ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΓΚΟΥΜΕ ΛΑΟΥΡΕΝΤΙΑΝ
ΑΜ: 031 18 014
ΕΞΑΜΗΝΟ: 7^{ο}
ΟΜΑΔΑ: 4
MAC ADDRESS: B4-69-21-1B-6C-FF
IPv4: Ασκ1 και 2: 10.3.20.17, Ασκ4: 10.3.20.22 και 10.3.20.28, Άσκ3: 10.3.20.37
ΌΝΟΜΑ ΥΠΟΛΟΓΙΣΤΉ: LAPTOP-B2DVAJKK
ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ: WINDOWS 10

ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 4: ΠΡΩΤΟΚΟΛΛΟ IPv4 ΚΑΙ ΘΡΥΜΜΑΤΙΣΜΟΣ

Άσκηση 1: Μετρήστε την καθυστέρηση

1.1) Προκειμένου να αποστείλουμε 3 πακέτα IPv4/ICMP στον ιστότοπο **www.mit.edu**, χρησιμοποιήθηκε η εντολή **“ping -4 -n 3” www.mit.edu**. Αναλυτικότερα, το **“-4”** χρησιμοποιείται προκειμένου να επιβληθεί το πρωτόκολλο IPv4 κατά την αποστολή, ενώ το τμήμα **“-n 3”** δηλώνει την αποστολή 3 πακέτων.

1.2) Το φίλτρο “not multicast and not broadcast” **συλλαμβάνει μόνο unicast traffic**, είναι επομένως χρήσιμο προκειμένου να καθαρίσουμε το δίκτυο από τυχόν θορύβους και να δούμε το traffic από και προς τον υπολογιστή μας, χωρίς τα multicast/broadcast μηνύματα.

1.3) Το ποσοστό απωλειών είναι **μηδενικό**, ενώ η μέση καθυστέρηση ανέρχεται σε 35ms, όπως βλέπουμε παρακάτω:

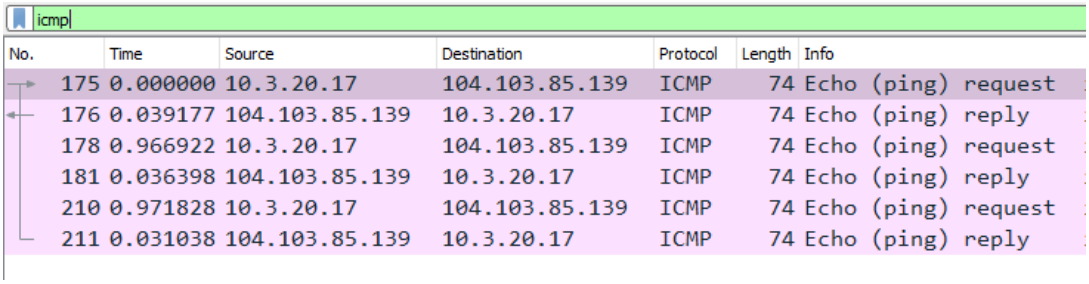
```
C:\Users\Άλεξ>ping -4 -n 3 www.mit.edu

Pinging e9566.dscb.akamaiedge.net [104.103.85.139] with 32 bytes of data:
Reply from 104.103.85.139: bytes=32 time=39ms TTL=59
Reply from 104.103.85.139: bytes=32 time=36ms TTL=59
Reply from 104.103.85.139: bytes=32 time=31ms TTL=59

Ping statistics for 104.103.85.139:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 31ms, Maximum = 39ms, Average = 35ms
```

1.4) Όπως βλέπουμε από το παρακάτω στιγμιότυπο τα RTT (Round Trip Times) είναι **Minimum = 31ms, Maximum = 39ms , Average = 35ms**.

