu için yanıt paketinin sorgu paketine oranı yükseltme faktörü olarak isimlendirilir ve yükseltme faktörü değerinin 4670 değerine kadar ulaşabilmesi söz konusudur [23].

Bu bölümde bahsedilen DNS servis engelleme saldırılarına karşı iki farklı savunma tekniği için literatürdeki güncel yaklaşımlar ele alınmıştır.

2.3.1. DNS dağıtık servis engelleme saldırılarına karşı istek limitleme

Ağ katmanı seviyesinde gerçekleştirilen istek limitleme yaklaşımları gelen isteklerin sayısının veya bant genişliğinin kısıtlanması ile gerçekleşir. Servis engelleme veya dağıtık servis engelleme saldırılarına karşı istek limitleme yaklaşımının amacı ise saldırı isteklerine karşı istek limitleme uygulanarak etkisini en aza indirmek ve dolayısıyla oluşabilecek bir servis kesintisi durumuna engel olmaktır. Bu yaklaşım, saldırı amaçlı paketlerin etkisini azaltmak için kullanılabilir. Fakat buradaki esas zorluk, kötü amaçlı paketlerin normal paketlerden ayırt edilmesinde etkili imzalara veya modellere olan ihtiyaçtan kaynaklanmaktadır. Bunun gerçekleştirilebilmesi normal kullanıcı trafiklerinde herhangi bir kesinti durumu olmadan saldırı trafiklerinin yüksek doğruluk ile tespit edilerek engellenmesi ile mümkündür. Fakat saldırı trafiklerinin yüksek doğrulukla tespit edilerek engellenmesinde kullanılacak davranışsal modeller veya imza temelli yaklaşımlar kullanılan uygulamaya ve yapılan saldırı karakteristiğine bağlı olarak değişim gösterdiğinden zordur.

Deccio ve arkadaşları [66] tarafından 2019 yılında yetkili ad sunucularının yaklaşık %16'sında istek limitleme yaklaşımının kullanıldığı ifade edilmiştir. Jarmo [67] TCP temelli servis engelleme saldırılarına karşı istek limitlemenin etkinliğini değerlendirmiş ve normal kullanıcı isteklerine karşı %50'ye varan paket düşürmelerin tolere edilebileceğini göstermiştir. Patil ve Ragha [68] tarafından, saldırının hedefindeki uç sistemin bağlı olduğu kenar yönlendirici üzerinden saldırının tespit edilmesi sonrasında saldırganların bağlı olduğu

yönlendirici üzerinde istek limitleme gerçekleştirilmesine dayalı bir çerçeve önermiştir. Wang ve arkadaşları [69] tarafından, dağıtık servis engelleme saldırılarına karşı istek limitleme ve filtreleme uygulanması için trafik artışlarındaki şablonlar kullanmış ve birden çok yönlendirici üzerinde eş zamanlı istek limitlemesi gerçekleştirmiştir. Poseidon [42] çalışmasında programlanabilir veri katmanı üzerinde yoğun DNS sorgusu saldırılarına karşı önlem olarak aynı IP adresi veya adres bloğundan alınan IP paketleri üzerinde bir istek limitleme fonksiyonu kullanılmıştır. Ayrıca kullanılan fonksiyonun karmaşıklığının yüksek olduğu ve anahtar üzerindeki paket işleme hattında yüksek kaynak tükettiği ifade edilmiştir. Benzer şekilde Jaqen [43] çalışmasında programlanabilir veri katmanında çeşitli yoğun istek saldırılarına karşı istek limitleme fonksiyonu önerilmiştir. Önerilen fonksiyon ağdaki akışların ve kullanıcı tanımlı oranların tutulduğu bir istek limitleme tablosunu kullanırken depolama alanı kısıtlamalarından ötürü doğrudan akış verileri yerine özetleme fonksiyonu değerlerinin olası yanlış pozitif sonuçları göz önünde bulundurularak kullanılabileceği önerilmiştir.

DNS servis engelleme saldırılarına karşı ağ katmanı yaklaşımlarının yanı sıra uygulama düzeyinde DNS sunucusu üzerinde istek limitleme yaklaşım da uygulanabilmektedir. Yaygın olarak bilinen RRL yaklaşımı [35], DNS sunucusunda gerçekleştirilen bir savunma mekanizmasıdır. RRL yaklaşımı, DNS sunucusu tarafından üretilen yanıtlar arasında sıklıkla ortaya çıkan özdeş yanıtların sınırlandırılması ilkesine dayanmaktadır. Bu yaklaşımda, gelen DNS sorgularına bir kısıtlama uygulamak yerine, her istek için yanıt paketlerinin oluşturulmasına izin verilir ve yanıt paketlerinde jeton kovası temelli özdeş yanıtlar üzerinde limitleme yapılır. Yanıtların sınıflandırılması ve aynı yanıtların tespitinde, IP adresi, bölge adı, sorgu adı ve yanıt kodu gibi çeşitli parametreler kullanılır. Bu yaklaşım, bir saldırı sırasında kötü amaçlı isteklere yönelik aynı ve yüksek hacimli DNS yanıtlarının oluşturulmasını engellemeyi amaçlar ve sonuç olarak DNS sunucularına yapılan saldırıların etkisini azaltmayı hedefler. RRL yaklaşımının etkinliği saldırının özelliğine göre değişiklik gösterebilir. Rozekrans ve arkadaşları [36] tarafından, yapılan saldırılarda sorgu çeşitliliğinin artması ile etkinliğin azaldığını gösterilmiştir. Bunun nedeni, farklı sorguların farklı yanıtlar oluşturması neticesinde RRL yaklaşımında bu yanıtların farklı gruplanmasıdır. Bu gibi durumlarda, ağ katmanında istemci IP adresi tabanlı istek limitleme uygulanması daha güçlü bir savunma mekanizması görevi görebilir. Örneğin, Google tarafından sunulan açık özyinelemeli DNS hizmeti IP adresi başına saniyede 1500 sorguya izin verir [70]. DNS

hizmetinde servis engelleme saldırılarına karşı çok katmanlı savunma mekanizmalarının kullanılmasının elzem olduğu aşikardır.

2.3.2. DNS yükseltme saldırılarına karşı birebir eşleştirme temelli savunma çalışmaları

DNS protokolünün veri iletiminde UDP protokolünün kullanması sebebiyle sahte IP adresleri kullanılarak yükseltme saldırıları kolaylıkla yapılabilmektedir. Küresel ölçekte bütün internet altyapısı göz önünde bulundurulduğunda, BCP38 [21] yönteminin sahte IP adreslerinin kullanılmasının engellenmesinin tek yolu olduğu söylenebilir. Fakat servis sağlayıcılar tarafından gerçekleştirilen kısıtlı adaptasyondan ötürü sahte IP adresleri kullanılarak saldırı gerçekleştirilmesi mümkün hale gelmektedir [22] . Bu nedenle her ne kadar DNS yükseltme saldırıları tamamen engellenemez olsa da, 2007 yılında Kambourakis [38] tarafından önerilen birebir eşleştirme yaklaşımı kullanılarak hedef sistem tarafında yansıtılmış DNS yanıtlarının tespit edilmesi ve engellenmesi gerçekleştirilebilir. Şekil 2.5 göz önünde bulundurulduğunda bu yaklaşımda DNS sorgu ve yanıt paketlerinin açık özyinelemeli DNS sunucular ve hedef sistem arasındaki ağ anahtarları üzerinde herhangi bir noktada izlenmesi ve görülmeyen DNS sorgularına alınan yanıtların engellenmesi esasına dayanmaktadır. Bu yaklaşım hedef sistem tarafından veya hedef sistemin hizmet aldığı Internet Servis Sağlayıcı (ISS) tarafından uygulanabilir. ISS' in bu yöntemi kenar anahtarlarında uygulaması ile güçlendirilmiş DNS yanıt paketleri ağ içerisine girmeden engellenebilir. Burada dikkat edilmesi gereken kritik nokta sahte IP adresi kullanan DNS sorgu paketlerinin tespit noktası olarak kullanılan anahtardan geçmiyor olması gerekliliğidir. Yapılan saldırıların kaynağı ISS' nin kendi ağ yapısı içerisinde olursa veya başka bir deyişle saldırı sorgu paketlerinin aynı kenar anahtar üzerinden geçmesi durumunda saldırı yanıtları ayrıştırılamadığı için engellenemez.

Çizelge 2.2'de birebir eşleştirme yaklaşımı kullanılarak yükseltilmiş DNS yanıt paketlerinin engellenmesi yaklaşımını kullanan literatürdeki çalışmalar yer almaktadır. Kambourakis [38] tarafından geçekleştirilen çalışmada IPTraf aracı kullanılarak toplanan istatistiksel DNS verileri kullanılarak birebir eşleştirme ile yükseltilmiş yanıtların engellendiğini göstermiştir. Sun ve arkadaşları [71] tarafından, DNS sorgu ve yanıt paketlerinin sayısı ve aradaki fark değerinin takip edilerek farksal değerin belirlenen eşik değerini aşıp aşmama durumuna bağlı olarak saldırı tespiti gerçekleştirilmiştir. Saldırı tespit edilme durumunda saldırı

olmayan DNS yanıtlarının tespitinde BF veri yapısı kullanılmıştır. BF veri yapısından veriler silinemediğinden 2 adet BF veri yapısı kullanılarak aynı anda bir BF veri yapısının temizlenmesi düzenli aralıklarla sağlanmıştır.

Çizelge 2.2. Birebir eşleştirme temelli çalışmalar

Referans	**Yıl**	**Yöntem**	**Araçlar**
Kambourakis [38]	2007	Sorguların takibi için istatistiksel DNS verisi kullanımı	IPTraf
Sun [71]	2008	DNS sorgu ve yanıt paketlerini farkının takibi ile saldırı tespiti	Veri seti ile simülasyon
Paola [40]	2011	2 adet BF kullanımı ve düzenli periyotlarda veri yapısının temizlenmesi	Juniper anahtar
Sattar [41]	2013	Sorgu ve yanıtlar için 2 ayrı BF kullanımı	-
Poseidon [42]	2020	Son 1 saat içerisinde sorgu yapılan IP adreslerinden gelen yanıtlara izin verilmesi	Barefoot Tofino
Jaqen [43]	2021	Sayan BF veri yapısının yapılmayan DNS sorgularının tespitinde kullanımı	Barefoot Tofino

Paola ve arkadaşları [40] tarafından benzer iki adet BF veri yapısı yaklaşımı kullanılarak DNS paketleri Juniper anahtar üzerinde izlenmiş ve yükseltilmiş DNS yanıtlarının engellenmesi sağlanmıştır. Sattar ve arkadaşları [41]tarafından DNS sorgu ve yanıt paketleri için iki farklı BF veri yapısı kullanılmış ve DNS yanıt süresine bağlı olarak saldırgan IP adreslerinin tespit edilerek engelleme listesine eklenmesi sağlanmıştır. Poseidon [42] çalışmasında son bir saat içerisinde yapılan DNS sorgularının hedef IP adresleri izlenerek yalnızca bu adreslerden gelen DNS yanıtlarına izin verilmiştir. Ancak programlanabilir veri katmanı üzerinde gerçekleştirilen bu çalışmada hedef DNS sunucularının saldırganlar tarafından kullanılması durumunda bu DNS sunucuların IP adreslerine izin verildiğinden bunlardan gelen yükseltilmiş DNS yanıtları engellenemez. Açık özyinelemeli DNS sunucuları yükseltme saldırılarına dahil olabildiğinden bu sunucuların IP adreslerini izin verilenler veya engellenenler listesine eklemek yerine durumsal olarak DNS paketlerinin izlenmesinin daha etkin bir savunma sağlayacağı aşikardır. Jaqen [43] çalışmasında DNS sorgu ve yanıtlarının durumsal olarak izlenmesinde sayan BF veri yapısı kullanılarak birebir eşleştirme yaklaşımı ile eşleşmeyen DNS yanıt paketlerinin engellenmesi sağlanmıştır. Bunun gerçekleştirilmesinde durumsal takip için kaynak ve hedef IP adresleri, port numaraları ve DNS başlık değerleri kullanılmıştır. Bildiğimiz kadarıyla Jaqen [43] çalışması, şu an için programlanabilir veri katmanında DNS yükseltme saldırılarına karşı bire bir eşleştirmeyi temel alan en yeni yaklaşımıdır. Ancak bu çalışmada üç adet sınırlamanın olduğundan söz edilebilir. Bunlardan birincisi parçalanmış DNS yanıt paketleri yükseltme

saldırılarında kullanılabilir fakat üst katman bilgileri (DNS başlık değerleri, port numarası vb.) sadece ilk parça paketinde yer alır ve sonraki parça paketlerde bu veri bulunmaz. Bu nedenle önerilen yaklaşım parçalanmış DNS yanıtlarına uygulanamaz. İkincisi, sayan BF veri yapısı içerisinde yanıt alınamayan DNS sorguları gibi çeşitli sebeplerden zamanla silinemeyen veriler oluşabilir ve bu durum veri yapısının doğruluğunu azaltabilir. Yüksek doğruluk için zamanla veri yapısının temizlenmesi gerekliliği söz konusudur. Üçüncüsü ise IP ve port numaraları saldırganlar tarafından kolaylıkla taklit edilebilir, bu nedenle DNS sorgularının eşsiz olarak tanımlanması ve takip edilmesinde uygulama seviyesinde kullanılan DNS işlem numarası parametresi kullanımı daha uygundur.

3. YÜKSEK DOĞRULUK SAĞLAYAN GÜNCELLENEBİLİR BLOOM FİLTRESİ TASARIMI

Bloom filtresi (BF), Burton Bloom tarafından 1970 yılında önerilen ve adını mucidinden alan olasılıksal bir veri yapısıdır. Bu veri yapısının en güçlü yönleri gerektirdiği düşük depolama alanı ve sağladığı sabit sorgulama süresi olarak nitelendirilebilir [24]. Bu yönüyle bakıldığında programlanabilir veri katmanında yüksek depolama gereksinimi olan fonksiyonlar için oldukça cazip bir çözüm olarak ortaya çıkar. Fakat yapısı gereği oluşan yanıtlar içerisinde yanlış pozitif (*false positive*) durumların oluşabilmesi söz konusudur. Bunun durumun sebebi ise veri yapısına yeni veriler eklendikçe kullanılan özetleme (*hash*) fonksiyonlarının çakışma olasılıklarının artması ve oluşan çakışmalar neticesinde yanlış pozitif sonuçlar oluşmasıdır. Esasen BF veri yapısına yeni veriler eklenebilir fakat doğası gereği eklenen elemanların çıkarılması söz konusu değildir. Bu nedenle programlanabilir veri katmanında yapılan çalışmalar incelendiğinde yeni veriler eklendikçe doğruluğun düştüğü ve bunun engellenmesi için çeşitli çalışmalarda veri yapısının tamamen silinerek yeninden kullanıldığı veya uygulama alanına özgü farklı BF veri yapılarının önerilerek kullanımı konusudur [25-32].

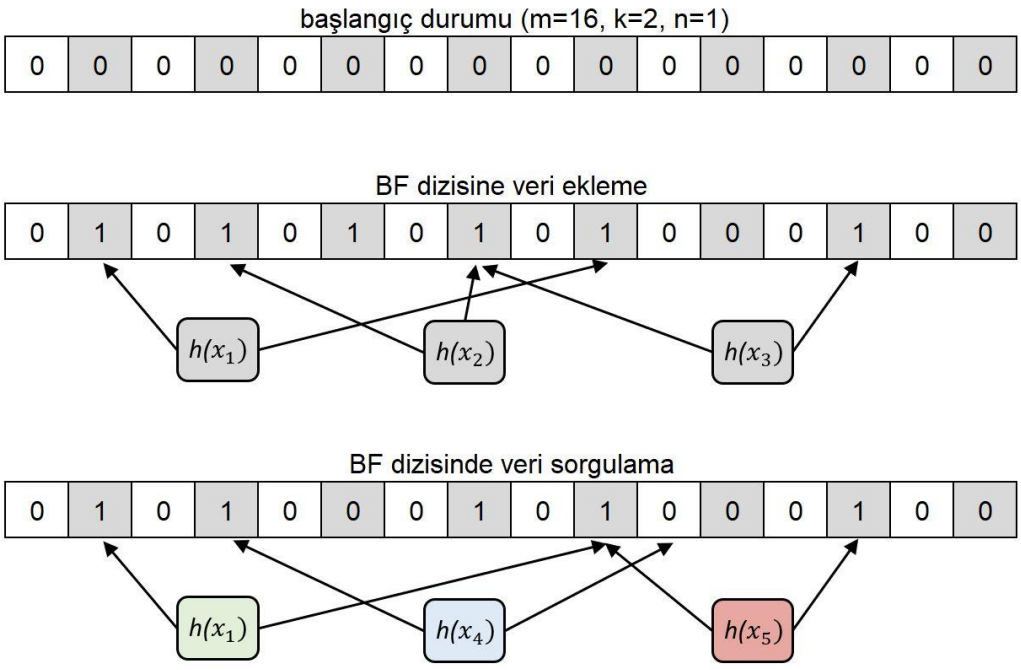
Bu bölümde bilgisayar ağlarında çok çeşitli uygulamalarda kullanılan bu veri yapısının temel bileşenleri ve çeşitli uygulamaları ile beraber önerilen güncellenebilir Bloom filtresi veri yapısının tasarımı ve çalışma yapısı detaylı olarak ele alınmıştır.

3.1. Bloom Filtresi Veri Yapısı

Bloom filtresi veri yapısı alan-etkin çalışan ve sabit sorgulama zamanı sağlayan olasılıksal bir veri yapısıdır [24]. Bu veri yapısı kullanılarak verilen bir elemanın bir veri kümesi içerisinde olup olmama durumu sabit sorgulama süresi ve düşük depolama alanı kullanılarak sorgulanabilir. Bu veri yapısının olasılıksal olarak nitelendirilmesinin sebebi ise oluşan yanıtlar içerisinde yanlış pozitif durumların oluşabilmesinden kaynaklanmaktadır. Fakat bunun tam tersi olan yanlış negatif (*false negative*) yanıtların oluşması söz konusu değildir.

S veri kümesinin $S = \{x_1, x_2, x_3, \dots, x_t\}$ şeklinde t adet elemandan oluştuğunu varsayalım. Bloom filtresi bu veri kümesini m-bit uzunluğunda ve bir bit genişliğinde bir veri dizisinde tutar. Buradaki her bir x veri kümesi elemanının m-bit uzunluğundaki diziye eklenmesinde

k adet birbirinden bağımsız özetleme fonksiyonu $\{h_1, h_2, \dots, h_k\}$ kullanılır. Bunun sonucunda BF dizisi üzerinde x elemanı için k adet konum $\{h_1(x), h_2(x), \dots, h_k(x)\}$ belirlenir ve belirlenen bu konumlara bir değeri yazılır. Belirlenen her bir $h_i(x)$ değeri 0 ile (m-1) değerleri arasındadır. Başlangıçta BF dizisi üzerinde bütün elemanlar sıfır değerindedir. Bu işlem *S* veri kümesinin bütün elemanları için tekrar edilir. Daha sonrasında gelen herhangi bir *x* elemanının *S* veri kümesinde olup olmadığının sorgulanmasında aynı şekilde *k* adet özetleme fonksiyonu kullanılarak $h_i(x)$ değerleri belirlenir ve belirlenen bu konumlardaki bütün değerlerin bir değerli olup olmaması durumuna bakılır. Şayet *k* adet konumdan herhangi birinin değeri sıfır ise, *x* elemanının *S* kümesi içerisinde olmadığı kesin bir doğrulukla söylenebilir. Fakat *k* adet konumdaki bütün değerlerin bir olması durumunda *x* elemanının *S* veri kümesinde olduğu kesin olmayan bir doğrulukla söylenebilir [24].



Şekil 3.1. Örnek bir BF veri yapısı üzerinden veri ekleme ve sorgulamanın gösterimi

Şekil 3.1' de örnek bir Bloom filtresi veri yapısı üzerinden veri ekleme ve sorgulama adımları gösterilmiştir. BF dizi boyutu (m) 16, dizi genişliği 1 bit (n) ve kullanılan özetleme fonksiyonu sayısı (k) 2 olarak seçilmiştir. Başlangıçta BF dizisinde bütün değerler sıfır iken eklenen x_1, x_2 ve x_3 elemanları iki adet özetleme fonksiyonu kullanılarak BF dizisine eklenmiştir. Ekleme esnasında x_2 ve x_3 elemanlarının birer özetleme fonksiyonları için BF dizisi üzerinde çakışma olduğu görülmektedir. Benzer şekilde BF dizisine yeni elemanlar

eklendikçe kullanılan özetleme fonksiyonlarının çakışma olasılıkları artar ve artan çakışmalar sonucunda yanlış pozitif sonuçlar ortaya çıkar. Başka bir deyişle, özetleme fonksiyonu çakışmaları yanlış pozitif sonuçların oluşmasına sebep olur.

BF dizisi üzerinde devamında x_1, x_4 ve x_5 elemanları için sorgulama yapılmıştır. x_1' in sorgulanmasında her iki özetleme fonksiyonunun sonucunda elde edilen konumlardaki değerlerin 1 olması neticesinde sorgulama sonucu pozitif olarak elde edilirken, x_4 için elde edilen iki konumdan birinde değerin 0 olması sebebiyle sorgulama sonucu negatif olarak elde edilmiştir. Fakat x_5 elemanı için yapılan sorgulama sonucunda her ne kadar bu eleman BF dizisine öncesinde eklenmemiş olsa da iki özetleme fonksiyonunun sonucunda elde edilen konumlardaki değerlerin 1 olması sebebiyle yanlış pozitif sonuç elde edilir. Her ne kadar Bloom filtresi veri yapısı yanlış pozitif sonuçlara sebep oluyor olsa da sağlamış olduğu kısıtlı depolama gereksinimi ve sabit sorgulama süresi avantajları sebebiyle kabul edilebilir düşük hata oranı ile yaygın kullanıma sahiptir.

Bloom filtresi veri yapısının doğası gereği eklenen elemanların diziden çıkarılması söz konusu değildir. Bunun nedeni BF dizisinde aynı konumu kullanan birden çok elemanın olması durumunda ilgili konumun değerinin 1 olarak kalmasıdır. Nitekim diziden eleman çıkarmak için ilgili konum değerinin 0 yapılması aynı konumu kullanan diğer bütün elemanlar için yanlış negatif sonuçların oluşmasına neden olur. Bu durumun neticesinde yeni bir yaklaşım olarak sayan Bloom filtresi [72] (*Counting Bloom Filter, CBF*) yaklaşımı ortaya çıkmıştır. Sayan Bloom filtresi yaklaşımında BF dizisinin genişliği artırılarak her bir konum birer sayaç olarak kullanılmaktadır. Bu sayede diziye eleman eklenmesi durumunda ilgili konumlardaki değerler birer artırılırken diziden eleman çıkarılması esnasında ilgili konum değerleri birer azaltılmaktadır. Herhangi bir elemanın durumunun sorgulanması için ise benzer şekilde ilgili bütün konum değerlerinin sıfırdan farklı olup olmaması durumuna bakılmaktadır.

Sayan Bloom filtresinin yanı sıra, verilerin silinebilmesine imkan sağlayan faklı yapıda alternatif Bloom filtresi temelli veri yapıları mevcuttur. Bunlarda Üçlü Bloom filtresinde [73] (*Ternary Bloom Filter, TBF)* CBF' nin aksine daha düşük genişlikte sayaçlar kullanarak veri yapısında daha düşük yanlış pozitif sonuçlar elde edildiği gösterilmiştir. TBF veri yapısında sayaç içerisinde sadece 0, 1 ve X değerleri atanabilmektedir. Dizi içerisindeki herhangi bir konuma 2 veya daha fazla verinin yazılması gerektiği durumda bu alanlara X

değeri atanmaktadır. X değeri atanan bu alanlar, veri eklenmesinde, veri silinmesinde ve sorgulamalarda referans olarak kullanılmamaktadır. Şayet herhangi bir x verisinin k adet noktada sorgulanmasında ilgili bütün alanların X değeri aldığı görülürse yapılan sorgulamanın sonucu tespit edilemez olarak kabul edilmektedir. Benzer şekilde herhangi bir x verisinin k adet noktadan silinmesinde ilgili bütün alanların X değeri aldığı görülürse x değeri silinemez kabul edilmektedir. Bu şekilde çakışmaların oluştuğu alanların etiketlenmesi ile beraber silinemez ve sorgulanamaz değerlerin tespit edilmesiyle yanlış pozitif sonuçların azaltılması ve yanlış silmelerden oluşabilecek yanlış negatif sonuçların önüne geçilmesi sağlanmış olur.

Silinebilir Bloom filtresi [74] *(Deletable Bloom Filter, DlBF)* olarak ortaya çıkan bir diğer veri yapısında ise 1 bit genişliğe sahip BF dizisi r adet bölgeye ayrılarak bu bölgelere veri eklenirken çakışma olup olmama durumu takip edilir. Şayet çakışmanın olduğu bir bölge tespit edilirse bu bölgeden veri silinmez, ancak çakışma oluşmamış bölgelerdeki verilerin silinebilmesine izin verilmektedir. Bunun gerçekleştirilmesinde m uzunluğundaki BF dizisi $m' = m - r$ olacak şekilde r adet m'/r uzunluğunda bölgelere ayrılır. Oluşturulan her bir bölgenin çakışmaya sahip olup olmama durumunu tutmak için oluşturulan r dizisindeki bütün bit değerleri başlangıçta 0 iken çakışma tespit edilen bölgeler için ilgili bit değerleri 1 olarak güncellenir. Devamında veri silinmesi esnasında ilgili bölgeler için r dizisindeki değerlere bakılarak çakışma olmayan bölgelerdeki verilerin silinmesine izin verilir. Herhangi bir x verisinin k adet alandan silinmesinde herhangi bir değerin 0 yapılabilmesi durumunda bu değer silinebilir kabul edilmektedir. Şayet k adet alanın tamamı çakışmaların oluştuğu alanlarda ise x verisi silinemez olarak kabul edilmektedir. Burada anlatılan veri yapılarının yanı sıra uygulama alanına özgü olarak geliştirilen ve verilerin silinmesine imkan veren Bloom filtresi temelli farklı veri yapılarından da söz edilebilir [39, 75-77] .

Her ne kadar orijinal Bloom filtresi tek bir dizi üzerinde birden çok özetleme fonksiyonu ile veri yazılmasını gerçekleştirmiş olsa da her bir özetleme fonksiyonu için ayrı dizilerin kullanılabilmesi veya BF dizisinin belirli sayıda özetleme fonksiyonu için bölünerek kullanılması da söz konusu olabilir. Bunun gerçekleştirilmesi her bir özetleme fonksiyonu özelinde veri yapısının boyutunun azalması sebebiyle yanlış pozitif olasılığında artışa sebep olabilir. Fakat esasen hafıza alanına kısıtlı erişim destekleyen donanımlar için BF dizisinin özetleme fonksiyonları için bölünmesi paralel hafıza erişimine olanak tanıdığı için sorgulamalar için hafıza erişim süresini kısaltır [8]. Anahtarlama cihazlarında IP paketlerinin

yönlendirme tablosuna uygun olarak işlenmesinde kullanılan en uzun ön ek eşleşmesi (*Longest Prefix Matching, LPM*) algoritması için Bloom filtresi kullanan [78] çalışmasında, dizi boyutunun bölünerek farklı özetleme fonksiyonlarına tahsis edilmesi önerilmiştir. Bu sayede her bir sorgu için hafıza erişiminde sorgulama sınırlarının aşılmadığı vurgulanırken yeterince büyük BF dizileri için yanlış pozitif oranının artmadığı ifade edilmiştir. Benzer şekilde OMASS [79] çalışmasında küçük boyutlardaki BF dizilerinde bu durumun yanlış pozitif oranını arttırmasının yanında tek bir hafıza erişimi ile sorgulama yapılabilmesine olanak tanıyarak azalan hafıza erişimi ile performans artışı yapıldığı gösterilmiştir. MPCBF[80] çalışmasında da benzer şekilde BF dizisi birden çok parçaya bölünerek bunlar farklı özetleme fonksiyonlarına tahsis edilmiştir. Böylece önerilen hiyerarşik bir veri yapısı üzerinden BF dizisi için daha iyi bir kullanım ile beraber CBF ile kıyaslandığında daha düşük bir yanlış pozitif oranı elde edilmiştir. UFBF [81] çalışmasında hem özetleme fonksiyonlarının çalışması hem de hafızadan veri sorgulama işlemlerinin paralel olarak donanım üzerinde gerçeklenmesi ile hız anlamında performans iyileştirmesi gerçekleştirilmiştir. Bu amaçla k adet özetleme fonksiyonu ile beraber her bir özetleme fonksiyonuna özgü k adet farklı BF dizisi kullanılmıştır. Bloom-1 [82] çalışmasında ise aynı elemana ait veriler aynı dizide olacak şekilde BF dizisi l adet w uzunluğunda alt dizilere bölünmüştür. Bunun gerçekleştirilmesinde öncelikle hangi alt diziye verinin yazılacağı tek bir özetleme fonksiyonu ile belirlenerek sonrasında belirlenen alt diziye veri yazılmasında bütün özetleme fonksiyonları kullanılmaktadır. Bu sayede örnek olarak verilen 3 adet özetleme fonksiyonu için 60 bit özetleme verisi kullanımı yerine 32 bit kullanılmış ve verinin sorgulanması tek bir hafıza erişimi gereksinimi ile gerçekleştirilebilir hale getirilmiştir.

3.1.1. Programlanabilir veri katmanlarında Bloom filtresi kullanımı

Bloom filtresi temelli veri yapıları çok çeşitli alanlarda ve yapılarda ağ güvenliğinin sağlanmasına yönelik yaklaşımlarda yaygın olarak kullanılmaktadır [83]. Bu bölümde bu veri yapısının programlanabilir veri katmanları özelinde çeşitli uygulamalar için kullanımı ele alınmıştır.

Programlanabilir veri katmanları esneklik, hat hızında paket işleme ve ölçeklenebilirlik açısından oldukça güçlüdür. Günümüzde modern programlanabilir veri katmanları port başına 400GbE' e kadar yüksek bant genişliği sunabilmektedir [84]. Diğer taraftan bu

donanımlar üzerinde statik rasgele erişim belleği *(Static Random Access Memory, SRAM)* veya üçlü adreslenebilir içerik tabanlı bellek *(Ternary Content-Addressable Memory, TCAM)* olarak adlandırılan ve yüksek maliyetlerden ötürü kısıtlı depolamaya imkan sağlayan hafıza alanları mevcuttur. Özellikle durumsal ağ fonksiyonları olarak bilinen güvenlik duvarı, akış takibi veya yük dengeleyici gibi çeşitli fonksiyonların yoğun depolama alanı gerektirmesinden ötürü programlanabilir veri katmanlarında gerçekleştirilmesi zordur [85]. Bu nedenle alan-etkin algoritmaların programlanabilir veri katmanlarında gerçekleştirilmesine ihtiyaç vardır. Bloom filtresi veri yapısının sağladığı sabit sorgulama süresi ve düşük depolama alanı gereksiniminden ötürü programlanabilir veri katmanları için oldukça cazip bir çözüm olarak ortaya çıkmaktadır.

Sağlamış olduğu avantajlardan ötürü programlanabilir veri katmanında Bloom fitresi veri yapısı araştırmacılar tarafından çeşitli uygulamalar için kullanılmıştır. Çizelge 3.1' de yapılan güncel çalışmalar yer almaktadır.

Çizelge 3.1. Programlanabilir veri katmanında Bloom filtresi kullanımı

Referans	**Yıl**	**Yöntem**	**Hedef Donanım**
Hill [25]	2018	Geleneksel ve sayan BF	Bmv2
Basuki [26]	2020	Paket başlıkları içerisinde BF	Bmv2
Silkroad [27]	2017	Geçiş tablosunda sadece okuma ve sadece yazma için BF	Intel Tofino
Pavlidis [28,29]	2020	Devir süresi temelli BF	SmartNIC [86]
Friday [30]	2020	BF temelli SYN imza takibi ve TCP paketler arası zaman takibi	Bmv2
Loom [31]	2021	Çoklu BF kullanımı	Intel Tofino
Wang [32]	2023	Kayan Pencere temelli BF	Intel Tofino

Hill tarafından [25] çalışmasında ağdaki akışların takibinde Bloom filtresi kullanımı ve etkinliği incelenmiştir. Çeşitli BF dizisi uzunluğu ve değişen özetleme fonksiyonları sayısına bağlı olarak farklı trafik ve istek yoğunluklarında veri yapısının sağlamış olduğu doğruluğun değişimi incelenmiştir. Yapılan test sonuçlarından düşük istek yoğunluklarında BF veri yapısı yüksek doğruluk sağladığı görülmüştür. Fakat artan istek yoğunluklarında silinemeyen verilerden ötürü özetleme fonksiyonlarında çakışmaların arttığı ve doğruluğun düştüğü gözlemlenmiştir. Basuki tarafından [26] çalışmasında ağdaki paketlerin topoloji üzerinde takip ettiği yolun izlenmesi ve aynı zamanda ağ gizliliğini korunması amacıyla bir çalışma gerçekleştirilmiştir ve bu amaçla Bloom filtresi verisi paket başlık alanlarının içerisinde tutulmuştur. Ağdaki atlama sayısının artması ile beraber özetleme

fonksiyonlarında artan çakışmalar nedeniyle yanlış pozitif oranının da arttığı gösterilmiştir. SilkRoad [27] çalışmasında dördüncü katmanda çalışan yük dengeleyici olarak yazılım temelli uygulamalar yerine modern programlanabilir anahtarlar kullanılarak yük dengeleme görevi maliyetinin ciddi oranda düşürüldüğü ifade edilmektedir. Gerçekleştirilen çalışmada yük dengeleyici arkasında bulunan sunucu IP adresleri havuzunun güncellenmesi esnasında kullanılan geçiş tablosu içerisinde BF veri yapısı kullanılmıştır. Oluşturulan veri yapısı belirli bir süre sadece yazma ve belirli bir süre sadece okuma amacıyla kullanılmıştır. Geçiş esnasında BF veri yapısında oluşan çakışmaların yanlış pozitif sonuçlara sebep olmasının önlenmesi amacıyla bu tür istekler tespit edildikten sonra geçiş tablosu yerine anahtarlama cihazına yönlendirilmiştir. Pavlidis tarafından [28,29] çalışmalarında devir süresi temelli sayan bloom filtresi yaklaşımı belirli bir süre içerisinde gelen toplam trafik akışlarının takip edilmesi ve sayılması amacıyla kullanılmış ve [28] çalışmasında 176 KB boyutunda bir BF dizisi ve 10 adet özetleme fonksiyonu kullanılarak 100 000 isteğin %0,1 yanlış pozitif hata oranı ile takip edebildiğini gösterilmiştir. Friday ve arkadaşları tarafından [30] çalışmasında ise yoğun SYN saldırılarına karşı TCP SYN paketlerinin çeşitli özellikleri (boyutu, TTL değeri, IP başlık uzunlukları vs) kullanılarak yapılan isteklerin takibi ve imza tabanlı izlenmesinde Bloom filtresi kullanılmıştır. Aynı zamanda TCP bağlantılarında paketler arası zamanın ölçülmesi ve takibinde de Bloom filtresi kullanılması söz konusudur. Loom isimli çalışmada [31] programlanabilir veri katmanında gerçekleştirilen yük dengeleyici için gelen isteklerin durumu birden çok BF veri yapısı oluşturularak tutulmuş ve yanlış pozitif durumların önlenmesi için bu veri yapıları sıralı olarak düzenli aralıklarla temizlenmiştir. Wang ve arkadaşları [32] çalışmasında ise kayan pencere temelli BF veri yapısı durumsal paket işlemenin doğrulanmasında kullanılmıştır. Bahsedilen kayan pencere yaklaşımında BF dizisi birden çok parçaya ayrılmış ve yeni gelen IP adresleri her zaman en güncel BF parçasına yazılmaktadır. Bunun yanı sıra en eski parça temizlenerek tekrar kullanım için hazırlanmaktadır.

