ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 7: ΠΡΩΤΟΚΟΛΛΑ TCP ΚΑΙ UDP

29 ΝΟΕΜΒΡΙΟΥ, 2022

ΘΟΔΩΡΗΣ ΑΡΑΠΗΣ – EL18028

Ονοματεπώνυμο: Θοδωρής Αράπης	**Ομάδα:** 2
Όνομα PC/ΛΣ: LAPTOP-B2DVAJKK/ WINDOWS 10	**Ημερομηνία:** 29/11/2022
Διεύθυνση IP: 192.168.0.194 (1η άσκηση) 147.102.236.143 (2η άσκηση) 192.168.0.193 (4η άσκηση)	**Διεύθυνση MAC:** B4-69-21-1B-6C-FF

Άσκηση 1: Μετάδοση δεδομένων με TCP

Εκτελούμε τις παρακάτω εντολές:

```
C:\WINDOWS\system32>telnet 1.1.1.1
Connecting To 1.1.1.1...Could not open connection to the host, on port 23: Connect failed

C:\WINDOWS\system32>telnet 2.2.2.2
Connecting To 2.2.2.2...Could not open connection to the host, on port 23: Connect failed

C:\WINDOWS\system32>telnet 147.102.40..1
Connecting To 147.102.40..1...Could not open connection to the host, on port 23: Connect failed
```

1.1

Το φίλτρο σύλληψης είναι το εξής: «host 193.168.0.194»

1.2

Το φίλτρο απεικόνισης είναι το εξής:

«ip.dst==1.1.1.1 or ip.dst==2.2.2.2 or ip.dst=147.102.40.1»

ip.dst==1.1.1.1 or ip.dst==2.2.2.2 or ip.dst==147.102.40.1

No.	Time	Source	Destination	Protocol	Length	Info
4633	0.000000	192.168.0.194	1.1.1.1	TCP	66	51785 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
5340	1.001909	192.168.0.194	1.1.1.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 51785 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
7454	2.000169	192.168.0.194	1.1.1.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 51785 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
12926	4.005253	192.168.0.194	1.1.1.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 51785 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
21749	8.000965	192.168.0.194	1.1.1.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 51785 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
29551	8.871582	192.168.0.194	2.2.2.2	TCP	66	52393 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
30385	1.001371	192.168.0.194	2.2.2.2	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 52393 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
33127	2.001213	192.168.0.194	2.2.2.2	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 52393 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
36392	4.001386	192.168.0.194	2.2.2.2	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 52393 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
43058	8.000888	192.168.0.194	2.2.2.2	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 52393 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
60929	22.591...	192.168.0.194	147.102.40.1	TCP	66	53048 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
61245	0.514990	192.168.0.194	147.102.40.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 53048 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
61620	0.515852	192.168.0.194	147.102.40.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 53048 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
62029	0.523784	192.168.0.194	147.102.40.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 53048 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
62381	0.512930	192.168.0.194	147.102.40.1	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 53048 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

1.3

Επιλέγοντας οποιοδήποτε από τα παραπάνω πακέτα, βλέπουμε στο TCP Layer το πεδίο:

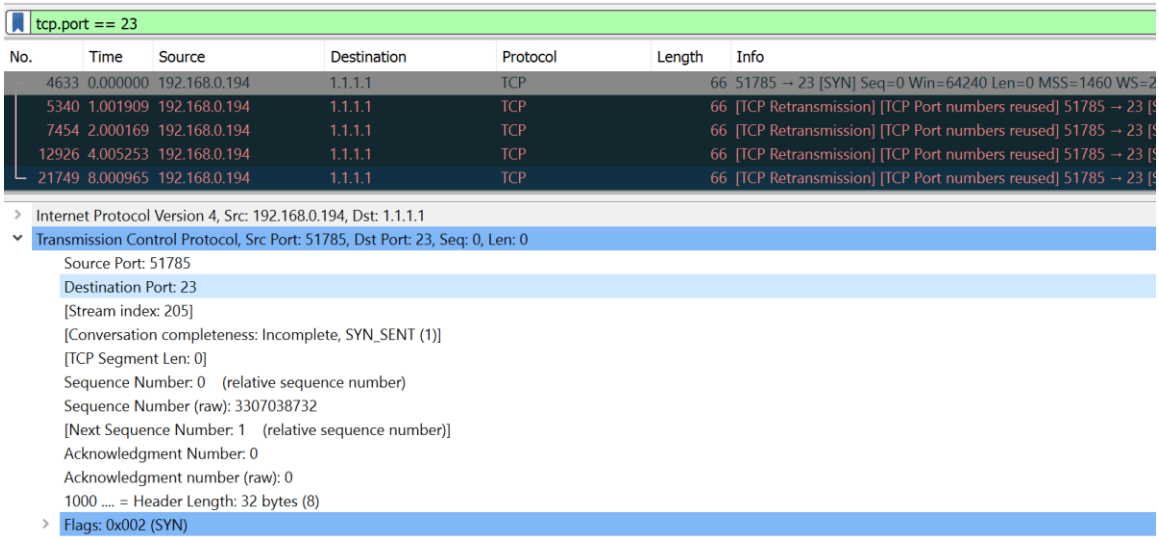
Destination Port: 23, το οποίο αντιστοιχεί στο πρωτόκολλο Telnet.

1.4

Το φίλτρο απεικόνισης είναι το εξής: «tcp.port==23»

1.5

Επιλέγοντας το πρώτο πακέτο TCP, παρατηρούμε πως το flag που είναι set για την έναρξη της επικοινωνίας είναι το SYN.



1.6

Παρατηρώντας την καταγραφή, βλέπουμε ότι γίνονται 5 προσπάθειες σύνδεσης για κάθε μία από τις προσπάθειες Α και Β (10 συνολικά), εκ των οποίων η πρώτη αποτελεί την προσπάθεια εκκίνησης εγκατάστασης και οι υπόλοιπες 4 επαναμετάδοση.

1.7

Μελετώντας την καταγραφή, μπορούμε να δούμε πως οι προσπάθειες γίνονται σε χρονικές στιγμές που είναι δυνάμεις του 2. Συγκεκριμένα, αν η πρώτη γίνεται τη στιγμή 0 (εκκίνηση εγκατάστασης), τότε έχουμε επαναμετάδοση τις στιγμές 2^0, 2^1, 2^2, 2^3.

1.8

Δεν παρατηρείται κάποια σημαντική διαφορά.

1.9

Στις προσπάθειες Α και Β παρατηρείται μόνο η προσπάθεια εκκίνησης εγκατάστασης (αποστολή SYN από τον client), ενώ στην περίπτωση Γ λαμβάνουμε επιπλέον από τον σέρβερ τα flags ACK και RST, με τα οποία γνωστοποιεί πως έλαβε το SYN και απορρίπτει τη σύνδεση. Οπότε παρατηρήσαμε το πρώτο βήμα κυρίως.

