

# Πρωτόκολλα και Αρχιτεκτονικές δικτύων

## Εργαστήριο 4

Τμήμα Πληροφορικής και τηλεπικοινωνιών  
Πανεπιστήμιο Ιωαννίνων, Άρτα

2023-05-02



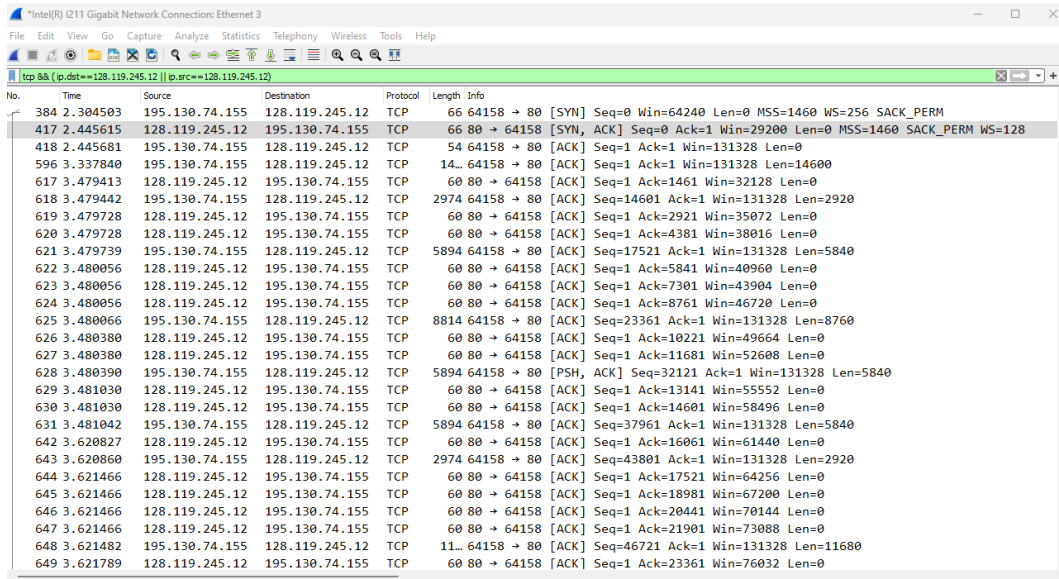
### Περιεχόμενα

1 Άσκηση <a href="http://gaia.cs.umass.edu">gaia.cs.umass.edu</a> - Πρώτη ματιά στο trace	2
2 Χαρακτηριστικά του TCP	3
3 Αλγόριθμος Συμφόρησης TCP	10

# 1 Άσκηση gaia.cs.umass.edu- Πρώτη ματιά στο trace

Για να πραγματοποιήσω capture των αντίστοιχων πακέτων θα πρέπει έπειτα από την επιλογή του κατάλληλου προσαρμογέα να εφαρμόσω και τα αντίστοιχα φίλτρα. Τα βήματα που θα ακολουθήσω είναι τα ακόλουθα:

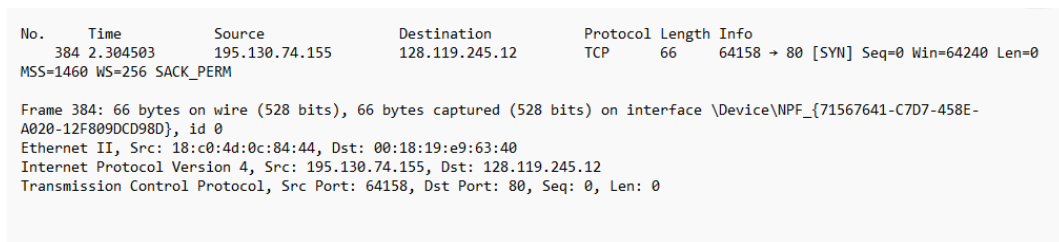
- nslookup www.gaia.cs.umass.edu
- Και εφαρμόστε τα φίλτρα της ακόλουθης εικόνας



No.	Time	Source	Destination	Protocol	Length	Info
384	2.304503	195.130.74.155	128.119.245.12	TCP	66	64158 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
417	2.445615	128.119.245.12	195.130.74.155	TCP	66	80 → 64158 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM WS=128
418	2.445681	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0
596	3.337840	195.130.74.155	128.119.245.12	TCP	14..	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=14600
617	3.479413	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=1461 Win=32128 Len=0
618	3.479442	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=14601 Ack=1 Win=131328 Len=2920
619	3.479728	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=2921 Win=35072 Len=0
620	3.479728	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=4381 Win=38016 Len=0
621	3.479739	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=17521 Ack=1 Win=131328 Len=5840
622	3.480056	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=5841 Win=40960 Len=0
623	3.480056	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=7301 Win=43904 Len=0
624	3.480056	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=8761 Win=46720 Len=0
625	3.480066	195.130.74.155	128.119.245.12	TCP	8814	64158 → 80 [ACK] Seq=23361 Ack=1 Win=131328 Len=8760
626	3.480380	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=10221 Win=49664 Len=0
627	3.480380	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=11681 Win=52608 Len=0
628	3.480390	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [PSH, ACK] Seq=32121 Ack=1 Win=131328 Len=5840
629	3.481030	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=13141 Win=55552 Len=0
630	3.481030	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=14601 Win=58496 Len=0
631	3.481042	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=37961 Ack=1 Win=131328 Len=5840
642	3.620827	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=16061 Win=61440 Len=0
643	3.620860	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=43801 Ack=1 Win=131328 Len=2920
644	3.621466	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=17521 Win=64256 Len=0
645	3.621466	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=18981 Win=67200 Len=0
646	3.621466	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=20441 Win=70144 Len=0
647	3.621466	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=21901 Win=73088 Len=0
648	3.621482	195.130.74.155	128.119.245.12	TCP	11..	64158 → 80 [ACK] Seq=46721 Ack=1 Win=131328 Len=11680
649	3.621789	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=23361 Win=76032 Len=0

Figure 1: Φίλτρα στο Wireshark

Η διαδικασία θα πρέπει να πραγματοποιηθεί αφού έχετε εκτελέσει τα βήματα που περιγράφονται στο αναφορά του εργαστηρίου για το upload του alice.txt. Έπειτα για πληροφορίες σχετικά με τα επιλεγμένα πακέτα επιλέγω στο wireshark *File → ExportPacketDissections → AsPlainText* και θα πρέπει να παρουσιάζεται αποτέλεσμα σαν της ακόλουθης εικόνας.



