ΕΘΝΙΚΌ ΜΕΤΣΌΒΙΟ ΠΟΛΥΤΕΧΝΕΊΟ

ΣΧΟΛΉ ΗΛΕΚΤΡΟΛΌΓΩΝ ΜΗΧΑΝΙΚΏΝ ΚΑΙ ΜΗΧΑΝΙΚΏΝ ΥΠΟΛΟΓΙΣΤΏΝ

ΔΊΚΤΥΑ ΕΠΙΚΟΙΝΩΝΙΏΝ

(2018-2019)

3η ΕΡΓΑΣΤΗΡΙΑΚΉ ΑΝΑΦΟΡΆ

Ονοματεπώνυμο : Χρήστος Τσούφης

Α.Μ. : 03117176

1. Πρωτόκολλα κυλιόμενου παραθύρου

Το πρωτόκολλο κυλιόμενου παραθύρου (Sliding Window Protocol, SWP) είναι ένα πρωτόκολλο που ανήκει στο στρώμα διασύνδεσης δεδομένων (Data Link) του μοντέλου OSI. Το πρωτόκολλο παρέχει full-duplex επικοινωνία και χρησιμοποιείται για έλεγχο ροής (flow control). Χαρακτηρίζεται από ένα "παράθυρο" αναγνώρισης το οποίο "κυλίεται" καθώς προχωράει η μετάδοση δεδομένων. Η βασική ιδέα στο πρωτόκολλο συρόμενου παραθύρου είναι ότι τόσο ο αποστολέας όσο και ο παραλήπτης διατηρούν ένα παράθυρο αναγνώρισης (acknowledgement). Σε κάθε πλευρά του είναι ενεργή μια διεργασία (process) η οποία αναλαμβάνει τη σωστή μετάδοση. Περιλαμβάνει δύο μέρη, ένα το οποίο στέλνει εξερχόμενα δεδομένα και ένα το οποίο διαχειρίζεται τα εισερχόμενα. Με τον τρόπο αυτό παρέχεται full-duplex μετάδοση, καθώς ο κάθε host μπορεί να είναι ταυτόχρονα αποστολέας και παραλήπτης. Το πρωτόκολλο είναι ικανό να θυμάται τα πακέτα που έχουν αναγνωρισθεί. Για κάθε μη αναγνωρισμένο πακέτο χρησιμοποιείται ξεχωριστός χρονομετρητής και αν ο προκαθορισμένος χρόνος λήξει (αν το πακέτο χαθεί) ο αποστολέας ξαναστέλνει τα δεδομένα. Το παράθυρο μετακινείται πέρα από όλα τα ήδη αναγνωρισμένα πακέτα, όταν ο αποστολέας επαναρυθμίζει τη θέση του. Στην πλευρά του παραλήπτη, όταν γίνει λήψη όλων των αναμενόμενων πακέτων το παράθυρο προχωράει. Το παράθυρο επομένως χωρίζει την ακολουθία των πακέτων σε αναγνωρισμένα, τα οποία έχουν αποσταλεί επιτυχώς, πακέτα εντός του παραθύρου τα οποία μεταδίδονται και μη απεσταλμένα πακέτα εκτός του παραθύρου. Παραλλαγές του πρωτοκόλλου SWP είναι το Selective Repeat, το Go Back N και το Stop and Wait, όπως θα μελετήσουμε παρακάτω.