3.2. Güncellenebilir Bloom Filtresi Yaklaşımı

Bloom filtresi veri yapısı programlanabilir veri katmanlarında belirli akış değerlerine kadar yüksek doğruluk sağlamaktadır. Fakat artan veri oranları ile beraber doğruluk değeri düşer, bunun en temel nedeni Bloom filtresinden güncelliğini yitirmiş verilerin silinememesidir. Bu durum BF veri yapısının en temel dezavantajıdır. Ağ akışlarının takibinde belirli bir doğruluk değerinin sağlanması için BF boyutu veya kullanılan özetleme fonksiyonu sayısı

artırılabilir. Bu durum BF veri yapısının kapasitesini ve doğruluğunu belirli bir süre yüksek oranlarda tutabilir. Fakat yine de silinemeyen verilerin artması ve sınırlı depolama alanından ötürü zamanla doğruluğunun düşmesi durumu kaçınılmazdır.

Esasen BF' nin doğruluğunun düşmesinin en temel nedeni artık güncelliğini yitirmiş ve gereksiz verilerin veri yapısında tutuluyor olmasından kaynaklanmaktadır. Bu nedenle aşağıdaki Şekil 3.2'de görülen zaman damgası kullanan güncellenebilir Bloom filtresi (GBF) veri yapısı akış takibinde kullanılmak üzere geliştirilmiştir. Önerilen yöntem BF dizisi üzerindeki verilerin güncelliğini ön tanımlı devir süresini referans alarak kontrol eder ve güncelliğini yitirmiş verileri dinamik olarak temizler. Verilerin güncellik süresi kullanılan uygulama ve çalışma yapısına göre değişiklik gösterebilir.

Şekil 3.2'de önerilen veri yapısı 2-bit genişliğindedir ve aynı zamanda her bir dizi elemanı için zaman damgası (ZD) alanı mevcuttur. BF dizisi üzerinde herhangi bir alana veri yazılması veya silinmesi esnasında ilgili alandaki ZD değeri kullanılarak mevcut verinin güncellik durumu kontrol edilir. Aynı zamanda dizi genişliğinin 2-bit olması aynı noktada çakışan verilerin tutulabilmesine olanak tanır.



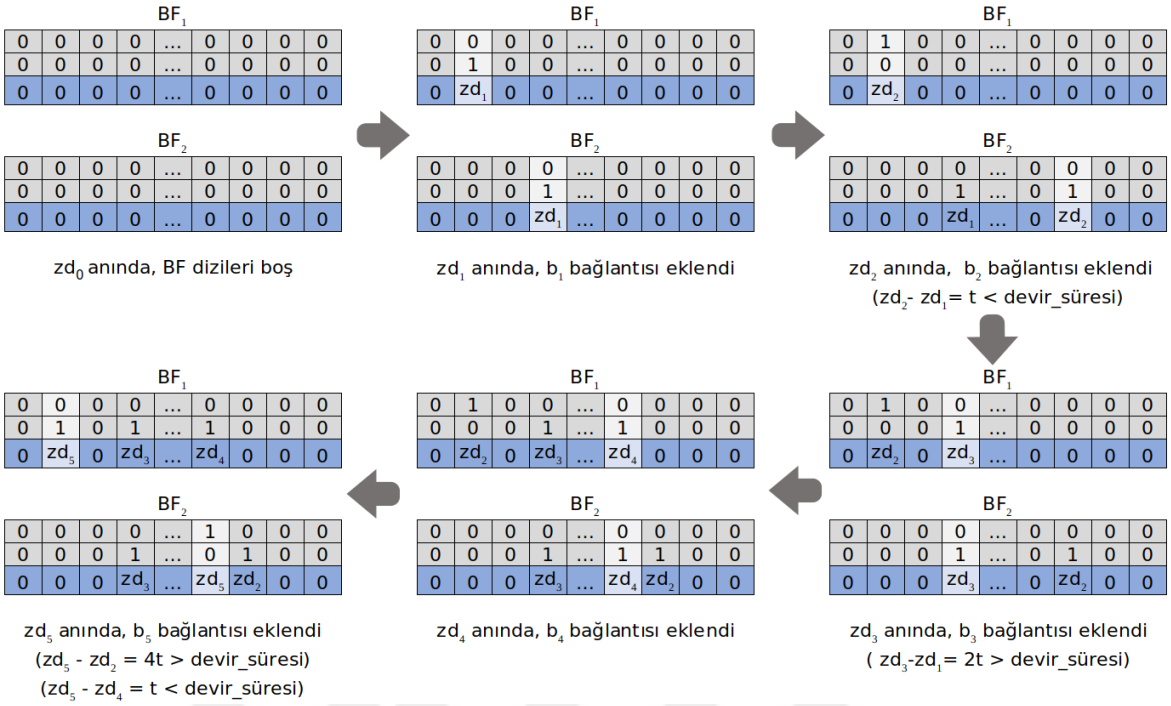
(a) 1-Bit Bloom Filtresi (b) 2-Bit Güncellenebilir Bloom Filtresi

Şekil 3.2. Güncellenebilir Bloom Filtresi Veri Yapısı

Şekil 3.3'te görüldüğü gibi örnek 5 adet bağlantı isteğinin belirtilen zamanlarda geldiğini ve aynı zamanda devir süresinin t ile $2t$ arasında olduğu varsayalım. Önerilen yaklaşımda gelen verilerin nasıl işlendiği detaylı olarak Şekil 3.4'te görsel olarak yer almaktadır.

Zaman	Zaman Damgası	Bağlantılar	
0	zd_0	0	
t	zd_1	b_1	
2t	zd_2	b_2	t < devir_süresi < 2t
3t	zd_3	b_3	
5t	zd_4	b_4	
6t	zd_5	b_5	

Şekil 3.3. Örnek 5 adet bağlantı isteği



Şekil 3.4. Önerilen güncellenebilir Bloom filtresi veri yapısı ve çalışmasının gösterimi

Önerilen yaklaşımda verilen örnek her bir bağlantı isteği için gerçekleşen adımlar şu şekilde sıralanabilir.

- zd_0 anında: BF dizileri boş.
- zd_1 anında: b_1 bağlantısı BF dizilerine eklenir.
- zd_2 anında: b_2 bağlantısı BF dizilerine eklenir, BF_1 dizisinde çakışma var, veri güncel, o sebeple değer artırıldı.
- zd_3 anında: b_3 bağlantısı BF dizilerine eklenir, BF_2 dizisinde çakışma var, veri güncel değil, o sebeple değer 1 olarak güncellendi.
- zd_4 anında: b_4 bağlantısı BF dizilerine eklenir, çakışma yok.
- zd_5 anında: b_5 bağlantısı BF dizilerine eklenir, hem BF_1 hem de BF_2'de çakışma var. BF_1 içerisindeki veri güncel değil, o sebeple değer 1 olarak güncellendi. BF_2 içerisindeki veri güncel, o sebeple değeri artırıldı.

Önerilen yöntem verilerin yazılacağı noktadaki mevcut verinin zamansal güncelliğinin kontrol edilmesi esasına dayanır. Şayet verinin yazılacağı alandaki eski verinin zaman damgası ile yeni gelen verinin zaman damgası arasındaki zamansal fark belirlenen devir süresinden az ise eski veri güncel kabul edilmekte ve ilgili alandaki değer bir artırılmaktadır. Aksi halde, hesaplanan zamansal fark değeri devir süresinden büyük ise eski verinin güncel olmadığı kabul edilmektedir ve değeri 1 olarak güncellenmektedir. Bu sayede hem eski verinin silinmesi hem de yeni verinin yazılması tek bir yazma işlemi ile gerçekleştirilir. BF dizisindeki verilerin güncelliğinin kontrol edilmesi için herhangi bir ilave kontrol mekanizmasına gerek yoktur.

Başka bir deyişle, önerilen model veri güncelliğinin doğrulanmasını zaman damgasına bağlı olarak gerçekleştirir. Önerilen yöntem hareketli bir zaman penceresi olarak düşünülürse devir süresini zaman penceresinin genişliği olarak kabul edebiliriz. Bu sayece tamamen dolu olan bir BF dizisi bir devir süresi geçtikten sonra tamamen boş kabul edilebilir, nitekim bir devir süresi sonrasında üzerindeki bütün veriler güncelliğini yitirmiş hale gelir. Bu aynı zamanda zamansal fark kontrolünde herhangi bir başlangıç veya bitiş zamanı olmadığı anlamına da gelmektedir, t anında gelen bir veri için bir devir süresi öncesine kadar olan veriler güncel kabul edilmektedir.

Önerilen modelin doğruluğunun artırılması için uygulama alanına özgü devir süresi belirlenmesi gerekir. Örneğin, herhangi bir web uygulamasına yapılan aktif bağlantıların takibi için önerilen yöntemin kullanılması durumunda ortalama bağlantı süresinin ölçülmesi ve buna uygun bir devir süresinin belirlenmesi gerekliliği söz konusudur. Gerçekleştirilen testlerde çeşitli web sayfaları için ortalama bağlantı süresi 57 milisaniye olarak ölçülmüştür. Bu durumda bu değerin üzerinde bir devir süresi belirlenmelidir. Bu değer kullanılan web sayfasının aktardığı veri boyutuna, kullandığı metotlara ve yapısına göre değişkenlik gösterebilir.

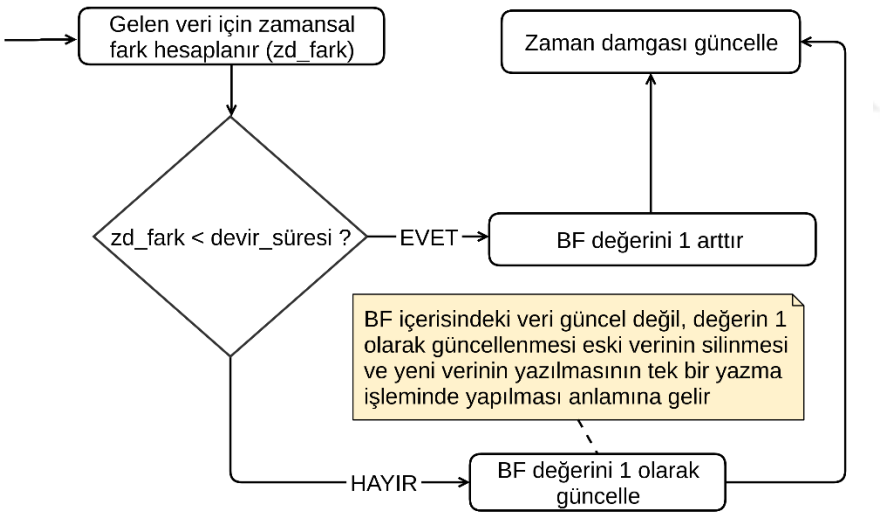
İkinci bir örnek olarak gelen SYN isteklerini durumsal olarak takip etmek ve SYN saldırı paketlerini tespit etmek için önerilen yöntemi kullandığımızı varsayalım. SYN saldırıları TCP bağlantı kurulma aşamasında TCP üçlü el sıkışma mekanizmasını sömürdüğünden, önerilen veri yapısını sadece bağlantı kurulum aşamasında SYN paketlerinin durumsal takibi yapılarak doğrulanmasında kullanılabilir. Bu durumda, çalışma esnasında ortalama üçlü el sıkışma süresi ölçülerek buna uygun bir devir süresi belirlenmesi gerekir. Gerçekleştirilen

testler esnasında çeşitli açık web sayfalarında herhangi bir SYN koruması kullanılmadığı durumlarda ortalama üçlü el sıkışma süresinin 13ms olduğu ölçülmüştür. Bu durumda bunun üzerinde bir değer olarak 15 ms devir süresi kullanımı uygun olabilir.

Önerilen GBF veri yapısına veri ekleme, silme ve sorgulama için akış diyagramları Şekil 3.5, Şekil 3.6 ve Şekil 3.7'de yer almaktadır. Güncelliğini yitirmiş verilerin veri yapısından silinmesi iki farklı durumda gerçekleşir.

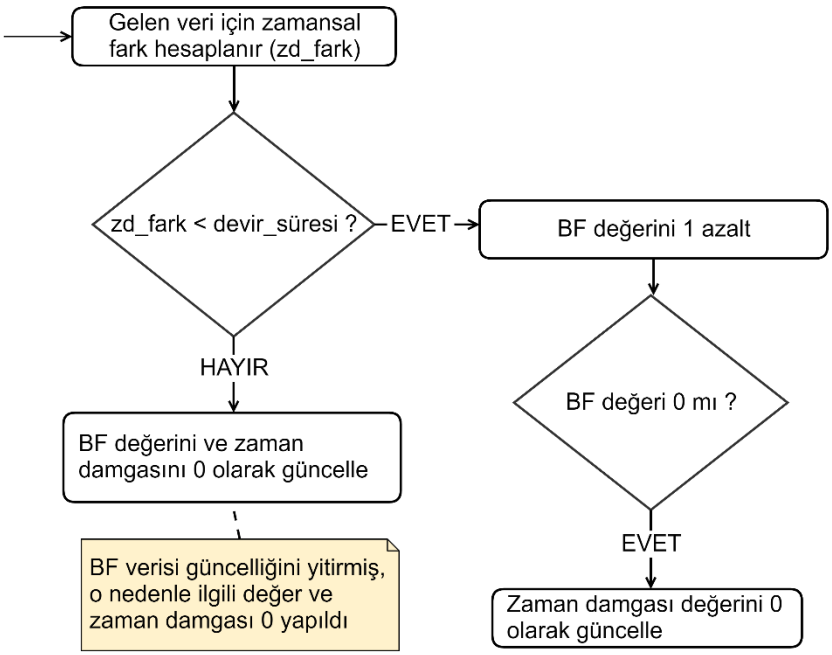
- Durum-1: Veri ekleme esnasında herhangi bir BF dizisi üzerinde çakışma oluşursa eski verinin devir süresine bağlı olarak güncelliği kontrol edilir ve şayet güncelliğini yitirmiş ise yeni veri üzerine yazılarak eski veri silinir.

- Durum-2: Kullanılan uygulamadan güncelliğini yitirmiş verilerin bilgisi edinilebilirse, BF dizilerinden ilgili veriler silinebilir.



Şekil 3.5. Durum-1 için veri ekleme ve silme akış diyagramı

Örneğin aktif bağlantıların takibinde GBF veri yapısının kullanılması durumunda gelen SYN paketi ile beraber veri GBF veri yapısına eklenebilir. Devamında aynı bağlantı için bağlantının sonlandırılması amacıyla gönderilen FIN paketi algılandığında eklenen veri GBF veri yapısından silinebilir.



Şekil 3.6. Durum-2 için veri silme akış diyagramı

GBF dizi genişliğinin 2-bit olarak kullanılması güncellenebilir olmasına imkan tanır. Farklı bir örnek olarak, TCP üçlü el sıkışma paketlerinin SYN saldırılarına karşı durumsal takip edilmesi durumunda gelen SYN paketleri ile beraber veri GBF veri yapısına yazılabilir. Fakat bu durumda verilerin güncellik kontrolü sadece devir süresi kullanılarak yapılabilir. Nitekim SYN saldırılarında bağlantı sonlandırılmasına yönelik herhangi bir paket gönderimi söz konusu değildir.

3.2.1. Güncellenebilir Bloom filtresi için depolama gereksinimleri

Modern programlanabilir anahtarlar paket anahtarlama kapasitesi (25.6 Tbps [87]) ve SRAM boyutu (50-100MB [27]) olarak son yıllarda önemli ölçüde ilerleme kaydetmiştir. Yine de depolama alanı boyutları MB seviyelerinde olduğundan alan etkin algoritmaların kullanılması elzemdir. Önerilen GBF veri yapısı için gereken depolama alanı aşağıdaki eşitlik 1' de görüldüğü gibi hesaplanabilir. Kullanılacak zaman damgası kaynağı esasen bir sayaçtır ve zaman damgası değeri bu sayaç için anlık ölçülen değerdir. Bu nedenle kullanılacak zaman damgası kaynağı için iki önemli parametreden söz edilebilir, bunlar zamansal çözünürlük *(time resolution*) ve başa dönme süresidir (*wrap-around time*). Önerilen veri yapısında zaman damgası bilgisi toplam depolama alanına önemli bir katkı

sağladığından ve gerekli alanı doğrudan etkilediğinden, bu parametrelerin dikkatli bir şekilde belirtilmesi önemlidir.

$$Gereken\ depolama\ alanı = m \text{ x } w \text{ x } k \quad (1)$$

m : Bloom filtre dizisi uzunluğu

w : Toplam bit genişliği (BF + Zaman Damgası)

k : Özetleme fonksiyonu sayısı

Gerçekleştirilen bu çalışmada zaman damgası kaynağı olarak bmv2 [88] sanal anahtarının sağlamış olduğu *global_ingress_timestamp* [89] kullanılmıştır. Bu kaynak esasen 48 bit genişliğinde ve 1µs çözünürlüğe sahip bir sayaç olarak çalışmaktadır. Gerçekleştirilen çalışmada 48 bit uzunluğundaki bu değer için değişken tip dönüşümü (*type-casting*) gerçekleştirilmeye çalışılarak 24 bit uzunluğuna düşürülmeye çalışılmıştır. Nitekim bu sayede gereken depolama alanı önemli ölçüde azaltılabilir. Fakat bmv2 anahtarın varsayılan olarak 48 bit olarak bu değeri sağlaması sebebiyle ulaşan her paket için tip dönüşümünün tekrar edilmesi durumunda sanal anahtarın performansının olumsuz etkilendiği anlaşıldığından bu değer 48 bit olarak kullanılmıştır. Bunun sonucunda aşağıdaki Çizelge 3.2' de görüldüğü üzere önerilen veri yapısının 50 KB depolama alanı gereksinimi hesaplanmıştır.

Çizelge 3.2. Zaman damgası uzunluğu ile depolama alanının değişimi (m=4096, k=2)

Zaman Damgası Uzunluğu	**Zamansal Çözünürlük**	**Başa Dönme Süresi (Saniye)**	**Gereken Depolama Alanı**
48 bit	1 μs	~ 281 000 000	50 KB
32 bit	1 *ms*	~ 4,2 000 000	34 KB
24 bit	1 *ms*	16 777	26 KB
16 bit	1 *ms*	65	18 KB

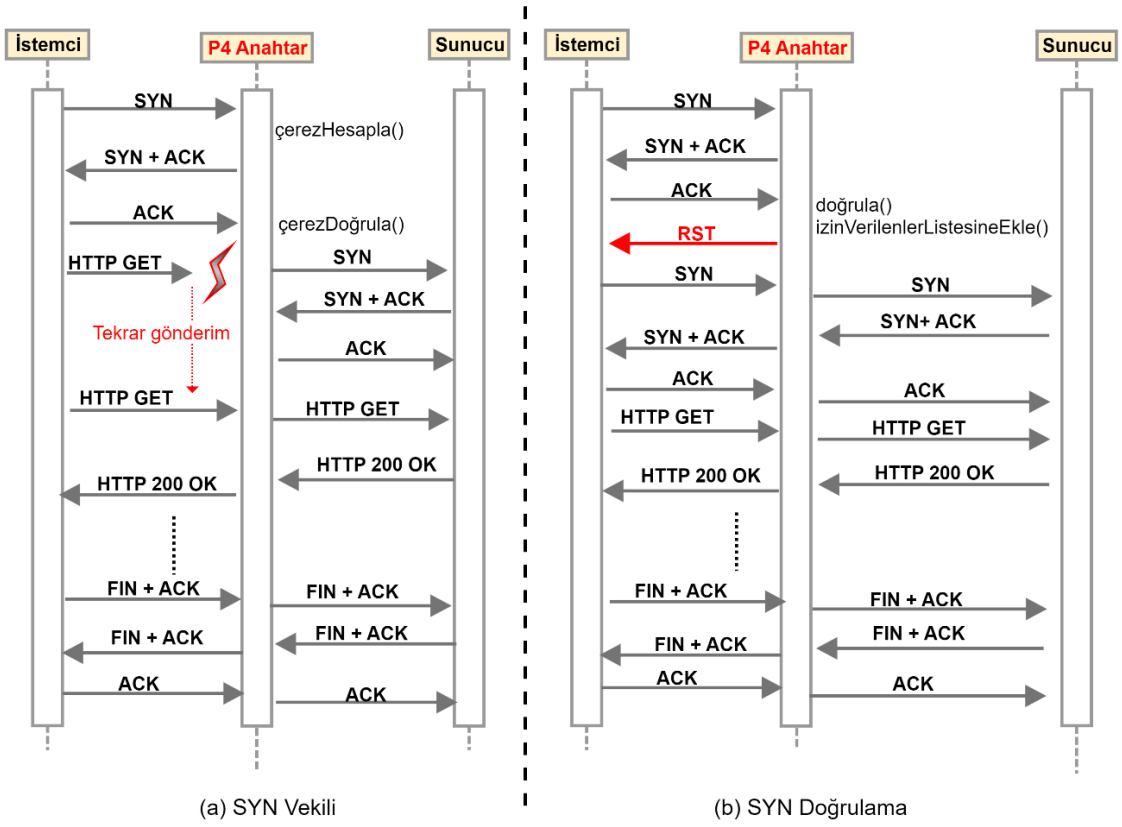
Gerçekleştirilen çalışmada standart 48-bit uzunluğunda zaman damgası kullanılması sebebiyle depolama alanı nispeten yüksek çıkmıştır. Esasen birçok ağ donanımları hassas zaman protokolü (*Precision Time Protocol, PTP* [90]) ve benzeri zaman protokollerini destekler. Bunun yanı sıra donanım üreticileri faklı çözünürlükte sayaç fonksiyonlarını kullanıma sunabilir. Örneğin Çizelge 3.2' de görüldüğü üzere 1ms çözünürlük değerine sahip ve 24 bit uzunluğunda bir sayaç önerilen modelin gereksinim duyduğu depolama alanının 50KB değerinden 26KB değerine dramatik olarak azalmasını sağlar.

Başa dönme süresi ile ilgili olarak, önerilen yaklaşım zamansal fark değerini iki zaman damgası arasında $t_2 - t_1$ şeklinde çıkarma işlemi yaparak hesaplar. $P4_{16}$ programlama dili doyuma ulaşan aritmetik (*saturation arithmetic*) işlemlerin gerçekleştirilebilmesini desteklemektedir [91]. Kullanılan zaman damgası kaynağı t_1 ile t_2 arasındaki bir anda maksimum değerine ulaşır ve tekrar 0 değerinden başlarsa t_2 değeri t_1 den daha düşük değerli olur. Fakat yine de $t_2 - t_1$ çıkarma işleminin sonucunda oluşan fark değeri pozitif değerli gerçek zamansal fark değerini verir. Bu durum başa dönme süresinde ciddi oranda bir azalma yapılabilmesine olanak tanır. Burada başa dönme süresi kullanılacak uygulamanın gerektirdiği maksimum süre değerinden daha küçük olamaz. Örneğin TCP üçlü el sıkışma zaman aşımı 30 saniye olarak kabul edilirse, Çizelge 3.2' de görülen 16 bit zaman damgası sayacı 65 saniye başa dönme süresini desteklediğinden kullanılabilir.

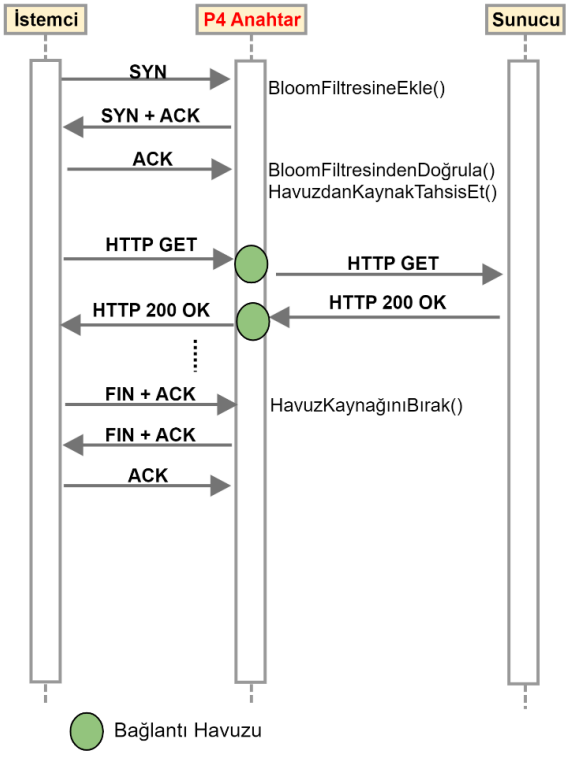
4. SYN SALDIRILARINA KARŞI P4 TEMELLİ SAVUNMA

Programlanabilir veri katmanında SYN saldırılarına karşı literatürde önerilen savunma yöntemleri incelendiğinde bunların tipik olarak iki gruba ayrıldığı söylenebilir. Bunlar SYN doğrulama (*SYN Authentication*) ve çerez temelli SYN vekili (*cookie-based SYN Proxy*) yaklaşımı olarak sınıflandırılabilir [33,34]. Her iki yaklaşımda da sunucu tarafından TCP başlıklarında yer alan sıra numarasının (*TCP Sequence Number*) ilk değerinin belirlenmesinde özel yöntemler kullanılarak çerez doğrulaması gerçekleştiren yaklaşım kullanılabilir, fakat bu yöntemler doğrulama sonrasındaki işlemlerde farklılık gösterir. SYN vekili yaklaşımında programlanabilir veri katmanı sunucu ve istemci arasında bir vekil sunucu gibi davranır. Bu bağlamda istemci ve P4 anahtar arasındaki TCP üçlü el sıkışma aşaması tamamlanıp doğrulandıktan sonra P4 anahtar arkasındaki sunucuya başka bir TCP bağlantısı başlatır. Bu nedenle TCP bağlantı kurulum süresi artar. Ayrıca istemciden üçlü el sıkışmadan hemen sonra gönderilen ACK paketi P4 anahtar ve arka uç sunucu arasındaki henüz tamamlanmamış TCP bağlantı kurulumundan ötürü düşürülür. Sonuç olarak uygulama seviyesinde tetiklenen bir zaman aşımı sebebiyle istemcinin bu paketi tekrar göndermesi beklenir. SYN doğrulamasında ise bağlantı doğrulandıktan sonra istemci IP adresi izin verilenler listesine eklenir ve ardından bağlantının sıfırlanarak yeniden kurulması için P4 anahtardan istemciye RST paketi gönderilir. Fakat bu davranış kullanılan yöntemin saldırganlar tarafından kolaylıkla tespit edilebilmesine neden olur. Üstelik bir istemciyi izin verilenler listesine eklemek, SYN doğrulama yaklaşımının kolaylıkla atlatılabilir hale gelmesine neden olur. Ayrıca normal istemciler açısından bakıldığında bu durum her TCP bağlantı isteği için gönderilen paket sayısını üçten yediye çıkararak ağ yükünde genel bir paket artışına neden olur. Her iki yaklaşımının gösterimi Şekil 4.1'de yer almaktadır.

Önerilen GBF ve bağlantı havuzu temelli SYN savunma yaklaşımı Şekil 4.2' de görülmektedir. Önerilen yöntemde TCP bağlantı kurulumu aşamasındaki üçlü el sıkışma mekanizmasının çalışması bir görev olarak değerlendirilerek bu görev arka uç sunucudan programlanabilir veri katmanına aktarılmıştır. Bunun gerçekleştirilmesi sayesinde SYN doğrulaması sonrasında herhangi bir ek paket gönderimlerine ihtiyaç duyulmaz. Başka bir deyişle, önerilen yaklaşım sayesinde savunma mekanizması sebebiyle ağ üzerinde ilave paket gönderimine bağlı yük artışının önüne geçilir. Üstelik bu sayede istemci açısından önerilen savunma yönteminin tespit edilebilmesi de söz konusu olmaz.



Şekil 4.1. Çerez temelli SYN vekili ve SYN doğrulama yaklaşımlarının görselleştirilmesi



Şekil 4. 2. Önerilen GBF ve bağlantı havuzu temelli SYN savunma

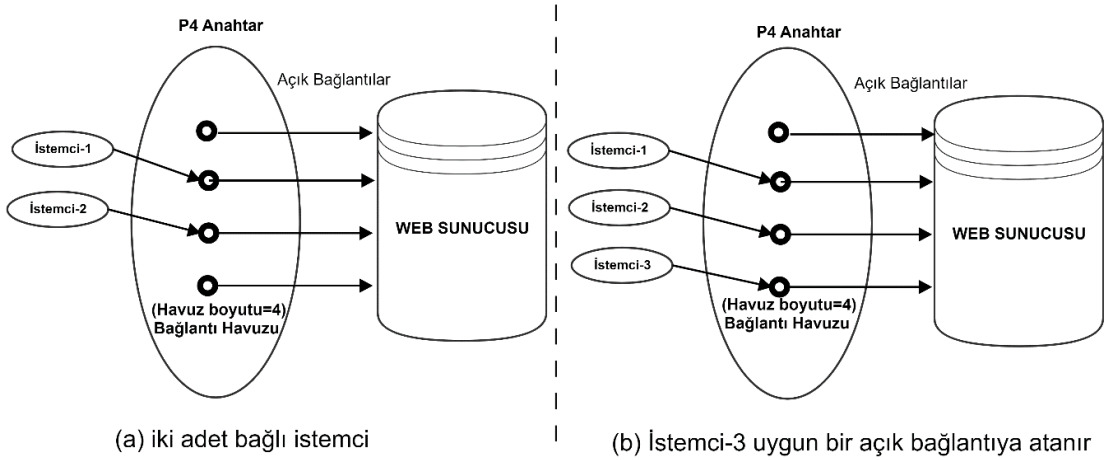
SYN istekleri için üçlü el sıkışmanın durumsal takibinde düşük depolama alanı gerektiren GBF veri yapısı kullanılmış ve sağladığı zamansal veri güncellik kontrolü sayesinde yapılan isteklerin doğrulanması yüksek doğrulukla gerçekleştirilmiştir. Ayrıca, doğrulanmış bir IP adresinin izin verilenler listesine eklenmesi gibi bir durum da söz konusu olmadığından aynı IP adresinden yapılan her bir istek eşsiz olarak değerlendirilir ve ayrı ayrı doğrulanır.

Bu bölümde gerçekleştirilen bağlantı havuzu yaklaşımının tasarımı ve GBF ile birlikte kullanımının programlanabilir veri katmanı üzerinde gerçekleştirilmesi detaylı olarak ele alınmıştır.

4.1.Bağlantı Havuzunun Programlanabilir Veri Katmanında Gerçekleştirilmesi

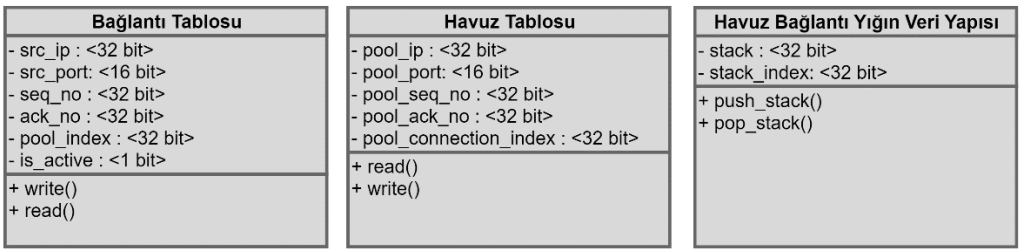
Bağlantı havuzu yaklaşımı belirli sayıda TCP bağlantısının kurulması sonrasında bu bağlantıların bir havuz içerisinde açık tutulması ve veri transferinde bu açık ve kullanıma hazır bağlantıların kullanılmasının amaçlandığı bir yaklaşımdır. TCP bağlantı kurulumu ve sonlandırmasının oluşturduğu ilave paketlerin ve bundan ötürü oluşabilecek gecikmelerin önlenmesi amacıyla kullanılan bir yaklaşımdır [92,93]. Şekil 4.3'de görülen bağlantı havuzu örneğinde P4 anahtar üzerinde 4 adet aktif bağlantı P4 anahtardan web sunucusuna doğru oluşturulmuştur. Web sunucusuna erişim isteği yapan iki adet istemci tarafından oluşturulan bağlantı havuzu üzerindeki iki adet açık bağlantı kullanılırken daha sonrasında gelen istemci-3 için uygun bir hazır bağlantının atandığı görülmektedir.

Esasen burada gerçekleştirilen yaklaşım ile istemci ve web sunucusu arasında gerçekleşecek bağlantılar için TCP üçlü el sıkışma işleminin web sunucusundan P4 anahtara aktarımı (*offloading*) söz konusudur. Nitekim, bağlantı havuzu hazırlandıktan sonraki bütün istekler için TCP üçlü el sıkışma mekanizması ile gerçekleşen bağlantı kurma ve sonlandırma için kullanılan bütün paketler sadece istemci ve P4 anahtar arasında gerçekleşir. Bu sayede istemcinin yaptığı istekler üzerinde gerekli kontroller ve doğrulamalar yapıldıktan sonra (SYN doğrulama vb.), bağlantı havuzundaki uygun bir açık bağlantı istemciye tahsis edilerek veri gönderimi için kullanılabilir.



