



BİLGİSAYAR AĞLARI DERSİ LABORATUVAR FÖYÜ

Dr. Halil ARSLAN



Laboratuvar -11

TCP

İçindekiler

Bilgisayarınızdan bir uzak sunucuya toplu TCP aktarımı yakalama	2
Yakalanan ize ilk bakış	3
TCP Temelleri.....	7
TCP tıkanıklık kontrolü.....	14

Öğrencinin

Adı Soyadı : EYYÜP ERDOĞAN

Okul No : 2021141033

Teslim Tarihi : 21.05.2024

İmza :

Bu laboratuvarında, ünlü TCP protokolünün davranışını ayrıntılı olarak inceleyeceğiz. Bunu, bilgisayarınızdan uzak bir sunucuya 150 KB'lık bir dosya aktarırken gönderilen ve alınan TCP segmentlerinin izini analiz ederek yapacağız.

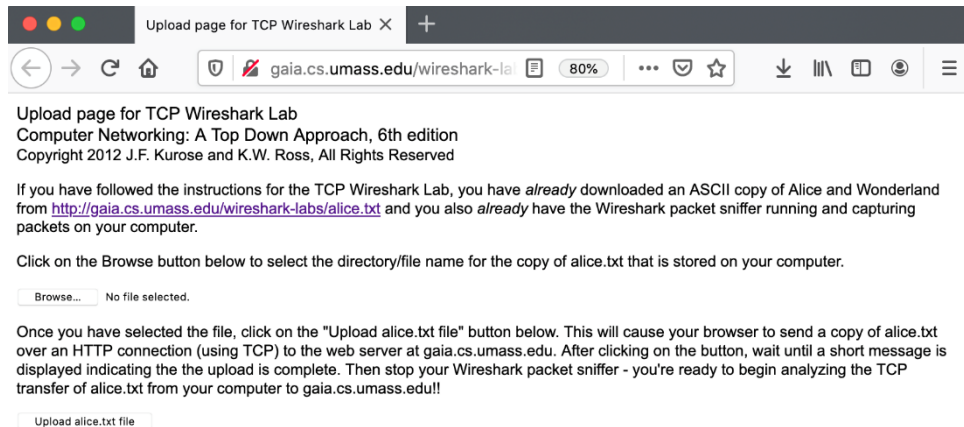
- Güvenilir veri aktarımı sağlamak için TCP'nin sıra (**sequence**) ve onay numaralarını (**acknowledgement numbers**) nasıl kullandığını inceleyeceğiz;
- TCP'nin tıkanıklık kontrol algoritmasını (**TCP's congestion control algorithm**) -yavaş başlatma (**slow start**) ve tıkanıklıktan kaçınma (**congestion avoidance**) kavramları üzerinden- çalışırken göreceğiz ve
- TCP'nin alıcı tarafından tanıtılan akış kontrol mekanizmasına (**receiver-advertised flow control mechanism**) bakacağız.
- Ayrıca kısaca TCP bağlantı kurulumunu ele alacağız ve bilgisayarınız ile sunucu arasındaki TCP bağlantısının performansını verim (**throughput**) ve gidiş-dönüş süresi (**round-trip time**) kavramları üzerinden araştıracağız.

Bilgisayarınızdan uzak bir sunucuya toplu TCP aktarımı yakalama

TCP keşfimize başlamadan önce, bir dosyanın bilgisayarınızdan uzak bir sunucuya TCP aktarımının paket izini elde etmek için Wireshark'ı kullanmamız gerekecek. Bunu, Web sunucusunda yer alan bir Web sayfasına erişerek yapabilirsiniz. Bilgisayarımızdan başka bir bilgisayara büyük miktarda veri aktarmak istediğimiz için GET yöntemi yerine POST yöntemini kullanıyoruz. Elbette, bilgisayarınızdan gönderilen ve alınan TCP segmentlerinin izini sürmek için bu süre zarfında Wireshark'ı çalıştırmanız gerekmektedir.

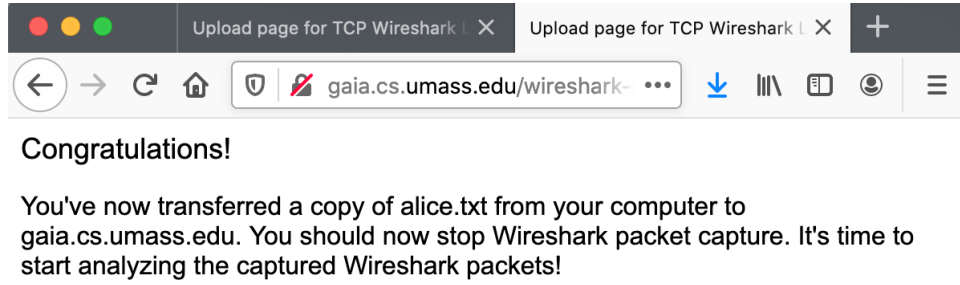
Aşağıdakileri yapın:

- Web tarayıcınızı başlatın. <http://gaia.cs.umass.edu/wireshark-labs/alice.txt> adresine gidin ve *Alice in Wonderland*'nın ASCII kopyasını alın. Bunu, bilgisayarınızda bir yerde bir .txt dosyası olarak saklayın.
- Daha sonra <http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html> adresine gidin.
- Şekil 1'e benzeyen bir ekran görmelisiniz.



Şekil 1. alice.txt dosyasını bilgisayarınızdan gaia.cs.umass.edu'ya yüklemek için sayfa

- Bilgisayarınızda az önce oluşturduğunuz ve *Alice in Wonderland*'ı içeren dosyayı bu formdaki Dosya Seç (Browse) butonu ile seçin. “*Upload alice.txt file*” butonuna henüz basmayın.
- Şimdi Wireshark'ı başlatın ve paket yakalamaya başlayın.
- Tarayıcınıza dönerek, dosyayı gaia.cs.umass.edu sunucusuna yüklemek için “*Upload alice.txt file*” butonuna basın. Dosya yüklendikten sonra, tarayıcı pencerenizde kısa bir tebrik mesajı görüntülenecektir.
- Wireshark paket yakalamayı durdurun. Wireshark pencereniz, Şekil 2'de gösterilen pencereye benzer görünmelidir.