1.10

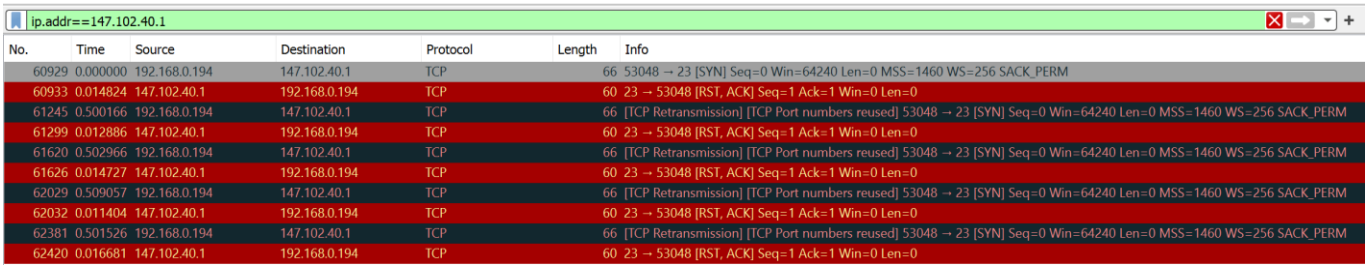
Σε κανένα τεμάχιο δε παρατηρήθηκε η σημαία FIN ως set, οπότε απλά εγκαταλείπει την προσπάθεια.

1.11

Αρκεί απλά το φίλτρο μας να είναι το «ip.addr==147.102.40.1»

1.12

Κάνει συνολικά 5 προσπάθειες.

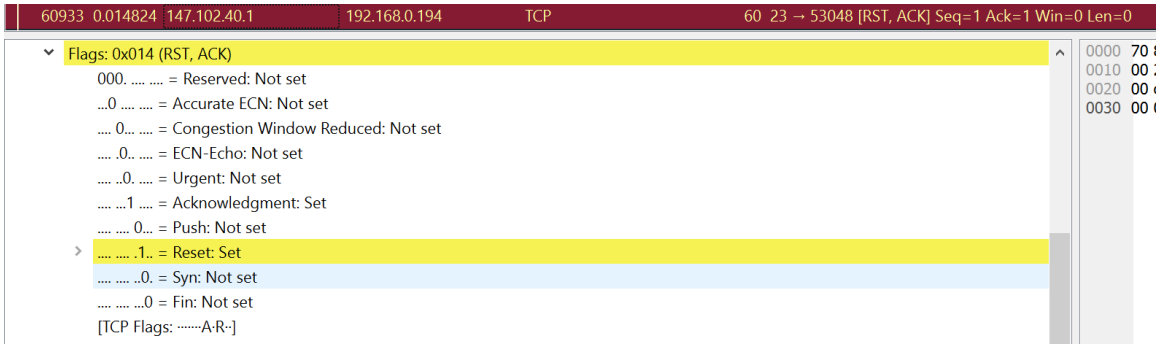


1.13

Η διαφορά με τις περιπτώσεις Α και Β είναι πως εδώ λαμβάνουμε απόκριση από τον server, η οποία έχει τα flags ACK και RST ενεργοποιημένα. Αυτό σημαίνει πως ο συγκεκριμένος σέρβερ υπάρχει, αλλά απορρίπτει τέτοιου είδους συνδέσεις όπως και περιμέναμε.

1.14

Επιλέγοντας το πρώτο πακέτο απάντησης TCP από τον 147.102.40.1 προς τον υπολογιστή μας, παρατηρούμε τα εξής flags μήκους 1 bit: Reserved, Accurate ECN, Congestion Window Reduced, ECN-Echo, Urgent, Acknowledgment, Push, Reset, Syn και Fin.



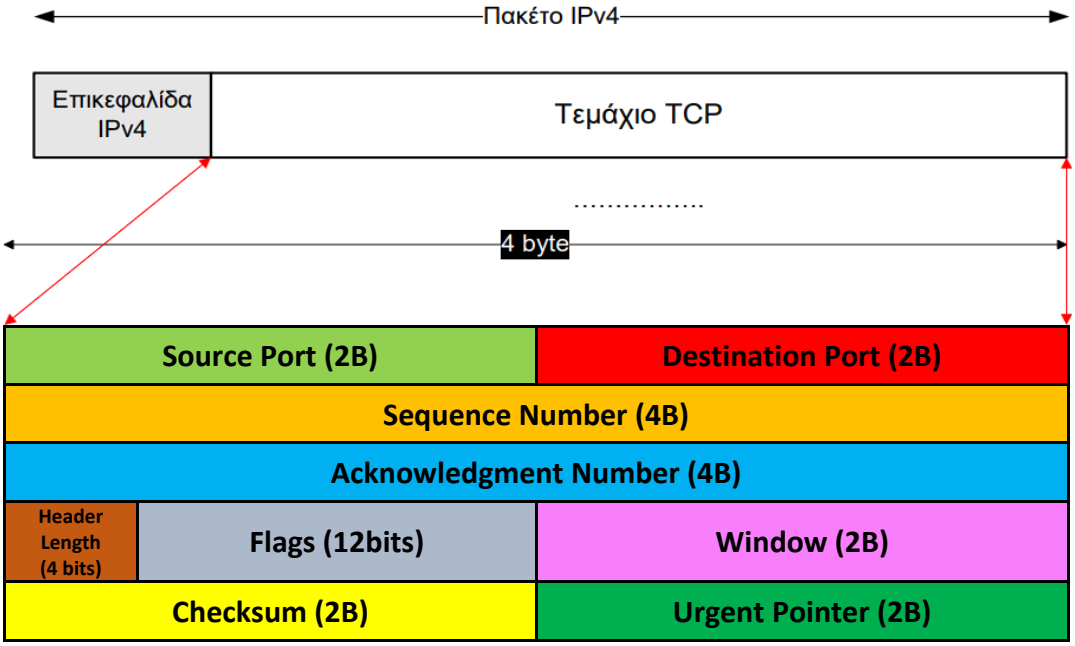
1.15

Η σημαία Reset: Set δηλώνει άρνηση εγκατάστασης.

1.16

Το ανωτέρω τεμάχιο αποτελείται από 20 bytes επικεφαλίδας και 0 bytes δεδομένων.

1.17



1.18

Σύμφωνα με την ιστοσελίδα, το μέγεθος της επικεφαλίδας TCP προσδιορίζεται από το πεδίο Data Offset το οποίο όπως λέει μας δίνει την τιμή σε λέξεις των 32 bits. Αντιθέτως, στο Wireshark το πεδίο αυτό ονομάζεται Header Length και για το συγκεκριμένο πακέτο έχει τιμή 0101 = 5 , το οποίο μας δίνει 20 bytes, όσα δηλαδή βρήκαμε προηγουμένως.