Şekil 4.3. Programlanabilir veri katmanında bağlantı havuzu

Bağlantı havuzunun programlanabilir veri katmanında kullanılması SYN doğrulama sebebiyle oluşacak ilave paket sayısını azalttığı için ağ performansına olumlu katkı sağlar. Bunun gerçekleştirilmesi için doğrulanan her bir istemci isteği havuz üzerinde uygun bir açık bağlantıya atanır ve TCP bağlantılarının P4 anahtar üzerinde durumsal olarak takibi gerçekleştirilir. Üstelik burada bahsedilen durumsal takip sadece kaynak ve hedef IP adresi ve port numaraları ile sınırlı değildir. Buradaki durumsal takip P4 anahtarın aynı zamanda TCP katmanında kullanılan sıra numarası (*Sequence Number, seq_No*) ve onay numarası (*Acknowledgement Number, ack_No*) değerlerini de takip etmesini gerektirmektedir. Bu durum P4 anahtar üzerinde TCP paket başlık değerlerinin mevcut açık bağlantılara uygun olarak dinamik ve çift yönlü olarak güncellenmesini gerektirmektedir.



Şekil 4.4. P4 yazmaçlarda oluşturulan bağlantı havuzu veri yapısı

Gerçekleştirilen bu çalışmada Şekil 4.4'te görülen veri yapısı P4 tarafından sağlanan yazmaçlar üzerinde oluşturularak bağlantı havuzu programlanabilir veri katmanında gerçekleştirilmiştir. Bu veri yapısı, aktif bağlantı ve havuzdaki açık bağlantı bilgilerinin yazmaçlarda depolanmasına olanak tanıyarak durumsal paket işlemeyi mümkün hale

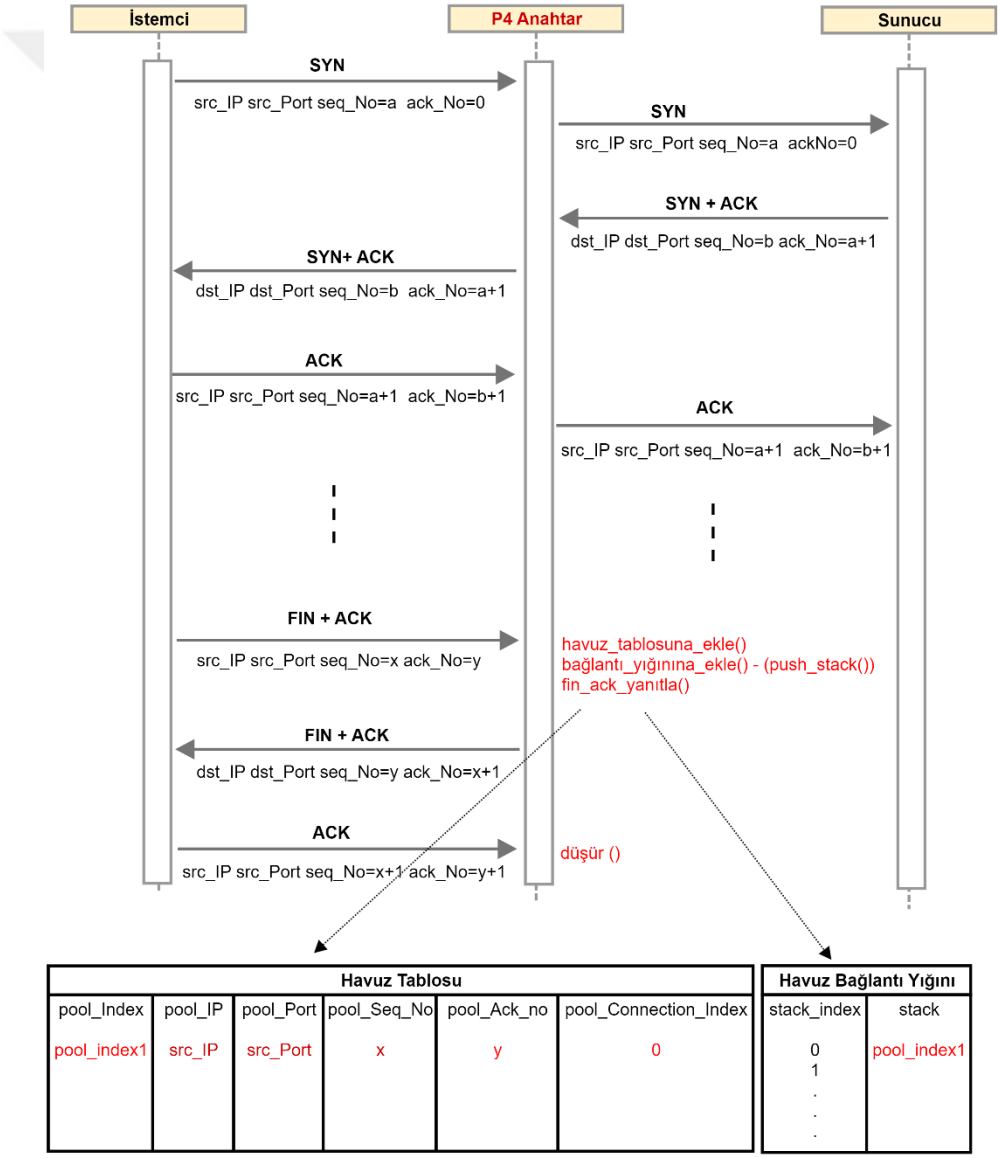
getirmektedir. P4 programlama dili yazmaçlar üzerinde arama işlemini doğrudan desteklemediğinden, bu veri yapısına ilişkin indeks değerlerini elde etmek için CRC16 özetleme algoritması kullanılmıştır. Böylece her bir akış için kaynak IP adresi (*Src_IP*), hedef IP adresi (*Dst_IP*), kaynak port numarası (*Src_Port*) hedef port numarası (*Dst_Port*) ve protokol olmak üzere 5 adet giriş verisi kullanılarak yeni gelen paketlerin hangi akışa ait olduğu bilgisinin takip edilmesine olanak sağlamaktadır.

Bağlantı havuzunda oluşturulan açık bağlantılar birer kaynak olarak düşünüldüğünde bu kaynakların her bir yeni bağlantıya dinamik olarak tahsis edilmesi gerekmektedir. Devamında ilgili bağlantı sonlandığında bu kaynağın tekrar havuza aktarılması gerekmektedir. Bu amaçla kaynak tahsisi ve yönetimi için yığın veri yapısı P4 programında gerçekleştirilmiştir. Her ne kadar P4 programlama dili yığın veri yapısını destekliyor olsa da bu durum sadece paket başlık bilgilerinin kullanımı ile sınırlıdır [94]. Bu sebeple P4 programlamanın sunduğu yazmaçlar üzerinde yığın veri yapısı oluşturularak, yığına veri eklemek ve çıkarmak için gerekli metotlar (*pop, push*) bağlantı havuzunda kaynak yönetimi için oluşturulmuştur.

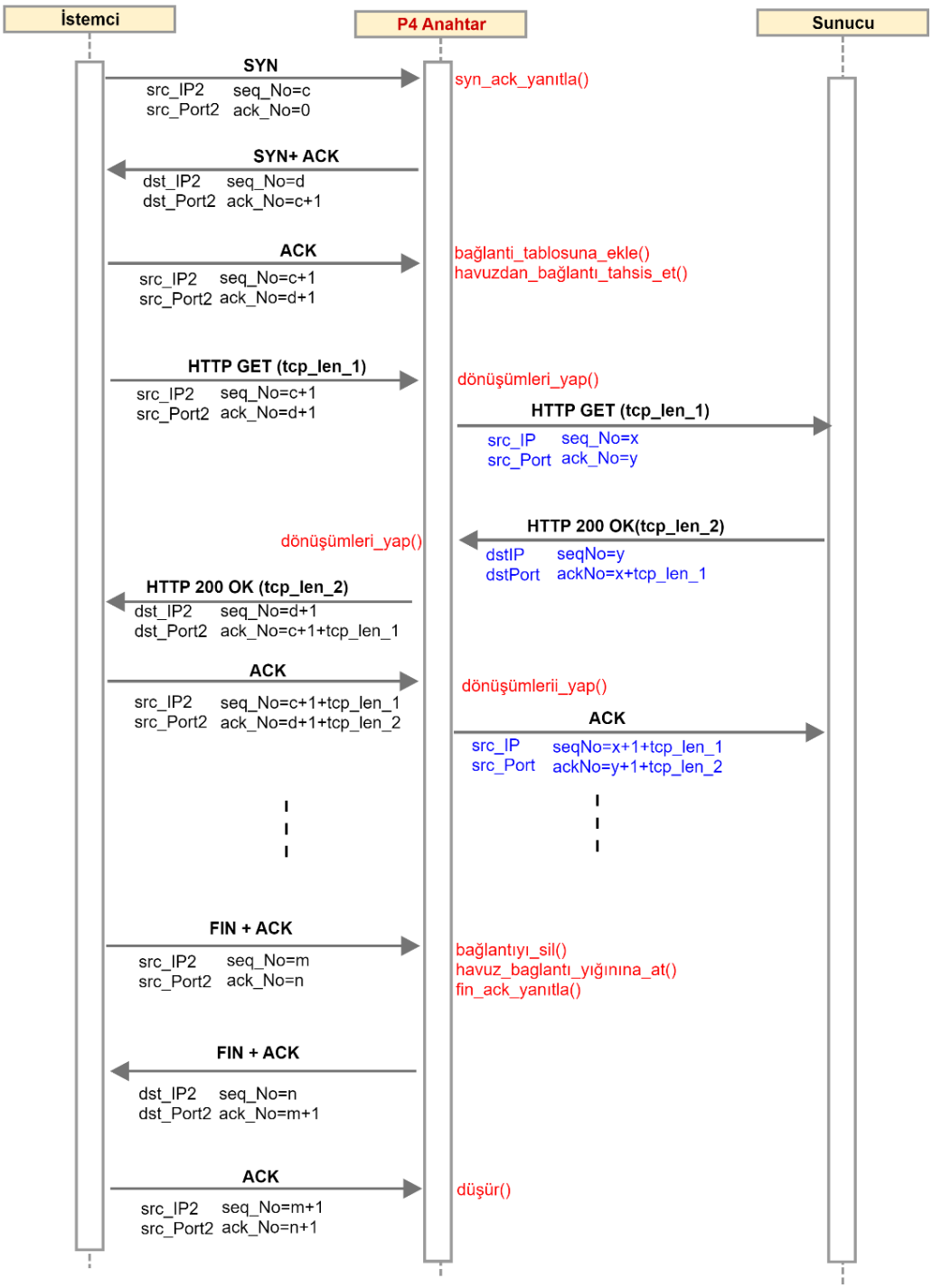
Bağlantı havuzunun gerçekleştirilmesi iki fazdan oluşmaktadır. Bunlardan ilki Şekil 4.5'te görüldüğü üzere bağlantı havuzunun hazırlanması aşamasıdır. Bunun gerçekleştirilmesi için gerekli sayıda istek istemci tarafından başlatılarak bu isteklerin P4 anahtardan geçmesi sağlanır. Bağlantının sonlandırılması için istemci tarafından gönderilen FIN+ACK paketi ile beraber P4 anahtar bu bağlantının verilerini havuz tablosuna ekler. Aynı zamanda bu bağlantının index değerini havuz bağlantı yığını tablosuna ekler ve bu yapılan isteğe karşılık FIN+ACK yanıtını P4 anahtar döner. Devamında aynı bağlantıya ait alınan ACK paketi P4 anahtar üzerinde düşürülür. Bu aşamada bağlantı durumu istemci tarafında sonlanmış olmasına rağmen sunucu üzerinde halen aktif durumdadır. Ayrıca bağlantı bilgileri P4 anahtar üzerinde kullanılmak üzere tutulmaktadır. Benzer şekilde istenen sayıda bağlantı P4 anahtar üzerinde oluşturularak bağlantı havuzunda tutulabilir.

Oluşturulan açık bağlantılar P4 anahtar üzerindeki oluşturulan havuzda beklediği süre boyunca veri transferinde kullanılmadığından bu bağlantıların sunucu tarafından sonlandırılmaması gerekmektedir. Fakat web sunucuları bu şekilde boş bekleyen açık bağlantıları sonlandırma eğilimindedir [95]. Bunun önlenmesi amacıyla kullanılan Apache web uygulamasında *keep_alive* parametresinin değerinin artırılması gerekmektedir. Ayrıca

paketlerin TCP başlık bilgilerinde yer alan zaman damgası (*tcp.options.timestamp*) değeri istemci ve sunucu arasında paket varış sürelerinin hesaplanmasında kullanılan bir değerdir. Fakat hem istemci hem de sunucunun bunu desteklemesi durumunda gönderilen bir paket içerisindeki zaman damgasının alınan yanıt paketinde bulunmasını gerekli kılmaktadır. Aksi halde paketlerin tekrar gönderiminin talep edilmesi söz konusu olmaktadır [96]. Bunun önlenmesi ve önerilen yaklaşımın yapısının olabildiğince basit tutulması amacıyla sunucu tarafında TCP zaman damgası (*tcp_timestamps*) özelliği pasif hale getirilmiştir. Bunun neticesinde P4 anahtarın TCP bağlantılarının sonlandırılması sürecini tam olarak yönetebilmesi sağlanmış olur.



Şekil 4.5. Bağlantı havuzunun hazırlanması



Şekil 4.6. Bağlantı havuzunun kullanılması

Bağlantı havuzunun ikinci aşaması Şekil 4.6'da görüldüğü üzere bağlantı havuzunun kullanım durumudur. Bu aşamada yapılacak bütün istekler için TCP üçlü el sıkışma P4 anahtar ve istemci arasında gerçekleşmektedir. Üçlü el sıkışmanın tamamlandığı bir istek için bağlantı havuzundan kaynak tahsisi yapılarak bağlantı süresince bu kaynağın kullanımı

gerçekleştirilir. Bağlantı süresince istemcinin kaynak IP adresi, kaynak port numarası, sıra numarası ve onay numarası değerleri çift yönlü olarak dönüştürülür.

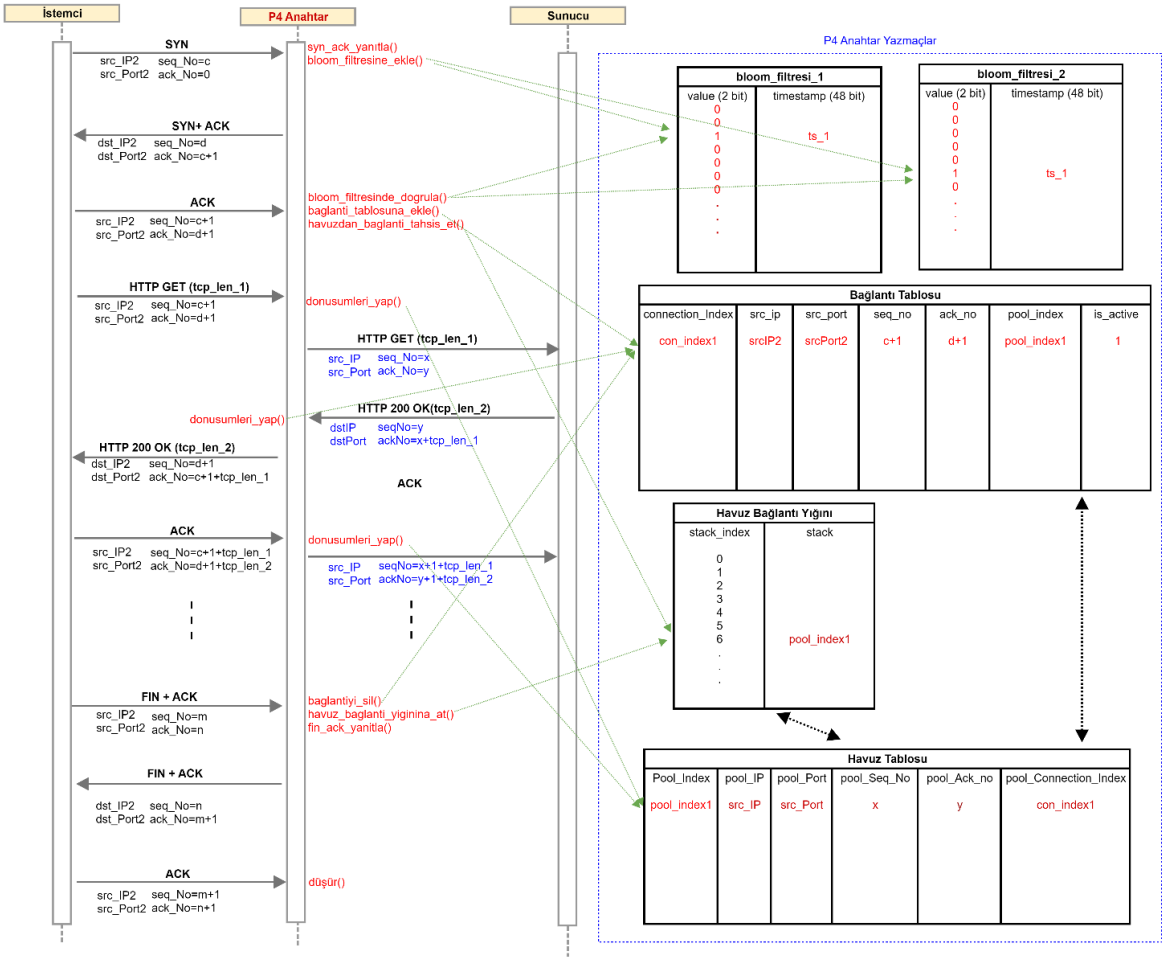
Örneğin Şekil 4.6'da görüldüğü üzere istemcinin kaynak IP adresi ve kaynak port numarası sırasıyla *src_IP2* ve *src_Port2* olarak görülürken bu değerler sunucuya iletilirken havuzdaki hazır bağlantının sahip olduğu *src_IP* ve *src_Port* değerlerine dönüştürülmüştür. Ayrıca *seq_No* ve *ack_No* değerleri her bir paket içerisinde sürekli olarak takip edilerek değişmektedir. Nitekim bu paket başlık değerlerini TCP protokolü gelen verilerin sıralanması, eksik verilerin tespiti, alınan verinin doğrulanması ve akış kontrolü için kullanmaktadır [97,98]. Bunun gerçekleştirilmesi için P4 anahtar hem istemci hem de sunucu tarafındaki kullanılan değeri kendi yazmaçlarında tutar ve yeni gelen paketler ile önceki paketler arasındaki farksal artışı hesaplayarak dönüşümlerde bu farksal artışı kullanır. Örneğin Şekil 4.6'da istemci tarafından gönderilen *HTTP GET* istek paketi *seq_No=c+1* ve *ack_No=d+1* değerlerine ve TCP paket boyutu (*TCP.length*) değeri *tcp_len_1* olan paketi göndermiştir. Bu paket sunucuya iletilirken bağlantı havuzundaki atanan bağlantının değerleri olan *seq_No=x* ve *ack_No=y* olarak dönüştürülerek gönderilir. Devamında sunucu tarafından gönderilen yanıt paketinde paket boyutuna bağlı olarak *ack_No=(x+tcp_len_1)* olarak gönderilir. P4 anahtar bu aşamada önceki değer ile yeni gelen paket arasındaki farksal artışın *tcp_len_1* kadar olduğunu hesaplar ve bunun sonucunda yanıt paketini istemciye gönderirken onay numarası olarak *ack_No=(c+1+ tcp_len_1)* olarak gönderir. Sürekli ve çift taraflı olarak sorgu numarası ve onay numarası değerleri arasındaki farksal artışlar P4 anahtar üzerinde bütün istemciler için takip edilerek bağlantı havuzunun kullanılması mümkün hale gelmiş olur.

Önerilen bağlantı havuzunun depolama gereksinimleri düşünüldüğünde 4096 aktif bağlantı için P4 donanım üzerinde 200KB değerinden daha düşük bir depolama alanına ihtiyaç duyulmaktadır. Buradaki aktif bağlantı değeri kullanılan uygulamanın gereksinimleri göz önünde bulundurularak artırılabilir.

4.2. SYN Saldırılarına Karşı GBF Temelli Savunmanın Gerçekleştirilmesi

Güncellenebilir bloom filtresi ve bağlantı havuzu yaklaşımlarının programlanabilir veri katmanında gerçekleştirilmesi, bu iki yöntemin birleştirilmesiyle elde edilen ve Şekil 4.7'te görülen SYN savunma yönteminin gerçekleştirilmesine olanak sağlamıştır. Gelen her bir

SYN isteğinin durumu Bloom filtrelerinde saklanır ve SYN+ACK yanıtı P4 anahtar tarafından gönderilir. Bunun devamında gelen ACK paketlerinin doğruluğu zaman bazlı güncellik kontrolü ile sağlanarak doğrulanan istekler için bağlantı havuzundan kaynak tahsisi yapılır.



Şekil 4.7. SYN saldırılarına karşı bağlantı havuzu ve GBF temelli savunma

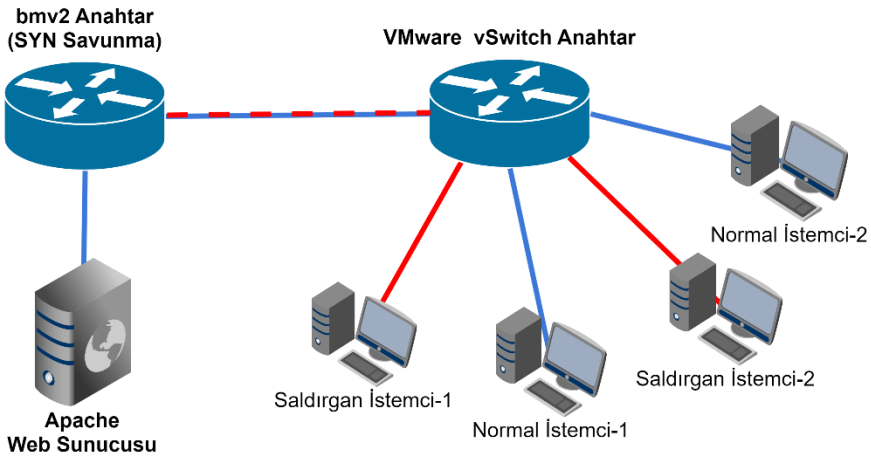
Önerilen yaklaşımda herhangi bir protokol ihlali, bağlantı yeniden kurulma isteği veya paket düşürme gibi bir durum söz konusu olmadığından istemcilerin önerilen savunma yönteminin tespit etmesi söz konusu değildir. Ayrıca, doğrulanmış bir IP adresinin izin verilenler listesine eklenmesi gibi bir durumda söz konusu değildir. Belirli bir IP adresinden yapılan her bir istek benzersiz olarak değerlendirilir ve ayrı ayrı doğrulanır.

Yapılan bazı çalışmalarda olasılıksal veri yapılarının hedef seti kapsama *(target-set coverage*) saldırılarına karşı savunmasız olduğunu gösterilmiş ve bunlar için güvenliğin artırılması amacıyla rasgele girdilerin (*salted inputs*) veya anahtarlı özetleme

fonksiyonlarının kullanımı gibi çeşitli yöntemler önerilmiştir [99-101]. Bu tez kapsamında önerilen yaklaşımda Bloom filtresi kullanılarak doğrulama gerçekleştirilirken giriş verisi olarak 5 parametre (*Src_IP*, *Src_Port, Seq_No*, *Ack_No*, *protocol*) kullanılmaktadır. Burada P4 anahtar tarafından oluşturulan *Seq_No* değeri rasgele giriş verisi olarak değerlendirilebilir. Bunun nedeni, *Seq_No* değerinin, paketin P4 anahtarına mikro saniye cinsinden ölçülen varış süresine göre belirlenmesidir. Bir paketin mikro saniye çözünürlükteki varış zamanı ağdaki anlık birçok değişkenlere bağlı olduğundan, hedef seti kapsama saldırılarına karşı savunma sağlamak için rasgele girdi olarak değerlendirilebilir. Ayrıca GBF veri yapısında herhangi bir nedenle veri yapısının dolması durumu oluşsa bile, bu durumun geçerliliği tek bir devir süresiyle sınırlıdır. Bu nedenle, önerilen yöntemin hedef seti kapsama saldırılarına karşı tasarımı gereği güçlü olduğu söylenebilir.

4.3. SYN Saldırılarına Karşı Savunma Performansının Değerlendirilmesi

Önerilen yaklaşımın etkinliğinin değerlendirilmesi amacıyla Şekil 4.8'da görülen bağlantı topolojisine sahip bir test ortamı oluşturularak çeşitli testler gerçekleştirilmiştir. VMware ESXI (sürüm 6.7.0 U2) doğrudan donanım üzerinde çalışan (*bare metal*) hipervizör [102] olarak kullanılarak 6 adet sanal makine oluşturulmuştur. Önerilen her iki fonksiyonun hem tek başına hem de birlikte çalışma performansları ayrı ayrı değerlendirilmiştir.



Şekil 4.8. SYN savunma test topolojisi

Çizelge 4.1. Test ortamında tahsis edilen kaynaklar

İsim	**CPU Sayısı**	**Hafıza**
Fiziksel Sunucu	8 Intel Core i7 – 7700 (3.6 GHz)	64GB
bmv2 anahtar	8	16GB
Apache Sunucu	1	16GB
Normal İstemci-1	4	16GB
Normal İstemci-2	4	16GB
Saldırgan İstemci-1	4	16GB
Saldırgan İstemci-2	4	16GB

Şekil 4.8'de görüldüğü üzere, VMware vSwitch [103] sanal anahtarına bağlı 2 adet normal ve 2 adet saldırgan istemcinin oluşturduğu trafik bmv2 sanal anahtarına aynı port üzerinden ulaşmaktadır. Ayrıca yapılan testlerde http sunucusu başarım ölçme aracı olarak kullanılan Apache ab [104] ve Apache JMeter [105] araçları ile beraber saldırı isteklerinin oluşturulmasında hping3 [106] aracı kullanılmıştır. Oluşturulan sanal makineler için bellek ve CPU kaynaklarının tahsisi Çizelge 4.1'de ayrıntılı olarak yer almaktadır.

4.3.1. Güncellenebilir Bloom filtresinin performansının değerlendirilmesi

Bağlantı takibinde önerilen güncellenebilir Bloom filtresi veri yapısının doğruluğunun ölçülmesi amacıyla 5 adet (*Src_IP*, *Dst_IP*, *Src_Port*, *Dst_Port*, protokol) parametre kullanılan özetleme fonksiyonuna giriş verisi olarak kullanılmıştır. Hill tarafından [25] çalışmasında kullanılan doğruluk ölçüm yaklaşımı benimsenerek gerçekleştirilen sıralı istekler üzerinden veri yapısı üzerindeki çakışma durumları ölçülmüştür. Bunun neticesinde önerilen veri yapısının sağladığı doğruluk değerleri elde edilmiştir. Buradaki sıralı isteklerden kastedilen yapılan her bir istek için kaynak port numarasının 2 arttırılması ile yeni isteklerin yapılmasıdır. Bu yaklaşım, Bloom filtresinde çakışmaların meydana geldiği andaki davranışlarını inceleyerek veri yapısının doğruluğunun değerlendirilmesine olanak sağlar.

Bağlantı takibinde doğruluğun incelenmesi amacıyla gerçekleştirilen testlerde artan istek sayısı ile beraber doğruluğun değişimi incelenmiştir. Bağlantı isteklerinin başlatılması için Apache JMeter ve Apache ab araçları kullanılmıştır. Gerçekleştirilen testlerde Bloom filtresi boyutu (m) 1024 ve özetleme fonksiyonu sayısı (k) 2 olarak kullanılmıştır. Şekil 4.9'de görüleceği üzere önerilen yaklaşımın doğruluğu hep yüksektir ve doğruluğun zamanla düşmesi gibi bir durum söz konusu değildir. Bunun nedeni önerilen GBF veri yapısının

sadece aktif isteklerin verisini tutuyor olmasıdır. Önceki 3.2 bölümünde ele alındığı üzere verilerin güncelliği iki farklı durumda doğrulanarak güncelliğini yitirmiş veriler veri yapısından silinmektedir. Burada yapılan test için bağlantı takibinde bağlantı sonlanması durumunda istemciden alınan FIN+ACK paketi ile beraber ilgili TCP bağlantı verisi GBF veri yapısından silinmektedir. Ayrıca devir süresi kullanılması herhangi bir sebepten silinemeyen veriler için güncellik kontrolü sağlayarak güncelliğini yitirmiş verilerin veri yapısında kalmasını ve dolayısıyla zamanla doğruluğunun düşmesini engeller.

Hill tarafından programlanabilir veri katmanlarında bağlantı takibinde Bloom filtresi veri yapısının kullanıldığı [25] çalışması en güncel çalışmadır. Çizelge 4.2'de görüldüğü üzere, bu çalışmada 64 000 TCP akışı için m = 408632 boyutunda bloom filtresi ve k = 4 adet özetleme fonksiyonu ile beraber %95 doğruluk sağladığı gösterilmiştir. Fakat devamında gelen istekler için çakışmalar arttığı için doğruluk düşmektedir. Bunun aksine bizim önerdiğimiz yaklaşım ile %99 üzerinde doğruluk değerini 2 adet özetleme fonksiyonu kullanan 1024 boyutunda Bloom filtresi ile sağlamaktadır. Üstelik önerilen zamansal güncellik kontrolü sayesinde bu doğruluk değerinin artan istek sayıları ile beraber zamanla azalması gibi bir durum söz konusu değildir. Önerilen yaklaşım bu açından incelendiğinde çok daha yüksek ve kalıcı doğruluk değerini 399 kat daha küçük Bloom filtresi uzunluğu ile sağlamaktadır.

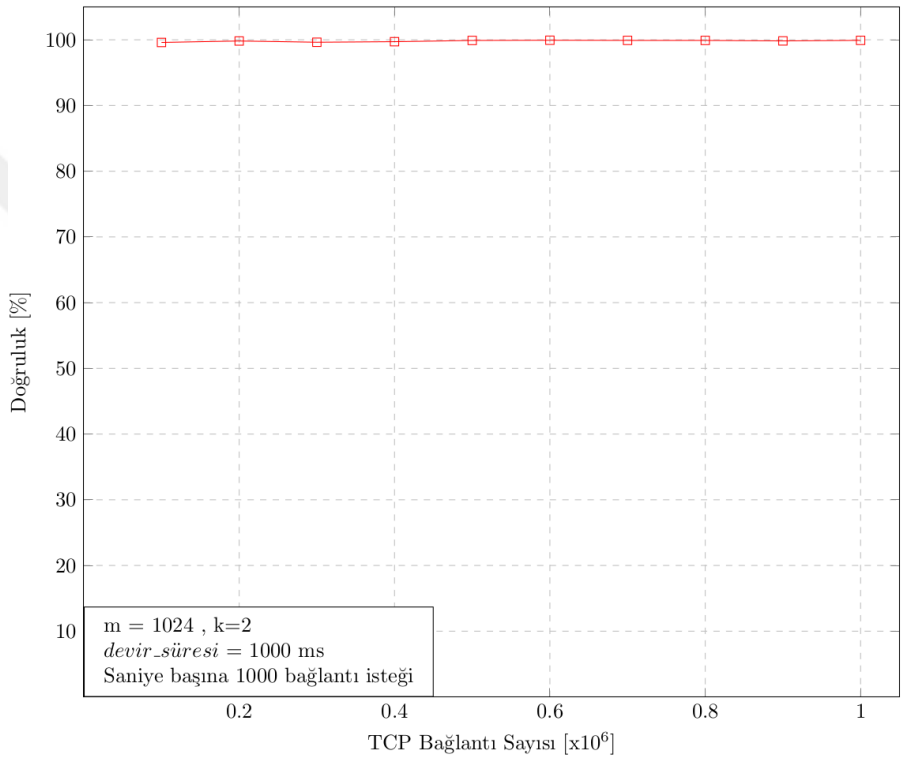
Çizelge 4.2. TCP bağlantı takibinde GBF' nin karşılaştırılması

Çalışma	**BF Uzunluğu (*m*)**	**Fonksiyon Sayısı (*k*)**	**Sağlanan Doğruluk**	**Sağlanan Doğruluk için bağlantı sayısı sınırı**
Hill [25]	408 632	4	%95	64 000
Önerilen GBF	1024	2	>%99	Sınırsız

Devir süresi temelli SYN istek takibinin değerlendirilmesi için Şekil 4.8'da görülen saldırgan istemcilerden hping3 aracı kullanılarak SYN paketleri gönderilirken GBF veri yapısının sağladığı doğruluk değişimi incelenmiştir. Kullanılan özetleme fonksiyonlarının giriş verisi olarak *Src_IP*, *Src_Port*, *Seq_No, Ack_No _Port ve* protokol parametreleri kullanılırken, SYN+ACK yanıtında rasgele *Seq_No* oluşturmak için zaman damgası değeri kullanılmıştır.