No.	Time	Source	Destination	Protocol	Length	Info
384	2.304503	195.130.74.155	128.119.245.12	TCP	66	64158 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

Frame 384: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF\_{71567641-C7D7-458E-A020-12F809DCD98D}, id 0

Ethernet II, Src: 18:c0:4d:0c:84:44, Dst: 00:18:19:e9:63:40

Internet Protocol Version 4, Src: 195.130.74.155, Dst: 128.119.245.12

Transmission Control Protocol, Src Port: 64158, Dst Port: 80, Seq: 0, Len: 0

Figure 2: Summary πακέτων σχετικά με gaia

Με βάση την εικόνα 2 η διεύθυνση ip του client είναι η 195.130.74.155 και το source port είναι το **64158**. Η διεύθυνση του παραλήπτη είναι γνωστή από το προηγούμενο ερώτημα και είναι η 128.119.245.12 και το destination port που χρησιμοποιεί είναι το 80 για να λαμβάνει segments μέσω tcp.

Ο στόχος της άσκησης είναι μία σειρά από TCP segments που ανταλλάσσονται μεταξύ του υπολογιστή σας και του gaia.cs.umass.edu. Στο υπόλοιπο μέρος αυτού του εργαστηρίου, θα χρησιμοποιήσουμε το trace των πακέτων που έχετε συλλάβει. Για να μπορέσετε να εντοπίσετε τέτοια πακέτα στο menu του Wireshark επιλέξτε *Analyze → EnabledProtocols* και ξεμαρκάρεται την επιλογή HTTP. Θα πρέπει να εμφανιστεί ένα trace αντίστοιχο της ακόλουθης εικόνας.

No.	Time	Source	Destination	Protocol	Length	Info
384	2.304503	195.130.74.155	128.119.245.12	TCP	66	64158 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=
418	2.445681	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=
596	3.337840	195.130.74.155	128.119.245.12	TCP	14...	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=
618	3.479442	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=14601 Ack=1 Win=13132
621	3.479739	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=17521 Ack=1 Win=13132
625	3.480066	195.130.74.155	128.119.245.12	TCP	8814	64158 → 80 [ACK] Seq=23361 Ack=1 Win=13132
628	3.480390	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [PSH, ACK] Seq=32121 Ack=1 Win=
631	3.481042	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=37961 Ack=1 Win=13132
643	3.620860	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=43801 Ack=1 Win=13132
648	3.621482	195.130.74.155	128.119.245.12	TCP	11...	64158 → 80 [ACK] Seq=46721 Ack=1 Win=13132
650	3.621798	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=58401 Ack=1 Win=13132
652	3.622121	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=61321 Ack=1 Win=13132
658	3.622462	195.130.74.155	128.119.245.12	TCP	14...	64158 → 80 [PSH, ACK] Seq=64241 Ack=1 Win=
660	3.622862	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=78841 Ack=1 Win=13132
666	3.623201	195.130.74.155	128.119.245.12	TCP	14...	64158 → 80 [ACK] Seq=81761 Ack=1 Win=13132
669	3.623608	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [PSH, ACK] Seq=96361 Ack=1 Win=
690	3.620860	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=102201 Ack=1 Win=1313
692	3.762784	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=105121 Ack=1 Win=1313
695	3.763103	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=108041 Ack=1 Win=1313
699	3.763423	195.130.74.155	128.119.245.12	TCP	8814	64158 → 80 [ACK] Seq=110961 Ack=1 Win=1313
702	3.763747	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=119721 Ack=1 Win=1313
707	3.764091	195.130.74.155	128.119.245.12	TCP	11...	64158 → 80 [PSH, ACK] Seq=125561 Ack=1 Win=
709	3.764402	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=137241 Ack=1 Win=1313
713	3.765055	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=140161 Ack=1 Win=1313
720	3.765380	195.130.74.155	128.119.245.12	TCP	7027	64158 → 80 [PSH, ACK] Seq=146001 Ack=1 Win=
771	3.950811	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=152974 Ack=778 Win=13
1525	8.911022	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=152974 Ack=779 Win=13

Figure 3: TCP πακέτα

## 2 Χαρακτηριστικά του TCP

1. Στο TCP, ο αριθμός SYN είναι ο αριθμός ακολουθίας του πρώτου πακέτου που αποστέλλεται από τον εκκινητή μιας νέας σύνδεσης. Χρησιμοποιείται για το συγχρονισμό των αριθμών ακολουθίας μεταξύ των δύο τελικών σημείων και τη δημιουργία ενός αξιόπιστου καναλιού επικοινωνίας. Για παράδειγμα, ας υποθέσουμε ότι ένας πελάτης θέλει να δημιουργήσει μια σύνδεση TCP με έναν διακομιστή. Ο πελάτης στέλνει ένα πακέτο με τη flag SYN ενεργοποιημένη, υποδεικνύοντας ότι θέλει να συγχρονίσει τους αριθμούς ακολουθίας και να δημιουργήσει μια νέα σύνδεση. Σε αυτό το πακέτο, ο πελάτης θέτει τον αριθμό SYN σε έναν αρχικό αριθμό ακολουθίας (ISN) που επιλέγει ο πελάτης. Ο διακομιστής απαντά με ένα πακέτο που έχει και τις σημαίες SYN και ACK ενεργοποιημένες, υποδεικνύοντας ότι επιβεβαιώνει το αίτημα του πελάτη για συγχρονισμό των αριθμών ακολουθίας και είναι έτοιμος να δημιουργήσει μια σύνδεση. Σε αυτό το πακέτο, ο διακομιστής θέτει τον αριθμό SYN στο δικό του ISN. Στη συνέχεια, ο πελάτης απαντά με ένα πακέτο που έχει τη flag ACK ενεργοποιημένη, υποδεικνύοντας ότι αναγνωρίζει την απάντηση του διακομιστή και είναι έτοιμος να ξεκινήσει την αποστολή δεδομένων. Σε αυτό το πακέτο, ο πελάτης θέτει τον αριθμό SYN στον επόμενο αριθμό ακολουθίας που θα χρησιμοποιήσει για το πρώτο πακέτο δεδομένων που θα στείλει. Ο αριθμός SYN είναι κρίσιμος για τη δημιουργία ενός αξιόπιστου καναλιού επικοινωνίας, καθώς εξασφαλίζει ότι και τα δύο τελικά σημεία χρησιμοποιούν συγχρονισμένους αριθμούς ακολουθίας για τη διάρκεια της σύνδεσης. Χωρίς συγχρονισμό, τα πακέτα θα μπορούσαν να χαθούν ή να παραδοθούν εκτός σειράς, με αποτέλεσμα αναξιόπιστη επικοινωνία. Ο αριθμός SYN είναι συνήθως μια τυχαία τιμή που επιλέγεται από το τελικό σημείο για να αποτρέψει τους επιτιθέμενους από το να προβλέψουν τους αριθμούς ακολουθίας και να εκμεταλλευτούν τη σύνδεση.