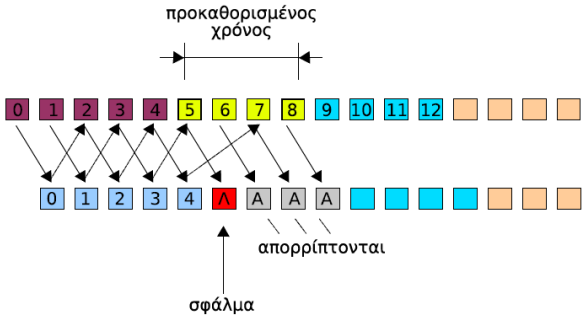
2. Πρωτόκολλα Stop and Wait και Go Back N

Πρωτόκολλο Stop and Wait

Στο πρωτόκολλο Stop and Wait (SW) το μήκος του παραθύρου μετάδοσης τίθεται ίσο με μονάδα. Αυτό καθιστά αναγκαία την αναγνώριση αποστολής κάθε πακέτου (αποστολή πακέτου και λήψη ACK) πριν σταλεί το επόμενο στη σειρά. Το συγκεκριμένο πρωτόκολλο είναι αργό και αναποτελεσματικό αφού κάθε στιγμή μπορεί να αποστέλλεται μόνο ένα πακέτο.

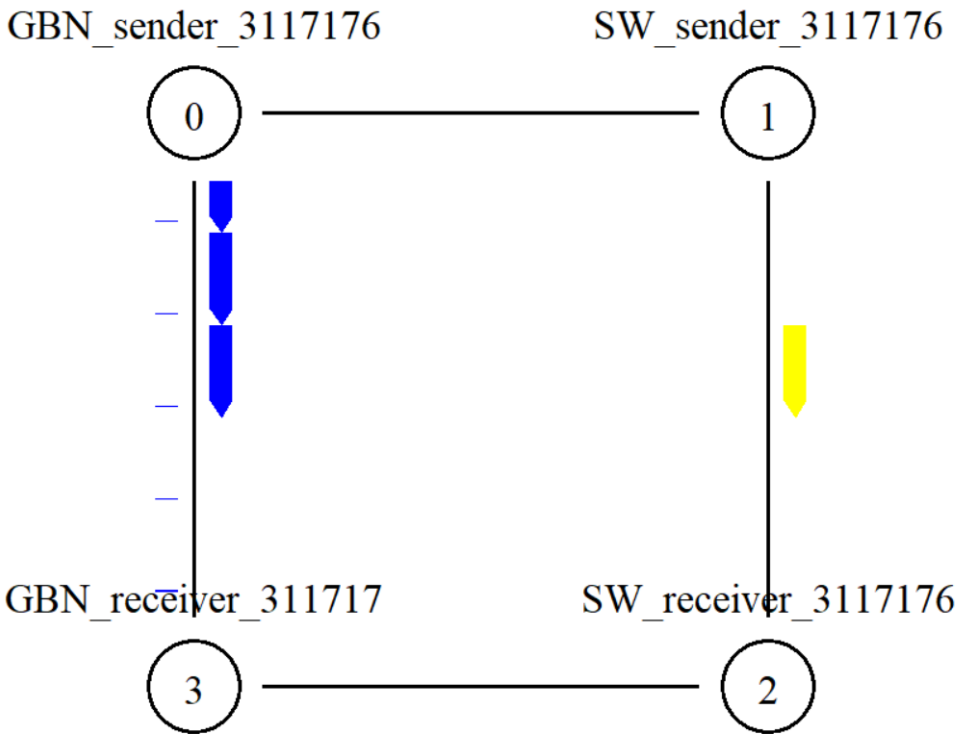
Πρωτόκολλο Go Back N

Στο πρωτόκολλο Go Back N (GBN) αν κάποιο πακέτο στην ακολουθία απουσιάζει, τότε απορρίπτονται όλα τα πακέτα που ακολουθούν (δεν αποστέλλεται ACK) και επομένως ο αποστολέας πρέπει να στείλει όλα τα πακέτα από το σημείο εκείνο και μετά (ως το πέρας του παραθύρου). Στην καλύτερη περίπτωση μόνο το πακέτο με αριθμό k=N αναμεταδίδεται (το τελευταίο εντός του παραθύρου μήκους N), ενώ στη χειρότερη περίπτωση έχουμε αποτυχία μετάδοσης στο k=1 οπότε αποστέλλουμε όλα τα πακέτα 1 ως N ξανά. Για το πρωτόκολλο GBN, η χρήση περιορίζεται πάντα από το μέγιστο: $u \leq \frac{N}{2BD+1} \leq 1$.