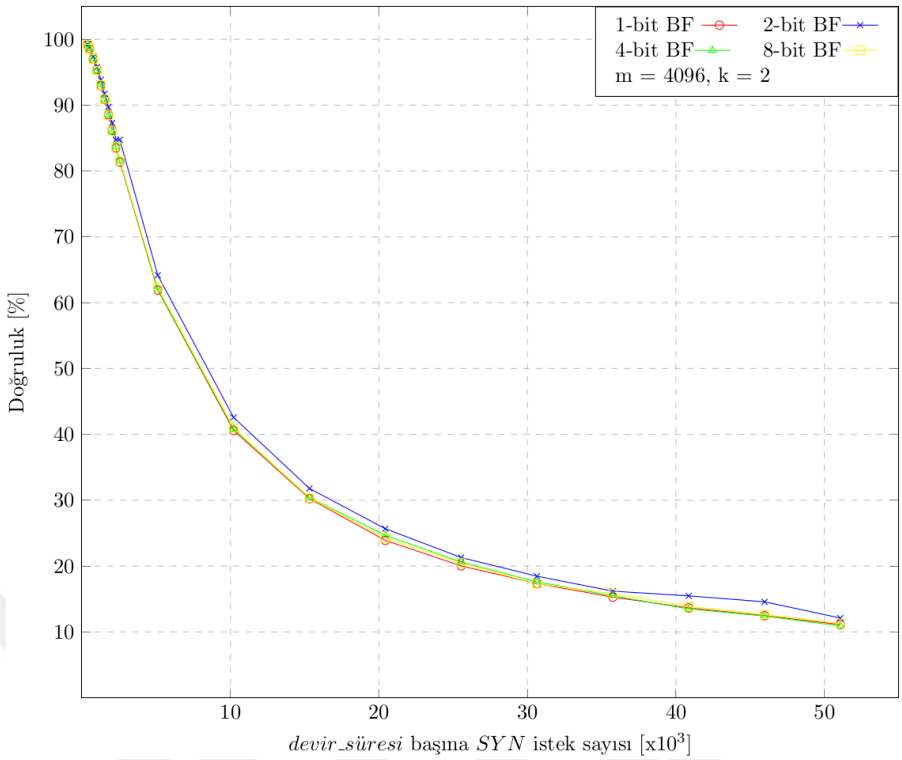
İlk aşamada Bloom filtresi dizi genişliğinin önerilen yöntem üzerindeki etkisini değerlendirilmesi amacıyla SYN isteklerinde takip doğruluğu 4 farklı genişlik değeri için

incelenmiştir. Yapılan testler sonucunda elde edilen sonuçların görüldüğü Şekil 4.10, dizi genişliğinin doğruluk performansına olan etkisinin sınırlı olduğunu göstermektedir. Bunun sebebi, dizi genişliğindeki artışın yalnızca aynı devir süresi içerisinde farklı bağlantı isteklerinden ötürü oluşan çakışma verilerinin depolanmasına imkan vermesi olduğu düşünülmektedir. Sonuç olarak, elde edilen ölçümlerde n=2 genişlik değeri için %1 'den %2'ye kadar değişen bir doğruluk artışının olduğu gözlemlenmiştir. Bu sebeple yapılan testler için önerilen yaklaşımda dizi genişlik değeri 2 olarak kullanılmıştır.

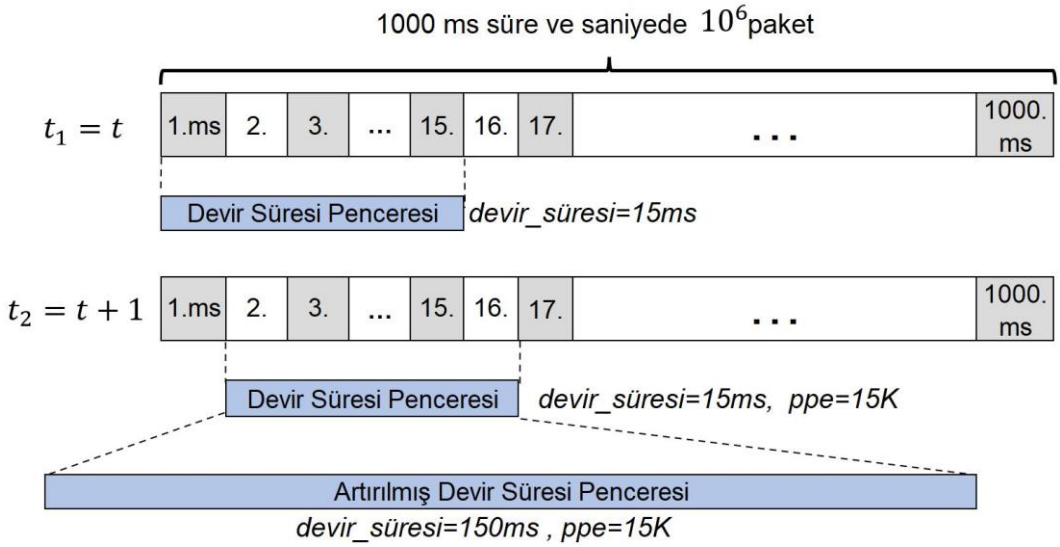


Şekil 4.9. GBF veri yapısının TCP bağlantı takibinde doğruluğu

Önceki 3 numaralı bölümde detaylandırıldığı üzere, uygulamaya özgü Bloom filtresi tasarımı saniye başına paket (*packets per second, pps*) sayısı yaklaşımı yerine devir süresi başına paket *(packet per epoch, ppe)* sayısının ölçülmesini gerektirmektedir. Örneğin hedef paket işleme kapasitesi değerini 1 Mpps olarak belirlenirse, BF veri yapısının 15ms seçilen devir süresi için devir başına 15 000 paket işlemesi gerekmektedir. Bu yaklaşımın görselleştirilmesi Şekil 4.11' da görülmektedir. Devamında yapılan hesaplamalar yer almaktadır.



Şekil 4.10. Çeşitli BF dizi genişlik değerleri için SYN takip doğruluğu



Şekil 4.11. Artırılmış devir süresinin 1Mpps için görselleştirilmesi

Kullanılan sanal anahtar olan bmv2'nin saniye başına paket işleme oranı olarak yüksek oranlarda paket işlemesini engelleyen sınırlı performansına rağmen, devir süresi ve devir süresi başına düşen paket sayıları artırılarak testler gerçekleştirilmiştir. Örneğin Şekil 4.11 ve aşağıda yapılan hesaplamalardan görüldüğü üzere 1 Mpps paket işleme kapasitesinin test

edilebilmesi için devir süresi başına 15 000 isteğin veri yapısında işlenmesi gerekir. Bu amaçla devir süresinin artırılması ile saniye başına paket işleme kapasitesi sınırlı olan bmv2 anahtarının devir süresi başına yüksek oranlarda paket işlemesi gerçekleştirilmiştir. Başka bir deyişle, saniye başına paket oranları yerine önerilen veri yapısı için devir başına paket doğruluk oranları ölçülmüş ve Şekil 4.12' da görülen sonuçlar elde edilmiştir.

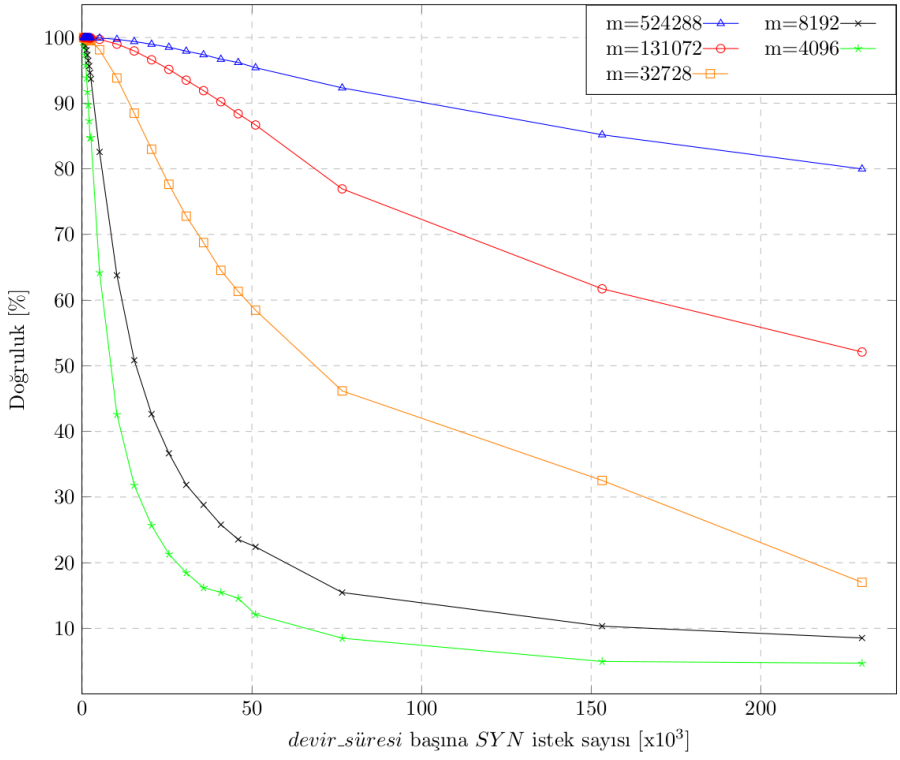
$$Hedef\ Paket\ işleme\ oranı = 1\ Mpps$$

$$devir_süresi = 15\ ms$$

$$Saniye\ başına\ devir\ sayısı = \frac{1000ms}{15ms} = 66{,}66$$

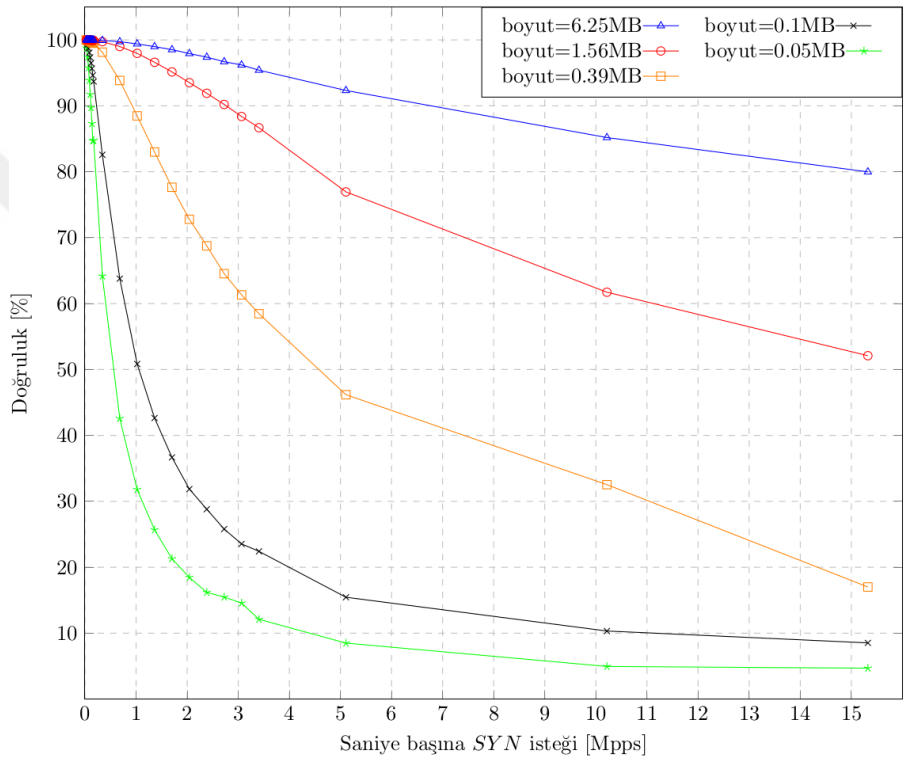
$$Devir\ başına\ paket\ sayısı = \frac{1\ Mpps}{66{,}66} = 15\ 000\ ppe$$

Önerilen GBF veri yapısının 5 farklı dizi uzunluğu boyutu için devir başına doğruluk değişimi Şekil 4.12' da yer almaktadır. Devir başına paket sayısı arttıkça nispeten küçük Bloom filtresi boyutlarında çakışmaların artması sebebiyle veri yapısının doğruluğu azalmaktadır. Ancak daha büyük Bloom filtresi boyutları için doğruluk değerleri kabul edilebilir seviyelerde kalmaktadır.



Şekil 4.12. Çeşitli boyutlarda GBF veri yapısının SYN takibinde devir başına doğruluğu

Yapılan testlerde ortalama TCP üçlü el sıkışma süresi 13 ms olarak ölçülmüştür. Bu sebeple 15 ms olarak kullanılan devir süresi ile beraber Şekil 4.12' da görülen SYN isteklerinin durumsal takibinde devir başına paket doğruluk oranları kullanılarak SYN istekleri için doğruluk değerleri saniye başına paket oranları olarak hesaplanmıştır. Bunun neticesinde Şekil 4.13'de görülen SYN doğruluk değişimi elde edilmiştir. Önerilen modelin yüksek doğruluğu, verilerin güncelliğinin devir süresine dayalı olarak kontrol edilmesi nedeniyle korunmaktadır.



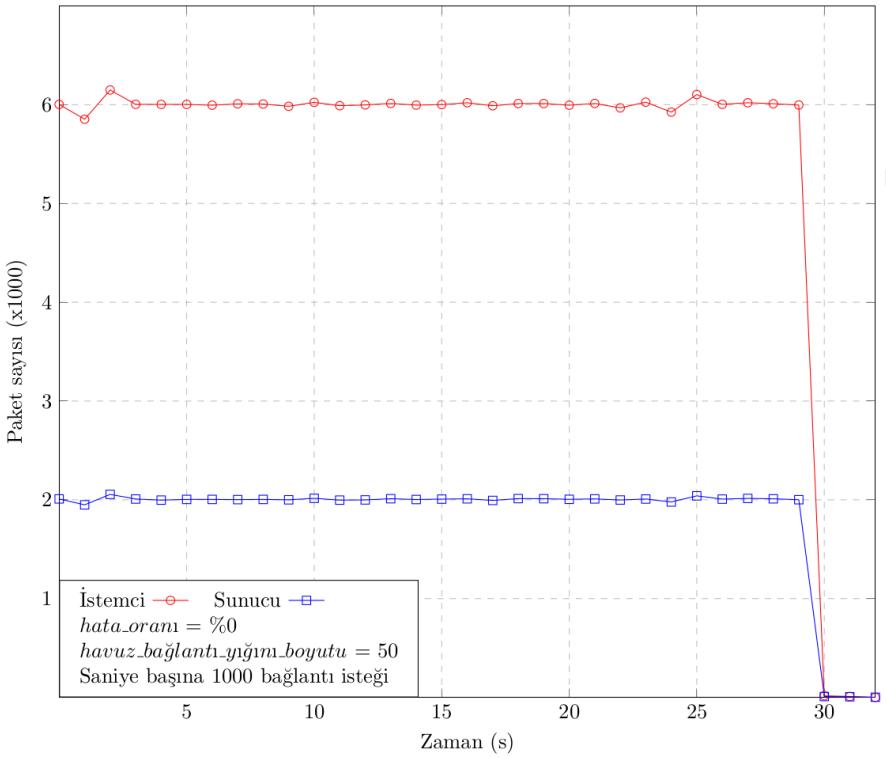
Şekil 4.13. Çeşitli boyutlarda GBF veri yapısının saniye başına paket oranları için doğruluğu

4.3.2. Bağlantı havuzunun performansının değerlendirilmesi

Web sunucusu olarak kullanılan Apache uygulamasında varsayılan değer olarak tanımlı eşzamanlı bağlantı limiti 150 olarak kullanılmıştır. Gerçekleştirilen testler esnasında eşzamanlı bağlantı sayısının 75 değerini aşması durumunda Apache uygulamasının ilk kurulan bağlantılardan başlayarak önceden kurulmuş bağlantıları sonlandırma eğiliminde olduğu görülmüştür. Tasarımımızda desteklenen maksimum eşzamanlı bağlantı değeri doğrudan bağlantı havuzundaki toplam aktif bağlantı sayısını belirlemektedir. Dolayısıyla

gerçekleştirilen testlerde havuz boyutunun 75 değerini geçmemesi gerektiğinden havuz boyutu 50 olarak seçilmiştir.

Bağlantı havuzu kullanılması sayesinde bağlantı kurulumundaki TCP üçlü el sıkışma görevi web sunucusundan programlanabilir veri katmanına aktarılmaktadır. Bunun sonucu olarak her bir istek için web sunucusunun işlemesi gereken paket sayıları azalırken yapılan isteklerde bağlantı havuzundan ötürü hata oluşmaması beklenmektedir. Bağlantı havuzunun etkinliğinin test edilmesi amacıyla Jmeter aracı kullanılarak testler gerçekleştirilerek hata oranı ile beraber sunucu ve istemci tarafındaki paket sayıları incelenmiştir. Yapılan testlerde Şekil 4.14' de görüldüğü üzere 30 saniye boyunca istemcilerden saniyede 1000 istek yapılmıştır. Test sonuçları, bağlantı havuzu kullanımının, bağlantı kurulumu ve sonlandırılması görevlerini %0 hata oranıyla P4 anahtara aktardığı ve sunucunun istek başına işlemesi gereken paket sayılarının önemli ölçüde azalttığını göstermektedir.



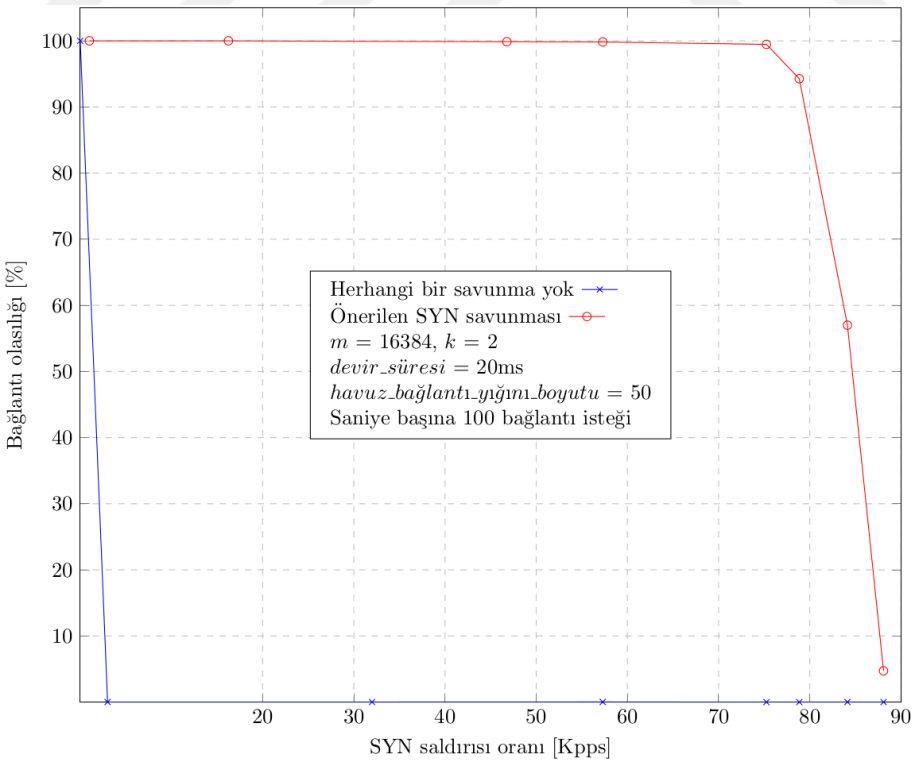
Şekil 4.14. Saniyede 1000 istek için sunucu ve istemci paket oranları

Bağlantı havuzu testlerinde havuz üzerinde 50 adet aktif bağlantı hazırlanmasına rağmen saniye başına 1000 bağlantı için en fazla 11 adet eş zamanlı bağlantının kullanıldığı görüşmüştür. Burada vurgulanması gereken iki önemli durumdan söz edilebilir. Bunlardan birincisi, isteklerin süresi ve eşzamanlı bağlantı gereksinimleri kullanılan uygulamaya göre

önemli ölçüde değişiklik gösterebilir ve bazı uygulamalar daha fazla eş zamanlı istek gereksinimine sahip olabilir. İkincisi ise, testlerimizde havuz bağlantı sayısı olarak kullanılan 50 değeri önerilen bağlantı havuzunun değil, P4 anahtarın arkasında yer alan web uygulamasının sınırlamalarından ötürü seçilmiştir. Daha yüksek eşzamanlı istek kapasitesine sahip web sunucuları için P4 anahtar üzerinde yer alan bağlantı havuzu boyutu bu sunucuların desteklenen eş zamanlı istek limitlerine göre potansiyel olarak arttırılabilir.

4.3.3. SYN saldırılarına karşı savunma performansının değerlendirilmesi

Önerilen SYN savunması için Şekil 4.5'te görüldüğü gibi güncellenebilir Bloom filtresi ve bağlantı havuzu yaklaşımları birlikte kullanılarak testler gerçekleştirilerek bu iki yaklaşımın birlikte kullanılması durumu incelenmiştir. Yapılan testlerde normal istemcilerden saniyede 100 istek oranında istekler oluşturulurken saldırgan istemcilerden çeşitli oranlarda SYN saldırıları hping3 aracı kullanılarak gerçekleştirilmiştir. Önerilen savunma mekanizmasının etkinliği saldırı altında iken normal istemci isteklerine yanıt alınabilme olasılıklarına bakılarak ölçülmüş ve netice olarak Şekil 4.15'te görülen sonuçlar elde edilmiştir.



Şekil 4.15. SYN savunma için artan saldırı yoğunluğu ile bağlantı olasılığının değişimi

Burada yapılan testlerde bmv2 anahtarın sınırlamalarının gerçekçi bir performans değerlendirmesini zorlaştırdığını belirtmek gerekir. SYN saldırılarına karşı savunma için gerçekleştirilen testlerde bmv2 anahtarın artan SYN istekleri ile beraber yüksek CPU tüketimi eğiliminde olduğu ve bununla beraber beklenmedik paket düşürmelere sebep olduğu görülmüştür. Ayrıca bmv2 sanal anahtarın ilgili bilgi sayfasında [107] yüksek performans gerektiren işlemler için uygun olmadığı ve yalnızca geliştirme ortamlarında kullanıma uygun olduğu belirtilmiştir. Bu nedenle CPU aşırı yüklenme durumlarını ve beklenmedik paket düşümlerini önlemek amacıyla bmv2 anahtarın CPU tüketimi takip edilerek daha düşük SYN saldırıları için testler gerçekleştirilmiştir. Şekil 4.13' te görüleceği üzere önerilen tasarım saniyede 75 000 SYN paketine kadar çok yüksek doğruluk sağlıyor olmasına rağmen, daha yüksek oranlardaki istekler için beklenmedik paket düşürmeler gerçekleşerek doğruluk değeri düşmüştür.

SYN savunma için yapılan testler sınırlı olarak gerçekleştirilmiş olsa da bunun nedeni bmv2 sanal anahtarının sınırlamalarından kaynaklanmaktadır. Modern programlanabilir anahtarlar Tbps seviyesinde paket anahtarlama kapasitesi ile beraber, önerilen yöntemin kullanılmasında çok daha yüksek performansa olanak sağlayabilir. Nitekim yapılan testlerde önerilen GBF veri yapısının SYN isteklerinin takibinde Mpps seviyelerinde doğruluk sağladığı gösterilmiştir.

5. DNS SALDIRILARINA KARŞI P4 TABANLI İSTEK LİMİTLEME UYGULAMASI VE GÜVENLİK DUVARI GELİŞTİRME

DNS sunucularında istek limitlemede yaygın olarak bilinen yanıt hızı limitleme *(Response Rate Limiting, RRL)* yaklaşımı, DNS yansıtma ve yükseltme saldırılarının etkisini azaltmak amacıyla bir DNS sunucusunun yanıt hızının sınırlandırılmasında bir metodolojinin ana hatlarını ortaya koymaktadır [35]. Temelde, RRL yaklaşımı DNS sunucusu tarafından alınan tüm DNS sorgularını işler, yanıtlar üretir ve belirli parametrelere (kaynak IP adresi, bölge adı, yanıt kodu, hata kodu vb.) dayalı olarak oluşan yanıtları kategorilere ayırır. Devamında ön tanımlı bir zaman penceresi içerisinde aynı kategoride yer alan DNS yanıtlarının oranı üzerinden yanıtların limitlenmesini jeton kovası algoritması kullanarak gerçekleştirir [36]. Yaygın olarak kullanılan RRL' nin iki temel sınırlamasından bahsedilebilir:

- Gelen sorgular için RRL tarafından limitleme uygulanmaz; tüm DNS sorguları işlenerek eşsiz olarak tespit edilen DNS yanıtları üzerinde limitleme uygulanır [37].

- İkincisi ise RRL yaklaşımının yalnızca yetkili ad sunucularında kullanılması önerilir. Özyinelemeli ad sunucularında RRL uygulanması yanlış pozitif sonuçlara yol açabildiğinden tavsiye edilmez. Bunun nedeni, istemcilerin işletim sisteminin bir parçası olan DNS çözümleyici bileşenlerinin özyinelemeli sunuculara olan isteklerinin sınırlı önbelleğe alma kapasitesinden ötürü aynı sorguların sık yapılmasıdır [35].

Bu noktadan hareketle, yapılan çalışmada alan-etkin bir yaklaşımın programlanabilir veri katmanında geliştirilerek uygulanabilmesi için GBF veri yapısı kullanılmış ve yoğun DNS sorgularının hedef sisteme ulaşmadan ağ katmanında proaktif olarak limitlenmesi amaçlanmıştır. Önerilen yaklaşım, DNS sunucusunun bekleyen iş yüküne *(backlog)* dinamik olarak adapte olan bir yaklaşımla henüz tamamlanmamış isteklerin sayısını takip ederek adaptif istek limitlemesi gerçekleştirmektedir. Bu sayede hem yetkili ad sunucuları hem de özyinelemeli ad sunucularına uygulanabilir olarak geliştirilen yöntem aynı zamanda DNS sunucusu kaynaklarının istemciler arasında daha adil tüketiminin gerçekleştirilmesini de sağlar.

DNS sorgu ve yanıt paketleri incelendiğinde ise protokolün yapısı gereği tipik olarak birebir eşleştirme *(one-to-one mapping)* davranışına sahiptir. Bu özellik istenmeyen DNS yanıt paketlerinin tespitinde ve engellenmesinde kullanılabilir. Böylece hedef tarafında DNS yükseltme saldırılarının etkileri hafifletilebilir. Bu yaklaşım ilk olarak 2007 yılında Kambourakis [38] tarafından önerilmiş ve yeni teknolojiler ortaya çıkmaya devam ettikçe çeşitli uygulamalarda bu yaklaşımı kullanan farklı uygulamalar araştırmacılar tarafından önerilmiştir [39-43].

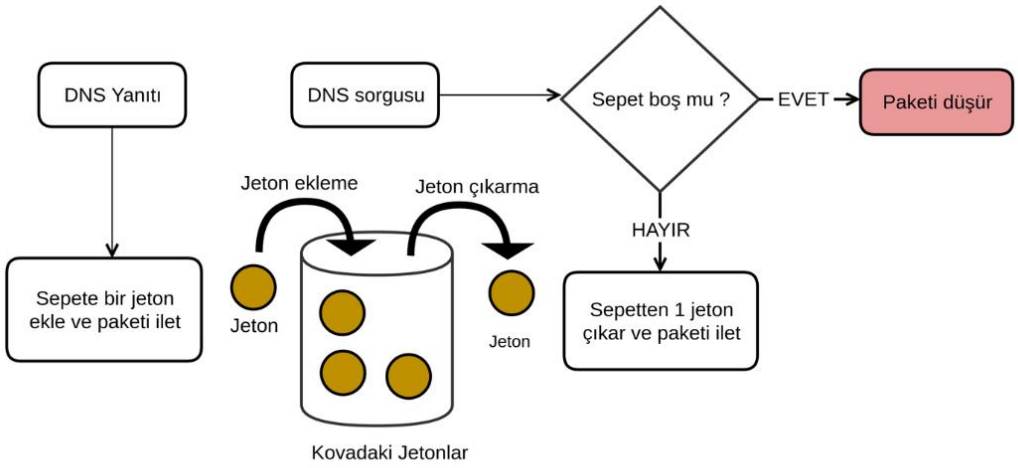
Birebir eşleştirme yaklaşımından hareketle, DNS sorgu ve yanıt paketlerinin programlanabilir veri katmanı üzerinde uygulama seviyesinde takip edilerek yansıtılmış DNS yanıt paketlerinin engellenmesi amaçlanmış ve bunun gerçekleştirilmesinde alan-etkin depolama sağlayan GBF veri yapısı kullanılarak önerilen DNS güvenlik duvarı geliştirilmiştir.

Bunun yanı sıra, DNS için eklenti mekanizmaları (*Extension Mechanisms for DNS, EDNS0*) standardının [44] kullanımı sayesinde 1500 Byte boyutundan daha yüksek boyutta DNS yanıt paketleri ağ katmanında gerçekleşen IP parçalanma (*IP Fragmentation*) standardı kullanılarak iletilebilmektedir. Üstelik UDP üzerinden DNS yanıt paketlerinde IP parçalanmanın kullanımı IP parçalanma standardının en yaygın kullanımı olarak kabul edilmektedir [45]. Fakat parçalanmış yanıt paketleri için birinci dışındaki diğer parça paketleri içerisinde port numarası, DNS işlem numarası gibi üst katman başlıkları yer almaz. Üstelik bu tür parçalanmış yanıt paketleri DNS yükseltme saldırılarında kullanılır [46]. Bu durumda parçalanmış DNS yanıt paketleri birebir eşleştirme davranışına uymaz ve aynı zamanda birinci dışındaki diğer parça paketleri içerisinde üst katman bilgisi olmaması sebebiyle paketlerin ağ üzerinde takibi ve ayrıştırılmasını zor hale getirir. Önerilen yaklaşımda birebir eşleştirme davranışına uymayan parçalanmış DNS yanıt paketlerinin de önerilen DNS güvenlik duvarı yaklaşımında durumsal takibe dahil edilmesi sağlanmış ve yüksek doğrulukla takibi gerçekleştirilmiştir. Üstelik parça paketler arası sürelerin mikro saniyeler mertebesinde olması sayesinde önerilen GBF veri yapısında çok düşük devir sürelerinin kullanılabilmesi mümkün hale gelmiş ve bunun neticesinde Tbps mertebelerine ulaşan trafik akışlarında parçalanmış DNS yanıt paketleri için durumsal takibin yüksek doğrulukla yapılabileceği görülmüştür.

Bu bölümde DNS servis engelleme saldırılarına karşı önerilen aktif istek limitleme ve güvenlik duvarı yaklaşımları için öncelikle programlanabilir veri katmanı üzerindeki tasarım detayları ve sonrasında performansına yönelik yapılan testlerin sonuçları yer almaktadır.

5.1. Aktif İstek Limitlemenin Güncellenebilir Bloom Filtresi ile Gerçekleştirilmesi

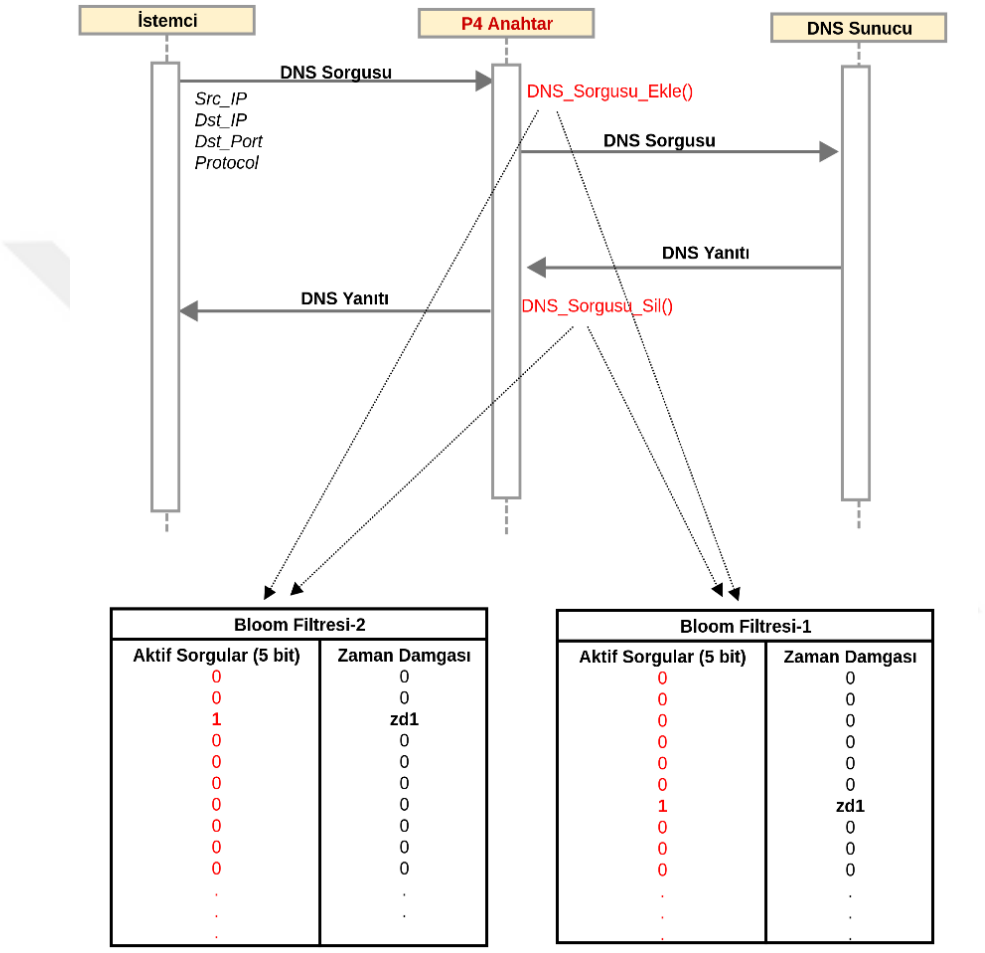
Jeton kovası (*Toket Bucket*), ağ katmanındaki belirli bir port veya trafik için bant genişliği veya paket sayısını sınırlamak için kullanılan bir algoritmadır. Bu yöntemde jeton kovası aslında belirli bir oranda jeton toplayan bir sayaç olarak çalışır. Kovaya eklenecek jetonların toplanma hızı jeton yenileme oranı (*Token Replenishment Rate*) olarak bilinir. Ayrıca kova maksimum kapasitesine ulaştıktan sonra yeni jeton almaz. Yeni paketler alındıkça alınan paketlerin boyutu veya sayısı oranında jeton kovadan atılırken, jeton yenileme oranında jeton her saniye kovaya eklenmektedir. Şayet herhangi bir anda gelen paket için kovada yeterince jeton yok ise paket düşürülebilir veya kuyruğa alınarak kovada yeterince jeton oluşması beklenebilir [108]. Bu sayede akış hızı kontrol edilerek belirli sınırlar içerisinde kalması sağlanmış olur.



Şekil 5.1. DNS istek limitlemesi için değiştirilmiş jeton kovası algoritması

Gerçekleştirilen bu çalışmada jeton kovası yaklaşımından esinlenilerek programlanabilir veri katmanında DNS isteklerinin limitlenmesinde benzer bir yaklaşım benimsenmiş ve Şekil 5.1'de görülen değiştirilmiş jeton kovası algoritması P4 donanım üzerinde gerçekleştirilmiştir. Önerilen yaklaşımda prensip olarak yapılan her DNS sorgusu ile kovadan bir adet jeton çıkarılırken alınan her DNS yanıtı ile bir jeton eklenmektedir. Bu sayede jeton kovasının boyutu maksimum eşzamanlı sorgu limitini belirlemektedir.

