AÇIK KAYNAK KODLU İŞLETİM SİSTEMLERİ ÜZERİNDE PARALEL YÜK DENGELEME MODELİ GELİŞTİRİLMESİ

Devrim SERAL

DOKTORA TEZİ
ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARALIK 2006
ANKARA

Devrim SERAL tarafından hazırlanan AÇIK KAYNAK KODLU İŞLETİM SİSTEMLERİ ÜZERİNDE PARALEL YÜK DENGELEME MODELİ GELİŞTİRİLMESİ adlı bu tezin Doktora tezi olarak uygun olduğunu onaylarım.

Prof. Dr. İnan GÜLER
Tez Yöneticisi

Bu çalışma, jurimiz tarafından oybirliği ile Elektronik ve Bilgisayar Eğitimi Anabilim Dalında Doktora tezi olarak kabul edilmiştir.

Başkan : Prof. Dr. Ziya B. GÜVENÇ

Üye : Prof. Dr. Ömer Faruk BAY

Üye : Prof. Dr. İnan GÜLER

Üye : Prof. Dr. Bilal GÜNEŞ

Üye : Prof. Dr. Ergün KASAP

Tarih : 19/12/2006

Bu tez Gazi Üniversitesi Fen Bilimleri Enstitüsü tez yazılım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Devrim SERAL

AÇIK KAYNAK KODLU İŞLETİM SİSTEMLERİ ÜZERİNDE PARALEL YÜK DENGELEME MODELİ GELİŞTİRİLMESİ

(Doktora Tezi)

Devrim SERAL

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Aralık 2006

ÖZET

Günümüzde, sürekli büyüyen İnternet trafik yükleri sunucu performanslarını etkilemektedir. Bu çalışmada, sunucu sistemler üzerinde İnternet trafiğinin oluşturduğu yükü paylaştırmak/dengelemek/azaltmak için paralel yük dengeleme sistemleri içinde yer alan yeni bir model önerilmiştir. Bu model, ağ trafiği için referans yerelliği prensibini kullanmaktadır. Önerilen model için bir benzetim uygulaması geliştirilmiş ve gerçek trafik verileri üzerinde benzetim yapılmıştır. Benzetim sonuçları, önerilen modelin İnternet trafik yüklerini matematiksel mod işlemine göre daha iyi dengelediğini göstermiştir.

Bilim Kodu : **702.3.006**
Anahtar Kelimeler : **Yük dengeleme, paralel yük dengeleme, referans yerelliği, ağ yük dengeleyici, açık kaynak kod**
Sayfa Adedi : **89**
Tez Yöneticisi : **Prof. Dr. İnan GÜLER**

DEVELOPING PARALLEL LOAD BALANCING MODEL ON OPEN SOURCE OPERATING SYSTEMS

(Ph.D. Thesis)

Devrim SERAL

GAZI UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

December 2006

ABSTRACT

Nowadays, consistent grows of Internet traffic workloads effects the server performance. In this study, a new model represented for parallel load balancer systems which share/balance/reduce Internet traffic loads on server systems. This model uses reference locality principle for network traffic. For proposed model, a simulation application have developed and real traffic trace driven simulations have made. Simulation results showed that, proposed model balance Internet traffic workloads better than mathematical modulus method.

Science Code : **702.3.006**
Keywords : **Load balancing, parallel load balancing, reference locality, network load balancer, open source code**
Page Number : **89**
Thesis Advisor : **Prof. Dr. İnan GÜLER**

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla ben yönlendiren hocam Prof. Dr. İnan GÜLER'e, yine çok kıymetli tecrübelerinden faydalandığım hocam Prof. Dr. Ergün KASAP'a, Gazi Üniversitesi Bilgi İşlem Dairesi Başkanı Ertan Türkmen'e üniversitemiz imkanlarını kullanmama izin verdiği için ve sıkıntılı sürecimde bana desteklerini hiçbirzaman esirgemeyen aileme ve arkadaşlarıma teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

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

Simgeler	**Açıklama**
τ	Çalışma kümesinin pencere boyutu
r_i	Erişim indisi
t	Zaman
ρ	Erişim yapılan hafıza sayfaları
P	Olasılık
R	Sıra
W	Çalışma kümesi
μ	Aritmetik Ortalama
σ	Standart Sapma

Kısaltmalar	**Açıklama**
ARP	Address Resolution Protocol (Adres Çözümleme Protokolü)
ATM	Asynchronous Transfer Mode (Asenkron Aktarım Modu)
BPF	Berkeley Packet Filter (Berkeley Paket Filtresi)
CDF	Cumulative Distribution Function (Birikimli Dağılım Fonksiyonu)
CCDF	Complementary Cumulative Distribution Function (Bütünler Birikimli Dağılım Fonksiyonu)
CAIDA	Cooperative Association for Internet Data Analysis (İnternet Verisi Analizi İşbirliği Kurumu)
CRC	Cylic Redundancy Check (Dönüşsel Artıklık Denetimi)
DARPA	Defence Advanced Resarch Project Agency (İleri Savunma Projeleri Ajansı)

Kısaltmalar	Açıklama
DNS	Domain Name Server (Etki alanı Ad Sunucusu)
HTTP	Hypertext Transfer Protocol
HTTPS	Secure Hypertext Transfer Protocol (Güvenli HTTP)
IP	Internet Protocol (İnternet Protokolü)
IRM	Independence Reference Model (Bağımsız Referans Modeli)
ISO	International Organization for Standardization (Uluslararası Standartlar Teşkilatı)
LRU	Least Recently Used (En Yakın Kullanımlı)
LRUSM	Least Recently Used Stack Model (En Yakın Kullanımlı Yığıt Modeli)
LVS	Linux Virtual Server (Linux Sanal Sunucu)
MAC	Media Access Control (Ortam Erişim Kontrolü)
MIT	Massachusetts Institute of Technology (Massachusetts Teknoloji Enstitüsü)
NCSA	National Center for Supercomputing Applications (Ulusal Süper bilgisayar Uygulamaları Merkezi)
NPU	Network Processor Unit (Ağ İşlem Birimi)
OSI	Open Systems Interconnection (Açık Sistem Bağlantısı)
PYDS	Paralel Yük Dengeleme Sistemi
RDBMS	Relational Database Management System (İlişkisel Veritabanı Yönetim Sistemi)
RFC	Request For Comments
RTT	Round Trip Time (Geri Dönüş Zamanı)
SMTP	Simple Mail Transfer Protocol (Basit Posta Aktarım Protokolü)

Kısaltmalar	Açıklama
SQL	Structured Query Language (Yapısal Sorgulama Dili)
TCP	Transmission Control Protocol (İletişim Kontrol Protokolü)
TTL	Time To Live (Yaşam Süresi)
UDP	User Datagram Protocol (Kullanıcı Veri iletim Protokolü)
VTYS	Veri Tabanı Yönetim sistemi

1. GİRİŞ

İnternet 20. yüzyılın en önemli buluşlarından biridir. 1960'lı yılların sonunda bir askeri iletişim projesi olan DARPA (Defence Advanced Resarch Project Agency – İleri Savunma Projeleri Ajansı) tarafından önce araştırma enstitülerini birbirine bağlamak için bir ağ yapısı tasarlanmıştır [1]. Bu ağ ile birden fazla bilgisayarın birbirine bağlanarak işlem güçlerini kullanabilmelerini ve veri alışverişinde bulunabilmeleri amaçlanmıştır. Ağ yapısı MIT'den Leonard Kleinrock'un paket anahtarlama teorisinden [2] yararlanarak geliştirilmiştir.

1970'li yıllara gelindiğinde bu konuda çalışmalar devam etmiştir. Farklı bilgisayar işletim sistemleri arasında iletişimi kolaylaştırmak için TCP/IP protokolü geliştirilmiştir [3].

Yıllar geçtikce bu ağa bağlı bilgisayarların sayısı gittikçe artmış ve bu ağ, "Ağların ağı" olarak da isimlendirilerek bugünkü İnternet'i meydana getirmiştir.

Son istatistikler incelendiğinde dünya nüfusunun %15'nin bu ağa dahil olduğu görülmektedir [4]. Diğer yandan gelişmiş ülkelerde bu oran %70'ler seviyesine yaklaşmaktadır.

Öte yandan yeni nesil bilgisayarlar dışındaki diğer cihazların da bu ağdan yararlanmaya başlaması ile İnternet üzerindeki trafik yoğunluğu buna paralel artacaktır. Dolayısıyla bu ağ üzerindeki servislerin daha iyi çalışabilmesi için istemcilerin yükünün en iyi şekilde dağıtılması gerekmektedir.

Bu nedenle bilim adamları istemcilerin yarattığı bu yükü nasıl en iyi şekilde dağıtırız sorusuna cevap aramaktadırlar. Bu tezin amacı istemcilerin oluşturduğu bu yükü dağıtmak için yeni bir metot ortaya koymaktır.

Dengeli Yük Dağıtma Problemi

İnternet'in büyümeye devam etmesi ile birlikte, bu ağ üzerinde akan trafik miktarının da artması kaçınılmaz olmuştur. Servis veren sunucular, önceleri kendileri üzerine gelen yükü karşılayabilirken, sonraları hem istemcilerin internet erişim hızlarının, hem de donanım kapasitelerinin artması ile birlikte trafik yükünü karşılama konusunda sorun yaşamaya başlamışlardır.

Tek sunucudan oluşan sistemlerin kapasitesi, donanım ve yazılımlar nedeni ile sınırlanmaktadır. Bu şekildeki sistemlerin kapasitesini dikey yönde artırmak, hem maliyetli hem de sınırlı oranda genişleme sağlamaktadır.

Bu yüzden dikey yönde genişleme yerine, sunucu sistemlerin sayısı artırılarak yatay yönde genişleme ile varolan sorunun üstesinden gelmeye çalışılmıştır [5].

Bu tezin konusu, yatay yönde genişleyen sunuculara internet trafiğini eşit şekilde dağıtabilecek, paralel yük dengeleme sistemi ortaya koyabilmektir.

Tezin Çerçevesi

Bu tezde, önce internetin altyapısını oluşturan protokollerden ve ip altyapısından bahsedilecektir. Daha sonra yük dengeleme sisteminin çalışma prensibinin dayandığı hafıza referans modellerine giriş yapılıp, ağ üzerinde referans yerelliği konusu anlatılmıştır.

Gazi Üniversitesi servis sunucularına gelen, trafik akış örneklerinden analizler yapılıp matematiksel mod işlemi ile yük dengeleme benzetim sonuçları verilmiştir. Yük dağıtım sistemlerinde, şu anda kullanılan matematiksel mod temelli yöntemlerin dengeleme için yeterli olmadığı ve yük tanımlamasında elde edilen bulgular ışığında geliştirilen sistem tanıtılacaktır. Daha sonra bu sistem ile alınan benzetim sonuçları verilmiştir.

Bu tezde, bölümler içinde ele alınan konular aşağıdaki özet bilgileri içermektedir.

- Bölüm 2'de öncelikle İnternet'in temelleri olan OSI katman modeli, ip protokolü ile bunun bileşenleri olan tcp ve udp protokolleri ele alınmıştır. Geçici referans yerelliği modeli anlatımından sonra, ağ ve web sistemlerinde geçici referans yerelliği üzerindeki çalışmalardan bahsedilip, yük dengeleme sistemleri üzerindeki çalışmalara yer verilmiştir. Bölümün sonunda ise paralel yük dengeleme sistemi çeşitlerine yer verilmiştir.

- Bölüm 3'de Gazi Üniversitesi aktif ağ cihazları üzerinden trafik toplama işleminin nasıl yapıldığı anlatılıp, daha sonra bu trafiklerin referans yerelliğini taşıyıp taşımadığı incelenmiştir. Bu trafikler kullanılarak ağ dengeleme benzetimi yapılmıştır.

- Bölüm 4'de paralel yük dengeleme benzetimi için tasarlanan Paralel Yük Dengeleme Sistemi (PYDS) anlatılmıştır. Bu sistemi kullanarak HTTP ve SMTP trafik örnekleri üzerinde yapılan paralel çalışma benzetimin sonuçları verilerek tartışılmıştır.

- Bölüm 5'de elde edilen sonuçlar tartışılmış, paralel yük dengeleme için bundan sonra bu alanda yapılacak çalışmalara ışık tutabileceği düşünülen iyileştirmelerin neler olabileceğine yer verilmiştir.

2. TEMEL BİLGİLER VE İLGİLİ ÇALIŞMALAR

Bu bölümde tez konusu ile ilgili temel bilgiler verildikten sonra paralel yük dengeleme ile ilgili çalışmalara yer verilmiştir.

2.1. OSI Referans Modeli

Bilgisayarlar arası iletişimin başladığı günden itibaren farklı bilgisayar sistemlerinin birbirleri arasındaki iletişim her zaman en büyük problemlerden birisi olmuş ve bu sorunun üstesinden gelebilmek için uzun yıllar boyunca çeşitli çalışmalar yapılmıştır. 1980'li yılların başında Uluslararası Standartlar Organizasyonu (International Standarts Organization-ISO), bilgisayar sistemlerinin birbirleri ile olan iletişiminde ortak bir yapı oluşturmak için bir çalışma başlatmıştır. Bu çalışmalar sonucunda, 1984 yılında Açık Sistem Bağlantıları (Open Systems Interconnection-OSI) referans modeli ortaya konulmuştur [6]. Bu model sayesinde, değişik bilgisayar firmalarının ürettikleri bilgisayarlar arasındaki iletişimi, bir standart oluşturmak ve farklı standartlar arası uyumsuzluk nedeni ile ortaya çıkan iletişim sorununu ortadan kaldırmak hedeflenmiştir. OSI referans modelinde, iki bilgisayar sistemi arasında yapılacak olan iletişim problemini çözmek için 7 katmanlı bir ağ sistemi önerilmiştir. Bir başka deyişle, bu temel problem 7 adet küçük probleme parçalanmış ve her bir problem için, ayrı ayrı bir çözüm yaratılmaya çalışılmıştır. Bu 7 katmanın en altında yer alan iki katman yazılım ve donanım, üstteki beş katman ise genelde yazılım yolu ile çözülmüştür.

OSI modeli, bir bilgisayarda çalışan uygulama programının, iletişim ortamı üzerinden, başka bir bilgisayarda çalışan diğer bir uygulama programı ile olan iletişiminin tüm adımlarını tanımlar. En üst katmanda görüntü ya da yazı şeklinde yola çıkan bilgi, alt katmanlara indikçe makine diline dönüşür ve sonuç olarak 1 ve 0 lardan ibaret elektrik sinyalleri halini alır [7]. Şekil 2.1'de OSI referans modeli katmanları gösterilmektedir.

Şekil 2. 1. OSI referans modeli katmanları.

OSI katmanlarının tanımlamaları [8]:

1. Fiziksel Katman: Bu katman, modelin elektiriksel ve donanımsal karakteristiklerini belirler. Bağlantının şekli, çalışma voltajı, sinyal modülasyonu ve kullanılacak kablonun özellikleri bunlar arasındadır. Bu katmanda çalışma için ağ arabirim kartları, tekrarlayıcılar vb. cihazlar örnek olarak verilebilir.

2. Veri İletim Katmanı: Prosedürel ve fonksiyonel olarak fiziksel katmana ulaşım stratejisini belirler. Aynı zamanda fiziksel katmandan gelen verinin hata düzeltmesi bu katmanda yapılır. Bu katmanda çalışan cihazlara köprüler ve ağ anahtarları örnek olarak verilebilir.

3. Ağ Katmanı: Değişken boyutlu olabilen, veriyi bir kaynaktan hedef adrese ulaştırmak için gerekli olan bir veya birden fazla ağ yolunu belirleyen katmandır. Yönlendiriciler bu katmanda çalışmaktadır. Ayrıca IP protokolü'de bu katmanda çalışmaktadır.

4. Taşıma Katmanı: Üst katmanlardan gelen verinin hedefe ulaşabilmesi için gerekli kontrollerin ve düzenlemelerin yapıldığı katmandır. Bu katmana örnek olarak TCP protokolü verilebilir.

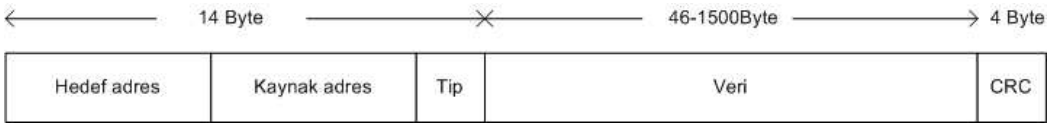
5. Oturum Katmanı: İletişimdeki uçlar arasındaki diyaloğun yönetildiği katmandır.

6. Sunum Katmanı: Ham verinin, sistemin anlayabileceği veri haline geldiği katmandır. Veri sıkıştırması, şifreleme ve bunun benzeri işlemler bu katmanda yerine getirilir.

7. Uygulama Katmanı: Katmanlar içerisinde kullanıcıya en yakın olanıdır. Bu katmanda, kullanıcıdan veri alınarak alt katmanlara iletilir. Metin işlemciler, banka terminal uygulamaları ve bunun gibi yazılımlar bu katmanda yer almaktadır.

2.2. MAC (Medium Access Control – Arabirim erişim kontrolü)

Arabirim erişim kontrolü (MAC) ethernet yerel ağ sistemlerinde veri iletim katmanında çalışan mekanizmadır. MAC ham veriye 14 byte'lık bir başlık ile veri sonuna da 4 byte'lık CRC verisi ekler [9]. Şekil 2.2'de verinin bu işlemden geçtikten sonraki yapısı gösterilmektedir.



Şekil 2. 2. Paket verisinin sarmalanmış yapısı.

Paket yapısı Şekil 2.2'de gösterildiği gibi temel olarak 14 Bytle'lık başlık, 1454 Byte'lık veri ve 4 Byte'lık CRC bölümlerinden oluşmaktadır.

Başlık

Başlık kısmı üç bölümden oluşmaktadır:

- Altı Byte'lık hedef MAC adresi. Buradaki adres tek, gurup ya da yayım adresi şeklinde olabilen, paketin gideceği cihaz adresidir.

- Altı Byte'lık kaynak MAC adresi. Veriyi gönderen cihazın adresi bu bölümdedir.

- İki Byte'lık verinin Tipi. Bu bölüm fiziksel katmanda farklı şekillerde iletişime olanak sağlamak içindir. IP ağları için değeri 0x0800 olmalıdır.

CRC

Verinin sonunda olan 32 bitlik (4 Byte) bu kısım, hata kontrolü için kullanılmaktadır. Veri iletim katmanına gelen veri kontrol edildikten sonra eğer CRC kontrolü olumsuz olursa göz ardı edilir. CRC kontrolü olumlu ise veri üst katmanlara aktarılır.

MAC Adresi

Başlık bilgisi içinde yer alan hedef ve kaynak adresleri aynı zamanda MAC adresi olarak da adlandırılır [10]. 6 Byte'lık yada 48 bitlik bu adres aynı zamanda MAC-48 olarak da bilinir. 48 Bitlik olduğu için 2^{48}'den 281,474,976,710,656 adet tekil MAC adresi olabilmektedir. MAC adresleri 8 bitlik guruplar halindedir, ancak okunması kolaylaştırılsın diye 16'lık sayı düzeninde gösterilmektedirler.

MAC adresinin 16'lık sayı düzende gösterimi aşağıdaki gibidir:
00-08-74-4C-7F-1D

Bu yapıdaki ilk üç Byte'lık kısım üreticilerin kodu olmaktadır. Geriye kalan üç Byte'lık kısım ise üretici tarafından tekil olarak üretilen cihazların adresi olmaktadır. Diğer yandan MAC adreslerinin bazı özel durumları da bulunmaktadır:

- Yerel ağdaki bütün bilgisayarların bahsi geçen paketi alması istenirse o zaman MAC adresi içindeki bütün bit'ler 1 yapılır. Buna yayım adresi (Broadcast adress)'de denir ve "FF:FF:FF:FF:FF:FF" şeklinde gösterilir.

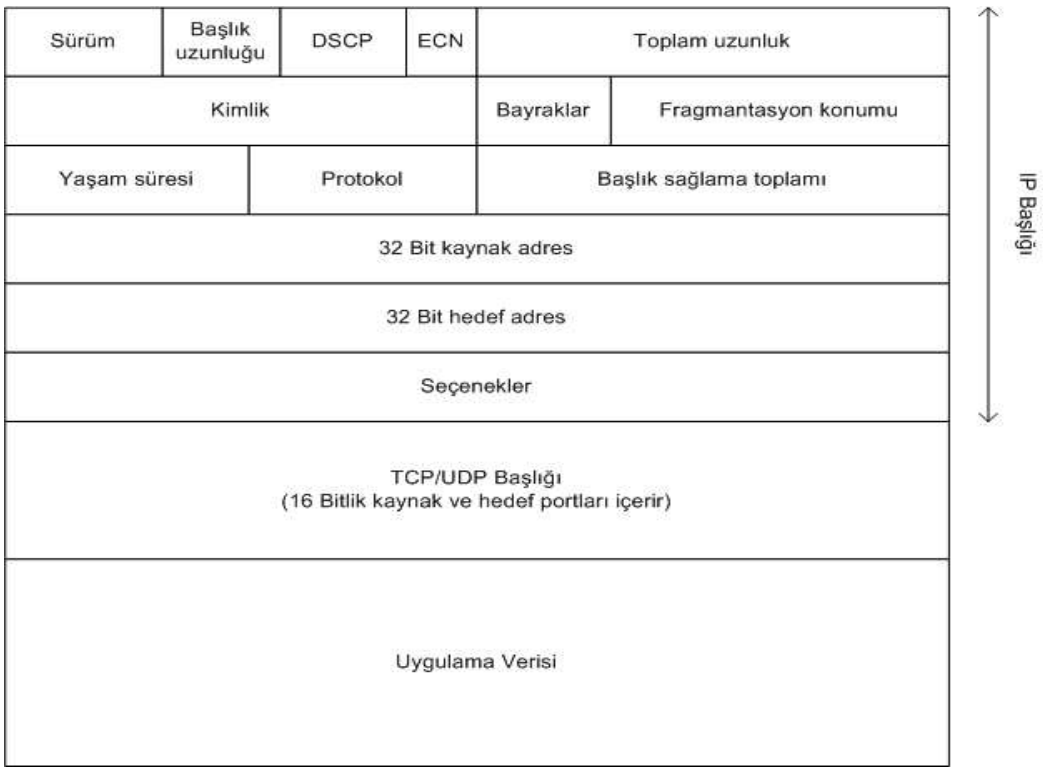
- Yerel ağdaki bir gurup bilgisayarın bahsi geçen paketi alması istenirse o zaman MAC adresinin ilk 8 biti içindeki en önemsiz bit (LSB) 1 olarak ayarlanır.

- Ağ yöneticisi tarafından ayarlanabilen ve özel olarak kullanılabilen MAC adresi için, ilk byte içindeki ikinci bitin 1 yapılması ile oluşturulur.

2.3. İnternet Protokolü (IP) ve Bileşenleri

İnternet homojen olmayan farklı yapılardaki cihazlardan oluşan bir ağdır. Dolayısı ile bu ağ üzerindeki farklı yapıdaki cihazların birbirleri ile konuşabilmeleri için ortak bir iletişim dili kullanmaları gerekmektedir.

İşte bu yüzden ortak iletişim dili olarak İnternet Protokolü (Kısaca IP) geliştirilmiştir [11]. Şekil 2.3'de IP protokolünün ve bileşenlerinin yapısı gösterilmiştir.



Şekil 2. 3. İnternet protokolü ve bileşenlerinin yapısı.

IP OSI modeline göre ağ katmanında çalışmaktadır. Bu protokolün esas görevi, IP veri dizilerini belirtilmiş hedef adrese ulaştırmaktır. Fakat gönderilen veri dizisinin karşı tarafa ulaşmasını ya da gönderilen sırada gitmesini garanti etmemektedir. Bu şekilde istenen güvenlik, sıralama ve çoklama önlemleri gibi işlemler daha üst katmanlarda yerine getirilmektedir [12].

IP protokolünün en önemli görevi yönlendirme fonksiyonudur. Şekil 2.3'de de görülebileceği üzere her IP başlığı kaynak ve hedef IP adreslerini içermektedir. Bu adresler kullanılarak, yönlendirme işlemi yapılmaktadır. Bunun yanında IP başlık bilgisinde aşağıda açıklanan alanlar bulunmaktadır.

- Sürüm: 4 bitlik sürüm bilgisi. Bu kısım 1980'lerden beri 4 değerine eşittir.

- Başlık uzunluğu: IP başlığının uzunluğunu gösteren 4 bitlik alandır.

- DSCP ve ECN: 6 bitlik Farklılaşmış Servis Kod Noktası (DSCP) alanı, (RFC 2474) ve 2 bitlik Kesinleşmiş Tıkanıklık Bildirimi(ECN) alanı (RFC 3168) eskiden kullanılan Servis Türü (TOS) alanının yerini almıştır.

- Toplam Uzunluk: 16 bitlik toplam uzunluk alanı, IP veri dizisinin IP başlık bilgisi ve diğer alanlarının toplam alanını içermektedir.

- Kimlik Bilgisi: 16 bitlik kimlik bilgisi, her bir IP paket dizisi için farklı değerde olup bölme ve birleştirme için kullanılmaktadır.

- Bayraklar ve Fragmentation: Bayraklar ve dağılma (fragmentation) alanı, bölme ve birleştirme için seçenekleri içermektedir.

- Yaşam süresi: 8 bitlik yaşam süresi alanı, gönderici tarafından ayarlanan ve geçilen her bir yönlendirici tarafından 1 eksiltilen ve sıfır değerine ulaşınca o paket dizisini göz ardı etmek için kullanılır.

- Protokol alanı: 8 bitlik protokol alanı, IP veri dizisinin içerdiği protokolü göstermektedir.

- Başlık sağlama: 16 bitlik başlık sağlama alanı, IP başlık bilgisi ve seçeneklerin sağlama bilgisini tutmaktadır.

- 32 bitlik kaynak ve hedef alanları, IPv4 adres bilgisini içermektedir. Ipv4 ile ilgili daha fazla bilgi bölüm 2.3.3'de verilmiştir.

- Seçenekler alanında, IP ile ilgili diğer seçenekler yer almaktadır.

2.3.1. Kullanıcı iletim protokolü (UDP)

UDP basit taşıma katmanı protokollerinden biridir. RFC 768'de ayrıntılı olarak yapısı verilmiştir [13]. Uygulamalar, verileri UDP soketine yazdıkları anda, veriler UDP veri dizileri haline dönüştürülüp daha sonra IP dizisi ile birleştirilerek hedef adrese gönderilirler. UDP veri dizisinin hedef adrese ulaşacağını garanti etmez [12].

Bu yüzden UDP protokolü ile veri iletimi güvenli değildir. Diğer yandan bu işlemler kullanıcı tarafından uygulama programında yerine getirilmektedir.

UDP'nin bir başka özelliği ise bağlantıdan bağımsız olarak çalışmasıdır. Bu yüzden sunucu ve istemci arasında uzun süreli bağlantılar yapılamaz.

2.3.2. İletim kontrol protokolü (TCP)

TCP ilk olarak Postel tarafından RFC 793 [3] ile tanımlanmıştır. Daha sonra tarih sırası ile RFC 1323 [14], RFC 2581 [15], RFC 2988 [16] ve RFC 3390 [17] ile yenilenmiştir.

TCP protokolü UDP protokolünden farklı özellikler göstermektedir. UDP protokolüne göre en önemli farklılığı güvenilirliktir. TCP gönderilen verinin karşı

tarafa ulaştığından emin olmak için kontroller yapar. Eğer verinin karşı tarafa ulaştığına ait bir geri bildirim dönmez ise veri yeniden gönderilmeye çalışılır. Belli bir zaman aşımına ulaşana kadar bu işlem devam eder. Eğer bu zaman aşımın sonucu verinin karşı tarafa ulaştırılamadığına kanaat getirilirse o zaman üst katmanlara bildirilir.

TCP, istemci ve sunucu arasındaki gidiş-dönüş zamanını (Round Trip Time - RTT) dinamik olarak tahmin eden algoritmaları içermektedir. Böylece cevap için gerekli olan zamanın ne olacağını bilmektedir. Örneğin yerel ağlar için bu süre milisaniyeler seviyesinde olabilirken uzak ağlar için bu süre saniyeler seviyesine çıkmaktadır.

TCP ayrıca verileri sıralama özelliğine de sahiptir. Bunun için gönderdiği her bir pakete sıra numarası vermektedir. Örneğin hedefe bir seferde gönderilebilecek veriden daha büyük miktarda veri gönderilmek istediğinde bu veri parçalara bölünerek numaralandırılır. Bu parçalar daha sonra hedef adrese gönderim sırasındaki gibi ulaşmamış ise, karşı taraftaki TCP protokolü bunu yeniden sıraladıktan sonra üst katmanlara iletir. Ya da karşı tarafa aynı sıra numaralı paketten iki tane ulaşırsa o zaman yine TCP protokolü birini göz ardı eder.