3. Σενάριο προσομοίωσης

Τοπολογία:



Πηγαίος κώδικας:

```
# Create a simulator object
set ns [new Simulator]

# Set the nam trace file
set nf [open lab3.nam w]
$ns namtrace-all $nf

# Set the trace file
set trf [open lab3.tr w]
$ns trace-all $trf

# Create the nodes
for {set i 0} {$i < 4} {incr i} {
set n($i) [$ns node]
}

$ns at 0.0 "$n(0) label GBN_sender_3117176"
$ns at 0.0 "$n(3) label GBN_receiver_3117176"
$ns at 0.0 "$n(1) label SW_sender_3117176"
$ns at 0.0 "$n(2) label SW_receiver_3117176"
```

```
# Create a duplex link between the nodes
for {set i 0} {$i < 4} {incr i} {
$ns duplex-link $n($i) $n([expr ($i+1)%4]) 2Mb 48ms DropTail
# Sets the queue limit of the two simplex links to the number specified.
$ns queue-limit $n($i) $n([expr ($i+1)%4]) 150
$ns queue-limit $n([expr ($i+1)%4]) $n($i) 150
}

$ns duplex-link-op $n(0) $n(1) orient right
$ns duplex-link-op $n(1) $n(2) orient down
$ns duplex-link-op $n(2) $n(3) orient left
$ns duplex-link-op $n(3) $n(0) orient up

# Define color index
$ns color 0 blue
$ns color 1 yellow

# Setup go-back-n sender-receiver
set tcp0 [new Agent/TCP/Reno]
$tcp0 set packetSize_ 2500
$tcp0 set window_ 8

# Disable modelling the initial SYN/SYNACK exchange
$tcp0 set syn_ false

# The initial size of the congestion window on slow-start
$tcp0 set windowInit_ 8

# Set flow ID
$tcp0 set fid_ 0
$ns attach-agent $n(0) $tcp0

set sink0 [new Agent/TCPSink]
$ns attach-agent $n(3) $sink0

$ns connect $tcp0 $sink0

set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0

# Setup stop-and-wait sender-receiver
set tcp1 [new Agent/TCP/Reno]
$tcp1 set packetSize_ 2500
$tcp1 set window_ 1
```

```
# Disable modelling the initial SYN/SYNACK exchange
$tcp1 set syn_ false

# The initial size of the congestion window on slow-start
$tcp1 set windowInit_ 1

# Set flow ID
$tcp1 set fid_ 1
$ns attach-agent $n(1) $tcp1

set sink1 [new Agent/TCPSink]
$ns attach-agent $n(2) $sink1

$ns connect $tcp1 $sink1

set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1

$tcp0 set window_ 8

#$tcp1 set window_ 1

# Define a 'finish' procedure
proc finish {} {
global ns nf trf
$ns flush-trace
# Close the trace file
close $nf
close $trf
exit 0
}

# Events
$ns at 1.0 "$ftp0 produce 120"
$ns at 1.0 "$ftp1 produce 120"
$ns at 12.0 "finish"

# Run the simulation
$ns run
```

4. Ανάλυση αποτελεσμάτων με τη βοήθεια του NAM

Ερωτήσεις:

- *Με τη βοήθεια του NAM, εντοπίστε τη χρονική στιγμή που ολοκληρώνεται η μετάδοση των 120 πακέτων FTP στην περίπτωση του πρωτοκόλλου: (i) Go back N και (ii) Stop and Wait. Εάν δεν επαρκεί ο δεδομένος χρόνος προσομοίωσης, αυξήστε καταλλήλως για να εντοπίσετε το ζητούμενο.*
  Η ολοκλήρωση της μετάδοσης των πακέτων παρατηρείται τις ακόλουθες χρονικές στιγμές:
  - ✓ Για το μεν πρωτόκολλο Go Back N, η αποστολή ολοκληρώνεται τη στιγμή 2.617859, ενώ το τελευταίο πακέτο επιβεβαίωσης στέλνεται τη στιγμή 2.668599.
  - ✓ Για το δε πρωτόκολλο Stop and Wait η αποστολή του τελευταίου πακέτου πραγματοποιείται τη στιγμή 13.711934.