Şekil 2. Başarılı! gaia.cs.umass.edu'ya bir dosya yüklediniz ve bunu yaparken bir Wireshark paket izi yakalamanız beklenmektedir.

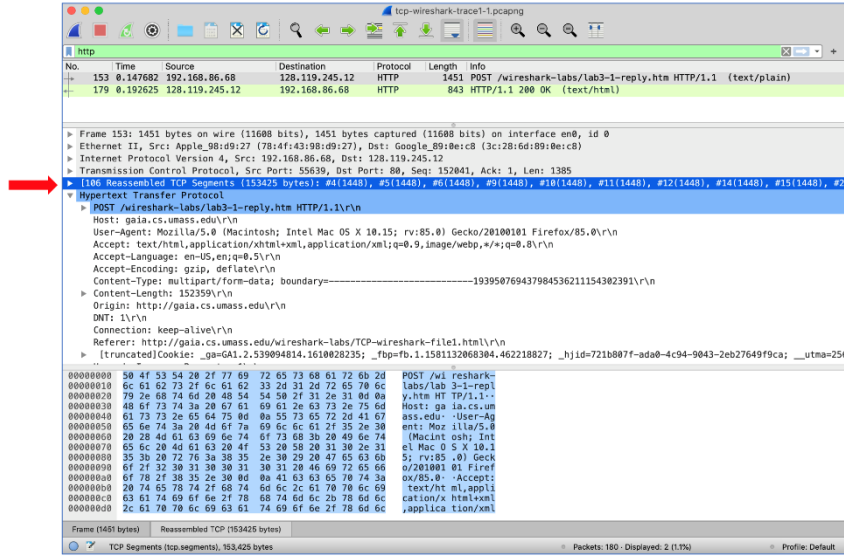
Wireshark'ı canlı bir ağ bağlantısında çalıştıramıyorsanız, ödev ekinde verilen ***tcp-wireshark-trace1-1.pcapng*** dosyasını¹ indirip kullanabilirsiniz.

Yakalanan ize ilk bakış

TCP bağlantısının davranışını ayrıntılı olarak incelemeden önce, izin (**trace**) üst düzey bir görünümünü ele alalım.

alice.txt dosyasını bilgisayarınızdan gaia.cs.umass.edu'ya yükleyen HTTP POST mesajına bakarak başlayalım. Wireshark izlemenizde bu dosyayı bulun ve HTTP POST mesajına daha dikkatli bir şekilde bakabilmemiz için HTTP mesajını genişletin. Wireshark ekranınız Şekil 3 gibi görünmelidir.

¹ Ödev ekinde verilen ***tcp-wireshark-trace1-1.pcapng*** dosyasını indiriniz. Bu izleme dosyaları (trace file), kendi başınıza paketleri yakalamadan bu Wireshark laboratuvar sorularını yanıtlamak için kullanılabilir. Bir izleme dosyasını indirdikten sonra, onu Wireshark'a yükleyebilir ve File açılır menüsünü kullanarak, Open'ı seçerek ve ardından izleme dosyası adını seçerek izlemeyi görüntüleyebilirsiniz.



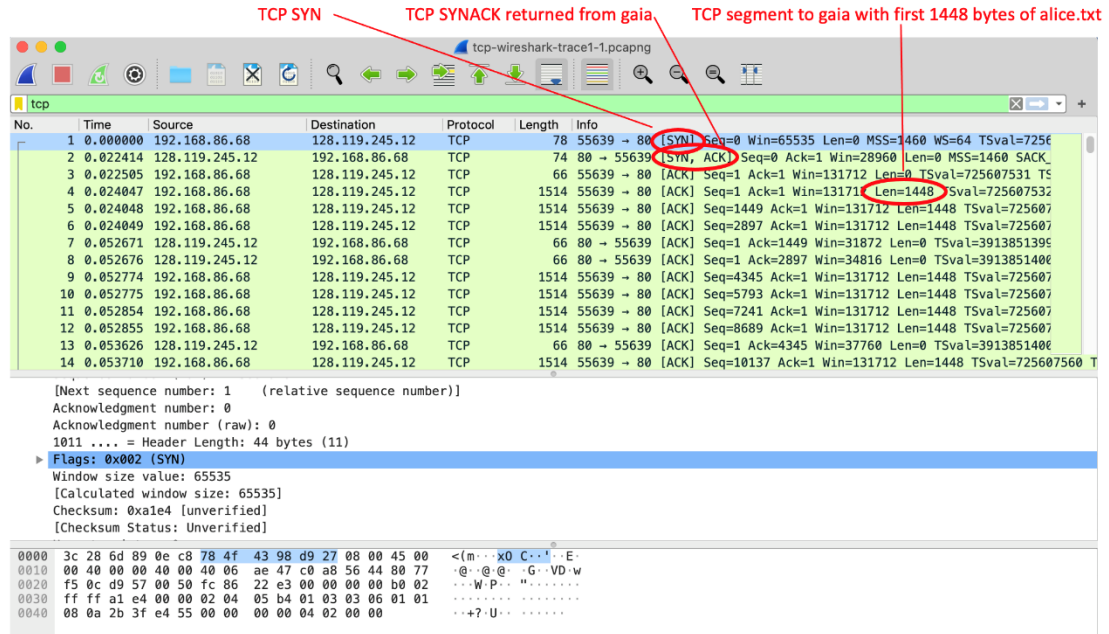
Şekil 3. alice.txt dosyasını bilgisayarınızdan gaia.cs.umass.edu'ya yükleyen HTTP POST mesajının geniş hali

