Furkan Yanteri

Hocam mail yoluyla ödevi tek yapmak için talepte bulunmuştum, kabul etmiştiniz.Tek kişiyim grup27 olarak geçiyorum.

1)

Ödevde verilen trace dosyasının ‘’iz.tr’’ **ilk satırı** aşağıdadır.

Bu satırdan hareketle her bir kelimenin ne aanlama geldiğini göstereceğim.

+	1.10531	16	20	tcp	40	-------	0	16.0	21.16	0	0
1	2	3	4	5	6	7	8	9	10	11	12

1) Ilk alan olaydır. Size dört muhtemel sembol '+' '-' 'r' '' verir. Bu dört sembol sırasıyla sıraya sokulan, sıraya girmeyen, alınan ve bırakılan nesneye karşılık gelir.
2) İkinci alan, olayın meydana geldiği zamanı verir
3) Üçüncü alan, olayın meydana geldiği bağlantıdaki giriş düğümünü verir
4) Dördüncü alan, olayın meydana geldiği çıkış düğümünü verir
5) Beşinci alan, paketin türü hakkında bilgi gösterir. Paketin UDP veya TCP olup olmadığı
6) Altıncı alan paket boyutunu verir
7) Yedinci saha bazı bayraklar hakkında bilgi verir
8) Sekiz alan, bir kullanıcının bir tcl komut dosyasındaki her akış için ayarlayabileceği akış kimliği (fid) 'dir. Ayrıca, NAM görüntüsündeki akış rengini belirlemek için kullanılır
9) Dokuzuncu alan kaynak adresidir.
10) Onuncu alan İlk hedef adresidir.
11) Onbirinci alan, ağ katmanı protokolünün paket sıra numarasıdır
12) Son alan, paketin benzersiz kimliğini gösterir

2)

Bir network için performans metrikleri genel olarak şu şekildedir:

availability, response time, channel capacity, latency, completion time, service time, bandwidth, throughput, relative **efficiency**, scalability, **performance** per watt, compression ratio, instruction path length and speed up.

Ödevde kullanılabilecek bazı performans kriterleri:

- PDF = ∑ received data packets/∑ sent data packets
- *Throughput:* Bir ağın gönderdiği ve aldığı verinin oranı.(bit/s) olarak değerlendirilir. In other words, it is the amount of data (bits) transferred from the destination node to the source node during a specified amount of time (s).
- EED = ∑ (arrive time−send time)/∑ (number of connections).
- Routing overheads (Oh) = ∑ routing control packets sent.
- Paket kaybı = Gönderilen paket sayısı – Alınan paket sayısı

Ben ödevde **LATENCY**, **TROUGHPUT**, **PAKET KAYIP YÜZDESİ**,**PAKET İŞLEM SÜRESİ** ve **YANSIMA** performans metriklerini kullandım ve ilgili bölümlerde bunları açıkladım.

Furkan Yanteri

1)LATENCY (Gecikme)

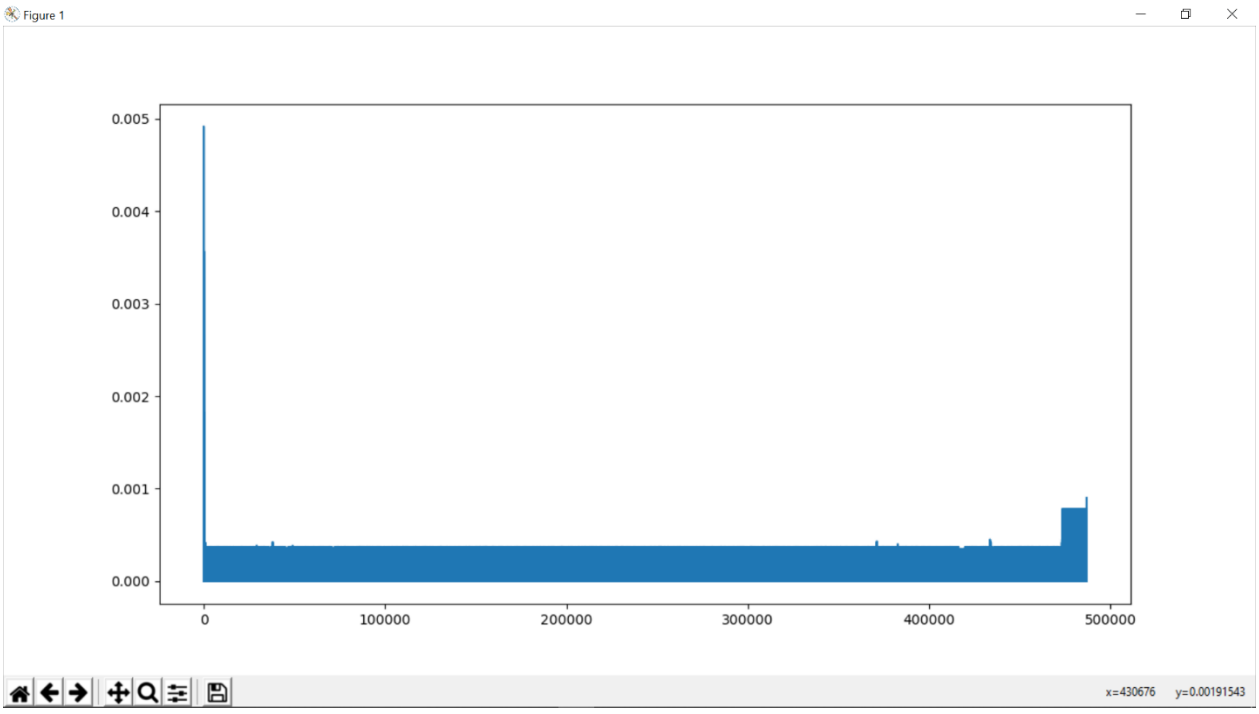
iz.tr | Latency.py

C: > Users > furkanyanteri > Desktop > fy > Mimar > Cozumler > deneme > son > Latency.py > ...

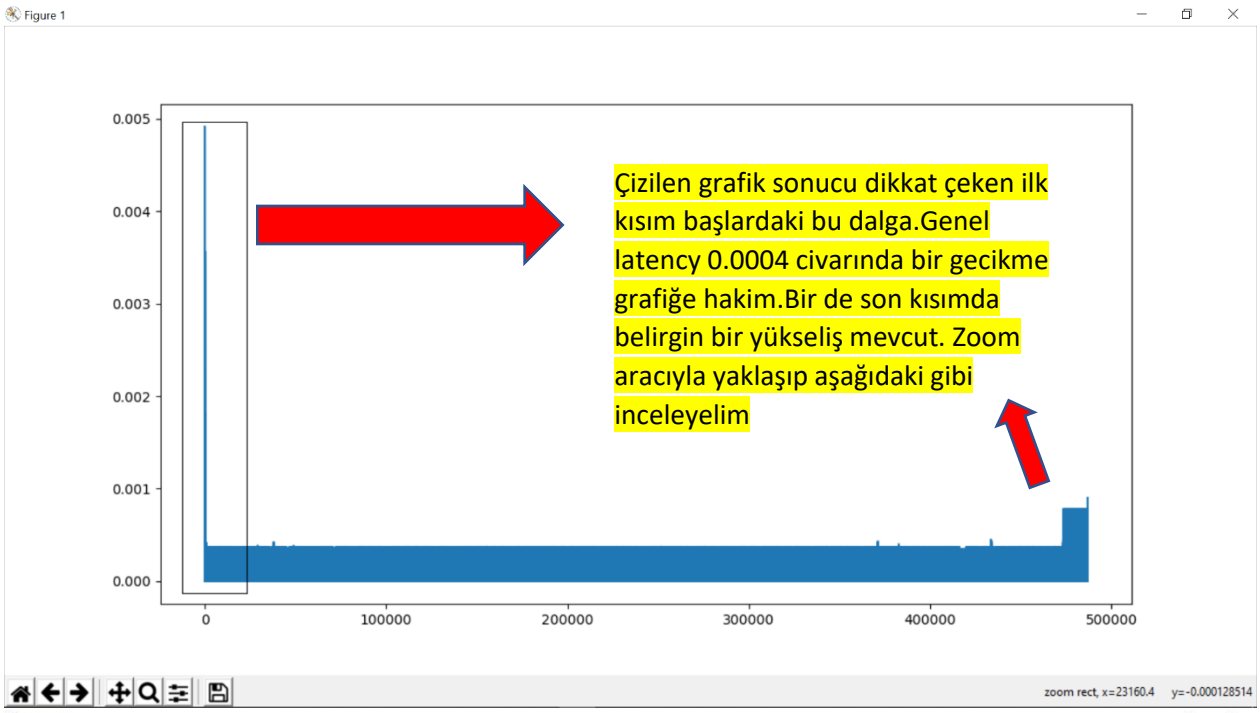
```
1   import matplotlib.pyplot as plt1
2   toplac=[];##icerisinde tutma icin gerekli.Sonradan append ile dolduracagiz
3   file = open("iz.tr", "r")
4   zaman1=zaman2=latency=0 ## Latency icin kullanilacak.Her ardisik iki olayin zamanlarinin farki lazim
5   for line in file:
6     #satiri ' ' karakterleriyle bolup verilerimizi alalim
7     fields = line.split(" ")
8     #hocam burada ingilizce isimleri kullaniyorum mecburen kusura bakmayiniz
9     time = fields[1]
10  #  print(float(time)+2) #burada sayisal deger basiyoruz
11    zaman2=float(time)
12    if zaman1==0:
13      toplac.append(0)
14      zaman1=zaman2
15      continue
16    toplac.append(zaman2-zaman1)
17    zaman1=zaman2
18
19  plt1.plot(toplac)
20  plt1.show()
21  file.close()
22
```