TCP akış kontrolü de sunmaktadır. TCP, iletişimde olduğu noktalara, bir anda ne kadar veriyi kabul edebileceğini bildirmektedir. Buna ilan edilmiş pencere (advertised window) adı verilir. Bu pencere, herhangi bir zamanda veri alış arabelleği için ayrılmış alanı ifade eder ve göndericinin bu arabelleği doldurmamasını garanti eder. Bu alan dinamik olarak değişmektedir. Örneğin, göndericiden veri geldiğinde bu alan küçülmekte, ancak veriyi alması gereken uygulama bu veriyi okuduğunda, bu alan yeniden artmaktadır. Eğer gönderici bu alanı doldurursa, veriyi alan uygulama bu alandaki veriyi okuyuncaya kadar, karşı taraftan yeni veri kabul edilmez.

Son olarak TCP çift yönlü iletişime izin vermektedir. Yani herhangi bir uygulama aynı iletişim kanalı üzerinden hem veri gönderme hem de alma işlemini aynı zamanda yapabilmektedir.

Yukarıda sayılan özelliklerinden dolayı TCP protokolü, hem uygulama tarafında karmaşıklığı giderdiği, hem de verileri iki uç arasında güvenilir biçimde birbirine ulaştırdığı için internet üzerinde en fazla kullanılan protokoldür. McCreary ve Claffy'in bu konudaki çalışması da TCP/IP'nin İnternet'de ne kadar fazla kullanıldığını göstermektedir [18].

2.3.3. Ipv4 adres yapısı

Ipv4 adresler IP protokolünde belirtildiği gibi 32 bit uzunluğunda ve dört farklı ondalık sayı gurubunun birbirinden noktalar ile ayrılmış bir yapıdadır. Bu yapıya noktalı-ondalık (dotted-decimal) gösterim adı verilmektedir. Bu yapı, adres türlerine göre Çizelge 2.1'de gösterilmektedir.

Çizelge 2. 1. Ipv4 adreslerinin beş farklı sınıfının adres aralıkları.

Kullanım	Sınıf	Aralık
Unicast	A,B,C	0.0.0.0 – 223.255.255.255
Multicast	D	224.0.0.0 – 239.255.255.255
Deneysel	E	240.0.0.0 – 255.255.255.255

Ipv4 ağ ya da alt ağ adreslerinden bahsedildiğinde, aslında 32 bitlik ağ adresi ve 32 bitlik ağ maskesi konu edilmektedir. Ağ maskesindeki 1 değerli bitler ağ adresini ve 0 değerli bitler'de host alanını kapsamaktadır. Ağ maskesindeki 1 değerli bitler genelde sol taraftan başlayarak sağa doğru devam ederken 0 değerli bitler ise en sağdan başlayarak sola doğru devam eder. Ağ maskesi içerisindeki soldan sağa doğru olan bitlerin sayısı önek uzunluğu olarak anılır. Örneğin 255.255.255.0 maskesinin önek uzunluğu 24 olmaktadır. Buna sınıfsız adresler (classless addresses) adı da verilmektedir.

Alt ağ adresleri

Ipv4 adresleri çoğunlukla alt ağlara bölünmektedirler. Bu adres hiyerarşisine farklı bir seviye katmaktadır:

- Ağ kimliği (Alana atanan)
- Alt ağ kimliği (Alan tarafından seçilen)
- Host kimliği (Alan tarafından seçilen)

En üst noktada ağ kimliği yer almaktadır. Ağ kimliği servis sağlayıcılara atanmış adreslerdir. Bu adresler IP dağıtan kurumlar tarafından servis sağlayıcılara atanırlar. Alt ağ ve host kimlik adresleri servis sağlayıcılar tarafından seçilebilir.

Örnek olarak 192.168.1.0/24 olarak atanan bir ağ adresi servis sağlayıcı tarafından alt ağlara bölünebilir. Örneğin 3 bitlik alt ağ kimliği kullanılırsa, 5 bitlik alan host kimliği olarak kullanılabilecektir. 3 bitlik ağ kimliğinden dolayı 256 adet IP olan bu ağ alanı 8'e bölünecektir. 5 bitlik kalan alan ise host kimliği olarak kullanılacaktır. Bölünen ağ numarası Çizelge 2.2'deki gibi olacaktır.

Çizelge 2. 2. Üç bitlik alt ağ kimliği ve Beş bitlik host kimliği ile bölünmüş ağ kimliği.

Alt ağ	Önek
0	192.168.1.0/27
1	192.168.1.32/27
2	192.168.1.64/27
3	192.168.1.96/27
4	192.168.1.128/27
5	192.168.1.160/27
6	192.168.1.192/27
7	192.168.1.224/27

2.4. Program Hafıza Referans Davranış Modelleri

Program hafıza referans davranış modelleri anlatımından önce, hafıza erişiminde kullanılan önemli bir kavrama yer verilmesi gereklidir. Bu kavram referans yerelliği (reference of locality) ya da verinin yerelliği olarak adlandırılmaktadır [19].

2.4.1. Referans yerelliği

Bu kavram program hafıza referans davranış çalışmalarında ortaya atılmıştır [20]. Bilgisayar sistemlerindeki hafıza alanlarındaki iş yükünün yarattığı referans yerelliği modeli tanımlamaları, başka sistemler için de kullanılabilir. Örnek olarak dosya sistemleri, ağlar ya da sunucu sistemleri üzerindeki yükler bu modelle tanımlanabilmektedir.
Referans yerelliğinin geçici ve uzamsal olmak üzere iki farklı türü bulunmaktadır.

2.4.1.1. Geçici referans yerelliği (Temporal Locality)

Eğer bir hafıza alanına daha önceden erişim yapılmışsa ve çok yakın zamanda aynı hafıza alanına yeniden erişim yapılması olası ise, bu duruma geçici "referans yerelliği yapılıyor" denir. Bu durum iki nedenden dolayı ortaya çıkmaktadır. Birincisi; bazı hafıza alanlarına diğer hafıza alanlarına göre daha fazla erişim olması, İkincisi; aynı hafıza alanlarına erişimin benzer zaman kümesinde olmasıdır [21].

2.4.1.2. Uzamsal referans yerelliği (Spatial Locality)

Eğer bir hafıza alanına erişilmiş ve yakın zamanda o hafıza alanının etrafındaki alanlara erişilmesi mümkün gözüküyorsa buna uzamsal referans yerelliği adı verilir. Bunu doğuran nedenler, aynı kayıta ait alanlar ya da matrisler içindeki ilişkili değerlerin birbirlerine yakın şekilde olmalarıdır. Bu yüzden önbelleğe alınan bir değerden sonra yakındaki bellek adresindeki veriye de ihtiyaç duyulabilmektedir [21].

2.4.2. Sanal hafıza sistemi

Bilgisayar bilimlerinde uygulamaların Sanal Hafızayı (VM) oluşturan hafıza sayfalarına nasıl erişeceğine ilişkin birçok çalışma yapılmıştır [22]. VM hafıza sistemi, uygulamalara fiziksel hafıza alanından daha fazla kullanılabilecek alan sunmaktadır. Sayfalama (Paging) VM sistemlerde kullanılan bir yöntemdir. Bu yöntemde hafıza alanları ikinin kuvveti olacak şekilde sabit boyutlu alanlara bölünür. Bunlara hafıza sayfaları adı verilir. VM sisteminde fiziksel hafıza alanının dışında daha farklı depolama alanları (disk, daha yavaş hafıza modülleri vb.) fiziksel hafıza alanının devamıymış gibi kullanılır. Bu işlem işletim sistemi tarafından yerine getirilir [23].

Uygulamaların bütününün fiziksel hafıza alanında bulunmasına gerek yoktur. Onun yerine uygulamanın ihtiyacı olduğu zaman bu hafıza alanı kullanılır. Ancak fiziksel hafıza alanı dışındaki kısımlardan verinin getirilmesinin sonucunda bu alanların fiziksel hafızaya göre daha yavaş olması nedeni ile pahalı bir operasyondur ve sistem performansı bu işlemden kritik olarak etkilenmektedir. Bu yüzden sürekli erişilen sayfaların fiziksel hafızada durması hız açısından daha önemlidir.

Diğer yandan fiziksel hafıza alanı, daha ucuz depolama alanlarına göre kısıtlı olduğundan uygulamaların kullanmadığı hafıza sayfaları, VM içindeki bu kısımlara aktarılır. Bu işlem hafıza alanlarında değiştirme kuralı ile yerine getirilir [24]. Bu kurallar arasında en fazla bilineni LRU (Least Recently Used) kuralıdır. Bu kural geçici referans yerelliği modeline avantaj sağlayarak, yakın zamanda erişilen hafıza alanlarına yeniden erişilebileceğini kabul ederek, o alanların hafızada tutulmasını sağlar. Bu çalışmada da bu kurala uyulmaya çalışılmıştır.

2.4.3. Çalışma kümesi modeli

Çalışma kümesi modeli sayfa referans davranışını modellemek için geliştirilmiştir [25]. Program çalışma kümesi $W(t,\tau)$, $(t-\tau, t)$ süresince uygulama tarafından referans edilen hafıza sayfa kümesini gösterir. τ parametresi çalışma kümesinin pencere

boyutudur. Çalışma kümesinin hafıza yönetimindeki ilkesi şöyledir; uygulama merkezi işlemciyi ancak kendi çalışma kümesi hafızada ise kullanabilir, diğer yandan aktif olarak çalışan bir uygulamanın eğer çalışma kümesi sayfası hafızada yok ise, o zaman hafızadan silinmesi gerektiği kabul edilir. Çünkü çalışma kümesi modeli hafıza alanlarını referans edecek uygulamaların eski referans yerlerine uyumluluk göstereceğini öngörmektedir.

Uygulamanın erişebileceği hafıza sayfaları kümesi $N = \{ 1,2,...,n \}$ şeklinde verilirse, çalışma kümesine konu olan uygulamanın çalışma esnasında eriştiği hafıza sayfaları da $\rho = r_1, r_2, ... , r_t, ...$ şeklinde olacaktır burada $r_t \in N$ ve t ayrık zamanda referans edilen sayfayı göstermektedir.

Denning ve Schwartz'ın çalışmaları sonucu çalışma-kümesi-ortalama-büyüklük fonksiyonu, atlanmış-sayfa-oranı fonksiyonu ve girişim-aralık fonksiyonları arasında birden fazla ilişki türetilmiştir [26]. Çalışma kümesinin boyutu $w(t,\tau)$, $W(t,\tau)$ içindeki sayfaların sayısına eşittir.
Örnek olarak;

$$w(t, \tau) = |W(t, \tau)| \tag{2.1}$$

Çalışma kümesini ilk k referansa göre ortalamasını aşağıdaki gibi alınabilir:

$$s_k(\tau) = \frac{1}{k} \sum_{t=1}^{k} w(t, \tau) \tag{2.2}$$

Çalışma-kümesi-ortalama-büyüklüğü ise:

$$s(\tau) = \lim_{k \to \infty} s_k(\tau) \tag{2.3}$$

şeklinde tanımlanır [26].

Atlanmış-sayfa-oranı m(τ) belirli bir zaman aralığında çalışma kümesine dönen sayfaların sayısı olarak tanımlanır. m(τ), τ olarak verilen pencere boyutunda yeni bir sayfanın referans edilme olasılığını gösterir. Genel girişim-aralık fonksiyonu f(x) ise;

$$f(x) = \sum_{i=1}^{n} \lambda_i f_i(x) \tag{2.4}$$

dir.

Yukarıda verilen fonksiyonda $f_i(x)$; sayfa i'nin girişim aralık fonksiyonunu ve λ_i; sayfa i'ye yapılan referansların frekansını gösterir. Analizler göstermiştir ki:

$$m(\tau) = s(\tau+1) - s(\tau) \tag{2.5}$$

ve

$$f(\tau) = m(\tau-1) - m(\tau) \tag{2.6}$$

dir.

Yukarıda da görülebileceği gibi $f(\tau)$, $m(\tau)$'nin negatif bir eğimi ve $m(\tau)$'de $s(\tau)$'nin bir eğimidir.

Çalışma kümesi için en önemli varsayım sayfa referans dizisini oluşturan durumun durağan olmasıdır. Bu durum program yeni sayfa kümesine (referans yerelliğine) geçmediği takdirde doğrudur. Ancak uygulamalar çalışırken yukarıda anlatılan durum, beklenen bir durumdur. Fakat yinede bu durağan olma durumu, çalışma kümesi modelinin program hafıza referans davranışı için iyi bir başlangıç noktası olmasını engellemez. Bundan başka ağ trafiği ve web döküman referans modellerinin açıklanmasında çalışma kümesi modeli kullanılabilir.

2.4.4. Bağımsız referans modeli (Independence reference model)

Bu modelde sayfa referans dizisi basitçe uygulamanın eriştiği sayfa numaralarını R_1, R_2, ..., R_t şeklinde alarak bağımsız rastgele bir ortak referans dağıtım tablosu oluşur [27].

$$Pr[R_t = A_i] = p_i, \quad 1 \le i \le N \text{ , } t > 0 \qquad (2.7)$$

Burda A_i N tekil sayfa arasındaki i. sayfadır.

Bu model düzensiz sayfa referans davranışlarını yakalar. Ancak aynı sayfalara yapılan başarılı referansları tanımlayamaz. Bu yüzden bu model geçici referans yerelliği tanımını karşılayamaz.
Harvard'da dil bilimci olan George Kingsley Zipf tarafından ortaya konan Zipf kanunu ise [28] bazı olayların gerçekleşme sıklığının ardışık sıklıkta gerçekleşen olaya göre üstsel kuvvette bir ilişki içinde olduğunu göstermiştir.

Olayların gerçekleşme sıklığını (P) olarak ve olayın bütün içindeki sırasını ise (R) ile gösterilirse

$$P(R) \approx 1/R^{\alpha} \qquad (\alpha \sim 1) \qquad (2.9)$$

Uygulama hafıza referans erişimlerinde [29], Web sunucu yüklerinde de [30] benzer davranışlar keşfedilmiştir.

Yine Adamic ve Huberman'ın çalışmalarında, internet trafikleri içerisindeki Zipf davranışları gösterilmiştir [31].

2.4.5. En yakın kullanımlı yığın modeli (LRUSM)

LRUSM (Least Recently Used Stack Model) başka bir uygulama hafıza modelidir [32]. Tek boyutlu ve tüm adreslerin içinde barındırıldığı bir yığın olduğu kabul

edilirse, bir hafıza sayfasına erişim yapıldığında bu adresin yığın içerisindeki uzaklığına göre yığın katalogu içerisinde işaretlenir. Erişim değerine göre bu veri katalog içerisine yerleştirilirken bu değer'den daha büyük değerlikli sayfa adresleri bir basamak aşağı doğru kayar, diğer yandan bu sayfa ilk defa erişime uğramışsa o zaman yığının en üst noktasına yani 0 pozisyonuna yerleştirilir. Bu aslında LRU modelinin yığının güncelleştirilmesi için kullanılmasıdır. Yığın içerisindeki sayfa adreslerinin yeri onların yığın uzaklıklarını gösterir. Uzaklık dizini "38,1,0,145,123,1,0,40,97,312,1" şeklinde ise, burada "0" değeri adresin, yığının en üst kısmında yer aldığını gösterir. "1" değeri ise adresin anlık erişilmiş adresten hemen önce erişildiğini ifade eder ve bu işlem bu şekilde devam eder gider.

LRUSM, geçici referans yerelliğini yığın uzaklığına bağlı olarak adresleyebilmektedir ve bunu ise, erişimin sürekli olduğu adreslerin yığın uzaklığı az olması sayesinde sağlamaktadır. Ancak bu tam olarak geçici referans yeri şartlarını yerine getirememektedir. Çünkü bu modelde zaman unsuru mevcut değildir.

2.4.6. Erişim uzaklığı modeli

Bu model W. Shi, M.H. MacGregor ve P. Gburzynski'nin paralel yönlendirme sistemleri için yaptıkları araştırmalarda ortaya konulmuştur [33].

Bu modelde LURSM'deki yığın uzaklığına benzer "erişim uzaklığı" kavramı kullanılmıştır. "a_1, a_2, ..., a_n" şeklinde sıralanan adreslere "r_1, r_2, ..., r_n" şeklinde erişildiği ve LRU yığın yöntemi ile bu sıralamaların oluşturulduğu kabul edilmiştir. Ek olarak yığının tüm ip adreslerini içerebildiği ve dizin göstericisinin üsten aşağı doğru sıfırdan başlayarak artığı da kabul edilmiştir. Yığın içindeki bir adrese erişim olduğunda o adres yığınının en üstüne yerleştirilir.

Bu modeli daha iyi anlatabilmek için A_i'nin "a_1, a_2, ..., a_n" şeklinde tekil adreslere sahip bir küme olduğunu ve $|A_i|$'nin en büyük erişim uzaklığı değeri olduğundan yola çıkarak;

$$r_{i+1} = \begin{cases} idx(a_{i+1}) & Eger \quad a_{i+1} \in A_i \\ |A_i| & Eger \quad a_{i+1} \notin A_i \end{cases} \tag{2.10}$$

Buradaki idx(a_i) a_i'nin LRU yığını içerisindeki indisini göstermektedir.

2.5. Ağ Ortamında Referans Yerelliği

Birçok çalışmada ağ ortamlarında referans yerelliği bulunduğu saptanmıştır. Referans yerelliği yerel ağdan (LAN) geniş bölge ağlarına (WAN), İnternette kullanılan farklı protokollere kadar birçok noktada bulunmaktadır. Referans yerelliği özellikle gönderme (forwarding) sistemleri için çok önemlidir. Bu sistemlerin performanslı çalışabilmesi için önbellekleme çok önemlidir. Bu bölümde ağ ortamındaki referans yerelliğini tanımlamak, ölçmek ve kullanmak tartışılmıştır.

2.5.1. Paket ağlarında referans yerelliği

Paket ağlarındaki referans yerelliği ile ilgili ilk çalışmalardan biri de Jain ve Routhier'in MIT'de yaptığı çalışmadır [34]. Bu çalışmada yerel ağda yapılan ölçümler sonucu ağdaki trafiğin Poisson [35] ya da Birleşik Poisson dağılımı ile modellenemeyeceğini, bunun yerine "paket treni" modelini ortaya koymuşlardır [34,35].

Bu modelde ağ içindeki iki nokta arasındaki paket akışının olduğu kabul edilir. Bu paket akışlarının da veri taşıyan trenler ile yapıldığı varsayılır. Paketin gelişi ile sonraki paketin aynı trenin dönüşüne ait olma olasılığını vardır. Bu da ağ protokollerinin istek-cevap doğasından gelmektedir. Bu keşiften sonra ağ cihazlarında kaynak ve hedef adresi önbelleklemesi çok önem kazanmışdır.

Yine MIT'den Feldmeier'in yaptığı çalışmalarda [36] ağ yönlendiricisi üzerinde birkaç "hedef adres, çıkış portu" şeklinde satır içeren basit bir önbelleğin bile performansı ciddi biçimde artırdığı gözlemlendi. Burada hedef adresin yeniden

kullanım sıklığı analiz edildi ve LRU önbelleğin yönlendirme tablosuna erişimini azalttığı belirlenmiştir.

Mogul ağ yerelliğini işlem seviyesinde araştırmıştır. Referans yerelliğinin sistem adresi seviyesinde olduğunu ortaya koymuştur. Yerel ağdan topladığı veriler üzerinde yaptığı analizlerde %75 oranında paketin hedef port numaralarının aynı ve paketlerin geldiği sistemlerde de kaynak port numaralarının aynı olduğunu tespit etmiştir [37]. Dolayısıyla burada da referans yerelliğinin olduğu söylenebilir. Aynı şekilde UDP protokolü üzerinde yapılan çalışmalarda da benzer referans yerellikleri tespit edilmiş ve ona göre önbellekleme ile UDP protokolünün performansı artırılmıştır [38].

CAIDA'dan Broido, Hyun ve Claffy'nin [39] ip trafiği üzerinde yaptıkları çalışmalarda trafik akış yaratan kaynakların diğerlerine göre farklılığının Pareto kuralı [40] 80/20 ile benzerlik gösterdiğini (%80 trafik %20 kaynak) bulmuşlardır. Daha derin araştırmaları sonucu bu oranların bazı yerlerde 90/10 hatta 95/5 ulaştığını da tespit etmişlerdir [39].

IP paket akışları içerisindeki kaynak yada hedef adresine göre önbelleklemenin yönlendirmeyi hızlandırdığı bilinmektedir. Önbelleklerde tüm ip adresini tutmak, ip adresinin ağ kısmını yada ağ kimliğini tutmak, aslında bir avantaj getirmemektedir. Ancak yinede adres bilgisi kontrol edilirken, ip adresinden ağ kısmının ayrılması gerektiğinden bu kontrol için fazladan bir işlem yapmak anlamına gelir. Diğer yandan referans yerelliğini sağlamak için bu tez içindeki verilen çalışmada önbelleklenen ip adreslerinin bütünü tutulacaktır.

2.5.2. Web sunucu yüklerinde referans yerelliği

Web sunucu erişim kayıtları üzerinde yapılan çalışmada popülerlik tabanlı bağımsız referans modelinin sunucular üzerindeki yükü tarif etmekte yetersiz kaldığı bulunmuştur [41]. Bunun yerine LRUSM modelinin uygulanması ile erişim kayıtlarının yığın uzaklığının sıra dışı dağılımı adreslediği bulunmuştur.

Arlitt ve Williamson'un altı değişik ağ trafiği verisi üzerinde yaptıkları çalışmada sunucular üzerinden istenen sayfaların erişim sırasına değil doğrudan belirli sayfalar üzerine yoğunlaştığını tespit etmişlerdir [42]. Burada referans yerelliğini yakalamak için LRUSM kullanılmıştır. Aynı zamanda toplam trafik üzerinde yapılan incelemelerde erişim şeklinin kesinlikle Poisson dağılımı şeklinde olmadığı görülmüştür.

Web trafiği üzerinde yapılan araştırmalar bu trafikler üzerinde döküman popülerliği ve kullanım-uzaklık dağılımlarının önemli rol oynadığı görülmüştür. Bu davranışların sırasıyla bağımsız referans modeli ve LRUSM tarafından adreslenebildiği gözlemlenmiştir.

He ve Karamcheti'nin veri-merkezli web sunucuların erişim kayıtlarında veri alanı, ağ bölgesi ve farklı zaman aralıklarında yaptıkları araştırmalarda, yüksek oranda ağ referans yerelliğinin olduğu tespit edilmiştir [43]. Bir sunucuda %10 istemcilerin IP'sinin %99.95'lik istem yarattığı gösterilmiştir.

2.6. Akış Seviyesinde Trafik Tanımlama

Ağ üzerindeki akış ortak özelliklere sahip bir gurup paketin birlikte aynı yönde hareket etmesi şeklinde tanımlanabilir. Akışların nasıl tanımlanacağı ile ilgili farklı yaklaşımlar vardır. Örneğin paket treni modelinde ağ akışı içindeki paketler trenlerin belirli istasyondan diğer belirli istasyona gitmesi gibi modellenmiştir. İnternette bazı ip başlık bilgileri akışı tanımlamak için seçilmiştir. Bu bilgiler en yalın halde {protokol, kaynak adres, kaynak port, hedef adres, hedef port} şeklinde tanımlanabilmektedir [44]. Claffy [45] akış tanımını parametrik bir metotla tanımlamaya çalışmıştır. Örneğin akış yönleri, tek/çift bitiş noktası (ları) ve fonksiyonel katman olarak tanımlama yapılmıştır.

Çoğu akış tanımlamaları zaman-aşımı temellidir [34,45,46]. Bu şu anlama gelmektedir: eğer bir akış içerisinde birbiri ile ilişkili iki paket arasındaki süre

önceden belirlenmiş zamanaşımı süresini doldurmamışsa, bu akış aktif kabul edilir. Aksi takdirde aktif olmayan akış olarak tanımlanır.

Diğer yandan birçok çalışma ağ trafiğini kümelenmiş trafik üzerinden tanımlamaya çalışmıştır. Bu çalışmalarda bir sistemden diğer sisteme yaratılan akışlar tek bir akış gibi değerlendirilmiştir. Kümelenmiş ağ trafikleri üzerindeki bulgular bu trafiklerin bütün zaman aralıklarında benzer yapıda seyir izlediğini göstermiştir [47].

Sarvotham İnternet trafiğini bağlantı seviyesinde incelemiştir [48]. Elde ettiği bulgu ise trafik içerisindeki yükselmelerin, kümelenmiş trafik modellerinde kabul edildiği gibi [49] aynı anda hareket eden akışlar yüzünden değil, onun yerine yüksek bağlantı hızlarına sahip noktalardan yapılan büyük boyutlu dosya transferlerinin olmasıdır. Bu bağlantılar alpha ve geriye kalanlarda beta trafik olarak adlandırılmışlardır. Alpha ve beta trafiklerinin nasıl ayrılacağı Savotham, Riedi ve Baraniuk'un çalışmalarında tartışılmıştır [50].
Yine CAIDA'dan Brownlee ve Claffy'in çalışmaları İnternet trafiği içerisindeki akış örneklerindeki ölçümleri ağ akışlarının boyutlarına göre (filler ve fareler) ve hayat sürelerine göre (kaplumbağa ve kız böceği) olarak sınıflandırılabileceğini göstermiştir [51]. Hayat süresi 15 dakikanın üzerinde olan akışlar kaplumbağa olarak adlandırılırlar. Bu tip akışlar bütün akışlar içerisinde %1 veya %2'lik yer tutmalarına rağmen trafiğin %50 yada %60'ını taşıyabilirler.

2.7. Web Sunucu Sistemlerinde Yük Dengeleme Yöntemleri

İnternet ağı üzerinde en fazla kullanılan protokollerden biri HTTP'dir [18]. Bu yüzden yük dengeleme sistemi çalışmalarında çoğunlukla web sunucular üzerindeki yük dengeleme çözümlerine odaklanılmıştır. Bundan dolayı ağ trafiği üzerindeki yük dengeleme metotları web sunucular için geliştirilip diğer protokollere yada uygulamalara da kolayca adapte edilebilmektedirler.

Daha önce teknik olarak, hizmet sunucusunun yükü arttığı zaman ve bu sunucu bu yükü kaldıramaz duruma geldiği durumda çözüm olarak sunucuyu güçlendirmek

veya daha fazla sistem gücüne sahip yeni bir sistemle değiştirmek gerekiyordu. Bu işlem hem zaman alıcı, hem sorun oluşma olasılığı yüksek hem de maliyetli bir operasyondur. Bu nedenle var olan sistemlerin daha uzun kullanılabileceği ve daha düşük maliyetli olacak şekilde çözüm yolları aranmaya başlanmıştır.

Böylece bir gurup bilgisayarın tek bir sistem gibi davrandığı yapılar tasarlanmaya başlanmıştır. Bu şekilde yapılandırılmış sistemlere kümeleme (clustering) sistemleri adı verilir [52]. Kümeleme sistemleri düşük maliyetli sistemlerle, yüksek performanslı ve ölçeklenebilir tek sistem gibi davranan yapıların geliştirilmesine yol açmıştır. Bu sistemlerin ortak çalıştırılması ile ilgili yüzlerce çalışma yapılmıştır.

Kümeleme sistemlerinin, istemciler tarafında şeffaf bir yapıda çalışması gereklidir. Ancak bazı kümeleme yöntemlerinde istemcilerden bağımsız yapılar tasarlanmasına rağmen küme içindeki sunucular üzerinde bazı değişikliklerin yapılması gerekir. Bu da şeffaf kümeleme yapısına uygunluk göstermemesine yol açar. Kümeleme sistemleri yapılandırılırken, istemciler ve kümeyi oluşturan sistem bileşenleri üzerinde mümkün olduğunca değişiklik yapılmaması tercih edilmektedir.

Yük dengeleme sistemlerindeki yaklaşımlar üç kısımda ele alınabilir. Bunlar:

- İstemci tabanlı yaklaşımlar.
- DNS (Domain Name Service) tabanlı yaklaşımlar.
- Yük Dengeleyici tabanlı yaklaşımlar.

Bu yaklaşımlar ilerdeki bölümlerde tek tek ele alınmıştır.

2.7.1. İstemci tabanlı yaklaşımlar

Bu yaklaşımda istemci sunucu kümesine nasıl ulaşacağına kendisi karar vermektedir. Bu şekildeki çalışmaya örnek olarak Netscape firmasının web tarayıcısında

kullandığı yöntem verilebilir. Netscape firmasının web tarayıcısı ile firmanın ana sayfası olan http://www.netscape.com adresine ulaşılmaya çalışıldığında, web tarayıcı *i* (i web sunucu sayısı olmak üzere) değerini www*i* şeklinde rast gele değiştirerek DNS çözümlemesini yapar. Böylece web tarayıcısı rast gele küme içindeki servis veren herhangi bir sunucuya ulaşır. Ancak bu sistemde karşı sistemlerdeki sorunları tespit edecek bir yöntem bulunmadığı için ne tam olarak dengeleme nede ölçeklenebilirlik sağlanmaktadır [53].

