ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ

$4^{Ο}$ ΕΡΓΑΣΤΗΡΙΟ

Σεραφείμ Τζελέπης AM:el18849,

Huawei MateBook 14, Windows 10,

Mac Address: 5C-3A-45-DC-95-1D,

Ομάδα : 4

1)

1.1: >ping /4 /n 3 www.mit.edu

1.2: Με το φίλτρο σύλληψης που εφαρμόσαμε ουσιαστικά κάνουμε capture μόνο τα unicast πακέτα από και προς την συσκευή μας.

1.3: Το ποσοστό απωλειών είναι 0% και ο μέσος χρόνος καθυστέρησης είναι 45ms.

1.4:

```
C:\Users\sertz>ping /4 /n 3 www.mit.edu

Pinging e9566.dscb.akamaiedge.net [23.79.130.108] with 32 bytes of data:
Reply from 23.79.130.108: bytes=32 time=47ms TTL=57
Reply from 23.79.130.108: bytes=32 time=45ms TTL=57
Reply from 23.79.130.108: bytes=32 time=44ms TTL=57
```

1.5:

Time1: 0.047638s

Time2: 0.045459s

Time3: 0.044537s

Παρατηρούμε ότι οι χρόνοι είναι προσεγγιστικά ίδιοι με αυτούς που έβγαλε το παράθυρο εντολών.

1.6: Το φίλτρο απεικόνισης ip.

1.7: Το φίλτρο που θα χρησιμοποιήσουμε είναι το: icmp and ip.addr == 23.79.130.108. Όπου 23.79.130.108 η IPv4 address του www.mit.edu.

1.8: Στάλθηκαν IPv4 echo(ping) requests.

1.9: Src: 192.168.1.19, Dst: 23.79.130.108

1.10: Ελήφθησαν IPv4 echo(ping) replies.

1.11: Src: 23.79.130.108, Dst: 192.168.1.19

1.12: Έχουν αλλάξει τα εξής:

Η IPv4 address του www.mit.edu είναι 23.79.130.108 ενώ η παλιά ήταν 18.7.22.83.

Το TTL είναι 57 ενώ ήταν 242.

Έχουν αλλάξει οι χρόνοι RTT.

Τέλος τώρα στάλθηκαν 3 πακέτα ενώ πριν είχαν σταλθεί 4.

2)

2.1: ping /4 /n 5 192.168.1.1 && ping /4 /n 5 192.168.1.19 && ping /4 /n 5 127.0.0.1

2.2: Το Wireshark κατέγραψε 5 Echo (ping) requests από τον υπολογιστή μου.

2.3: Προορισμός τους ήταν η IPv4 address: 192.168.1.1 η διεύθυνση προκαθορισμένης πύλης δηλαδή.

2.4: Όχι. Αυτό συμβαίνει διότι βλέποντας και το σχήμα το πακέτο πάει από τον υπολογιστή μας στην κάρτα δικτύου(οδηγός ethernet) από όπου εφόσον δεν προορίζεται για πολλαπλή διανομή ή εκπομπή αλλά είναι τοπική διεύθυνση θα μπει στην ουρά εισόδου(οδηγός loopback) και από εκεί στον υπολογιστή μας, συνεπώς το πακέτο δεν θα βρεθεί ποτέ στο τοπικό δίκτυο από όπου θα μπορούσε να ληφθεί από το Wireshark.

2.5: Αντιστοίχως με το 2.4 το πακέτο μας από την έξοδο του υπολογιστή θα μπει στην ουρά εισόδου(οδηγός loopback) και συνεπώς δεν θα μπορέσει να ληφθεί από το Wireshark.

2.6: Όταν κάνουμε ping στην διεύθυνση 127.0.0.1 το πακέτο δεν θα φύγει ποτέ από τον υπολογιστή μας ασχέτως με το αν είμαστε ή όχι συνδεδεμένοι στο διαδίκτυο. Από την άλλη όταν κάνουμε ping στην ip address μας το πακέτο αρχικά θα μπει στον οδηγό ethernet και άρα πρέπει να είμαστε συνδεμένοι στο δίκτυο.

2.7: Και οι δυο ιστοσελίδες φορτώνουν κανονικά, ωστόσο το ping που κάνουμε στο www.amazon.com έχουμε replies στα requests μας, στην περίπτωση που κάνουμε ping στο www.netflix.com δεν δεχόμαστε κάποιο reply. Μια πιθανή εξήγηση είναι το υποδίκτυο του προορισμού να μπλοκάρει τα πακέτα ICMP για λόγους ασφάλειας από DoS Attacks.