- *Ποιο είναι το ελάχιστο μέγεθος παραθύρου εκπομπής (Nopt) που εξασφαλίζει ελάχιστο χρόνο μετάδοσης του συνόλου των πακέτων στο πρωτόκολλο Go back N;*
  Το μέγεθος του ελαχίστου παραθύρου εκπομπής κατά το οποίο επιτυγχάνεται και ο ελάχιστος συνολικός χρόνος μετάδοσης υπολογίζεται με χρήση του τύπου N_{opt} = 2*BD + 1. Είναι γνωστό ότι τα πακέτα είναι μεγέθους (2.500+40=) 2540 bytes και επίσης υπάρχει επιπλέον επιβάρυνση 40 bytes λόγω των πακέτων επιβεβαίωσης. Για να υπολογιστεί το BD, υπολογίζονται πρώτα τα bytes που μπορούν να βρίσκονται κάθε στιγμή στη ζεύξη, η οποία έχει 48ms καθυστέρηση, τα οποία είναι $\frac{2\ Mb}{8}$*0,048 = 12.000 bytes, άρα BD = $\frac{12.000\ \text{bytes}}{2580\ \text{bytes}}$ = 4,651 πακέτα. Άρα N_{opt}=10,3, το οποίο στρογγυλοποιείται στο 11 ώστε να μην υπάρχει χρόνος στον οποίο δεν στέλνονται δεδομένα.

- *Με βάση το ελάχιστο αυτό μέγεθος παραθύρου που προσδιορίσατε στο προηγούμενο ερώτημα, τροποποιήστε τις εντολές:*
  ```
  $tcp0 set window_ X
  $tcp0 set windowInit_ X
  ```
  εκτελέστε την προσομοίωση και υπολογίστε τη χρονική στιγμή που ολοκληρώνεται η μετάδοση των 120 πακέτων FTP για το πρωτόκολλο Go back N με τη βοήθεια του NAM.
  Τη στιγμή 2.267866 ολοκληρώνεται η αποστολή των μπλε πακέτων, ενώ το τελευταίο πακέτο επιβεβαίωσης φτάνει τη στιγμή 2.314866.

- *Πόση είναι η μέγιστη καθυστέρηση διάδοσης της ζεύξης n(0)-n(3) ώστε το αρχικό μέγεθος παραθύρου (N=8) να οδηγεί σε συνεχή χρησιμοποίηση της ζεύξης (no idle time);*
  Για N=8, BD = $\frac{N-1}{2}$ = 3,5 πακέτα. Όμως BD = $\frac{\frac{2\ Mb}{8}*delay}{2580}$, άρα delay = 0,03612, δηλαδή η καθυστέρηση θα είναι 36 ms.

- *Εκτελέστε πάλι την προσομοίωση με το μέγεθος παραθύρου του πρωτοκόλλου Go back N που βρήκατε στο δεύτερο ερώτημα (Nopt), όταν η ζεύξη n(0)-n(3) έχει τετραπλάσια και υποδιπλάσια καθυστέρηση διάδοσης της αρχικής. Εντοπίστε τη χρονική στιγμή που ολοκληρώνεται η μετάδοση των 120 πακέτων FTP στον κόμβο n3 στις δύο αυτές περιπτώσεις. Τι παρατηρείτε;*
  Στην πρώτη περίπτωση που η καθυστέρηση της ζεύξης τετραπλασιάζεται ανάμεσά τους (192 ms), η στιγμή που ολοκληρώνεται η αποστολή πακέτων είναι 5,241622, ενώ το τελευταίο πακέτο επιβεβαίωσης φτάνει τη στιγμή 5,419429. Σε αυτήν την περίπτωση παρατηρείται ότι