İstemci tabanlı yaklaşımlarda bir başka yöntem ise akıllı istemciler kullanmaktır. Bu şekildeki istemcilerde sunucu fonksiyonelliği, istemciye bir java applet yardımı ile atanır [54]. Ancak bu yöntemde istemci ve sunucu arasındaki mesaj trafiği artar, bu ise yükü azaltacağına daha fazla yük yaratır. Bu sistem ölçeklenebilirlik ve bulunurluk sağlamasına rağmen taşınabilirlik sağlamamaktadır.

2.7.2. DNS tabanlı yaklaşımlar

Dağıtık sunucu mimarileri, istek yönlendirme mekanizmasını kullandığı için istemci-tabanlı yaklaşım sorunlarından etkilenmez. Mimarinin şeffaflığı tipik olarak sunucu kümesinin dışarı doğru tek bir sanal arabirim sağlaması ile olur. Bu en azından URL (Uniform Resource Locator) seviyesinde olmalıdır.

Sunucu sistemlerinin yetkili DNS'i (Küme DNS'i) dışarıdan gelen alan adı ile ilgili istekleri küme içerisindeki herhangi bir sunucunun IP'sini seçerek cevaplar. Bu şekildeki çalışma küme DNS'inin değişik kurallar dahilinde istemcilerden gelecek istekleri küme içerisindeki sunuculara dağıtmasına yardımcı olur.

Ancak küme DNS'i sunucu kümesine ağdan gelecek paket istekleri konusunda sınırlı kontrole sahiptir. İnternet DNS sistemi, DNS sistemleri üzerindeki yükleri azaltmak için dağıtık yapıda çalışmaktadır. Bu yüzden istemcilerin kendi DNS'lerinden gelen sorgu talepleri, küme DNS'i tarafından cevaplandıktan sonra bu cevap ara DNS sunucularda da ön belleklenirler. Bu önbellekleme süresi yetkili DNS sunucusunun vereceği TTL (Time to Live -Yaşam Süresi) ile belirlenir. Bu TTL süresi dolana

kadar ara DNS sunucular yetkili DNS sunuculara sorgu yapmazlar ve istemciler'in isim-ip dönüşüm sorgularını kendileri cevaplarlar. TTL süresi dolduktan sonra ya da önbelleklenen isim-ip dönüşüm kayıdı yoksa, yetkili DNS sunuculara isim-ip dönüşümünü yapmak için yeniden sorgu yapılır.

Önbelleklemeyi ortadan kaldırmak için küme DNS'i TTL süresini 0 ya da 0'a yakın verebilir. Ancak ara DNS sunucuları bazen çok küçük TTL değerlerini kabul etmeyebilir. Diğer yandan istemci üzerindeki önbelleklemede TTL süresi tutulmaz.

DNS temelli yük dengeleme yapılarında iki farklı algoritma kullanılır. Bunlar Sabit Süreli TTL algoritması ve Uyumlaşan Süreli TTL algoritmasıdır.

Sabit Süreli TTL Algoritmalar

Bu algoritmada DNS'in hangi sunucuyu seçeceği, sistem durumuna göre, yani istemci yükü ya da istemcinin konumu, sunucu yükü yada bunların birleşimi şeklinde sınıflandırma yapıldıktan sonra ortaya çıkar.

Bu şekilde algoritma kullanan DNS sistemlerine örnek olarak RR-DNS verilebilir. Bu sistemlerin ilk uygulaması NCSA'da yapılmıştır [55]. NCSA web sistemlerine artan yükü dengelemek için DNS tabanlı bir çözüm öngörmüş ve DNS sunucusunu web sunucuların adreslerini çözerken dairesel olarak (Round Robin) dağıtan bir sistem tasarlamışlardır. Ancak bu sistem yük dağılımını ara DNS'lerin önbelleklemelerinden dolayı tam olarak gerçekleştirememiştir. Diğer yandan web sunucuların göçmesi durumunda bunları tespit edecek mekanizmalara sahip olmadığı için tam anlamıyla yük dengeleme başarılamamıştır.

Uyumlaşan Süreli TTL Algoritmalar

Sunucu sistemleri üzerine yükün yoğunluğu DNS TTL süresi ile kontrol edilebilir. Sabit süreli TTL algoritmaları istemcilerden gelen isteklerde oluşan kaymaları yada

olası sunucu kapasitesindeki ayrılıkları adresleyemez, bu yüzden değişken (uyumlaşan) TTL süreli algoritmalar geliştirilmiştir [56].

Bu sistem, her adres çözüm isteğinde dağıtık sistem yükü ve her bir sistemin kapasitesine göre TTL değerini değiştirmektedir. Yük bir sistem üzerinde artırılmak istendiğinde TTL süresini artırmakta, tam tersi durumda ise TTL süresini azaltmaktadır.

2.7.3. Yük dengeleyici tabanlı yaklaşımlar

Yük dengeleyici, istemciler ile sunucu kümesi arasında yer alan ve vekil sunucu gibi davranan sistemlere verilen isimdir. Bu alanda çalışan sistemler yük dengeleyici ya da dağıtıcı gibi isimlerle anılmaktadırlar.

Yük dengeleyicilerin servis ip'si (leri) olur ve bu ip'de (ler) aynı zamanda küme ip'si olarak adlandırılır. Küme ip'si küme içindeki bütün sistemlerin, dışarı doğru sanki tek sistem gibi gösterilmesine yardımcı olur.

Yük dengeleyici sistemlerin çalışma yöntemleri birbirinden farklıdır. Bu sistemleri OSI katman modellerinde çalıştıkları katmanlara göre gruplandırmak doğru olacaktır. Bu yüzden bu kısımda yük dengeleyiciler, eğer katman 4 anahtarlamayı, katman 2 paket yönlendirme ile yapıyorsa L4/2, eğer katman 4 anahtarlamayı, katman 3 paket yönlendirme ile yapıyorsa L4/3 ve eğer katman 7'de anahtarlamayı, katman 2'de yaparsa L7/2 ya da katman 3'de yaparsa L7/3 olarak adlandırılmıştır [57].

2.7.4.1. L4/2 katmanı yük dengeleyiciler

İlk dengeleme sistemleri, OSI referans modeline göre L4/2 seviyesinde geliştirilmiştir. Bu sistemler, hem performans hem de ölçeklenebilirlik alanında ciddi yenilikler getirmişlerdir [58]. Bu yapıdaki sistemlerin çalışması şöyledir: Sistemde yük dengeleyici rolündeki makina bir küme ip'sine sahip olur. Aynı zamanda küme içerisindeki uygulama sunucuları da hem kendi ip'lerine hem de ikincil olarak

loopback arabirimlerine verilen küme ip'sine sahip olur. İstemcilerden gelen tüm paketler yük dengeleyici tarafından karşılanır. Gelen paketler yük dengeleyici tarafından alındıktan sonra 2. OSI katmanı seviyesinde MAC bilgisi üzerinde değişiklik yapılarak yerel ağ üzerinde yük dengeleyici tarafından belirlenmiş uygulama sunucusuna gönderilir.

Gelen paket TCP/IP başlangıç paketi ise, yük dengeleyici kendi üzerinde önceden ayarlanmış dağıtım stratejisine göre paketin hangi sunucuya gideceğine karar verir ve paketi gönderirken aynı zamanda da bağlantı tablosuna bununla ilgili bir kayıt ekler.

Eğer bağlantı başlangıç bağlantısı değilse yük dengeleyici, önce kendi üzerindeki bağlantı tablosundan bağlantının öncekilerden birinin devamı olup olmadığına bakar ve bu bağlantı tabloda mevcut ise, önceden seçilmiş küme içindeki sunucuya yönlendirip devamlılık sağlar. Eğer paket bunların dışında bir şekilde yük dengeleyiciye ulaşmış ise o zaman göz ardı edilir.

Bu sistemler sadece istekler yük dengeleyici üzerinden geldiği ve istemcilere cevaplar doğrudan sunucular tarafından verildiği için yüksek performanslıdır. Ancak bu şekilde çalışan sistemlerin mutlaka aynı ağda olması gereklidir.

Yukarıda anlatılan sistem modeline uygun ilk uygulama Bell Laboratuarlarında 1996 yılında geliştirilen ONE-IP uygulamasıdır [59]. Bu çalışmada NetBSD çekirdeğinin değiştirilmesi ile iki ayrı dağıtım metodu ortaya konmuştur. Birinci yöntem'de yük dengeleyiciye gelen paket hash işleminden geçirildikten sonra sonuca göre küme içindeki sunucuların birine yönlendirilir. İkinci yöntem ise istemcilerden gelen bütün paketler, küme içindeki sunucuların tümüne yayım (broadcast) yöntemi ile ulaştırılır [60]. Her bir sunucu üzerinde kendine ait bir filtreleme sistemine sahiptir. Gelen paketler istemci ip'sine göre küme içindeki sunucular tarafından kabul yada ret edilir. Bu iki yöntemde de istemcilerin sunucular üzerindeki yarattığı yükler göz önüne alınmazlar.

Bu sistemlerdeki hata durumu, hem yük dengeleyici hem de sunucular için özel bir uygulama ile sağlanır. İlk metot kullanılırken, sunucularda meydana gelen bir sorun tespit edilirse yük dengeleyici hash tablosunu değiştirir ve yükü ona göre dağıtmaya başlar. Eğer ikinci yöntem kullanılıyorsa, sorun olması halinde tüm küme içerisindeki sistemler durumdan haberdar edilerek her bir sunucunun filtreleme haritalarının değiştirilmesi sağlanır. Diğer yandan yük dengeleyicide sorun olması halinde, beklemede olan dengeleyici devreye girerek işlemlerin devamlılığını sağlar. Ancak bu sistem, durum bilgilerini senkronize edecek mekanizmaya sahip olmadığı için, önceki bağlantıların yeniden kurulması gerekmektedir.

Ticari olarak L4/2 çalışan ilk ürün IBM'in eNetwork yük dengeleyicisidir [61]. Bu ürün 1998 Olimpiyat oyunlarında başarı ile denenmiştir. Bu sistemin OneIP'den farkı dağıtımı küme içindeki sistemler üzerinden aldığı yük bilgilerine göre düzenlemesidir. Yük dağılımını alınan bu veriler ışığında düzenleyerek dağıtımın nasıl yapılacağına karar verir. Ayrıca bu sistemde durum bilgileri aynı zamanda yedekte bekleyen sunucuya aktarılmakta, eğer ana yük dengeleyici sistemde herhangi şekilde bir sorun çıkarsa, yedek sunucu işi kendi üzerine aldığında bile önceki bağlantıların kaybolmaması sağlanmaktadır. Bu özellikle SSL ve FTP protokolleri için de çok önemlidir.

Nebraska-Lincoln üniversitesinin geliştirdiği LSMAC [62] sistemi ise kullanıcı seviyesinde (user level) çalışan bir uygulamadır. Bu sistem eNetwok yük dengeleyicisi ile karşılaştırılacak seviyede performans elde etmiştir. Diğer yük dengeleyici sistemleri gibi bu sistemin de küme havuzu içindeki sunucular üzerindeki hataları tespit etmek için mekanizması mevcuttur. Bu mekanizma, belirli aralıklar ile küme sunucularına ARP isteği göndererek çalışmaktadır. Eğer sunucu ARP isteğine cevap vermiyorsa o zaman aktif sunucu listesinden çıkarılır. Ek olarak geçen paketlerdeki TCP sıfırlama mesajlarına göre de sunucular üzerinde sorun olup olmadığını tespit edebilir.

Zhang'ın Linux işletim sistemi çekirdek seviyesinde yaptığı çalışmalarda ise IBM'in yük dengeleyicisi ile benzer şekilde L4/2 seviyesinde de çalışabilen bir sistem

tasarlanmıştır [63]. Bu sistemde yapılan yönlendirmeye doğrudan yönlendirme (Direct Routing) adı verilmiştir. Ağ üzerine gelen paketler yük dengeleyici tarafından karşılanmakta ve yük dengeleyici üzerindeki yük dağıtım mekanizmalarında verilen karara göre küme içindeki sunuculara paketler MAC adresleri değiştirilerek gönderilir. Burada en önemli sorun, servis sunucuların dışarıda kullanılan küme ip'sine (servis ip'sine) cevap vermemesini sağlamaktır. Diğer yandan bu sistemde tek başına hata toleransını sağlayacak mekanizmalar mevcut değildi. Bu sorun ayrı yardımcı uygulamalar ile çözülmektedir [64].

2.7.4.2. L4/3 katmanı yük dengeleyiciler

Bu şekilde çalışan yük dengeleyiciler OSI 2. katman'da çalışan ve bir önceki bölümde anlatılan yöntemler ile benzerlik göstermektedirler. Ancak L4/2'ye göre daha az performanslı olduğunu söylemek yanlış olmayacaktır. Bu yapının L4/2'den farkı küme içindeki sunucu sistemlerin her birinin kendine ait bir IP'sinin bulunmasıdır. L4/2'deki gibi ayrıca her bir sisteme servis ip'si verilmesine gerek yoktur.

L4/3 çalışan küme, istemci tarafından bakıldığında tek bir sunucu gibi, küme içindeki servis sunucuları tarafından bakıldığında ise ağ geçidi olarak görülür. İstemci tarafından gönderilen istek paketi olağan yönlendirme kurallarından geçerek yük dengeleyicinin küme ip'sine ulaşır.

Eğer gelen paket TCP/IP bağlantısı için başlangıç paketi ise, yük dengeleyici havuz içindeki sunuculardan birini kendi üzerindeki dağıtım algoritmalarına göre seçer. Aynı zamanda kendi üzerindeki bağlantı tablosuna da bağlantının nereden geldiğini ve kendi seçtiği sunucunun ip'lerini ilişkilendirdiği bir kayıt ekler. Daha sonra paketin servis ip'si olan hedef ip adresini önceden seçtiği sunucu ip'sine çevirerek aynı zamanda değişen bu paketin üzerinde bütünlük işlemleri de (CRC, Sağlama toplamı) yaparak sunucuya gönderir.

Eğer gelen paket bağlantı başlatma paketi değilse, yük dengeleyici önce kendi üzerinde tuttuğu bağlantı tablosundan yararlanarak bu paketin önceden bağlantı kurmuş bir sistem ile ilgili olup olmadığına karar verir. Eğer gelen paket önceki bağlantılarla ilişkili ise, bağlantı tablosunda önceden belirlenmiş servis sunucusuna yukarıda anlatıldığı gibi hedef ip güncellenerek aktarılır. Aksi takdirde paket göz ardı edilir.

Bu dağıtım yapısında bütün servis ip'sine gelen paketlerin dengeleyici üzerinden akması gerekmektedir. Bu yöntemle ilgili daha ayrıntılı bilgi RFC 2391'de [65] verilmiştir.

Berkeley'de bu konuda yapılan çalışmalarda Magic Router adı verilen bir uygulama geliştirilmiştir [66]. Bu uygulama için Linux işletim sistemi çekirdeğinde değişiklikler yapılmıştır. Sistem üç adet yük dengeleme metodu ve hata toleransı yöntemini desteklemektedir. Sistem yük dağıtımına karar vermeden önce, her bir sunucu üzerindeki yük ve aktif bağlantı sayısı ile orantılı olan bir mekanizma çalıştırır. Bu oran yardımı ile trafik her zaman en az yükü olan sunucuya aktarılır.

Yapı içindeki hata tespiti ARP isteği veya TCP paketlerinde oluşan sıfırlama (reset) ile tespit edilir. İşlemin çalışma prensibi önceki bölümlerde anlatılan LSMAC sistemindeki ile aynıdır. Dengeleyicide sorun olması halinde yedekte bekleyen yük dengeleyici görevi kendi üzerine alarak sistemin devamlılığını sağlar.

Cisco'nun geliştirdiği LocalDirector [67] ise L4/3 metodunu kullanan ilk ticari örneklerdendir. Bu sistem dengeleme metotlarının yanında, durum tablosunu yedekte bekleyen yük dengeleyiciye de aktarabilecek mekanizmalara sahiptir.

Nebraska-Lincoln üniversitesinin geliştirdiği LSNAT [62] ise RFC 2391'de tanımlanan IPNAT sisteminin kullanıcı seviyesinde çalışan bir uygulamasıdır. Aynı LSMAC uygulamasındaki gibi Linux işletim sistemi kütüphanelerini kullanarak çalışacak şekilde tasarlanmıştır. Bu sistemde hata tespiti ve toleransı için geliştirilen yöntemler vardır. Ancak bu sistemin diğerlerinden farkı, beklemede bir sunucu

tutmak yerine, sunucu kümesi içerisindeki tüm sunucuların yük dengeleyici rolünü üstlenebilmesidir. Ayrıca durum tablosu tüm sistemlere dağıtılmaktadır. Sunucu havuzundaki sunucuların birinde sorun olması durumunda, yine aynı şekilde sorunlu sistem, yük dengeleyici tarafından aktif sunucu listesinden çıkartılır. Aynı zamanda diğer sunucularda üzerlerinde çalışan bir uygulama ile bunun senkronizasyonunu sağlamaktadır.

Linux LVS (Linux Virtual Server) projesinde [63] çekirdek seviyesinde (kernel – level) çalışan bir model geliştirilmiştir. Çalışma mantığı temel olarak RFC 2391'e tamamen uygundur. Bu sistemde de istemcinin ulaşması için bir servis ip'si bulunmakta ve bütün istekler bu servis ip'sine ulaşmaktadır. Daha sonra istekler dağıtım algoritmaları yardımı ile, servis sunucu kümesindeki seçilmiş sunucuya, hedef ip adresinde değişiklik yapılarak aktarılır. Hata toleransı ve bulunurluk konusunda sistem özel mekanizmalar içermez. Bu işlemler yardımcı programlar tarafından yerine getirilir [64, 68, 69].

2.7.4.3. L7 seviyesinde çalışan dengeleyiciler

OSI katmanı L4/2 ve L4/3'de çalışan dengeleme sistemleri birçok dengeleme problemini çözmüştür. Ancak bunun yanında L7 (Uygulama Katmanı) seviyesinde dengeleme yapacak sistemler üzerinde çalışmalar başlamıştır. Bu katmanda çalışan dengeleyicilere İçerik-Tabanlı dengeleyiciler adı verilmektedir. Bu sistem modelinde istemcilerden gelen paketler içeriklerine göre küme içindeki servis sunucularına dağıtılmaktadır.

LARD [70] Rice Üniversitesinde geliştirilen içerik-temelli dengeleme sistemidir. Bu sistemde küme içinde bulunan her bir servis sunucusu ayrı uygulamalar için içerik servisi sunmaktadır. Bu yapıyı daha iyi anlatmak için bir örnek üzerinden gitmek daha doğru olacaktır. İçerik servisleri metin tabanlı içerik için M ve resim yada video içerikleri için R ve V olarak tanımlandığı varsayılırsa, M içeriği hizmeti bir sunucuda, R ve V içeriklerinin de başka bir sunucu tarafından verildiği kabul

edilirse, dengeleyici istemcilerden gelen paketleri alıp inceledikten sonra içeriğine göre sınıflandırarak ilgili içerik sunucularına dağıtabilme yeteneğine sahiptir.

Bu sistemin çalışabilmesi için hem dengeleyici hem de sunucu çekirdeğinde bağlantıyı taşıyacak TCP-Handoff [71] desteğinin olması gereklidir. Bu özellik ile dengeleyici tarafından karşılanıp paket içeriği sınıflandırılan paket arka taraftaki sunuculara aktarılabilmektedir.

2.8. Paralel Yük Dengeleme Sistemleri

Bu tezin konusu paralel yük dengeleme sistemlerinde referans yerelliği temelli bir sistem modeli sunmak olduğu için, paralel yük dengeleme sistemlerindeki önceki çalışmalar da önem taşımaktadır. Bu sistemlerin ortak özellikleri diğer sistemlerden farklı olarak birden fazla yük dağıtım sisteminin paralel şekilde çalışarak gelen trafik yükünü dengelemesidir. Bunu yerine getirirken de yüksek bulunur şekilde çalışmaya devam etmesidir.

Bu konudaki çalışmalardan biri Verwoerd ve Hunt'ın [72] geliştirmiş oldukları GLOB (Generic Load Balancing) sistemidir. Bu sistemde yük dengeleyiciler paralel olarak çalışmaktadırlar. Sistem tek yük dengeleyici sistemlerin yetersiz kaldığı durumlar için geliştirilmiştir. Sistem içindeki dengeleyiciler diğer yük dengeleyici sistemlerden çok farklı şekilde çalışmamaktadır. Dengeleyici tüm sistemlere ortak sanal bir servis IP'si ve yine ortak bir MAC adresi verilir. Bu sistem bir HUB cihazına bağlanır. Özel olarak geliştirilmiş bir yazılım sayesinde yük dengeleyiciler üzerindeki firewall kuralları dinamik olarak bütün sistemler eşit yüke gelinceye kadar değiştirilir. Bu işlem için uygulanan metot şöyledir; Uygulama yazılımı belirli aralıklar ile her bir dengeleyici sistem üzerindeki yükü firewall istatistiklerinden alarak belirlediği ortalama yük değerini bulur. Daha sonra yük dengeleyici bu değere göre fazla yükte olan sunucu yükünü daha az olanlara aktarır. Bu işlemi ip'lerin kaynak ve hedef port numaralarının bölünmesi ile yerine getirilmektedir. Ancak internet trafik yüklerinde bu şekilde port numaralarına göre yükü tanımlamak uygun sonuçlar vermeyebilir.

Buna benzer bir başka çalışma ise Xiong, Yan ve Guo'nun [73] MDDR sistemidir. Bu sistem GLOB sistemindeki gibi paralel çalışan birden fazla yük dengeleyiciye sahiptir ve gelen yükün dağıtımı istek paketlerinin ip kaynak port numarasını matematiksel mod işleminden geçirerek yapılmaktadır. Sistemin çalışması Eş. 2.11'deki verildiği gibidir.

p: port numarası ve m: makine sayısı olarak i: de makinenin kendisini gösterdiği kabul edilirse;

$$p \bmod m = \begin{cases} i & Kabulet \\ Diger & \text{Redet} \end{cases} \tag{2.11}$$

Bu sistemde de bir servis ip'si ve yine aynı MAC numarasının verilmesi zorunludur. Paketler paralel çalışan yük dengeleyicilerin tümüne gelir, bu dengeleyicilerde paketleri filtreleme modülü sayesinde kabul ya da ret eder. Bu modül her bir sisteme elle ayrı ayrı sistem numarasını parametre olarak verilerek girilir. Böylece yukarıdaki kural dahilinde yük dengeleyiciler istemciden gelen paketin, kaynak port numarasına göre paketleri kabul edip etmeyeceklerine karar verirler. Bu sistemde hata toleransının ve yük dengelemenin tam anlamı ile olduğunu söylemek çok güç olacaktır.

3. İNTERNET TRAFİĞİ YERELLİĞİNİN İNCELENMESİ

Ağ trafikleri Bölüm 2'de ele alındığı gibi referans yerelliği yardımı ile modellenebilmektedir. Bu çalışmada ortaya konan paralel yük dengeleme sistemi için, referans yerelliği yaklaşımı temel alındığından çeşitli ağ protokollerinin içerisindeki trafik referans yerelliğinin varlığının gösterilmesi gerekmektedir.

Bu yüzden ilk olarak farklı protokoller içeren ağ trafiklerinin toplanabilmesi için hangi yöntemlerin kullanıldığı ve toplanan bu veriler ile ilgili ayrıntılara yer verilmiştir.

Daha sonra farklı protokoller için toplanan ağ trafikleri analizleri yapılarak referans yerelliği taşıdıkları gösterilmiştir.

En son bölümde ise farklı protokoller için matematiksel mod işleminin benzetimi yapılarak sonuçlar tartışılmıştır.

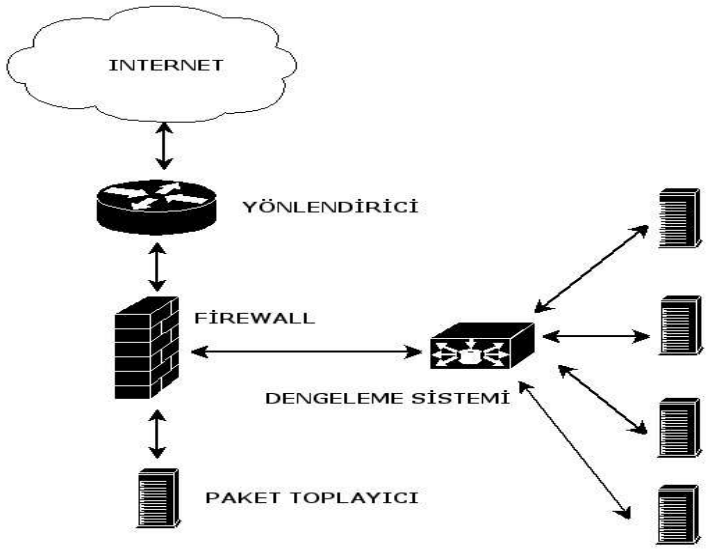
3.1. Ağ Trafiği Toplama

Ağ trafiği toplama işlemi, ağ üzerindeki sorunları tespit edebilmek, yeni protokollerin denenmesi sırasında davranışlarını görebilmek, ağ üzerinden akan yükün tanımlanabilmesi ve performans değerlendirmelerini yapabilmek için gereklidir [74].

Yukarıda verilen gereksinimler doğrultusunda, ağ trafiğindeki akışları tanımlamak ve performans değerlendirmeleri yapabilmek, yük dengeleme sistemi tasarımına yardımcı olacaktır. Aslında hazır trafik akış verileri [75,76] bu işlem için kullanılabilmektedir. Ancak bu veriler üzerinde gizliliği sağlayabilmek için değişiklikler yapılmaktadır [77]. Dengeleme sistemi tasarımında sentetik ya da değiştirilmiş trafik verilerinin kullanılması yanlış sonuçlar alınmasına yol açabileceğinden bu çalışmada bu türde veri kullanmaktan kaçınılmıştır. Bu yüzden yeniden trafik akış verisi toplanmıştır.

Trafik toplama işlemi için en uygun yer, izlenmesi istenilen trafiğin tümünün geçtiği noktalar olmalıdır. Toplama işlemi ana yönlendirici yada servis noktalarının bağlı olduğu anahtarlama sistemleri üzerinde yada kabloları üzerinde port-miroring veya ağ dinleme (network tapping) [78] gibi fiziksel yöntemlerle yapılabilmektedir. Diğer yandan trafik toplama işlemi uygulama katmanında da yapılabildiği için bu işlemin yazılım yardımıyla yapılması uygun görülmüştür. Bu işlem için en uygun yer olan Gazi Üniversitesinin tüm trafiğinin geçtiği ateş duvarı (firewall) sistemi üzerinden örnek ağ trafikleri toplanmıştır.

Yapılan araştırmalar sonucu, yazılımsal trafik toplama araçlarından [79] birçoğunun PCAP [80] kütüphanesi temelli çalıştığı tespit edilmiştir. Her bir uygulamanın kendine has özellikleri olmasına rağmen trafik toplama için gerekli olan akış verilerini temel olarak tanımlayacak [44] ve akışların zaman verilerini tutan alanlar içerecek bir uygulamanın kullanılması daha uygun olacaktır. Bu yüzden hem endüstriyel bir standart olan hem de birçok analiz programı tarafından da işlenebildiği için akış verilerini tutan Netflow 5 [81] formatında toplayabilen ve kaydedebilen softflowd/flowd [82] uygulamaları seçilmiştir. İki ayrı uygulamanın kullanılmasının nedeni, firewall sisteminin asli görevini yerine getirmesine engel olmamaktır.



Şekil 3. 1. Trafik toplama sisteminin yapısı.

Şekil 3.1'de trafik verisi toplama sistemi verilmektedir. Farklı protokoller için toplanması istenen trafik verileri firewall sistemi üzerine kurulan softflowd uygulaması ile toplanarak akış toplayıcı olarak belirlenmiş başka bir sunucuda çalışan flowd uygulamasına aktarılmıştır.

Bu sistemde softflowd uygulaması BPF (Berkeley Packet Filter) [83] desteklediği için izlenmesi istenilen servislerin ip'lerine göre filtreleme yapılarak bütün akış içerisinden sadece ilgili ip'lere giden veri akışları kaydedilmiştir.

Bu şekilde HTTP, SMTP ve HTTPS olmak üzere üç farklı protokol için akış verisi toplanmıştır. Çizelge 3.1-3'de toplanan verilerin ayrıntısı verilmiştir.