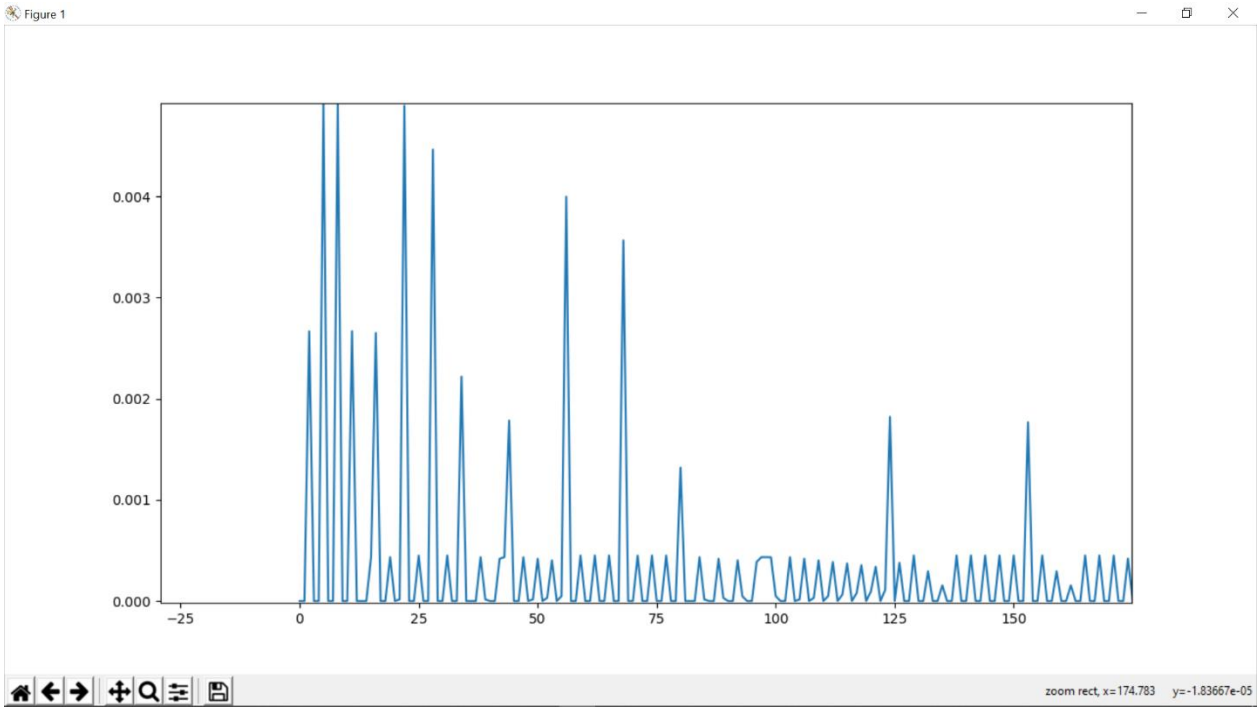
Genel hatları itibariyle performans kriterleri için yaptığım bu .py dosyaları için çalışma prensibi benzerdir.Bütün bu .py dosyalarında yorum satırları ile yapmak istediklerimi anlattım.Haricen bu rapor içerisinde yer yer eklemeler yapılacaktır.

Grafik matplotlib ile çizildi.Ardışık her olaylar için zaman farkı alıp toplaç içerisine bu verileri koyup plt1 e parametre olarak verip grafik aşağıdaki gibi çizilmiştir.



Furkan Yanteri





Görüldüğü gibi paket paket inceleme yapabileceğimiz seviyede net olarak görülüyor.Performans ile ilgili olarak başlangıçta 100 paket kadar bir zamanda sistemde hafif bir gecikme dalgalanması mevcut.Fakat bu kadar büyük bir izdosyası olduğunu düşünürsek çok ciddi değil.

```
print((sum(toplac)/len(toplac))*100000)
```

şeklinde bir kod kullanarak:

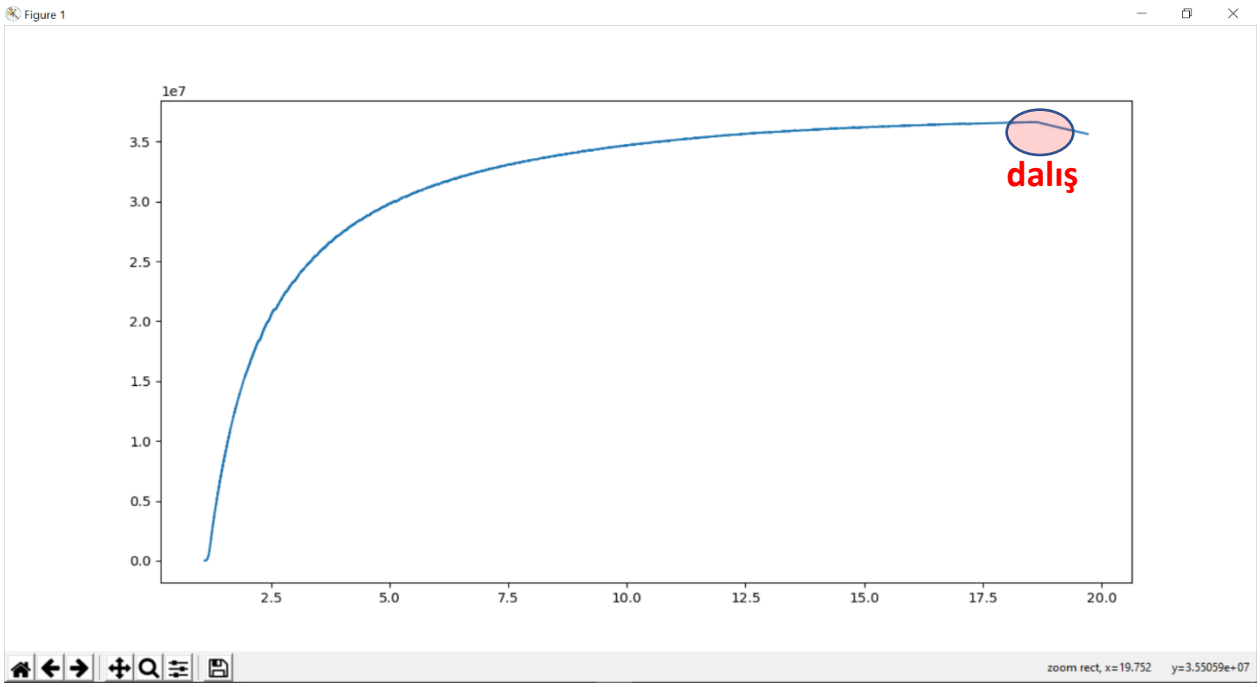
Ortalama Gecikme = 0.000038218901190106886 sn/pkt olarak bulunabilir.

