| **Ονοματεπώνυμο: Πυλιώτης Αθανάσιος** | | **Ομάδα: 3** |
|---|---|---|
| **Όνομα PC/ΛΣ:** DESKTOP-5DLG3IF | | **Ημερομηνία: 30/11/2022** |
| **Διεύθυνση IP: 147.102.201.226** | **Διεύθυνση MAC:** | **98:54:1b:bd:69:97** |

# Εργαστηριακή Άσκηση 8
# TELNET, FTP και TFTP

**Απαντήστε στα ερωτήματα στον χώρο που σας δίνεται παρακάτω και στην πίσω σελίδα εάν δεν επαρκεί. Το φυλλάδιο αυτό θα παραδοθεί στον επιβλέποντα.**

## *1*

1.1 TCP εμφανέστατα

1.2 Transmission Control Protocol, Src Port: 58563, Dst Port: 23 (TCP)

1.3 Port 23 which is TCP port

1.4 telnet

1.5 Will Echo (εγώ σε αυτόν)

Telnet (αυτός σε εμένα)

Don't Echo

Will Echo

Won't Echo (εγώ σε αυτόν)

Και μετά στέλνεται το login

1.6 όχι δεν ζητάει να τους επαναλάβει, μάλιστα του λέει Don’t Echo, και ο υπολογιστής μου απαντάει Won’t Echo

1.7 ναι και ναι

1.8 Ναι, μας στέλνει Wil Echo που σημαίνει πως θα στέλνει αυτός Echo στον υπολογιστή μου.

1.9 Ναι έχει προηγηθεί.

1.10 login: ...aabbccdd , όπως είναι εμφανές από τα χρώματα, πρώτα στέλνει αυτός πως περιμένει να login, μετά λαμβάνει χαρακτήρα χαρακτήρα και στέλνει μετά στέλνει πάλι echo ο εξυπηρετητής σε εμάς.

1.11 Είναι απολύτως φυσιολογικό, αφού ο εξυπηρετητής απαντάει πως θα κάνει echo και εμείς δεν κάνουμε, οπότε όταν εμείς στέλνουμε αυτός μας στέλνει το echo για κάθε χαρακτήρα.

1.12 Display filter: telnet and ip.src == 147.102.201.226 and ip.dst == 147.102.40.15

1.13 4 πακέτα, ένα για κάθε χαρακτήρα.

1.14 4 πακέτα πάλι, ένα για κάθε χαρακτήρα.

1.15 όχι

1.16 όχι, δεν υπάρχει κάποια.

1.17 Γιατί δεν γίνεται echo. Γενικά η διακίνηση κωδικών είναι κάτι που δεν θέλουμε και για λόγους ασφάλειας δεδομένων λογικά υπάρχει προεγκατεστημένο Don’t Echo για τον κωδικό και δεν χρειάζεται το command κάθε φορά.

1.18 Δεν φαίνεται να έχει πολύ καλή ασφάλεια, μόνο τα στοιχειώδη όπως να μην εμφανίζεται το password με echo, αλλά η μεταφορά δεδομένων δεν είναι encrypted, οπότε πάιρνει 2/10 στην ασφάλεια και δεν θα το συνιστούσα σε κανένα.

## *2*

2.1 host 147.102.40.15

2.2 Επιτρέπει το debugging κατά την εκτέλεση της εντολής, δηλαδή δείχνει όλα τα commands μεταξύ client και server.

2.3 TCP, είναι εμφανές από την καταγράφη (βάζουμε display filter udp και δεν δείχνει τίποτα)

2.4 Transmission Control Protocol, Src Port: 58661, Dst Port: 21, Seq: 0, Len: 0 → έλεγχος

Transmission Control Protocol, Src Port: 20, Dst Port: 58662, Seq: 0, Len: 0 → δεδομένα

2.5 Από την πλευρά του εξυπηρετητή.

2.6 Request command: OPTS UTF8 ON, USER, PASS, HELP, PORT, NLST, QUIT (με arguments στις USER, PASS, PORT)

2.7 Ναι εμφανίζονται όλες οι εντολές ως ---> <COMMAND>.

**Παράδειγμα 1:**

ftp> ls

---> PORT 147,102,216,3,249,164

200 PORT command successful

---> NLST

**Παράδειγμα 2:**

ftp> bye

---> QUIT

221 Goodbye.

2.8 Wireshark: USER anonymous\r\n

Command prompt: ---> USER anonymous

2.9 1 πακέτο μόνο

2.10 Wireshark: PASS labuser@cn\r\n

Command prompt: ---> PASS labuser@cn

2.11 Στέλνεται ένα request, ένα response και ένα ακόμα πακέτο, άρα 1 καθώς στέλνεται με το request.

2.12 Ομοιότητα: Επιστρέφουν και τα δύο σαν Echo το όνομα, Διαφορά: δεν επιστρέφει τον κωδικό το TELNET ενώ το FTP τον επιστρέφει.

2.13 όχι

2.14 AUTH* → AUTHENTICATION, CCC* → Clear Command Channel

2.15 από εμένα 1, από τον εξυπηρετητή 9, 1 για κάθε γραμμή στο command prompt.

2.16 έχουν όλες οι γραμμές τον ίδιο κωδικό (214) και η διαφορά της τελευταίας με τις προηγούμενες είναι πως οι προηγούμενες έχουν μια '-' να ακολουθεί του κωδικού (214-), ενώ τη τελευταία γραμμή ένα κενό ' ' (214 )

2.17 Παριστάνουν την IPv4 address.