Çizelge 3. 1. HTTP protokolü için toplanan verilerin ayrıntısı.

Akış başlama zamanı	11:59:03 01/02/2005
Akış bitiş zamanı	20:39:01 06/02/2005
Akış sayısı	1769436
Tekil IP Sayısı	38013
Geçen veri miktarı	2967553085 Byte
Geçen paket sayısı	29788460 Adet

Çizelge 3. 2. SMTP protokolü için toplanan verilerin ayrıntısı.

Akış başlama zamanı	17:09:07 24/06/2005
Akış bitiş zamanı	14:41:30 06/09/2005
Akış sayısı	2498052
Tekil IP Sayısı	248380
Geçen veri miktarı	177029900952 Byte
Geçen paket sayısı	201920248 Adet

Çizelge 3. 3. HTTPS protokolü için toplanan verilerin ayrıntısı.

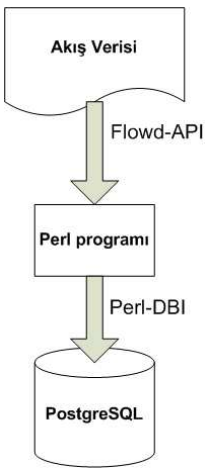
Akış başlama zamanı	01:21:23 22/09/2005
Akış bitiş zamanı	17:16:12 11/10/2005
Akış sayısı	961281
Tekil IP Sayısı	37013
Geçen veri miktarı	1237592569 Byte
Geçen paket sayısı	13211408 Adet

3.2. Ağ Trafiğinin Veritabanına Aktarılması

Netflow 5 formatında toplanan akış verileri ham veri olarak kaydedilmiştir. Bu ham veriler üzerinde işlem yapabilmek kolay olmadığı için bu verilerin daha kolay biçimde işlem yapılabilecek bir ortama geçirilmesi trafik tanımlama ve analiz sürecine yardımcı olacağına karar verilmiştir. Bu yüzden, verilerin bir Veri Tabanı Yönetim Sistemi'ne (VTYS - RDBMS Relational Database Management System) [84] aktarılması, veriler üzerinde hızlı ve etkili analiz için gereklidir.

Bunun için PostgreSQL (VTYS) [85] kullanılmıştır. Bu VTYS'nin seçilmesinin nedeni ip işlemleri ile ilgili fonksiyonlara sahip olması ve aynı zamanda dıştan yordam (procedure) eklenebilmesine olanak tanımasıdır.

Ham haldeki trafik akış verileri kendileri ile ilgili VTYS tablolarına Şekil 3.2'de gösterildiği gibi aktarılmıştır. Aktarma işlemi Perl [86] programlama dili ile yazılmış bir betik aracılığı ile yapılmıştır.



Şekil 3. 2. Akış verisinin PostgreSQL VTYS'ye aktarılması işlemi.

Akış verileri için VTYS içerisinde yapılandırılan tablo alanları Çizelge 3.4'de verilmiştir.

VTYS'ye kaydedilen akış verileri üzerinde işlem yapabilmek için SQL [84] kullanılmıştır.

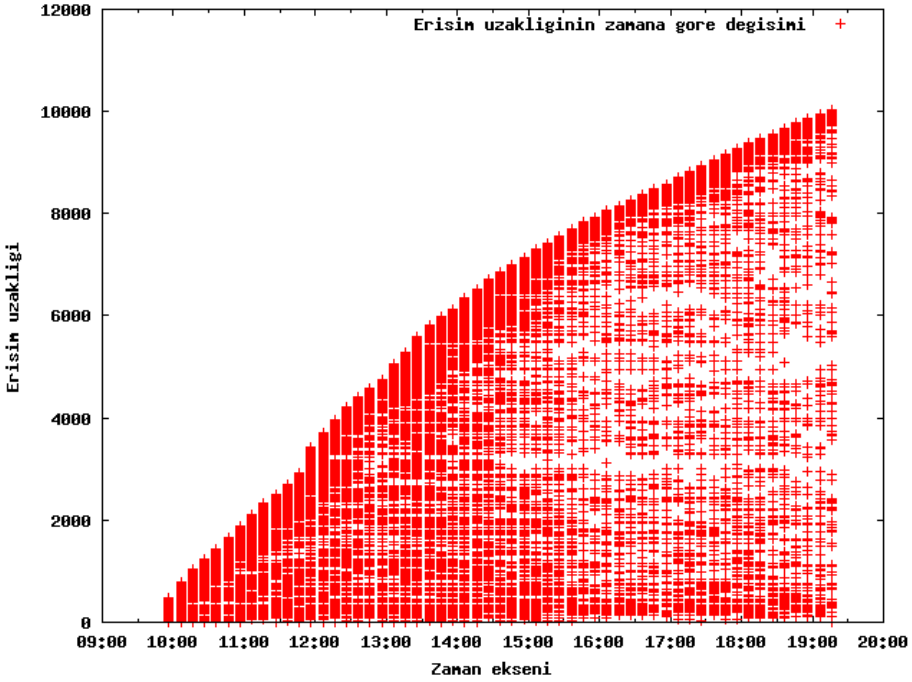
Çizelge 3. 4. Akış verileri için VTYS'de kullanılan tablo alanları.

Alan adı	Açıklama
Src_ip	Akışın kaynak ip'si. Servisi alan istemcinin ip'sidir.
Src_port	Akışın kaynak portu. Servisi alan istemcinin bağlantı portu.
Dst_ip	Akışın hedef ip'si. Servis veren yerin ip'si.
Dst_port	Akış hedef portu. Servis veren yerin servis portu (HTTP,SMTP,HTTPS)
Packet_count	Akış içindeki paket sayısı.
Packet_octets	Akış içindeki akan trafiğin byte cinsinden miktarı.
Flow_start	Akışın başlama zamanı.
Flow_stop	Akışın bitiş zamanı.

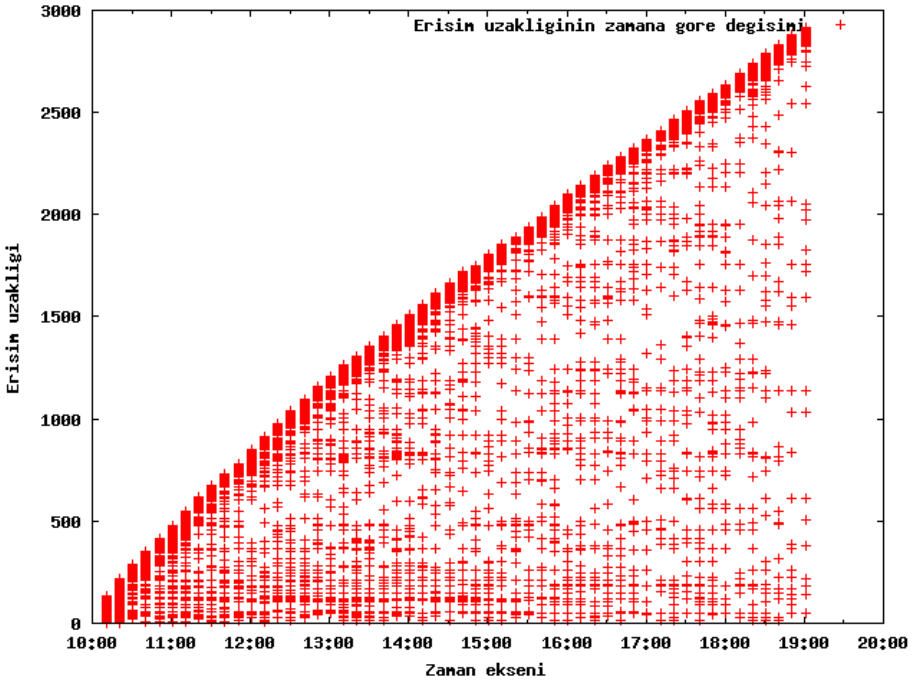
3.3. Trafik Verilerinin Erişim Uzaklığı ile Modellenmesi

Toplanan ağ verilerinin, Bölüm 2.4.6'da anlatılan erişim uzaklığı yöntemi ile modellenmesi, kaynak ip'lerin örnek trafikler içerisindeki davranışları hakkında daha ayrıntılı bilgi edinilmesine yardımcı olmuştur.

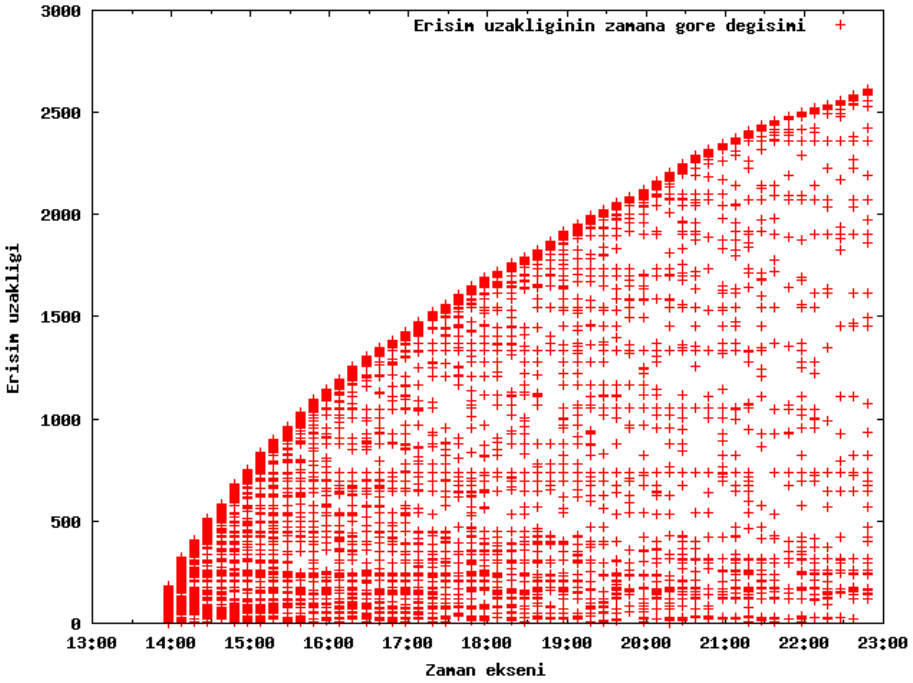
Modelleme için üç farklı protokol için toplanan veriler üzerinde Eş. 2.10 uygulanmıştır. Üç protokol için fonksiyonun uygulanmasından sonra zaman ekseninde oluşturdukları grafikler Şekil 3.3-Şekil 3.5 gösterilmiştir. Ele alınan her protokol birbirinden farklı özellikler gösterdiği için HTTP protokolü için 10000 tekil kaynak ip, HTTPS ve SMTP protokolleri için ise dokuz saatlik zaman aralığında erişim uzaklığı grafikleri çıkarılmıştır.



Şekil 3. 3. HTTP protokolünün erişim uzaklığının zamana göre değişimi.



Şekil 3. 4. HTTPS protokolünün erişim uzaklığının zamana göre değişimi.



Şekil 3. 5. SMTP protokolünün erişim uzaklığının zamana göre değişimi.

Grafikler incelendiğinde iki tane sonuca ulaşmak mümkündür:

- İlk defa erişim yapmış ip'ler yığında ilk defa eriştikleri için üst sınırda toplanarak yoğunlaşırlar.

- Üst sınırın altındaki kısımlardaki erişim uzaklıkları yeniden erişim yapan ip'leri ifade eder.

Bu sonuçlardan yola çıkarak üst sınır altında kalan noktaların zaman ekseninde ilerlemeye rağmen oluşması, ağ trafiklerinin referans yerelliği davranışını gösterdiğini ortaya koyar.

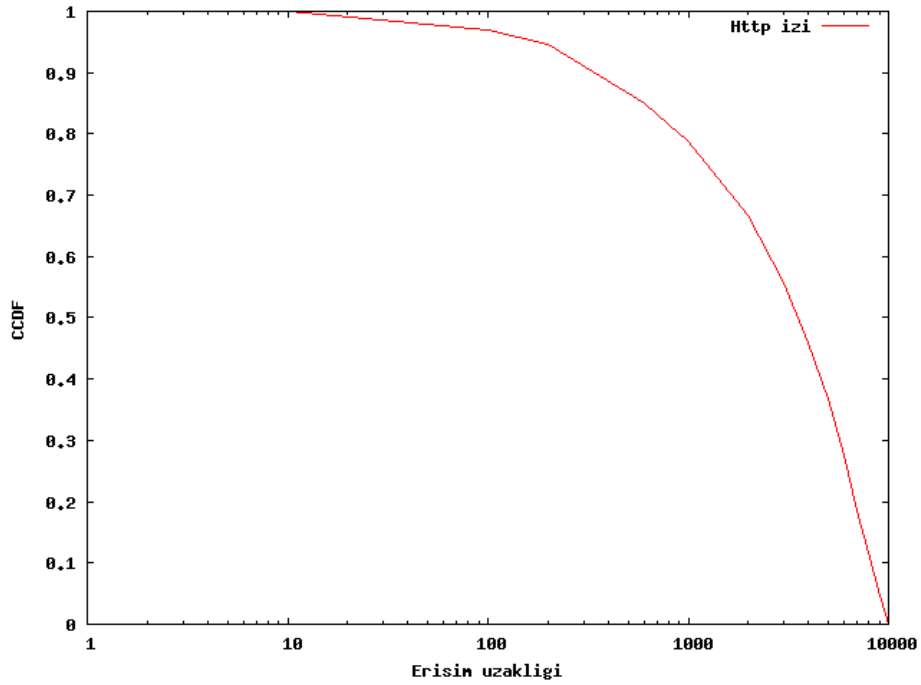
Referans yerelliğini daha ayrıntılı göstermek için istatistikde kullanılan CCDF (Complementary cumulative distribution function) kullanılacaktır [87].

3.3.1. Erişim uzaklığının CCDF yöntemi ile gösterilmesi

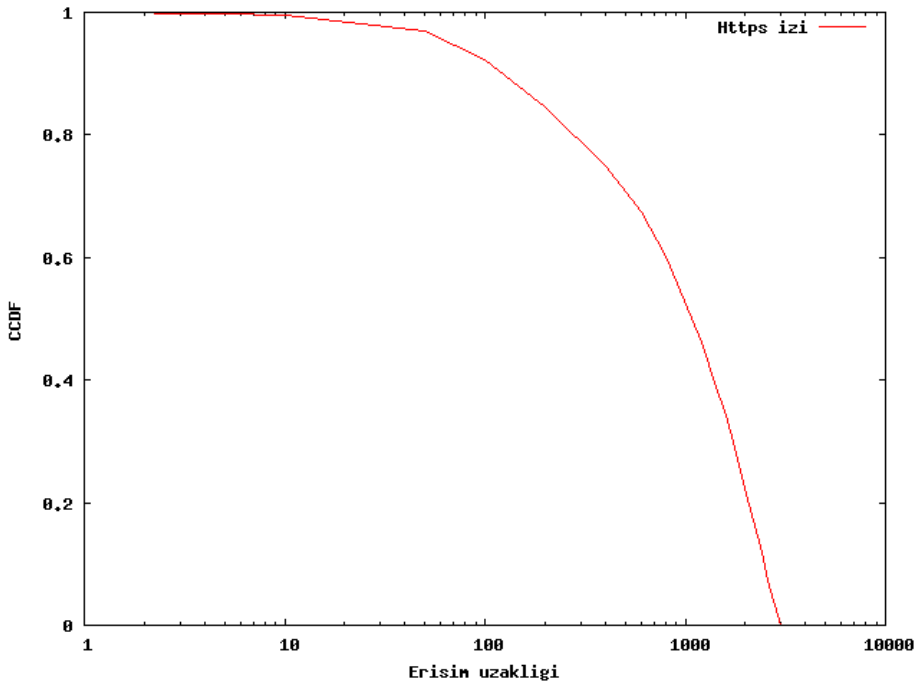
Trafik verilerinin referans yerelliklerini gösterebilmek için CCDF yöntemi kullanılacaktır. CCDF yöntemi yeniden erişim olasılığını daha iyi yakaladığı [32] ve önerilen dengeleme sisteminin verimliliğini etkileyen faktörleri gösterdiği için önemlidir. CCDF Eş. 3.1'de gösterilmektedir. Burada F_c CCDF fonksiyonunu, P olasılık fonksiyonunu ve F ise CDF (Cumulative Distribution Function) göstermektedir. CCDF verilen x değerinden büyük olabilecek X değişkenlerin olasılığını ölçer.

$$F_c(x) = P(X > x) = 1 - F(x) \tag{3.1}$$

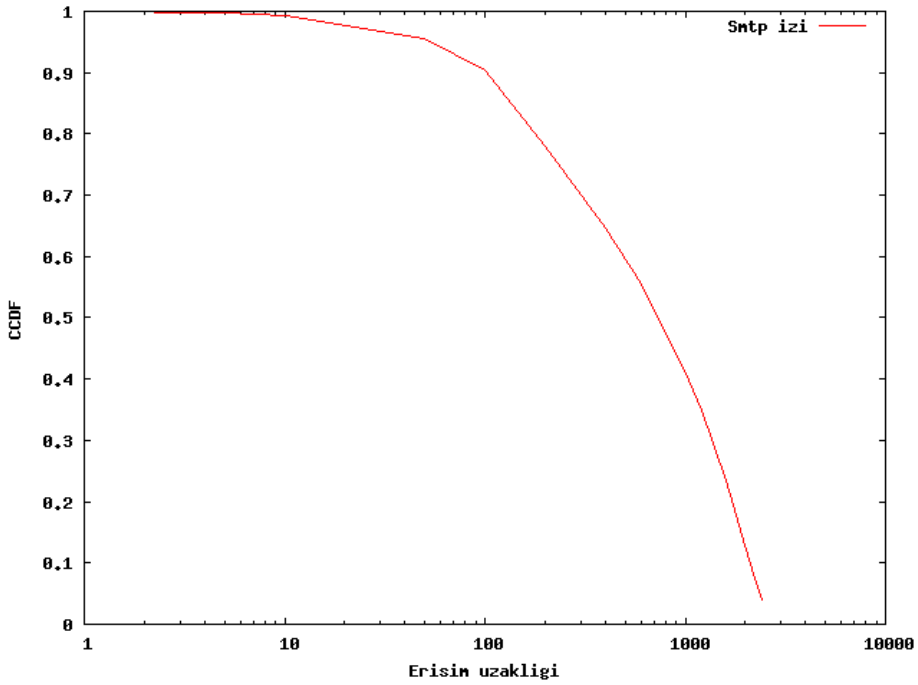
Üç farklı protokol için toplanan veri izleri üzerinde Eş. 3.1 uygulanarak Şekil 3.6-8'deki grafikler elde edilmiştir.



Şekil 3. 6. HTTP protokolünün erişim uzaklığının CCDF değerleri.



Şekil 3. 7. HTTPS protokolünün erişim uzaklığının CCDF değerleri.



Şekil 3. 8. SMTP protokolünün erişim uzaklığının CCDF değerleri.

Farklı ağ protokolleri için Şekil 3.6-8 gösterilen CCDF eğrileri her bir protokolün referans yerelliği davranışlarını göstermektedir. Belirli ip adreslerinin sürekli olarak erişim yapması erişim olasılığını 1'e yaklaştırmaktadır. Diğer yandan daha az erişim yapılması yada erişim uzaklığı yığınına ilk defa giriş yapılması değerleri 1'den uzaklaştırmaktadır. Grafiklerdeki dikey düşüşlerin sebebi ise örneklemenin kısıtlı ağ izleri üzerinde yapılmasıdır.

Akış verilerinin sonsuz uzaklıkta olması dikey düşüşleri engelleyecektir. Ancak bu seferde ağ trafiği içindeki farklı erişim sayılarından dolayı aşağı doğru eğimli bir eğri oluşması beklenir.

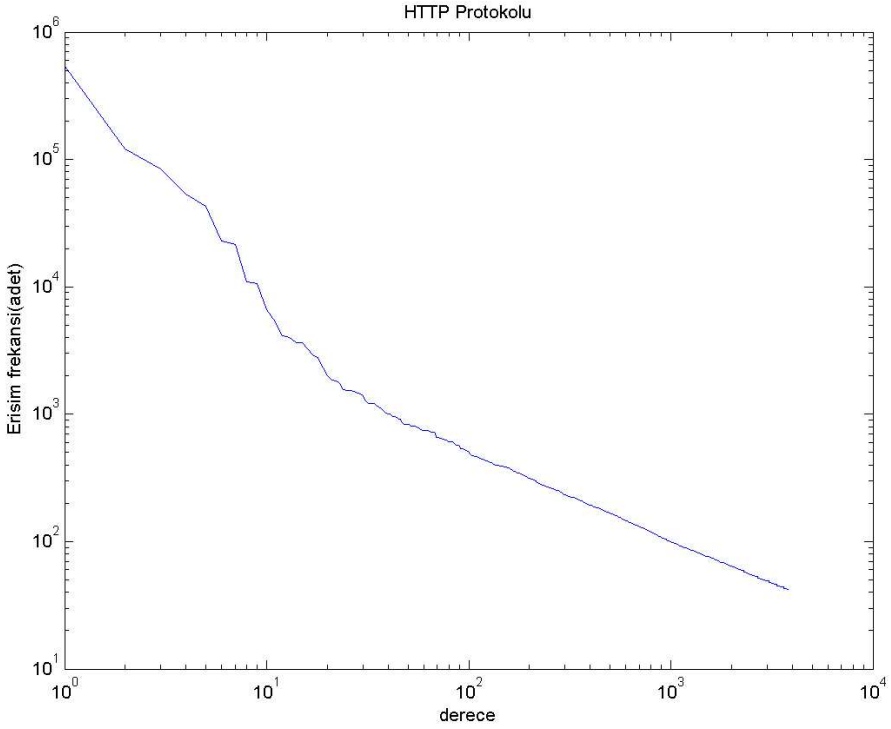
3.4. Trafik Verilerinin Akış Seviyesindeki Karakteristiği

Önceki bölümde ip'lerin erişim uzaklığı modeli ile davranışları açıklanmaya çalışılmıştır. Bu bölümde ise bu ip'lerin yaratmış oldukları akışların (flows) davranışları ele alınacaktır.

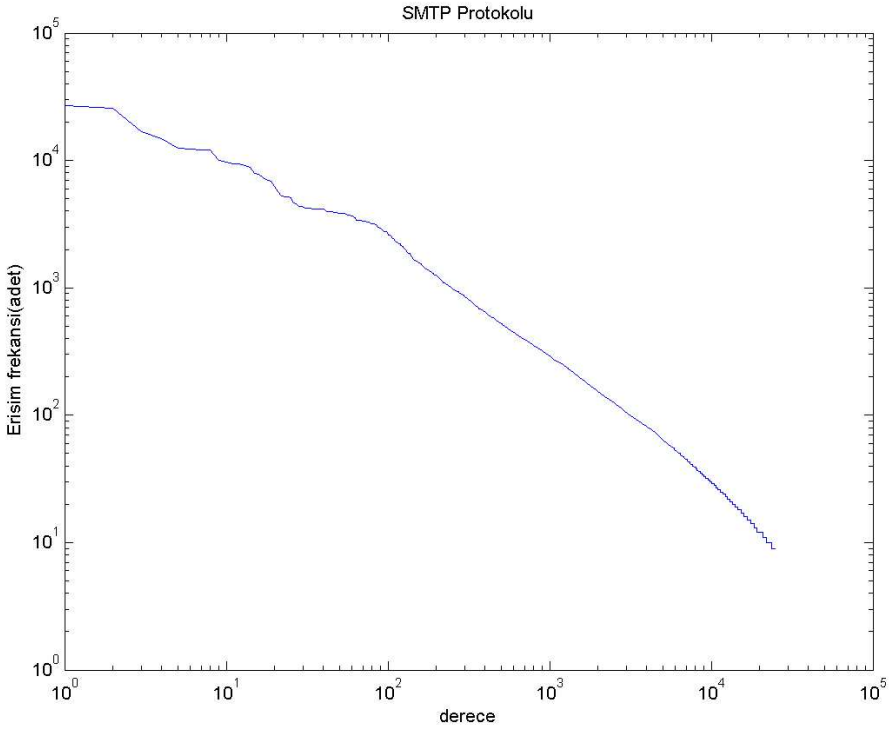
Trafik verileri üzerinde yapılan çalışmalar sonucu her üç trafik verisinde de bir miktar ip'nin diğerlerinden çok daha fazla trafik yarattığı tespit edilmiştir. Bu verilerin bölüm 2.5.1'de bahsedilen referans yerelliği ile örtüştüğü tespit edilmiştir. En dikkat çekici ise olanı ise CAIDA'daki araştırmadaki [39] trafik davranışları ile benzerlik göstermesidir.

Şekil 3.9-3.11'de görülebildiği üzere tüm trafik verileri üzerinde en fazla erişimi oluşturan %10'luk kısım grafiğe alınmıştır. Protokollere ait grafikler, en fazla erişimi oluşturan ip numaralarının en yüksek erişimden – en düşük erişim yapana göre sıralanması sonucu aldığı sıra numarasına göre (dereceye göre) çizilmiştir. Şekil 3.9-3.15'de x-ekseni yine aynı sıralamaya (derece) karşılık gelmektedir.

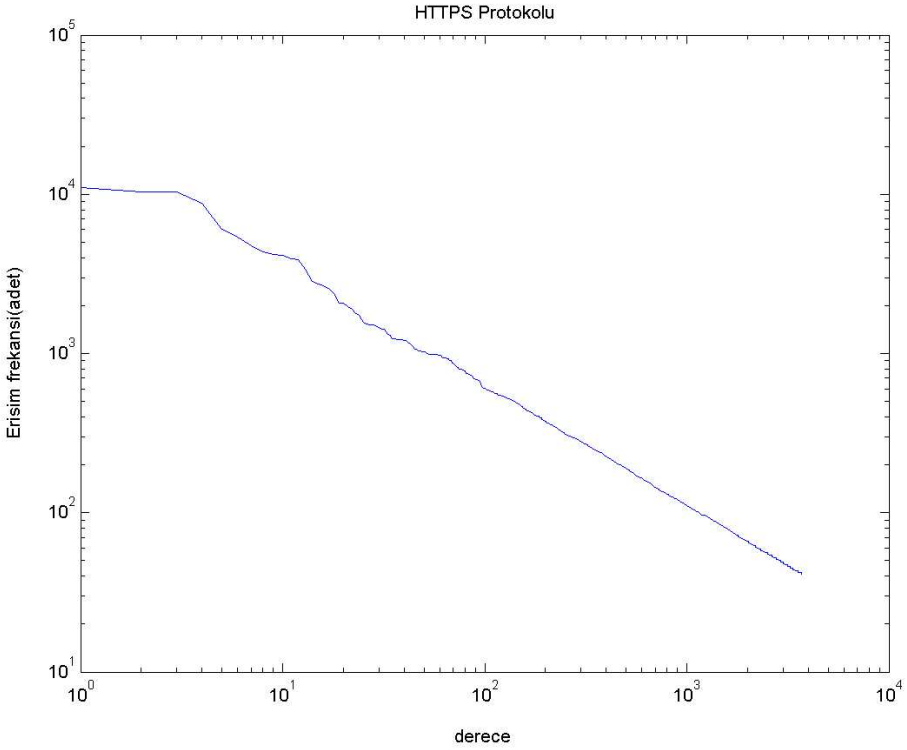
En fazla erişim frekansına sahip IP numaraları frekans olarak daha sık erişim yaptığından bu tür ip'lerin topluluğuna "referans yerelliği" yapıyor denir ve toplam trafik içerisinde genelde %10'luk kısmı oluştururlar [39,40].



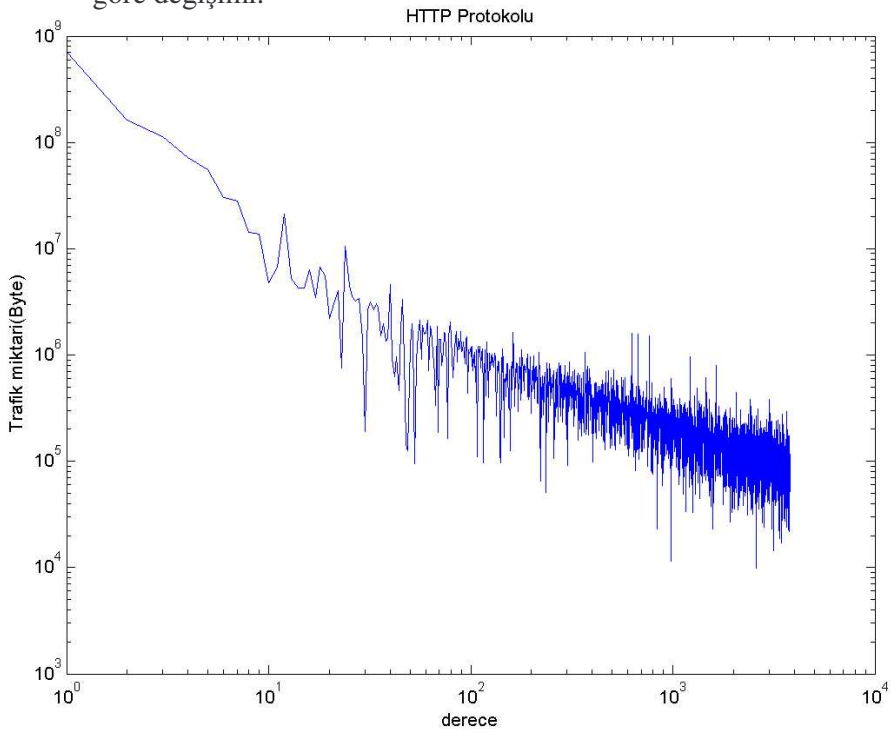
Şekil 3. 9. HTTP protokolü için erişim frekansının dereceye (sıra numarasına) göre değişimi.