Furkan Yanteri

2)TROUGHPUT

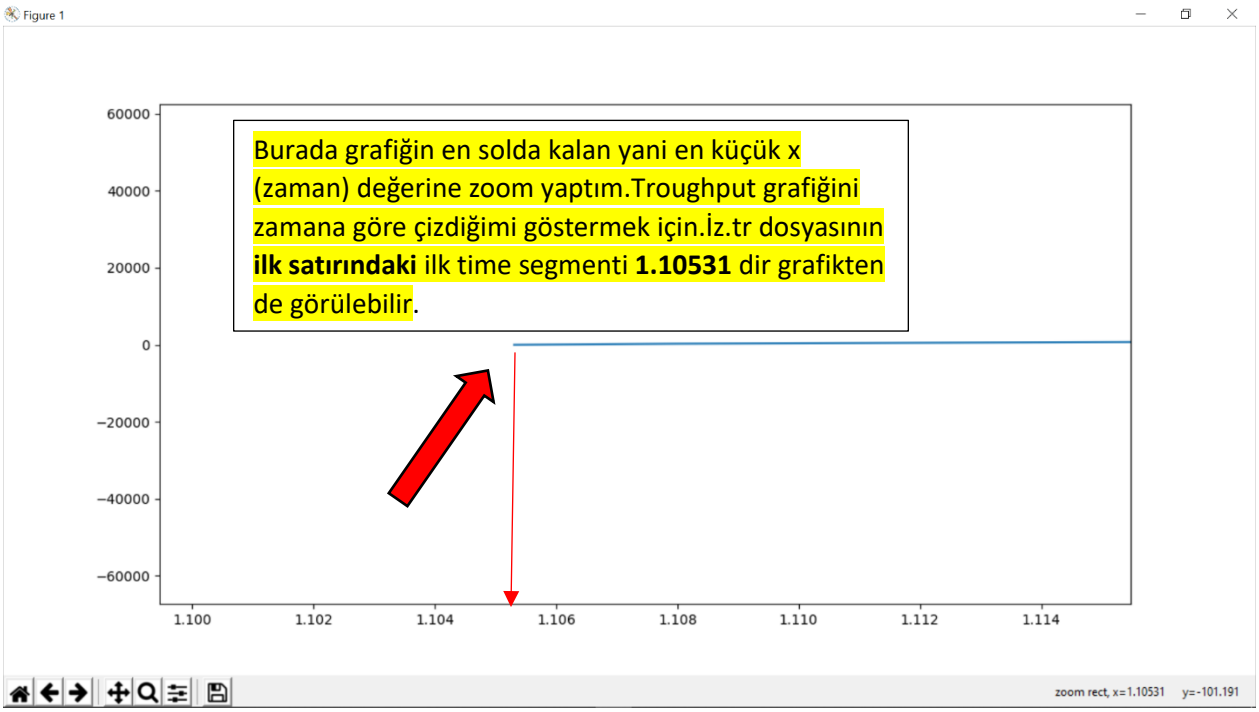
```python
import matplotlib.pyplot as plt1
toplac=[]#icerisinde tutma icin gerekli.Sonradan append ile dolduracagiz
saniye=[]#Her satirda yer alan saniye degerlerini de tutuyorum ki saniyeye gore troughput cizebilelim
veri=0#bunu o ana kadar elden gecen verilerin bit olarak toplamini tutmak icin kullanalim
file = open("iz.tr", "r")
for line in file:

  #satiri ' ' karakterleriyle bolup verilerimizi alalim
  fields = line.split(" ")
  #hocam burada ingilizce isimleri kullaniyorum mecburen kusura bakmayiniz
  event = fields[0]
  time = float(fields[1])
  pkt_size = float(fields[5])

  saniye.append(time)#o anki duvar saatini yani saniye degerini biriktir
#  print(float(time)+2) #burada sayisal deger basiyoruz
  if event=="r":
    veri+=pkt_size*8 #bit olarak kullanacagim cunku bps yani bit/sec olarak geciyor troughput

  toplac.append(veri/time)#burada belli oluyor (toplam veri/o anki zaman) olarak verileri topluyorum

print("--------------")

plt1.plot(saniye,toplac)#troughput grafigini saniyeye gore ciziyorum duvar saatine
plt1.show()
file.close()
```

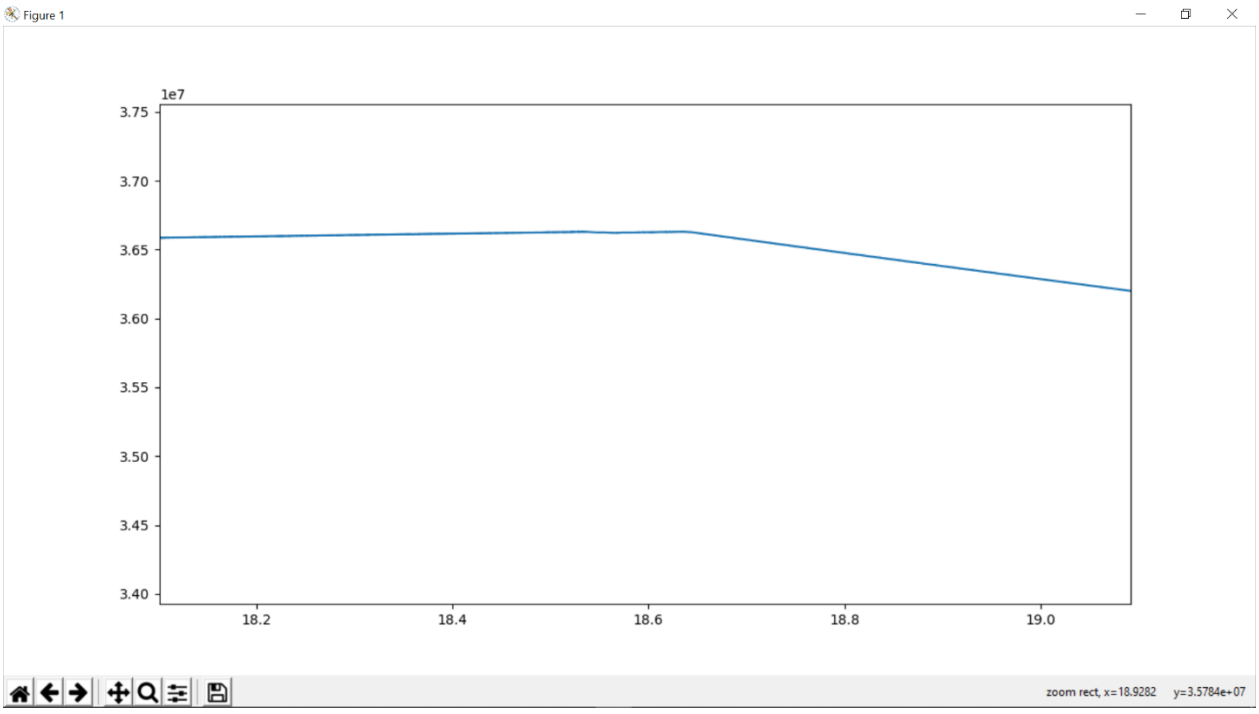
<u>Troughput grafiğini çizerken sadece eventi ‘r’ olanları kullandım</u>.Eventi +,-,r,d olanları kullanmadım çünkü bunlar troughputa eklenmemeli diye düşündüm.Yani enque veya deque edilen paketler veya drop edilen paketlerin totalde yaratılan saniye başına trafiğe etki etmediğini düşündüm.Aşağıda çizdirdiğim grafiklerde troughput **bit/saniye** formatındadır.Kodun içinde **8** ile çarpılmasının sebebi budur.Çünkü trace dosyasında paket size kısmı **byte** formatındadır.