Για το παραδειγμά μας επιλέγουμε το πρώτο πακέτο και τα αποτελέσματα παρουσιάζονται στην ακόλουθη εικόνα

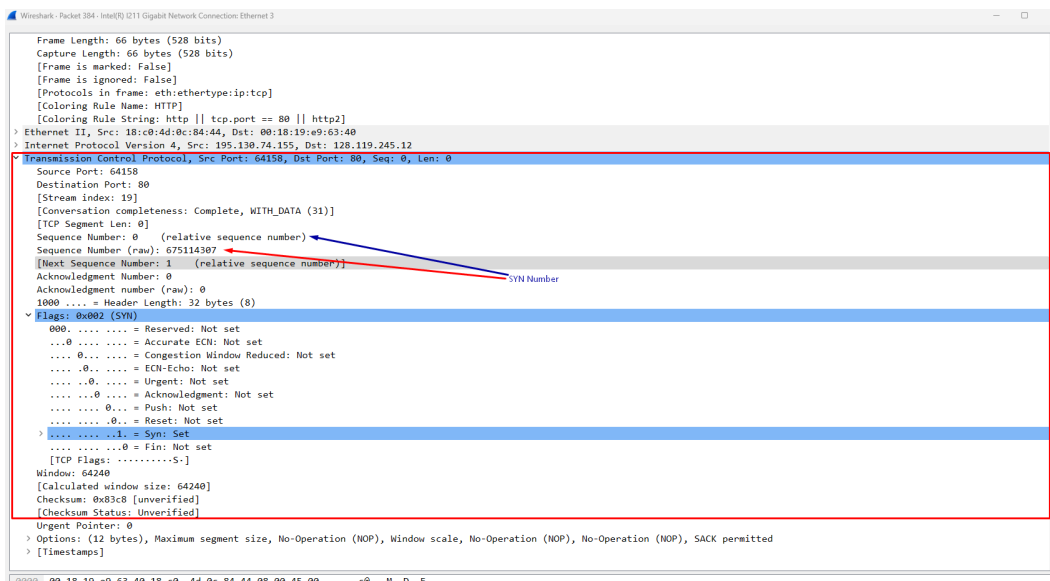


Figure 4: Αριθμός ακολουθίας

Σε ένα τμήμα TCP, η flag SYN στην επικεφαλίδα TCP χρησιμοποιείται για να υποδείξει ότι πρόκειται για τμήμα SYN. Η flag SYN είναι το δεύτερο bit στο πεδίο 6-bit TCP Control Flags της επικεφαλίδας TCP, αμέσως μετά τη flag URG. Όταν ένα τμήμα TCP αποστέλλεται για την έναρξη μιας νέας σύνδεσης, η flag SYN τίθεται σε 1 στην επικεφαλίδα TCP. Αυτό υποδεικνύει ότι το τμήμα είναι τμήμα SYN και ότι ο αποστολέας θέλει να συγχρονίσει τους αριθμούς ακολουθίας με τον παραλήπτη και να δημιουργήσει μια νέα σύνδεση. Όταν ο παραλήπτης λαμβάνει ένα τμήμα SYN, απαντά με ένα τμήμα SYN-ACK που έχει και τις σημαίες SYN και ACK ρυθμισμένες στην επικεφαλίδα TCP. Αυτό δείχνει ότι ο παραλήπτης έχει επιβεβαιώσει το αίτημα του αποστολέα για συγχρονισμό των αριθμών ακολουθίας και είναι έτοιμος να δημιουργήσει μια σύνδεση. Η χρήση των τμημάτων SYN αποτελεί βασικό χαρακτηριστικό της τριπλής χειραψίας του TCP, η οποία χρησιμοποιείται για τη δημιουργία ενός αξιόπιστου καναλιού επικοινωνίας μεταξύ δύο τελικών σημείων. Η χειραψία τριών κατευθύνσεων περιλαμβάνει την ανταλλαγή τριών τμημάτων: ένα τμήμα SYN, ένα τμήμα SYN-ACK και ένα τμήμα ACK, και εξασφαλίζει ότι και τα δύο τελικά σημεία συμφωνούν σχετικά με τους αρχικούς αριθμούς ακολουθίας και άλλες παραμέτρους σύνδεσης πριν από την έναρξη της μετάδοσης δεδομένων.

No.	Time	Source	Destination	Protocol	Length	Info
384	2.384503	195.130.74.155	128.119.245.12	TCP	66	64158 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
418	2.445681	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0
596	3.337840	195.130.74.155	128.119.245.12	TCP	14	64158 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=14600
618	3.479442	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=14601 Ack=1 Win=131328 Len=2920
621	3.479739	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=17521 Ack=1 Win=131328 Len=5840
625	3.480066	195.130.74.155	128.119.245.12	TCP	8814	64158 → 80 [ACK] Seq=23361 Ack=1 Win=131328 Len=8760
628	3.480390	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [PSH, ACK] Seq=32121 Ack=1 Win=131328 Len=5840
631	3.481042	195.130.74.155	128.119.245.12	TCP	5894	64158 → 80 [ACK] Seq=37961 Ack=1 Win=131328 Len=5840
643	3.620860	195.130.74.155	128.119.245.12	TCP	2974	64158 → 80 [ACK] Seq=43801 Ack=1 Win=131328 Len=2920
648	3.621482	195.130.74.155	128.119.245.12	TCP	11	64158 → 80 [ACK] Seq=46721 Ack=1 Win=131328 Len=11680

Figure 5: Flags στο Wireshark

- Επιλέγω το κατάλληλο πακέτο που είναι το δεύτερο κατά σειρά και τα αντίστοιχα αποτελέσματα παρουσιάζονται στην ακόλουθη εικόνα