το διάστημα στο οποίο η ζεύξη παραμένει πρακτικά αχρησιμοποίητη είναι αρκετά μεγάλο και περίπου ίσο με το χρόνο που απαιτείται για να φτάσουν οι επιβεβαιώσεις. Στην δεύτερη περίπτωση η καθυστέρηση της ζεύξης υποδιπλασιάζεται (24 ms), η στιγμή που ολοκληρώνεται η αποστολή πακέτων είναι 2,243300, ενώ το τελευταίο πακέτο επιβεβαίωσης φτάνει τη στιγμή 2,269238. Σε αυτήν την περίπτωση η ζεύξη ανάμεσά τους παραμένει φορτωμένη με πακέτα αδιάκοπα και όπως ήταν αναμενόμενο η αποστολή ολοκληρώνεται σε πολύ λιγότερο χρόνο.

5. Ανάλυση αποτελεσμάτων με τη βοήθεια του αρχείου ίχνους (trace file)

Πηγαίος κώδικας:

```
BEGIN {
data_0=0;

packets_0=0;
data_1=0;
packets_1=0;
}
/^r/&&/tcp/ {
flow_id = $8;
if (flow_id == 0) {
data_0 += $6;
packets_0++;
last_ts_0 = $2;
}
if (flow_id == 1) {
data_1 += $6;
packets_1++;
last_ts_1 = $2;
}
}
END {
printf("Total Data received for flow ID 0\t: %d Bytes\n", data_0);
printf("Total Packets received for flow ID 0\t: %d\n", packets_0);
printf("Last packet received for flow ID 0\t: %s sec\n", last_ts_0);
printf("Total Data received for flow ID 1\t: %d Bytes\n", data_1);
printf("Total Packets received for flow ID 1\t: %d\n", packets_1);
printf("Last packet received for flow ID 1\t: %s sec\n", last_ts_1);
}
```

Ερωτήσεις:
Θεωρώντας μέγεθος παραθύρου (i) N=8 και (ii) N=Nopt (όπως προσδιορίστηκε στο δεύτερο ερώτημα της ενότητας 4) με καθυστέρηση μετάδοσης σε όλες τις ζεύξεις ίση με d msec (όπου d=100 + ΑΜ mod 100 και ΑΜ ο αριθμός μητρώου σας δηλ. d=176), να απαντήσετε στα ακόλουθα σύμφωνα με τα δεδομένα του αρχείου ίχνους:

- *Ποιος είναι ο αριθμός των πακέτων που παρελήφθησαν, πόσα δεδομένα παρελήφθησαν από τον παραλήπτη κατά τη διάρκεια της προσομοίωσης για κάθε ροή κίνησης;*
  Παρατηρείται αμέσως, λόγω της μεγάλης καθυστέρησης και όπως φαίνεται παρακάτω, ότι ο χρόνος προσομοίωσης είναι αρκετά μικρός οπότε στέλνονται μόνο τα 30 από τα 120 πακέτα από τον κόμβο n1 (SW_sender). Για το λόγο αυτό αυξάνουμε τον χρόνο προσομοίωσης σε 50 sec.
  Για N = 8:

```
Total Data received for flow ID 0       : 304760 Bytes
Total Packets received for flow ID 0    : 120
Last packet received for flow ID 0      : 6.7066 sec
Total Data received for flow ID 1       : 304760 Bytes
Total Packets received for flow ID 1    : 120
Last packet received for flow ID 1      : 47.40908 sec
```

  Άρα συνολικά έχουμε 120 πακέτα και 304760 bytes και για τις δύο ροές.