2.18 Οι αριθμοί υπολογίζονται ως εξής: (first*256 + second) = 229*256 + 38 = 58662 correct

2.19 NLST

2.20 Για να βεβαιωθεί πως τα ports είναι ανοιχτά και να εγκαταστήσει τη σύνδεση επιτυχώς.

Πρακτικά η σύνδεση γίνεται από τον client και ο server το μόνο που κάνει είναι να τον ενημερώνει από ποιο port ακούει. Μετά ο client στέλνει ACK πως το αναγνωρίζει και γίνεται μετά η σύνδεση. Συνολικά έχει 4 βήματα.

2.21 QUIT

2.22 server: 221 Goodbye.\r\n (είναι ευγενικός)

2.23 tcp.flags.fin == 1 (να είναι ενεργοποιημένο το flag fin )

2.24 για τα δεδομένα ξεκινάμε εμείς (client) και για τον έλεγχο ο εξυπηρετητής (server) και εμείς απαντάμε.

2.25 Transmission Control Protocol, Src Port: 58696, Dst Port: **21**.

Transmission Control Protocol, Src Port: **37350**, Dst Port: 58697 (είναι στο PASV, στο οποίο μαθαίνουμε ποια θύρα είναι αυτή)

Χρησιμοποιούνται πολλές θύρες και όχι η 20. Η 21 είναι για έλεγχο και η 37350 για μεταφορά δεδομένων.

2.26 OPTS UTF8 ON, USER, PASS, syst, site help, PWD, noop, CWD, TYPE A, PASV, LIST

2.27 USER anonymous, PASS IEUser@

2.28 LIST αφού έχει προηγηθεί PASV

2.29 227 Entering Passive Mode (147,102,40,15,145,230).\r\n

2.30 Του πελάτη και μετά ο εξυπηρετητής την αναγνωρίζει.

2.31 37350 = 145*256 + 230

2.32 Χρησιμοποιεί τον αριθμό της θύρας που χρησιμοποιούσε μέχρι τώρα για έλεγχο +1 για τη μεταφορά δεδομένων, δηλαδή ήταν 58696 και χρησιμοποιεί τη θύρα 58697.

2.33 3 και μεταφέρουν max 536 bytes δεδομένων (ή 576 bytes μέγιστο μέγεθος πακέτων TCP με τα IPv4 header and TCP Header). Τα 2 πρώτα μεταφέρουν ακριβώς τόσο και το τελευταίο 307 bytes και ολοκληρώνει τη μεταφορά.

2.34 Το μέγεθος αυτό, 576 bytes, όπως έχουμε δει πολλές, είναι το MRU και δεν μπορεί να σταλεί μεγαλύτερο πακέτο μεμονωμένα στο TCP σε τοπικό δίκτυο. Για αυτό βλέπουμε και μέγιστο μέγεθος δεδομένων 536 bytes.

2.35 Από τη πλευρά του πελάτη γίνεται η απόλυση για τον έλεγχο και ο εξυπηρετητής απαντάει πως όντως τελείωσε με FIN ACK.

2.36 Από τη πλευρά του πελάτη γίνεται η απόλυση για τα δεδομένα και δεν υπάρχει απάντηση σχετική από τον εξυπηρετητή.

## *3*

3.1 UDP

3.2 Src Port: 57262, Dst Port: 69 → UDP/TFTP port

3.3 Src Port: 42024, Dst Port: 57262

3.4 69 αντιστοιχεί στο UDP/TFTP

3.5 with Transfer IDentifiers TID TFTP manages to find the ports that need to be used. Destination is the port that we made the connection and source is the port of the originator of the packet.

3.6 ASCII

Example: Data: 000300010d0a0d0a0d0a0d0a0d0a0d0a0d0a4e6574776f726b20576f726b696e672047726f75

3.7 read request and they specify the transfer type, and in our case, it is

Trivial File Transfer Protocol

Opcode: Read Request (1)

Source File: rfc1350.txt

Type: **netascii**

3.8 read request (1), data transfer (το οποίο γίνεται και με ενδιαφέρον τρόπο, καθώς στέλνονται δεδομένα που στους πρώτους 8 χαρακτήρες μετράνε το sequence του πακέτου και μετά στέλνονται δεδομένα μόνο με 8 χαρακτήρες μεγέθους 4 bytes που ενημερώνει ο πελάτης πως έλαβε το αντίστοιχο πακέτο στέλνοντας το ίδιο sequence και μόνη διαφορά το 4$^{ο}$ ψηφίο να είναι 4 αντί για 3)

3.9 εφαρμόζοντας τη παραπάνω μέθοδο. Στέλνει ένα sequence σαν απάντηση που τελειώνει με τον sequence του frame. 001, 002, 003 κτλ και είναι ίδιο με την αρχή του μηνύματος που στέλνει ο εξυπηρετητής με μόνη διαφορά το 4$^{ο}$ ψηφίο που είναι 4 αντί για 3.

3.10 UDP μεταφορά δεδομένων είναι και είναι επικεφαλίδα UDP.

3.11 544 bytes μηνύματα IPv4 εκ των οποίων 20 bytes IPv4 header and 8 bytes UDP Header and 516 bytes data.

3.12 32 bytes εκ των οποίων 20 bytes IPv4 header and 8 bytes UDP Header and 4 bytes data, only for the “acknowledgement” of the packet.

3.13 Απλά χρειάζεται να σταλεί πακέτο δεδομένων με λιγότερα από 511 bytes και μετά να σταλεί το acknowledgement και να το λάβει όντως ο πελάτης.