Furkan Yanteri

Yukarıdaki şekilde görüldüğü üzere Troughput azalarak artan bir yapıdadır.Burada yandaki rakamlar örnek olarak 3.2 gibi gözükebilir ama y ekseninin en tepesine bakarsak e7 yazdığını görebiliriz.Yani grafikte y ekseninde **3.2 gibi görünen bir değer 3.2*10^7 değeridir.**





Yukarıdaki şekilde, ilk şekilde işaretlenen dalış bölgesinin zoom edilmiş hali gözükmektedir.Yuvarlak olarak 18,63 saniyesinden sonra bir dalış var.

if veri/time>max:

max=veri/timemax

Şeklinde koda bir ekleme yaparak troughputun **maksimumu** gördüğü andaki değerini **36629382.51230616 bit/saniye** olarak buldum.

Benzer şekilde trace dosyasının tamamında Troughput için **ortalama** değeri **3155774.4703301694** olarak buldum.

3)Paket Kayıp Yüzdesi

Öncelikle trace dosyasında dikkatimi çeken şey event kısmında 's' olan pakete hiç rastlanmaması.Belki de trace dosyalarının çeşitlerinden birisidir bu bilgiyi bulamadım.Bu yüzden Packet Delivery Fraction(PDF) Performans Metriğini hesaplamadım ve çizmedim.

C: > Users > furkanyanteri > Desktop > fy > Mimar > Cozumler > deneme > son > Paket_Kayip_Yuzdesi.py > ...

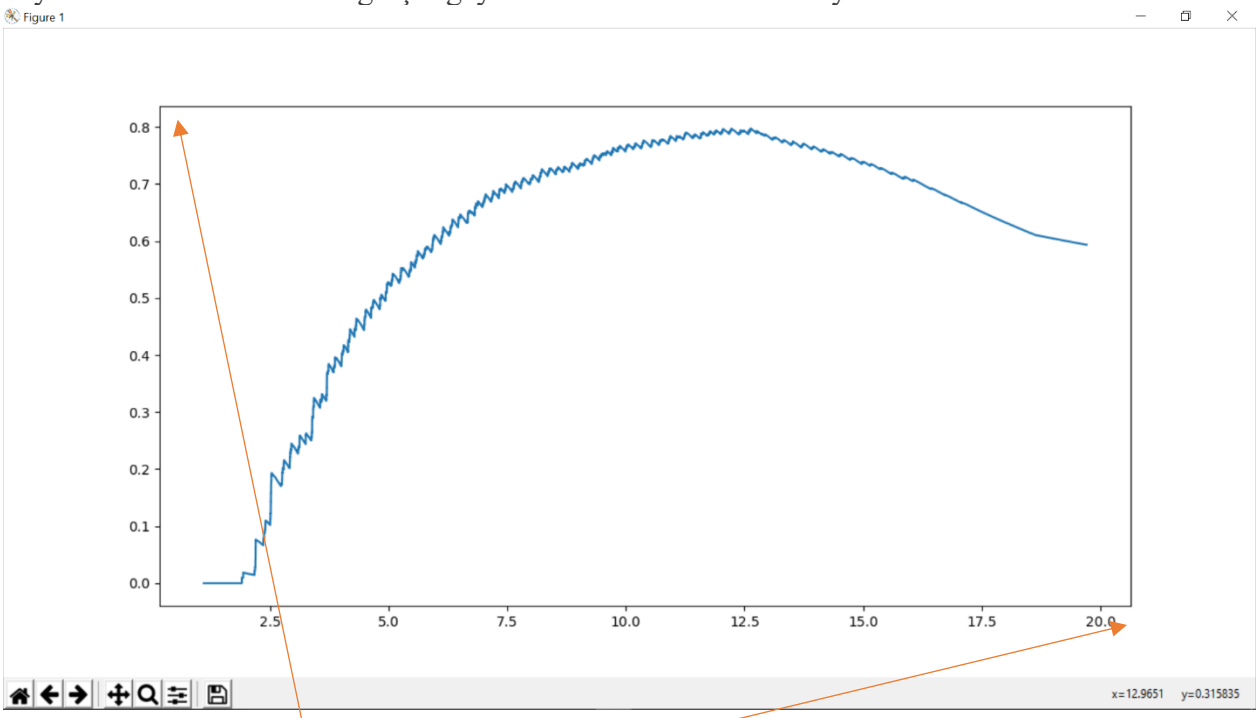
```
import matplotlib.pyplot as plt1
toplac=[];##icerisinde tutma icin gerekli.Sonradan append ile dolduracagiz
zamanci=[];#grafigi zaman eksenine gore cizmeye calisalim duvar saati
file = open("iz.tr", "r")
byte_kayip=0 #kaybolan paketlerin byte cinsinden toplam buyuklugu
byte_toplam=0 #butun paketler icin sizelari toplayalim oranlamak icin
for line in file:
  #satiri ' ' karakterleriyle bolup verilerimizi alalim
  fields = line.split(" ")
  event = fields[0]
  time = float(fields[1])
  pkt_size = float(fields[5])
  if event=="d":
    byte_kayip+=pkt_size #paket kayip ise kayip bytler silsilesine bir yenisini daha ekle
  byte_toplam+=pkt_size #kayipsa da degilse de paket boyutuna ekle cunku bu ornek uzay teskil eder

  toplac.append(byte_kayip/byte_toplam*100) #100 ile carpmamim sebebi bu ihtimali yuzdelik olarak gostermek
  zamanci.append(time)

print("-----> Paket Kayip Yuzdesi Maksimum Degeri:",max(toplac))
print("-----> Paket Kayip Yuzdesi Ortalama Degeri:",sum(toplac)/len(toplac))

#plt1.plot(toplac)
plt1.plot(zamanci,toplac)
plt1.xlabel=("zaman")
plt1.ylabel=("Paket Kayip Yuzdesi")
plt1.show()
file.close()
```

Bu metrikte yaklaşımım aslında şu.**Bir ağ trafiğinde gönderilen paketlerden drop edilen yani kaybedilen paketlerin alınan bütün paketlere oranı ne kadar düşük olursa temelde performansın o kadar iyi olacağını beklemek**.Burada bu oranı alırken paketlerin **boyutları** üzerinden bir yüzdelik oran çıkarımında bulundum.Yani toplam kaybedilen bytelar/bütün alınan bytelar gibi bir formülasyon yapılabilir.Bunu zaman veya duvar saatine göre tabloya yerleştirdim.

Furkan Yanteri

```
C:\Windows\System32\cmd.exe

C:\Users\furkanyanteri\Desktop\fy\Mimar\Cozumler\deneme\son>python Paket_Kayip_Yuzdesi.py
-----> Paket Kayip Yuzdesi Maksimum Degeri: 0.7972096035493158
-----> Paket Kayip Yuzdesi Ortalama Degeri: 0.6066866602474482

C:\Users\furkanyanteri\Desktop\fy\Mimar\Cozumler\deneme\son>
```

Grafiğe geçmeden önce yukarıda terminalde bu kod çalıştırıldığında gösterilen iki değer var.Bunlar maksimum ve ortalama Paket Kayıp Yüzdeleri.Yani nihayetinde yaklaşık 20 saniye içinde(20 trace dosyasındaki son paketin time ı) ortalama **%0.6066866602474482** lık bir paket kayıp yüzdesi ile gerçekleştirilmiş.Yapılmak istenilen işe göre çok berbat bir oran da olabilir veya bir video aktarım örneği için gayet tolere edilebilir bir düzeydedir.