  Για N = 11:

```
Total Data received for flow ID 0       : 304760 Bytes
Total Packets received for flow ID 0    : 120
Last packet received for flow ID 0      : 5.17364 sec
Total Data received for flow ID 1       : 304760 Bytes
Total Packets received for flow ID 1    : 120
Last packet received for flow ID 1      : 47.40908 sec
```

  Το μόνο που άλλαξε είναι ο χρόνος που χρειάστηκε για να ολοκληρωθεί η αποστολή των πακέτων στη ροή με *ID*=0, ο οποίος μειώθηκε.

- *Εξετάζοντας το αρχείο ίχνους, προσδιορίστε σε πόσο χρόνο απεστάλησαν αυτά τα δεδομένα στις δύο περιπτώσεις για κάθε ροή κίνησης. Ποιος ο μέσος ρυθμός μετάδοσης δεδομένων σε bps και ποια είναι η χρησιμοποίηση του καναλιού;*
  Με βάση τα συγκεκριμένα δεδομένα προκύπτουν τα εξής συμπεράσματα για τις δύο περιπτώσεις.

  Για N=**8**: Για τη ροή *SW* η αποστολή ολοκληρώθηκε τη στιγμή 6,7066, οπότε χρειάστηκαν 5,7066 sec ώστε να σταλούν όλα τα πακέτα, δεδομένου ότι ξεκινούν να στέλνονται πακέτα τη στιγμή 1. Άρα ο μέσος ρυθμός μετάδοσης θα είναι $\frac{304.760 \text{ bytes}}{5{,}7066 \text{ sec}}$ = 53.405 Bps = 427.238 bps. Η χρησιμοποίηση θα είναι $\frac{427.238 \text{ bps}}{2 \text{ Mbps}}$ = 21,4 %. Για τη ροή *GBN* η αποστολή ολοκληρώθηκε τη στιγμή 47,40908, οπότε χρειάστηκαν 46,40908 sec ώστε να σταλούν όλα τα πακέτα, δεδομένου ότι ξεκινούν να στέλνονται πακέτα τη στιγμή 1. Άρα ο μέσος ρυθμός μετάδοσης θα είναι $\frac{304.760 \text{ bytes}}{46{,}40908 \text{ sec}}$ = 6428 Bps = 51426 bps. Η χρησιμοποίηση θα είναι $\frac{51.426 \text{ bps}}{2 \text{ Mbps}}$ = 2,5 %.

  Για N=**11**: Για τη ροή *SW* η αποστολή ολοκληρώθηκε τη στιγμή 5,17364, οπότε χρειάστηκαν 4,17364 sec ώστε να σταλούν όλα τα πακέτα, δεδομένου ότι ξεκινούν να στέλνονται πακέτα τη στιγμή 1. Άρα ο μέσος ρυθμός μετάδοσης θα είναι $\frac{304.760 \text{ bytes}}{4{,}17364 \text{ sec}}$ = 73.020 Bps = 584.161 bps. Η χρησιμοποίηση θα είναι $\frac{584.161 \text{ bps}}{2 \text{ Mbps}}$ = 29,2 %. Για τη ροή *GBN* η αποστολή ολοκληρώθηκε

τη στιγμή 47,40908, οπότε χρειάστηκαν 46,40908 sec ώστε να σταλούν όλα τα πακέτα, δεδομένου ότι ξεκινούν να στέλνονται πακέτα τη στιγμή 1. Άρα ο μέσος ρυθμός μετάδοσης θα είναι $\frac{304.760 \text{ bytes}}{47{,}40908 \text{ sec}}$ = 6.428 Bps = 51.426 bps. Η χρησιμοποίηση θα είναι $\frac{51.426 \text{ bps}}{2 \text{ Mbps}}$ = 2,5 %.