3)

3.1: host 147.102.40.15

3.2: ip.src == 192.168.1.9

3.3:

Version: 4 bits

Header Length: 4 bits

Differentiated Services Field: 8 bits (Differentiated Services Codepoint: 6 bits, Explicit Congestion Notification: 2 bits)

Total Length: 16 bits

Identification: 16 bits

Flags: 3 bits

Fragment Offset: 13 bits

Time to Live: 8 bits

Protocol: 8 bits

Header Checksum: 16 bits

Source Address: 32 bits

Destination Address: 32 bits

3.4: Total Length έχει τις εξής διαφορετικές τιμές: 56, 40, 47, 66, 46, 54, 41, 43 κλπ. Επίσης η τιμή του identification αλλάζει ανά πακέτο όπως επίσης διαφορετικές τιμές συναντώνται στην επικεφαλίδα header checksum.

3.5: Ναι το Header Length είναι 20 bytes για όλα τα πακέτα.

3.6: Το μικρότερο που παρατηρούμε είναι 40 και το μεγαλύτερο 66.

3.7: Το Differentiated Services Field έχει τιμή 0 και ουσιαστικά αποτελεί έναν μηχανισμό για την διευκρίνηση και την διαχείριση της 'κίνησης' στο δίκτυο εξασφαλίζοντας έτσι την κατάλληλη λειτουργία του δικτύου ανάλογα και στις ανάγκες του.

3.8: Οι τιμές του identification είναι μοναδικές για το κάθε πακέτο και ακολουθούν αύξουσα πορεία(Το n + 1 πακέτο έχει ως identification την τιμή του identification του ν-οστού πακέτου + 1) .

3.9: Το Don't Fragment flag έχει τιμή 1

3.10: Το Fragment Offset έχει τιμή 0.

3.11: Το Protocol έχει τιμή 6 και αντιστοιχεί στο πρωτόκολλο TCP.

3.12: Το Header Checksum προκύπτει από το συμπλήρωμα ως προς 1 του αθροίσματος όλων των bytes του ip header συνεπώς αφού έχουμε ότι οι διάφορες τιμές των πεδίων του header διαφέρουν ανά πακέτο, είναι απολύτως λογικό να διαφέρει και η τιμή του checksum.

4)

4.1: > ping /4 /n 1 /f /l <size> <address> , όπου size το μέγεθος σε bytes των δεδομένων του request και όπου <address> η IPv4 διεύθυνση στην όποια κάνουμε request.

4.2: Χρησιμοποιώντας την εντολή ping /4 /n 1 /f /l 1472 192.168.1.1 , η αποστολή επιτυγχάνει και μάλιστα η τιμή 1472 είναι η μέγιστη τιμή για την όποια έχουμε επιτυχία.

4.3: Η ελάχιστη τιμή για την όποια απαιτείται θρυμματισμός είναι τα 1473 bytes.

4.4: Το φίλτρο σύλληψης που χρησιμοποιήθηκε είναι το “not broadcast and not multicast”

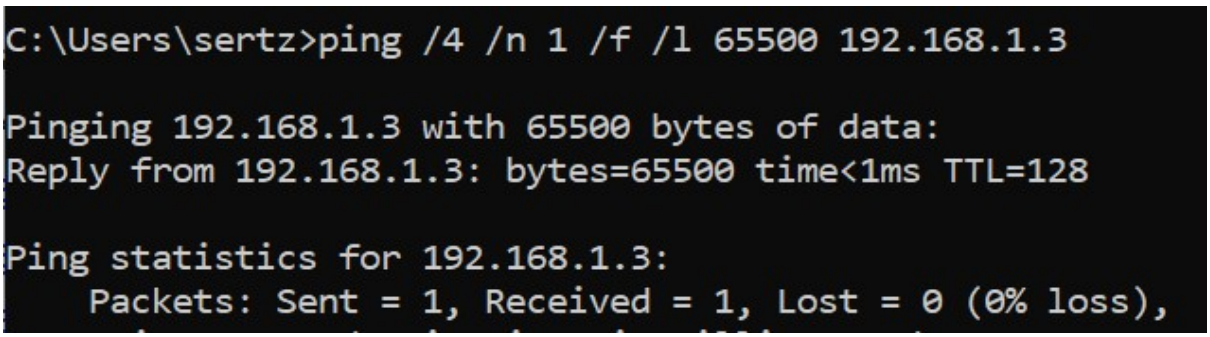
4.5: icmp and (ip.src == 192.168.1.1 or ip.dst == 192.168.1.1)

4.6: Όχι δεν παράγονται καθώς αφού είναι μεγαλύτερα του MTU δεν θα εισέλθουν στο τοπικό δίκτυο από όπου και θα τα καταγράφονταν από το Wireshark.