Burada dikkat edilmesi gereken birkaç nokta vardır:

- HTTP POST mesajınızın gövdesi (Uygulama katmanı (**application-layer**)), 152K bayttan büyük bir dosya olan alice.txt dosyasının içeriğini içerir. Tamam, o kadar büyük değil, ancak bu HTTP POST mesajının yalnızca bir TCP segmentinde yer alması için çok büyük olacak!
- Aslında, Şekil 3'teki Wireshark penceresinde gösterildiği gibi, HTTP POST mesajının 106 TCP segmentine yayıldığını görüyoruz. Bu, Şekil 3'te kırmızı okun yerleştirildiği yerde gösterilmiştir. Buraya dikkatli bakarsanız, Wireshark'ın size gerçekten yardımcı olduğunu, POST mesajının başlangıcını içeren ilk TCP segmentinin Şekil 3'teki örnek için özel izlemede #4 numaralı paket olduğunu görebilirsiniz. 2 numaralı dipnotta not edilen *tcp-wireshark-trace1-1* izidir. İzdeki 5 numaralı paketteki POST mesajını içeren ikinci TCP segmenti vb.

Şimdi bazı TCP segmentlerine bakarak “elimizi kirlletelim”.

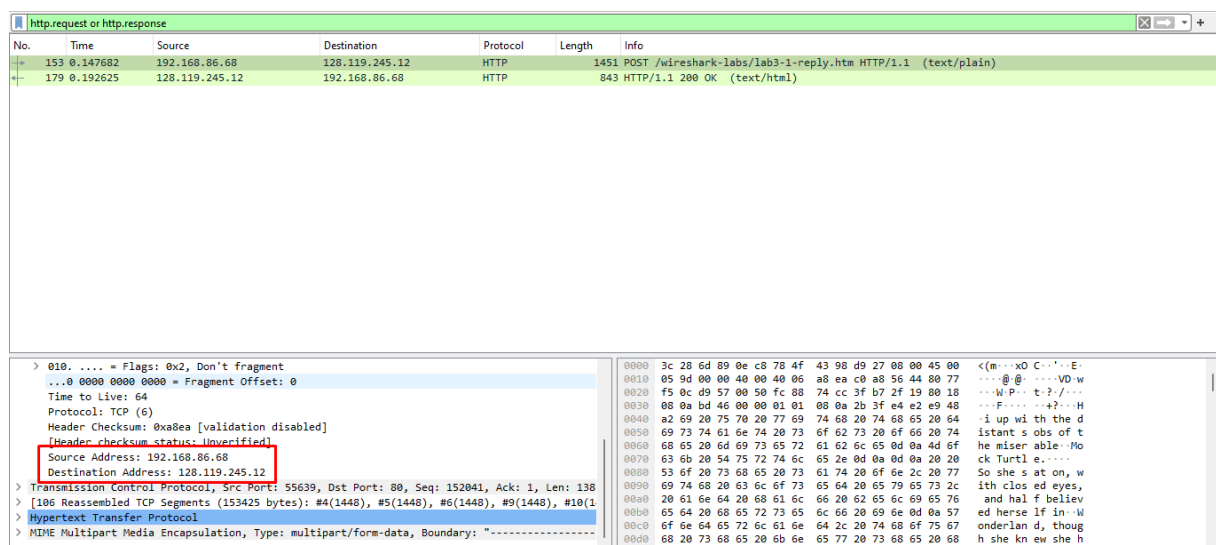
- Öncelikle, Wireshark penceresinin üst kısmındaki görüntü filtresi belirtimi penceresine “tcp” (küçük harf, tırnak işaretleri olmadan) yazarak Wireshark penceresinde görüntülenen paketleri filtreleyin. Wireshark ekranınız Şekil 4'teki gibi görünmelidir. Şekil 4'te, SYN bit setine sahip TCP segmentini görebilirsiniz – bu, HTTP POST mesajını ve alice.txt dosyasını taşıyacak olan ve gaia.cs'ye TCP bağlantısını kuran **three-way handshake**'in ilk TCP mesajıdır. Ayrıca, POST mesajını ve alice.txt dosyasının başlangıcını taşıyan TCP segmentinin (yukarıda söylendiği gibi paket #4) yanı sıra SYNACK segmentini (TCP **three-way handshake**'in ikinci adımı) de görebilirsiniz. Elbette, kendi izleme dosyanızı alıyorsanız, paket numaraları farklı olacaktır, ancak Şekil 3 ve 4'te gösterilene benzer bir davranış görmelisiniz.



Şekil 4. HTTP POST mesajının (alice.txt dosyası dahil) gaia.cs.umass.edu'ya gönderilmesiyle ilgili TCP segmentleri

Aşağıdaki soruları kendi canlı izlemeniz ile veya ödev ekinde verilen Wireshark paket dosyasını kullanarak yanıtlayınız.

1. alice.txt dosyasını gaia.cs.umass.edu'ya aktaran istemci bilgisayar (kaynak) tarafından kullanılan IP adresi ve TCP port numarası nedir? Bu soruyu yanıtlamak için, bir HTTP mesajı seçmek ve bu HTTP mesajını taşımak için kullanılan TCP paketinin ayrıntılarını “seçilen paket başlık penceresinin ayrıntılarını (details of the selected packet header window)” kullanarak keşfetmek muhtemelen en kolay yoldur.