- *Τροποποιείστε το script ώστε να υπολογίσετε το χρόνο λήψης της επιβεβαίωσης (τύπος πακέτου ack) του τελευταίου πακέτου στις δύο περιπτώσεις για κάθε ροή κίνησης. Επισυνάψτε στην απάντησή σας το τροποποιημένο script, καθώς και screenshot από το trace file όπου φαίνεται το εν λόγω record.*

  Πηγαίος κώδικας:

```
BEGIN {
data_0=0;
packets_0=0;
data_1=0;
packets_1=0;
}

/^r/&&/tcp/ {
flow_id = $8;
if (flow_id == 0) {
data_0 += $6;
packets_0++;
last_ts_0 = $2;
}
if (flow_id == 1) {
data_1 += $6;
packets_1++;
last_ts_1 = $2;
}
}
/^r/&&/ack/ {
flow_id = $8;
if (flow_id == 0) {
last_ack_0 = $2;
}
if (flow_id == 1) {
last_ack_1 = $2;
}
}
END {
printf("Total Data received for flow ID 0\t: %d Bytes\n", data_0);
printf("Total Packets received for flow ID 0\t: %d\n", packets_0);
printf("Last packet received for flow ID 0\t: %s sec\n", last_ts_0);
printf("Last ack received for flow ID 0\t: %s sec\n", last_ack_0);
printf("Total Data received for flow ID 1\t: %d Bytes\n", data_1);
printf("Total Packets received for flow ID 1\t: %d\n", packets_1);
printf("Last packet received for flow ID 1\t: %s sec\n", last_ts_1);
printf("Last ack received for flow ID 1\t: %s sec\n", last_ack_1);
}
```

Για N=8:

```
Total Data received for flow ID 0       : 304760 Bytes
Total Packets received for flow ID 0    : 120
Last packet received for flow ID 0      : 6.7066 sec
Last ack received for flow ID 0 : 6.89576 sec
Total Data received for flow ID 1       : 304760 Bytes
Total Packets received for flow ID 1    : 120
Last packet received for flow ID 1      : 47.40908 sec
Last ack received for flow ID 1 : 47.59824 sec
```

```
r 6.82464 3 0 ack 40 ------- 0 3.0 0.0 112 261
r 6.82464 2 1 ack 40 ------- 1 2.0 1.0 14 262
+ 6.82464 1 2 tcp 2540 ------- 1 1.0 2.0 15 270
- 6.82464 1 2 tcp 2540 ------- 1 1.0 2.0 15 270
r 6.8348 3 0 ack 40 ------- 0 3.0 0.0 113 263
r 6.84496 3 0 ack 40 ------- 0 3.0 0.0 114 264
r 6.85512 3 0 ack 40 ------- 0 3.0 0.0 115 265
r 6.86528 3 0 ack 40 ------- 0 3.0 0.0 116 266
r 6.87544 3 0 ack 40 ------- 0 3.0 0.0 117 267
r 6.8856 3 0 ack 40 ------- 0 3.0 0.0 118 268
r 6.89576 3 0 ack 40 ------- 0 3.0 0.0 119 269
r 7.0238 1 2 tcp 2540 ------- 1 1.0 2.0 15 270
+ 7.0238 2 1 ack 40 ------- 1 2.0 1.0 15 271
- 7.0238 2 1 ack 40 ------- 1 2.0 1.0 15 271
r 7.21296 2 1 ack 40 ------- 1 2.0 1.0 15 271
+ 7.21296 1 2 tcp 2540 ------- 1 1.0 2.0 16 272
- 7.21296 1 2 tcp 2540 ------- 1 1.0 2.0 16 272
```

```
r 46.63244 1 2 tcp 2540 ------- 1 1.0 2.0 117 474
+ 46.63244 2 1 ack 40 ------- 1 2.0 1.0 117 475
- 46.63244 2 1 ack 40 ------- 1 2.0 1.0 117 475
r 46.8216 2 1 ack 40 ------- 1 2.0 1.0 117 475
+ 46.8216 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
- 46.8216 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
r 47.02076 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
+ 47.02076 2 1 ack 40 ------- 1 2.0 1.0 118 477
- 47.02076 2 1 ack 40 ------- 1 2.0 1.0 118 477
r 47.20992 2 1 ack 40 ------- 1 2.0 1.0 118 477
+ 47.20992 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
- 47.20992 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
r 47.40908 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
+ 47.40908 2 1 ack 40 ------- 1 2.0 1.0 119 479
- 47.40908 2 1 ack 40 ------- 1 2.0 1.0 119 479
r 47.59824 2 1 ack 40 ------- 1 2.0 1.0 119 479
```