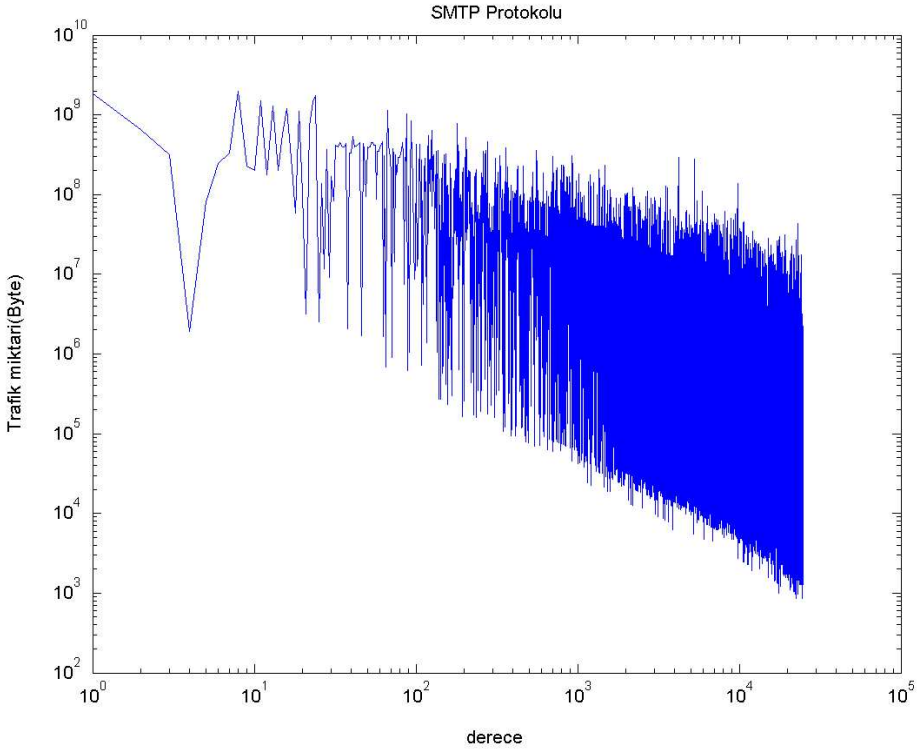
Şekil 3. 10. SMTP protokolü için erişim frekansının dereceye (sıra numarasına) göre değişimi.



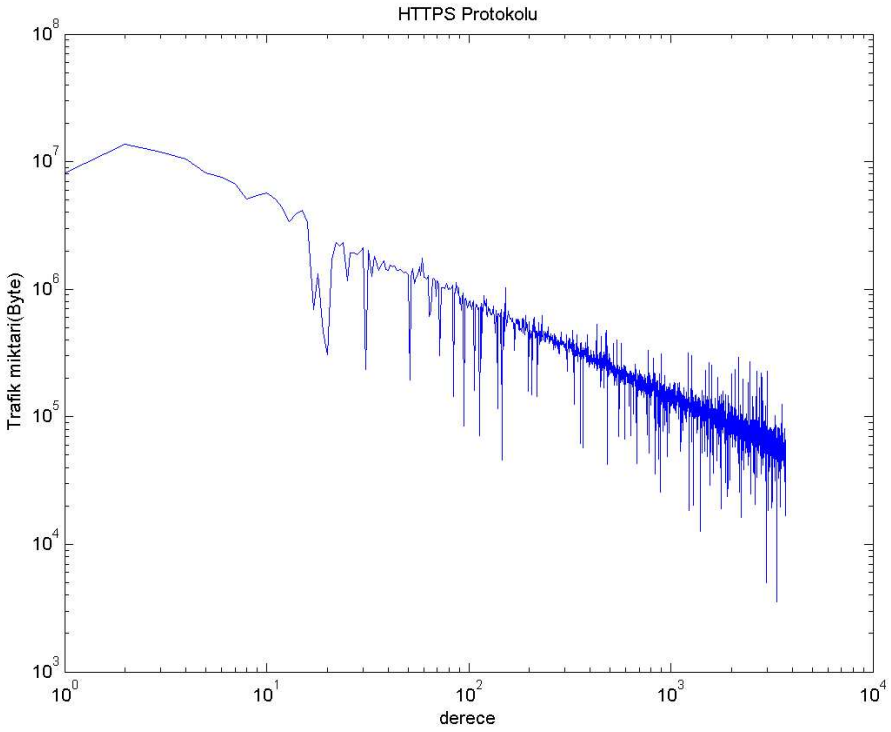
Şekil 3. 11. HTTPS protokolü için erişim frekansının dereceye (sıra numarasına) göre değişimi.



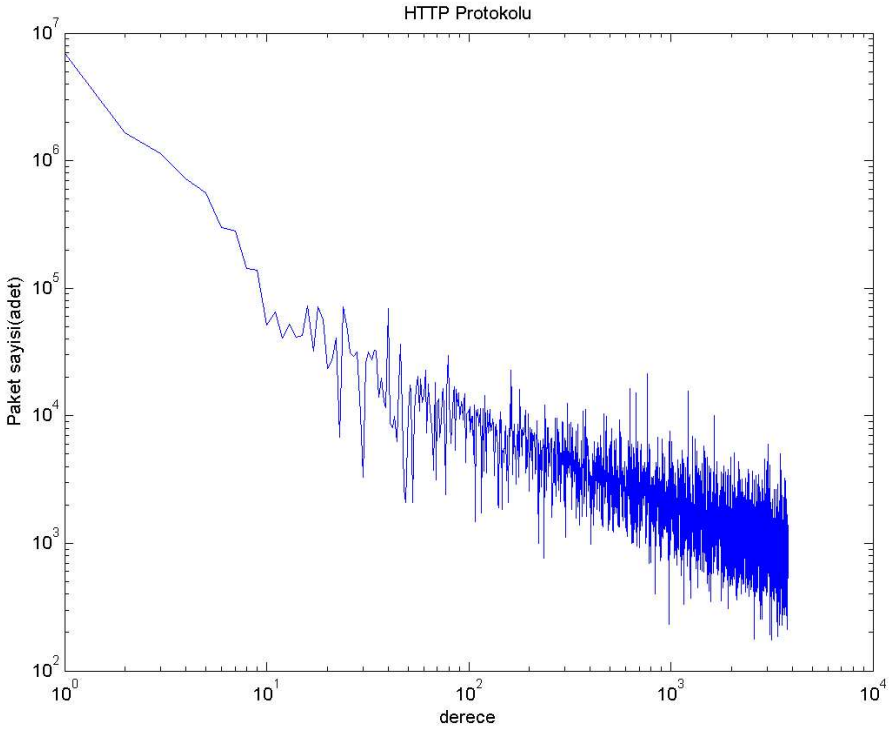
Şekil 3. 12. HTTP protokolü için veri miktarının dereceye (sıra numarasına) göre değişimi.



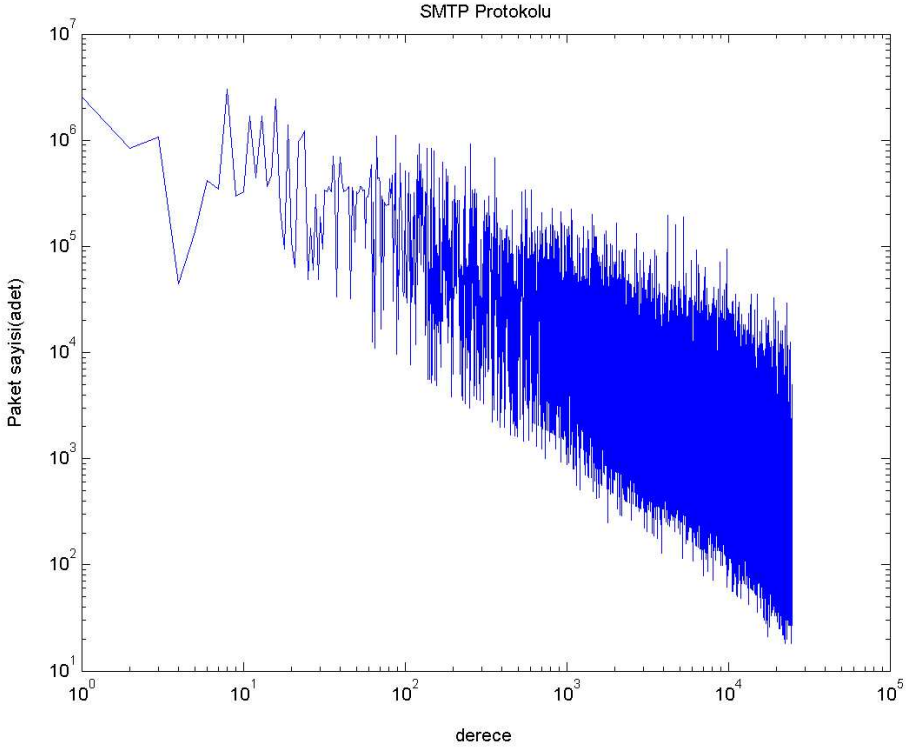
Şekil 3. 13. SMTP protokolü için veri miktarının dereceye (sıra numarasına) göre değişimi.



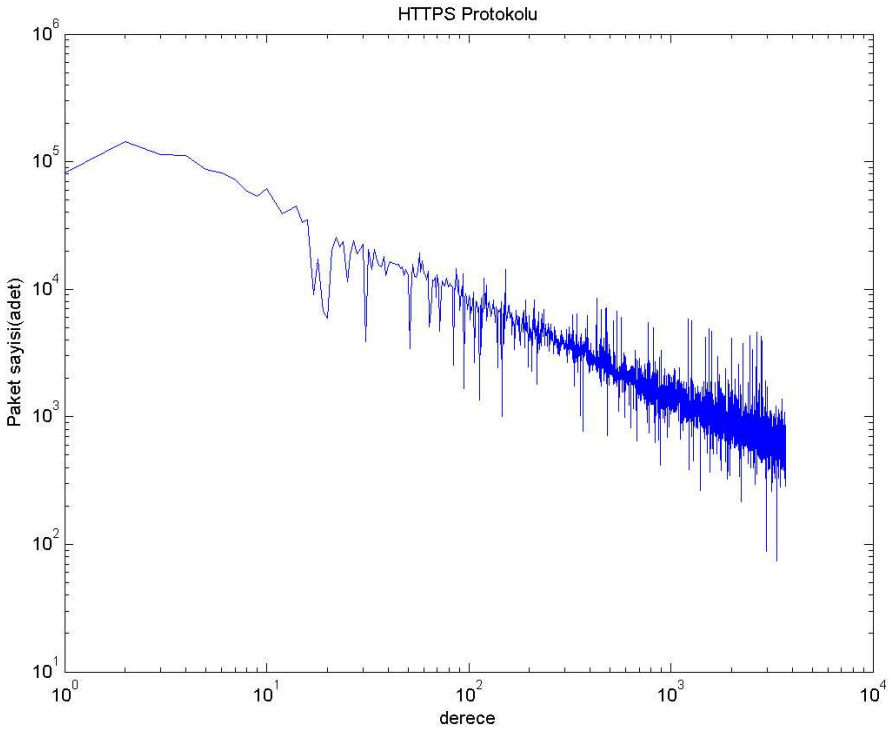
Şekil 3. 14. HTTPS protokolü için veri miktarının dereceye (sıra numarasına) göre değişimi.



Şekil 3. 15. HTTP protokolü için paket miktarının dereceye (sıra numarasına) göre değişimi.



Şekil 3. 16. SMTP protokolü için paket sayısının dereceye (sıra numarasına) göre değişimi.



Şekil 3. 17. HTTPS protokolü için paket sayısının dereceye (sıra numarasına) göre değişimi.

Şekil 3.9-Şekil 3.11 üç protokolün erişim sayısı temel alınarak yarattıkları toplam trafiğe göre oluşturulan grafikler gösterilmektedir. Burada dikkat çekici olan HTTP ve HTTPS protokolünün erişim frekansına göre oluşan grafiklerin hem trafik hem de paket sayısı ile yaklaşık olarak aynı olmasıdır. Bu yüzden sadece trafik miktarını göz önüne almak yeterli olmaktadır.

Tüm protokollerde erişim frekansına göre x ekseni temel alınarak çizilen trafik ve paket ile ilgili grafiklerde ani çıkışlar olduğu (özellikle SMTP) görülmüştür. Bunun nedeni erişim frekansının tek başına, tam anlamıyla trafik ve paket adeti değerlerindeki geçici referans yererliğini adresleyememesidir. Bu yüzden erişim frekansları yanında trafik miktarının da tutulması, daha doğru yük dengeleme kararları vermek için yardımcı olacaktır. Sonuç olarak her üç protokolde de referans yerelliğinin belirgin olduğu ve yük dengelemede bu referans yerelliğinin bize yardım edebileceği tespit edilmiştir.

Hipotezimizi güçlendirmek için toplam trafik içerisindeki %10 tekil ip'nin yarattığı akışların tümüne göre yapılan incelemeler Çizelge 3.5 ve 3.6'da verilmiştir.

Çizelge 3. 5. Tüm trafik içinde %10'luk akış yaratan tekil IP'lerin trafik miktarının toplam trafiğe oranı

Protokol	Toplam Trafik	Tüm trafik içinde %10 tekil IP'lerin yarattığı trafik miktarı	Oran
HTTP	2967553085	2112030093	%71.17
SMTP	177029900952	168288197033	%95
HTTPS	1237592569	679976401	%54.94

Çizelge 3. 6. Tüm paketler içinde %10'luk akış yaratan tekil IP'lerin paket miktarının toplam paket miktarına oranı

Protokol	Toplam Paket	Tüm paketler içinde %10 tekil IP'lerin yarattığı paket sayısı	Oran
HTTP	29788460	17205847	%70.42
SMTP	201920248	189191003	%93.69
HTTPS	13211408	7157909	%54.17

3.4.1. Analizlerden çıkan sonuçların değerlendirilmesi

Trafik analizlerinin incelenmesi sonucu farklı zaman aralıklarında toplanmış tüm protokollerin trafik akışlarında yoğun geçici referans yerelliği tespit edilmiştir. Aynı zamanda erişim frekans grafikleri Eş. 2.9'daki Zipf [28] kanununu destekleyecek şekilde çıkmıştır. Diğer yandan derece (erişim frekansı sırası) temel alınarak çizilen trafik ve paket sayısı grafiklerinde yine Zipf kanununu destekleyecek şekilde dağılım bulunmaktadır. Ancak erişim frekansının daha az olduğu daha büyük derecelerde trafik miktarı yada paket adetleri grafiklerindeki sapmalardan sadece erişim frekansının sistemler üzerindeki yükü adresleyemeyeceği görülmüştür.

Diğer yandan tüm ip'ler içinde en yüksek erişim frekansını oluşturan %10'luk kısmın trafik miktarı ve trafik adetinin Pareto kuralını [40] (%20 lik kısım %80 lik kaynak kullanır) doğrulayan davranışı göze çarpmaktadır.

Ağ trafiklerinde tespit edilen bu davranışlar yardımı ile paralel çalışan yük dengeleme sisteminin temel yapısı oluşturulmuştur.

3.5. Dengeleme Benzetimi

Toplanan internet trafik izleri üzerinde dengeleme benzetimi yaparak yukarıdaki trafik davranışlarının yük dağılımını nasıl etkilediği gösterilecektir.

Benzetim işlemi tüm zaman aralıklarında birer dakika ara ile olmak üzere her üç ağ trafik verisi üzerinde kaynak ip adresleri temel alınarak yapılmıştır. Kaynak ip'lerin temel alınmasının nedeni, 2.5.1 ve 2.5.2 bölümlerinde de anlatılan çalışmalarda referans yerelliğinin kaynak ip'den kaynaklanmasından dolayıdır.

Benzetim Eş. 3.2'nin yardımıyla yapılmıştır. Burada *s* kaynak ip numarası, *m* yük dengeleyici sayısını ve *i* ise yük dengeleyicinin numarasını ifade etmektedir.

$$s \bmod m = \begin{cases} i & Kabul\ et \\ Diger & Red\ et \end{cases} \qquad (3.2)$$

Benzetim yük dengeleyicilerin sayısını üç adet kabul ederek yapılmıştır. Yani m=3 ve i= 0,1,2 şeklinde kabul edilmiştir.

Bu veriler üzerinde yapılan analizlerin grafiksel tablolara dökülmesi içinde Tobi Oetiker'in geliştirdiği bir endüstri standardı kabul edilen rrdtool [88] yazılımı kullanılmıştır.

Veriler VTYS'den aktif-aktif çalışan yük dengeleyicilerin mod temelli çalışma yöntemine benzetecek şekilde alınarak RRD (Round Robin Database) [88] formatındaki dosyalara aktarılmışlardır.

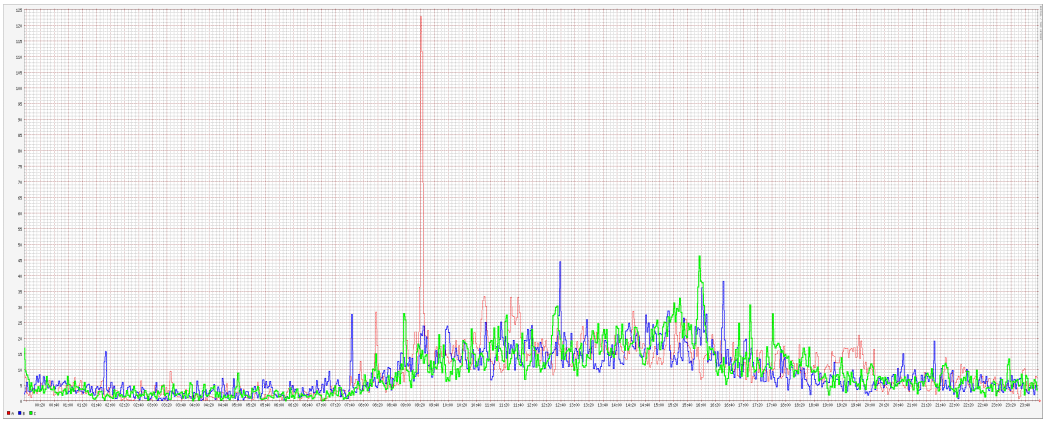
Benzetim yapabilmek için VTYS'deki veriler içerisindeki bazı özel durumları dikkate almak gerekmiştir. Benzetim yapılan verilerin 60 saniyelik aralıklarla rrd

veritabanına aktarılmasından dolayı bir üyelik fonksiyonu modülüne ihtiyaç duyulmuştur. Bu üyelik fonksiyonu PostgreSQL'in kendi doğal dili pgsql ile yazılmıştır.

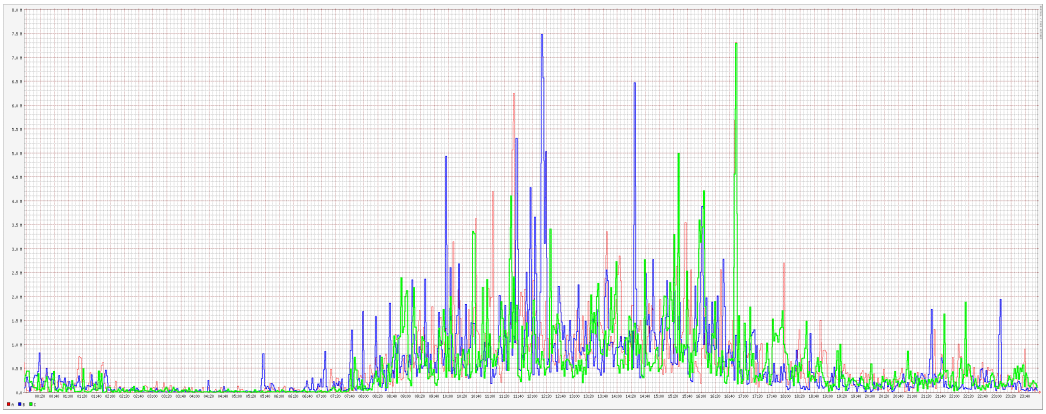
VTYS içindeki akış verisinin başlama zamanı fs, bitiş zamanı ff ve zaman aralığı başlama zamanı ts ve bitiş zamanı da tf olarak kabul edilirse, üyelik fonksiyonu aşağıdaki gibi olmaktadır:

```
if  (ts < ff ) AND ( ff < tf ) then
    if (fs < ts) then
       Return ( ff – ts ) / ( ff – fs )
    Else
       Return 1
    Endif
Elsif    ( fs < tf ) and ( tf <= ff ) then
    if (ts < fs) then
      Return ( tf –fs) / ( ff –fs)
    Else
      Return ( tf – ts) / ( ff – fs)
    Endif
Endif
Return 0
```

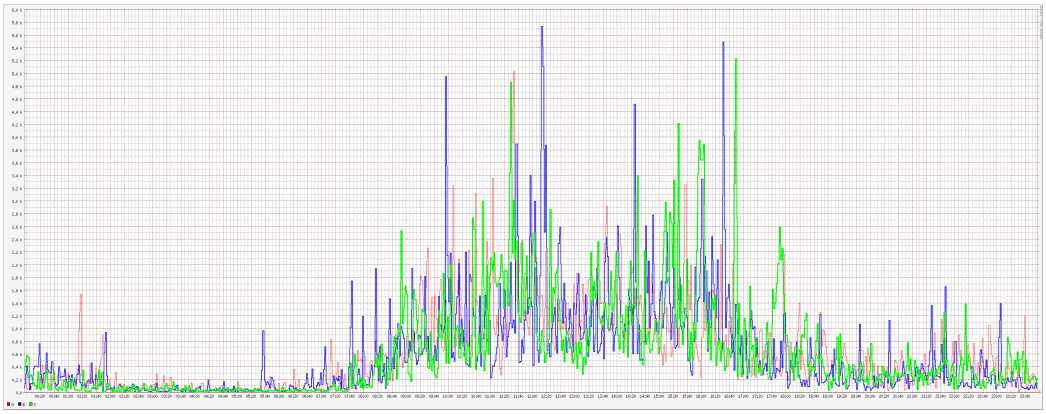
Zaman aralıklarına göre benzetim sonuçları HTTP protokolü için Şekil 3.18'de, SMTP protokolü için Şekil 3.19'da ve HTTPS için de Şekil 3.20'de verilmiştir.



Şekil 3. 18. HTTP Protokolü için benzetim grafiği (Kırmızı: 1.dengeleyici, Mavi: 2.dengeleyici, Yeşil: 3.dengeleyici).



Şekil 3. 19. SMTP protokolü için benzetim grafiği (Kırmızı: 1.dengeleyici, Mavi: 2.dengeleyici, Yeşil: 3.dengeleyici).



Şekil 3. 20. HTTPS protokolü için benzetim grafiği (Kırmızı: 1.dengeleyici, Mavi: 2.dengeleyici, Yeşil: 3.dengeleyici).

3.5.1. Benzetim sonuçlarının analizi

Yapılan benzetim sonrası Şekil 3.18-20'de de görülen grafikler ortaya çıkmıştır. Bu grafikler trafik tutulan tüm süreç için üretilmiştir. Ancak üretilen grafik sayısı çok fazla olduğu için her bir protokol için birer tane örnek verilmiştir. Aşağıda bu grafiklerden elde edilen sonuçlar sıralanacaktır:

- Aktif-Aktif çalışmada matematiksel mod modeline göre dengeleme yapan sistemlerin yük dağılımını düzgün yapamadığı ve bu nedenle bu sistemlerin tam denge durumunda olmadığı görülmüştür.

- Analizi yapılan tüm protokol trafiklerinde (HTTP, SMTP, HTTPS), ip kaynağına (dereceye) göre paket yoğunluğu ve paket sayısının değişimi benzerlik göstermemektedir.

- İstemcilerden gelen trafik İnternet'in doğasından dolayı düzensizdir. Bu trafiğin mod yapısına uygun olarak dengeleyicilere dağıtılması dengesizlik durumunu tam çözememektedir. Bu nedenle dengelemenin başka şekilde çözülmesinin gereklilik olduğu ortaya çıkmıştır.

Yukarıda elde edilen bulgular ışığında matematiksel mod işlemi ile kaynak ip yada port numaralarına göre yapılan dengelemenin sağlıklı bir şekilde yükü dağıtamadığı tespit edilmiştir bu nedenle akış trafiği üzerinde çalışılmaya devam edilmiş, bir sonraki bölümde yeni bir model önerilmiştir.

4. PARALEL YÜK DENGELEME MODELİ

Bu çalışmada, sürekli artan internet trafiğinin servis sunucuları üzerinde oluşturduğu yükü azaltmak/dengelemek/paylaştırmak amacıyla geliştirilen yük dağıtım sistemleri ele alınmış ve paralel yük dengeleme sistemleri içinde yer alan yeni bir model önerilmiştir.

Bu modeli temel alarak çalışan benzetim sistemi tasarlanmış ve sistem aktif ağ yükleri üzerinde çalıştırılarak modelin başarısı ölçülmüştür. Ölçüm sonuçları incelenerek sonuçlar verilmiştir.

Bu bölümde ilk olarak paralel yük dengeleme sistemi mimarisi anlatılmış daha sonra bu sistemi oluşturan parçalar tek tek ele alınarak incelenmiştir.

Daha sonra SMTP ve HTTP protokolleri için benzetim işlemi yapılarak alınan sonuçlar değerlendirilmiştir.

4.1. Paralel Yük Dengeleme Sistem Mimarisi

Trafik verilerinin analizi sonucu, servis sunucularına gelen ağ trafikleri üzerinde referans yerelliğinin olduğu önceki bölümlerde tartışılmıştı. Bu tespit edilen bulgular ışığında, yük dengeleme sistemi kaynak ip'lerin oluşturduğu referans yerelliğini temel alacak şekilde tasarlanmıştır.

Tasarlanan Paralel Yük Dengeleme Sistemi (PYDS) tek makina üzerinde iki farklı dengeleyici tipinde çalışmakta ve Bölüm 2.7.4'de anlatılan paralel yük dengeleme sistemlerinin çalışmasını benzetim yoluyla yerine getirmektedir.

Sistemin, aktif ve pasif çalışma olmak üzere iki çalışma kipi vardır. Aktif çalışma kipinde ağ akış verileri aracı bir uygulama vasıtası ile PYDS'ye aktarılmaktadır. Pasif çalışma kipinde ise ağ akış verileri dosya sisteminden okunarak akış toplama ve istatistik modülüne aktarılmaktadır.

Önerilen paralel yük dengeleme sistemi temel olarak üç modülden meydana gelmektedir. Bunlar:

- Sisteme gelen ağ trafiği içinden, akış verisi toplama ve istatistik tutma yeteneğinde sahip ve aynı isimle anılan modül.

- Akış verilerinin durumunu denetleyen, yük dengelenmesine karar veren ve bu dengelemeyi filtreleme tablosu aracılığıyla yapan yönetim modülü.

- Yönetim uygulaması tarafından filreleme tablosu dinamik olarak değiştirilebilen filtreleme modülü.

Bu üç ana modülün ayrıntıları aşağıda anlatılmıştır.

4.1.1. Akış toplama ve istatistik tutma modülü

Bu çalışmada geliştirilen paralel yük dengeleme sistemi benzetim uygulamasında geçici referans yerelliğini tespit edebilmek için bir akış toplama ve istatistik tutma mekanizmasının olması gereklidir.