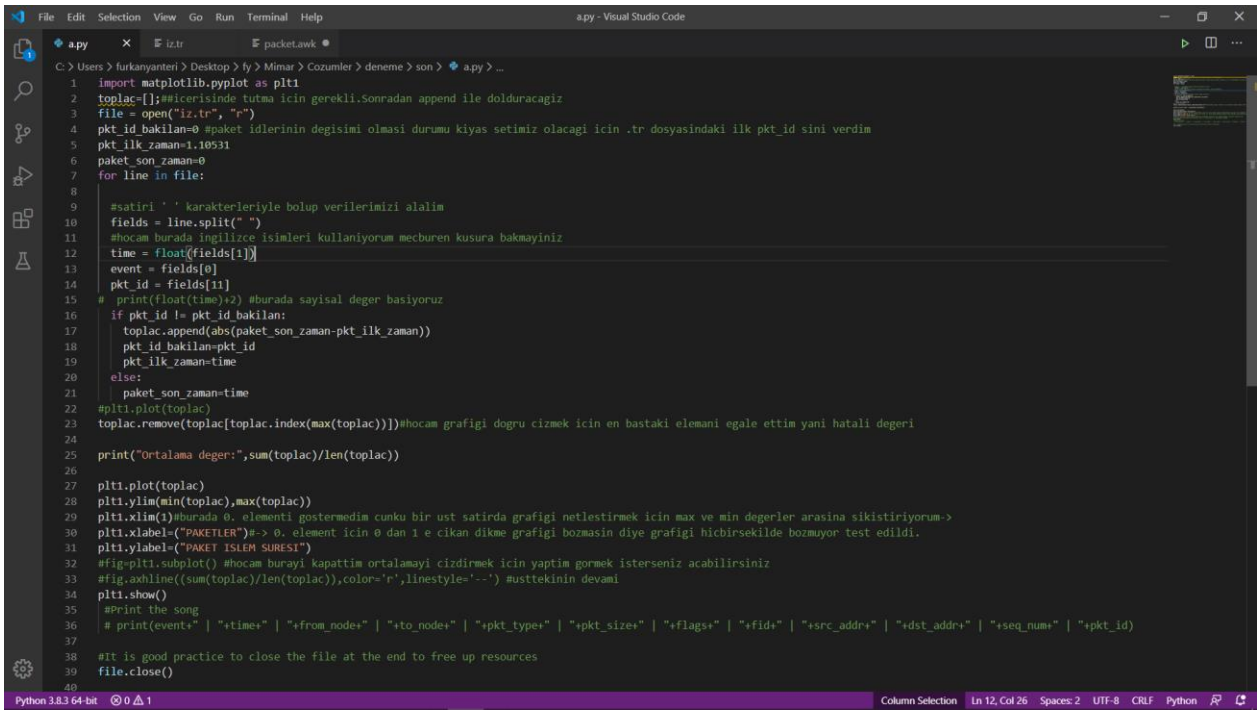


Yukarıdaki paket kayıp yüzdesinin paket zamanına göre grafiğidir.Grafikten zamana göre çizildiği zaman ekseninde gidilen son noktanın trace dosyasındaki son paketin zamanıyla aynı olduğu görülerek anlaşılabilir.16.saniye civarında azalmakta olan grafiğin dalgalanmaları azalmış olup daha kararlı ve daha yavaş bir şekilde azalmaya devam ettiği görülmektedir.Bir ağın performansı paket kayıp yüzdesi ile ters orantılıdır.Fakat paket kayıp toleransı yüksek olan bir işlem için daha çok kayıp vererek daha çok paket gönderilebileceği için hızlanma da sağlanabilir.

4)Paket İşlem Süresi

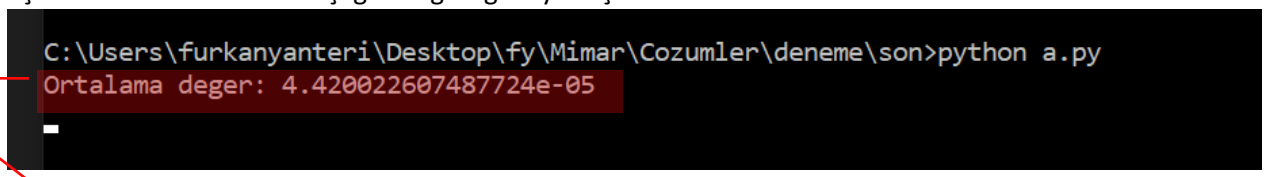
Hocam ismini ben böyle yazdım.**End to end delay** dedikleri performans metriğinden esinlendim.Bu performans metriğinde gelen her satır için, bir önceki satırla aynı paket ile ilgili bir işlem mi yapılıyor diye bakıyorum.Bunu yapabilmek için **paket id** leri kullanıyorum ve **başka bir pakete geçmeden önce o paket için ne kadar vakit harcanmış** onu hesaplıyorum. Paket değişmeden hemen önce(Paket işlem bitiş zamanı-Paket işlem başlangıç zamanı)

Furkan Yanteri



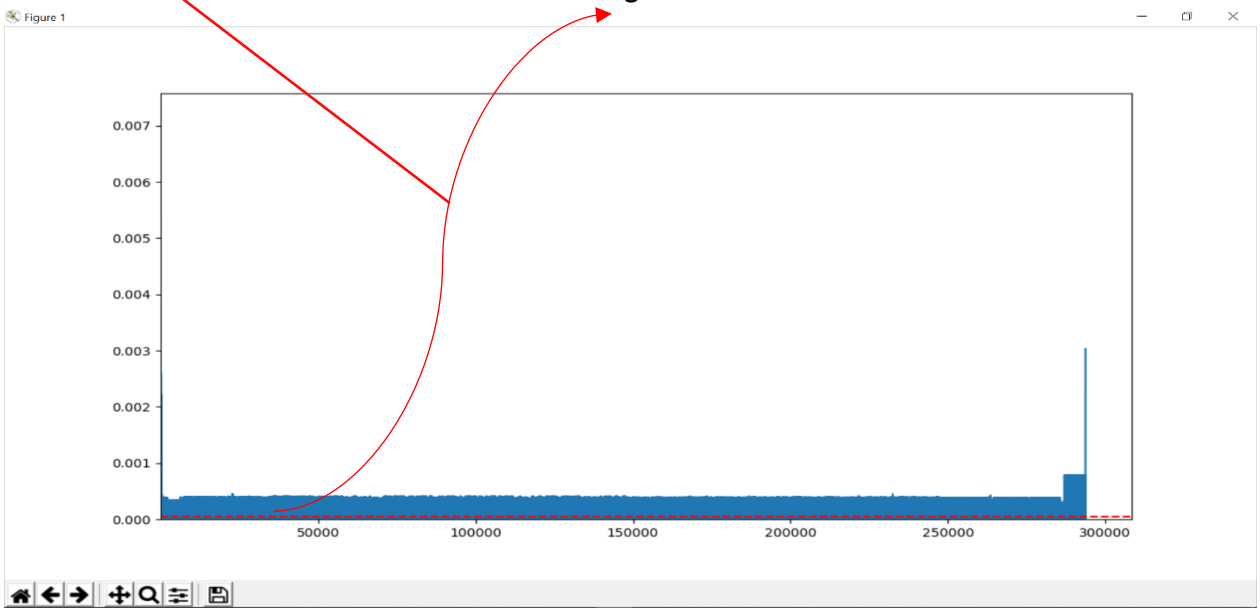
```python
import matplotlib.pyplot as plt1
toplac=[];##icerisinde tutma icin gerekli.Sonradan append ile dolduracagiz
file = open("iz.tr", "r")
pkt_id_bakilan=0 #paket idlerinin degisimi olmasi durumu kiyas setimiz olacagi icin .tr dosyasindaki ilk pkt_id sini verdim
pkt_ilk_zaman=1.10531
paket_son_zaman=0
for line in file:

  #satiri ' ' karakterleriyle bolup verilerimizi alalim
  fields = line.split(" ")
  #hocam burada ingilizce isimleri kullaniyorum mecburen kusura bakmayiniz
  time = float(fields[1])
  event = fields[0]
  pkt_id = fields[11]
#  print(float(time)+2) #burada sayisal deger basiyoruz
  if pkt_id != pkt_id_bakilan:
    toplac.append(abs(paket_son_zaman-pkt_ilk_zaman))
    pkt_id_bakilan=pkt_id
    pkt_ilk_zaman=time
  else:
    paket_son_zaman=time
#plt1.plot(toplac)
toplac.remove(toplac[toplac.index(max(toplac))])#hocam grafigi dogru cizmek icin en bastaki elemani egale ettim yani hatali degeri

print("Ortalama deger:",sum(toplac)/len(toplac))

plt1.plot(toplac)
plt1.ylim(min(toplac),max(toplac))
plt1.xlim(1)#burada 0. elementi gostermedim cunku bir ust satirda grafigi netlestirmek icin max ve min degerler arasina sikistiriyorum->
plt1.xlabel=("PAKETLER")#-> 0. element icin 0 dan 1 e cikan dikme grafigi bozmasin diye grafigi hicbirsekilde bozmuyor test edildi.
plt1.ylabel=("PAKET ISLEM SURESI")
#fig=plt1.subplot() #hocam burayi kapattim ortalamayi cizdirmek icin yaptim gormek isterseniz acabilirsiniz
#fig.axhline((sum(toplac)/len(toplac)),color='r',linestyle='--') #usttekinin devami
plt1.show()
 #Print the song
 # print(event+" | "+time+" | "+from_node+" | "+to_node+" | "+pkt_type+" | "+pkt_size+" | "+flags+" | "+fid+" | "+src_addr+" | "+dst_addr+" | "+seq_num+" | "+pkt_id)

#It is good practice to close the file at the end to free up resources
file.close()
```