1.19

Όπως αναφέραμε, το πεδίο Header Length έχει στα περιεχόμενα του τεμαχίου την τιμή 5_{16} = 0101 = 5_{10} οπότε αφού μετράει σε λέξεις των 32 bits (4 bytes), το γινόμενο 4·5 μας δίνει 20 bytes.

1.20

Δεν υπάρχει πεδίο το οποίο να μας πληροφορεί για το συνολικό μήκος του τεμαχίου.

1.21

Το μήκος του τεμαχίου TCP μπορεί να βρεθεί εάν από το συνολικό μήκος του IPv4 πακέτου (πεδίο Total Length) αφαιρέσουμε το μήκος της IPv4 επικεφαλίδας (πεδίο Header Length). Εάν θέλουμε να βρούμε τα δεδομένα του TCP segment, θα πρέπει από την τιμή που προέκυψε να αφαιρέσουμε την τιμή του TCP Header Length. (Σημείωση: Τα παραπάνω έχουν τιμές για λέξεις των 32 bits).

1.22

Το μήκος επικεφαλίδας του πρώτου τεμαχίου TCP που στέλνει ο υπολογιστής μας στο 147.102.40.1 για την εγκατάσταση της TCP σύνδεσης είναι 32 bytes.

1.23

Παρατηρούμε πως υπάρχει διαφορά στο μήκος των παραπάνω 2 τεμαχίων κατά 12 bytes, η οποία και οφείλεται στο πεδίο Options το οποίο δεν υπήρχε στην απάντηση από τον 147.102.40.1, ενώ καταλαμβάνει 12 bytes στο πρώτο TCP τεμάχιο που εμείς αποστέλλουμε. Να σημειωθεί πως στη συγκεκριμένη περίπτωση προέκυψε μέγεθος επικεφαλίδας 32 bytes, το οποίο είναι ακέραιο πολλαπλάσιο μιας λέξης (32 bits). Εάν προέκυπτε διαφορετικά, θα υπήρχε TCP Padding προκειμένου να γίνει το μήκος ακέραιο πολλαπλάσιο των 4 bytes.

Άσκηση 2: Εντολή ping σε άλλο υποδίκτυο

2.1

Χρησιμοποιήσαμε το φίλτρο σύλληψης: «(tcp) and (ip host edu-dy.cn.ntua.gr)»

Εγκατάσταση σύνδεσης

2.2

Για την έναρξη της επικοινωνίας προσπαθεί να συνδεθεί στη θύρα 21, η οποία και αντιστοιχεί στο πρωτόκολλο FTP ελέγχου.

2.3

Αντίστοιχα για τη σύνδεση μεταφοράς δεδομένων, συνδέεται στη θύρα 20, η οποία και αντιστοιχεί στο πρωτόκολλο FTP μεταφοράς δεδομένων.

2.4

Χρησιμοποιήσαμε το φίλτρο απεικόνισης: «tcp.port==21»

2.5

Ανταλλάσσονται τα παρακάτω 3 πακέτα (τριπλή χειραψία):

tcp.port==21

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	147.102.236.143	147.102.40.15	TCP	66	52128 → 21 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=1 SACK_PERM
2	0.001749	147.102.40.15	147.102.236.143	TCP	66	21 → 52128 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=536 WS=64 SACK_PERM
3	0.001839	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=1 Ack=1 Win=8192 Len=0

2.6

Χρησιμοποιούνται οι σημαίες SYN και ACK, Πιο συγκεκριμένα, χρησιμοποιείται η σημαία SYN στο πρώτο τεμάχιο από εμάς προς τον σέρβερ, ύστερα οι σημαίες SYN και ACK κατά την απόκριση του σέρβερ και τέλος η σημαία ACK από εμάς προς τον σέρβερ.

2.7

Το μέγεθος των επικεφαλίδων TCP των παραπάνω τεμαχίων (screenshot) είναι 32, 32 και 20 bytes αντίστοιχα.

2.8

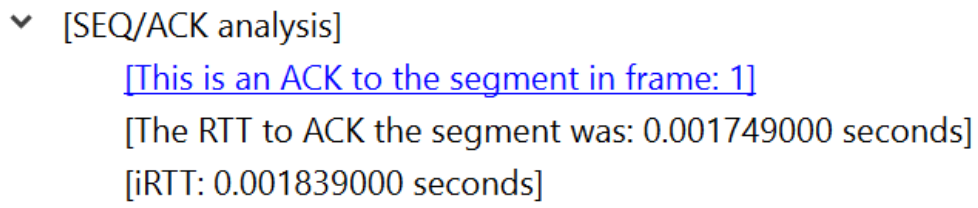
Το μέγεθος δεδομένων των τεμαχίων αυτών είναι μηδενικό, όπως είναι αναμενόμενο αφού στην τριπλή χειραψία δεν ανταλλάσσονται δεδομένα.

2.9

Από το στιγμιότυπο στο 2.5, παρατηρούμε πως η τριπλή χειραψία διαρκεί 0.001839 seconds.

2.10

Μόνο τα δύο από τα τρία παραπάνω τεμάχια περιέχουν το πεδίο «SEQ/ACK analysis». Επιλέγοντας ένα από αυτά βλέπουμε ότι το IRTT συμφωνεί με τον παραπάνω χρόνο.



2.11

Η δική μας πλευρά ανακοινώνει τον σχετικό/απόλυτο αριθμό σειράς 0/1575067678, ενώ η πλευρά του σέρβερ τον σχετικό/απόλυτο αριθμό σειράς 0/2143573936.

2.12

Παρατηρώντας το τεμάχιο TCP με το οποίο ο ftp server δηλώνει πως αποδέχεται τη σύνδεση, βλέπουμε το ACK number να είναι 1/1575067678. Συγκρίνοντας με το πρώτο τεμάχιο που στείλαμε εμείς, βλέπουμε ότι το Sequence number είναι το ακριβώς προηγούμενο νούμερο. Είναι λογικό να αυξάνεται το νούμερο κατά ένα αφού αναμένεται το επόμενο byte δεδομένων και αφού δεν έχουμε δεδομένα τότε η αύξηση είναι μοναδιαία.

2.13

Όσον αφορά το 3ο τεμάχιο της τριπλής χειραψίας, το Sequence Number του είναι το Acknowledgment Number του προηγούμενου τεμαχίου, ενώ το Acknowledgment Number του είναι το Sequence Number του προηγούμενου τεμαχίου αυξημένο κατά 1.

2.14

Το μήκος δεδομένων των τριών τεμαχίων της τριπλής χειραψίας είναι μηδενικό, αφού όπως γνωρίζουμε δεν ανταλλάσσονται δεδομένα κατά τη διαδικασία αυτή.

2.15

Παρατηρώντας στο παράθυρο λεπτομερειών βλέπουμε πως τα πεδία Sequence Number και Acknowledgment Number καταλαμβάνουν 4 bytes, επομένως η μέγιστη τιμή που μπορούν να λάβουν είναι 2^{32} - 1 = 4.294.967.295.