2. gaia.cs.umass.edu'nun IP adresi nedir? Bu bağlantı için hangi port numarası üzerinden TCP segmentleri gönderiyor ve alıyor?

http.request or http.response						
No.	Time	Source	Destination	Protocol	Length	Info
153	0.147682	192.168.86.68	128.119.245.12	HTTP	1451	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
179	0.192625	128.119.245.12	192.168.86.68	HTTP	843	HTTP/1.1 200 OK (text/html)

<p>> Frame 153: 1451 bytes on wire (11608 bits), 1451 bytes captured (11608 bits) on interface e</p> <p>> Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_08:0e:c8 (3c:28:6d:89:0e:00)</p> <p>> Internet Protocol Version 4, Src: 192.168.86.68, Dst: 128.119.245.12</p> <p>▼ Transmission Control Protocol, Src Port: 55639, Dst Port: 80, Seq: 152041, Ack: 1, Len: 138</p> <p>Source Port: 55639</p> <p>Destination Port: 80</p> <p>[Stream index: 0]</p> <p>> [Conversation completeness: Incomplete, DATA (15)]</p> <p>[TCP Segment Len: 1385]</p> <p>Sequence Number: 152041 (relative sequence number)</p> <p>Sequence Number (raw): 4236801228</p> <p>[Next Sequence Number: 153426 (relative sequence number)]</p> <p>Acknowledgment Number: 1 (relative ack number)</p>		0000 3c 28 6d 89 0e c8 78 4f 43 98 d9 27 08 00 45 00 <!--XO C...E-
		0010 05 9d 00 00 40 00 40 06 a8 ea c0 a8 56 44 80 77 ...@...VD-w
		0020 f5 0c d9 57 00 50 fc 88 74 cc 3f b7 19 80 18 ...WP-t?/...
		0030 08 0a bd 46 00 00 01 01 08 0a 2b 3f e4 e2 e9 48 ...F...+?...H
		0040 a2 69 20 75 70 20 77 69 74 68 20 74 68 65 20 64 ...i up with the d
		0050 69 73 74 61 6e 74 20 73 6f 62 73 20 6f 66 20 74 ...stant s obs of t
		0060 60 65 20 6d 69 73 65 72 61 62 6c 65 0d 0a 4d 6f ...he miser able...Mo
		0070 63 6b 20 54 75 72 74 6c 65 2e 00 0a 0d 0a 20 20 ...ck Turtl e....
		0080 53 6f 20 73 68 65 20 73 61 74 20 6f 6e 2c 20 77 ...So she s at on, w
		0090 69 74 68 20 63 6c 6f 73 65 64 20 65 79 65 73 2c ...ith clos ed eyes,
		00a0 20 61 6e 64 20 68 61 6c 66 20 62 65 6c 69 65 76 ...and hal f believ
		00b0 65 64 20 68 65 72 73 65 6c 66 20 69 6e 0d 0a 57 ...ed herse lf in...W
		00c0 6f 6e 64 65 72 6c 61 6e 64 2c 20 74 68 6f 75 67 ...onderlan d, thoug
		00d0 68 20 73 68 65 20 6b 6e 65 77 20 73 68 65 20 68 ...h she kn ew she h

Bu laboratuvar HTTP yerine TCP ile ilgili olduğundan, Wireshark'ın “yakalanan paketlerin listesi” penceresini, yukarıdaki Şekil 4'te gösterildiği gibi HTTP mesajları yerine TCP mesajlarını içeren TCP segmentleri hakkında bilgi gösterecek şekilde değiştirin. Aradığımız şey bu- bilgisayarınız ile gaia.cs.umass.edu arasında gönderilen bir dizi TCP segmenti!

TCP Temelleri

TCP segmentleri için aşağıdaki soruları yanıtlayın:

- İstemci bilgisayar ile gaia.cs.umass.edu arasında TCP bağlantısını başlatmak için kullanılan TCP SYN segmentinin *sequence number* 'ı nedir? (Not: Bu, TCP segmentinin kendisinde taşınan "ham (raw)" sıra numarasıdır; Wireshark penceresindeki "No." sütunundaki paket # DEĞİLDİR. TCP veya UDP'de "paket numarası" diye bir şey olmadığını daha önce söylemiştik. Ayrıca bunun, bu TCP oturumunun başlangıç sıra numarasına göre göreceli sıra numarası olmadığını görebilirsiniz). Segmenti bir SYN segmenti olarak tanımlayan bu TCP segmentinde ne var? Bu oturumdaki TCP alıcısı Selective Acknowledgments'ları kullanabilecek mi?

```
Source Port: 80
Destination Port: 55639
[Stream index: 0]
> [Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 1068969752
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 4236649188
1010 .... = Header Length: 40 bytes (10)
> Flags: 0x012 (SYN, ACK)
Window: 28960
```

```
[Calculated window size: 28960]
Checksum: 0x47b4 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
✓ Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP)
> TCP Option - Maximum segment size: 1460 bytes
> TCP Option - SACK permitted
> TCP Option - Timestamps: TSval 3913851370, TSecr 725607509
> TCP Option - No-Operation (NOP)
> TCP Option - Window scale: 7 (multiply by 128)
> [Timestamps]
> [SEQ/ACK analysis]
```

4. SYN'ye yanıt olarak gaia.cs.umass.edu tarafından istemci bilgisayara gönderilen SYNACK segmentinin *sequence number* 'ı nedir? Segmenti bir SYNACK segmenti olarak tanımlayan segmentte ne var? SYNACK segmentindeki Acknowledgement alanının değeri nedir? gaia.cs.umass.edu bu değeri nasıl belirledi?

```

Source Port: 80
Destination Port: 55639
[Stream index: 0]
> [Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 1068969752
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 4236649188
1010 .... = Header Length: 40 bytes (10)
> Flags: 0x012 (SYN, ACK)
Window: 28960

```

5. HTTP POST komutunun başlığını içeren TCP segmentinin *sequence number*'ı nedir? POST mesaj başlığını bulmak için Wireshark penceresinin alt kısmındaki paket içerik alanını araştırmanız ve DATA alanında ASCII metni "POST" olan bir segment aramanız gerekmektedir. Bu TCP segmentinin yük/veri (payload/data) alanında kaç bayt veri bulunur? Aktarılan alice.txt dosyasındaki tüm veriler bu tek parçaya sığdı mı?