Kodda yaptığım işlemlerle ilgili açıklamaları yorum olarak ekledim.Aşağıdaki grafikte görünen kırmızı kesikli ortalama çizgisini çizdirdiğim kısım **32 ve 33.satırlarda** yorum haline alındı hocam.İsterseniz açabilirsiniz o zaman tam aşağıdaki grafiğin aynısı çıkar.



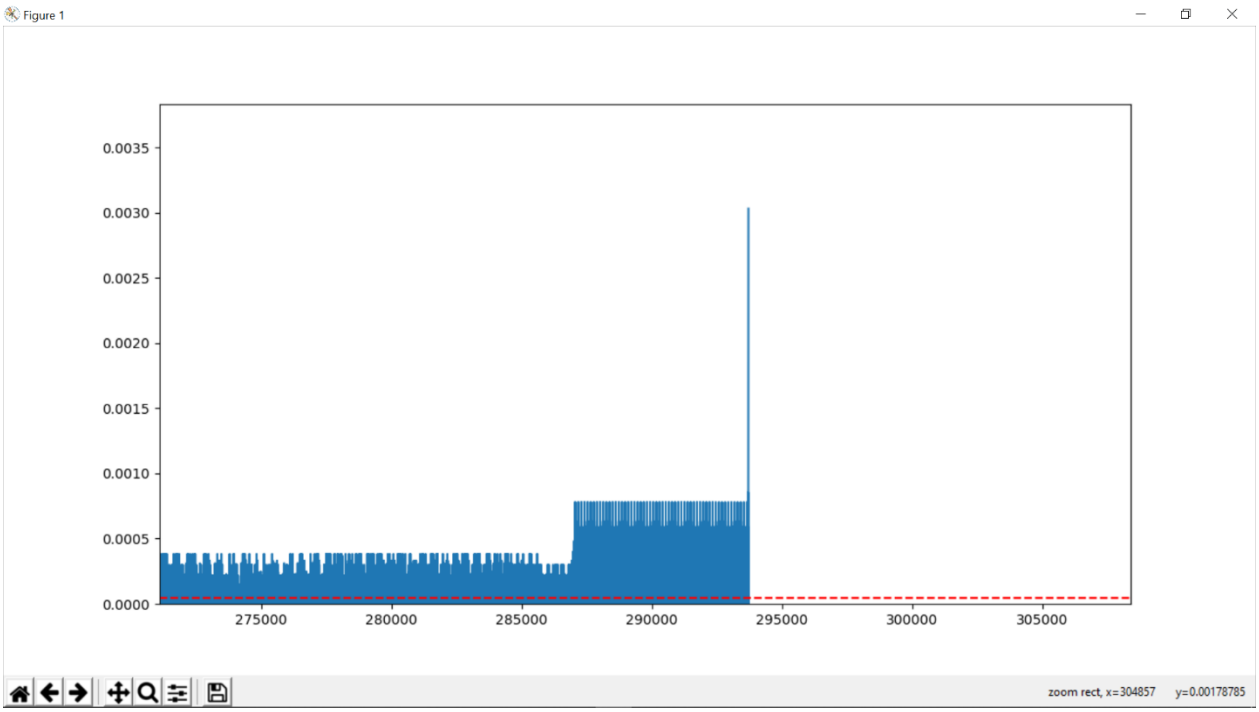
```
C:\Users\furkanyanteri\Desktop\fy\Mimar\Cozumler\deneme\son>python a.py
Ortalama deger: 4.420022607487724e-05
```

Aşağıdaki grafikte çizilen ortalama çizgisinin(kırmızı kesikli) değerini net olarak göstermek için böyle bir yol denedim hocam.**Yani ele alınan bir paketin, başka bir paket ele alınmadan yapılan işlemlerde duvardaki saatteki ilerleme miktarının ortalama değeridir.**



Furkan Yanteri

Yukarıdaki grafikten görüleceği üzere ortalama olarak 0,00004420022607487724 gibi bir değer bulunuyor.Aşağıdaki şekilde ise grafiğin sonlarda yaptığı pik kısmına bir zoomlama yaptım.Belki de ağı sonlandırma için yapılan işlemlerde tepki süresi daha fazladır sadece bir tahmin atmasyon.



5)Yansıma (TCP-ACK)

Hocam bunun için bulduğum isim yansıma.Anlatılmak istenen ise şu:
Bir TCP paketi enque edildikten sonra (event = + durumu) gelen ve alınan(event = r durumu) ilk ack paketi arasındaki duvar saati farkı.Bu ne kadar az ise o kadar iyi bir performans olduğunu varsayabiliriz.

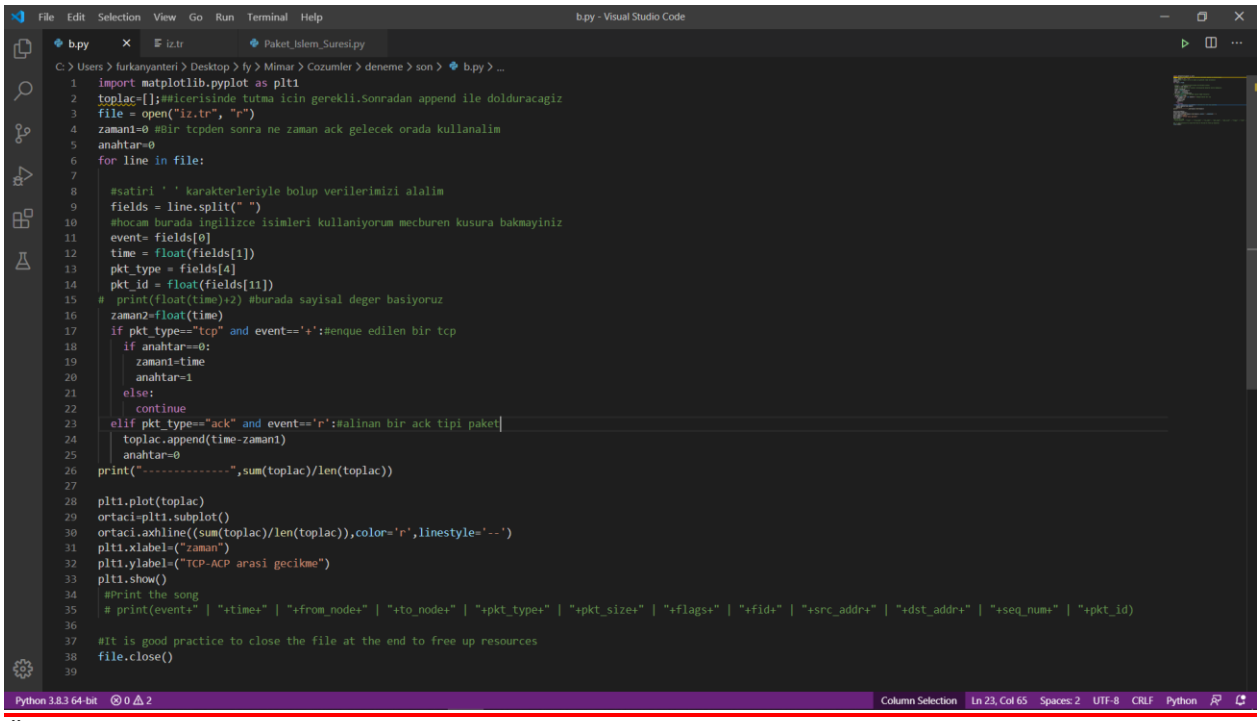
TCP, IP katmanının üzerinde çalışan ve kullanıcılarına (HTTP, SMTP, vb) paket kayıplarına karşı güvenli bir iletim ortamı sunan bağlantı temelli (connection oriented) yani iki partinin de bağlantıyı başlatıp birbirinin sağlığını gözlemlediği bir protokoldür.