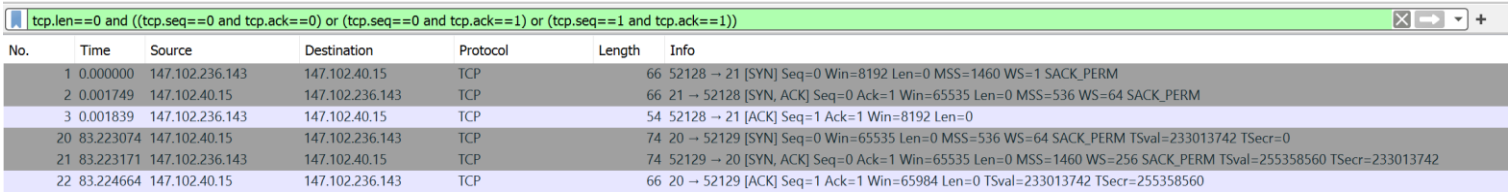
2.16

Η σύνταξη του φίλτρου είναι:

«tcp.len==0 and ((tcp.seq==0 and tcp.ack==0) or (tcp.seq==0 and tcp.ack==1) or

(tcp.seq==1 and tcp.ack==1))»

Τα αποτελέσματα είναι:



2.17

Από το παραπάνω στιγμιότυπο, βλέπουμε στο πρώτο τεμάχιο την τιμή Win=8.192, η οποία αντιστοιχεί στο μέγεθος του παραθύρου λήψης που ανακοινώνει ο υπολογιστής μας.

2.18

στο δεύτερο τεμάχιο βλέπουμε την τιμή Win=65.535, η οποία αντιστοιχεί στο μέγεθος του παραθύρου λήψης που ανακοινώνει ο σέρβερ.

2.19

Η σχετική πληροφορία υπάρχει στο πεδίο Window της TCP επικεφαλίδας.

2.20

Η τιμή της κλίμακας παραθύρου που ανακοινώνει ο υπολογιστής μας είναι 0 (multiply by 1), ενώ ο σέρβερ ανακοινώνει την τιμή 6 (multiply by 64).

2.21

Η σχετική πληροφορία βρίσκεται στο πεδίο Options της επικεφαλίδας TCP μόνο των δύο πρώτων τεμαχίων που ανταλλάσσουν ο υπολογιστής μας και ο σέρβερ.

2.22

Από το στιγμιότυπο του ερωτήματος 2.16 βλέπουμε πως ο υπολογιστής μας ανακοινώνει MSS = 1460 bytes.

2.23

Επαναλαμβάνουμε την διαδικασία που είχαμε κάνει στην 6η εργαστηριακή άσκηση και κάνουμε ping στην ιστοσελίδα www.google.com με icmp payload 1473 bytes, το οποίο και αποτυγχάνει να φτάσει στον προορισμό μιας και απαιτείται θρυμματισμός ενώ εμείς έχουμε την σημαία «-f» ενεργή. Στο παραπάνω payload, αν προσθέσουμε τις επικεφαλίδες IP (20 bytes) και ICMP (8 bytes) έχουμε MTU = 1501 bytes. Δοκιμάζοντας τώρα για icmp payload 1472 bytes (MTU = 1500 bytes) το πακέτο επιτυγχάνει να φτάσει στον προορισμό.

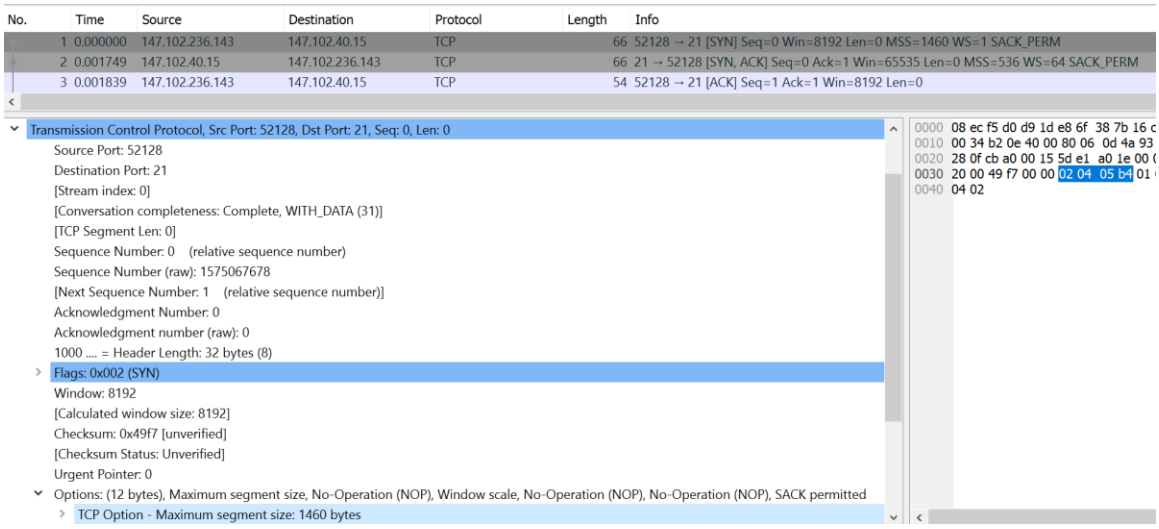
```
C:\Users\Theodore>ping -n 1 -f -l 1473 www.google.com

Pinging www.google.com [142.250.184.196] with 1473 bytes of data:
Packet needs to be fragmented but DF set.

Ping statistics for 142.250.184.196:
    Packets: Sent = 1, Received = 0, Lost = 1 (100% loss),

C:\Users\Theodore>ping -n 1 -f -l 1472 www.google.com

Pinging www.google.com [142.250.184.196] with 1472 bytes of data:
Reply from 192.168.1.1: Packet needs to be fragmented but DF set.

Ping statistics for 142.250.184.196:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
```

Συνεπώς, η διεπαφή του υπολογιστή μας έχει MTU = 1500 bytes. Στην καταγραφή παρατηρούμε ότι έχουμε MSS = 1460 bytes στο οποίο αν προσθέσουμε τις επικεφαλίδες IP (20 bytes) και TCP (20 bytes) έχουμε σύνολο 1500 bytes.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	147.102.236.143	147.102.40.15	TCP	66	52128 → 21 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=1 SACK_PERM

2.24

Η τιμή MSS βρίσκεται στο υποπεδίο TCP-Option – Maximum segment size του πεδίου Options της επικεφαλίδας TCP.



2.25

Από τις καταγραφές βλέπουμε πως ο edu-dy.cn.ntua.gr ανακοινώνει MSS = 536 bytes.

2.26

Αν κάνουμε ping στο edu-cy.cn.ntua.gr με 549 και 548 θα συμβεί ότι συνέβη και προηγουμένως για τις τιμές 1473 και 1472 αντίστοιχα. Συνεπώς, με την ίδια λογική προκύπτει MTU = 576 (MSS = 536 + 20 + 20 bytes (IP και TCP headers)).