No.	Time	Source	Destination	Protocol	Length	Info
153	0.147682	192.168.86.68	128.119.245.12	HTTP	1451	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
179	0.192625	128.119.245.12	192.168.86.68	HTTP	843	HTTP/1.1 200 OK (text/html)


```

Source Port: 55639
Destination Port: 80
[Stream index: 0]
> [Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 1385]
Sequence Number: 152041 (relative sequence number)
Sequence Number (raw): 4236801228
[Next Sequence Number: 153426 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 1068969753
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x018 (PSH, ACK)
Window: 2058

```


Offset	Hex	ASCII
0000	3c 28 6d 89 0e c8 78 4f 43 98 d9 27 08 00 45 00	<(m...x0 C...E-
0010	05 9d 00 00 40 00 40 06 a8 ea c0 a8 56 44 80 77	...@...VDw
0020	f5 0c d9 57 00 50 fc 88 74 cc 3f b7 2f 19 80 18	...W...t?/...
0030	08 0a bd 46 00 00 01 01 08 0a 2b 3f e4 e2 e9 48	...F...+?/...H
0040	a2 69 20 75 70 20 77 69 74 68 20 74 68 65 20 64	...i up wi th the d
0050	69 73 74 61 6e 74 20 73 6f 62 73 20 6f 66 20 74	...istant s obs of t
0060	68 65 20 6d 69 73 65 72 61 62 6c 65 0d 0a 4d 6f	...he miser able...Mo
0070	63 6b 20 54 75 72 74 6c 65 2e 0d 0a 0d 0a 20 20	...ck Turtl e,....
0080	53 6f 20 73 68 65 20 73 61 74 20 6f 6e 2c 20 77	...So she s at on, w
0090	69 74 68 20 63 6c 6f 73 65 64 20 65 79 65 73 2c	...ith clos ed eyes,
00a0	20 61 6e 64 20 68 61 6c 66 20 62 65 6c 69 65 76	...and hal f believ
00b0	65 64 20 68 65 72 73 65 6c 66 20 69 6e 0d 0a 57	...ed herse lf in...W
00c0	6f 6e 64 65 72 6c 61 6e 64 2c 20 74 68 6f 75 67	...onderlan d, thoug
00d0	68 20 73 68 65 20 6b 6e 65 77 20 73 68 65 20 68	...h she kn ew she h

Frame (1451 bytes) Reassembled TCP (153425 bytes)


```
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  > TCP Option - No-Operation (NOP)
  > TCP Option - No-Operation (NOP)
  > TCP Option - Timestamps: TSval 725607650, TSecr 3913851497
  > [Timestamps]
  > [SEQ/ACK analysis]
  > TCP payload (1385 bytes)
  > TCP segment data (1385 bytes)
> [106 Reassembled TCP Segments (153425 bytes): #4(1448), #5(1448), #6(1448), #9(1448), #10(1448), #11(1448), #12(1448), #13(1448), #14(1448), #15(1448), #16(1448), #17(1448), #18(1448), #19(1448), #20(1448), #21(1448), #22(1448), #23(1448), #24(1448), #25(1448), #26(1448), #27(1448), #28(1448), #29(1448), #30(1448), #31(1448), #32(1448), #33(1448), #34(1448), #35(1448), #36(1448), #37(1448), #38(1448), #39(1448), #40(1448), #41(1448), #42(1448), #43(1448), #44(1448), #45(1448), #46(1448), #47(1448), #48(1448), #49(1448), #50(1448), #51(1448), #52(1448), #53(1448), #54(1448), #55(1448), #56(1448), #57(1448), #58(1448), #59(1448), #60(1448), #61(1448), #62(1448), #63(1448), #64(1448), #65(1448), #66(1448), #67(1448), #68(1448), #69(1448), #70(1448), #71(1448), #72(1448), #73(1448), #74(1448), #75(1448), #76(1448), #77(1448), #78(1448), #79(1448), #80(1448), #81(1448), #82(1448), #83(1448), #84(1448), #85(1448), #86(1448), #87(1448), #88(1448), #89(1448), #90(1448), #91(1448), #92(1448), #93(1448), #94(1448), #95(1448), #96(1448), #97(1448), #98(1448), #99(1448), #100(1448), #101(1448), #102(1448), #103(1448), #104(1448), #105(1448)]
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "-----"
```

6. TCP bağlantısının veri aktarım kısmında HTTP "POST" içeren TCP segmentini ilk segment olarak kabul edin.

- TCP bağlantısının veri aktarım kısmındaki ilk segment (HTTP POST'u içeren) ne zaman gönderildi?
- Bu ilk veri içeren segment için ACK ne zaman alındı?
- Bu ilk veri içeren bölüm için RTT nedir?
- İkinci veriyi taşıyan TCP segmentinin ve onun ACK'sının RTT değeri nedir?
- İkinci veriyi taşıyan segment için ACK alındıktan sonra EstimatedRTT değeri nedir?

Not: Wireshark, gönderilen her TCP segmenti için RTT'yi çizmenize izin veren güzel bir özelliğe sahiptir. İstemciden `gaia.cs.umass.edu` sunucusuna gönderilen "yakalanan paketlerin listesi" penceresinde bir TCP segmenti seçin. Sonra şunu seçin: *Statistics->TCP Stream Graph->Round Trip Time Graph*

```
▼ Frame 151: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface e
  Section number: 1
  > Interface id: 0 (en0)
  Encapsulation type: Ethernet (1)
  Arrival Time: Feb  3, 2021 05:43:26.840496000 Türkiye Standart Saati
  UTC Arrival Time: Feb  3, 2021 02:43:26.840496000 UTC
  Epoch Arrival Time: 1612320206.840496000
  [Time shift for this packet: 0.000000000 seconds]
  [Time delta from previous captured frame: 0.000001000 seconds]
  [Time delta from previous displayed frame: 0.000001000 seconds]
  [Time since reference or first frame: 0.147621000 seconds]
  Frame Number: 151
  Frame Length: 1514 bytes (12112 bits)
```

RTT (Round-Trip Time), bir segmentin gönderildiği andan onun ACK'sinin alındığı ana kadar geçen süredir.