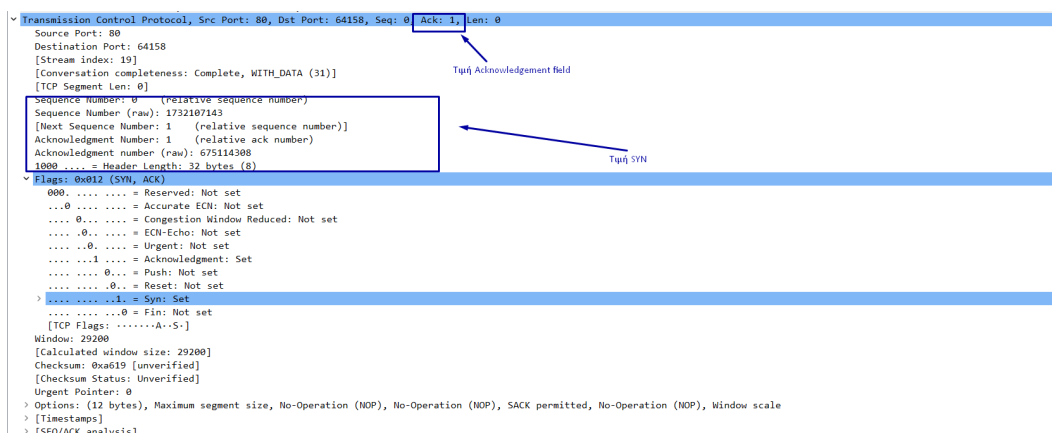


Figure 6: Gaia Response

Με ποιο τρόπο καθορίστηκε η τιμή αυτή από το `gaia.cs.umass.edu`

Στο TCP, το πεδίο ACK σε ένα τμήμα χρησιμοποιείται για την επιβεβαίωση της λήψης δεδομένων που έχουν ληφθεί σωστά. Η τιμή του πεδίου ACK σε ένα τμήμα καθορίζεται από τον παραλήπτη και υποδεικνύει τον επόμενο αριθμό ακολουθίας που αναμένει να λάβει από τον αποστολέα. Όταν ο αποστολέας στέλνει ένα τμήμα TCP με δεδομένα, θέτει τον αριθμό ακολουθίας στο πρώτο byte δεδομένων του τμήματος. Όταν ο παραλήπτης λαμβάνει το τμήμα, ελέγχει τον αριθμό ακολουθίας για να βεβαιωθεί ότι ταιριάζει με τον αναμενόμενο αριθμό ακολουθίας για τη σύνδεση. Εάν ο αριθμός ακολουθίας είναι σωστός, ο παραλήπτης επιβεβαιώνει τη λήψη των δεδομένων θέτοντας το πεδίο ACK στον επόμενο αριθμό ακολουθίας που αναμένει να λάβει. Για παράδειγμα, ας υποθέσουμε ότι ένας πελάτης θέλει να στείλει ένα μήνυμα σε έναν διακομιστή χρησιμοποιώντας το TCP και το μήνυμα έχει μήκος 500 bytes. Ο πελάτης χωρίζει το μήνυμα σε δύο τμήματα: το τμήμα 1 περιέχει τα πρώτα 300 bytes και το τμήμα 2 περιέχει τα υπόλοιπα 200 bytes. Όταν ο πελάτης στέλνει το τμήμα 1, θέτει τον αριθμό ακολουθίας σε 0, υποδεικνύοντας ότι το πρώτο byte του μηνύματος είναι ο αριθμός byte 0. Όταν ο διακομιστής λαμβάνει το τμήμα 1, ελέγχει τον αριθμό ακολουθίας και βλέπει ότι είναι 0, όπως αναμενόταν. Στη συνέχεια, ο διακομιστής θέτει το πεδίο ACK στο τμήμα απάντησής του σε 300, υποδεικνύοντας ότι αναμένει να λάβει το επόμενο byte, το οποίο είναι το byte με αριθμό 300. Όταν ο πελάτης στέλνει το τμήμα 2, θέτει τον αριθμό ακολουθίας σε 300, υποδεικνύοντας ότι το πρώτο byte αυτού του τμήματος είναι ο αριθμός byte 300. Όταν ο διακομιστής λαμβάνει το τμήμα 2, ελέγχει τον αριθμό ακολουθίας και βλέπει ότι είναι 300, όπως αναμενόταν. Στη συνέχεια, ο διακομιστής θέτει το πεδίο ACK στο τμήμα απάντησής του σε 500, υποδεικνύοντας ότι έλαβε όλα τα δεδομένα του μηνύματος. Θέτοντας το πεδίο ACK στον επόμενο αναμενόμενο αριθμό ακολουθίας, ο παραλήπτης επιβεβαιώνει στον αποστολέα ότι τα δεδομένα έχουν ληφθεί και υποδεικνύει επίσης την ποσότητα δεδομένων που μπορεί να λάβει στη συνέχεια χωρίς να ξεχειλίζει ο απομονωτής του. Αυτό εξασφαλίζει ότι ο αποστολέας μπορεί να συνεχίσει να στέλνει δεδομένα χωρίς να υπερφορτώνει τον παραλήπτη ή το δίκτυο.

**Ποιο στοιχείο του segment προσδιορίζει ότι πρόκειται για ένα SYNACK segment;**

Στο TCP, ένα τμήμα SYN-ACK είναι ένα τμήμα απάντησης που αποστέλλεται από τον παραλήπτη για να επιβεβαιώσει τη λήψη ενός τμήματος SYN και να ξεκινήσει το τελευταίο βήμα της τριπλής χειραφσίας για την εγκαθίδρυση μιας σύνδεσης. Το τμήμα SYN-ACK αναγνωρίζεται από το συνδυασμό των flags SYN και ACK στην επικεφαλίδα TCP. Πιο συγκεκριμένα, σε ένα τμήμα TCP, η flag SYN στο πεδίο TCP Control Flags χρησιμοποιείται για να υποδείξει ότι το τμήμα είναι τμήμα SYN και η flag ACK χρησιμοποιείται για να υποδείξει ότι το τμήμα είναι επιβεβαίωση ενός προηγούμενως ληφθέντος τμήματος. Όταν ο παραλήπτης στέλνει ένα τμήμα SYN-ACK, θέτει και τις δύο σημαίες SYN και ACK στο πεδίο TCP Control Flags σε 1, υποδεικνύοντας ότι το τμήμα είναι τόσο SYN όσο και επιβεβαίωση. Το τμήμα SYN-ACK αποστέλλεται ως απάντηση σε ένα τμήμα SYN που ελήφθη από τον αποστολέα. Το τμήμα SYN αποστέλλεται από τον αποστολέα για να ξεκινήσει μια σύνδεση με τον παραλήπτη και έχει τη flag SYN σε 1 στο πεδίο TCP Control Flags. Όταν ο παραλήπτης λαμβάνει ένα τμήμα SYN, στέλνει ένα τμήμα SYN-ACK για να επιβεβαιώσει