2.27

Σύμφωνα με τα παραπάνω, το μεγαλύτερο τεμάχιο TCP που μπορεί να στείλει ο υπολογιστής μας (επικεφαλίδα TCP + δεδομένα) είναι 556 bytes.

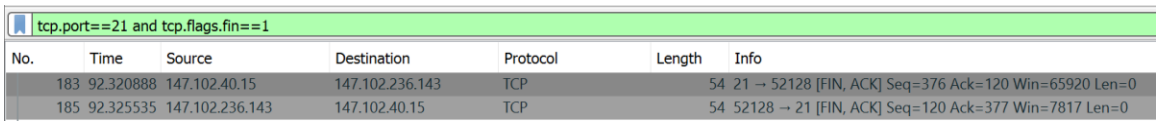
Απόλυση σύνδεσης

2.28

Ενεργοποιείται η σημαία FIN.

2.29

Το φίλτρο απεικόνισης είναι: «tcp.port==21 and tcp.flags.fin==1».



2.30

Ο σέρβερ εκκινεί τη διαδικασία απόλυσης σύνδεσης.

2.31

Ανταλλάσσονται 2 TCP τεμάχια όπως φαίνεται στο παραπάνω screenshot.

2.32

Και οι 2 TCP επικεφαλίδες έχουν μέγεθος 20 bytes.

2.33

Έχουν μηδενικό μέγεθος δεδομένων

2.34

Επιλέγουμε το πακέτο με το οποίο ο υπολογιστής μας εκκινεί την απόλυση της σύνδεσης και βλέπουμε στην IP επικεφαλίδα το πεδίο Total Length να έχει τιμή 40. Η τιμή αυτή προκύπτει ως το άθροισμα της IP επικεφαλίδας (το οποίο από το πεδίο Header Length του IP βλέπουμε πως είναι ίσο με 20) με το συνολικό μέγεθος του TCP τεμαχίου, το οποίο είδαμε πως δεν έχει δεδομένα παρά μόνο 20 bytes επικεφαλίδας.

2.35

Με την ίδια λογική με το ερώτημα 3.35 προκύπτει το ίδιο μήκος πακέτου IPv4.

2.36

Η πλευρά του σέρβερ μετέδωσε συνολικά 1035 bytes, ενώ ο υπολογιστής μας 995 bytes.

2.37

Για να βρούμε πόσα μας έστειλε ο σέρβερ θα εφαρμόσαμε το φίλτρο «tcp.port==21 and ip.src==147.102.40.15» και θα αθροίσαμε τα bytes της στήλης Length.

tcp.port==21 and ip.src==147.102.40.15

No.	Time	Source	Destination	Protocol	Length	Info
2	0.001749	147.102.40.15	147.102.236.143	TCP	66	21 → 52128 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=536 WS=64 SACK_PERM
4	0.006510	147.102.40.15	147.102.236.143	FTP	128	Response: 220 ProFTPD 1.3.4a Server (ProFTPD Default Installation) [147.102.40.15]
6	0.024411	147.102.40.15	147.102.236.143	FTP	74	Response: 200 UTF8 set to on
9	10.985683	147.102.40.15	147.102.236.143	FTP	129	Response: 331 Anonymous login ok, send your complete email address as your password
12	27.034782	147.102.40.15	147.102.236.143	FTP	104	Response: 230 Anonymous access granted, restrictions apply
15	35.964436	147.102.40.15	147.102.236.143	FTP	73	Response: 200 Type set to I
18	83.213689	147.102.40.15	147.102.236.143	FTP	83	Response: 200 PORT command successful
23	83.224664	147.102.40.15	147.102.236.143	FTP	125	Response: 150 Opening BINARY mode data connection for PCATTCP.exe (61440 bytes)
179	83.245668	147.102.40.15	147.102.236.143	FTP	77	Response: 226 Transfer complete
182	92.320888	147.102.40.15	147.102.236.143	FTP	68	Response: 221 Goodbye.
183	92.320888	147.102.40.15	147.102.236.143	TCP	54	21 → 52128 [FIN, ACK] Seq=376 Ack=120 Win=65920 Len=0
186	92.329271	147.102.40.15	147.102.236.143	TCP	54	21 → 52128 [ACK] Seq=377 Ack=121 Win=65920 Len=0

Όμοια για το πλήθος των bytes που έστειλε ο υπολογιστής μας θα εφαρμόσουμε το φίλτρο «tcp.port==21 and ip.dst==147.102.40.15» και θα αθροίσαμε τα bytes της στήλης Length.

tcp.port==21 and ip.dst==147.102.40.15

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	147.102.236.143	147.102.40.15	TCP	66	52128 → 21 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=1 SACK_PERM
3	0.001839	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=1 Ack=1 Win=8192 Len=0
5	0.022467	147.102.236.143	147.102.40.15	FTP	68	Request: OPTS UTF8 ON
7	0.074211	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=15 Ack=95 Win=8098 Len=0
8	10.956853	147.102.236.143	147.102.40.15	FTP	70	Request: USER anonymous
10	11.030556	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=31 Ack=170 Win=8023 Len=0
11	27.024484	147.102.236.143	147.102.40.15	FTP	81	Request: PASS kriskoutsi@gmail.com
13	27.078603	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=58 Ack=220 Win=7973 Len=0
14	35.961461	147.102.236.143	147.102.40.15	FTP	62	Request: TYPE I
16	36.007188	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=66 Ack=239 Win=7954 Len=0
17	83.210741	147.102.236.143	147.102.40.15	FTP	84	Request: PORT 147,102,236,143,203,161
19	83.220822	147.102.236.143	147.102.40.15	FTP	72	Request: RETR PCATTCP.exe
180	83.245769	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=114 Ack=362 Win=7831 Len=0
181	92.318066	147.102.236.143	147.102.40.15	FTP	60	Request: QUIT
184	92.320951	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [ACK] Seq=120 Ack=377 Win=7817 Len=0
185	92.325535	147.102.236.143	147.102.40.15	TCP	54	52128 → 21 [FIN, ACK] Seq=120 Ack=377 Win=7817 Len=0

Μεταφορά δεδομένων

2.38

Η σύνταξη του φίλτρου απεικόνισης είναι: «tcp.port==20.»