1.5) Στο Wireshark, εφαρμόζουμε το display filter ICMP προκειμένου να μελετήσουμε ευκολότερα τα πακέτα που θέλουμε. Εκεί, βλέπουμε τις παρακάτω τιμές (αφού προηγουμένως έχουμε επιλέξει “Seconds Since Previous Displayed Format”):



Επομένως, βλέπουμε τους χρόνους **0.039177s ή 39.177ms, 0.036398s ή 36.398ms και 0.031038s 31.038ms**. Εύκολα παρατηρούμε, επομένως πως οι τιμές που κατέγραψε το Wireshark συμφωνούν με τις αντίστοιχες του παραθύρου εντολών.

1.6) Με το φίλτρο **“ip”**, εμφανίζονται μόνο πακέτα IPv4.

1.7) Αντίστοιχα, με το φίλτρο **“icmp”** παρατηρούμε την κίνηση ICMP που προκάλεσε η εντολή ping.

1.8) Το είδος των μηνυμάτων ICMP που στείλαμε κατά την εκτέλεση ping το βλέπουμε στο πεδίο Type των λεπτομερειών της επικεφαλίδας ICMP και αυτό είναι το **“Echo (ping) request”**, δηλαδή αίτημα.

1.9) Οι **IPv4 διευθύνσεις πηγής και προορισμού** των παρακάτω **αιτημάτων** είναι **10.3.20.17 και 104.103.85.139** αντίστοιχα.

1.10) Τα μηνύματα που έλαβε ο υπολογιστής μας είναι τύπου **“Echo (ping) reply”**.

1.11) Οι **IPv4 διευθύνσεις πηγής και προορισμού** των παρακάτω **απαντήσεων** είναι **104.103.85.139 και 10.3.20.17** αντίστοιχα.

1.12) Οι διαφορές σε σχέση με την καταγραφή του παρελθόντος είναι οι εξής:

- **Στείλαμε κατ’ εντολήν 3 πακέτα αντί για 4 by default.**
- **Είχαμε διαφορετικούς χρόνους round trip και μη ίσους μεταξύ τους.**
- **Το TTL (Time To Live) στην περίπτωσή μας ήταν 59 αντί για 242.**
- **Τέλος, η IP της σελίδας www.mit.edu είναι πλέον 104.103.85.139.**

Άσκηση 2: Περισσότερα για το Ping

2.1) Χρησιμοποιήθηκαν 3 διαδοχικά pings, η σύνταξη του καθενός ήταν η εξής: **“ping -4 -n 5 target_IP”** , όπου target_IP η εκάστοτε IP που χρειάστηκε για το κάθε ένα από τα ερωτήματα i, ii, iii. Εναλλακτικά, θα μπορούσε να χρησιμοποιηθεί μία εντολή, η εξής:
“ping -4 -n 5 10.3.20.1 && ping -4 -n 5 10.3.20.17 && ping -4 -n 5 127.0.0.1”.

2.2) Το **Wireshark έχει καταγράψει μόνο 5** από τα ICMP request που απέστειλε ο υπολογιστής μας.

2.3) Τα παραπάνω 5 αιτήματα είχαν ως αποδέκτη το **Gateway gate (router)**.

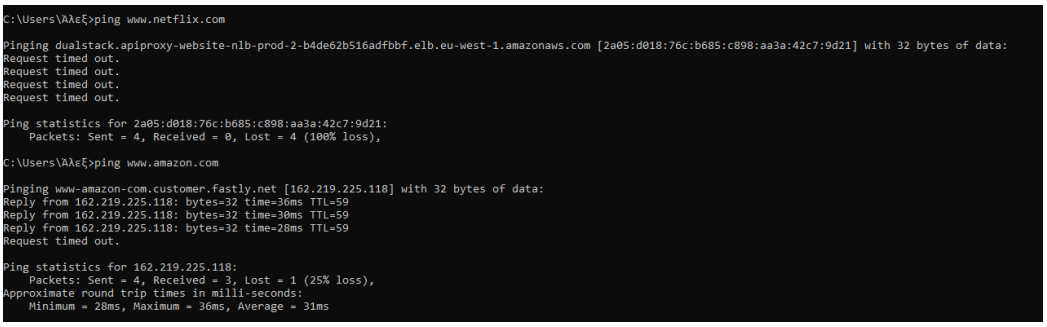
2.4) Δε παρατηρούμε αποστολή μηνυμάτων ICMP Echo Request στο δίκτυο με πηγή και προορισμό την IP του υπολογιστή μας. Αυτό είναι λογικό, καθώς

σύμφωνα με το σχήμα της εκφώνησης, εφόσον η IP μας είναι τοπική διεύθυνση IPv4, τοποθετείται στην ουρά εισόδου IPv4, επομένως δε φεύγει ποτέ προς το τοπικό δίκτυο.