Για N=11:

```
Total Data received for flow ID 0       : 304760 Bytes
Total Packets received for flow ID 0    : 120
Last packet received for flow ID 0      : 5.17364 sec
Last ack received for flow ID 0 : 5.3628 sec
Total Data received for flow ID 1       : 304760 Bytes
Total Packets received for flow ID 1    : 120
Last packet received for flow ID 1      : 47.40908 sec
Last ack received for flow ID 1 : 47.59824 sec
```

```
- 5.17364 3 0 ack 40 ------- 0 3.0 0.0 119 261
r 5.27136 3 0 ack 40 ------- 0 3.0 0.0 110 251
r 5.27136 2 1 ack 40 ------- 1 2.0 1.0 10 252
+ 5.27136 1 2 tcp 2540 ------- 1 1.0 2.0 11 262
- 5.27136 1 2 tcp 2540 ------- 1 1.0 2.0 11 262
r 5.28152 3 0 ack 40 ------- 0 3.0 0.0 111 253
r 5.29168 3 0 ack 40 ------- 0 3.0 0.0 112 254
r 5.30184 3 0 ack 40 ------- 0 3.0 0.0 113 255
r 5.312 3 0 ack 40 ------- 0 3.0 0.0 114 256
r 5.32216 3 0 ack 40 ------- 0 3.0 0.0 115 257
r 5.33232 3 0 ack 40 ------- 0 3.0 0.0 116 258
r 5.34248 3 0 ack 40 ------- 0 3.0 0.0 117 259
r 5.35264 3 0 ack 40 ------- 0 3.0 0.0 118 260
r 5.3628 3 0 ack 40 ------- 0 3.0 0.0 119 261
r 5.47052 1 2 tcp 2540 ------- 1 1.0 2.0 11 262
+ 5.47052 2 1 ack 40 ------- 1 2.0 1.0 11 263
- 5.47052 2 1 ack 40 ------- 1 2.0 1.0 11 263
```

```
r 46.63244 1 2 tcp 2540 ------- 1 1.0 2.0 117 474
+ 46.63244 2 1 ack 40 ------- 1 2.0 1.0 117 475
- 46.63244 2 1 ack 40 ------- 1 2.0 1.0 117 475
r 46.8216 2 1 ack 40 ------- 1 2.0 1.0 117 475
+ 46.8216 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
- 46.8216 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
r 47.02076 1 2 tcp 2540 ------- 1 1.0 2.0 118 476
+ 47.02076 2 1 ack 40 ------- 1 2.0 1.0 118 477
- 47.02076 2 1 ack 40 ------- 1 2.0 1.0 118 477
r 47.20992 2 1 ack 40 ------- 1 2.0 1.0 118 477
+ 47.20992 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
- 47.20992 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
r 47.40908 1 2 tcp 2540 ------- 1 1.0 2.0 119 478
+ 47.40908 2 1 ack 40 ------- 1 2.0 1.0 119 479
- 47.40908 2 1 ack 40 ------- 1 2.0 1.0 119 479
r 47.59824 2 1 ack 40 ------- 1 2.0 1.0 119 479
```