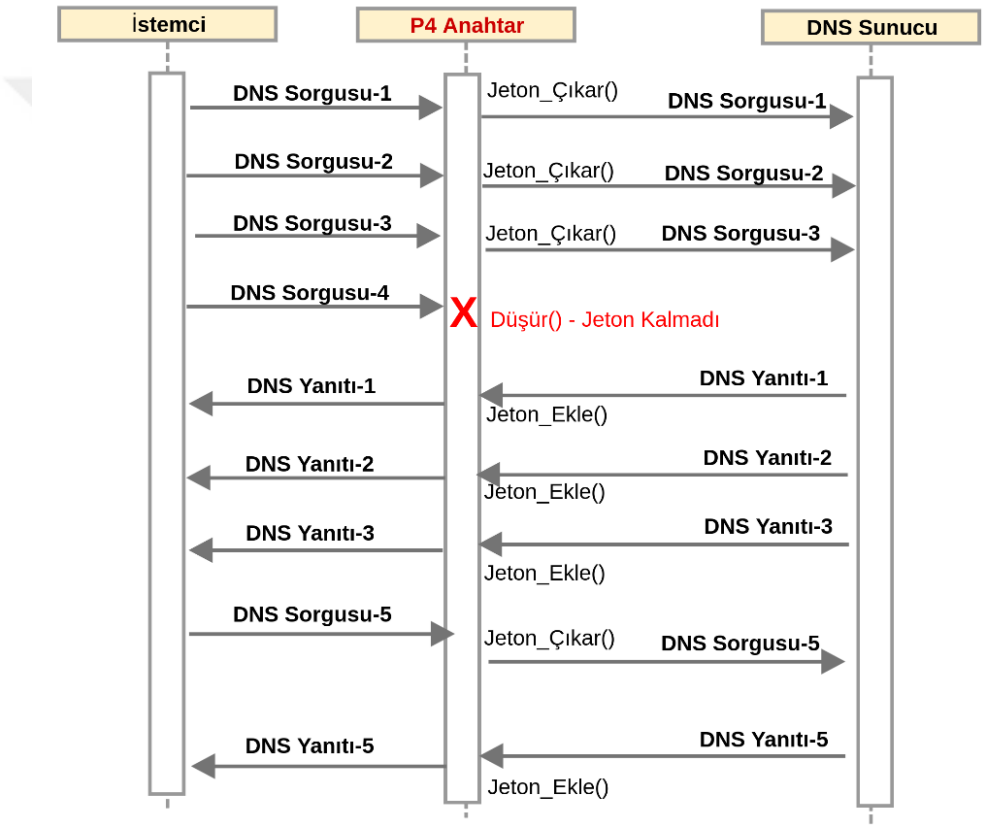
Eşzamanlı sorgu limiti parametresi kısaca *esl* olarak tanımlanmıştır ve değeri uygulanan DNS sunucu kapasitesine göre belirlenir. Bu parametrenin değeri bir ile maksimum kova boyutu arasında seçilir. Ayrıca GBF'nin kullandığı devir süresi zamansal jeton yenileme oranı görevini de yerine getirmektedir. Bu yaklaşım Bloom filtresi veri yapısının her bir dizi elemanında DNS istek limitlemesinin yapılabilmesini mümkün kılmaktadır.



Şekil 5.2. Güncellenebilir Bloom filtresi ile DNS paketlerinin durumsal takibi

DNS protokolü yapısı gereği birebir eşleştirme *(one-to-one mapping)* yapısında çalıştığından GBF veri yapısında DNS sorgu ve yanıt paketlerinin durumsal olarak takip edilmesi cazip hale gelmektedir. Başka bir deyişle, DNS protokolü durumsuz çalışan UDP protokolünü kullanmasına rağmen önerilen yaklaşım DNS isteklerinin durumsal olarak takip edilmesini mümkün kılar. Önerilen yöntem sayesinde DNS sunucusuna gönderilen isteklerin ve yanıtların durumsal takibi programlanabilir veri katmanında gerçekleştirilmiştir.

Şekil 5.2' de görüldüğü üzere istemciden gelen her DNS sorgusu veri yapısına eklenir. Bunun gerçekleştirilmesi esnasında verinin yazılacağı alanın indeks değerinin belirlenmesinde kaynak IP adresi (*Src_IP*), hedef IP adresi (*Dst_IP*), hedef port numarası (*Dst_Port*) ve protokol değerleri kullanılacak özetleme fonksiyonlarına giriş verisi olarak kullanılmaktadır. Bu değerlerin kullanılmasının sebebi Bloom filtresi veri yapısında aynı istemciden gelen bütün isteklerin aynı dizi elemanları içerisinde tutulmasıdır. Böylece istemci IP adresi başına aktif DNS sorgularının durumsal verisi tutularak DNS yanıtları alındıkça bu veri silinir.



Şekil 5.3. Örnek eşzamanlı DNS sorgusu limitleme (*esl* = 3)

Bunun yanı sıra aynı istemciden eş zamanlı birden çok DNS sorgusu da gelebilir. Örneğin Şekil 5.2'te görüleceği üzere aktif sorgular için veri yapısının genişliği (n) 5 bit olarak verilmiştir. Bu durumda aynı istemciden 31 adet ($2^n - 1$) eş zamanlı DNS sorgusuna kadar veri yapısında tutulabileceği anlamına gelmektedir.

Önerilen değiştirilmiş jeton kovası algoritmasının GBF veri yapısında kullanılması ile gerçekleştirilen DNS sorgusu limitleme yaklaşımı için örnek olarak *esl=3* olarak seçilirse eşzamanlı DNS sorguları için limitleme Şekil 5.3'te görüldüğü gibi gerçekleşmektedir. GBF veri yapısının sağlamış olduğu zamansal kontrol sayesinde veri güncelliği her gelen verinin yazılması esnasında kontrol edilir ve güncel olmayan veriler silinmektedir. DNS protokolüne özgü bir yaklaşım için GBF veri yapısı kullanıldığından gelen istekler için zamansal fark (t_{fark}) hesaplandıktan sonra DNS zaman aşımı süresi göz önünde bulundurularak bu süreyi aşan veriler silinmektedir. Bu sayede veri yapısının yüksek doğruluk değerini sağlaması gerçekleştirilmiş olur.

Önerilen yaklaşımda istek limitlemesi iki farklı seviyede gerçekleşmektedir:

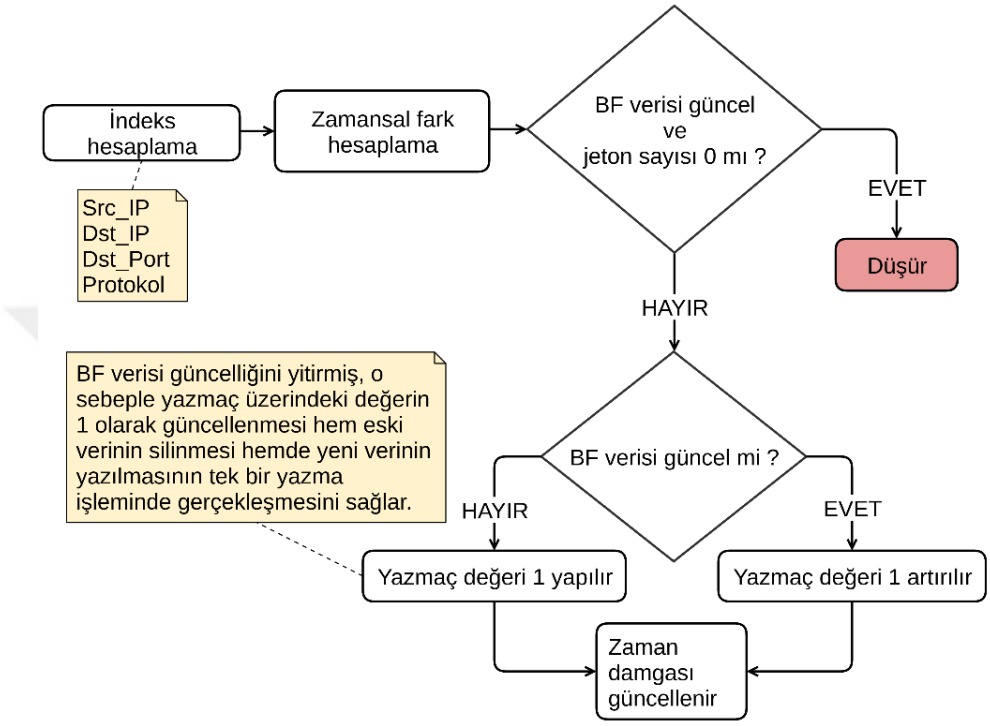
- İstemci IP adresi başına istek limitlemesi uygun bir *esl* değerinin seçilmesi ile gerçekleştirilir. Şekil 5.3'te örnek bir çalışma *esl = 3* değeri için görselleştirilmiştir.

- Toplam aktif istek limitlemesi, DNS sunucusuna gelen toplam istekler için Bloom filtresi uzunluğu (*m*) ve *esl* değerlerinin birlikte kullanılması ile gerçekleştirilir. Örneğin, *m* = 4096 ve *esl* = 3 seçilirse, P4 anahtar DNS sunucusuna gelen bütün istekleri ve onların yanıtlanma durumlarını aktif olarak takip ederek azami 12 288 (*m x esl*, özetleme fonksiyonu çakışmaları göz ardı edilirse) aktif isteğin işlenmesine izin verecektir. P4 anahtar bu değerden daha yüksek oranda gelen istekleri limitleyerek engellenecektir. Bu sayede DNS sunucusu sadece yanıtlayabileceği oranda bir sorgulama yüküne maruz kalarak dağıtık servis kesintisi saldırıları esnasında bile adres çözümleme işleminde kesinti olmasını engeller. Her ne kadar bu yaklaşım saldırgan olmayan istemcilerden gelen istekler içinde paket düşürmelere sebep olabilse de DNS sunucunun saldırıdan ötürü tamamen yanıt veremez duruma gelmesini engellemektedir.

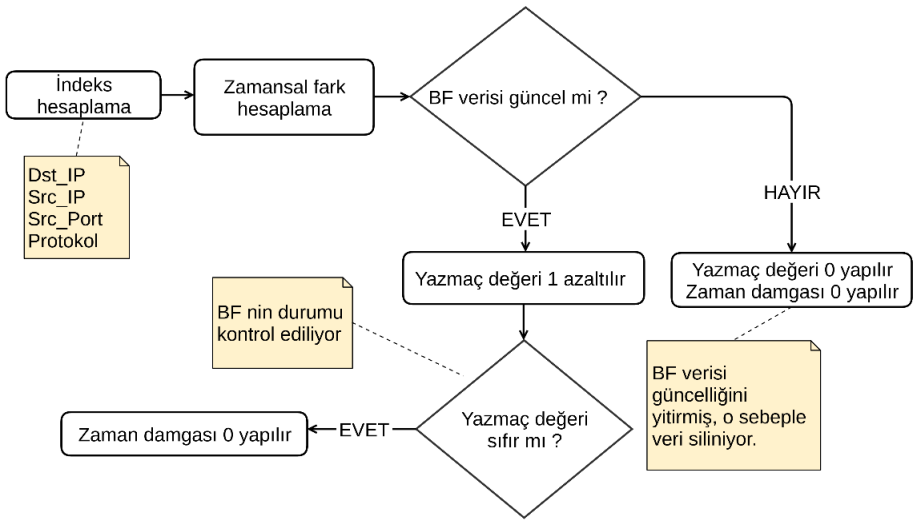
Şekil 5.4 ve Şekil 5.5'de DNS sorgu ve yanıt paketlerinin gerçekleştirilen P4 programında işlenmesinde gerçekleştirilen akış diyagramları görülmektedir. Önerilen yaklaşım sadece istemciden yapılan ve yanıt bekleyen aktif DNS isteklerini takip etmektedir.

DNS sunucuları önbelleklerinde bulunan adresler için adres çözümleme işlemini çok hızlı gerçekleştirirken önbellekte bulunmayan adresler için bu süre artmaktadır. Bu şekilde

önbellekte bulunmayan ve henüz yanıtlanmamış sorguların artması durumunda DNS sunucusunun bekleyen iş yükü artacaktır. Bu açıdan bakıldığında, aslıdan önerilen yaklaşım DNS sunucusunun bekleyen iş yüküne *(backlog)* dinamik olarak adapte olur ve henüz tamamlanmamış isteklerin sayısını takip ederek limitlenmesini sağlar.



Şekil 5.4. Aktif istek limitleme için DNS sorgu paketlerinin işlenmesi



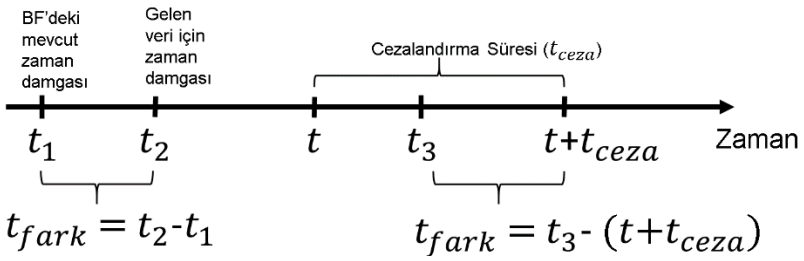
Şekil 5.5. Aktif istek limitleme için DNS yanıt paketlerinin işlenmesi

İstemci IP adresi başına istek limitlemesi aynı zamanda saldırganların yoğun istek yaparak aşırı sunucu kaynağı tüketmesini engeller ve DNS sunucusunun kaynaklarının istemciler arasında daha adil tüketiminin gerçekleştirilmesini sağlar.

5.2. Saldırganlar için Zaman Bazlı Cezalandırma Tasarımı

GBF tasarımında zaman damgası alanı gelen verinin güncelliğinin zaman damgası temelli olarak kontrol edilmesinde mevcut veri ve yeni gelen veri arasındaki zamansal fark (t_{fark}) değerinin hesaplanması esasına dayanmaktadır. Fakat bunun yanı sıra, zaman damgası alanı tespit edilen bir saldırganın belirlenen cezalandırma süresince (t_{ceza}) engellenmesi veya gelen DNS sorguları için sorgular arası zamanın (*inter-query time*) ölçülmesi için de kullanılabilir. Üstelik bunun gerçekleştirilmesi Şekil 5.8'da görüldüğü gibi sadece birkaç ilave kontrolün eklenmesi ile mümkün hale gelmektedir.

Şekil 5.4'te görüldüğü üzere, jeton sayısının değerinin sıfır olması durumunda yeni jetonların kovaya eklenmesine kadar geçen sürede gelen DNS sorguları düşürülür. Kovada kullanılabilir jeton olmamasına rağmen yeni sorgular yapan istemciler saldırgan olarak sınıflandırılarak cezalandırılabilir. Nitekim bu davranış sadece yoğun istekler yapan saldırganlar tarafından gerçekleştirilir.

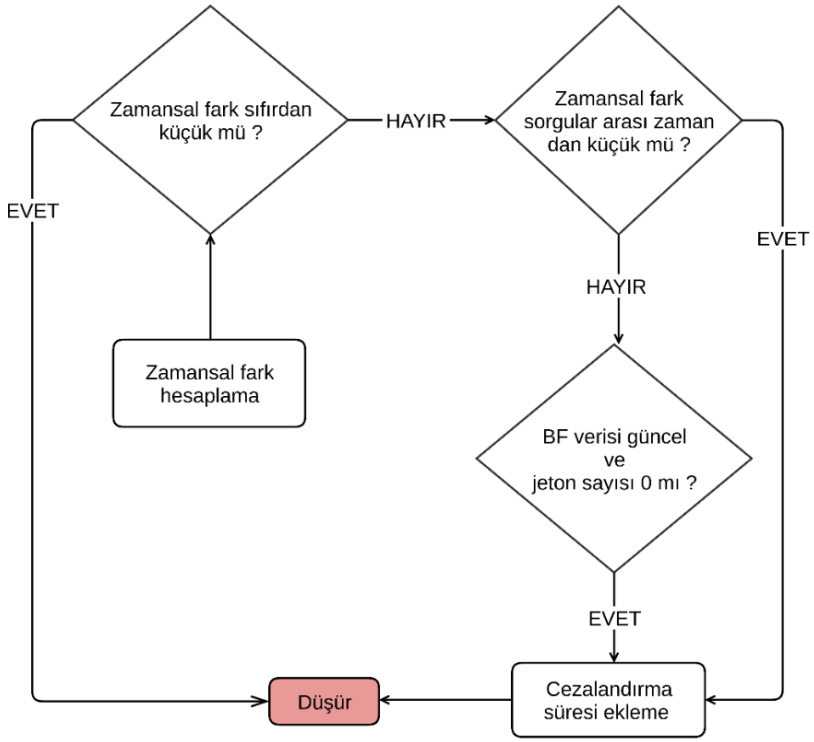


Şekil 5.6. Güncellenebilir Bloom filtresinde ceza süresinin gerçeklenmesi

Şekil 5.6' da görüldüğü üzere t_2 anında gelen bir sorgu için veri yapısında zamansal fark hesaplaması t_2 değerinden t_1 değerinin çıkarılması ile hesaplanır. Herhangi bir t anında bir saldırganın tespit edilerek cezalandırmasına karar verildiğini varsayalım. Bu durumda Bloom filtresi veri yapısında ilgili zaman damgası kovasına $t + t_{ceza}$ değeri yazılırsa, t anından $t + t_{ceza}$ anına kadar aynı istemciden gelen bütün istekler için hesaplanacak zamansal fark değeri negatif değerli olacaktır. Bu durumda zamansal fark değerinin negatif

olduğu durumlarda gelen sorgular Şekil 5.7'da görüldüğü gibi ilave bir kontrol ile düşürülebilir.

GBF veri yapısında zaman damgası aynı zamanda aynı istemciden en son gelen sorgu paketinin zaman damgası bilgisini barındırdığından sorgular arası zaman değerinin de ölçülebilmesi mümkün hale gelir. Bu sayede aynı istemciden gelen yoğun DNS sorguları, sorgular arası sürenin ölçülmesi ile tespit edilerek engellenebilir. Bu sebeple hesaplanan t_{fark} değerinin sıfırdan büyük fakat izin verilen sorgular arası süre eşik değerinden küçük olması durumunda ilgili istemci saldırgan olarak sınıflandırılabilir. DNS sunucuları ve istemciler arasında etkileşim için analitik bir model geliştirmek amacıyla yapılan [109] çalışmasında normal istemci davranışında sorgular arası sürenin 1 ms değerinden başladığı gösterilmiştir. Bu sebeple 1 ms değerinden daha düşük bir eşik değeri seçilerek bu değerinden daha düşük aralıklarla sorgu yapan saldırganlar tespit edilebilir.



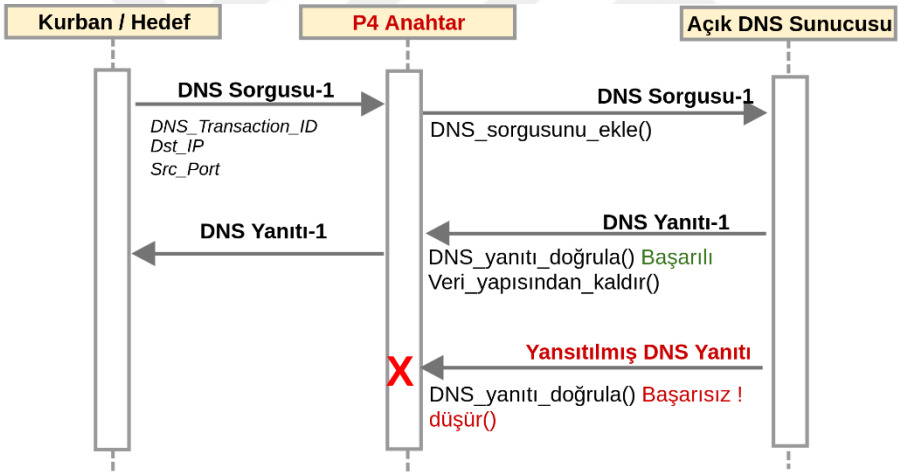
Şekil 5.7. Zaman temelli cezalandırma akış diyagramı

Geleneksel yaklaşımda saldırgan IP adreslerinin tespit edilmesi sonrasında bu adresler engelleme listesine eklenir. Fakat bu durumda artan saldırgan IP adresi sayısı ile beraber anahtar kapasitesinin yetersiz kalması söz konusu olabilir. Aynı zamanda bu sürecin

kontrolcü tarafından gerçekleştirilmesi gereksiniminden ötürü engelleme listesine eklenme süresi saldırının yoğunluğuna bağlı olarak artabilir. Fakat burada önerilen cezalandırma yaklaşımında herhangi bir ilave depolama alanına ihtiyaç duyulmaz. GBF veri yapısının kullanımı ile beraber saldırgan IP adreslerinin yerine bu adreslerin veri yapısında kullandığı indeks alanı cezalandırılmaktadır. Bu yöntemin seçilmesinin sebebi cezalandırma süresi boyunca aynı indeks alanı saldırgan ve normal istemcilerin kullanması olasılığının düşük olarak kabul edilmesidir. Üstelik cezalandırma süresini sona erdirmek için ilave bir kontrol mekanizmasına da ihtiyaç duyulmaz.

5.3. DNS Güvenlik Duvarının Güncellenebilir Bloom Filtresi ile Gerçekleştirilmesi

DNS yükseltme saldırılarına karşı güvenlik duvarının gerçekleştirilmesi amacıyla GBF veri yapısı kullanılarak DNS sorgularının durumsal takibi gerçekleştirilmiştir.



Şekil 5.8. DNS güvenlik duvarı durumsal sorgu takibi

Şekil 5.8'de görüldüğü üzere, hedef veya kurban olarak adlandırılan istemci tarafından yapılan bütün DNS sorgular GBF veri yapısına eklenir. Gönderilen sorgular için alınan ilgili yanıtlar GBF veri yapısında doğrulanarak iletilir ve devamında sorgunun verisi veri yapısından silinir. Şekil 5.8' de görüldüğü üzere DNS sorgusunun GBF veri yapısında verinin yazılacağı alanın indeks değerinin belirlenmesinde kaynak port numarası (*Src_Port*), hedef IP adresi (*Dst_IP*) ve DNS işlem numarası (*DNS_Transaction_ID*) parametreleri kullanılan özetleme fonksiyonlarına giriş verisi olarak kullanılmaktadır. DNS yanıtının doğrulanmasında benzer şekilde aynı parametreler kullanılır ve doğrulanan yanıtlar iletilir.

Yansıtılmış DNS saldırılarından ötürü gelen DNS yanıt paketlerinin doğrulanması başarısız olur ve bu paketler düşürülür. Ayrıca herhangi bir sebepten alınamayan DNS yanıtları olursa GBF veri yapısının sağlamış olduğu zamansal güncellik kontrolü sayesinde ilgili veriler bir devir süresi sonrasında geçersiz hale gelir ve silinir.

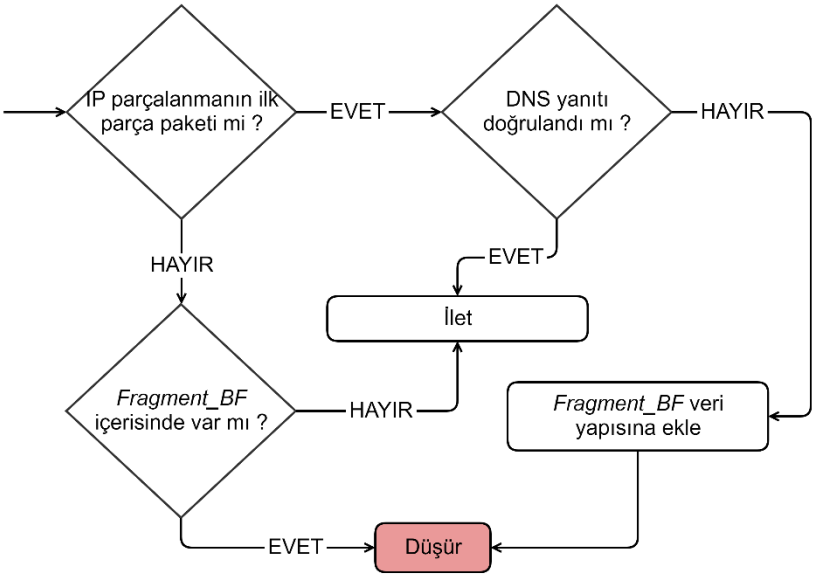
5.4. Parçalanmış DNS Yanıtlarının Durumsal Takibi

Günümüzde standart hale gelen EDNS0 [44] kullanımı sayesinde 1500 Byte boyutundan daha yüksek boyutta DNS yanıt paketleri IP parçalanma (*IP Fragmentation*) kullanılarak iletilebilmektedir. EDNS0'ın yaygın olarak benimsenmesiyle, UDP üzerinden DNS yanıt paketlerinde IP parçalanmanın kullanımı IP parçalanma standardının en yaygın kullanımı olarak kabul edilmektedir [45] . Bu sebeple tek bir DNS sorgu paketine karşılık yüksek boyutta bir DNS yanıt paketi için iki veya daha fazla parçalanmış paketten oluşan DNS yanıtları alınabilir. Sonuç olarak durumsal DNS takibinde kullanılan birebir eşleştirme davranışına uyulmaz ve bu veri katmanında durumsal paket işlemeyi zorlaştırır. Bunun üstesinden gelmek ve yükseltilmiş DNS saldırı paketlerini engellemek için IP parçalanma paketlerinin tamamı engellenebilir fakat bu durum IP parçalanmayı kullanan bütün uygulamaların engellenmesi anlamına gelmektedir. Bu sebeple alternatif bir yöntem olarak parçalanmış DNS yanıt paketlerinin durumsal takibe dahil edilmesi ile sadece saldırı paketleri engellenebilir. Bu sayede IP parçalanma kullanan ve saldırı paketi olmayan gerek DNS yanıt paketleri gerekse farklı uygulamalar engellenmemiş olur.

Gerçekleştirdiğimiz çalışmada, ikinci bir GBF veri yapısı yükseltilmiş DNS saldırısı amacıyla oluşturulan ve IP parçalanma kullanan DNS paketlerinin durumsal takibi için kullanılmaktadır. Şekil 5.9'da görüldüğü üzere, *Fragment_BF* olarak isimlendirilen ikinci bir GBF veri yapısı sadece saldırı amacıyla oluşturulan parçalanmış DNS yanıt paketlerinin durumsal takibi için kullanılmaktadır. IP parçalanmanın yapısı gereği port numarası, DNS işlem numarası gibi üst katmanlara ait veriler aynı akışa ait parça paketler içerisinden sadece ilk paket içerisinde yer almaktadır. Bu sebeple önerilen yaklaşımda alınan DNS yanıt paketlerinde IP parçalama kullananların saldırı paketi olup olmadığının tespit edilmesi ve sınıflandırılması için IP parçalanmanın ilk parça paketi kullanılmaktadır.

Gerçekleştirilen tasarımda *Fragment_BF* olarak isimlendirilen veri yapısı yükseltilmiş DNS yanıt paketleri için tahsis edilmiştir. Başka bir deyişle, parçalanmış saldırı paketleri için

engelleme listesi olarak kullanılan bir veri yapısıdır. Alınan ilk saldırı paketi ile beraber, IP katmanında yer alan kaynak adresi (*Src_IP*), hedef adresi (*Dst_IP*) ve kimlik numarası (*Identification*) parametreleri kullanılan özetleme fonksiyonlarına giriş verisi olarak kullanılmaktadır. Devamında aynı devir süresi içerisinde ve aynı parametreleri kullanan devamındaki parça paketler engellenmektedir. *Fragment_BF* veri yapısı gelen parçalanmış DNS yanıtları için verileri sadece bir devir süresi boyunca tutmaktadır. Gerçekleştirilen testlerde IP parçalanma kullanan bir yanıt için ilk ve son paketler arası zaman ortalama 7 µs olarak ölçülmüştür. Bu durumda devir süresi örneğin 10µs olarak bu değerden daha yüksek bir değer olacak şekilde seçilebilir. Bu şekilde çok düşük değerli bir devir süresi kullanılması çok düşük boyuttaki GBF veri yapısında çok yüksek oranda saldırı verisinin tutulabilmesini mümkün hale getirmektedir.



Şekil 5.9. Parçalanmış DNS yanıtlarının durumsal takibi

Diğer taraftan GBF veri yapısı olasılıksal bir veri yapısı olması sebebiyle hedef odaklı kapsama saldırılarına maruz kalabilir. Parçalanmamış DNS yanıt paketlerinin takibinde veri yapısına veri sadece kurban tarafından gönderilen DNS sorguları için yazıldığından saldırganlar tarafından bu şekilde bir saldırıya maruz kalınması mümkün değildir. Parçalanmış DNS yanıt paketleri için ise IP katmanında yer alan kimlik numarası salt giriş verisi olarak kabul edilebilir. Nitekim bu değer saldırganlar tarafından değil saldırı esnasında yansıtma ve yükseltme aracı olarak kullanılan açık özyinelemeli DNS sunucusu tarafından belirlenmektedir.

5.5. Önerilen DNS Fonksiyonlarının Performansının Değerlendirilmesi

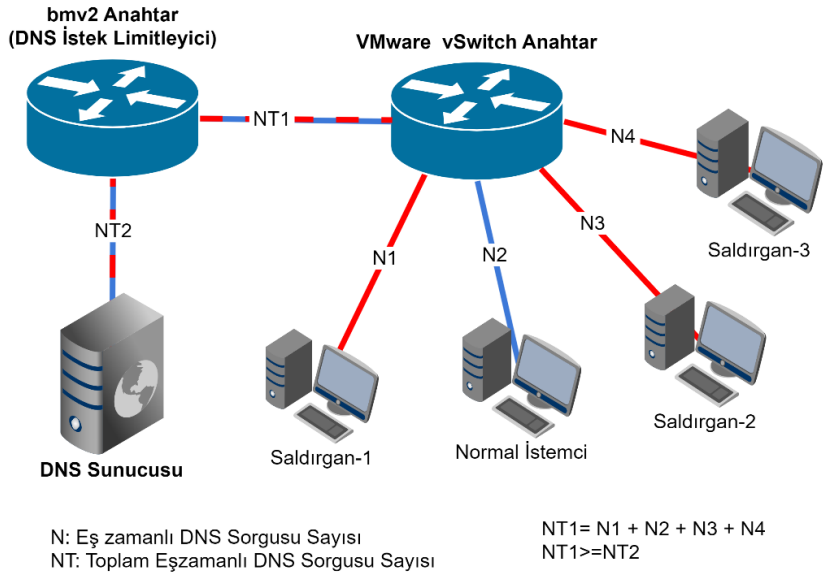
Önerilen iki yaklaşımın etkinliğinin test edilmesi amacıyla Şekil 5.10 ve Şekil 5.23'te görülen iki farklı bağlantı topolojisi oluşturularak çeşitli testler gerçekleştirilmiştir. Her iki topolojide 3 adet saldırgan istemci ve bir adet normal istemci bulunmaktadır. DNS istek limitleme testlerinde 4 adet istemcinin tamamı VMware vSwitch [110] isimli sanal anahtara bağlı iken DNS sunucusu bmv2 [88] sanal anahtarına bağlanmıştır. Her iki durumda istekler bmv2 anahtarına aynı port üzerinden gelmektedir. Ayrıca DNS sunucusu için BIND9 [111] (versiyon 9.16.1) kullanılırken Dnsmeter [112] aracı performans ölçümleri için kullanılmıştır. Dnsmeter yüksek oranda DNS sorgusu oluşturarak gönderim yaparken gelen yanıtların işlenmesini de sağlamaktadır. Gerçekleştirilen test ortamında 6 adet sanal sunucu VMware ESXi (versiyon 6.7.0 U2) sanallaştırma ortamında kullanılmıştır. Tahsis edilen kaynakların detayları aşağıdaki Çizelge 5.1'de yer almaktadır.

Çizelge 5.1. Test ortamında tahsis edilen kaynaklar

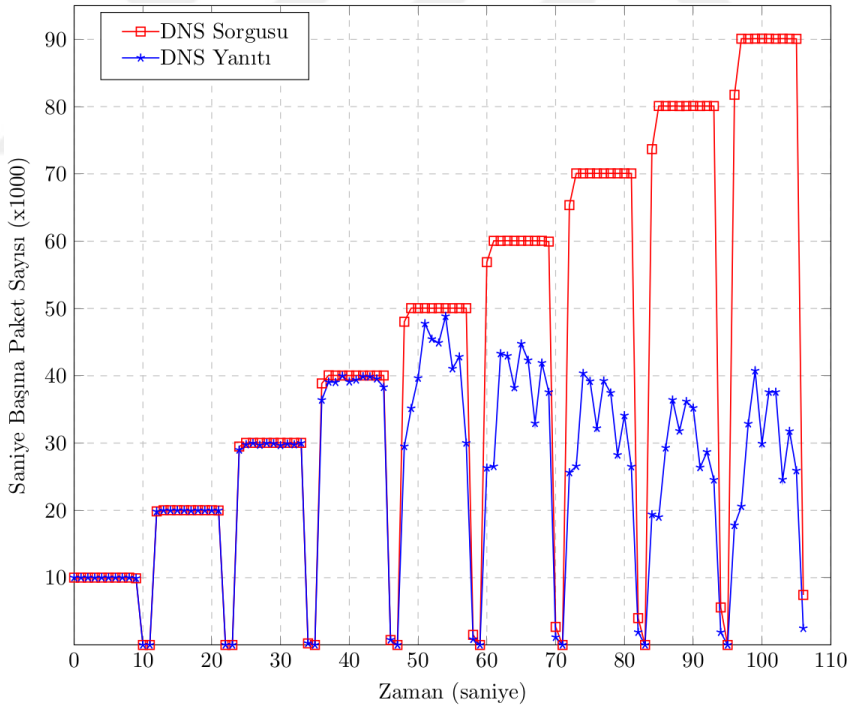
İsim	**CPU Sayısı**	**Hafıza**
Fiziksel Sunucu	8 Intel Core i7 – 7700 (3.6 GHz)	64GB
bmv2 anahtar	8	16GB
DNS Sunucusu	4	16GB
Normal İstemci	4	4GB
Saldırgan-1	4	4GB
Saldırgan-2	4	4GB
Saldırgan-3	4	4GB

Önerilen yaklaşımların detaylı performans testlerine geçmeden kullanılan sanal ortamın paket anahtarlama kapasitenin tespit edilmesi için yük testleri gerçekleştirilmiştir. Nitekim bmv2 anahtarının yüksek kapasiteli paket anahtarlama için uygun olmamasından ötürü [107] artan paket anahtarlama yükü ile beraber beklenmedik paket düşürmelere sebep olabilmektedir.