2.5) Όπως και παραπάνω, **δε παρατηρούμε αποστολή μηνυμάτων ICMP Echo Request προς τη διεύθυνση του βρόχου επιστροφής (127.0.0.1)**, ακριβώς επειδή η διεύθυνση αυτή είναι τοπική, επομένως τα πακέτα δε προωθούνται ποτέ στο τοπικό δίκτυο και επομένως δε γίνονται Capture.

2.6) Κάνουμε ping στη διεύθυνσή μας προκειμένου να ελέγξουμε ότι η κάρτα δικτύου λειτουργεί κανονικά, επομένως, μπορούμε να το κάνουμε μόνο εφόσον είμαστε συνδεδεμένοι στο διαδίκτυο. Από την άλλη, **κάνουμε ping στη διεύθυνση βρόχου επιστροφής όταν θέλουμε να ελέγξουμε εάν είναι ορθά εγκατεστημένη η TCP/IP Stack**, κάτι που δεν απαιτεί συνδεσιμότητα στο διαδίκτυο. Επομένως, εάν θέλουμε να εκκινήσουμε διαδικασία troubleshooting, ξεκινάμε κάνοντας Ping σε αυτή τη διεύθυνση, ύστερα στην IPv4 μας εφόσον το προηγούμενο Ping μας έδωσε απάντηση, στη συνέχεια στο router μας και τέλος σε μια IP διεύθυνση εξωτερικού δικτύου, ώστε να δούμε ότι όλα πάνε καλά.

2.7) Παρατηρούμε τα εξής αποτελέσματα:



Δηλαδή, ενώ και οι 2 σελίδες τρέχουν κανονικά όταν τις πληκτρολογούμε στον browser, **δε λαμβάνουμε echo replies από το netflix παρά μόνο από την amazon**. Κάποιοι από τους λόγους για τους οποίους μπορεί να συμβαίνει αυτό είναι οι παρακάτω:

- Κατά τη διαδρομή των πακέτων από τον υπολογιστή μας στον server του Netflix, **υπάρχει περίπτωση να παρεμβάλλεται κάποιο firewall**, το οποίο μπλοκάρει τα μηνύματα πρωτοκόλλου ICMP, ενδεχομένως για να μη προκληθεί συμφόρηση των server από τέτοια – άρα χάνονται πακέτα κατά το request.
- Είναι επίσης πιθανό, ο κόμβος προορισμού ή κάποια ενδιάμεση συσκευή να **μην είναι επαρκώς πληροφορημένη για το δίκτυο του αποστολέα**, δηλαδή του υπολογιστή μας και έτσι να μην είναι δυνατή η σωστή επιστροφή της απάντησης – άρα χάνονται πακέτα κατά το reply.

Άσκηση 3: Επικεφαλίδες IPv4

3.1) Το φίλτρο σύλληψης που χρησιμοποιήθηκε είναι το **“host 147.102.40.15”**.

3.2) Για να δούμε τα πακέτα IPv4 που έστειλε ο υπολογιστής μας, πληκτρολογούμε στο πεδίο φίλτρου απεικόνισης **“ip.src==10.3.20.37”**.