2.39

Όπως βλέπουμε από το ακόλουθο στιγμιότυπο, ανακοινώνονται τιμές MSS = 536 και MSS = 1460 από την πλευρά του edu-dy.cn.ntua.gr και τη δικιά μας αντίστοιχα.

tcp.port==20

No.	Time	Source	Destination	Protocol	Length	Info
20	83.223074	147.102.40.15	147.102.236.143	TCP	74	20 → 52129 [SYN] Seq=0 Win=65535 Len=0 MSS=536 WS=64 SACK_PERM TSval=233013742 TSecr=0
21	83.223171	147.102.236.143	147.102.40.15	TCP	74	52129 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM TSval=255358560 TSecr=233013742
22	83.224664	147.102.40.15	147.102.236.143	TCP	66	20 → 52129 [ACK] Seq=1 Ack=1 Win=65984 Len=0 TSval=233013742 TSecr=255358560

2.40

Το μέγεθος του μεγαλύτερου τεμαχίου TCP που μπορεί να στείλει ο σέρβερ στον υπολογιστή μας ανέρχεται σε 556 bytes (536 (MSS) + 20 (TCP header) = 556 bytes).

2.41

Βρίσκουμε την διαφορά των χρόνων των δύο πρώτων τεμαχίων στην καταγραφή κατά την εγκατάσταση της σύνδεσης και έχουμε RTT = 83.223171 – 83.223074 = 0.000097 seconds.

No.	Time
20	83.223074
21	83.223171

2.42

Ο υπολογιστής μας δεν στέλνει ACK για κάθε πακέτο που λαμβάνει. Συγκεκριμένα, κατά χρονική σειρά στέλνει ACK κατά την ακολουθία πακέτων: 2, 2, 3, 1, 7, 1, 5, 3, 2, 4, 2, 2, 6, 5, 4, 3, 4, 2, 4, 2, 2, 2, 5, 1, 5, 5, 3, 2, 5, 4, 2, 6, 5 και 3.

2.43

Ο εξυπηρετητής έστειλε 114 τεμάχια δεδομένων.

2.44

Ο υπολογιστής μας έστειλε 34 τεμάχια ACK για τα δεδομένα που έλαβε.

2.45

Το μέγεθος του παραθύρου που ανακοινώνει ο υπολογιστής μας είναι Win = 1048832 bytes.

2.46

Δεν είναι ίδια. Η τιμή αυτή προκύπτει αν πολλαπλασιάσουμε το window scale factor που ανακοίνωσε ο υπολογιστής μας (2^8 = 256 στο πρώτο πακέτο που στέλνει με port 20) με το μέγεθος παραθύρου που δείχνει το τεμάχιο ACK που επιλέξαμε (4097 bytes). Οπότε συνολικά 256·4097 = 1048832 bytes.

2.47

Όχι δεν αλλάζει η τιμή του παραθύρου στα τεμάχια που στέλνει ο υπολογιστής μας. Το ότι δεν αλλάζουν οι τιμές, σημαίνει πως δεν υπάρχει υπερφόρτωση του buffer του υπολογιστή μας.

2.48

Αν είχαμε μηδενική τιμή window size θα σήμαινε πως ο receiver buffer του υπολογιστή μας θα είχε γεμίσει και δεν θα μπορούσε να λάβει άλλα δεδομένα.

2.49

Συνολικά το πλαίσιο έχει μέγεθος 590 bytes. Όσον αφορά τις επικεφαλίδες, έχουν μέγεθος 14/20/20 για τα Ethernet/IP/TCP πρωτόκολλα αντίστοιχα.

2.50

Στο πεδίο Total Length της IP επικεφαλίδας βλέπουμε την τιμή 576 bytes. Δεδομένου ότι οι επικεφαλίδες IP και TCP είναι αθροιστικά 40 bytes, το μέγεθος των δεδομένων του TCP τεμαχίου ανέρχεται σε 536 bytes, το οποίο συμφωνεί με το 2.40.

2.51

Διαβάζοντας το documentation βλέπουμε πως υπό κανονικές συνθήκες δε θα στέλνονταν ποτέ δεδομένα μεγαλύτερα από την παραπάνω τιμή. Υπάρχει περίπτωση μόνο να σταλούν δεδομένα μεγαλύτερα από αυτά που μπορεί να διαχειριστεί το ενδιάμεσο δίκτυο που μεσολαβεί των 2 κόμβων, οπότε και εκεί να πρέπει να γίνει fragmentation από τον αποστολέα των πακέτων και να αποφευχθεί να γίνει από τους δρομολογητές.

2.52

Εφαρμόζουμε ξανά το φίλτρο απεικόνισης tcp.port==20. Επιλέγουμε το πρώτο πακέτο TCP που έστειλε ο υπολογιστής μας και βλέπουμε το Sequence Number (raw) του, το οποίο είναι: 1444215367. Στη συνέχεια επιλέγουμε το τελευταίο πακέτο TCP που έστειλε ο υπολογιστής μας, το οποίο έχει τιμή Sequence Number (raw): 1444215368, επομένως κατά τη σύνδεση δεδομένων ο υπολογιστής μας δεν έστειλε κανένα byte δεδομένων (η διαφορά των δύο Sequence Number οφείλεται στο τεμάχιο με flag SYN, το οποίο flag έχει μήκος 1 byte και ενώ δεν περιέχει δεδομένα, αυξάνει το Sequence Number). Εφαρμόζοντας την ίδια μέθοδο αντίστροφα, βρίσκουμε πως ο server μας έστειλε 61440 bytes (αφαιρώντας τώρα 2 bytes, ένα για το τεμάχιο με flag SYN, κα ένα για αυτό με flag FIN).

2.53

Εφαρμόζουμε ξανά το φίλτρο ftp-data. Από εκεί βρίσκουμε τη διαφορά χρόνου καταγραφής μεταξύ τελευταίου και πρώτου πακέτου, η οποία είναι 0.010477 seconds. Εφόσον προηγουμένως βρήκαμε πόσα data μας έστειλε, ο ρυθμός μετάδοσης που μας τα έστειλε είναι (61440 bytes / 0.010477 sec) = 5864.27412 Kbytes/sec.

2.54

Δεν εντοπίστηκε κάποιο Retransmission TCP πακέτο, επομένως δεν υπήρξαν αναμεταδόσεις τεμαχίων.

Άσκηση 3: Αποφυγή συμφόρησης στο TCP

3.1

Η σύνταξη του φίλτρου είναι: «tcp.port==20»

tcp.port==20

No.	Time	Source	Destination	Protocol	Length	Info
20	29.690533	147.102.40.15	94.65.141.44	TCP	74	20 → 19586 [SYN] Seq=0 Win=65535 Len=0 MSS=536 WS=64 SACK_PERM TSval=32103835 TSecr=0
21	29.705159	94.65.141.44	147.102.40.15	TCP	66	19586 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1452 WS=256 SACK_PERM
22	29.705207	147.102.40.15	94.65.141.44	TCP	54	20 → 19586 [ACK] Seq=1 Ack=1 Win=65920 Len=0