σει τη λήψη του τμήματος SYN και να δηλώσει ότι είναι έτοιμος να δημιουργήσει μια σύνδεση. Ο παραλήπτης θέτει το πεδίο ACK στην επικεφαλίδα TCP στον επόμενο αριθμό ακολουθίας που αναμένει να λάβει από τον αποστολέα και θέτει τη flag SYN σε 1 για να συγχρονίσει τους αριθμούς ακολουθίας με τον αποστολέα. Μόλις ο αποστολέας λάβει το τμήμα SYN-ACK, στέλνει ένα τμήμα ACK ως απάντηση για να επιβεβαιώσει τη λήψη του τμήματος SYN-ACK και να ολοκληρώσει την τριμερή χειραψία. Το τμήμα ACK έχει τη flag ACK σε 1 στο πεδίο TCP Control Flags και περιλαμβάνει τον επόμενο αριθμό ακολουθίας που θα χρησιμοποιήσει ο αποστολέας στη σύνδεση TCP. Συνοπτικά, ο συνδυασμός των flags SYN και ACK στο πεδίο TCP Control Flags προσδιορίζει ένα τμήμα SYN-ACK, το οποίο είναι ένα τμήμα απάντησης που αποστέλλεται από τον παραλήπτη για να επιβεβαιώσει τη λήψη ενός τμήματος SYN και να ξεκινήσει το τελευταίο βήμα της τριπλής χειραψίας για τη δημιουργία μιας σύνδεσης TCP.

3. Για να βρω τα HTTP πακέτα πρέπει να ενεργοποιήσω μέσω του Wireshark την επιλογή HTTP (*Analyze → Enabled\_Protocols → Check\_HTTP*). Το πακέτο που θα εντοπίσετε θα πρέπει να είναι σαν της ακόλουθης εικόνας.

No.	Time	Source	Destination	Protocol	Length	Info
3132	18.640180	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=56941 Win=143104 Len=0
3133	18.640189	195.130.74.155	128.119.245.12	TCP	5894	54880 → 80 [ACK] Seq=122641 Ack=1 Win=131328 Len=5840 [TCP segment of a reass...
3134	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=58401 Win=146048 Len=0
3135	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=59861 Win=148992 Len=0
3136	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=61321 Win=151936 Len=0
3137	18.640513	195.130.74.155	128.119.245.12	TCP	8814	54880 → 80 [PSH, ACK] Seq=128481 Ack=1 Win=131328 Len=8760 [TCP segment of a ...
3138	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=62781 Win=154880 Len=0
3139	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=64241 Win=157696 Len=0
3140	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=65701 Win=160640 Len=0
3141	18.640844	195.130.74.155	128.119.245.12	TCP	8814	54880 → 80 [ACK] Seq=137241 Ack=1 Win=131328 Len=8760 [TCP segment of a reass...
3142	18.641157	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=67161 Win=163584 Len=0
3143	18.641157	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=68621 Win=166528 Len=0
3144	18.641165	195.130.74.155	128.119.245.12	TCP	5894	54880 → 80 [ACK] Seq=146001 Ack=1 Win=131328 Len=5840 [TCP segment of a reass...
3145	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=70081 Win=169472 Len=0
3146	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=71541 Win=172288 Len=0
3147	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=73001 Win=175232 Len=0
3148	18.641493	195.130.74.155	128.119.245.12	HTTP	1187	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
3149	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=74461 Win=178176 Len=0
3150	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=75921 Win=181120 Len=0
3151	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=77381 Win=183296 Len=0
3152	18.642139	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=78841 Win=183296 Len=0
3153	18.642139	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=80301 Win=183296 Len=0
3154	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=81761 Win=183296 Len=0
3155	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=83221 Win=183296 Len=0
3156	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=84681 Win=183296 Len=0
3157	18.642780	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=86141 Win=183296 Len=0
3158	18.642780	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=87601 Win=183296 Len=0

Figure 7: Πακέτο HTTP Post

Ο αριθμός ακολουθίας είναι ο ακόλουθος:

Transmission Control Protocol, Src Port: 64158, Dst Port: 80, Seq: 146001, Ack: 1, Len: 6973	
Source Port: 64158	
Destination Port: 80	
[Stream index: 19]	
[Conversation completeness: Complete, WITH_DATA (31)]	
[TCP Segment Len: 6973]	
Sequence Number:	146001 (relative sequence number)
Sequence Number (raw):	675260308
[Next Sequence Number:	152974 (relative sequence number)]
Acknowledgment Number:	1 (relative ack number)
Acknowledgment number (raw):	1732107144
0101 ... = Header Length: 20 bytes (5)	
Flags: 0x018 (PSH, ACK)	
Window: 513	
[Calculated window size: 131328]	
[Window size scaling factor: 256]	
Checksum: 0x83a8 [unverified]	
[Checksum Status: Unverified]	
Urgent Pointer: 0	

Figure 8: HTTP Sequence number

4. Για τα πρώτα έξι πακέτα παρατηρώ



No.	Time	Source	Destination	Protocol	Length	Info
384	2.304503	195.130.74	128.119.24	TCP	66	64158 > 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
768	3.909036	128.119.24	195.130.74	HTTP	831	HTTP/1.1 200 OK (text/html)
771	3.950811	195.130.74	128.119.24	TCP	54	64158 > 80 [ACK] Seq=152974 Ack=778 Win=130560 Len=0
1524	8.910931	128.119.24	195.130.74	TCP	60	80 > 64158 [FIN, ACK] Seq=778 Ack=152974 Win=192000 Len=0
1525	8.911022	195.130.74	128.119.24	TCP	54	64158 > 80 [ACK] Seq=152974 Ack=779 Win=130560 Len=0
1526	8.911121	195.130.74	128.119.24	TCP	54	64158 > 80 [FIN, ACK] Seq=152974 Ack=779 Win=130560 Len=0
1539	9.057098	128.119.24	195.130.74	TCP	60	80 > 64158 [ACK] Seq=779 Ack=152975 Win=192000 Len=0

Sequence Number	Acknowledgement Number
108041	1
110961	1
119721	1
125561	1
137241	1
140161	1

Table 1: Sequence και Acknowledgement numbers για τα πρώτα έξι πακέτα

5. Στην παράκατω εικόνα έχω τους χρόνους αποστολής και λήψης των Acknowledgement flags. Για να πραγματοποιήσω την ακόλουθη διαδικασία πρέπει να επιλέξω τα πακέτα από το HTTP και μετά επιλέγω από το μενού *File* → *ExportPacketDissections* → *AsCSV*

Time
3.763103
3.763423
3.763747
3.764091
3.764402
3.765055

6. Η τιμή του RTT για κάθε ένα από τα 6 Segments παρουσιάζεται στον ακόλουθο πίνακα. Τα segments που κατηγοριοποιούνται με flag ACK είναι 3.