3.3) Για τα πεδία της επικεφαλίδας των πακέτων IPv4, έχουμε:

- **Version τα 4 MSB του 1ου byte της επικεφαλίδας.**
- **Header Length τα 4 LSB του 1ου byte της επικεφαλίδας.**
- **Differentiated Services Field το 2ο byte (8 bits), το οποίο περιλαμβάνει τα υποπεδία Differentiated Services Codepoint (6 MSB του byte) και Explicit. Congestion Notification (2 LSB του byte).**
- **Total Length τα 2 επόμενα bytes (16 bits).**
- **Identification τα 2 επόμενα bytes (16 bits).**
- **Flags τα επόμενα 3 bits**.
- **Fragment Offset τα επόμενα 13 bits.**
- **Time to Live το επόμενο 1 byte (8 bits).**
- **Protocol το επόμενο 1 byte (8 bits).**
- **Header Checksum τα επόμενα 2 bytes (16 bits).**
- **Source Address τα επόμενα 4 bytes (32 bits).**
- **Destination Address τα επόμενα 4 bytes (32 bits).**

3.4) Τα πεδία με διαφορετικές τιμές μεταξύ πρώτου και τελευταίου πακέτου IPv4 που έστειλε ο υπολογιστής μας είναι: **Total Length, Identification, Header Checksum**.

3.5) Σε κάθε IPv4 πακέτο, το μήκος της επικεφαλίδας είναι **ίδιο (20 bytes)**.

3.6) Ελάχιστο μήκος πακέτου 40 bytes, ενώ **μέγιστο μήκος πακέτου 65 bytes**.

3.7) Το πεδίο Differentiated Services Field έχει τιμή **0x00** σε όλα τα πακέτα, το οποίο αντιστοιχεί στο **Standard Service Class** με **Default Forwarding** συμπεριφορά. Δηλαδή, traffic, το οποίο δε πληροί προϋποθέσεις κάποιας άλλης κλάσης χρησιμοποιεί τη συγκεκριμένη **(best effort forwarding χαρακτηριστικά)** που δεν εξασφαλίζει, ωστόσο ότι τα δεδομένα θα διαδοθούν ή ότι θα διαδοθούν αξιόπιστα.

3.8) Για το πεδίο Identification, παρατηρούμε ότι από το πρώτο μέχρι το τελευταίο IPv4 πακέτο που στέλνουμε **οι τιμές αυξάνονται κατά 1 σε κάθε πακέτο** (πρώτο πακέτο $0xae32_{16} = 44594_{10}$ και τελευταίο $0xae62_{16} = 44642_{10}$).

3.9) Η σημαία **Don’t Fragment** έχει σε κάθε πακέτο τιμή **1 (Set)**.

3.10) Το πεδίο **Fragment Offset** έχει παντού **τιμή 0**.

3.11) Σε κάθε πακέτο, το πεδίο **Protocol** έχει τιμή **0x06** και αντιστοιχεί στο **πρωτόκολλο TCP**.

3.12) Για να υπολογίσουμε το header checksum (ως αποστολέας), χωρίζουμε το IP header σε 16bit λέξεις και αφού τις αθροίσουμε, παίρνουμε το συμπλήρωμα ως προς ένα του αθροίσματος (με κάποια έξτρα βήματα εάν προκύψει κρατούμενο και έχοντας θεωρήσει ως 0000_{16} το πεδίο του header checksum). Επομένως, αντιλαμβανόμαστε πως το **header checksum βασίζεται στις τιμές των πεδίων της IP επικεφαλίδας του εκάστοτε πακέτου, κάποιες από τις οποίες όπως είδαμε διαφέρουν** (Total Length εν γένει, αλλά όχι απαραίτητα διαφορετικό, αλλά Identification μοναδικό σε κάθε πακέτο) **δίνοντας έτσι μοναδικό checker headsum σε κάθε πακέτο**.