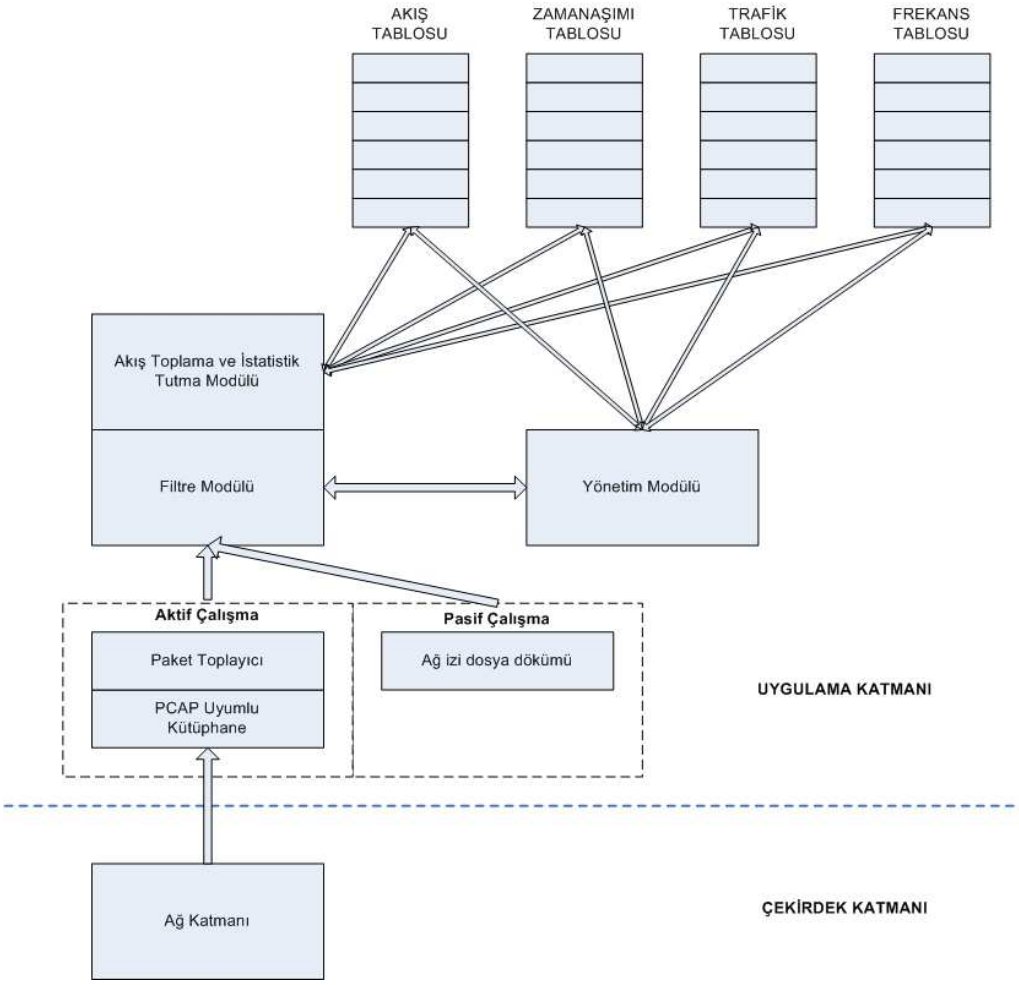
Sistem önceden de bahsedildiği gibi iki kipte çalışabilmektedir. Sistem aktif çalışma kipinde çalışırken, ihtiyacı olduğu akış verilerini bir paket toplama uygulaması vasıtası ile almaktadır.

Paket tutma uygulaması için Linux işletim sisteminde kullanılabilecek iki yöntem bulunmaktadır. Birinci yöntem, çekirdek seviyesinde paket toplama işlemini yapmak, ikinci yöntem ise kullanıcı seviyesinde çalışan bir uygulama ile bu işlemi yapmaktır.

Çekirdek seviyesinde programlama, hem zor, hem de sorun çıkması durumunda tüm sistemi etkileyebileceği için çok tercih edilen bir yöntem değildir [89]. Bunun yerine, kullanıcı seviyesinde çalışan bir uygulama geliştirilmesi daha uygun

olmaktadır. Ancak kullanıcı seviyesinde paket toplama işleminde, Deri'nin [90] de işaret ettiği gibi, PCAP [80] kütüphanesi kullanan uygulamaların, yoğun trafik yükü altında, işletim sistemi üzerinde fazladan yük oluşturduğu ve bu yüzden kullanıcı seviyesine geçen paketlerde de kayıplar oluştuğu bilinmektedir.

Deri'nin bu konu üzerinde yaptığı çalışmalarda, PF_RING [90] ve nCap [91] adında PCAP kütüphanesi ile uyumlu, iki tane uygulama arabirimi geliştirilmiştir. Bu uygulama arabirimleri sayesinde, ağ katmanına gelen tüm paketler performans kaybı olmadan işlenebilmektedir. Bu yüzden, paket toplama uygulamasında Deri'nin uygulama arabirimleri kullanılmıştır.



Şekil 4. 1. Paralel Yük Dengeme Sisteminin genel yapısının gösterimi.

Pasif kipte çalışmada ise, doğrudan ağ izleri dökümlerinden, akış verileri alınarak paket tutma ve istatistik modülüne aktarım yapılır.

Şekil 4.1'de PYDS içinde, paket toplama ve istatistik uygulamasının yeri ve diğer bölümler ile ilişkisi gösterilmektedir.

Sistemin çalışma yapısını ve tabloların nasıl kullanıldığı aşağıda anlatılmıştır.

4.1.1.1. Sistemin çalışması ve hafıza tablolarının kullanılması

Paket toplama ve istatistik tutma modülünde, akış verilerinin özelliklerine göre sıralanması, bu verilerin daha kolay işlenmesi için önemlidir. Bunu sağlamak için, dört ayrı hafıza tablosu kullanılmıştır.

Hafıza tabloları içerisinde, yüksek hızlı arama ve veri güncellemelerin yapılması gerekmektedir. Bu yüzden, bu tablolarda Red-Black Tree (Kırmızı-Siyah Ağaç) [92] hafıza yapıları kullanılmıştır.

Kırmızı-Siyah Ağaç yapısında, veriye erişim tek veya çift bağlı dizilere göre hızlı olmaktadır. Bu hız ise, aşağıdaki eşitlik ile (Big O notation) [93] ifade edilebilir.

$$b = O (\log (n)) \quad (4.1)$$

Burada;
n : Kırmızı-Siyah Ağaç yapısındaki veri miktarıdır.
b: Erişim basamak derecesi olmaktadır.

Modül içinde yer alan hafıza tablolarının yapıları, aşağıdaki şekilde sıralanabilir:

1. Akış verisi, Bölüm 2.6 akış seviyesinde trafik tanımlama bölümünde anlatıldığı gibi {protokol, kaynak adres, kaynak port, hedef adres, hedef port} alanlarının ve ek olarak akış verisinin sisteme dahil olduğu süreyi, içerecek yapıdadır. Veriler, kaynak

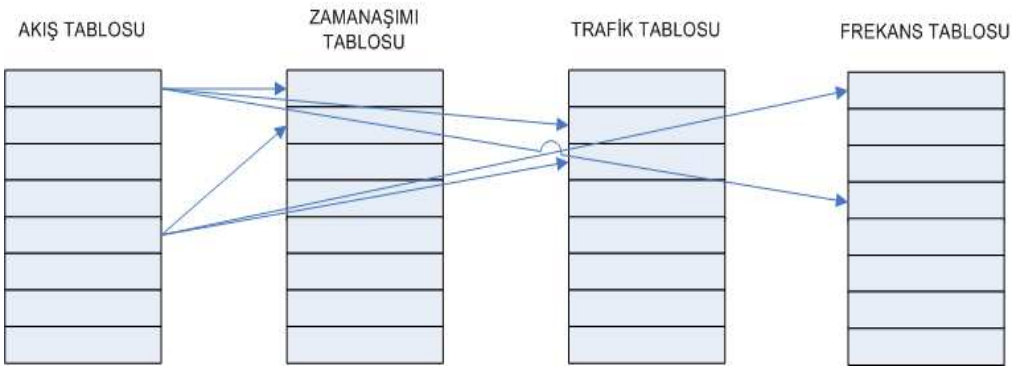
ip numarasına göre sıralanırlar. Ayrıca bu tabloyu, diğer üç tabloya ilişkilendirmek için göstericiler bulunmaktadır Akış tablosu (AKISLAR).

2. Akış verisinin son güncelleme süresini tutan ve AKISLAR tablosu ile bağlantısı olan zaman aşımı tablosu (AKIS_ZA).

3. Akış verisinin trafik miktarını Byte cinsinden tutan ve AKISLAR tablosu ile bağlantısı olan trafik tablosu (AKIS_TRAFIK).

4. Akış verisinin yeniden sisteme girme olasılığını tutan ve AKISLAR tablosu ile bağlantısı olan frekans tablosu (AKIS_FREKANS).

Bu tabloların birbirleri ile ilişkileri Şekil 4.2'de gösterilmiştir.



Şekil 4. 2. Akış toplama ve istatistik tutma modülünde hafıza tablolarının birbirleri ile ilişkileri.

Modülün çalışması, Şekil 4.3'deki akış şemasında'da gösterilmiştir. Çalışma basamakları, maddeler halinde aşağıdaki gibi sıralanabilir:

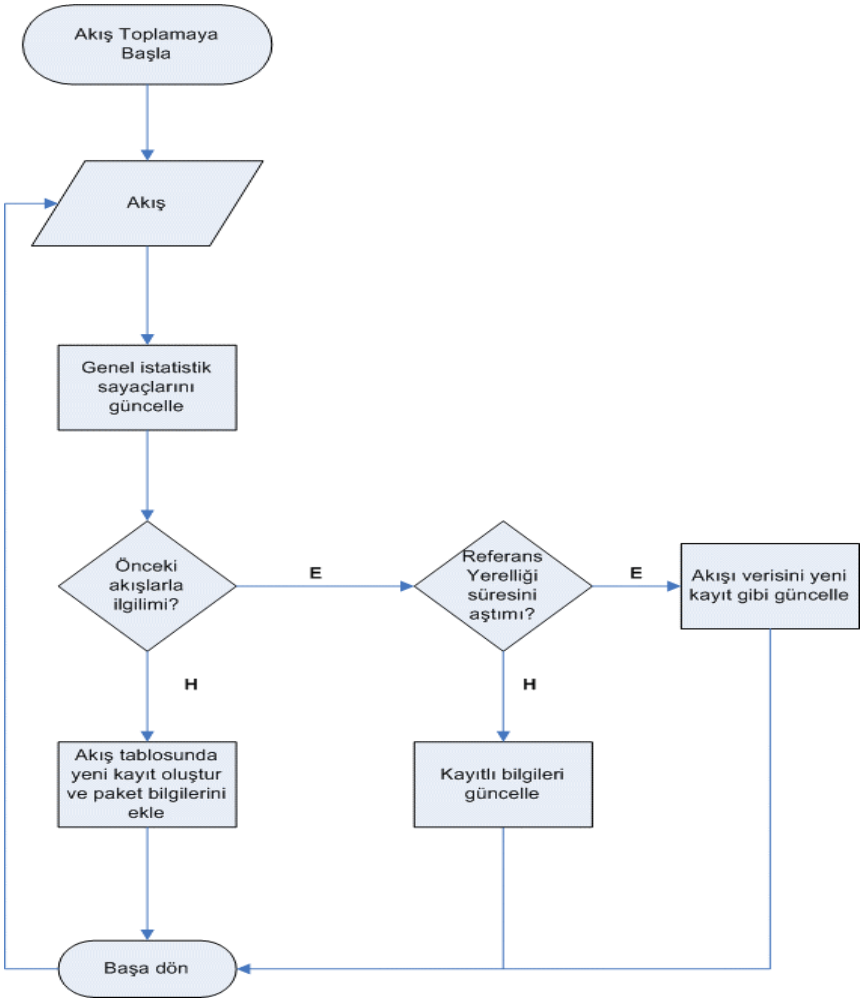
1. İnternet'ten gelen akış verisi modüle ulaşır.

2. Genel istatistik sayaçları güncellenir.

3. Akış verisinin akış tablosu içinde önceden kaydı olup olmadığına bakılır. Eğer önceden kayıt varsa ve sistem yöneticisi tarafından belirlenmiş geçici referans zamanı aşılmamışsa kayıt bilgileri güncellenir. Aksi takdirde akış verisi yeniymiş gibi değerlendirilir.

4. Akış verisi, akış tablosunda yer almıyorsa ilgili bilgiler akış tablosuna ve diğer tablolara yeni kayıt olarak eklenir.

5. İşlem biter başa dönülür.



Şekil 4. 3. Akış toplama ve istatistik tutma modülünün akış şeması ile gösterimi.

4.1.2. Yönetim modülü

PYDS içerisindeki en önemli görevi yönetim modülü yapmaktadır. Bu modül, hem paket toplama ve istatistik modülü, hem de filtreleme modülü hafıza tabloları ile ilişki içerisindedir. Modül, dengesizlik durumunu, hafıza tablolarında tutulan akış kayıtlarından yararlanarak ve filtreleme tablosu içindeki kayıtları değiştirerek gidermeye çalışmaktadır.

Bu modülün uygulama akış diagramı Şekil 4.4'de verilmiştir.

PYDS benzetim uygulaması, çalıştırıldıktan sonra ilk olarak parametre dosyasından yönetici tarafından değiştirilebilen parametreleri okur, sistem yöneticisi tarafından önceden belirlenen süre boyunca kendini beklemeye alır. Bu süre, benzetim denemelerinde edinilen tecrübe ışığında 300 sn olarak belirlenmiştir (karar aralığı).

Bekleme süresi sona erdikten sonra, modül önce çöp toplama (garbage collection) işlemi başlatır. Bu işlem sonucu, akış kaydının son erişim süresi, sistem yöneticisi tarafından önceden belirlenmiş zamanaşımı süresini aşmış olanlar, tüm hafıza tablolarından silinirler. Silme işlemi sırasında, tüm akış kayıtları incelendiği için, kayıtların silinenler dışında kalanların istatistikleri, sonradan denge karar mekanizmasında kullanılmak üzere geçici bir hafıza tablosuna aktarılır.

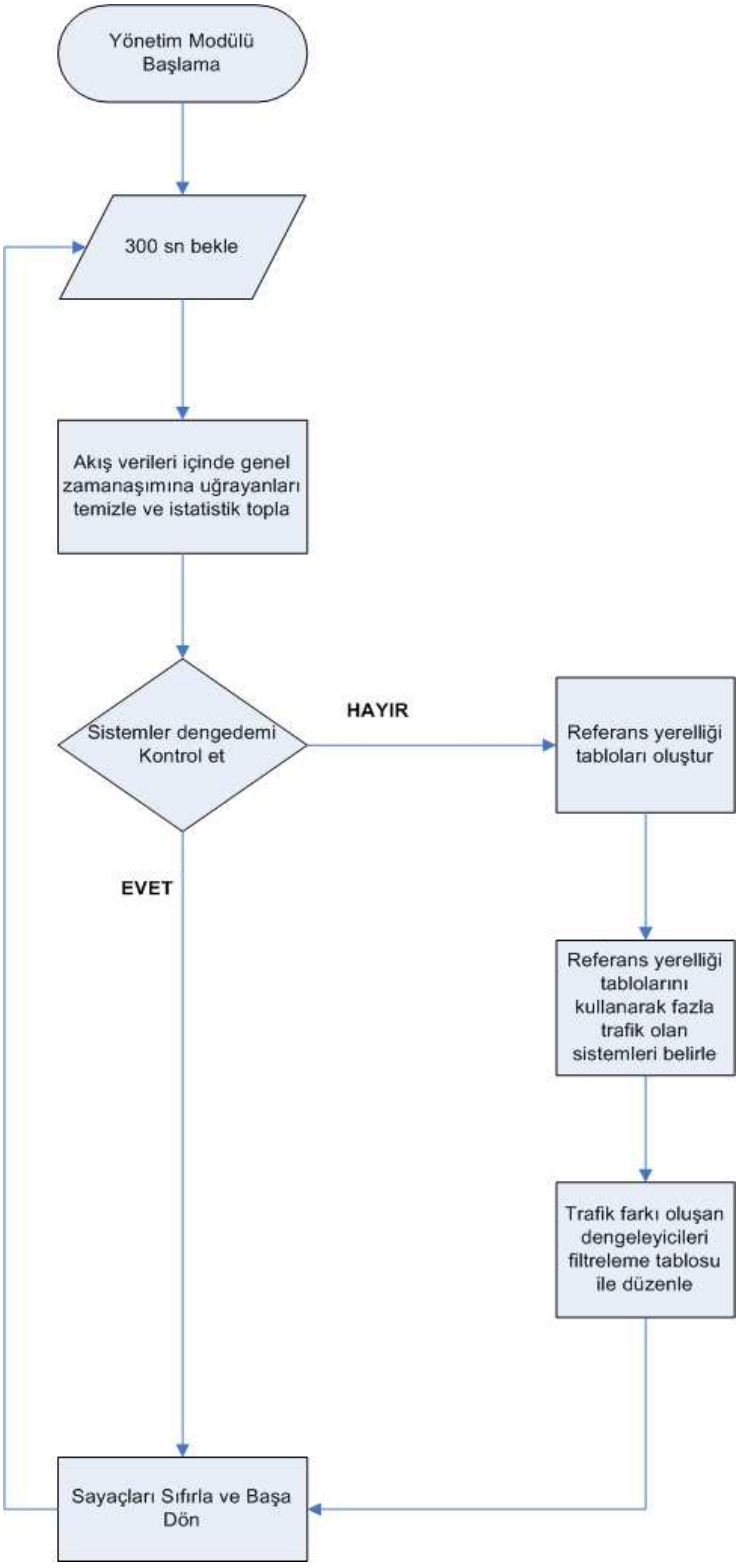
Paralel çalışan *n* kadar dengeleyicinin herbiri için ayrı bir trafik sayacı bulunmaktadır. Bu sayaçların yardımı ile her bir sisteme düşmesi gereken ortalama trafik Eş. 4.2 yardımı ile bulunur (karar anında).

$$\Delta T = \frac{\sum_{i=1}^{n} T_i}{n} \tag{4.2}$$

T_i : i. Dengeleyici üzerinden geçen toplam trafik

n : Dengeleyici sayısı,

ΔT : Ortalama trafik miktarıdır.



Şekil 4. 4. Yönetim modülünün program akış şeması.

Dengeleyicilerin her birinin trafik ortalamasına göre trafik farkları, Eş. 4.3 kullanılarak bulunur (karar anında).

$$F_i = T_i - \Delta T \qquad (0 < i \leq N) \tag{4.3}$$

F_i : i. dengeleyici'nin ortalama trafik ile farkı.
T_i : i. dengeleyicinin üzerinden geçen trafik miktarı.
ΔT : Ortalama trafik miktarı.
N : Dengeleyici adeti.

Farkın pozitif olması, dengeleyici üzerinde ortalamaya göre daha fazla yük olduğunu, negatif olması halinde ise, dengeleyici üzerindeki yük miktarının ortalamadan daha az olduğunu gösterir.

Eğer dengeleyiciler üzerindeki bu fark, sistem yöneticisi tarafından önceden belirlenmiş bir oranı geçerse, o zaman sistem içindeki yük dengeleyiciler arasında, dengesizlik durumu olduğu ortaya çıkar. Böyle bir durumda, yönetim modülü tarafından dengeleme işlemine geçilir.

4.1.2.1. Denge karar sistemi

Dengeleyiciler arasında, trafik yükünde dengesizlik tespit edildiği durumda, dengeleme karar sistemi devreye girer. Bu sistem, Bölüm 3'de analiz sonuçlarında elde edilen bulgular göz önüne alınarak tasarlanmıştır ve dengeleme işlemini yerine getirir.

Bölüm 2.4.3'deki çalışma kümesi modelinde anlatıldığı gibi, Eş. 2.1'den yararlanarak, önceden belirlenmiş referans yerelliği süresi "τ" yardımı ile (τ=1 saat alınmıştır), referans yerelliği hafıza tablosu oluşturulur. Bu tablo oluşturulurken, bu sürenin yanısıra, ortalama trafik ve ortalama erişim frekansı değerleride göz önüne alınır. Aşağıda bu tablonun nasıl oluşturulduğu anlatılmıştır.

Akış çalışma kümesinin, $W(t,\tau)$ şeklinde ve çalışma kümesindeki her bir akışın frekans adeti F_i, trafik miktarı T_i, çalışma kümesinin ortalama erişim frekansı ΔAF_i, ortalama trafik miktarı ΔAT_i, tüm akışlar içerisinde en büyük erişim frekansının EF_i ve en büyük trafik miktarının ET_i ile tanımlandığı kabul edilmiştir.

Akış tablosunda, i. dengeleyiciye ait veriler üzerinden,

$$\Delta AF_i = \frac{\sum_{j=1}^{m} F_j}{m} \tag{4.4}$$

Burada, F_j: i. dengeleyiciye ait j. akış frekansı, m: toplam akış sayısı, ΔAF_i : i. dengeleyicinin ortalama erişim frekansıdır.

i. dengeleyiciye ait veriler üzerinden,

$$\Delta AT_i = \frac{\sum_{j=1}^{m} T_j}{m} \tag{4.5}$$

Burada, T_j: i. dengeleyiciye ait j. akış trafiği, m: toplam akış sayısı, ΔAT_i : i. dengeleyicinin ortalama trafik miktarıdır.

Akış çalışma kümesindeki akışlardan, ΔAF_i ve ΔAT_i değerlerine eşit yada büyük olanlardan referans yerelliği tablosu oluşturulur. Oluşturulan referans yerelliği tablosundaki her bir akış için, denge katsayısı değeri Eş. 4.4'de verildiği gibi hesaplanır.

$$K_{ij} = (F_j/EF_i) \text{ x frekans\_katsayısı} + (T_j/ET_i) \text{ x trafik\_katsayısı} \tag{4.6}$$

Burada, K_{ij}: i. dengeleyicinin j. akışına ait denge katsayısıdır. frekans_katsayısı ve trafik_katsayısı parametriktir (frekans_katsayısı=0,9; trafik_katsayısı=0,1 alınmıştır, %90 frekans, %10 trafik olarak düşünülmüştür).

Referans yerellik tablosu, denge katsayısı değerlerine göre büyükten küçüğe doğru sıralanır. Bu sıralama, denge karar mekanizmasının en uygun değeri bulabilmesi için gereklidir. Dengeleyiciler arasında aktarılacak kayıtlar denge katsayı sırasına göre büyükten küçüğe doğru alınır.

Eş. 4.3'de hesaplanan F_i değeri pozitif olan dengeleyicinin, referans yerellik tablosunun kayıtları arasından, yukarıdaki sıralamaya göre toplamı fark trafiğinden elde edilen kadar olanı seçilir ve alınır.

Akış trafik farkı aktarım uygulaması ile bu işlemin nasıl yapıldığı aşağıda gösterilmiştir.

```
if  ( Fi > 0 )  then
  i = 0; toplam_trafik=0; P::Gosterici;
    While ( ri != NULL ) {
       if ( toplam_trafik >= Fi )
                break;
       if  ( ri.trafik  < Fi ) {
                toplam_trafik =  toplam_trafik  + ri.trafik;
                P = ri; P++;
       }
    }
```

Daha sonra seçilen bu kayıt/kayıtlar, F_i değeri negatif olan dengeleyicilere, filtre tablosu üzerinde değişiklik yapılarak aktarılır. Bu işlem daha ayrıntılı şekilde aşağıda anlatılmaktadır.

Her bir dengeleyici için, A, akış verilerinin ip adreslerini içeren ve her bir elemanı tekil ve a_j şeklinde gösterilen bir küme olsun. A kümesinin büyüklüğü m=|A| şeklinde kabul edilsin. Bu küme Eş. 4.7'de gösterilmektedir.

$$A = \{ a_0, a_1, ..., a_j \} \qquad 0 \leq j < m \tag{4.7}$$

Diğer yandan dengeleyicileri ifade eden küme D, N ise dengeleyici makina adeti ve bu kümenin her bir elemanı d_i ile ifade edilirse, bu küme Eş. 4.8'deki gibi gösterilebilir.

$$D = \{ d_0, ..., d_i \} \qquad 0 \leq i < N \tag{4.8}$$

Filtreleme tablosuna yerleştirme ise, Çizelge 4.1'de gösterilen örnekteki gibi olacaktır. Filtreleme tablosunun gelen trafiği nasıl değerlendireceği bir sonraki bölümde anlatılmıştır.

Çizelge 4. 1. Filtreleme tablosu örneği.

Kaynak ip	Hedef Yük Dengeleyici
a_1	d_0
a_5	d_1

Çizelge 4.1'de verilen örnekte a_1 ip'sinden gelen akışlar, d_0 dengeleyicisine ve a_5 ip'sinden gelen akışlar, d_1 dengeleyicisine atanmıştır.

4.1.3. Filtreleme modülü

Filtreleme modülü PYDS'de, yönetim modülü ile ortak olarak çalışan ve gelen veri paketinin hangi yük dengeleyiciyi ilgilendirdiğine karar veren kısımdır. Şekil 4.5'de filtreleme modülünün çalışması akış şeması olarak verilmiştir.

Bu modül iki katmanlı olarak çalışmaktadır. Birinci katmanda, filtreleme tablosu yer almakta, ikinci katmanda ise standart matematiksel mod işlemi ile yük dağılımına

karar verilmektedir. Filtreleme tablosu, matematiksel mod işlemine göre öncelik taşımaktadır.

Paralel yük dengeleme benzetim uygulamasına gelen akış verisinin kaynak ip'si, önce filtreleme tablosundaki değerlerle karşılaştırılır. Eğer yönetim modülü filtre tablosuna önceden kaynak ip ile ilgili kayıt eklemiş ise, bu ip'den gelen tüm paketler atanmış dengeleyici ile ilişkilendirilir ve bu dengeleyiciye yönlendirilir. Bu tablonun örnek yapısı Çizelge 4.1'de verilmiştir.

Eğer filtreleme tablosunda, akış verisinin kaynak ip'sine ait bir kayıt bulunamamış ise bu durumda matematiksel mod işlemi sonucu elde edilen değer kullanılarak bu ip'den gelen tüm paketler ilgili dengeleyici ile ilişkilendirilir.

Şekil 4. 5. Paket filtreleme modülü akış şeması.

4.2. HTTP ve SMTP Trafik İzleri Üzerinde PYDS ile Benzetim

Geliştirilen paralel yük dengeleme sisteminin başarısını görebilmek için, Gazi Üniversitesi web ve e-posta sunucularına gelen istemci trafikleri toplanmıştır. Çizelge 4.2'de HTTP (web) protokolü ve Çizelge 4.3'de ise SMTP (e-posta) protokolü için toplanan akış trafik ayrıntıları verilmiştir.

Çizelge 4. 2. HTTP protokolüne ait akış trafik ayrıntsı.

Akış başlama zamanı	11:06 20/11/2006
Akış bitiş zamanı	18:05 24/11/2006
Akış adeti	9017704
Geçen veri miktarı	8428999644 Byte

Çizelge 4. 3. SMTP protokolüne ait akış trafik ayrıntısı.

Akış başlama zamanı	22:40 27/11/2006
Akış bitiş zamanı	15:15 04/12/2006
Akış adeti	751660
Geçen veri miktarı	19460667816 Byte

PYDS sistemi, pasif kipte önceden toplanan akış verileri kullanılarak çalıştırılmıştır. HTTP ve SMTP protokolleri için 2 ve 3 makinanın paralel benzetimi yapılmıştır. Benzetim sonuçlarının grafiksel gösterimi aşağıda daha ayrıntılı anlatılmıştır.

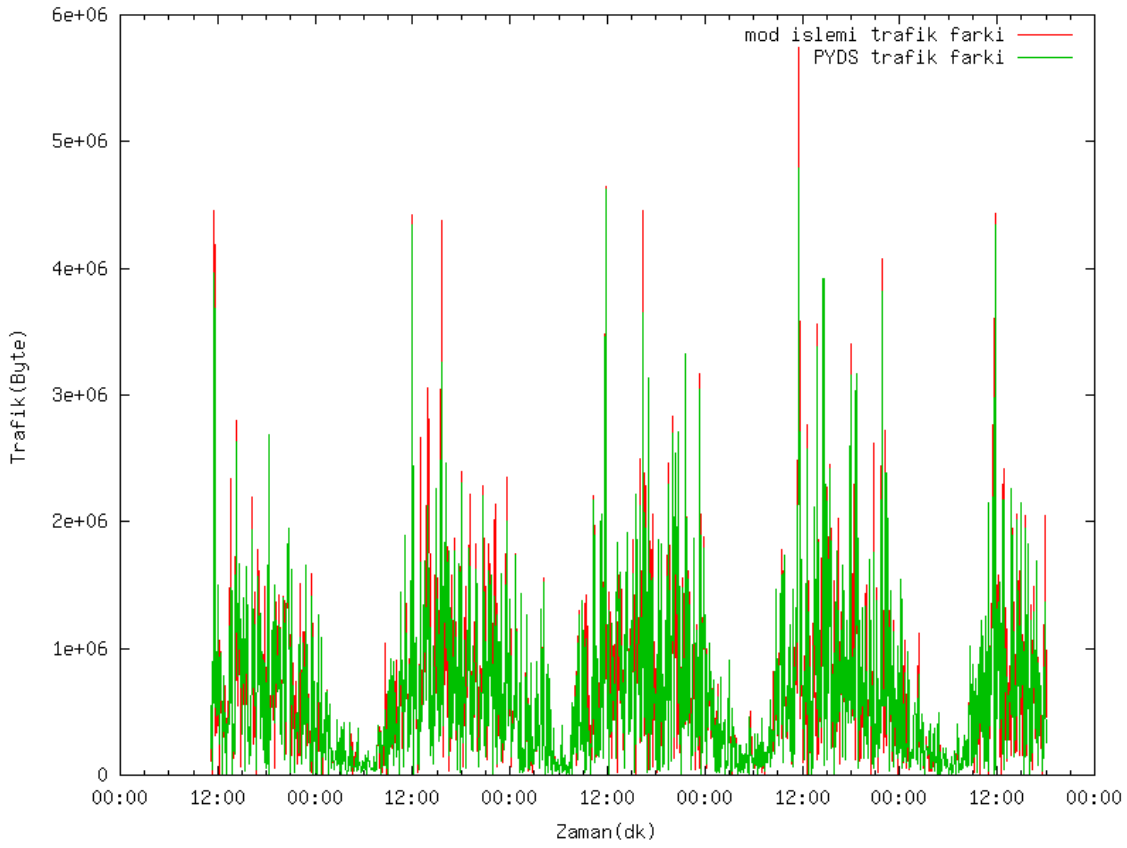
Üretilen grafikler, sistemdeki makinaların beşer dakikalık arayla üzerlerine düşen trafik yüklerinin, ortalama sistem trafiği ile farkının mutlak değerinin toplamıdır. Bu işlem Eş. 4.9'deki gibi gösterilmiştir.