Şekil 5.11'de görüldüğü üzere saniyede 10 000 DNS sorgusundan başlanarak 90 000 DNS sorgusuna kadar 10 000 artımlarla farklı oranlarda DNS sorgusu için test ortamında DNS paketlerinin yanıtlanma durumlarına bakılmıştır. Bunun sonucunda saniyede 30 000 DNS sorgusundan itibaren paket düşürmelerin gerçekleştiği görülmüştür. Bu nedenle sonrasında gerçekleştirilen testler için güvenilir sonuçlar elde etmek amacıyla saniye de 20 000 DNS sorgusuna kadar testler gerçekleştirilerek sonuçlar elde edilmiştir.



Şekil 5.10. DNS istek limitleme test topolojisi



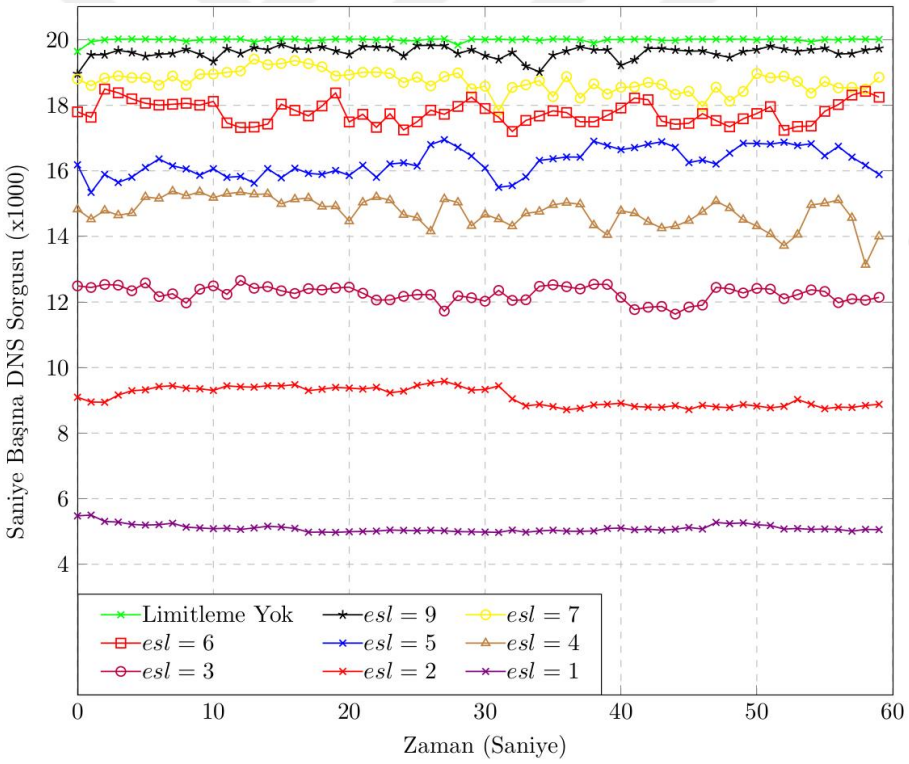
Şekil 5.11. Test ortamı DNS sorgusu yük testi

5.5.1. DNS istek limitleme parametrelerinin etkisinin incelenmesi

Yetkili ad sunucusu (*Authoritative Name Server, ANS*) testleri için ilk olarak sunucunun kendi adres uzayında yer alan adresler için farklı oranlarda DNS sorguları ile testler

gerçekleştirilmiştir. Özyinelemeli ad sunucusu (*Recursive Name Server, RNS*) testleri için ise 1 200 000'den fazla gerçek alan adı bilgisi barındıran bir veri seti kullanılmıştır. CIC-Bell-DNSEXF-2021 [113] veri seti ile beraber kendi topladığımız alan adlarının birleşiminden oluşan veri seti kullanılmıştır.

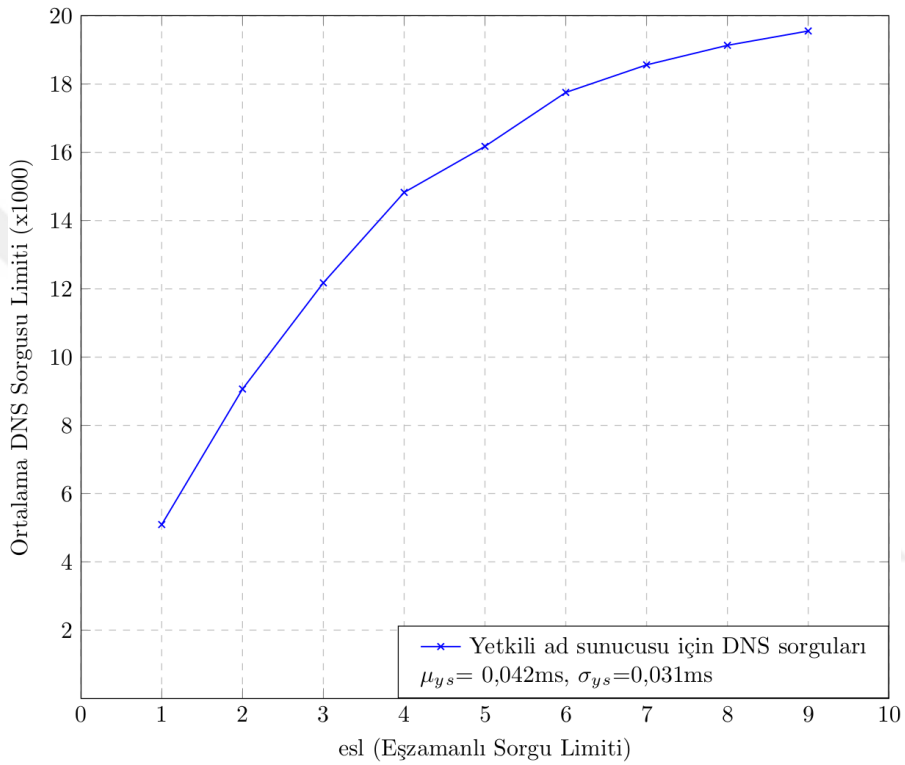
Önerilen yaklaşımın istemci başına istek limitleme performansının ölçülmesi amacıyla çeşitli eş zamanlı sorgu limiti değerlerinde tek bir istemciden yapılan istekler için istek limitleme performansı incelenmiştir. Gerçekleştirilen istek limitleme aktif sorguların limitlenmesini gerçekleştirdiği için, istek limitleme oranları DNS sunucusunun sorgu yanıtlama süresine bağlı olarak değişmektedir. Bu sebeple farklı *esl* değerleri için saniyede 20 000 DNS sorgusu tek bir istemciden 60 saniye süresince uygulanarak yetkili ad sunucusu ve özyinelemeli ad sunucusu için istek limitleme sonuçları ayrı ayrı elde edilmiştir.



Şekil 5.12. Yetkili ad sunucusunda tek istemci için çeşitli esl değerlerinde DNS sorgusu limitleme

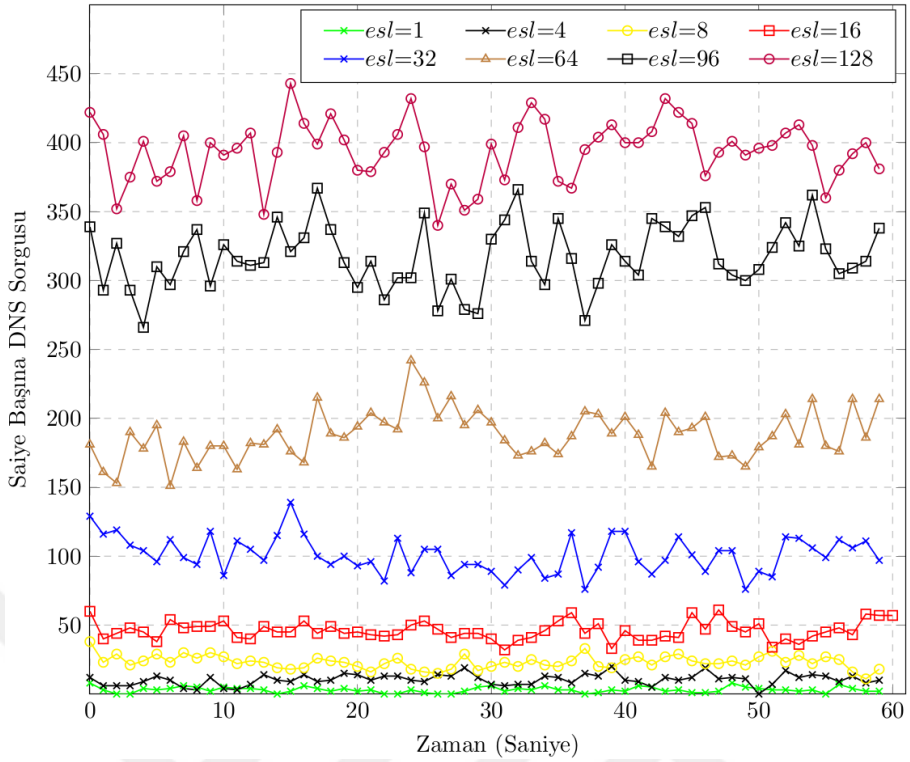
Şekil 5.12' de görüldüğü üzere, önerilen yaklaşımın değişen *esl* değerleri için farklı oranlarda istek limitlemeyi başarılı bir şekilde gerçekleştirdiği görülmektedir. Şekil 5.13 ise ortalama sorgulama limitlerinin farklı esl değerleri ile değişimini göstermektedir.

Ortalama yanıt süresi olan μ_{ys} değerinin ve yanıt sürelerinin varyansı olan σ_{ys} değerleri yetkili ad sunucusu için oldukça düşük olarak ölçülmüştür. Gerçekleştirilen testlerde *esl*=1 değerinin kullanılması aslında istemcinin tek bir jetona sahip olduğu anlamına gelmektedir. Buna rağmen yüksek oranlarda DNS sorgularına izin verilmesinin sebebi önerilen yaklaşımın DNS sunucusunun yanıt süresine adaptif çalışmasıdır. Şekil 5.13'te görüldüğü üzere yapılan testlerde μ_{ys}= 0,042 ms ve σ_{ys} = 0,031 ms olarak ölçülmüştür.

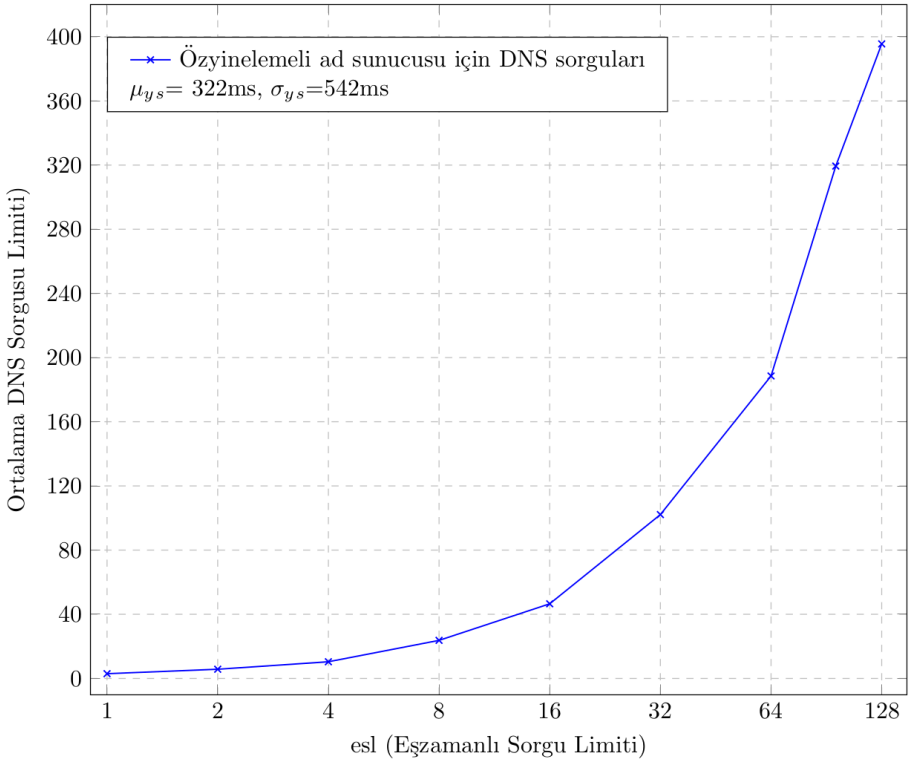


Şekil 5.13. Yetkili ad sunucusunda tek istemci için ortalama sorgu limiti değişimi

Şekil 5.14 ve Şekil 5.15' de özyinelemeli ad sunucusuna tek bir istemciden yapılan sorgulamalar için elde edilen DNS sorgu limitleme sonuçları görülmektedir. Tutarlı sonuçların elde edilmesi için herhangi iki test arasında en az 60 dakika süre beklenmiştir. Bunun sebebi açık özyinelemeli DNS sunucuların bu sorguları kendi önbelleklerinde tutarak devamında yapılan testlerde kendi önbelleklerinde bulunan adresler için hızlı yanıt dönmeleri sonucunda oluşacak tutarsız test sonuçlarının önüne geçmektir. Yapılan testler sonucunda μ_{ys} = 322 ms ve σ_{ys} = 542 ms olarak ölçülmüştür.

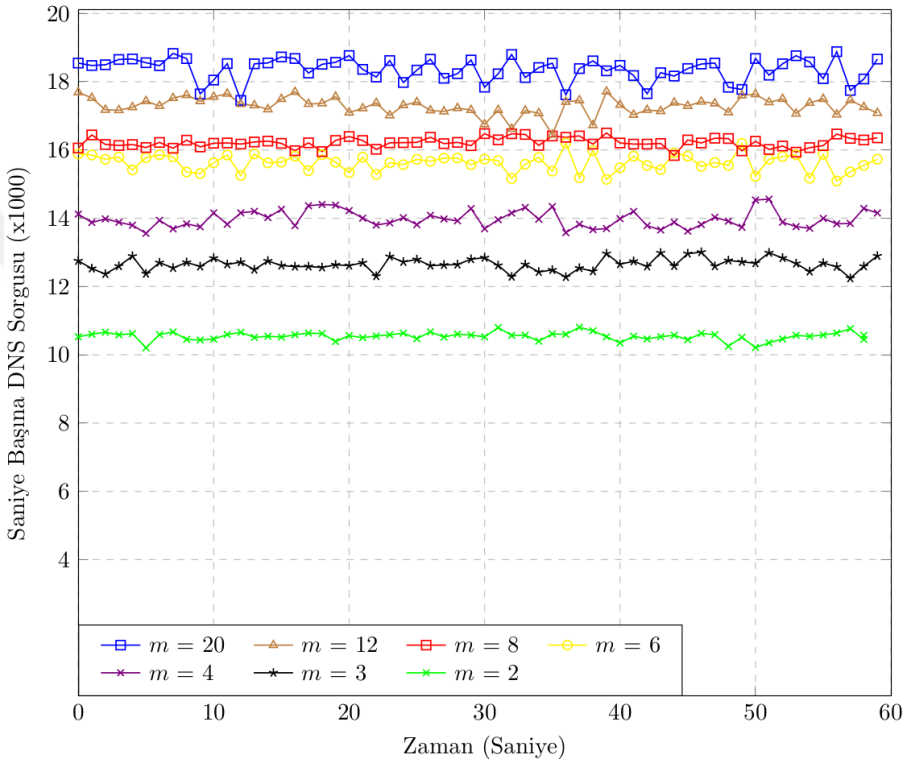


Şekil 5.14. Özyinelemeli ad sunucusunda tek istemci için çeşitli *esl* değerlerinde DNS sorgusu limitleme



Şekil 5.15. Özyinelemeli ad sunucusunda tek istemci için ortalama sorgu limiti değişimi

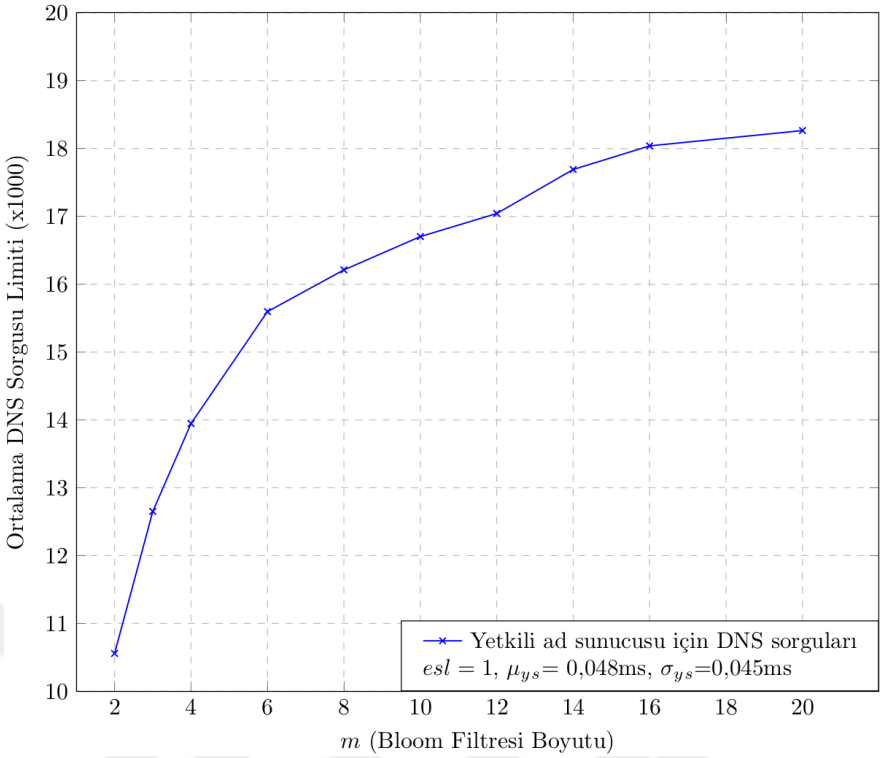
Özyinelemeli adres çözümlemede yanıt süresi farklı alan adları için çok çeşitli sebeplerden ötürü farklı olmaktadır. Bu durum yanıtlanma sürelerinin varyansının yüksek çıkmasına sebep olur. Bunun sonucunda adres çözümleme süresinin varyansının yüksek olması istek limitleme oranlarında salınımların Şekil 5.14' te görüldüğü üzere daha yüksek olmasına sebep olmaktadır. Örneğin esl = 32 değeri için, ortalama saniyede 102 sorgu limiti gerçekleşirken bu değerin 76 ile 139 arasında değişebildiği gözlemlenmiştir.



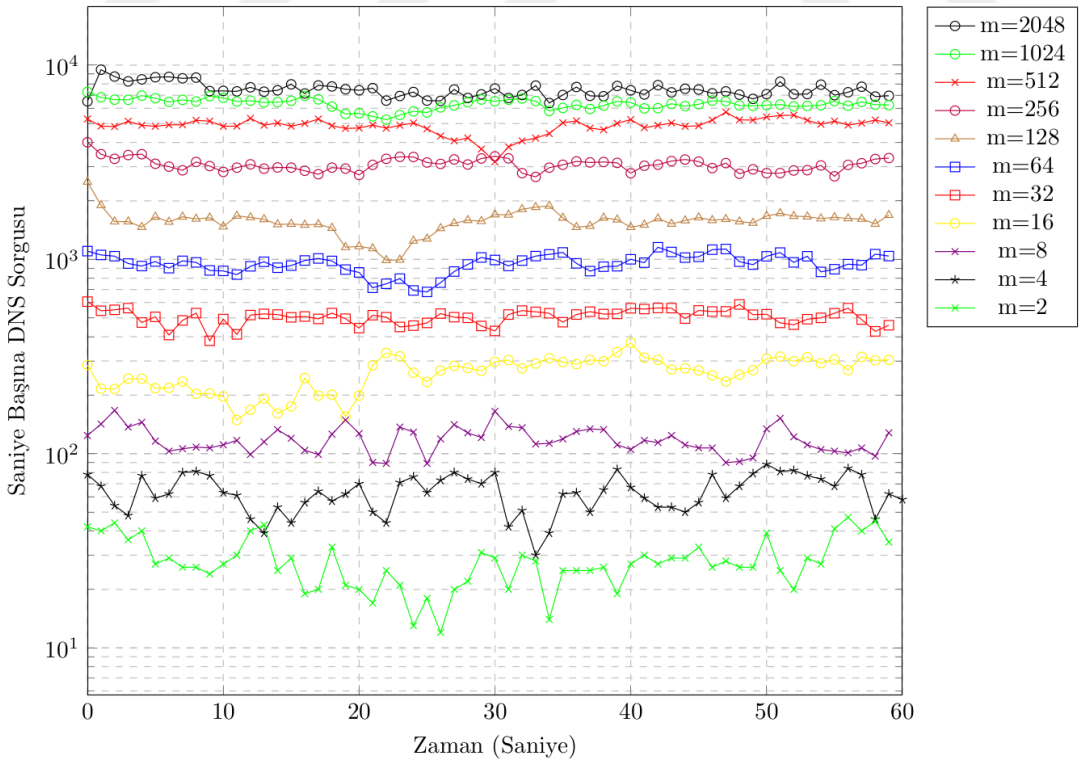
Şekil 5.16. Yetkili ad sunucusunda toplam eşzamanlı DNS sorgusu istek limitleme

Toplam eşzamanlı istek limitleme performansının test edilmesi amacıyla saniyede 20 000 DNS sorgusu 60 saniye süresince değişen Bloom filtresi (m) boyutlarında 2^{24}(≈ 1,677 milyon) IP adresinden DNS sorguları yapılarak testler gerçekleştirilmiştir.

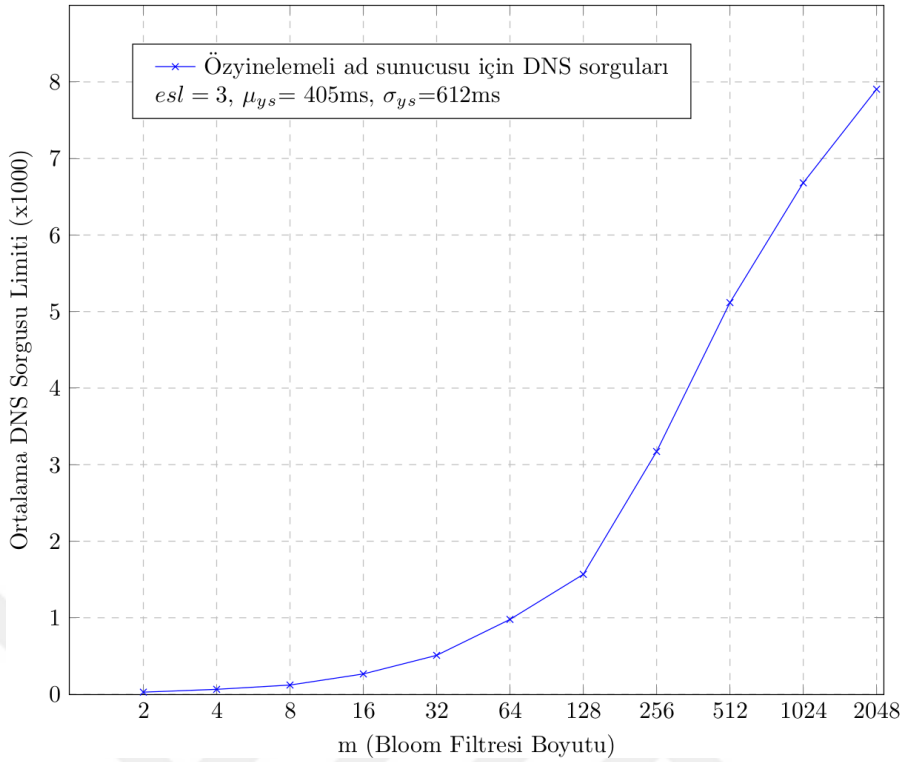
Yetkili ad sunucusu testleri için esl = 1 olarak sabit tutulurken m parametresinin 2 ile 20 arasındaki farklı değerleri için testler yapılmıştır. Elde edilen sonuçlar Şekil 5.16 ve Şekil 5.17' de görülmektedir. Sonuçlar incelendiğinde artan Bloom filtresi boyutu ile orantılı olarak toplam DNS sorgusu limitinin arttığı görülmektedir. Yapılan testlerde Şekil 5.17'de görüldüğü üzere μ_{ys}= 0,048 ms ve σ_{ys} = 0,045 ms olarak ölçülmüştür.



Şekil 5.17. Yetkili ad sunucusunda toplam sorgu limiti ortalamasının değişimi



Şekil 5.18. Özyinelemeli ad sunucusunda çeşitli *m* değerlerinde toplam DNS sorgusu limitleme

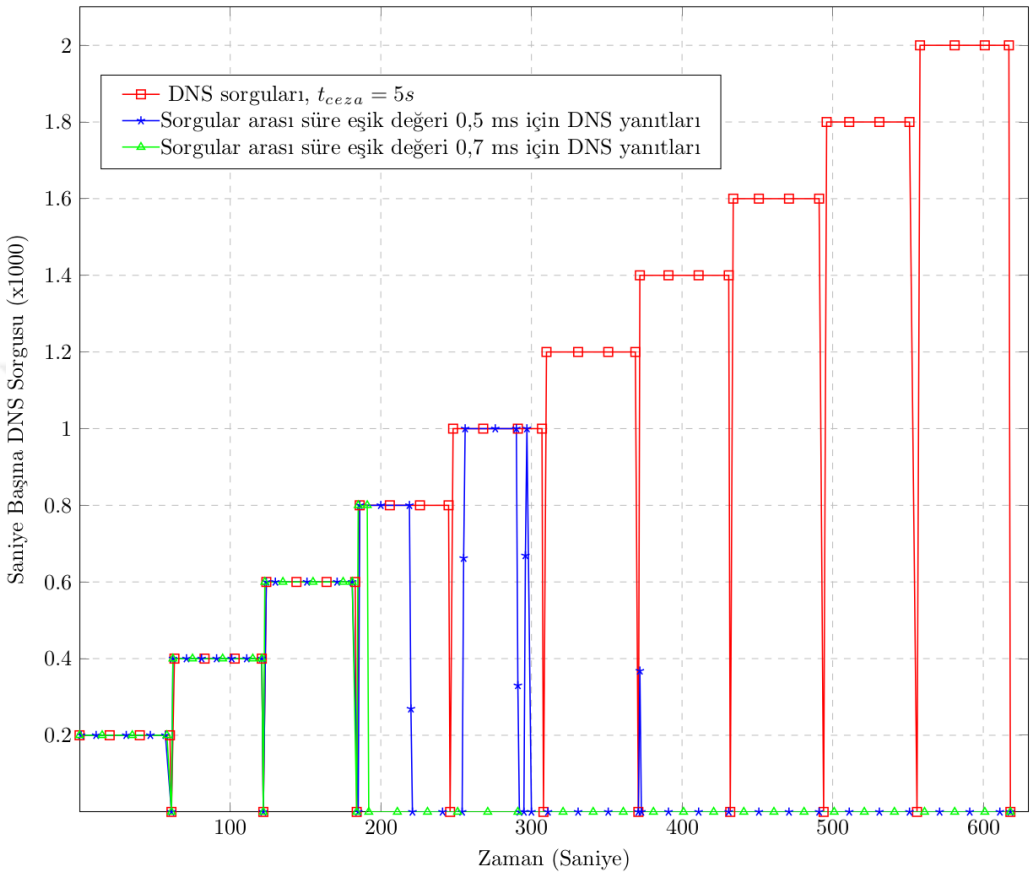


Şekil 5.19. Özyinelemeli ad sunucusunda toplam sorgu limiti ortalamasının değişimi

Özyinelemeli ad sunucusu testleri için toplam istek limitleme performansının test edilmesi amacıyla *esl* = 3 sabit değeri için m değeri 2 ile 2048 arasında değiştirilerek testler gerçekleştirilmiştir. Elde edilen sonuçlar Şekil *5.*18 ve Şekil *5.*19' da yer almaktadır. Elde edilen sonuçlar yetkili ad sunucusu test sonuçları ile karşılaştırıldığında daha yüksek *m* değerleri için daha düşük sorgu limitlerinin oluştuğu görülmektedir. Daha önce yapılan testler ile benzer şekilde istek limitleme değerlerindeki salınımların özyinelemeli ad sunucusu testlerinde daha belirgin olduğu görülmüştür.

DNS sorgularında sorgular arası süre için eşik değeri kullanılarak cezalandırma yaklaşımının test edilmesi amacıyla farklı sorgulama oranlarında testler gerçekleştirilmiştir. Yapılan testlerde tek istemciden saniyedeki DNS sorgu sayısı değeri 200 ile 2000 arasında 200 artımlarla ve her artım 60 saniye sürecek şekilde tekrarlanmıştır. Sorgular arası süre için eşik değeri 0,5 ms ve 0,7 ms olmak üzere iki farklı değerde test edilmiştir. Cezalandırma süresi t_{ceza} = 5 saniye olarak seçilmiştir. Düşük cezalandırma süresi seçilmesinin sebebi cezalandırma yaklaşımının farklı oranlarda birden çok kez tetiklenebilmesinin görülmesidir.

Şekil 5.20' de elde edilen sonuçlardan görüleceği üzere artan sorgular arası süre ile daha düşük istek oranlarında cezalandırma yaklaşımı tetiklenmektedir. Elde edilen sonuçlar önerilen cezalandırma yaklaşımının başarılı bir şekilde çalıştığını göstermektedir.



Şekil 5.20. Sorgular arası süre kullanılarak artan sorgu yoğunluğunun cezalandırılması

Yapılan testlerden elde edilen sonuçlardan açıkça görülmektedir ki önerilen yöntem hem yetkili ad sunucuları hem de özyinelemeli ad sunucuları için DNS sorgusu limitlemeyi iki farklı seviyede başarılı bir şekilde gerçekleştirmektedir. Özyinelemeli ad sunucuları için hem yanıtlanma süresi hem de yanıtlanma süresi varyansının artmasından ötürü DNS sorgu limitlemedeki salınımlar daha belirgin olarak oluşmaktadır. Önerilen yaklaşımın DNS sunucusunun aktif istek durumuna ve dolayısıyla bekleyen iş yüküne adaptif olarak çalıştığı göz önünde bulundurulduğunda salınımların oluşması beklenen bir durumdur.

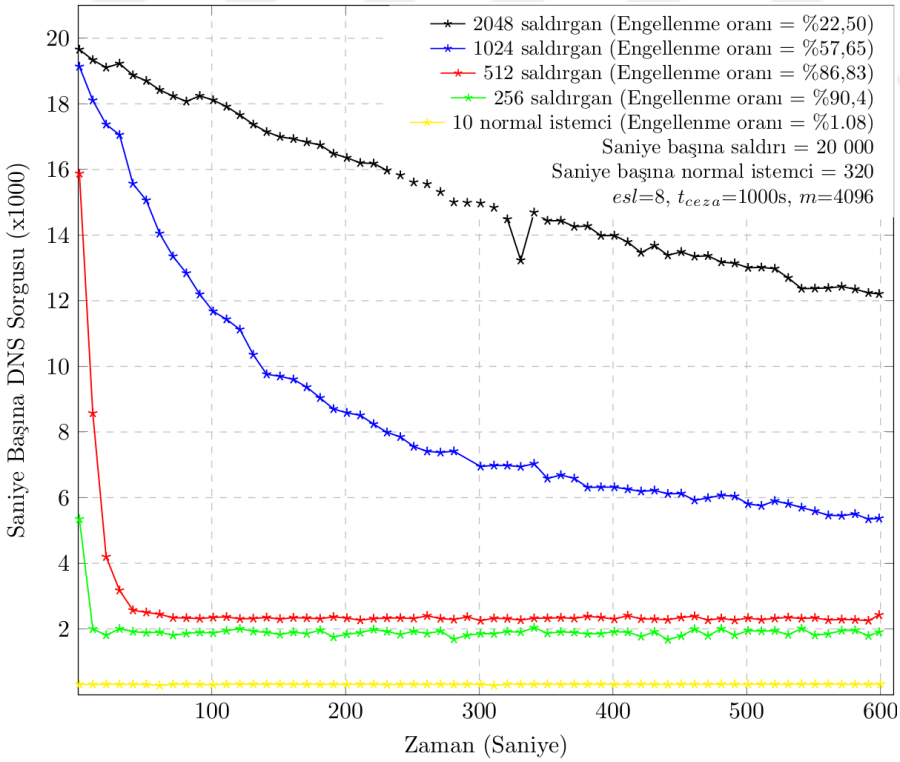
5.5.2. Yoğun DNS sorgularına karşı istek limitleme performansı

Bir önceki 5.5.1 bölümünde yapılan testlerde önerilen DNS istek limitleme yaklaşımının performansını etkileyecek çeşitli parametrelerin tek başına etkilerini incelemeye yönelik

testler yapılarak önerilen yaklaşımın beklendiği gibi çalıştığı görülmektedir. Bu bölümde yapılan testlerde ise DNS sunuculara yapılan yoğun sorgular neticesinde aşırı kaynak tüketimi sebebiyle hizmet kesintisini hedefleyen DNS saldırılarına karşı önerilen istek limitlemenin performansı incelenmiştir.

Önerilen istek limitleme yaklaşımının performansı hem özyinelemeli hem de yetkili ad sunucularına karşı yapılan saldırılar için incelenmiştir. Gerçekleştirilen testlerde değişen oranda saldırgan IP adresi kullanılarak saniyede 20 000 DNS saldırı isteği ile beraber 32 normal istemcinin her birinden saniyede 10 DNS isteği yapılacak şekilde testler gerçekleştirilmiş ve cezalandırma süresi t_{ceza}= 1000 saniye olarak seçilmiştir.

Yetkilendirilmiş ad sunusu için yapılan testlerde ortalama yanıtlanma süresi 0,4 ms civarında ölçülürken DNS zaman aşımı süresi 10 ms olarak belirlenmiştir. Farklı saldırgan IP adresi için yapılan her bir test 600 saniye süresince uygulanarak Şekil 5.21' deki sonuçlar elde edilmiştir.