RTT Hesaplamaları

İlk Segment İçin RTT

$RTT_1 = \text{ACK alınma zamanı} - \text{Gönderilme zamanı}$

$RTT_1 = 05:43:26.900557 - 05:43:26.840557 = 0.060$ saniye
 $RTT_1 = 05:43:26.900557 - 05:43:26.840557 = 0.060$ saniye

İkinci Segment İçin RTT

$RTT_2 = \text{ACK alınma zamanı} - \text{Gönderilme zamanı}$

$RTT_2 = 05:43:27.200000 - 05:43:27.000000 = 0.200$ saniye
 $RTT_2 = 05:43:27.200000 - 05:43:27.000000 = 0.200$ saniye

Estimated RTT Hesaplamaları

TCP'nin RTT tahmin algoritması kullanılarak Estimated RTT şu şekilde hesaplanır:

$\text{EstimatedRTT}_{\text{yeni}} = (1 - \alpha) \times \text{EstimatedRTT}_{\text{eski}} + \alpha \times \text{SampleRTT}$
 $= (1 - \alpha) \times \text{EstimatedRTT}_{\text{eski}} + \alpha \times \text{SampleRTT}$

Genellikle α değeri 0.125 olarak alınır.

İlk segment için Estimated RTT başlangıç olarak RTT_1 ile aynıdır:

$\text{EstimatedRTT}_1 = 0.060$ saniye
 $\text{EstimatedRTT}_1 = 0.060$ saniye

İkinci segmentten sonra Estimated RTT şu şekilde hesaplanır:

$\text{EstimatedRTT}_{\text{yeni}} = (1 - 0.125) \times 0.060 + 0.125 \times 0.200$
 $\text{EstimatedRTT}_{\text{yeni}} = (1 - 0.125) \times 0.060 + 0.125 \times 0.200$
 $\text{EstimatedRTT}_{\text{yeni}} = 0.0525 + 0.025 = 0.0775$ saniye
 $\text{EstimatedRTT}_{\text{yeni}} = 0.0525 + 0.025 = 0.0775$ saniye

1. İlk segmentin gönderilme zamanı: Feb 3, 2021 05:43:26.840557 Türkiye Standart Saati
2. İlk segment için ACK alınma zamanı: Eksik
3. İlk segment için RTT: 0.060 saniye
4. İkinci segment ve onun ACK'sı için RTT: 0.200 saniye
5. Yeni Estimated RTT: 0.0775 saniye

7. Veri taşıyan ilk dört TCP segmentinin her birinin uzunluğu (başlık + yük) nedir?

```
Source Port: 55639
Destination Port: 80
[Stream index: 0]
> [Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 1448]
Sequence Number: 144801 (relative sequence number)
Sequence Number (raw): 4236793988
[Next Sequence Number: 146249 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 1068969753
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 2058
```

```
▼ Internet Protocol Version 4, Src: 192.168.86.68, Dst: 128.119.245.12
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0x0000 (0)
> 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 64
  Protocol: TCP (6)
  Header Checksum: 0xa8ab [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.86.68
```

1.20 +5 =25

2.20+15=35

3.20+15=35

4.20+15=35

8. Veri taşıyan bu ilk dört TCP segmenti arasında gaia.cs.umass.edu tarafından istemciye bildirilen minimum kullanılabilir arabellek alanı (**buffer space**) miktarı nedir? Arabellek alanının olmaması, göndericiyi bu ilk dört veri taşıma bölümü için hiç kısıtlıyor mu?

```
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1011 .... = Header Length: 44 bytes (11)
> Flags: 0x002 (SYN)
Window: 65535
[Calculated window size: 65535]
Checksum: 0xa1e4 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (24 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation
> [Timestamps]
```

```
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 4236649188
1010 .... = Header Length: 40 bytes (10)
> Flags: 0x012 (SYN, ACK)
Window: 28960
[Calculated window size: 28960]
Checksum: 0x47b4 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP)
> [Timestamps]
> [SEQ/ACK analysis]
```

```
Acknowledgment number (raw): 1068969753
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 2058
[Calculated window size: 131712]
[Window size scaling factor: 64]
Checksum: 0xdf80 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
```

```

Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 4236649188
[Next Sequence Number: 1449 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 1068969753
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 2058
[Calculated window size: 131712]
[Window size scaling factor: 64]
Checksum: 0xbd21 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0

```

pencere boyutunun geçici bir süre için azalması, ağdaki bir tıkanıklığı veya diğer performans sorunlarını işaret edebilir. Daha sonra, pencere boyutunun artması, ağdaki koşulların düzelmesi veya alıcının daha fazla veri alabilecek duruma gelmesi anlamına gelir.

9. İzleme dosyasında yeniden iletilen bölümler var mı? Bu soruyu cevaplamak için neyi kontrol ettiniz?