4.7: Από το Wireshark βλέπουμε ότι το μέγεθος του πακέτου είναι 1514 Bytes εκ των όποιων τα 14 είναι το Ethernet Header συνεπώς το μέγιστο IPv4 πακέτο, δηλαδή το MTU είναι 1500 Bytes.

4.8: Αφού το IPv4 header είναι 20 Bytes, το ICMP header είναι 8 Bytes και το MTU του τοπικού μας δικτύου είναι 1500 Bytes η μέγιστή τιμή δεδομένων του ICMP που δεν χρειάζονται θρυμματισμό είναι 1500 – 20 – 8 = 1472 Bytes. Στην γενική περίπτωση όπου γνωρίζουμε ότι το μέγιστο μέγεθος IPv4 πακέτου είναι 65535 Bytes τότε η μέγιστη τιμή δεδομένων του ICMP θα είναι 65535 – 20 – 8 = 65507 Bytes

4.9: Όχι η μέγιστη τιμή για την οποία επιτυγχάνει είναι τα 65500 Bytes δεδομένων ICMP



4.10: Θεωρητικά μιλώντας η ping θα μας άφηνε να βάλουμε στην παράμετρο buffer size που αντιστοιχεί στα δεδομένα του ICMP ως μέγιστη τιμή τον αριθμό 65507. Στην πράξη όμως μας αφήνει μόνο μέχρι την τιμή 65500 η οποία συνεπώς είναι η μέγιστη, σχηματίζοντας έτσι μέγιστο μέγεθος IPv4 πακέτου τα 65500 + 8 + 20 = 65528 Bytes.

```
C:\Users\sertz>ping /4 /n 1 /f /l 65507 192.168.1.1
Bad value for option /l, valid range is from 0 to 65500.
```

4.11: Όχι

4.12: Έχει μεταφερθεί σε 5 πακέτα το όποιο εξηγείται καθώς η μέγιστη τιμή ICMP είναι 1472 Bytes(στην πραγματικότητα σαν θρύμματα το μέγιστο μήκος είναι 1480 καθώς προφανώς υπάρχει μόνο ICMP επικεφαλίδα, επομένως το πρώτο έχει μέγιστο μήκος δεδομένων 1472 ενώ τα υπόλοιπα έχουν 1480) και συνεπώς στην περίπτωση μας που έχουμε 6000 συνολικά Bytes θα χρειαστούν 5.

4.13:

1. Identification: 0xcd3a, Don’t Fragment bit: 0, More Fragments Bit: 1, Fragment Offset: 0
2. Identification: 0xcd3a, Don’t Fragment bit: 0, More Fragments Bit: 1, Fragment Offset: 1480
3. Identification: 0xcd3a, Don’t Fragment bit: 0, More Fragments Bit: 1, Fragment Offset: 2960
4. Identification: 0xcd3a, Don’t Fragment bit: 0, More Fragments Bit: 1, Fragment Offset: 4440
5. Identification: 0xcd3a, Don’t Fragment bit: 0, More Fragments Bit: 0, Fragment Offset: 5920

4.14: Don’t Fragment: No set

4.15: Fragment Offset: 0

4.16: Data: 1480 Bytes, Total Length: 1500

4.17: Fragment Offset: 1480

4.18: Ναι

4.19: More Fragments Bit: 1

4.20: Fragment Offset, Header Checksum.

4.21: Κάθε ICMP πακέτο μεταφέρει 1480 Bytes, επομένως το Fragment Offset είναι πολλαπλάσιο του. Συνεπώς στο προτελευταίο, το 4^{ο} πακέτο, το fragment offset θα είναι 3(πακέτα που προηγήθηκαν) x 1480 = 4440 και του τελευταίου θα είναι 4 x 1480 = 5920

4.22: Όλα τα πακέτα έχουν διαφορετική τιμή στα πεδία Fragment Offset και Header Checksum. Επίσης το τελευταίο πακέτο διαφέρει από τα άλλα πακέτα στη τιμή του πεδίου More Fragment Bits.