RTT
0.141178000
0.141178000
0.141178000
0.141178000
0.141178000
0.141178000

Table 2: Acknowledgement RTT

7. Το μήκος των πακέτων παρουσιάζεται στην ακόλουθη εικόνα. Για να υπολογίσω το μήκος των πακέτων χρειάζεται να τα εξάγω όπως στο βήμα 3.

Length
2974
8814
5894
11734
2974
5894

Figure 9: Packet Length

8. Στο TCP, το μέγεθος της προσωρινής αποθήκευσης αναφέρεται συχνά ως μέγεθος παραθύρου, το οποίο αντιπροσωπεύει την ποσότητα δεδομένων που μπορεί να αποθηκεύσει ο παραλήπτης ανά πάσα στιγμή. Το μέγεθος του παραθύρου διαφημίζεται από τον παραλήπτη στην επικεφαλίδα TCP κάθε πακέτου που στέλνει πίσω στον αποστολέα. Στη συνέχεια, ο αποστολέας προσαρμόζει την ποσότητα δεδομένων που στέλνει με βάση το διαφημιζόμενο μέγεθος παραθύρου. Εάν το μέγεθος του παραθύρου είναι μικρό, ο αποστολέας θα επιβραδύνει το ρυθμό μετάδοσής του για να αποφύγει την υπερφόρτωση του απομονωτή του παραλήπτη. Αντίθετα, εάν το μέγεθος του παραθύρου είναι μεγάλο, ο αποστολέας μπορεί να μεταδώσει περισσότερα δεδομένα κάθε φορά, αυξάνοντας τη συνολική απόδοση της σύνδεσης. Επομένως, το μέγεθος του παραθύρου παίζει καθοριστικό ρόλο στον έλεγχο της ροής δεδομένων μεταξύ του αποστολέα και του παραλήπτη στο TCP.

Βασίζόμενος στα πακέτα που έχω επιλέξει προκύπτει ότι το ελάχιστο buffer size είναι 130560 bytes

No.	Time	Source	Destination	Protocol	Length	Info
3132	18.640180	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=56941 Win=143104 Len=0
3133	18.640189	195.130.74.155	128.119.245.12	TCP	5894	54880 → 80 [ACK] Seq=122641 Ack=1 Win=131328 Len=5840 [TCP segment of a reassembled PDU]
3134	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=58401 Win=146048 Len=0
3135	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=59861 Win=148992 Len=0
3136	18.640506	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=61321 Win=151936 Len=0
3137	18.640513	195.130.74.155	128.119.245.12	TCP	8814	54880 → 80 [PSH, ACK] Seq=128481 Ack=1 Win=131328 Len=8760 [TCP segment of a reassembled PDU]
3138	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=62781 Win=154880 Len=0
3139	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=64241 Win=157696 Len=0
3140	18.640834	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=65701 Win=160640 Len=0
3141	18.640844	195.130.74.155	128.119.245.12	TCP	8814	54880 → 80 [ACK] Seq=137241 Ack=1 Win=131328 Len=8760 [TCP segment of a reassembled PDU]
3142	18.641157	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=67161 Win=163584 Len=0
3143	18.641157	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=68621 Win=166528 Len=0
3144	18.641165	195.130.74.155	128.119.245.12	TCP	5894	54880 → 80 [ACK] Seq=146001 Ack=1 Win=131328 Len=5840 [TCP segment of a reassembled PDU]
3145	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=70801 Win=169472 Len=0
3146	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=71541 Win=172288 Len=0
3147	18.641482	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=73001 Win=175232 Len=0
3148	18.641493	195.130.74.155	128.119.245.12	HTTP	1187	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
3149	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=74461 Win=178176 Len=0
3150	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=75921 Win=181120 Len=0
3151	18.641807	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=77381 Win=183296 Len=0
3152	18.642139	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=78841 Win=183296 Len=0
3153	18.642139	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=80301 Win=183296 Len=0
3154	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=81761 Win=183296 Len=0
3155	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=83221 Win=183296 Len=0
3156	18.642455	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=84681 Win=183296 Len=0
3157	18.642780	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=86141 Win=183296 Len=0
3158	18.642780	128.119.245.12	195.130.74.155	TCP	60	80 → 54880 [ACK] Seq=1 Ack=87601 Win=183296 Len=0

Figure 10: Buffer size στα επιλεγμένα πακέτα

9. Όχι δεν υπάρχουν επαναμεταδιδόμενα segments στα trace. Για να υπήρχαν θα έπρεπε να είχαμε ένα trace όπως το παρακάτω.



No.	Time	Source	Destination	Protocol	Length	Info
738	3.806932	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=102201 Win=183296 Len=0
748	3.904106	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=105121 Win=183296 Len=0
750	3.904425	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=108041 Win=183296 Len=0
751	3.904892	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=110961 Win=183296 Len=0
752	3.905217	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=113881 Win=183296 Len=0
753	3.905534	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=116801 Win=183296 Len=0
754	3.905858	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=119721 Win=183296 Len=0
755	3.906185	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=122641 Win=183296 Len=0
756	3.906185	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=125561 Win=183296 Len=0
757	3.906510	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=128481 Win=183296 Len=0
758	3.906510	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=131401 Win=183296 Len=0
759	3.907051	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=134321 Win=183296 Len=0
760	3.907051	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=137241 Win=183296 Len=0
761	3.907375	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=140161 Win=183296 Len=0
762	3.907699	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=143081 Win=183296 Len=0
763	3.907699	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=146001 Win=183296 Len=0
764	3.908022	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=148921 Win=183296 Len=0
765	3.908022	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=150381 Win=186112 Len=0
766	3.908387	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=151841 Win=189056 Len=0
767	3.908710	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=1 Ack=152974 Win=192000 Len=0
768	3.909036	128.119.245.12	195.130.74.155	HTTP	831	HTTP/1.1 200 OK (text/html)
771	3.950811	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=152974 Ack=778 Win=130560 Len=0
1524	8.910931	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [FIN, ACK] Seq=778 Ack=152974 Win=192000 Len=0
1525	8.911022	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [ACK] Seq=152974 Ack=779 Win=130560 Len=0
1526	8.911121	195.130.74.155	128.119.245.12	TCP	54	64158 → 80 [FIN, ACK] Seq=152974 Ack=779 Win=130560 Len=0
1539	9.057098	128.119.245.12	195.130.74.155	TCP	60	80 → 64158 [ACK] Seq=779 Ack=152975 Win=192000 Len=0