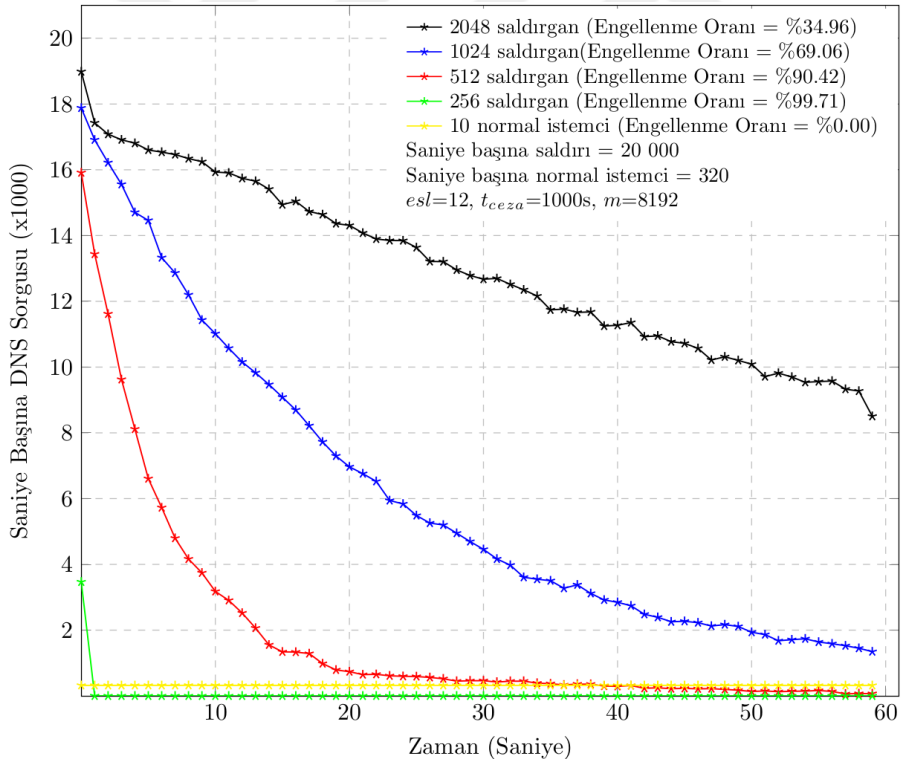


Şekil 5.21. Yetkili ad sunucuları için istek limitleme performansı

Benzer şekilde özyinelemeli ad sunucusu için yapılan testlerde DNS sorgu zaman aşımı süresi 10 saniye olarak seçilmiştir. Bu değerin seçilmesinin sebebi yapılan testlerde

özyinelemeli sorguların 10 saniye süresine kadar sürebildiğinin görülmesidir. Gerçekleştirilen her test 60 saniye süre ile tekrarlanmış ve Şekil 5.22'de görülen sonuçlar elde edilmiştir.

Yapılan test sonuçlarından görüleceği üzere aynı saldırı oranı için saldırgan IP adreslerinin sayısının artması ile saldırgan isteklerine uygulanan istek limitleme sonucu engellenen isteklerin oranı azalmaktadır. Bununla beraber kullanılan cezalandırma yaklaşımı ile beraber zamanla daha fazla saldırgan adresi tespit edilerek engellenmektedir. Her iki DNS sunucusuna yapılan saldırılar karşılaştırıldığında, Şekil 5.22'de istek limitlemenin daha kısa sürede saldırganları cezalandırdığı görülmektedir. Bunun nedeni özyinelemeli sorguların daha fazla sunucu kaynağı gerektirmesidir. Önerilen yaklaşımın DNS sunucusunun bekleyen iş yüküne adaptif çalışması neticesinde özyinelemeli sorgularda daha kısa sürede saldırgan adreslere cezalandırma uygulanmaktadır.



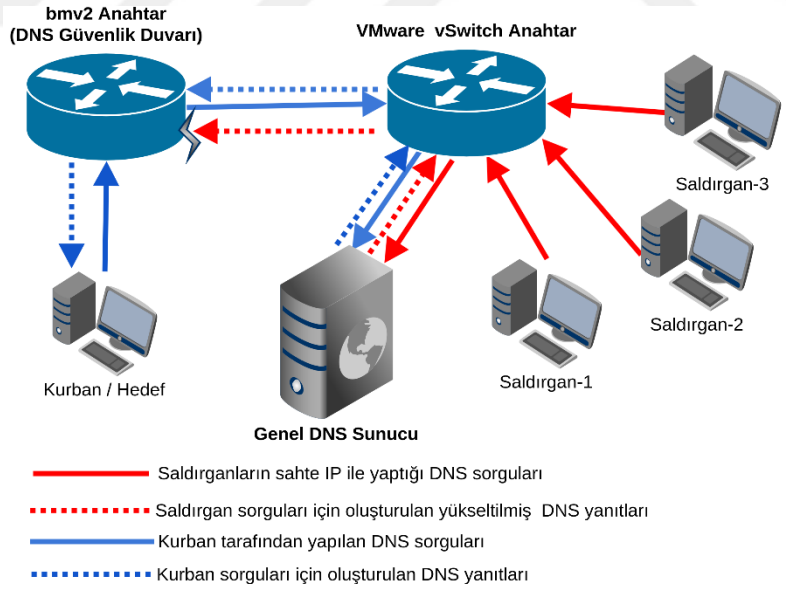
Şekil 5.22. Özyinelemeli ad sunucuları için istek limitleme performansı

5.5.3. DNS güvenlik duvarı performansının incelenmesi

Önerilen DNS güvenlik duvarı yaklaşımının etkinliğinin incelenmesi amacıyla Şekil 5.23'te görülen bağlantı topolojisi oluşturularak testler gerçekleştirilmiştir. DNS sorgularının

durumsal takibinin etkinliğinin ölçülmesi amacıyla saldırgan istemciler ve hedef olarak seçilen istemciden aynı DNS sunucusuna doğru DNS sorguları yapılmıştır. Önerilen yaklaşımda sadece hedef olarak seçilen istemcinin DNS yanıtları iletilirken saldırganların oluşturduğu yansıtılmış DNS yanıtları güvenlik duvarı tarafından engellenmektedir. Yükseltilmiş DNS yanıtlarının oluşturulması amacıyla yüksek boyutlu DNS metin kayıtları (*TXT Record*) oluşturularak oluşturulan yanıtlar için yükseltme oranı 36 olarak ölçülmüştür.

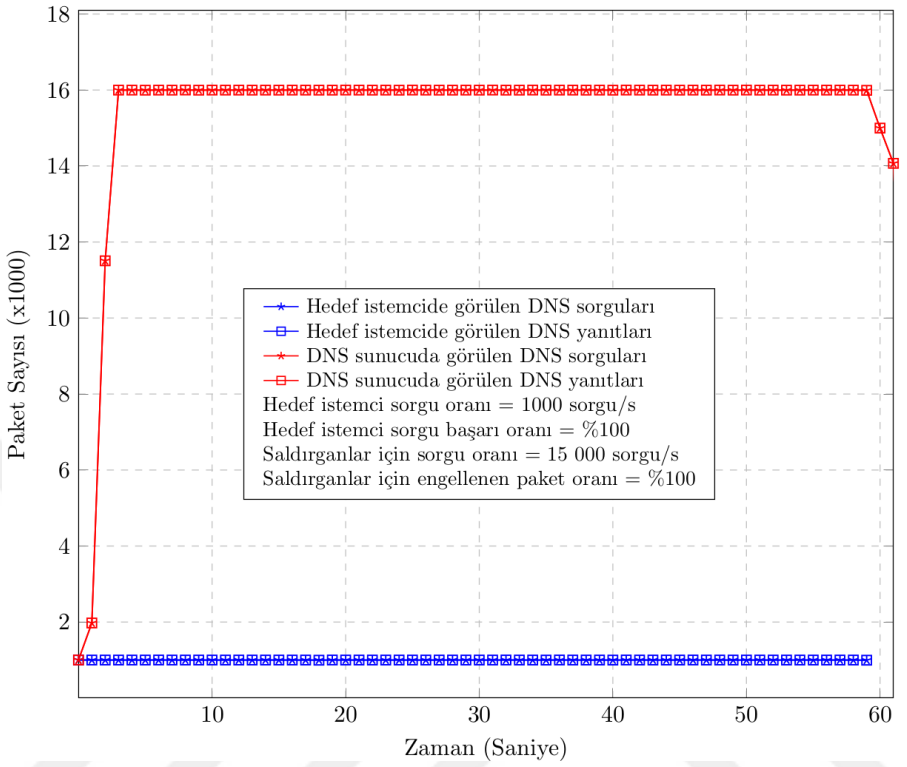
Oluşturulan güvenlik duvarının etkinliği iki farklı durumda test edilmiştir. Bunlardan birincisinde saldırganlar tarafından sahte IP kullanılarak yapılan saniyede 15 000 DNS sorgusu ile beraber hedef istemci tarafından yapılan saniyede 1000 DNS sorgusu için test sonuçları elde edilmiştir. Elde edilen sonuçlarda hedef olarak seçilen istemcinin isteklerinin yanıtlanma oranı ile beraber saldırganların yansıtılmış yanıtlarının hedefe ulaşma oranı incelenmiştir. Gerçekleştirilen test yalnızca DNS isteklerinin durumsal takibinin performansının ölçülmesi hedeflenmiştir, bu sebeple yapılan testlerde yansıtılan DNS yanıtları bu aşamada yükseltilmemiştir. Bunun neticesinde Şekil 5.24'te görülen sonuçlar elde edilmiştir.



Şekil 5.23. DNS Güvenlik Duvarı Test Topolojisi

İkinci durumda ise hem istemci hem de saldırganların DNS sunucusunda yüksek boyutlu metin kayıtlarına sorgu yapması sağlanmıştır. İlk durumda yapılan testler ile benzer şekilde yansıtılmış ve yükseltilmiş saldırı yanıtlarının engellenme durumları ile beraber hedef istemcinin sorgulama doğruluğu incelenmiştir. Buradaki testlerde ise IP parçalanma

kullanan yükseltilmiş DNS yanıtlarının durumsal takibinin etkinliği test edilerek Şekil 5.25'te görülen sonuçlar elde edilmiştir.

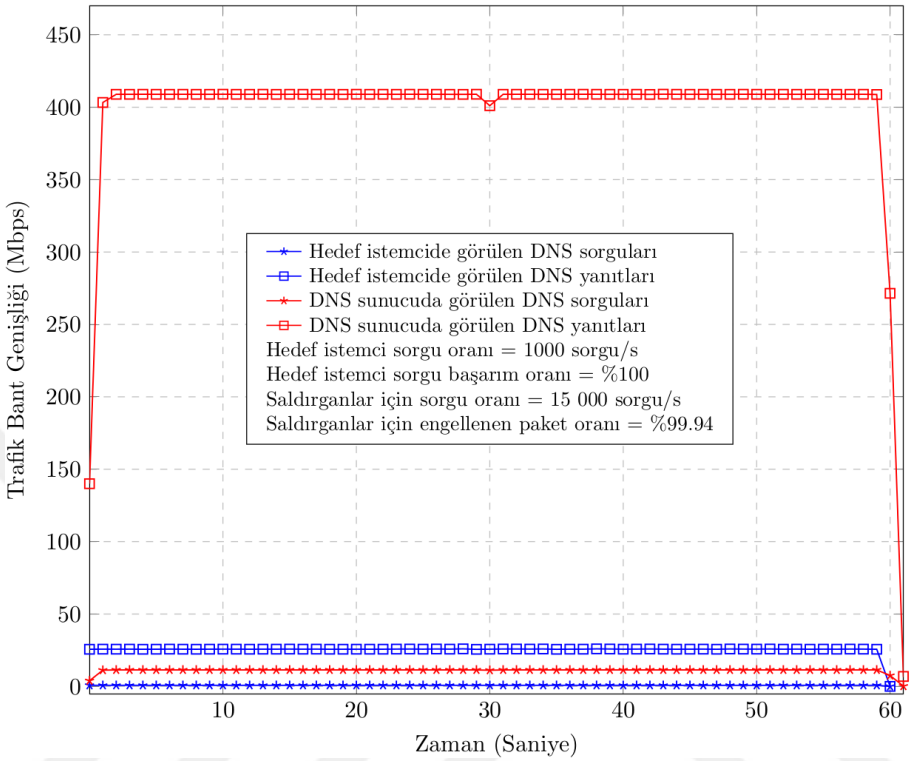


Şekil 5.24. DNS yansıtma saldırısında durumsal takip performansı

IP parçalanma kullanan DNS yanıtlarının durumsal takibinde sadece zamansal veri kontrolü gerçekleştirilerek gelen paketlerin saldırı yanıtlarının bir parçası olup olmadığına bakılmıştır. Her ne kadar her bir DNS yanıtı için son gelen parça (*fragment*) paketi ile ilgili veri GBF veri yapısından silinebilir olsa da bunun gerçekleştirilmesi paket işleme karmaşıklığını artıracağından özellikle saldırı esnasında yüksek oranda gelen paketlerin işlenmesinde P4 anahtar üzerinde yük artışına sebep olabilir. Bu sebeple parçalanmış ve yükseltilmiş DNS yanıt paketlerinin takip edilmesinde sadece zamansal veri güncellik kontrolü düşük hata oranı göz önünde bulundurularak kullanılmıştır.

Önceki 3 bölümünde anlatıldığı üzere uygulamaya özgü GBF tasarımı saniye başına paket oranlarının hesaplanması yerine devir başına paket oranlarının hesaplanmasını gerektirmektedir. Kullanılan bmv2 sanal anahtarının sınırlı paket anahtarlama kapasitesi sebebiyle yüksek oranlardaki trafiklerde paket düşümlerine sebep oluyor olmasına rağmen devir süresi artırılarak aynı devir süresi içerisindeki işlenen paket sayısı artırılarak veri yapısının devir başına doğruluk değerleri incelenebilir. Böylece saniye başına paket sayısı

veya saniye başına veri boyutu değerleri ölçülen devir sayısı başına doğruluk değerleri kullanılarak hesaplanabilir.

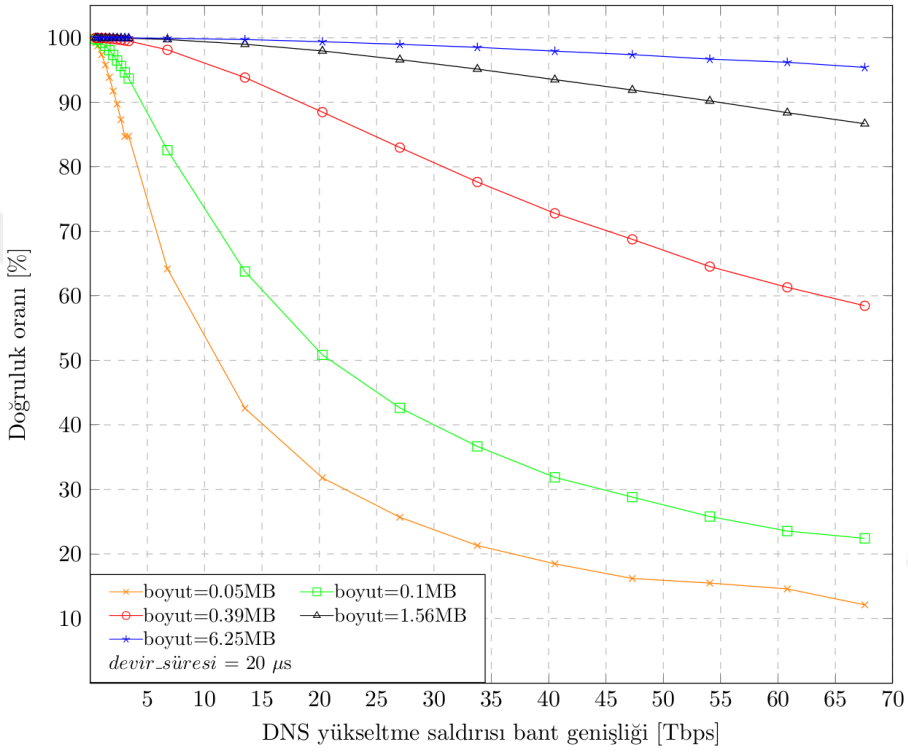


Şekil 5.25. DNS yükseltme saldırısında durumsal takip performansı

Çeşitli devir başına paket oranları için GBF veri yapısının doğruluk değerlerinden yararlanarak önerdiğimiz DNS güvelik duvarı yaklaşımının DNS yükseltme saldırılarına karşı etkinliğini değerlendirmek için aynı doğruluk değerleri kullanılmıştır. Gerçekleştirilen testlerde parçalanmış DNS yanıtları için parça paketler arası sürenin maksimum 7µs olduğu görülmüş ve bu sebeple devir süresi 20 µs olarak kullanılmıştır. Bu değer aynı zamanda parçalanmış DNS yanıt paketlerinin takibinde parça paketler arası zaman aşım süresi olarak nitelendirilebilir. Sonuç olarak Şekil 5.26'da görülen doğruluk değerleri elde edilmiştir.

Önerilen veri yapısının çeşitli GBF boyutları ve DNS yükseltme saldırı hacimleri için doğruluk oranları Şekil 5.26' da görülmektedir. Parçalanmış ve saldırı amacıyla gönderilmiş olan DNS yanıt paketlerinin takibinde *Fragment_BF* olarak isimlendirilen veri yapısının bir engelleme listesi olarak kullanılmaktadır. Nitekim artan saldırı yoğunluğu ve düşen doğruluk oranları esasen saldırı trafiklerinin engellenmeyeceği anlamına gelmemektedir. Bloom filtresi veri yapısı doğası gereği yanlış negatif sonuçlar üretmez. Bu veri yapısının engelleme listesi olarak kullanılması ile beraber prensip olarak bu durum bütün DNS yükseltme

paketlerinin engellenmesini sağlar. Ancak azalan doğruluk oranı, saldırı esnasında IP parçalanmasını kullanan diğer farklı bazı uygulamalar için saldırı dışı paketlerinde saldırı paketi olarak (yanlış pozitif) tanımlanabileceğini gösterir. Bunun temel nedeni olasılıksal veri yapısının kullandığı özetleme fonksiyonlarındaki çakışmalardır. Bu nedenle önerilen yaklaşımın tüm DNS yükselme saldırı paketlerini etkili bir şekilde engelleyebileceği söylenebilir.



Şekil 5.26. Çeşitli DNS yükseltme saldırısı boyutları için GBF doğruluk oranının değişimi

Parçalanmış DNS yanıtlarının takibinde belirtilmesi gereken tek bir istisna durum oluşabilir. Önerilen yaklaşım ilk parça paketinde bulunan üst katman verilerini kullanarak saldırı paketlerini tespit etmekte ve devamında gelen aynı akışa ait parça paketler ile beraber engellemektedir. Şayet önerilen yöntemi barındıran programlanabilir veri katmanına aynı DNS yanıtına ait herhangi bir parça paket ilk parça paketinden önce ulaşırsa, bu paket ile ilgili veri henüz *Fragment_BF* dizisine eklenmemiş olacağından engellenmeyecektir.

6. SONUÇ VE ÖNERİLER

Bu tez çalışması kapsamında programlanabilir veri katmanının sağladığı üstünlükler kullanılarak çeşitli saldırı yöntemlerine karşı kullanılmak üzere güvenlik fonksiyonları gerçekleştiriliş ve önerilen fonksiyonların etkinlikleri yapılan testler neticesinde ortaya koyulmuştur. Bu bağlamda özellikle vurgulamak gerekir ki, programlanabilir veri katmanlarının sağlamış olduğu hat hızında paket işleyebilme kabiliyeti sayesinde yüksek yoğunluklu ve dağıtık servis engelleme saldırılarına karşı etkin fonksiyonların geliştirilmesinde oldukça güçlü savunma yöntemlerinin hayata geçirilmesini mümkün hale getirmektedir. Bu tez kapsamında önerilen güvenlik fonksiyonları sağlanan bu üstünlükten hareketle gerçekleştirilmiştir. Ayrıca başlangıçta ortaya konan araştırma soruları ile ilgili olarak aşağıdaki sonuçlara ulaşılmıştır.

1. Gerçekleştirilen bu tez çalışması kapsamında TCP bağlantı kurulumu görevi arka uç sunucudan programlanabilir veri katmanına başarılı bir şekilde aktarılarak, arka uç sunucunun bağlantı kurulumu ve sonlandırma paketleri dışındaki istek paketlerini işlemesi sağlanmıştır. Bu sayede hem arka uç sunucunun işlemesi gereken paket oranı azaltılmış hem de TCP bağlantı kurulumundaki zafiyetin sömürülmesinin engellenmesi için gereken savunma mekanizmalarının programlanabilir veri katmanında gerçekleştirilebilmesi sağlanmıştır.

2. Programlanabilir veri katmanlarının sunduğu düşük depolama alanı sebebiyle ağ paketlerinin durumsal takibinin alan-etkin bir yaklaşımla gerçekleştirilmesi için yeni bir veri yapısı olarak güncellenebilir Bloom filtresi veri yapısı oluşturulmuştur. Yapılan testlerden elde edilen sonuçlardan önerilen GBF veri yapısının düşük depolama alanı kullanımı ile beraber yüksek doğruluk sağladığı görülmektedir. TCP bağlantı kurulum paketlerinin durumsal takibinde önerilen veri yapısının sağladığı zamansal veri güncellik kontrolü kullanılarak gelen isteklerin doğrulanması sağlanmış ve değişen istek oranları ve dizi boyutları için doğruluk değişimleri elde edilmiştir. Gerçekleştirilen testlerde programlanabilir veri katmanı üzerinde 6,25 MB depolama alanı kullanılarak yaklaşık 3,4 Mpps SYN isteğinin durumsal takibinin %95,4 doğrulukla gerçekleştirildiği gösterilmiştir. Bu doğruluk değeri yaklaşık 15 Mpps SYN isteği için %80 olarak elde edilmiştir.

3. Önerilen GBF veri yapısının genişliği artırılarak 2 bit olarak belirlenmiş ve bunun yanında zaman damgası alanı eklenerek veri yapısı üzerinde zaman bazlı kontrollerin yapılabilmesi sağlanmıştır. Gerçekleştirilen testlerde bağlantı takibinde GBF veri yapısının sonlanan bağlantıları silebilme kabiliyeti sayesinde Hill ve arkadaşları tarafından [25] gerçekleştirilen bu alandaki en güncel çalışma ile kıyaslandığında önerilen yöntemin 399 kat daha düşük bir dizi uzunluğu ile %99 üzeri doğruluk sağladığı görülmüştür. Üstelik bu doğruluğun zamanla düşmesi gibi bir durum da söz konusu değildir.

4. SYN saldırılarında programlanabilir veri katmanlarında gerçekleştirilen güncel literatür çalışmaları incelendiğinde bunların ilave paket düşümleri ve gönderimleri gerektirdiği bu nedenle önerilen yaklaşımların bağlantı kurulumu için gereken paket sayılarını artırarak ağdaki paket oranını artırdığı görülmektedir. Üstelik bu durumda kullanılan SYN savunması yaklaşımının saldırganlar tarafından kolaylıkla tespit edilmesi ve atlatılabilmesi söz konusudur. Önerilen SYN savunması yaklaşımında GBF ve bağlantı havuzunun birlikte kullanımı gerçekleştirilmiştir. Bu sayede istemciler tarafından önerilen savunma yönteminin atlatılabilmesi olasılığı ve aynı zamanda ilave paket gönderimi veya paket düşürme ya da bağlantı sıfırlama gibi durumlar ortadan kaldırılmıştır.

5. DNS sunucuların yapılan servis engelleme saldırılarına karşı saldırı paketlerinin henüz DNS sunucusuna ulaşmadan ağ katmanında proaktif olarak engellenmesi amaçlanmış ve bu kapsamda programlanabilir veri katmanında iki farklı yöntem önerilerek bunların etkinliği yapılan testler ile incelenmiştir. Gerçekleştirilen her iki yöntemde durumsuz olarak nitelendirilen UDP protokolünü kullanan DNS paketlerinin durumsal takibi programlanabilir veri katmanında gerçekleştirilmiştir. İstek limitleme yaklaşımında tespit edilen saldırganlar önerilen GBF temelli cezalandırma yaklaşımıyla engellenerek saldırı isteklerinin DNS sunucusuna erişiminin kısıtlanması ağ katmanında gerçekleştirilmiş.

6. DNS istek limitleme yaklaşımında zamana dayalı istek limitleme yaklaşımı yerine, istemci başına eşzamanlı ve aktif istekler programlanabilir veri katmanında takip edilerek DNS sunucunun henüz yanıt vermediği isteklerin sayısı üzerinden limitleme gerçekleştirilmiştir. Bu yaklaşım önerilen yöntemin aktif istek oranına adaptif

çalışmasını sağlayarak, RRL yaklaşımının aksine, hem yetkili ad sunucularında hem de özyinelemeli ad sunucularında kullanılabilmesini mümkün hale getirmiştir. Yapılan testlerde özyinelemeli ad sunucuları için yetkili ad sunucularına nazaran istek limitlemenin, saldırganların sunucunun iş yükünü artırması sebebiyle, daha kısa sürede gerçekleştiği görülmüştür.

7. Önerilen istek limitleme yaklaşımı aktif sorguların sınırlandırılmasında hem istemci IP adresi başına hem de toplam isteklerin oranı üzerinden limitleme yapılabilmesini sağlamaktadır. Bu sayede özellikle saldırı amacıyla dağıtık olarak yüksek oranda gelen isteklere rağmen DNS sunucusu sadece yanıtlayabileceği oranda bir sorgulama yüküne maruz kalarak dağıtık servis engelleme saldırıları esnasında bile adres çözümleme işleminde kesinti olmasını engeller. Her ne kadar bu yaklaşım saldırgan olmayan istemcilerden gelen istekler içinde paket düşürmelere sebep olabilse de DNS sunucunun saldırıdan ötürü tamamen yanıt veremez duruma gelmesini engellemektedir. Gerçekleştirilen testlerde değişen dizi boyutu ve eşzamanlı sorgu limiti oranları için limitleme oranlarının değişimine yönelik sonuçlar elde edilmiştir.

8. Önerilen DNS güvenlik duvarı yaklaşımı, birebir eşleştirme temelli olarak DNS paketlerinin durumsal takibinin programlanabilir veri katmanında yapılması ve yapılmayan DNS sorguları için gelen DNS yanıtlarının engellenmesi esasına dayanmaktadır. Önerilen yaklaşımda uygulama seviyesinde durumsal takip gerçekleştirilerek yüksek doğrulukla saldırı paketlerinin engellenmesi sağlanmıştır. Gerçekleştirilen testlerde hem kurban istemcinin meşru DNS sorguları için hem de saldırgan istemcilerin yansıtma aracı olarak aynı DNS sunucusunu kullanmasına rağmen yapılan isteklerin uygulama seviyesinde durumsal takibi sayesinde yüksek doğrulukta sonuçlar elde edilmesi sağlanmıştır.

9. UDP üzerinden DNS yanıt paketlerinde IP parçalanmanın kullanımı IP parçalanma standardının en yaygın kullanımı olarak kabul edilmektedir [45]. Fakat parçalanmış yanıt paketleri incelendiğinde birinci dışındaki diğer parça paketleri içerisinde port numarası, DNS işlem numarası gibi üst katman başlık verileri yer almaz. Üstelik bu tür parçalanmış yanıt paketleri DNS yükseltme saldırılarında kullanılır [46]. Bu durumda parçalanmış DNS yanıt paketleri birebir eşleştirme davranışına uymaz ve aynı zamanda birinci dışındaki diğer parça paketleri içerisinde üst katman bilgisi

olmaması sebebiyle paketlerin ağ üzerinde takibi ve ayrıştırılmasını zor hale getirir. Bu nedenle yeni bir yaklaşım olarak parçalanmış DNS yanıt paketlerinin durumsal takibi için *Fragment_BF* olarak isimlendirilen yeni bir GBF temelli veri yapısı oluşturularak saldırı paketlerinin durumsal takibi ve engellenmesi sağlanmıştır. Önerilen yaklaşımda kullanılan veri yapısı ile değişen saldırı oranları ve BF boyutu için sağladığı doğruluk değişimleri elde edilmiştir.

10. GBF veri yapısı kullanılarak önerilen DNS savunma fonksiyonları için alan-etkin algoritmaların gerçekleştirilmesi sağlanmıştır. Parçalanmış DNS yanıt paketlerinin durumsal takibinde, parça paketler arası sürenin mikro saniyeler mertebesinde olması ve durumsal takip için çok düşük devir süresi gerektirmesi sayesinde, çok düşük depolama alanı kullanımı ile yüksek oranda paketler önerilen GBF veri yapısı ile yüksek doğrulukla takip edilebilmektedir.

DNS saldırılarına karşı önerilen güvenlik duvarı için gerçekleştirilen testlerde hem hedef sistemin hem de saldırganların aynı DNS sunucusunu kullanmasına rağmen önerilen yaklaşımın uygulama seviyesinde çalışması sayesinde saldırı paketlerini yüksek doğrulukla engellediği görülmüştür. Üstelik çok yüksek boyutlara ulaşabilen DNS yükseltme saldırıları düşünüldüğünde, önerilen GBF veri yapısının çok düşük depolama alanlarında çok yüksek başarım gösterdiği sonucuna ulaşılmıştır. Yapılan testler sonucunda devir süresinin 20µs olarak seçilmesi ile 6,25 MB depolama alanı gerektiren GBF veri yapısı ile yaklaşık 67,5 Tbps boyutunda yükseltilmiş DNS yanıt paketlerinin %95,4 doğruluk ile engellenebileceği sonucuna ulaşılmıştır. Ayrıca yapılan testlerdeki kullanılan topolojide DNS güvenlik duvarı doğrudan hedef sistemin önünde konumlandırılmış olmasına rağmen esasen bu yöntemin yükseltme aracı olarak kullanılan özyinelemeli DNS sunucusu ve hedef sistem arasındaki herhangi bir ağ anahtarı üzerinde kullanılabilmesi mümkündür. Önerilen fonksiyonun kullanılacağı veri katmanı üzerinden saldırı amacıyla oluşturulan DNS sorgu paketlerinin geçmemesi gerekliliğinin sağlandığı herhangi bir noktada bu fonksiyonu kullanılabilir. Bu açıdan incelendiğinde bu fonksiyonu kullanmak isteyen kuruluşlar için doğrudan internet çıkış noktalarına konumlandırabilirler. İnternet servis sağlayıcıları düşünüldüğünde ise, benzer olarak kendi ağ yapılarına yükseltilmiş DNS saldırı yanıtlarının girişini önlemek amacıyla diğer servis sağlayıcılar ile olan ara bağlantı noktasındaki anahtarlama cihazlarında (*leaf router*) bu fonksiyonu konumlandırarak koruma sağlayabilirler.

Bu tez çalışması kapsamında yapılan testlerde programlanabilir veri katmanı olarak kullanılan bmv2 anahtarın yüksek paket anahtarlama kapasitesine sahip olmaması sebebiyle gerçekçi bir performans değerlendirmesini zorlaştırdığını belirtmek gerekir. SYN saldırıları için yapılan testlerde artan paket oranı ile CPU kaynak tüketiminin arttığı ve SYN paketlerinin oranının saniyede yaklaşık 75 000 değerini aşması ile beraber beklenmedik paket düşürmelerin oluşmaya başladığı gözlemlenmiştir. Kullanılan bmv2 sanal anahtarın ilgili bilgi sayfasında [107] yüksek oranda paket anahtarlama için uygun olmadığı ve geliştirme ortamlarında kullanımının önerildiği belirtilmiştir. Bu nedenle CPU aşırı yüklenme durumlarını ve beklenmedik paket düşümlerini önlemek amacıyla bmv2 anahtarın CPU tüketimi takip edilerek daha düşük SYN saldırıları için testler gerçekleştirilmiştir. Benzer şekilde önerilen DNS fonksiyonları için gerçekleştirilen performans testleri öncesinde oluşturulan test ortamının DNS paket anahtarlama kapasitesinin ölçülmesine yönelik testler gerçekleştirilmiş ve paket düşümlerinin yaşanmadığı anahtarlama oranlarında önerilen yöntemler test edilmiştir. Her ne kadar yapılan testlerde birim zaman başına paket oranları bmv2 anahtarın sınırlamaları sebebiyle artırılamamış olmasına rağmen zamansal veri güncelliği sağlayan ve devir süresini esasen zaman penceresi olarak kullanan GBF veri yapısı kullanılması sayesinde veri yapısında aynı devir süresi içerisinde daha fazla veri yazılması sağlanarak veri yapısının sağladığı doğruluk değerleri yapılan testlerden elde edilmiştir. Elde edilen devir başına doğruluk değişim değerleri, uygulamaya özgü olarak belirlenen devir süresi kullanılarak, devir başına doğruluk değerleri birim zaman için yeniden hesaplanmış ve önerilen yöntemler için doğruluk değişimleri elde edilmiştir. Bu çalışmada önerilen yöntemler ve elde edilen performans ölçümlerinin daha da geliştirilmesi amacıyla bu alanda araştırma yapmak isteyen araştırmacılar tarafından gerçek fiziksel donanımlar kullanılarak buradaki önerilen yöntemlerin etkinliğinin incelenmesi gerçekleştirilebilir. Bunun yanı sıra GBF veri yapısında kullanılan özetleme fonksiyonları düşünüldüğünde bmv2 anahtarın desteklediği iki farklı özetleme fonksiyonu sınırlaması sebebiyle yapılan bütün testler için iki özetleme fonksiyonu kullanılmıştır. Bunun yanı sıra farklı sayıda özetleme fonksiyonu kullanarak veya aynı özetleme fonksiyonları için farklı giriş verileri belirlenerek kullanılan özetleme fonksiyonu sayısı arttırabilir ve böylece veri yapısının doğruluğuna olan katkısı incelenebilir.

SYN saldırılarına karşı geliştirilen savunma yaklaşımında veri katmanı esasen TCP bağlantısı açısından bir vekil gibi davranarak istemci ve sunucu arasında iki farklı TCP bağlantısı üzerinden bağlantı havuzu yaklaşımını kullanarak veri iletimini gerçekleştirir. Veri

iletimi esnasında TCP sıra ve onay numarası değerleri çift taraflı olarak her paket için dönüştürülerek iletilir. Gelecek çalışmalarda önerilen yaklaşımın TCP protokolünün kullandığı tıkanıklık (*congestion*) ve akış (*flow*) kontrol mekanizmalarına olan etkisi incelenebilir.

Önerilen GBF veri yapısı bu tez çalışması kapsamında TCP üçlü el sıkışma mekanizmasının durumsal takibinde, TCP bağlantılarının durumsal takibinde, DNS protokolünde isteklerin adaptif olarak limitlenmesinde, yükseltilmiş DNS yanıtlarının birebir eşleştirme yaklaşımıyla sorgu ve yanıtların durumsal takibinde, birebir eşleştirmeye uymayan parçalanmış DNS yanıt paketlerinin durumsal takibinde, tespit edilen saldırganların zaman bazlı cezalandırılmasında ve DNS sorguları için sorgular arası sürenin ölçülmesinde kullanılmıştır. Kullanılan bütün bu fonksiyonların gerçekleştirilebilmesinde sağladığı alan-etkin depolama ve sabit sorgulama süresi ile beraber önerilen veri yapısının hem verilerin silinebilmesine imkan sağlarken hem de zamansal veri güncellik kontrolü sağlamaktadır. Ağ protokollerinin yapısı düşünüldüğünde birçok protokol ön tanımlı zaman aşım sürelerine sahiptir. Ağ katmanındaki yapılan saldırıların esasen ağ protokollerinin zafiyetlerini kullandığı düşünülürse, önerilen GBF veri yapısı daha farklı birçok protokole özgü savunma yöntemlerinde kullanılabileceği değerlendirilmektedir.