İki parti arasında TCP bağlantısı 3-Way Handshake adı verilen bir proses ile başlar. 3-Way Handshake ile kurulan TCP bağlantısının parametreleri belirlenir.

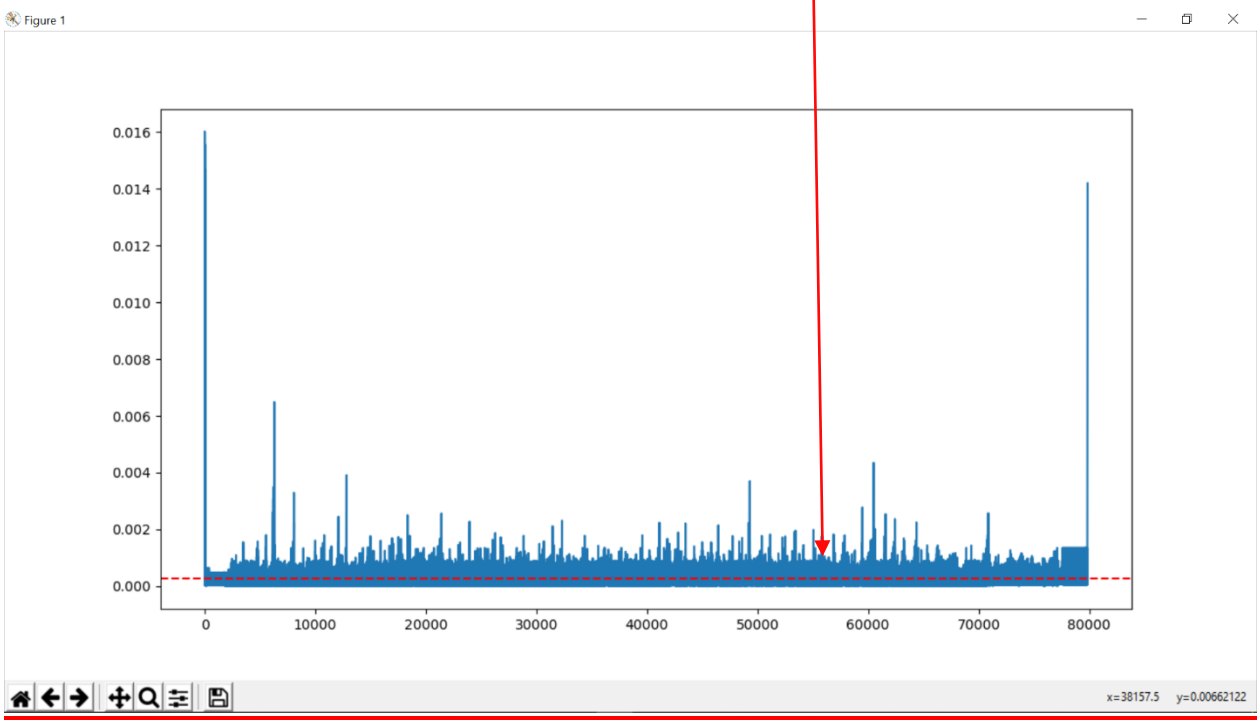
Bir ağ paketinin İstanbul'dan Avustralya'ya ulaşma (Ping) süresi yaklaşık 400ms civarındadır. Bu da Avustralya ile TCP bağlantısı kurmak için 3 paket x 400ms = 1200ms geçeceği anlamına gelir. Anlaşılacağı üzere 3-Way Handshake yeni bağlantı maliyetini yükseltir.

Ben de bundan dolayı ilk enque edilen TCP ve ilk alınan ACK arasındaki farkı bir performans kriteri olarak değerlendirdim.

Furkan Yanteri



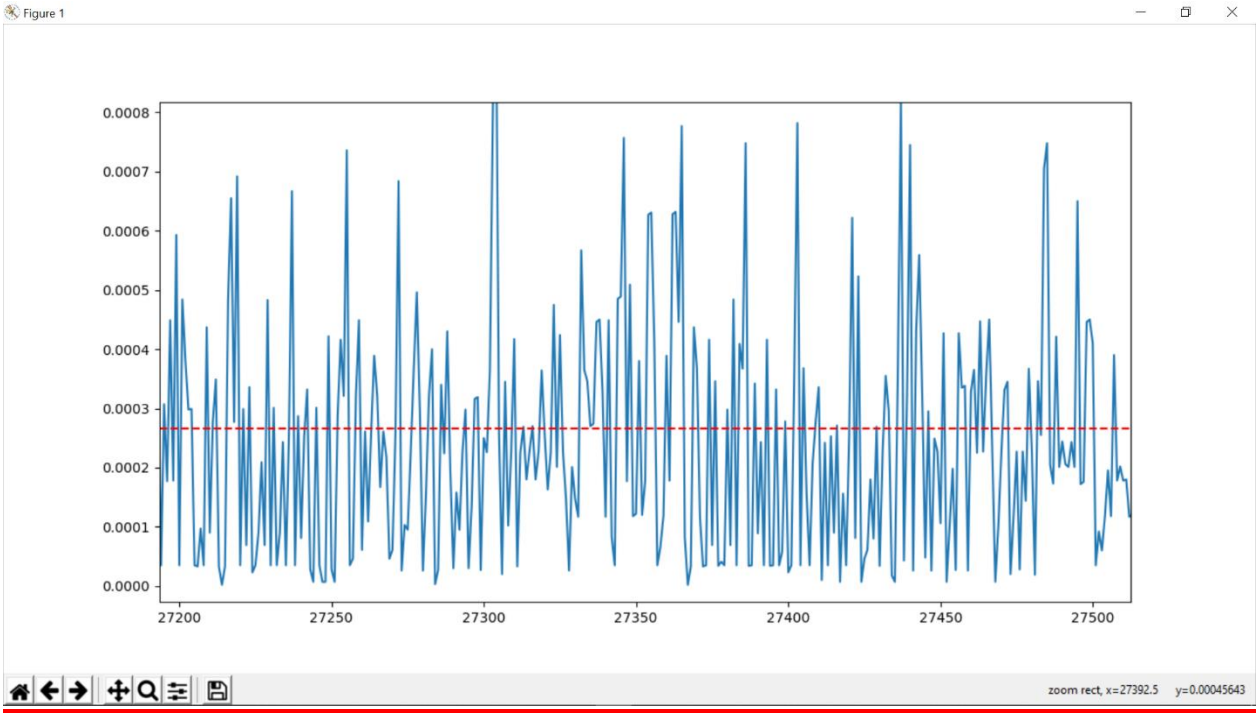
Ödevi yüklerken 29 ve 30.satırları yorum haline alıyorum ortalamayı gösteren doğru.