Figure 11: Επαναμεταδιδόμενα segments

Σε τι είδους έλεγχο του trace βασίσατε την απάντησή σας στην ερώτηση αυτή;

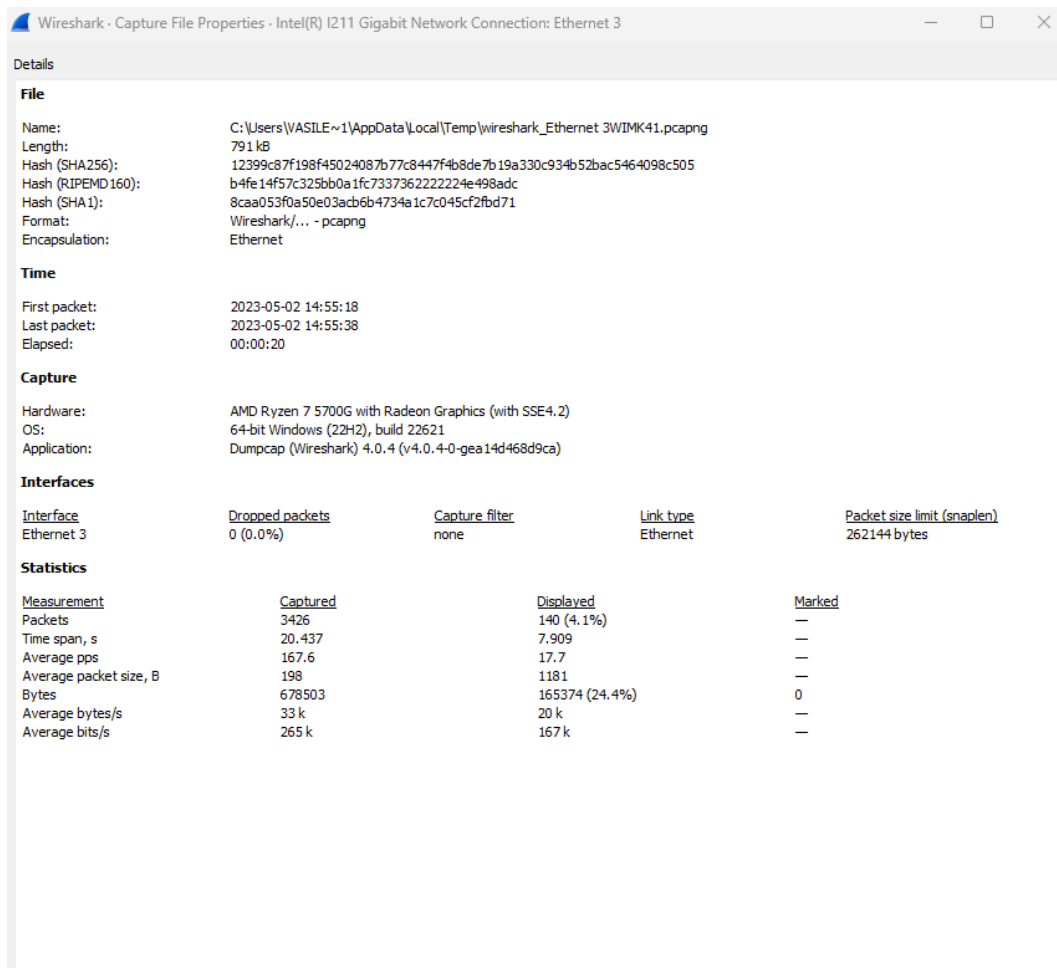
Ένας τρόπος για να προσδιορίσετε αν υπάρχουν επαναμεταδιδόμενα τμήματα σε ένα αρχείο ιχνών Wireshark είναι να αναζητήσετε πακέτα με διπλούς αριθμούς ακολουθίας. Αυτό δείχνει ότι το ίδιο τμήμα μεταδόθηκε πολλές φορές από τον αποστολέα. Επιπλέον, μπορείτε να αναζητήσετε πακέτα με τη σημαία επαναμετάδοσης (που επισημαίνεται ως "ReTransmitted" στο Wireshark) ρυθμισμένη στην επικεφαλίδα TCP, η οποία υποδεικνύει ότι το πακέτο είναι επαναμετάδοση ενός τμήματος που έχει αποσταλεί προηγουμένως. Για να επαληθεύσετε περαιτέρω ότι ένα πακέτο είναι επαναμετάδοση, μπορείτε να συγκρίνετε τον αριθμό ακολουθίας του πακέτου με τον αναγνωρισμένο αριθμό ακολουθίας που έλαβε ο παραλήπτης στο πακέτο ACK. Εάν ο αριθμός ακολουθίας του πακέτου είναι μικρότερος από τον αναγνωρισμένο αριθμό ακολουθίας, αυτό σημαίνει ότι το πακέτο επανεκπέμφθηκε από τον αποστολέα. Συνοπτικά, για να προσδιορίσετε αν υπάρχουν επαναμεταδιδόμενα τμήματα σε ένα αρχείο ιχνών Wireshark, μπορείτε να χρησιμοποιήσετε τον έλεγχο της ιχνής για να αναζητήσετε διπλούς αριθμούς ακολουθίας, πακέτα με ενεργοποιημένη τη σημαία επαναμετάδοσης και να συγκρίνετε τους αριθμούς ακολουθίας με τον αναγνωρισμένο αριθμό ακολουθίας.