DNS servisi için önerilen adaptif istek limitleme yaklaşımının farklı uygulamalar için kullanılabilmesi mümkündür. Örneğin web sunucularında benzer bir yaklaşımla yoğun HTTP isteklerine karşı istemci IP adresi başına istek limitleme gerçekleştirilebilir. Benzer şekilde, web uygulamaları için yavaş istek saldırısı [114] *(slow HTTP)* olarak bilinen saldırı yöntemine karşı da önerilen istek limitleme yaklaşımı kullanılabilir. Bu saldırı yönteminde yoğun istekler yerine normal istemci davranışına benzer oranlarda istekler yapılır ve bu nedenle saldırı isteklerinin tespiti normal istemci davranışına benzediği için daha zordur. Burada yapılan isteklerin sonlandırılmadan aktif tutulması ve aynı zamanda benzer yeni istekler yapılması amaçlanarak sunucu kaynaklarının tamamının saldırgan isteklerine tahsis edilmesi hedeflenmektedir. Bu sayede yeni gelen istemci isteklerine sunucu yanıt veremez hale gelir. Önerilen istek limitleme yaklaşımı istemci IP adresi başına hem aktif isteklerin limitlenmesini sağlarken hem de çok uzun süreler aktif kalacak istekler için devir süresi temelli zaman aşım süresi kontrolü sağlayabilir.

DNS yükseltme saldırılarına karşı önerilen DNS güvenlik duvarı yaklaşımı benzer yapıdaki farklı yansıtma ve yükseltme saldırılarına karşı kullanılabilmesi de mümkündür. Esasen UDP protokolünü kullanan ve internet üzerinden açık olarak hizmet veren uygulamaların yansıtma aracı olarak kullanılabilir. Örneğin NTP yükseltme saldırılarında açık olarak hizmet veren NTP sunucuları yükseltme aracı olarak kullanılmaktadır. Benzer şekilde son yıllarda ortaya çıkan Memcached uygulamasını yükseltme aracı olarak kullanan saldırılarda söz konusudur. Memcached uygulaması UDP protokolü kullanımını destekler ve açık kaynak olarak dağıtık yapıda yüksek performanslı ön-bellekleme yapılması amacıyla web hizmetlerinde yaygın olarak kullanılmaktadır. Kundo ve arkadaşları tarafından [115] 2022 yılında yayınlanan çalışmada bu saldırı yönteminin gerçekleştirilebilmesi için yükseltilmiş yanıtların oluşturulmasında kullanılabilecek 15 000 üzerinde Memcached kullanan uygulamanın olduğunu göstermiştir. Bu ve buna benzer UDP protokolünü kullanarak hizmet veren farklı uygulamalara yönelik olarak önerilen güvenlik duvarı yaklaşımı kullanılabilir.

Sonuç olarak, gerçekleştirilen bu tez çalışmasında da hedeflendiği üzere, Tbps mertebesinde paket anahtarlama kapasitesine sahip veri katmanlarının sağladığı hat hızında paket işleyebilme kabiliyeti, yoğun ve yüksek oranda ağ saldırılarına karşı güçlü savunma mekanizmalarının hayata geçirilmesine imkan sağlamaktadır. Dolayısıyla bu alanda gerçekleştirilebilecek yenilikçi yaklaşımların gerek ulusal gerekse uluslararası ölçekte ağ güvenliğinin sağlanmasına yönelik olarak yüksek katma değer yaratacağı aşikardır.

KAYNAKLAR

1. A. AlSabeh, J. Khoury, E. Kfoury, J. Crichigno, ve E. Bou-Harb, "A survey on security applications of P4 programmable switches and a STRIDE-based vulnerability assessment", *Computer Networks*, c. 207, s. 108800, Nis. 2022, doi: 10.1016/j.comnet .2022.108800.

2. J. Cao, J. Bi, Y. Zhou, ve C. Zhang, "CoFilter: A high-performance switch-assisted stateful packet filter", *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*, Budapest: Association for Computing Machinery, New York, United States, Ağu. 2018, ss. 9-11.

3. R. Ricart-Sanchez, P. Malagon, J. M. Alcaraz-Calero, ve Q. Wang, "NetFPGA-Based Firewall Solution for 5G Multi-Tenant Architectures", *2019 IEEE International Conference on Edge Computing (EDGE)*, IEEE, Tem. 2019, ss. 132-136. doi: 10.1109/ EDGE.2019.00037.

4. R. Ricart-Sanchez, P. Malagon, J. M. Alcaraz-Calero, ve Q. Wang, "Hardware-Accelerated Firewall for 5G Mobile Networks", *2018 IEEE 26th International Conference on Network Protocols (ICNP)*, IEEE, Eyl. 2018, ss. 446-447. doi: 10.1109/ ICNP.2018.00066.

5. J. Xing, W. Wu, ve A. Chen, "Ripple: A Programmable, Decentralized Link-Flooding Defense Against Adaptive Adversaries", *30th USENIX Security Symposium (USENIX Security 21)*, USENIX Association, Ağu. 2021, ss. 3865-3881. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/xing

6. J. Xing, Q. Kang, ve A. Chen, "NetWarden: Mitigating Network Covert Channels while Preserving Performance", *29th USENIX Security Symposium (USENIX Security 20)*, USENIX Association, Ağu. 2020, ss. 2039-2056. URL: https://www.usenix.org/confe rence/usenixsecurity20/presentation/xing

7. G. Li vd., "NETHCF: Enabling Line-rate and Adaptive Spoofed IP Traffic Filtering", *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, IEEE, Eki. 2019, ss. 1-12. doi: 10.1109/ICNP.2019.8888057.

8. M. Dimolianis, A. Pavlidis, ve V. Maglaris, "A Multi-Feature DDoS Detection Schema on P4 Network Hardware", *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, IEEE, Şub. 2020, ss. 1-6. doi: 10.1109/ICIN4845 0.2020.9059327.

9. T. Datta, N. Feamster, J. Rexford, ve L. Wang, "SPINE: Surveillance Protection in the Network Elements", *9th USENIX Workshop on Free and Open Communications on the Internet (FOCI 19)*, Santa Clara, CA: USENIX Association, Ağu. 2019. URL: https://www.usenix.org/conference/foci19/presentation/datta

10. F. Hauser, M. Haberle, M. Schmidt, ve M. Menth, "P4-IPsec: Site-to-Site and Host-to-Site VPN With IPsec in P4-Based SDN", *IEEE Access*, c. 8, ss. 139567-139586, 2020, doi: 10.1109/ACCESS.2020.3012738.

11. F. Hauser, M. Schmidt, M. Haberle, ve M. Menth, "P4-MACsec: Dynamic Topology Monitoring and Data Layer Protection With MACsec in P4-Based SDN", *IEEE Access*, c. 8, ss. 58845-58858, 2020, doi: 10.1109/ACCESS.2020.2982859.

12. Q. Qin, K. Poularakis, K. K. Leung, ve L. Tassiulas, “Line-Speed and Scalable Intrusion Detection at the Network Edge via Federated Learning”, *2020 IFIP Networking Conference (Networking)*, 2020, ss. 352-360.

13. J. Hypolite, J. Sonchack, S. Hershkop, N. Dautenhahn, A. DeHon, ve J. M. Smith, “DeepMatch: Practical deep packet inspection in the data plane using network processors*.”, Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies*, New York, NY, USA: ACM, Kas. 2020, ss. 336-350. doi: 10.1145/3386367.3431290.

14. G. K. Ndonda ve R. Sadre, “A Two-level Intrusion Detection System for Industrial Control System Networks using P4”, 2018. doi: 10.14236/ewic/ICS2018.4.

15. S. Kaur, K. Kumar, ve N. Aggarwal, “A review on P4-Programmable data planes: Architecture, research efforts, and future directions”, *Comput Commun*, c. 170, ss. 109-129, Mar. 2021, doi: 10.1016/j.comcom.2021.01.027.

16. Bin Xiao, Wei Chen, Yanxiang He, ve E. H.-M. Sha, “An Active Detecting Method Against SYN Flooding Attack”, *11th International Conference on Parallel and Distributed Systems (ICPADS’05)*, IEEE, ss. 709-715. doi: 10.1109/ICPADS.2005.67

17. O. Yoachimik ve J. Pacheco, “DDoS threat report for 2023 Q4”. Son Erişim Tarihi: 11 Şubat 2024. URL: https://blog.cloudflare.com/ddos-threat-report-2023-q4/

18. M. Antonakakis vd., “Understanding the Mirai Botnet”, *26th USENIX Security Symposium (USENIX Security 17)*, Vancouver, BC: USENIX Association, Ağu. 2017, ss. 1093-1110. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessio ns/presentation/antonakakis

19. O. Yoachimik, “Cloudflare blocks an almost 2 Tbps multi-vector DDoS attack”. Son Erişim Tarihi: 11 Şubat 2024. URL: https://blog.cloudflare.com/cloud flare-blocks-an-almost-2-tbps-multi-vector-ddos-attack/

20. O. Yoachimik ve J. Pacheco, “DDoS threat report for 2023 Q3”. Son Erişim Tarihi: 11 Şubat 2024. URL: https://blog.cloudflare.com/ddos-threat-report-2023-q3/

21. P. Ferguson ve D. Senie, “Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing”, *Network Working Group*, RFC 2827.

22. K. Benton, L. J. Camp, T. Kelley, ve M. Swany, “Filtering IP source spoofing using feasible path reverse path forwarding with SDN”, *2015 IEEE Conference on Communications and Network Security (CNS)*, IEEE, Eyl. 2015, ss. 733-734. doi: 10.1109/CNS.2015.7346909.

23. C. Rossow, “Amplification Hell: Revisiting Network Protocols for DDoS Abuse”, *Proceedings 2014 Network and Distributed System Security Symposium, Reston, VA: Internet Society*, 2014. doi: 10.14722/ndss.2014.23233.

24. B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors”, *Commun ACM*, c. 13, sy 7, ss. 422-426, Tem. 1970, doi: 10.1145/362686.362692.

25. J. Hill, M. Aloserij, ve P. Grosso, “Tracking Network Flows with P4”, *2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS)*, IEEE, Kas. 2018, ss. 23-32. doi: 10.1109/INDIS.2018.00006.

26. A. Indra Basuki, D. Rosiyadi, ve I. Setiawan, “Preserving Network Privacy on Fine-grain Path-tracking Using P4-based SDN”, *2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET), IEEE*, Kas. 2020, ss. 129-134. doi: 10.1109/ICRAMET51080.2020.9298588.

27. R. Miao, H. Zeng, C. Kim, J. Lee, ve M. Yu, “SilkRoad”, *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, New York, NY, USA: ACM, Ağu. 2017, ss. 15-28. doi: 10.1145/3098822.3098824.

28. A. Pavlidis, *Automated Monitoring and Security Services in Federated Software-defined Network Infrastructures*, National Technical University of Athens, School of Electrical and Computer Engineering, Athens, 2020. Son Erişim Tarihi: 27 Ocak 2024. URL: https://dspace.lib.ntua.gr/xmlui/bitstream/handle/123456789/50875/20200630_dissertation_APAVLIDIS.pdf

29. M. Dimolianis, A. Pavlidis, ve V. Maglaris, “A Multi-Feature DDoS Detection Schema on P4 Network Hardware”, *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, IEEE, Şub. 2020, ss. 1-6. doi: 10.1109/ICIN4845 0.2020.9059327.

30. K. Friday, E. Kfoury, E. Bou-Harb, ve J. Crichigno, “Towards a Unified In-Network DDoS Detection and Mitigation Strategy”, *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, IEEE, Haz. 2020, ss. 218-226. doi: 10.1109/NetSoft48620 .2020.9165336.

31. J. Zhang, Y. Gao, S. Wen, T. Pan, ve T. Huang, “Loom: Switch-based Cloud Load Balancer with Compressed States”, *2021 IEEE 29th International Conference on Network Protocols (ICNP)*, IEEE, Kas. 2021, ss. 1-11. doi: 10.1109/ICNP52444.202 1.9651928.

32. Q. Wang, M. Pan, S. Wang, R. Doenges, L. Beringer, ve A. W. Appel, “*Foundational Verification of Stateful P4 Packet Processing*”, Dagstuhl: Schloss Dagstuhl, Leibniz Zentrum für Informatik, 2023, ss. 32:1-32:20. doi: 10.4230/LIPIcs.ITP.2023.32.

33. D. Scholz, S. Gallenmüller, H. Stubbe, B. Jaber, M. Rouhi, ve G. Carle, “Me Love (SYN-) Cookies: SYN Flood Mitigation in Programmable Data Planes”, Mar. 2020.

34. D. Scholz, S. Gallenmüller, H. Stubbe, ve G. Carle, “SYN Flood Defense in Programmable Data Planes*”, Proceedings of the 3rd P4 Workshop in Europe*, New York, NY, USA: ACM, Ara. 2020, ss. 13-20. doi: 10.1145/3426744.3431323.

35. P. Vixie ve V. Schryve, “DNS Response Rate Limiting (DNS RRL)”, *Internet Systems Consortium*, Inc. Son Erişim Tarihi: 10 Şubat 2024. URL: https://www.isc.org/pubs/tn/isc-tn-2012-1.txt

36. T. Rozekrans, M. Mekking, ve J. Koning, “Defending against DNS reflection amplification attacks”, Amsterdam, Şub. 2013. Son Erişim Tarihi: 10 Şubat 2024. URL: https://nlnetlabs.nl/downloads/publications/report-rrl-dekoning-rozekrans.pdf

37. E. Winstead, “Respoonse Rate Limiting with BIND”, *Proceedings of the APRICOT Conference. APNOG, APRICOT 2014*, Petaling Jaya, Malaysia, Şub. 2014.

38. G. Kambourakis, T. Moschos, D. Geneiatakis, ve S. Gritzalis, “A Fair Solution to DNS Amplification Attacks”, *Second International Workshop on Digital Forensics and Incident Analysis (WDFIA 2007)*, IEEE, Ağu. 2007, ss. 38-47. doi: 10.1109/WDFIA .2007.4299371.

39. L. Luo vd., “Efficient Multiset Synchronization”, *IEEE/ACM Transactions on Networking*, c. 25, sy 2, ss. 1190-1205, Nis. 2017, doi: 10.1109/TNET.2016.2618006.

40. S. Di Paola ve D. Lombardo, “Protecting against DNS Reflection Attacks with Bloom Filters”, 2011, ss. 1-16. doi: 10.1007/978-3-642-22424-9_1.

41. U. Sattar, T. Naqash, M. R. Zafar, K. Razzaq, ve F. bin Ubaid, “Secure DNS from amplification attack by using modified bloom filters”, *Eighth International Conference on Digital Information Management (ICDIM 2013)*, IEEE, Eyl. 2013, ss. 20-23. doi: 10.1109/ICDIM.2013.6694018.

42. M. Zhang vd., “Poseidon: Mitigating Volumetric DDoS Attacks with Programmable Switches”, *Proceedings 2020 Network and Distributed System Security Symposium*, Reston, VA: Internet Society, Şub. 2020. doi: 10.14722/ndss.2020.24007.

43. Z. Liu vd., “Jaqen: A High-Performance Switch-Native Approach for Detecting and Mitigating Volumetric {DDoS} Attacks with Programmable Switches”, *30th USENIX Security Symposium (USENIX Security 21)*, USENIX Association, Ağu. 2021, ss. 3829-3946.

44. Damas Joao, Graff Michael, ve Vixie Paul, “Extension Mechanisms for DNS (EDNS(0))”, https://www.rfc-editor.org/rfc/rfc6891.txt.

45. Kazunori Fujiwara ve Paul Vixie, “Fragmentation Avoidance in DNS”, https://www.ietf.org/archive/id/draft-ietf-dnsop-avoid-fragmentation-07.html.

46. M. Anagnostopoulos, G. Kambourakis, P. Kopanos, G. Louloudakis, ve S. Gritzalis, “DNS amplification attack revisited”, *Comput Secur*, c. 39, sy PART B, ss. 475-485, Kas. 2013, doi: 10.1016/j.cose.2013.10.001.

47. “Software-Defined Networking: The New Norm for Networks”, Nis. 2012. Son Erişim Tarihi: 03 Şubat 2024. URL: https://opennetworking.org/wpcontent/uploads /2011/09/wp-sdn-newnorm.pdf

48. “OpenFlow Switch Specification - Version 1.5.1”, Mar. 2015. Son Erişim Tarihi: 03 Şubat 2024. URL: https://opennetworking.org/wp-content/uploads/2014/10/op enflow-switch-v1.5.1.pdf

49. P. Bosshart vd., “P4: Programming Protocol-Independent Packet Processors”, *ACM SIGCOMM Computer Communication Review*, c. 44, sy 3, ss. 87-95, Tem. 2014, doi: 10.1145/2656877.2656890.

50. The P4 Language Consortium, “P4(16) Language Specification (version 1.2.4).” Son Erişim Tarihi: Şubat 03, 2024. URL: https://staging.p4.org/p4-spec/docs/P4-16-v1.2.4.html

51. F. Hauser vd., "A survey on data plane programming with P4: Fundamentals, advances, and applied research", *Journal of Network and Computer Applications*, c. 212, s. 103561, Mar. 2023, doi: 10.1016/j.jnca.2022.103561.

52. "P4Runtime Specification (version 1.3.0)". Son Erişim Tarihi: 04 Şubat 2024. URL: https://staging.p4.org/p4-spec/p4runtime/v1.3.0/P4Runtime-Spec.html

53. "P4 Portable NIC Architecture (PNA, version 0.7)". Son Erişim Tarihi: 04 Şubat 2024. URL: https://staging.p4.org/p4-spec/docs/PNA-v0.7.html

54. "P4_16 Portable Switch Architecture (PSA, version 1.2)". Son Erişim Tarihi: 04 Şubat 2024. URL: https://staging.p4.org/p4-spec/docs/PSA-v1.2.html

55. CableLabs, Cisco Systems, Intel, VMware, ve Xilinx, "Telemetry Report Format Specification (Version 2.0)". Son Erişim Tarihi: 04 Şubat 2024. URL: https://staging.p4.org/p4-spec/docs/telemetry_report_v2_0.pdf

56. Alibaba vd., "In-band Network Telemetry (INT) Dataplane Specification (Version 2.1)". Son Erişim Tarihi: 04 Şubat 2024. URL: https://staging.p4.org/p4-spec/docs/INT_v2 _1.pdf

57. F. Paolucci, F. Civerchia, A. Sgambelluri, A. Giorgetti, F. Cugini, ve P. Castoldi, "P4 edge node enabling stateful traffic engineering and cyber security", *Journal of Optical Communications and Networking*, c. 11, sy 1, 2019, doi: 10.1364/JOCN.11.000A84.

58. S. MAHRACH ve A. HAQIQ, "DDoS Flooding Attack Mitigation in Software Defined Networks*", International Journal of Advanced Computer Science and Applications*, c. 11, sy 1, 2020, doi: 10.14569/IJACSA.2020.0110185.

59. Z.-Y. Shen, M.-W. Su, Y.-Z. Cai, ve M.-H. Tasi, "Mitigating SYN Flooding and UDP Flooding in P4-based SDN", *2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS)*, IEEE, Eyl. 2021, ss. 374-377. doi: 10.23919 /APNOMS52696.2021.9562660.

60. T.-Y. Lin vd., "Mitigating SYN flooding Attack and ARP Spoofing in SDN Data Plane", 2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE, Eyl. 2020, ss. 114-119. doi: 10.23919/APNOMS50412.2020.9236 951.

61. F. Musumeci, V. Ionata, F. Paolucci, F. Cugini, ve M. Tornatore, "Machine-learning-assisted DDoS attack detection with P4 language*", ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, IEEE, Haz. 2020, ss. 1-6. doi: 10.1109/ICC40277.2020.9149043.

62. F. Musumeci, A. C. Fidanci, F. Paolucci, F. Cugini, ve M. Tornatore, "Machine-Learning-Enabled DDoS Attacks Detection in P4 Programmable Networks", *Journal of Network and Systems Management*, c. 30, sy 1, s. 21, Oca. 2022, doi: 10.1007/s10922-021-09633-5.

63. P. Golchin, L. Anderweit, J. Zobel, R. Kundel, ve R. Steinmetz, "*In-Network SYN Flooding DDoS Attack Detection Utilizing P4 Switches*", Universität Tübingen, Nis. 2022.

64. S. Yoo, X. Chen, ve J. Rexford, "SmartCookie: Blocking Large-Scale SYN Floods with a Split-Proxy Defense on Programmable Data Planes", *USENIX Security*, Philadelphia, 2024. Son Erişim Tarihi: 04 Şubat 2024. URL: https://www. usenix.org/system /files/sec24summer-prepub-1-yoo.pdf

65. "eBPF and XDP". Son Erişim Tarihi: 04 Şubat 2024. URL: https://docs.suricata.io/en/latest/capture-hardware/ebpf-xdp.html#ebpf-and-xdp

66. C. Deccio, D. Argueta, ve J. Demke, "A Quantitative Study of the Deployment of DNS Rate Limiting", *2019 International Conference on Computing*, Networking and Communications (ICNC), IEEE, Şub. 2019, ss. 442-447. doi: 10.1109/ICCNC.2019 .8685601.

67. J. Mölsä, "Effectiveness of rate-limiting in mitigating flooding DOS attacks", *Proceedings of the Third IASTED International Conference on Communications*, Internet, and Information Technology, Oca. 2004, ss. 155-160.

68. R. Yogesh Patil ve L. Ragha, "A rate limiting mechanism for defending against flooding based distributed denial of service attack", *2011 World Congress on Information and Communication Technologies*, IEEE, Ara. 2011, ss. 182-186. doi: 10.1109/WICT.2011. 6141240.

69. Xiaofeng Wang, Xiaofeng Hu, Fei Wang, Xicheng Lu, ve Jinshu Su, "Unfair rate limiting on traffic aggregates for DDoS attacks mitigation", *IET International Conference on Information Science and Control Engineering 2012 (ICISCE 2012)*, Institution of Engineering and Technology, 2012, ss. 3.51-3.51. doi: 10.1049/cp.2012. 2448.

70. Google, "Google Public DNS for ISPs". Son Erişim Tarihi: 10 Şubat 2024. URL: https://developers.google.com/speed/public-dns/docs/isp

71. C. Sun, B. Liu, ve L. Shi, "Efficient and Low-Cost Hardware Defense Against DNS Amplification Attacks", *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, IEEE, 2008, ss. 1-5. doi: 10.1109/GLOCOM.200 8.ECP.397.

72. Li Fan, Pei Cao, J. Almeida, ve A. Z. Broder, "Summary cache: a scalable wide-area Web cache sharing protocol", *IEEE/ACM Transactions on Networking*, c. 8, sy 3, ss. 281-293, Haz. 2000, doi: 10.1109/90.851975.

73. H. Lim, J. Lee, H. Byun, ve C. Yim, "Ternary Bloom Filter Replacing Counting Bloom Filter", *IEEE Communications Letters*, c. 21, sy 2, ss. 278-281, Şub. 2017, doi: 10.1109/LCOMM.2016.2624286.

74. C. Rothenberg, C. Macapuna, F. Verdi, ve M. Magalhaes, "The deletable Bloom filter: a new member of the Bloom family", *IEEE Communications Letters*, c. 14, sy 6, ss. 557-559, Haz. 2010, doi: 10.1109/LCOMM.2010.06.100344.

75. M. T. Goodrich ve M. Mitzenmacher, "Invertible bloom lookup tables", *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, IEEE, Eyl. 2011, ss. 792-799. doi: 10.1109/Allerton.2011.6120248.

76. S. Cohen ve Y. Matias, "Spectral bloom filters", *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, New York, NY, USA: ACM, Haz. 2003, ss. 241-252. doi: 10.1145/872757.872787.

77. L. Luo, D. Guo, R. T. B. Ma, O. Rottenstreich, ve X. Luo, "Optimizing Bloom Filter: Challenges, Solutions, and Comparisons", *IEEE Communications Surveys and Tutorials*, c. 21, sy 2, ss. 1912-1949, Nis. 2019, doi: 10.1109/COMST.2018.2889329.

78. S. Dharmapurikar, P. Krishnamurthy, ve D. E. Taylor, "Longest Prefix Matching Using Bloom Filters", 2003.

79. M. Mitzenmacher, P. Reviriego, ve S. Pontarelli, "OMASS: One Memory Access Set Separation*", IEEE Trans Knowl Data Eng*, c. 28, sy 7, ss. 1940-1943, Tem. 2016, doi: 10.1109/TKDE.2016.2535286.

80. K. Huang vd., "A Multi-partitioning Approach to Building Fast and Accurate Counting Bloom Filters", *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*, IEEE, May. 2013, ss. 1159-1170. doi: 10.1109/IPDPS.2013.51.

81. J. Lu vd., "Ultra-Fast Bloom Filters using SIMD Techniques", *IEEE Transactions on Parallel and Distributed Systems*, c. 30, sy 4, ss. 953-964, Nis. 2019, doi: 10.1109/TPDS.2018.2869889.

82. Yan Qiao, Tao Li, ve Shigang Chen, "Fast Bloom Filters and Their Generalization", *IEEE Transactions on Parallel and Distributed Systems*, c. 25, sy 1, ss. 93-103, Oca. 2014, doi: 10.1109/TPDS.2013.46.

83. S. Geravand ve M. Ahmadi, "Bloom filter applications in network security: A state-of-the-art survey", *Computer Networks*, c. 57, sy 18, ss. 4047-4064, Ara. 2013, doi: 10.1016/j.comnet.2013.09.003.

84. "Intel® TofinoTM 2". Son Erişim Tarihi: 27 Ocak 2024. URL: https://www.intel.com/co ntent/www/us/en/products/details/network-io/intelligent-fabr ic-processors/tofino-2.html

85. S. Kaur, K. Kumar, ve N. Aggarwal, "A review on P4-Programmable data planes: Architecture, research efforts, and future directions", *Comput Commun*, c. 170, ss. 109-129, Mar. 2021, doi: 10.1016/j.comcom.2021.01.027.

86. "Netronome Agilio CX SmartNICs". Son Erişim Tarihi: 27 Ocak 2024. URL: https://www.netronome.com/products/agilio-cx/

87. "Intel® TofinoTM 3 Intelligent Fabric Processors", Product Brief. Son Erişim Tarihi: 28 Ocak 2024. URL: https://www.intel.com/content/dam/www/central-libraries/us/en/docu ments/2022-05/tofino-3-intelligent-fabric-processors-brief.pdf

88. "BEHAVIORAL MODEL (bmv2)", https://github.com/p4lang/behavioral-model/blob /main/README.md.

89. "BMv2 timestamp implementation notes". Son Erişim Tarihi: 28 Ocak 2024. URL: https://github.com/p4lang/behavioral-model/blob/main/docs/simple_switch.m d#bmv2-timestamp-implementation-notes

90. “Precision Time Protocol, PTP”. Son Erişim Tarihi: 28 Ocak 2024. URL: https://en.wikipedia.org/wiki/Precision_Time_Protocol

91. “Operations on fixed-width bit types (unsigned integers)”. Son Erişim Tarihi: 28 Ocak 2024. URL: https://p4.org/p4-spec/docs/P4-16-v-1.2.3.html#sec-bit-ops

92. K. Gupta ve M. Mathuria, “Improving performance of web application approaches using connection pooling”, *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, IEEE, Nis. 2017, ss. 355-358. doi: 10.1109/ICECA. 2017.8212833.

93. F. Liu, “A Method of Design and Optimization of Database Connection Pool”, *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, IEEE*, Ağu. 2012, ss. 272-274. doi: 10.1109/IHMSC.2012.161.

94. The P4 Language Consortium, “P4(16) Language Specification (version 1.2.3)”. Son Erişim Tarihi: 14 Ocak 2024. URL: https://p4.org/wp-content/uploads/2022/07/P4-16-spec.html#sec-parse-header-stacks

95. “Apache Core Features - KeepAliveTimeout Directive”. Son Erişim Tarihi: 14 Ocak 2024. URL: https://httpd.apache.org/docs/2.4/en/mod/core.html#keepa livetimeout

96. V. Jacobson, B. Braden, ve D. Borman, “ TCP Extensions for High Performance, RFC 1323”. Erişim 14 Şubat 2024. URL: https://www.ietf.org/rfc/rfc132 3.txt

97. “Transmission Control Protocol, RFC 793”. Son Erişim Tarihi: 14 Ocak 2024. URL: https://datatracker.ietf.org/doc/html/rfc793

98. R. Braden, “Requirements for Internet Hosts -- Communication Layers, RFC1122”. Son Erişim Tarihi: 14 Ocak 2024. URL: https://datatracker.ietf.org/doc/html /rfc1122

99. P. Reviriego ve D. Ting, “Breaking Cuckoo Hash: Black Box Attacks”, *IEEE Trans Dependable Secure Comput*, c. 19, sy 4, ss. 2421-2427, Tem. 2022, doi: 10.1109/TDSC. 2021.3058336.

100. P. Reviriego ve O. Rottenstreich, “Pollution Attacks on Counting Bloom Filters for Black Box Adversaries”, *2020 16th International Conference on Network and Service Management (CNSM)*, IEEE, Kas. 2020, ss. 1-7. doi: 10.23919/CNSM50824.2020. 9269076.

101. D. Clayton, C. Patton, ve T. Shrimpton, “Probabilistic Data Structures in Adversarial Environments”, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, New York, NY, USA: ACM, Kas. 2019, ss. 1317-1334. doi: 10.1145/3319535.3354235.

102. “What is a bare metal hypervisor?” Son Erişim Tarihi: 13 Ocak 2024. URL: https://www.vmware.com/topics/glossary/content/bare-metal-hypervisor.html

103. “VMware Virtual Networking Concepts”. Son Erişim Tarihi: 13 Ocak 2024. URL: https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/virtual_networking_concepts.pdf

104. “ab - Apache HTTP sunucusu başarım ölçme aracı”. Son Erişim Tarihi: 13 Ocak 2024. URL: https://httpd.apache.org/docs/2.4/programs/ab.html

105. “Apache JMeter”. Son Erişim Tarihi: 13 Ocak 2024. URL: https://jmeter.apac he.org/

106. “hping3 Usage Example”. Son Erişim Tarihi: 13 Ocak 2024. URL: https://www.kali.org/ tools/hping3/

107. “Performance of bmv2”. Son Erişim Tarihi: 10 Aralık 2023. URL: https://github.com /p4lang/behavioral-model/blob/main/docs/performance.md

108. Puqi Perry Tang ve Tsung-Yuan Charles Tai, 1- Network Traffic Characterization Using Token Bucket Model. IEEE, 1999.

109. K. Schomp, M. Rabinovich, ve M. Allman, “Towards a Model of DNS Client Behavior”, 2016, ss. 263-275. doi: 10.1007/978-3-319-30505-9_20.

110. “VMware virtual switch”, Son Erişim Tarihi: 13 Ocak 2024. URL: https://docs.vmware. com/en/VMware-Smart-Assurance/10.1.0/ip-manager-delopment-guide-101/GUID-C7E 7752E-D122-44EA-83DE-33F690B22313.html.

111. Internet Systems Consortium (ISC), “BIND (Berkeley Internet Name Domain)”, https://www.isc.org/bind/. Son Erişim Tarihi: 10 Aralık 2023. URL: https://gitlab. isc.org/isc-projects/bind9/

112. “dnsmeter”, https://github.com/DNS-OARC/dnsmeter/tree/develop. Son Erişim Tarihi: 10 Aralık 2023. URL: https://github.com/DNS-OARC/dnsmeter/tree/develop

113. S. Mahdavifar vd., “Lightweight Hybrid Detection of Data Exfiltration using DNS based on Machine Learning”, *2021 the 11th International Conference on Communication and Network Security*, New York, NY, USA: ACM, Ara. 2021, ss. 80-86. doi: 10.1145/3507509.3507520.

114. S. Suroto, “A Review of Defense Against Slow HTTP Attack,” *JOIV : International Journal on Informatics Visualization*, vol. 1, no. 4, pp. 127–134, Nov. 2017, doi: 10.30630/joiv.1.4.51.

115. M. Kondo, R. Tanabe, N. Shintani, D. Makita, K. Yoshioka, and T. Matsumoto, “Amplification Chamber: Dissecting the Attack Infrastructure of Memcached DRDoS Attacks,” 2022, pp. 178–196. doi: 10.1007/978-3-031-09484-2_10.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ŞAHİN, Mehmet Emin

Derece	**Eğitim Birimi**	**Mezuniyet Tarihi**
Doktora	Gazi Üniversitesi / Bilişim Sistemleri	2024
Yüksek lisans	İstanbul Teknik Ü. / Elektronik Müh.	2014
Lisans	Selçuk Ü. / Elektrik-Elektronik Müh.	2011
Lise	Dumlupınar Lisesi	2006

İş Deneyimi

Yıl	**Yer**	**Görev**
2018-Halen	TÜBİTAK	Ağ ve Sistem Yöneticisi
2013-2018	NETAŞ Telekomünikasyon AŞ.	Uluslararası Teknik Destek Müh.

Yabancı Dil

İngilizce, İsveçce

Yayınlar

M. E. Şahin and M. Demirci, “Employing the Programmable Data Plane for Effective Defense Against DDoS Attacks Targetting DNS,” Dec. 2023, Under Review.

M. E. Şahin ve M. Demirci, “Programlanabilir veri katmanlarında akış takip yöntemi,” Ağustos 2023, Buluş Başvuru No: 2023/009645, Türk Patent Ens., Gazi Üniversitesi ve TÜBİTAK.

M. E. Şahin and M. Demirci, “ConPoolUBF: Connection pooling and updatable Bloom filter based SYN flood defense in programmable data planes,” Computer Networks, vol. 231, p. 109802, Jul. 2023, doi: 10.1016/J.COMNET.2023.109802.

M. E. Sahin and S. Ozdemir, “Detection of Malicious Requests on Web Logs Using Data Mining Techniques,” in *2019 4th International Conference on Computer Science and Engineering (UBMK)*, IEEE, Sep. 2019, pp. 463–468. doi: 10.1109/UBMK.2019.8907087.



GAZİLİ OLMAK AYRICALIKTIR.