Άσκηση 4: Θρυμματισμός (Fragmentation) στο IPv4

4.1) Προκειμένου να στείλουμε **ένα μόνο** πακέτο **IPv4** συγκεκριμένου **μεγέθους size** στη **διεύθυνση ip_address** χωρίς αυτό να θρυμματιστεί, πληκτρολογούμε την εντολή: **ping -4 -n 1 -l size -f ip_address** . *Επεξήγηση: -4 → για να εφαρμοστεί IPv4 πρωτόκολλο, -n 1 → αποστολή ενός πακέτου, -l size → καθορίζεται το μέγεθος του πακέτου, -f → κάνουμε set τη σημαία μη θρυμματισμού*

4.2) Θα επιχειρήσουμε αποστολή πακέτων στο router μας, με IP διεύθυνση 10.3.20.1. Δοκιμάζοντας μέγεθος πακέτου 1480 bytes, λαμβάνουμε το εξής μήνυμα “Packet needs to be fragmented but DF set”, όπως φαίνεται παρακάτω.

```
C:\Users\Αλεξ>ping -4 -n 1 -l 1480 -f 10.3.20.1

Pinging 10.3.20.1 with 1480 bytes of data:
Packet needs to be fragmented but DF set.

Ping statistics for 10.3.20.1:
    Packets: Sent = 1, Received = 0, Lost = 1 (100% loss),
```

Επομένως μειώνουμε σταδιακά το μέγεθος κατά 1 byte σε κάθε ping μέχρι να φτάσουμε στο **μέγιστο μήκος που δεν απαιτεί διάσπαση**, το οποίο βρίσκουμε έτσι ίσο με **1472 bytes,** όπως επίσης βλέπουμε παρακάτω:

```
C:\Users\Αλεξ>ping -4 -n 1 -l 1473 -f 10.3.20.1

Pinging 10.3.20.1 with 1473 bytes of data:
Packet needs to be fragmented but DF set.

Ping statistics for 10.3.20.1:
    Packets: Sent = 1, Received = 0, Lost = 1 (100% loss),

C:\Users\Αλεξ>ping -4 -n 1 -l 1472 -f 10.3.20.1

Pinging 10.3.20.1 with 1472 bytes of data:
Reply from 10.3.20.1: bytes=1472 time=6ms TTL=255

Ping statistics for 10.3.20.1:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 6ms, Average = 6ms
```

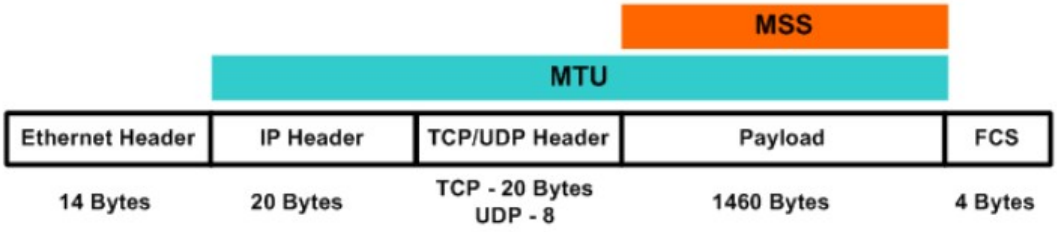
4.3) Προφανώς, η μικρότερη τιμή για την οποία **απαιτείται διάσπαση** είναι τα **1473 bytes**.

4.4) Χρησιμοποιήσαμε το **φίλτρο σύλληψης "not (multicast and broadcast)"**.

4.5) Το **φίλτρο απεικόνισης** που χρησιμοποιήσαμε **"ip.addr==10.3.20.1"**.

4.6) Το **Wireshark δε καταγράφει πακέτα IPv4 με την τιμή της ερώτησης 4.3**, διότι ενώ απαιτούν fragmentation για να αποσταλούν, έχουμε βάλει το option -f, το οποίο επιβάλλει να μη γίνει fragmentantion στο προς αποστολή πακέτο, με αποτέλεσμα αυτό να μη φεύγει ποτέ προς το τοπικό δίκτυο.

4.7) Το συνολικό μέγεθος των Ethernet frames που στείλαμε είναι 1514 bytes όπως καταγράφονται από το Wireshark. Ωστόσο, σύμφωνα με την παρακάτω εικόνα, το MTU δε περιλαμβάνει το Ethernet Header, αλλά ούτε και το FCS. Δεδομένου ότι το FCS ούτως η άλλως δε καταγράφεται από το Wireshark, το **MTU** (Maximum Transmission Unit), είναι ίσο με 1514 – 14 = **1500 bytes**, όπου 14 bytes το ethernet header.