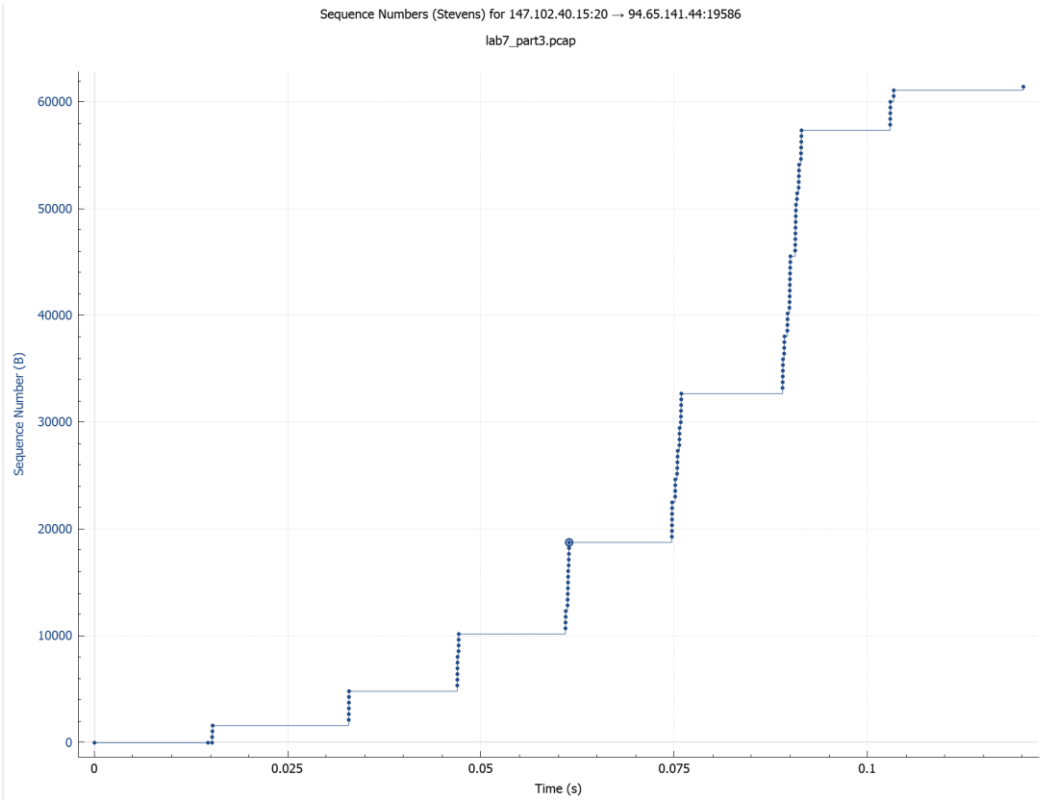
3.2

Η ζητούμενη διεύθυνση IPv4 του υπολογιστή είναι 94.65.141.44.

3.3

Ακολουθώντας την ίδια διαδικασία με πριν, βρίσκουμε το RTT να είναι ίσο με 0.014626 seconds, αρκετά μεγαλύτερος από το προηγούμενο RTT στο ερώτημα 2.41 (0.000097 seconds).

▾ [SEQ/ACK analysis]
[This is an ACK to the segment in frame: 20]
[The RTT to ACK the segment was: 0.014626000 seconds]
[iRTT: 0.014674000 seconds]



3.4

Με εξαίρεση τα 2 πρώτα TCP πακέτα, τα οποία είναι μέρος της τριπλής χειραψίας, παρατηρούμε πως τα FTP data στέλνονται σε “ριπές” πακέτων. Δηλαδή στέλνονται πολλά μαζί το ένα πίσω από το άλλο ανά τακτά χρονικά διαστήματα, κάθε φορά περισσότερα σε πλήθος από την προηγούμενη στη γενική περίπτωση (προφανώς στο τέλος το “σκαλοπάτι” είναι μικρότερο, καθώς στάλθηκαν όλα τα δεδομένα εκεί).

3.5

Στο πρώτο RTT ο edu-dy.cn.ntua.gr έστειλε 4 τεμάχια. Από το documentation βρίσκουμε ότι:

```
IW, the initial value of cwnd, MUST be set using the following
guidelines as an upper bound.

If SMSS > 2190 bytes:
    IW = 2 * SMSS bytes and MUST NOT be more than 2 segments
If (SMSS > 1095 bytes) and (SMSS <= 2190 bytes):
    IW = 3 * SMSS bytes and MUST NOT be more than 3 segments
if SMSS <= 1095 bytes:
    IW = 4 * SMSS bytes and MUST NOT be more than 4 segments
```

Όπου SMSS = Sender MSS και IW η αρχική τιμή του cwnd (congestion window, ένα όριο από τη μεριά του αποστολέα για τα δεδομένα που μπορεί να στείλει σε ένα δίκτυο πριν λάβει ACK). Παρατηρούμε από το παράθυρο Sequence Numbers, 4 πακέτα κατά το πρώτο RTT, το οποίο συμφωνεί πλήρως με τον τρίτο περιορισμό, ότι δηλαδή μπορούν να σταλούν μέχρι και 4 Segments (δεδομένου ότι ο edu-dy.cn.ntua.gr βλέπουμε πως έχει MSS 536 bytes).