ΔT'nin ortalama trafik, T_i i. dengeleyici üzerine düşen trafik, n'i dengeleyici sayısı ve F'de fark olarak kabul edilirse;

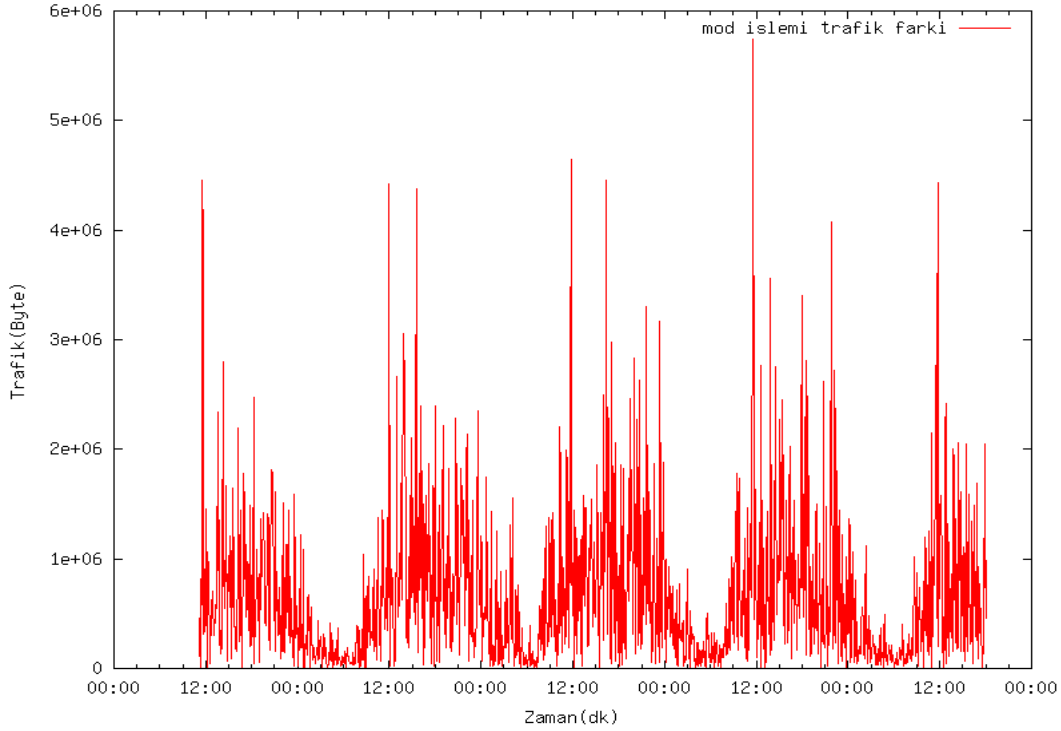
$$\Delta T = \frac{1}{n}\sum_{i=1}^{n} T_i \qquad F = \sum_{i=1}^{n} |T_i - \Delta T| \tag{4.9}$$

4.2.1. HTTP ve SMTP protokolü benzetim grafikleri

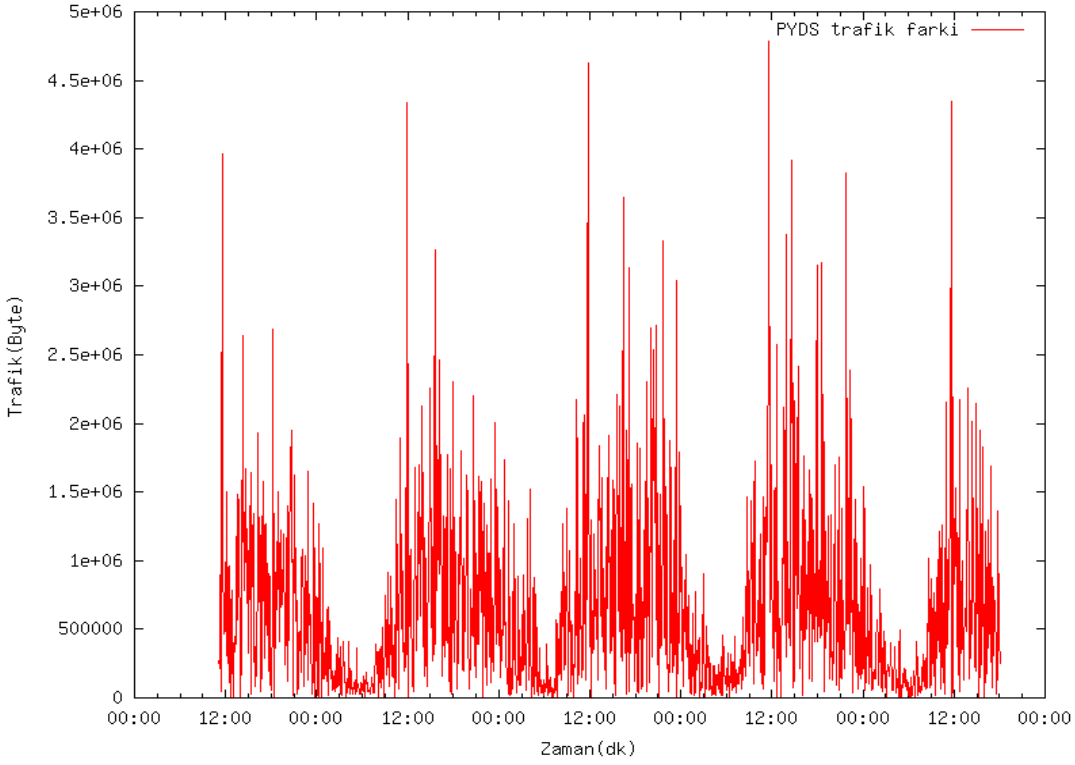
HTTP protokolü akış trafiği için PYDS sistemine paralel iki makina benzetimi yaptırılmıştır. Aynı akış verilerinin matematiksel mod işlemi ve PYDS ile işlenmesi sonucu oluşan fark grafiklerinin birlikte olduğu grafik Şekil 4.6'da verilmiştir.



Şekil 4. 6. HTTP protokolü için iki makina ile yapılan benzetimin mod ve PYDS trafik fark grafiklerinin birlikte gösterimi.



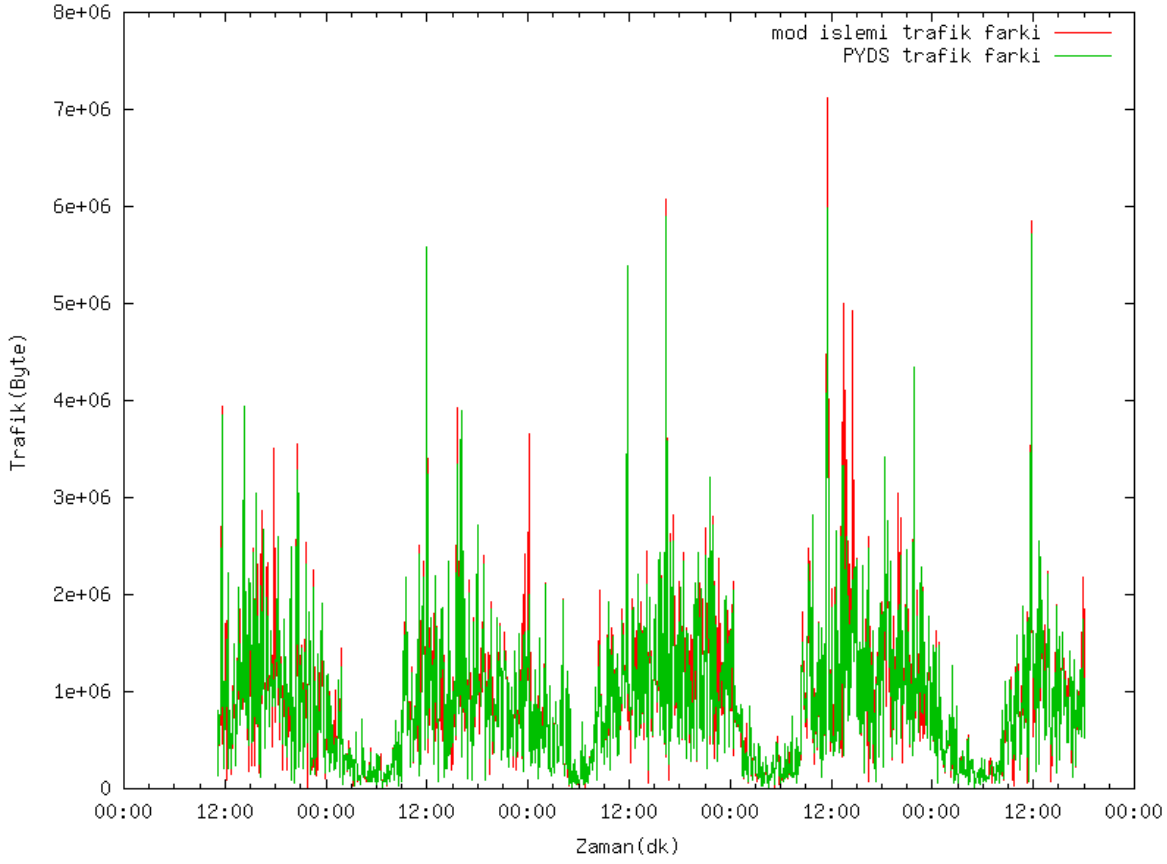
Şekil 4. 7. HTTP protokolü için iki makina ile yapılan benzetimin mod fark grafiği.



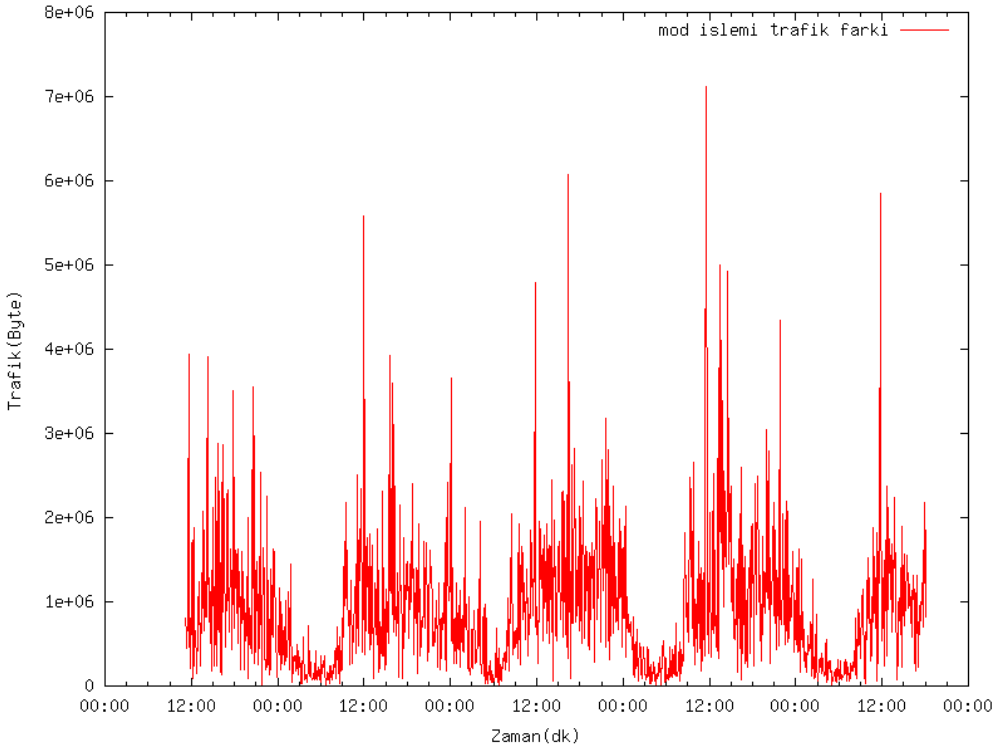
Şekil 4. 8. HTTP protokolü için iki makina ile yapılan benzetimin PYDS fark grafiği.

Şekil 4.7 ve Şekil 4.8'de ise HTTP protokol verileri kullanılarak, PYDS'nin iki dengeleyici benzetimi ve sadece matematiksel mod işlemi sonucu oluşan grafikler ayrı ayrı gösterilmiştir. Bu iki grafiğin birleşimi Şekil 4.6'daki grafiği meydana getirmektedir.

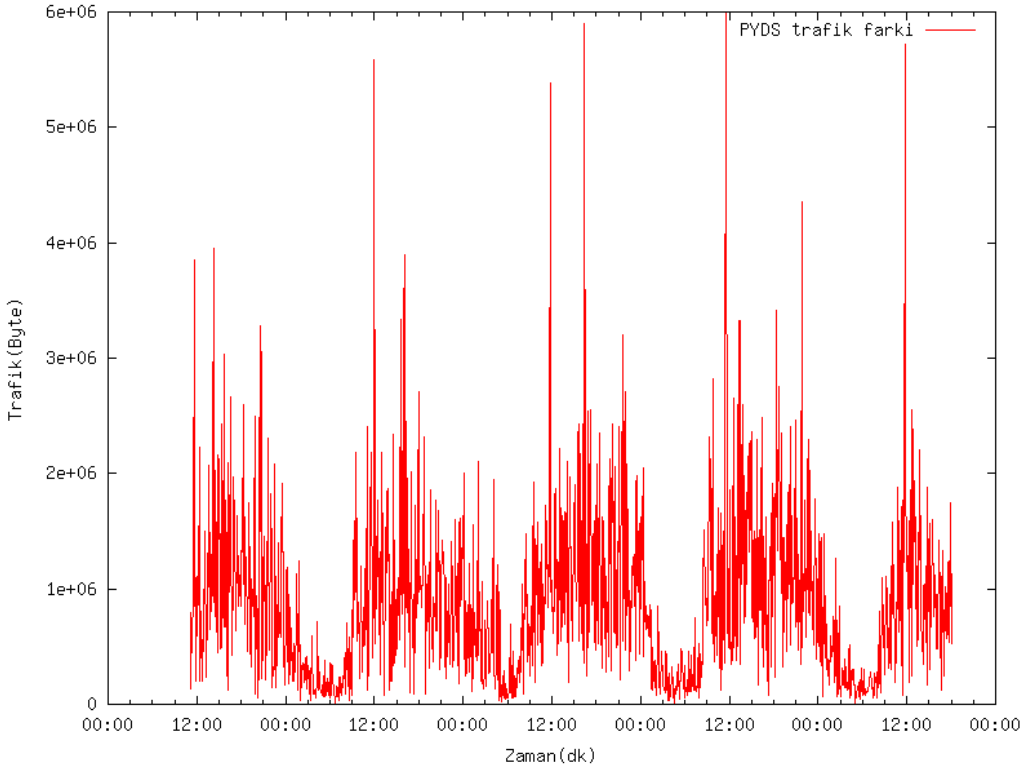
Aynı şekilde HTTP protokolü akış trafiği için PYDS paralel üç makina benzetimi yapılmıştır. Aynı akış verilerinin matematiksel mod işlemi ve PYDS ile işlenmesi sonucu oluşan fark grafiklerinin birlikte olduğu grafik Şekil 4.9'da verilmiştir.



Şekil 4. 9. HTTP protokolü için üç makina ile yapılan benzetimin mod ve PYDS trafik fark grafiklerinin birlikte gösterimi.



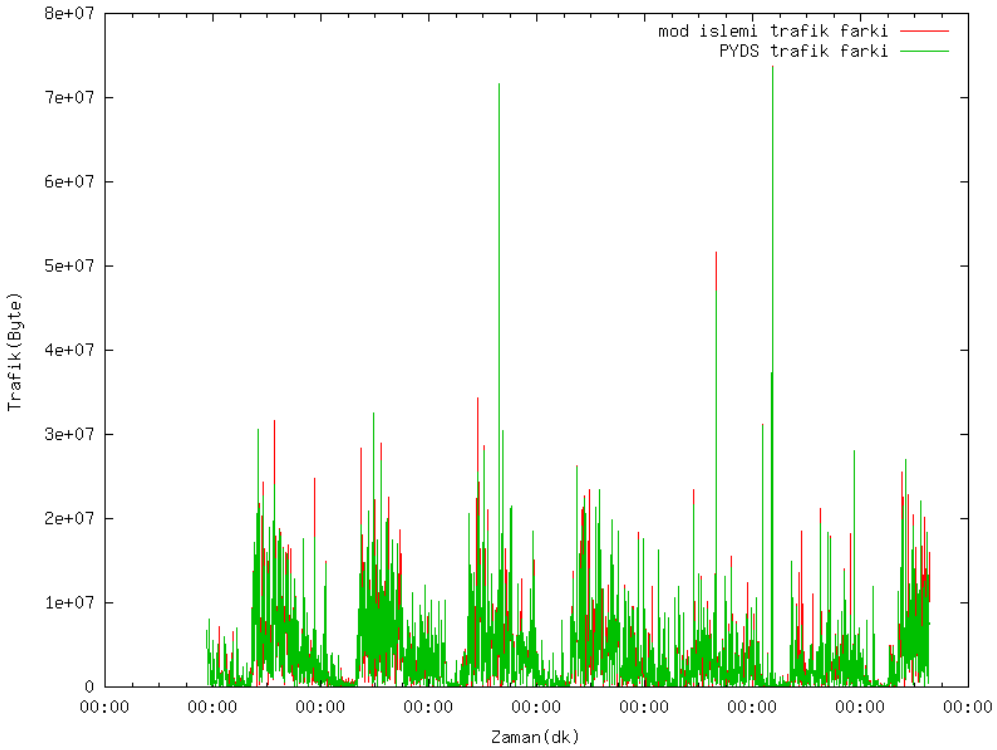
Şekil 4. 10. HTTP protokolü için üç makina ile yapılan benzetimin mod fark grafiği.



Şekil 4. 11. HTTP protokolü için üç makina ile yapılan benzetimin PYDS fark grafiği.

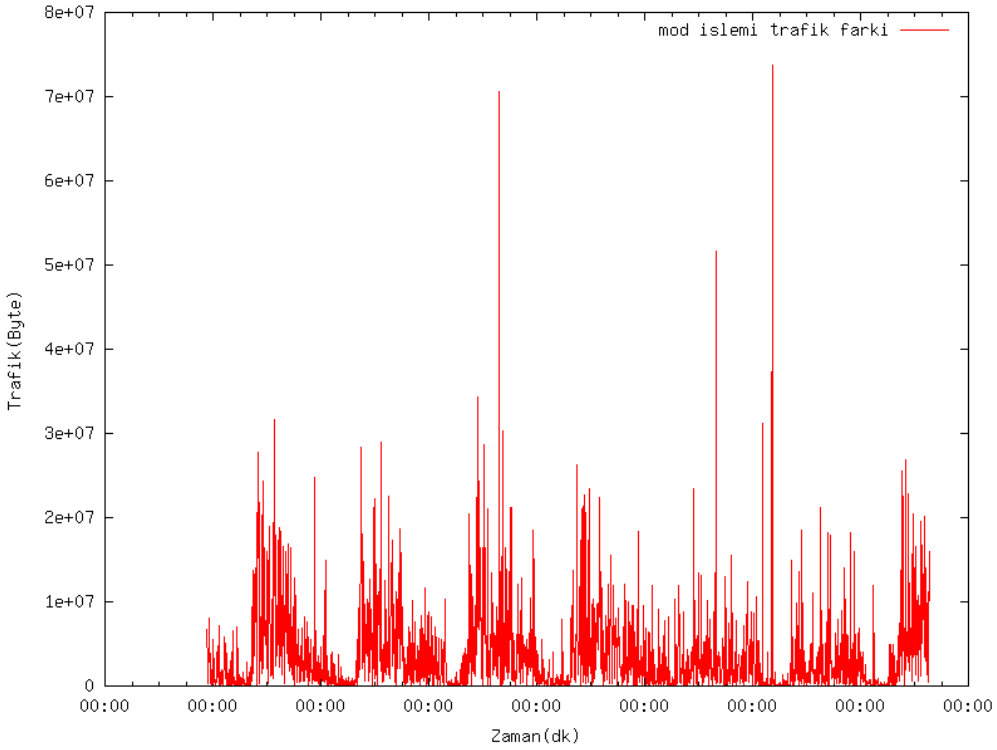
Şekil 4.10 ve Şekil 4.11'de ise HTTP protokol verileri kullanılarak, PYDS'nin üç dengeleyici benzetimi ve sadece matematiksel mod işlemi sonucu oluşan grafikler ayrı ayrı gösterilmiştir. Bu iki grafiğin birleşimi Şekil 4.9'daki grafiği meydana getirmektedir.

SMTP protokolü akış trafiği için PYDS'e paralel iki makina benzetimi yapılmıştır. Aynı akış verilerinin matematiksel mod işlemi ve PYDS ile işlenmesi sonucu oluşan fark grafiklerinin birlikte olduğu grafik Şekil 4.12'de verilmiştir.

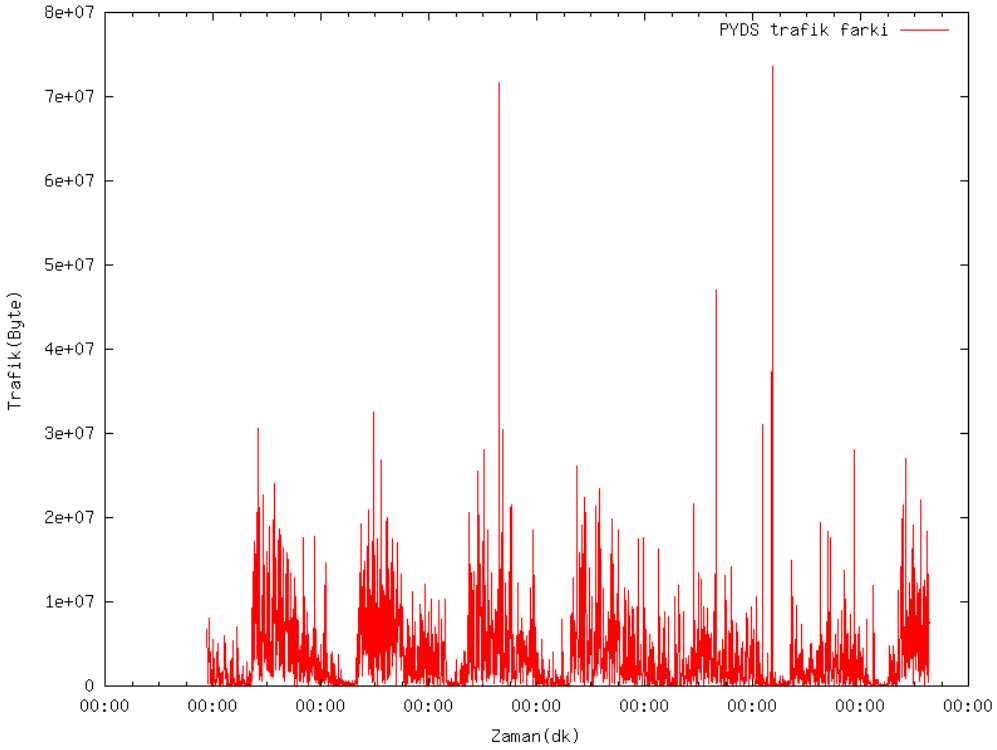


Şekil 4. 12. SMTP protokolü için iki makina ile yapılan benzetimin mod ve PYDS trafik fark grafiklerinin birlikte gösterimi.

Şekil 4.13 ve Şekil 4.14'de ise SMTP protokol verileri kullanılarak, PYDS'nin iki dengeleyici benzetimi ve sadece matematiksel mod işlemi sonucu oluşan grafikler ayrı ayrı gösterilmiştir. Bu iki grafiğin birleşimi Şekil 4.12'daki grafiği meydana getirmektedir.

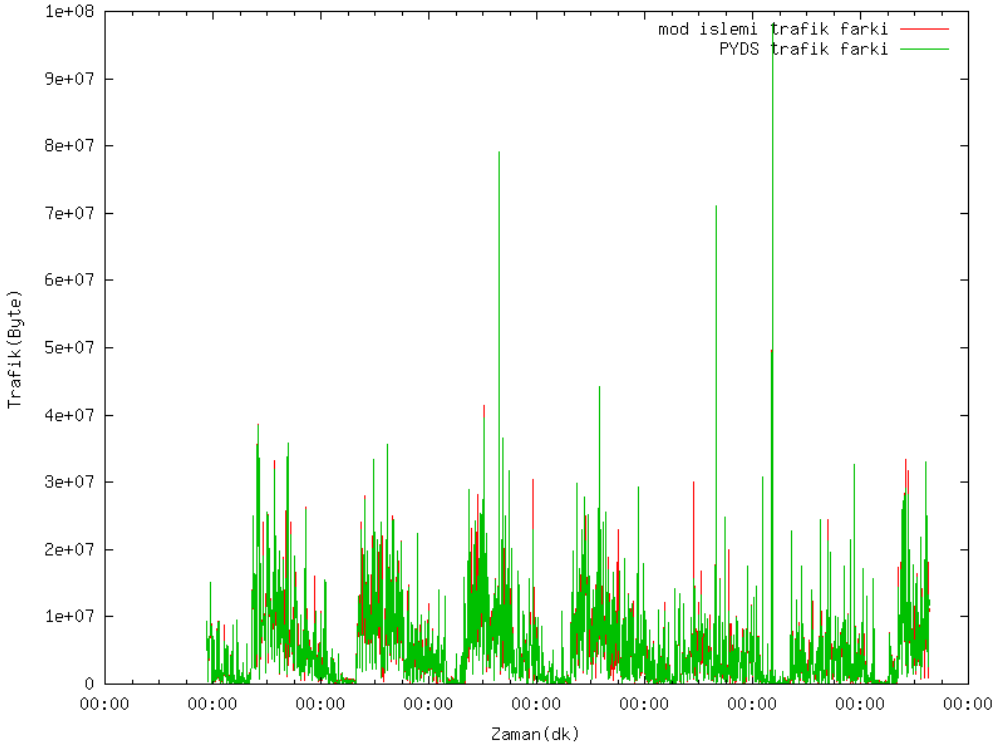


Şekil 4. 13. SMTP protokolü için iki makina ile yapılan benzetimin mod fark grafiği.



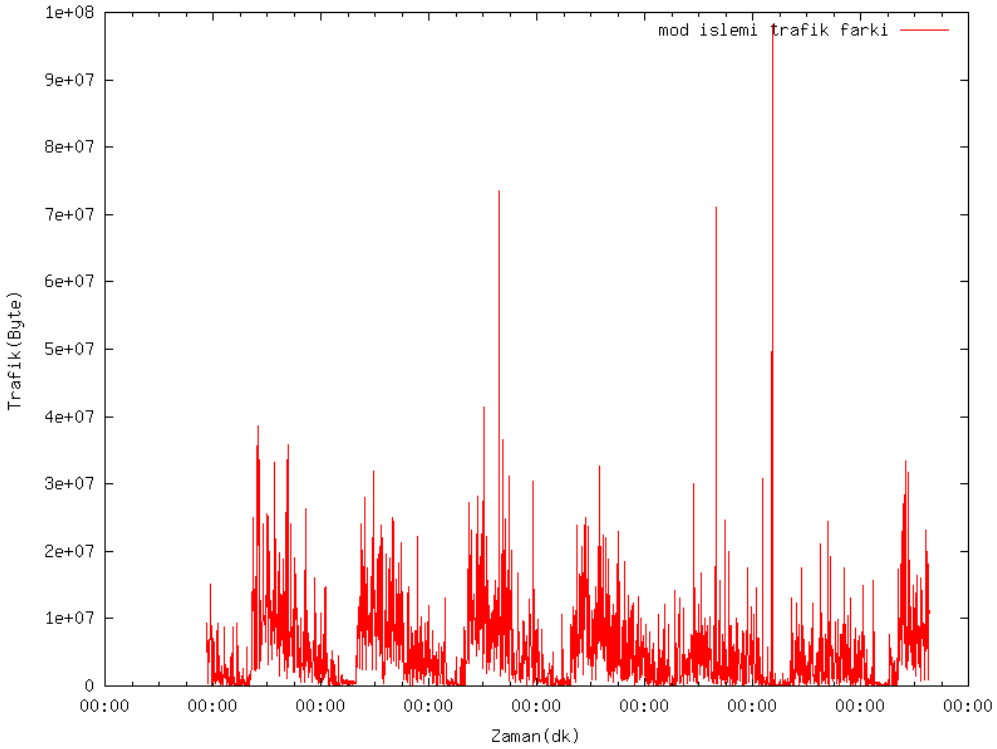
Şekil 4. 14. SMTP protokolü için iki makina ile yapılan benzetimin PYDS fark grafiği.