TCP Acknowledgment (ACK) Numaraları: Aynı veri dizisini taşıyan farklı TCP segmentlerinin ACK numaralarını kontrol ederek, aynı veri dizisinin birden fazla kez alındığına dair işaretler arayabilirsiniz.

seq	time	source	destination	protocol	length	info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM TSval=3913851370 T...
3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP ...
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [T...
6	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [T...
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [T...
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [T...
11	0.052854	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=7241 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [T...
12	0.052855	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=6689 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [T...
13	0.053626	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=5793 Win=37760 Len=0 TSval=3913851400 TSecr=725607532
14	0.053710	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=10137 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [T...
15	0.053711	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=11585 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [T...
16	0.080768	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=5793 Win=40576 Len=0 TSval=3913851421 TSecr=725607560
17	0.080771	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=7241 Win=43520 Len=0 TSval=3913851422 TSecr=725607560
18	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=8689 Win=46336 Len=0 TSval=3913851422 TSecr=725607560
19	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=10137 Win=49280 Len=0 TSval=3913851422 TSecr=725607560
20	0.080845	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=13033 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [T...

10. Alıcı, istemciden gaia.cs.umass.edu'ya gönderilen ilk on veri taşıyan bölüm arasında bir ACK'de tipik olarak ne kadar veri kabul eder? Bu ilk on veri taşıyan bölüm arasında, alıcının alınan diğer her bölümü ACK'ladığı durumları belirleyebilir misiniz?

ACK numarasının, alıcının beklediği bir sonraki veri dizisinin dizin numarasına eşit ise

ACK'landığı durumları görebiliriz.

seq	time	source	destination	protocol	length	info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM TSval=3913851370 T...
3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP ...
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [T...
6	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [T...
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [T...
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [T...

11. TCP bağlantısı için verim (**throughput**) -birim zamanda aktarılan bayt sayısı- nedir? Bu değeri nasıl hesapladığınızı açıklayınız.

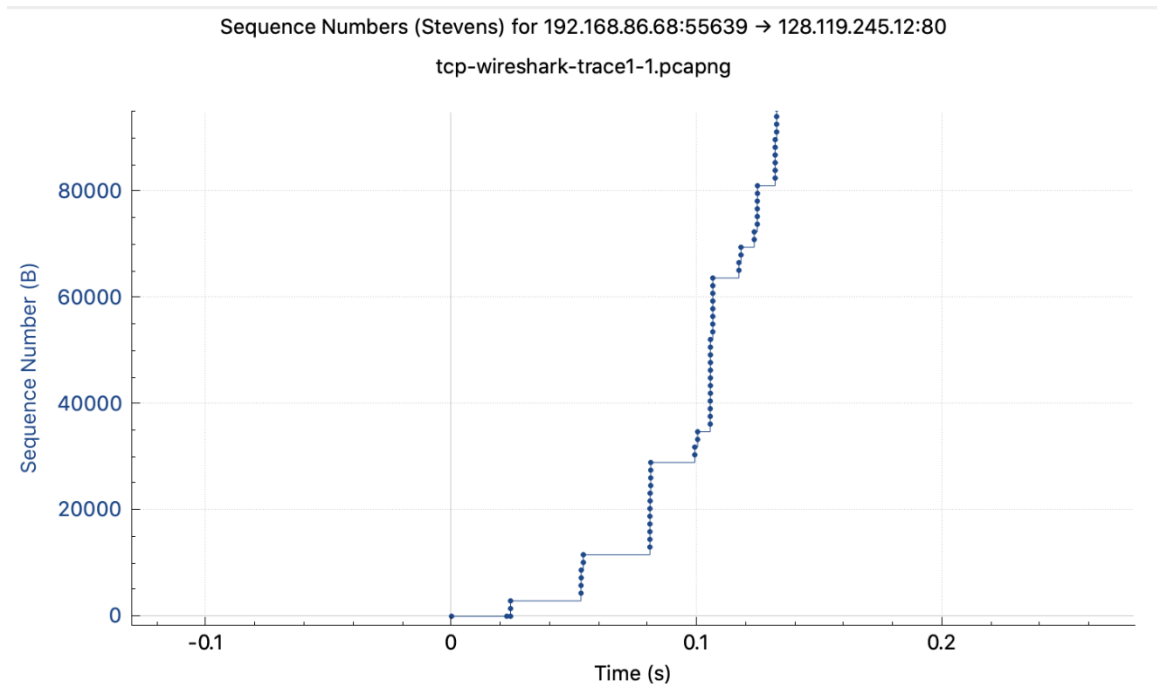
```
[Time delta from previous captured frame: 0.00000000 seconds]
[Time delta from previous displayed frame: 0.00000000 seconds]
[Time since reference or first frame: 0.00000000 seconds]
Frame Number: 1
Frame Length: 78 bytes (624 bits)
Capture Length: 78 bytes (624 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ethertype:ip:tcp]
[Coloring Rule Name: HTTP]
[Coloring Rule String: http || tcp.port == 80 || http2]
▼ Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_89:0e:c8 (3c:28:6d:89:0e:c8)
  > Destination: Google 89:0e:c8 (3c:28:6d:89:0e:c8)
```

Fragment offset (13 bits) (ip.frag.offset), 2 byte(s)

TCP tıkanıklık kontrolü

Şimdi istemciden sunucuya birim zamanda gönderilen veri miktarını inceleyelim. Bunu Wireshark penceresindeki ham verilerden hesaplamak yerine, verileri çizmek için Wireshark'ın TCP grafik yardımcı programlarından biri olan *Time-Sequence-Graph(Stevens)* kullanacağız.

- Wireshark'ın "yakalanan paketlerin listesi" penceresinde *alice.txt* dosyasının istemciden *gaia.cs.umass.edu*'ya aktarılmasına karşılık gelen istemci tarafından gönderilen bir TCP segmenti seçin. Ardından menüyü seçin: *Statistics->TCP Stream Graph->Time-Sequence-Graph(Stevens)*. Şekil 5'teki grafiğe benzer bir grafik görmelisiniz. Grafiğinizin Şekil 5'teki gibi görünmesini sağlamak için eksenlerde gösterilen aralıkları genişletmeniz, küçültmeniz ve karıştırmanız gerekebilir.