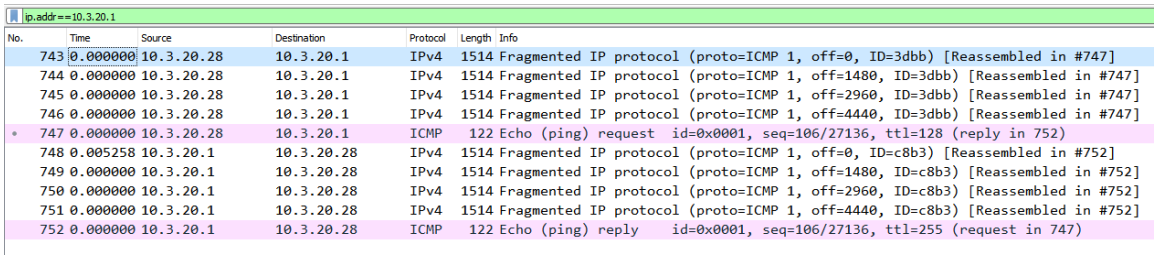
4.8) Αν δε θέλουμε fragmentation, τότε το ICMP πρέπει να ναι **1472 bytes** για IPv4 πακέτα μέγιστου μήκους. Αν, ωστόσο, επιτρέψουμε fragmentantion και επιχειρήσουμε να στείλουμε πακέτα πολλών χιλιάδων bytes, τότε το μέγιστο ICMP μπορεί να φτάσει **65.535 – 20 – 8 = 65.507 bytes**, όπου 20 bytes το IP header και 8 bytes το ICMP header. Παρόλα αυτά, το λειτουργικό μας σύστημα επιτρέπει έως και 65.500 bytes, σε αντίθεση με ένα σύστημα Linux που επιτρέπει έως και 65.507.

4.9) Για 65.507 bytes ICMP πακέτου, η εντολή **αποτυγχάνει**. Η μέγιστη τιμή, για την οποία πετυχαίνει είναι **65.500 bytes**.

4.10) Το μέγεθος του μεγαλύτερου IPv4 πακέτου που μπορεί να παράγει η εντολή ping είναι επομένως **65.528 bytes**.

4.11) Το μήνυμα ICMP Echo Request **δε μεταφέρθηκε ως ένα πακέτο IPv4**.

4.12) Όπως παρατηρούμε στο παρακάτω Screenshot, χρειάστηκαν **5 πακέτα IPv4 για την αποστολή του Ping** και αυτό, καθώς όπως είδαμε μπορούμε να στείλουμε μη θρυμματισμένα ICMP πακέτα μέγιστου μήκους 1.472 bytes.



ip.addr==10.3.20.1

No.	Time	Source	Destination	Protocol	Length	Info
743	0.000000	10.3.20.28	10.3.20.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=3dbb) [Reassembled in #747]
744	0.000000	10.3.20.28	10.3.20.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=3dbb) [Reassembled in #747]
745	0.000000	10.3.20.28	10.3.20.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=3dbb) [Reassembled in #747]
746	0.000000	10.3.20.28	10.3.20.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4440, ID=3dbb) [Reassembled in #747]
747	0.000000	10.3.20.28	10.3.20.1	ICMP	122	Echo (ping) request id=0x0001, seq=106/27136, ttl=128 (reply in 752)
748	0.005258	10.3.20.1	10.3.20.28	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=c8b3) [Reassembled in #752]
749	0.000000	10.3.20.1	10.3.20.28	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=c8b3) [Reassembled in #752]
750	0.000000	10.3.20.1	10.3.20.28	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=c8b3) [Reassembled in #752]
751	0.000000	10.3.20.1	10.3.20.28	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4440, ID=c8b3) [Reassembled in #752]
752	0.000000	10.3.20.1	10.3.20.28	ICMP	122	Echo (ping) reply id=0x0001, seq=106/27136, ttl=255 (request in 747)

Το **καθένα από τα πακέτα που στείλαμε είχε μέγεθος 1480 bytes** όσον αφορά το ICMP (Total Length (1514) = Ethernet Header (14) + IP Header (20) + ICMP packet (1480)), **εκτός του τελευταίου** το οποίο έχει συνολικά 122 bytes και αφαιρώντας 34 όπως πάνω μας μένουν 88 bytes. Ωστόσο, το τελευταίο αυτό θραύσμα φέρει και το ICMP header, επομένως αφαιρώντας άλλα 8 bytes παίρνουμε 80 bytes, τα οποία σε συνδυασμό με 4 * 1.480 = 5.920 bytes μας δίνουν το συνολικό μήκος των 6.000 bytes.