SMTP protokolü akış trafiği için PYDS'e üç makina benzetimi yaptırılmıştır. Aynı akış verilerinin matematiksel mod işlemi ve PYDS ile işlenmesi sonucu oluşan fark grafiklerinin birlikte olduğu grafik Şekil 4.15'de verilmiştir.

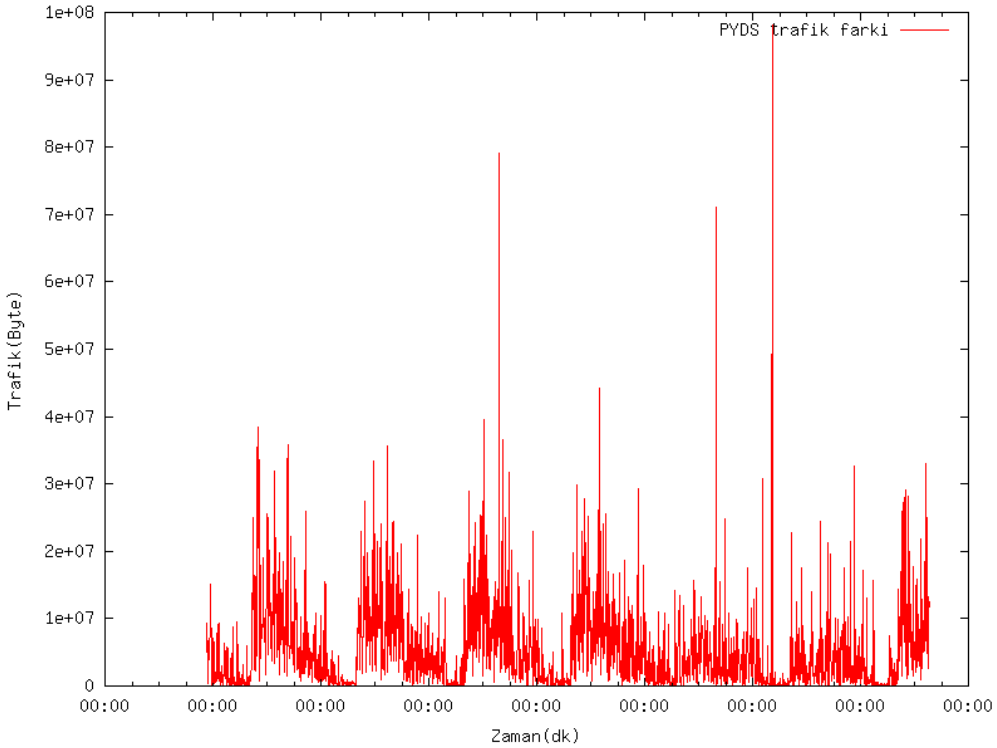


Şekil 4. 15. SMTP protokolü için üç makina ile yapılan benzetimin mod ve PYDS trafik fark grafiklerinin birlikte gösterimi.

Şekil 4.16 ve Şekil 4.17'de ise SMTP protokol verileri kullanılarak, PYDS'nin üç dengeleyici benzetimi ve sadece matematiksel mod işlemi sonucu oluşan grafikler ayrı ayrı gösterilmiştir. Bu iki grafiğin birleşimi Şekil 4.15'deki grafiği meydana getirmektedir.



Şekil 4. 16. SMTP protokolü için üç makina ile yapılan benzetimin mod fark grafiği.



Şekil 4. 17. SMTP protokolü için üç makina ile yapılan benzetimin PYDS fark grafiği.

4.2.2. HTTP ve SMTP protokolü fark verilerinin ve grafiklerinin analizi

Trafik fark grafikleri, yük dengeleme sistemlerinde dengesizlik durumunu en iyi şekilde gösterecek yöntemlerden biridir. Paralel şekilde çalışan dengeleyiciler arasında trafik dengesizliği ne kadar az olursa fark grafiklerindeki y eksenindeki değerleri 0 değerine o kadar yakın olmaktadır. Diğer yandan trafiğin az olması ya da olmaması halinde de y eksenindeki değerin 0'a yaklaşması ya da 0 olması beklenen bir durumdur.

Bir önceki bölümde HTTP ve SMTP protokolleri için kaydedilmiş akış verilerinin benzetim yoluyla elde edilmiş fark grafikleri verilmiştir.

HTTP protokolü için, iki dengeleyicinin paralel çalışma benzetiminin yapıldığı ve matematiksel mod ve referans yerelliği kullanarak dengeleme yapan PYDS'nin trafik fark grafiklerinin birlikte gösterildiği Şekil 4.6 ele alınırsa, yeşil olan PYDS fark grafiğinin, kırmızı renkli matematiksel mod trafiğine göre birçok noktada daha küçük değerler aldığı görülmektedir. Diğer yandan, keskin trafik atılımlarında aynı şekilde PYDS değerlerinin mod değerlerinin altında kaldığı görülebilir. Benzer şekilde üç dengeleyicinin benzetim grafiği Şekil 4.9, PYDS'nin iki dengeleyici benzetim grafikleri ile karşılaştırıldıklarında daha az başarım gösterdiği söylenebilir. Ancak her iki benzetim durumunda da, PYDS'nin trafiği matematiksel mod ile çalışan dengeleyicilere göre daha iyi dengelediği söylenebilir. Bu durum aynı zamanda fark trafiği verileri üzerinde yapılan matematiksel analizler ile de gösterilecektir.

Fark grafikleri üzerinde dengesizlik dağılım miktarı, standart sapmanın aritmetiksel ortalamaya oranı olan değişkenlik katsayısı (Coefficient of Variation – CV) [94] ile tanımlanabilir. Değişkenlik katsayısı Eş. 4.10'da verilmiştir.

$$C_v = \frac{\sigma}{\mu} \tag{4.10}$$

HTTP protokolü için değişkenlik katsayısı hesaplamaları Çizelge 4.4'de verilmiştir.

Çizelge 4. 4. HTTP protokolü için değişkenlik katsayı değerleri.

Makina Adeti	σ	μ	C_v
2 (mod)	705079.928	661172.022	1.066
2(PYDS)	671192.638	636888.273	1.053
3(mod)	799003.651	908722.343	0.879
3(PYDS)	749454.582	878120.209	0.853

Çizelge 4.4'de HTTP protokolü için verilen tabloda birinci sütunda benzetim yapılan makina adetini, ikinci sütunda standart sapma, üçüncü sütunda aritmetik ortalama ve son sütun da Eş. 4.10 ile hesaplanan değişkenlik katsayısı değerleri izler. Birinci sütunun satırlarında parantez içinde hangi tür sistem için benzetim yapıldığı tanımlanmıştır.

Değişkenlik katsayısı değerlerinin PYDS için daha düşük olması fark grafiklerinde de görülen dengesizlik durumunun PYDS tarafından daha iyi şekilde giderildiğini göstermektedir.

SMTP protokolü için, iki dengeleyicinin paralel çalışma benzetiminin yapıldığı ve matematiksel mod ve referans yerelliği kullanarak dengeleme yapan PYDS'nin, trafik fark grafiklerinin birlikte gösterildiği Şekil 4.12 ele alınırsa, yeşil olan PYDS fark grafiğinin, kırmızı renkte olan matematiksel mod trafiğine göre değerlerin bazı noktalarda daha küçük olduğu görülmektedir. Diğer yandan, ani dengesizlik durumlarını da tolere edebildiği görülebilir. Benzer şekilde üç dengeleyicinin benzetim yapıldığı Şekil 4.15 ele alındığında, PYDS'nin iki dengeleyici benzetimindekinden daha az başarım gösterdiği görülebilir. Ancak PYDS'nin SMTP protokolü için HTTP protokolünde elde ettiği kadar iyi sonuçlar vermediği dikkat çekicidir. Bunun nedeni SMTP protokolünün referans yerelliğine modeline tam uygun olacak şekilde davranış göstermemesi olabilir.

SMTP protokolü için değişkenlik katsayısı hesaplamaları Çizelge 4.4'de verilmiştir. PYDS'nin iki dengeleyicili benzetimde değişkenlik katsayısının daha az olması,

dengelemeyi daha iyi yapabildiğini göstermektedir. Ancak PYDS üç dengeleyici benzetiminde matematiksel mod işlemine göre farklı sonuç vermiştir.

Çizelge 4. 5. SMTP protokolü için değişkenlik katsayı değerleri.

Makina Adeti	σ	μ	C_v
2 (mod)	5484436.539	3711352.643	1.477
2(PYDS)	5346499.172	3676837.186	1.454
3(mod)	6753056.852	5274805.785	1.280
3(PYDS)	6907262.499	5326733.460	1.296

5. SONUÇ VE ÖNERİLER

Bilgisayar iletişim sistemlerinden, paralel yük dengeleme sistemi (PYDS) olarak geliştirilen bu model, yazılım seviyesinde çalışmaktadır. Referans yerelliğini temel alan ilk dengeleme modelidir. Bu model ile yapılan dengeleme benzetimlerinde, halen tüm PYDS'de kullanılan matematiksel mod işlemi ile karşılaştırma yapılmış ve mod ile dengelemeye oranla daha iyi sonuçlar elde edilmiştir.

Bu çalışmada, Gazi Üniversitesi (155 MB ATM SONET, gelen trafik ~1,58 TB/ay, giden trafik ~544 GB/ay – ulaknet trafik istatistikleri) servislerini kullanan istemcilerin, yarattıkları trafik akışları üzerine yapılan çalışmalarla, bu trafik akışları içerisinde referans yerelliğinin olduğu tespit edilmiştir. Bu tespit daha önce yapılan benzer çalışmalarla uyum içindedir.

Bu model ile aynı anda çalışan yük dengeleyiciler üzerinden geçen trafik miktarının, biribirine eşit hale getirilerek dengelenmesine çalışılmıştır, geliştirilen model gerçek trafik verileri ile benzetim yapılarak denenmiştir, olumlu sonuçlar alınmıştır.

Ancak üç dengeleyicili SMTP protokolünde olduğu gibi, trafik akışlarının referans yerelliğini adresleyecek karakteristik dışına çıkması halinde geliştirilen sistem verimli olamamaktadır. Geliştirilen model referans yerelliği bulunan gerçek sistemlerde kullanılabilir.

Geliştirilen model Gazi Üniversitesi trafiği üzerinde denenmiş, sistemde herhangi bir yavaşlamaya neden olmamıştır. Geliştirilen model üniversite girişindeki ateş duvarı (firewall) üzerinde çalışırken performans düşüşü veya sistemin durması oluşmamış, trafik akışı etkilenmemiş, uygulama başarı ile yapılmıştır.

PYDS'nin gerçek sistemlerde de paralel olarak çalışacak şekilde denenmesi ve sonuçların benzetimdeki sonuçlarla karşılaştırılması da önemlidir. Bu şekilde bir geliştirilmede, çekirdek seviyesinde çalışabilen filtreleme sisteminin eklenmesi sistem performansını olumlu yönde etkileyecektir.

Yük tanımlarının ve servislerin sürekli değişmesi ile birlikte, yeni yük dağıtım sistemlerine ihtiyaç duyulacaktır. Bu geliştirilecek sistemlerin çok daha hızlı karar verebilen ve hata toleransını mümkün olacak en aza indirecek şekilde tasarlanması gerekecektir. Bu konuda yazılımsal çözümlerin yanında, donanım yardımı ile de çalışabilen sistemler, darboğaz oluşturan filtreleme sistemi için yardımcı olacaktır.

Özellikle ağ katmanında çalışan cihazlarda kullanılan, akıllı ağ işlemcileri (Network Processor Unit – NPU) [95] kullanılarak, filtre tablosunun ağ katmanı seviyesinde çalışmasını sağlamak, paralel yük dengeleme çalışmalarında çok daha iyi performans alınmasına yardımcı olacaktır.

KAYNAKLAR

1. Roberts, L. G., "Multiple Computer Networks and Intercomputer Communication", ***ACM Symposium on Operating Systems Principles*** (1967).

2. Kleinrock, L., "Information Flow in Large Communication Nets", ***RLE Quarterly Progress Report, MIT***, July (1961).

3. İnternet: Postel, J., "TCP/IP, Transmission Control Protocol", ***IETF***, http://www.ietf.org/rfc/rfc0793.txt (1981). *

4. ITU's Strategy & Policy Unit , "ITU Internet Reports 2006: digital life", ***ITU TELECOM World Hong Kong***, 12-16, (2006).

5. Cardellini, V., Colajanni, M., Philip S. Yu. "Dynamic Load Balancing on Web-Server Systems", ***IEEE Internet Computing***, Mayıs-Haziran (1999).

6. İnternet: International Organization for Standardization, ***ISO standard 7498-1:1994***, http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip (1994). *

7. Stevens, W. R., "Unix Network Programming Vol 1", ***Prentice Hall PTR***, 18 (1998).

8. Çağıltay, K., "Herkes İçin Internet", ***Tübitak***, Bölüm 2 (1994).

9. Halsall, F., "Data Communications, Computer Networks and Open Systems", ***Addison-Wesley***, 258-264 (1997).

10. McQuerry, S., "CCNA Self-Study: Introduction to Cisco Networking Technologies", ***Cisco Press***, 65 (2004).

11. İnternet: Postel, J., "Internet Protocol (IP)", ***IETF***, http://www.ietf.org/rfc/rfc0791.txt (1981). *

12. Stevens, W. R., Fenner, B., Rudoff, A. M., "Unix Network Programming", ***Addison-Wesley***, (2004).

13. İnternet: Postel, J., "UDP, User Datagram Protocol", ***IETF***, http://www.ietf.org/rfc/rfc0768.txt (1981). *

14. İnternet: Jacobson, V., Braden, R., Borman, D., "TCP Extensions TCP Extensions for High Performance", ***IETF***, http://www.ietf.org/rfc/rfc1323.txt (1992). *

15. İnternet: Allman, M., Paxson, V., Stevens, W. R., ”TCP Congestion Control”, ***IETF***, http://www.ietf.org/rfc/rfc2581.txt (1999). *

16. İnternet: Allman, M., Paxson, V., ”Computing TCP’s Retransmission Timer”, ***IETF***, http://www.ietf.org/rfc/rfc2988.txt (2000). *

17. İnternet: Allman, M., Floyd, S., Partridge, C., “Increasing TCP’s Initial Window”, ***IETF***, http://www.ietf.org/rfc/rfc3390.txt (2002). *

18. McCreary, S., Claffy, K., “Trends in Wide Area IP Traffic Patterns”, ***CAIDA*** (2000).

19. Bhatt, M. J., “Locality Of Reference” , ***PloP’97 Conference***, (1997).

20. Denning, P.J, Schwartz, S., “Properties of working set model” , ***Communications of the ACM***, 15(3):191-198 Temmuz (1972).

21. Snir, M., Yu, J., “On the Theory of Spatial and Temporal Locality”, ***Technical Report No. UIUCDCS-R-2005-2611, Department of Computer Science***, UIUC, Temmuz (2005).

22. Class, R., “In the Beginning: Recollections of Software Pioneers”, Bölüm 6, ***IEEE Press***, (1997).

23. Davis, W. S., Rajkumar, T. M., “Operating Systems, Systematic View 6th ed.”, ***Addison Wasley***, 125-130 (2005).

24. Tanenbaum, A. S., “Operating Systems: Design and Implementation (Second Edition)”, ***Prentice-Hall***, 286-290 (1997).

25. Denning, P. J., “The Working Set Model for Program Behavior”, ***Communications of the ACM***, 323-333 Mayıs (1968).

26. Denning, P.J., Schwartz, S., “Properties of working set model” , ***Communications of the ACM***, 191-198 (1972).

27. Aho, A. V., Denning, P. J., Ullman, J. D., Principles of optimal page replacement , ***Communications of the ACM***, 80-93 (1971).

28. Zipf, G. K., “Human Behaviour and the Principle of Least-Effort”, ***Addison-Wesley***, Cambridge MA, (1949).

29. Bunt, R. B., Murphy, J. M., “The measurement of locality and behavior of programs”, ***The Computer Journal***, 238-245 (1984).

30. Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S., "Web caching and Zipf-like distributions, evidence and implications", ***IEEE INFOCOM 1999*** (1999).

31. Adamic, L. A., Huberman, B. A., "Zipf's law and internet", ***Glottometrics 3***, 143-150 (2002).

32. Spirn, J. R., "Distance string models for program behavior", ***IEEE Computer***, 9:14-20 (1976).

33. Shi, W., MacGregor, M.H., "Gburzynski, P., Traffic locality characteristics in a parallel forwarding system". ***International Journal of Communication Systems***, 16:823-839 (2003).

34. Jain, R., Routhier, S. A., "Packet Trains and New Model for Compter Network Traffic", ***IEEE Journal of Selected Areas Communications***, (1986).

35. Larson, R., Farber, B., "Elementary Statistics Picturing the World 2nd ed.", ***Prentice Hall***, 189 (2003).

36. Feldmeier, D. C., "Improving Gateway Performance With A Routing-Table Cache", ***IEEE INFOCOM***, (1988).

37. Mogul, J. C., "Network Locality at the Scale of Process", ***Digital Equipment WRL Research Report 91/11***, Kasım (1991).

38. Partridge, C., Pink, S., "A Faster UDP" , ***IEEE/ACM Transactions on Networking***,1:(4) (1993).

39. Broido, A., Hyun, Y., Claffy, K., "Their share: diversity and disparity in IP traffic", ***Presented at the PAM workshop*** , (2004).

40. Reed, W. J., "The Pareto, Zipf and other power laws", ***Economics Letters***, 74:(1): 15-19 (2001).

41. Almeida, V., Bastavros, A., Crovella,M., Oliveria, A., "Characterizing Refernce Locality in the WWW", ***IEEE Conference on Parallel and Distributed Information Systems***, (1996).

42. Arlitt, M.F., Williamson, C.L., "Internet Web Servers: Workload Characterization and Performance Implications", ***IEEE/ACM Transactions on Networking***, Vol 5. No: 5, Ekim (1997).

43. He, C., Karamcheti, V., "An Analysis of Usage Locality for Data-Centric Web Services", ***WWW 2005***, (2005).

44. Brownlee, N., Murray, M., "Streams Flows and Torrents", ***RIPE PAM 2001*** (2001).

45. Claffy, K., Braun, H., Polyzos, G., "A parameterizable methodology for Internet traffic flow profiling", ***IEEE Journal of Selected Areas Communications***, 13(8):1481-1494 (1995).

46. Caceres, R., Danzig, P. B., Jamin, S., Mitzel, D. J., "Characteristics of Wide, Area TCP/IP Conversations", ***ACM SIGCOMM'91***, (1991).

47. Leland, W.E., Taqqu, M.S. , Wilson, S.V.,"On the Self-Similar Nature of Ethernet Traffic", ***IEEE/ACM Transactions on Networking***, 2:(1) (1994).

48. Sarvotham, S., Baraniuk, R.,"Connection-level Analysis and Modeling of Network Traffic", ***ACM SIGCOMM INTERNET MEASUREMENT WORKSHOP***, (2001).

49. Willinger, W., Taqqu, M. S., Sherman, R., Wilson, D.V., "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level", ***IEEE/ACM Transactions on Networking***, 5:(1) (1997).

50. Sarvotham, S., Riedi, R., Baraniuk, R., "Network Traffic Analysis and Modeling at the Connection Level Technical Report" , ***ECE Dept., Rice University***, (2001).

51. N. Brownlee, K.C Claffy, "Understanding Internet Traffic Streams: Dragonflies and Tortoises", ***IEEE Communications***, 40(10):110-117 (2002).

52. Pfister, G. F., "In Search of Clusters 2nd Edition", ***Prentice Hall***, 88-101 (1998).

53. Mosedale, D., Foss, W., McCool, R., "Lessons Learned Administrating Netscape's Internet Site", ***IEEE Internet Computing***, 1(2):28-35 (1997).

54. Yoshikawa, C., Chun, B., Eastham, P., Vahdat, A., Anderson, T., Culler, D., "Using Smart Clients to Build Scalable Services", ***Proc. Usenix 1997, Usenix Assoc.***, Berkeley, (1997).

55. Kwan, T.T., McGrath, R. E., Reed, D. A., "NCSA's World Wide Web server: Design and Performance.", ***Computer***, 28(11):68-74 Kasım (1995).

56. Cardellini, V., Colojanni, M., Yu, P. S., "DNS Dispatching Algorithms with State Estimators for Scalable Web Server Clusters". ***World Wide Web J.***, Baltzer Science, 2:(3):101-113 (1999).

57. Schroeder, T., Goddard, S., Rammamurthy, B., "Scalable Web Server Clustering Technologies", ***IEEE Network***, 14(3):38-45 (2000).

58. Dias, D.M., Kish, W., Mukherjee, R., Tewari R., "A scalable and highly available web server", ***Compcon'96***, 85-92 (1996).

59. Damani, O. P., Chung, P.Y.E., Huang, Y., Kintala, C., Wang, Y. M., "ONE-IP: Techniques for hosting a service on a cluster of machines", ***Computer Networks and ISDN Systems***, 29:1019-1027 (1997).

60. Chu, C., "IP Cluster", ***IETF***, http://tools.ietf.org/html/draft-rfced-info-chu-00.txt, (1996). *

61. Goldszmidth, G., Hunt, G., "Net dispatcher: A TCP Connection Router", ***IBM Resarch Report***, RC 20853, Mayıs (1997).

62. Gan, X., Schroeder, T., Goddard S., Ramamurthy, B., "LSMAC and LSNAT: Two Approaches for Cluster-based Scalable Web Servers", ***Proceedings of ICC 2000, New Orleans, Louisiana***, 2:1164-1168 (2000).

63. Zhang, W., "Linux Virtual Server for Scalable Network Services", ***Published at Ottawa Linux Symposium***, (2000).

64. Roelle, H., "A Hot-Failover State Machine for Gateway Services and Its Application to a Linux Firewall", Proceedings of the 13th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM 2002), Lecture Notes in Computer Science (LNCS) IFIP/IEEE, ***Springer***, 181-194 (2002).

65. Srisuresh, P., Gan, D., "Load Sharing using IP Network Address Translation (LSNAT)" , ***IETF***, http://www.ietf.org/rfc/rfc2391.txt (1998). *

66. Anderson, E., Patterson, D., Brewer, E., "The Magicrouter, an Application of Fast Packet Interposing", ***Submitted for publication in the 2nd Symposium on Operating Systems Design and Implementation***, (1996).

67. Delgadillo, K, "Cisco Distributed Director", ***Cisco White Paper*** (1999)

68. Robertson, A. L., "Linux-HA Heartbeat Design", ***Proceedings of the 4th International Linux Showcase and Conference, Atlanta***, (2000).

69. Service Monitoring Daemon, http://www.kernel.org/software/mon (2006). *

70. Pai, V., "Locality-Aware Request Distribution in Cluster-based Network Servers", ***Proc. ACM 8th Int'l. Conf. Arcitectural Support for Prog. Langs. An Op. Sys***., 205-216 (1998).

71. Bakre, A., Badrinath, B. R., "Handoff and System Support for Indirect TCP/IP", ***Proc. Second USENIX Symp. Mobile and Location-Independent Computing***, 11-24 (1995).

72. Verwoerd, T., Hunt, R., "GLOB: Generic Load Balancing", ***Proceedings of the 9th IEEE International Conference on Networks, Bangkok***, 70 (2001).

73. Xiong Z., Yan, P., Guo, C., "MDDR: a solution for improving the scalability of dispatcher-based Web server cluster", ***Communications,*** 1:38-42 (2005).

74. Williamson, C., "Internet Traffic Measurement", ***IEEE Internet Computing***, 5(6):70-74 (2001).

75. Weigle, E., Feng, W., "TICKETing High-Speed Traffic with Commodity Hardware and Software", ***Passive & Active Measurement Workshop (PAM2002)*** (2002).

76. İnternet: CAIDA veri kaynakları, http://www.caida.org/data/index_source.xml (2006). *

77. Seshadri, M. S., Bent, J., Kosar, T., "Network processors: Guiding design through analysis", Course Project, http://www.cs.wisc.edu/~johnbent/Projects/ (2001). *

78. Casey, B. J., "Implementing Ethernet in the industrial environment", Electrical Engineering Problems in the Rubber and Plastics Industries, ***IEEE Conference Record of 1990***, 32-40 (1990).

79. İnternet: CAIDA Measurement and Analysis Tools, http://www.caida.org/tools/measurement (2006). *

80. McCanne, S., Leres, C., Jacobson, V., "PCAP, Packet Capture Interface", ***Lawrence Berkeley Laboratory, Berkeley***, http://www.tcpdump.org/pcap3_man.html (2006). *

81. İnternet: Claise, B., "Cisco Systems Netflow Services Export Version 9", ***IETF***, http://www.ietf.org/rfc/rfc3954.txt (2004). *

82. İnternet: Miller, D., "Softflowd Uygulaması", http://www.mindrot.org/softflowd.html (2006). *

83. McCanne, S., Jacobson V., "The BSD Packet Filter: A New Architecture for User-level Packet Capture", ***Proceedings of the Winter 1993 USENIX Conference Usenix*** , 259-270 (1992).

84. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks", ***Communications of the ACM***, 13(6):100, (Haziran 1970).

85. Worsley, J.C., Darje, J. D., "Pratical PostgreSQL", ***O'Reilly Media***, (2002).

86. Wall, L., Christiansen, Orwant, J., "Programming Perl (3rd ed.)", ***O'Reilly Media***, 1-8 (2000).

87. Upton, G., Cook, I., "Understanding Statistics", ***Oxford University Press***, 265-273 (2000).

88. Oetiker, T., "MRTG – The Multi Router Traffic Grapher, Usenix", ***12th System Administration Conference (LISA'98)***, (1998).

89. Love, R., "Linux Kernel Development", 2nd Edition, ***Novell Press***, 5-12 (2005).

90. Deri, L., "Improving Passive Packet Capture: Beyond Device Polling", ***Proceedings of SANE 2004*** (2004).

91. Deri, L., "NCap: Wire-speed Packet Capture and Transmission", ***Proceedings of E2EMON*** (2005).

92. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C., "Introduction to Algorithms" , Second Edition, ***MIT Press and McGraw-Hill***, 13:273–301 (2001).

93. Knuth, D. E., "Big Omicron and Big Omega and Big Theta", ***SIGACT News***, 18-24 (1976).

94. Beatty, J. R., "Statistical Methods 5th ed.", ***McGraw-Hill Primis***, 1:4-50 (1997).

95. İnternet: Shah, N., "Understanding Network Processors", http://www.cs.berkeley.edu/~plishker/UnderstandingNPs.pdf, (2001). *

* Bu şekilde verilmiş kaynakların içerdiği bilgiler hiçbir basılı kaynakta bulunmadığından internet kaynağı şeklinde gösterilmek zorunda kalınmıştır.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: SERAL, Devrim
Uyruğu	: K.K.T.C
Doğum tarihi ve yeri	: 13.08.1977 Lefkoşa
Medeni hali	: Bekar
Telefon	: 0 (312) 286 82 00
e-mail	: devrim@gazi.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Gazi Üniversitesi/Elt ve Bil Eğt. Böl.	2000
Lisans	Gazi Üniversitesi/Elt ve Bil Eğt. Böl.	1998
Lise	Gazi Lisesi/Matematik	1994

İş Deneyimi

Yıl	Yer	Görev
1999-2001	Gazi Üniversitesi	Arş. Görevlisi
2001-2003	Gantek Teknoloji	Sistem Dst. Müh.
2003-2005	Alto Teknoloji	Sistem Dst. Müh.
2005-	Meteksan Net	İnternet Sis. Müh.

Yabancı Dil

İngilizce

Hobiler

Basketbol, Koşma, Politika, Kitap Okumak