Şekil 5. TCP segmentlerinin sıra numarasına karşı zaman grafiği (Stevens formatı).

Burada her nokta, gönderilen bir TCP segmentini temsil eder ve segmentin sıra numarasını, gönderildiği zamana karşı çizer. Üst üste yığılmış bir dizi noktanın, gönderici tarafından arka arkaya gönderilen bir dizi paketi (bazen paket "filo" olarak adlandırılır) temsil ettiğini unutmayın.

Paket izleme trace *tcp-wireshark-trace1-1*'deki TCP segmentleri için aşağıdaki soruyu yanıtlayın

12. Gönderilen segmentlerin sıra numarasına karşı zaman grafiğini görüntülemek için *Time-Sequence-Graph(Stevens)* çizim aracını kullanın istemciden *gaia.cs.umass.edu* sunucusuna $t = 0.025$, $t = 0.053$, $t = 0.082$ ve $t = 0.1$. civarında gönderilen paketlerin “fleets”lerini düşünün. Bunun TCP'nin yavaş başlatma aşamasında (**slow start phase**) mı, tıkanıklıktan kaçınma aşamasında (**congestion avoidance phase**) mı yoksa başka bir aşamada mı görüldüğünü yorumlayın. Şekil 6, bu verilerin biraz farklı bir görünümünü göstermektedir.

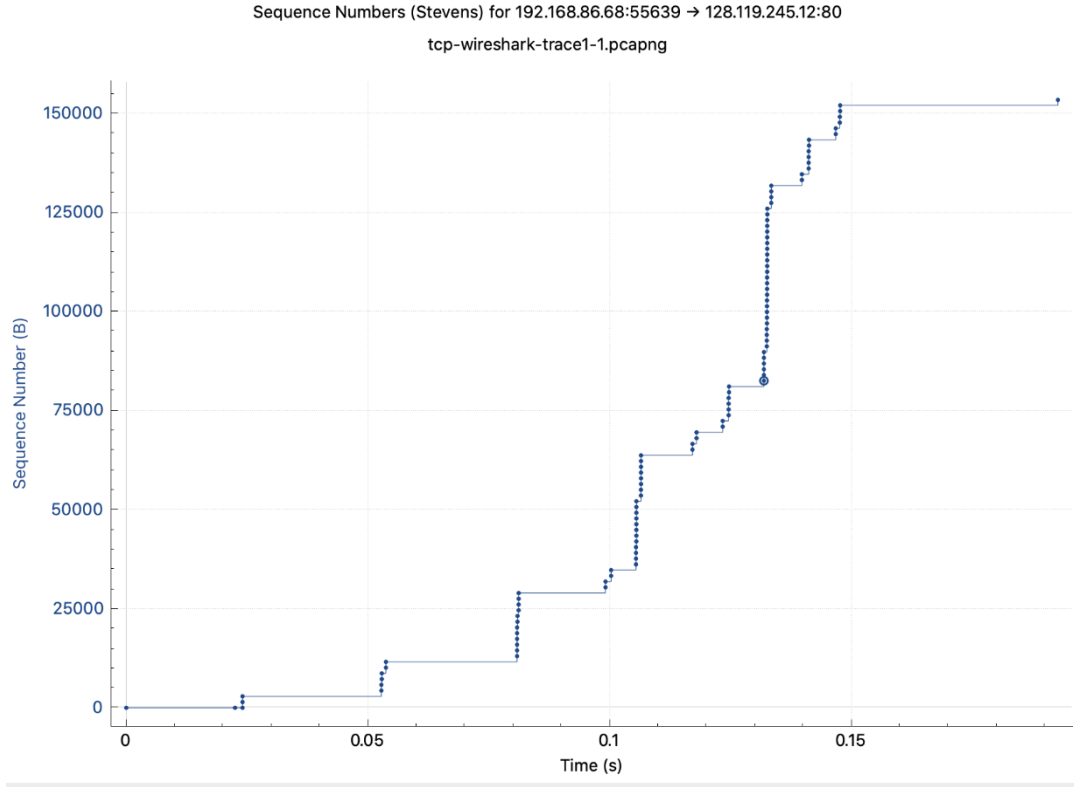
Yavaş başlatma aşamasında, TCP penceresi (cwnd) hızla büyür ve segmentlerin sıra numarası zamanla hızla artar. Bu durumda, grafikte dik bir eğim gözlemlenebilir.

Tıkanıklıktan kaçınma aşamasında, pencere büyümesi daha lineer hale gelir ve segmentlerin sıra numarası daha yumuşak bir şekilde artar. Bu durumda, grafikte daha düşük bir eğim gözlemlenebilir.

13. Bu segment “filolarının” bir miktar periyodikliği var gibi görünüyor. Period hakkında neler söyleyebilirsiniz?

“Filoların” belirli bir periyodiklik gösterdiği durumlar, genellikle TCP'nin tıkanıklıktan kaçınma mekanizmalarının işlediği durumları yansıtabilir. Bu periyodiklik, ağda oluşan tıkanıklık durumlarına yanıt olarak TCP akış kontrol mekanizmalarının davranışını yansıtabilir.

TCP bağlantılarında periyodik davranışlar, ağda tıkanıklık olup olmadığını belirlemek ve tıkanıklıktan kaçınmak için kullanılan akış kontrol algoritmalarının bir sonucu olabilir. TCP'nin tıkanıklıktan kaçınma mekanizmaları, ağdaki trafiğin durumunu izler ve uygun önlemleri alarak ağdaki tıkanıklığı azaltmaya veya önlemeye çalışır. Bu önlemler genellikle pencere boyutu ayarlamaları, ACK (onay) paketlerinin alınma hızı ve paketlerin gönderilme hızı gibi parametrelerin dinamik olarak ayarlanmasını içerir.



Şekil 6. Şekil 5'teki ile aynı verilerin başka bir görünümü.