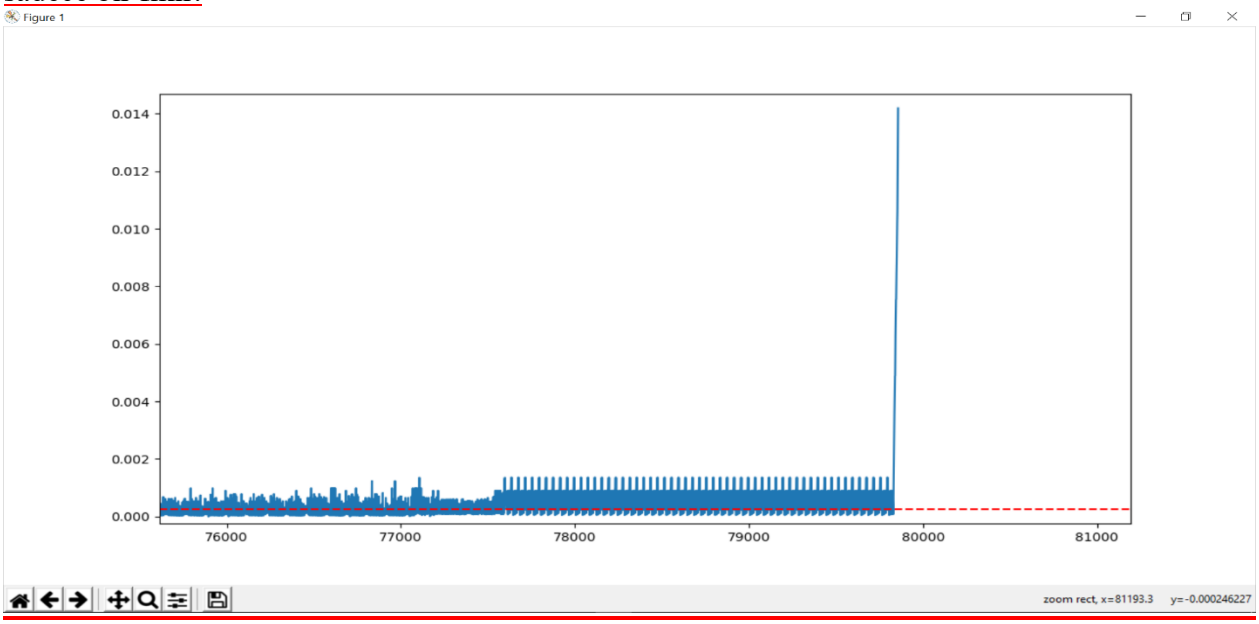
Yukarıdaki görselden hareketle şunu söyleyebiliriz.Genel olarak kararlı olmakla beraber yer yer zıplamalar mevcut.Öğrendiğime göre bu 3lü handshake protokolleriyle ilgili bir durum ve **paket paket iş görülmediği** yani paktlerin insan gözüyle ayrımsanamayacağı düzeyler olduğu için bazı bölgelerde birikmeler (**bufferized handshake**) durumundan kaynaklanıyor.Zaten bu yöntem **öncelik** hesapları sayesinde dalgalanma olsa da totalde yüksek hızlara ulaşılmasını sağlayan bir faktörmüş.

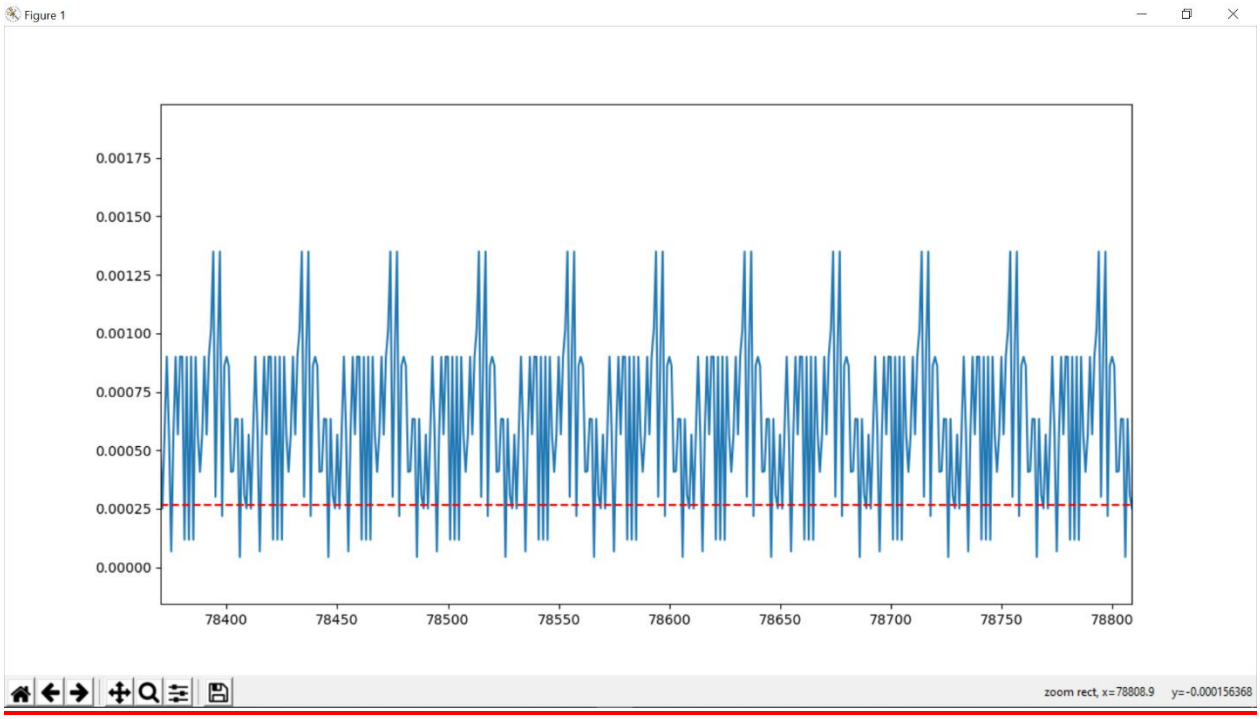
Furkan Yanteri

Aşağıdaki görselde grafiğin geneline hakim olan kararlı bir durum örneğinin zoom yapılmış hali var.Yansıma durumunun 0 olduğu durumları bilerek çıkarmadım.Bir performans metriği olarak ortalama çizgisinin grafikle olan durumu gayet güzel ve stabil duruyor(1000lik ölçekte) fakat paket düzeyine yaklaştıkça(aşağıdaki gibi) haberleşme handshake protokollerine göre normal seviyede olan bir anormallik mevcut.



Aşağıdaki görselde grafiğin son kısmına yaklaştım.Demek ki son kısımlarda yansıma durumu mükemmel derecede düzgün bir salınım yapıyor.Sebebini bilmiyorum ama son kısımda demek ki nizami bir handshake durumu olabilir veya nizami olarak bazı prosedürler gerçekleşiyor olabilir sadece bir fikir.





Son kısma yakından bir göz atalım.Buradan daha net belli oluyor ki sanki sonda **kopyala yapıştır** yapılmışçasına düzgün bir salınım var.

Kaynakça

- https://medium.com/@gokhansengun/tcp-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-1-484612c5264f
- http://shyzhou.blogspot.com/2009/06/80211-trace-file-analysis-in-ns2.html
- https://www.sciencedirect.com/topics/computer-science/packet-loss-rate
- https://ns2blogger.blogspot.com/p/awk-scripting-on-ns2.html
- https://books.google.com.tr/books?id=uIiKDQAAQBAJ&pg=SA7-PA23&lpg=SA7-PA23&dq=packet+loss+of+trace+file&source=bl&ots=EMCY-pVV9k&sig=ACfU3U1tYqvBpXDQ5NRB3vNmXpds4EC35Q&hl=tr&sa=X&ved=2ahUKEwirhff8nNfpAhX4DWMBHf05CkoQ6AEwDnoECAgQAQ#v=onepage&q=packet%20loss%20of%20trace%20file&f=false
- https://books.google.com.tr/books?id=gQqBxYCiWjwC&pg=PA259&lpg=PA259&dq=packet+loss+of+trace+file&source=bl&ots=BvkmXj0cW0&sig=ACfU3U2wwtrQRQ6Jf31yxpbyi8Rz9f-OdQ&hl=tr&sa=X&ved=2ahUKEwirhff8nNfpAhX4DWMBHf05CkoQ6AEwEHoECAwQAQ#v=onepage&q=packet%20loss%20of%20trace%20file&f=false
- https://matplotlib.org/2.1.2/api/_as_gen/matplotlib.pyplot.plot.html
- https://slogix.in/how-to-calculate-end-to-end-delay-using-awk-script-in-ns2
- https://www.researchgate.net/post/how_to_calculate_average_packet_end_to_end_delay_to_plot_graph
- https://www.w3schools.com/python