10. Στο TCP, ο παραλήπτης επιβεβαιώνει συνήθως μια συνεχόμενη ακολουθία bytes, η οποία αντιπροσωπεύει το επόμενο byte που αναμένει να λάβει από τον αποστολέα. Η ποσότητα δεδομένων που αναγνωρίζεται σε μια επιβεβαίωση μπορεί να ποικίλλει ανάλογα με το διαφημιζόμενο μέγεθος παραθύρου του παραλήπτη και την ποσότητα δεδομένων που αποστέλλει ο αποστολέας. Συνήθως, ο παραλήπτης αναγνωρίζει όλα τα δεδομένα μέχρι και το τελευταίο συνεχόμενο byte που έλαβε σωστά. Για παράδειγμα, εάν ο δέκτης έχει λάβει σωστά τα bytes 1 έως 500 και αναμένει το byte 501 στη συνέχεια, θα αναγνωρίσει το byte 500 στο πακέτο επιβεβαίωσης. Αυτό σημαίνει ότι ο δέκτης αναγνωρίζει ένα εύρος 500 bytes σε μία επιβεβαίωση. Ωστόσο, σε ορισμένες περιπτώσεις, ο δέκτης μπορεί να μην αναγνωρίσει όλα τα δεδομένα μέχρι το τελευταίο συνεχόμενο byte που έλαβε σωστά. Αυτό μπορεί να συμβεί εάν ο απομονωτής του δέκτη είναι γεμάτος και δεν μπορεί να επεξεργαστεί άμεσα όλα τα εισερχόμενα δεδομένα. Σε τέτοιες περιπτώσεις, ο δέκτης μπορεί να αναγνωρίσει μόνο ένα μέρος των δεδομένων που έλαβε και να καθυστερήσει την αναγνώριση των υπολοίπων μέχρι να έχει περισσότερο διαθέσιμο χώρο στο buffer. Συνοπτικά, η ποσότητα των δεδομένων που αναγνωρίζονται σε μια επιβεβαίωση μπορεί να ποικίλλει ανάλογα με παράγοντες όπως το μέγεθος του ρυθμιστικού διαύλου του δέκτη, η ποσότητα των δεδομένων που αποστέλλει ο αποστολέας και η ικανότητα του δέκτη να επεξεργάζεται τα εισερχόμενα δεδομένα.

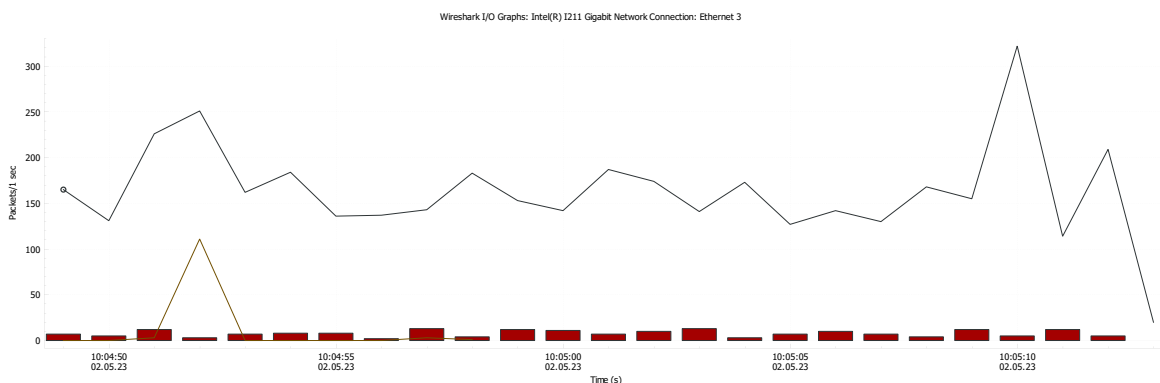
11. Η formula για υπολογισμό του Throughput είναι η ακόλουθη:

$$\text{Throughput} = \frac{\text{Total Data Size}}{\text{Total Time Taken}} \quad (1)$$

Για να υπολογίσω το Throughput των segments που έχω Φιλτράρει ακολουθώ επιλέγω από το menu *Statistics* → *CaptureFileProperties*:



Για γραφική αναπαράσταση των πακέτων ανά δευτερόλεπτο επιλέγω *Statistics* → *I/OGraphs*



### 3 Αλγόριθμος Συμφόρησης TCP

Στο Wireshark, μπορείτε να προσδιορίσετε πότε αρχίζει και τελειώνει η φάση αργής εκκίνησης του TCP, καθώς και πότε μεταβαίνει στη φάση αποφυγής συμφόρησης, αναλύοντας το παράθυρο συμφόρησης TCP (cwnd) και τον αριθμό των bytes που επιβεβαιώνονται (ack). Δείτε πώς μπορείτε να το κάνετε:

- Φίλτρο για κυκλοφορία TCP: Εφαρμόστε ένα φίλτρο για να εμφανίσετε μόνο την κυκλοφορία TCP στη σύλληψη του Wireshark. Μπορείτε να το κάνετε αυτό πληκτρολογώντας "tcp" στο πεδίο φίλτρου στο επάνω μέρος του παραθύρου Wireshark.
- Αναλύστε το παράθυρο συμφόρησης TCP (cwnd): Αναζητήστε το πακέτο TCP SYN (συγχρονισμού) στη σύλληψη και σημειώστε την αρχική τιμή του cwnd στην επικεφαλίδα TCP. Η φάση της αργής εκκίνησης αρχίζει με cwnd ενός τμήματος (συνήθως περίπου 1.460 bytes). Καθώς τα δεδομένα μεταδίδονται και αναγνωρίζονται, το cwnd αυξάνεται εκθετικά, διπλασιάζοντας με κάθε χρόνο διαδρομής γύρου μέχρι να επιτευχθεί ένα κατώφλι.
- Προσδιορίστε το τέλος της φάσης αργής εκκίνησης: Μπορείτε να εντοπίσετε το τέλος της φάσης αργής εκκίνησης αναζητώντας το πρώτο πακέτο στη λήψη που έχει τιμή cwnd που υπερβαίνει το κατώφλι αργής εκκίνησης (sssthresh). Το κατώφλι αργής εκκίνησης ορίζεται συνήθως σε μια τιμή μεταξύ 10 και 100 τμημάτων. Μόλις η τιμή cwnd υπερβεί το sssthresh, αρχίζει η φάση αποφυγής συμφόρησης.
- Αναλύστε τον αριθμό των bytes που αναγνωρίστηκαν (ack): Κατά τη διάρκεια της φάσης αποφυγής συμφόρησης, το cwnd αυξάνεται γραμμικά, προσθέτοντας ένα τμήμα για κάθε RTT. Για να προσδιορίσετε πότε τελειώνει η φάση αποφυγής συμφόρησης, αναζητήστε το πρώτο πακέτο στη λήψη που έχει τιμή cwnd που υπερβαίνει τον αριθμό των bytes που αναγνωρίστηκαν (ack) από την έναρξη της φάσης αποφυγής συμφόρησης.

Αναλύοντας το παράθυρο συμφόρησης TCP και τον αριθμό των bytes που αναγνωρίστηκαν, μπορείτε να προσδιορίσετε πότε αρχίζει και τελειώνει η φάση αργής εκκίνησης του TCP, καθώς και πότε γίνεται η μετάβαση στη φάση αποφυγής συμφόρησης.