4.13) Καταγράφουμε τις εξής πληροφορίες:

- **Θραύσμα 743: Identification = 0x3dbb, Don’t Fragment Bit = 0 (Not set), More Fragments Bit = 1 (Set), Fragment Offset = 0**
- **Θραύσμα 744: Identification = 0x3dbb, Don’t Fragment Bit = 0 (Not set), More Fragments Bit = 1 (Set), Fragment Offset = 1.480**
- **Θραύσμα 745: Identification = 0x3dbb, Don’t Fragment Bit = 0 (Not set), More Fragments Bit = 1 (Set), Fragment Offset = 2.960**
- **Θραύσμα 746: Identification = 0x3dbb, Don’t Fragment Bit = 0 (Not set), More Fragments Bit = 1 (Set), Fragment Offset = 4.440**
- **Θραύσμα 747: Identification = 0x3dbb, Don’t Fragment Bit = 0 (Not set), More Fragments Bit = 0 (Not Set), Fragment Offset = 5.920**

4.14) Το **Don’t Fragment Bit**, το οποίο έχει τιμή 0 (Not Set), δηλώνει πως το πακέτο έχει θρυμματιστεί.

4.15) Το **πεδίο Fragment Offset** μάς δείχνει πως πρόκειται για το πρώτο θραύσμα, καθώς έχει τιμή 0.

4.16) Το συνολικό μήκος του πρώτου θραύσματος είναι **1.514 bytes**.

4.17) Ξανά, το πεδίο **Fragment Offset** μάς δείχνει πως δε πρόκειται για το πρώτο θραύσμα, καθώς έχει τιμή 1.480 (όσο το μέγεθος του ICMP payload του προηγούμενου πακέτου).

4.18) Ναι, ακολουθούν και άλλα θραύσματα.

4.19) Από τις πληροφορίες της επικεφαλίδας του 2ου θραύσματος, συμπεραίνουμε πως υπάρχουν και άλλα θραύσματα, καθώς το **More Fragments Bit έχει τιμή 1(Set)**.

4.20) Τα πεδία της IPv4 επικεφαλίδας, τα οποία διαφοροποιούνται μεταξύ πρώτου και δεύτερου θραύσματος είναι τα πεδία **Fragment Offset και Header Checksum**.

4.21) Σχετικά με το προτελευταίο θραύσμα, το οποίο βλέπουμε πως έχει τιμή για το πεδίο Fragment Offset 4.440, διαπιστώνουμε πως **ο αριθμός αυτός ισούται με όσα δεδομένα (σε bytes) ICMP έχουν αποσταλεί πριν από αυτό** (3 * 1.480 = 4.440). Ακριβώς με την ίδια λογική, όσον αφορά το τελευταίο πακέτο, **έχουν αποσταλεί προηγουμένως 4 θραύσματα μεγέθους 1.480 bytes** (συνολικό ICMP μέγεθος), άρα 4 * 1.480 = 5.920.

4.22) Κάθε θραύσμα έχει διαφορετικό **header checksum καθώς και διαφορετικό fragment offset**. Επιπλέον, όσον αφορά τα **Flags**, τα 2 πρώτα θραύσματα έχουν τιμές **0x20**, το 3ο **0x21**, το 4ο **0x22** ενώ το τελευταίο **0x02**. Τέλος, τα 4 πρώτα θραύσματα έχουν **More Fragments Bit = 1 (Set), ενώ το τελευταίο 0 (Not set)**.