3.6

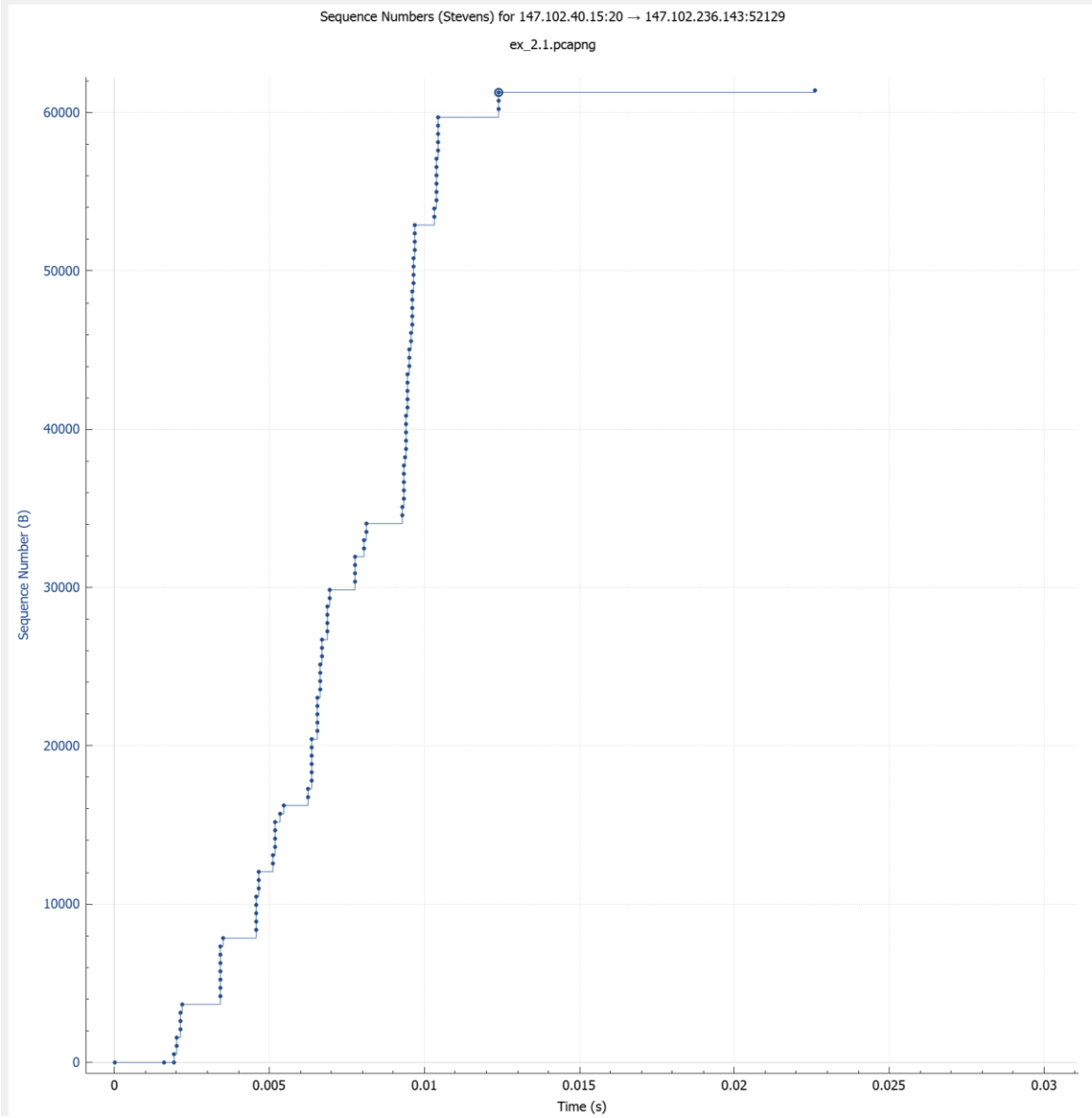
Στο δεύτερο RTT έστειλε 6 τεμάχια, στο τρίτο 10 τεμάχια και στο τέταρτο 16 τεμάχια.

3.7

Στο πρώτο RTT στάλθηκε 1 ACK, στο δεύτερο 2 και στο τρίτο 3. Παρατηρούμε πως σε κάθε ριπή αποστολής δεδομένων στέλνονται ολοένα και περισσότερα ACK μετά από κάθε ριπή που λαμβάνει. Πιο συγκεκριμένα, βλέπουμε ότι για κάθε πλήθος ACK που λαμβάνει η πηγή, αυξάνεται το congestion window κατά το διπλάσιο του πλήθους αυτού. Αυτό επιβεβαιώνει και τον Slow Start αλγόριθμο, καθώς ξεκινάει με μικρό ρυθμό μετάδοσης για να δοκιμάσει το δίκτυο και αυξάνει σταδιακά.

3.8

Το ζητούμενο διάγραμμα:



Παρατηρούμε ότι πάλι είναι φανερό το Slow Start μοτίβο αναμετάδοσης. Μάλιστα, στην περίπτωσή μας η μετάδοση των δεδομένων διαρκεί συνολικά αρκετά μικρότερο χρόνο από την μετάδοση της δοσμένης καταγραφής, γεγονός που δεν κάνει πολύ ξεκάθαρα εμφανές τον μοτίβο αύξησης του congestion window.

Άσκηση 4: Μετάδοση δεδομένων με UDP

4.1

Το φίλτρο σύλληψης είναι το «udp».

4.2

Σχετικά με το πρώτο UDP datagram που έστειλε ο υπολογιστής μας έχουμε τα εξής πεδία:

- Source Port (2 bytes)
- Destination Port (2 bytes)
- Length (2 bytes)
- Checksum (2 bytes)

4.3

Το συνολικό μήκος της επικεφαλίδας UDP είναι 8 bytes.

4.4

Το δεδομενόγραμμα αυτό είναι ενθυλακωμένο σε ένα πακέτο IPv4 και έχει συνολικό μέγεθος 42 bytes.

4.5

Το πεδίο Length εκφράζει το συνολικό μέγεθος του UDP datagram

4.6

Προφανώς το ελάχιστο μέγεθος δεδομενογραμμάτων UDP που μπορεί να μεταφερθεί από ένα πακέτο IPv4 είναι 8 bytes και αυτό συμβαίνει όταν το datagram δε μεταφέρει δεδομένα παρά μόνο την επικεφαλίδα.

4.7

Όπως αναφέραμε παραπάνω, το ελάχιστο μέγεθος δεδομενογράμματος UDP είναι 8 bytes, οπότε το IPv4 πακέτο μπορεί να μεταφέρει το λιγότερο 8 bytes μήνυμα. Σχετικά με το μέγιστο μέγεθος μηνύματος, είδαμε πως το πεδίο Length παραπάνω αφορά το συνολικό μήκος και ότι είναι 2 bytes = 16 bits, επομένως, θα περιμέναμε η μέγιστη τιμή που μπορεί να λάβει είναι 216 - 1 = 65.535 bytes. Ωστόσο, στο σημείο αυτό πρέπει να θυμηθούμε πως το πεδίο Total Length της IPv4 επικεφαλίδας είναι επίσης 2 bytes, με μέγιστη τιμή 65.535 bytes, επομένως η μέγιστη τιμή που μπορεί να λάβει ένα UDP Datagram, άρα και το μέγιστο μέγεθος μηνύματος που μπορεί να μεταφέρει το IPv4 με χρήση πρωτοκόλλου UDP, είναι 65.535 – 20 = 65.515 bytes, όπου 20 bytes το ελάχιστο μέγεθος μιας IP επικεφαλίδας.

4.8

Το πεδίο Header Length ενός IPv4 πακέτου αποτελείται από 4 bits, επομένως παίρνει μέγιστη τιμή $1111_2 = 15_{10}$ και δεδομένου ότι μετράει το μέγεθος σε λέξεις των 4 bytes, το μέγιστο IPv4 header είναι 60 bytes. Επομένως, προκειμένου ένα πακέτο UDP να σταλεί/παραληφθεί με βεβαιότητα πρέπει να έχει συνολικό μήκος μέχρι και (576-60) = 516 bytes. Άρα το μέγιστο μήκος μηνύματος είναι 512 bytes.

4.9

Παρατηρούμε τα πρωτόκολλα SSDP και QUIC.

4.10

Το φίλτρο απεικόνισης είναι το «dns».

4.11

Η IPv4 διεύθυνση του DNS server που χρησιμοποιήθηκε είναι 168.192.0.193

4.12

Θύρες προέλευσης/προορισμού για την ερώτηση στον DNS Server είναι αντίστοιχα: 53464/53.

4.13

Θύρες προέλευσης/προορισμού για την ερώτηση στον DNS Server είναι αντίστοιχα: 53/53464.

4.14

Η θύρα 53 αντιστοιχεί στο πρωτόκολλο DNS.