Ονοματεπώνυμο: Μάρκος Δεληγιάννης	Όνομα PC: Lenovo-Laptop
Ομάδα: 1	Ημερομηνία: 14 / 3 / 2023

Εργαστηριακή Άσκηση 3
Τοπικά δίκτυα και μεταγωγείς LAN

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 PC1: **ifconfig em0 192.168.1.1/24**
PC2: **ifconfig em0 192.168.1.2/24**

1.2 1) **ifconfig em0 up**
2) **ifconfig em1 up**

1.3 Εντολές: PC1: **ping 192.168.1.2** PC2: **ping 192.168.1.1**
Παρατηρούμε ότι **δεν υπάρχει επικοινωνία**, καθώς λαμβάνουμε μηνύματα "host is down".

1.4 Εκτελούμε την εντολή **tcpdump**. Δεν παρατηρούμε μηνύματα ICMP, αλλά μόνο πακέτα ARP request από το μηχάνημα που εκτελεί κάθε φορά την ping. Αυτό είναι αναμενόμενο, καθώς τα PC1 και PC2 είναι σε διαφορετικά LAN segments και συνεπώς είναι απαραίτητη η ύπαρξη μίας γέφυρας για την μεταξύ τους επικοινωνία. Εντούτοις, δεν έχουμε διαμορφώσει κατάλληλα το B1 ώστε να λειτουργεί ως γέφυρα, με αποτέλεσμα αυτό να μην προωθεί τα πλαίσια μεταξύ των LAN1 και LAN2.

1.5 Εντολές: 1) **ifconfig bridge create** (αποτέλεσμα το bridge0)
2) **ifconfig bridge0 addm em0 addm em1 up**

1.6 **Ναι**, υπάρχει επικοινωνία, όπως είναι αναμενόμενο.

1.7 Το TTL είναι **64**, οπότε γνωρίζοντας ότι τα πακέτα ip της ping ξεκίνησαν με TTL=64 συμπεραίνουμε ότι το PC1 **απέχει μόνο 1 βήμα** από το PC2. Αυτό είναι λογικό, αφού μόνο οι δρομολογητές αλλάζουν την τιμή του TTL (επίπεδο δικτύου), και όχι οι γέφυρες (επίπεδο ζεύξης δεδομένων).

1.8 Παρατηρούμε ότι οι καταχωρήσεις στους δύο πίνακες συμφωνούν πλήρως. Αυτό σημαίνει ότι η γέφυρα, παρόλο που μεσολάβησε μεταξύ του PC1 και PC2, δεν άλλαξε τις διευθύνσεις MAC προέλευσης των πλαισίων, είναι δηλαδή **διαφανής**.

1.9 Εντολές: **tcpdump -evvv -i em0** και **tcpdump -evvv -i em1**
Παρατηρούμε τα ίδια πλαίσια και στις δύο καταγραφές, οπότε πράγματι η γέφυρα B1 προωθεί κανονικά τα πλαίσια μεταξύ των LAN1 και LAN2.

1.10 **Όχι**, δεν γίνεται κάποια αλλαγή στις διευθύνσεις MAC και IPv4.

1.11 **Όχι**, δεν παρατηρούμε κάποια άλλη αλλαγή. Αυτό συνάδει με τον **διαφανή χαρακτήρα** της γέφυρας.

1.12 **Όχι**, δεν υπάρχει κάποια ένδειξη της ύπαρξης του B1. Η αιτιολόγηση αυτού είναι όμοια με αυτή του 1.7, δηλαδή η traceroute εντοπίζει δρομολογητές (οντότητες στο επίπεδο δικτύου) και όχι γέφυρες (επίπεδο ζεύξης δεδομένων).

1.13 PC1: **ping 192.168.1.2** B1: **tcpdump -evvv -i em1**

1.14 Εντολή: **ifconfig em0 192.168.2.1/24**
Παρατηρούμε ότι η γέφυρα εξακολουθεί να προωθεί τα πακέτα ICMP echo request στο LAN2.

1.15 **Όχι**, δεν είναι επιτυχές, καθώς παρόλο που η διεπαφή em0 του PC2 λαμβάνει το πλαίσιο Ethernet (έχει σωστή MAC), το πακέτο IPv4 δεν έχει τη σωστή διεύθυνση, οπότε απορρίπτεται, και συνεπώς κανείς δεν απαντά με ICMP echo reply.

1.16 **Όχι**, δεν μπορούμε. Εντολή: **ping 192.168.1.1**

1.17 Εντολές: 1) **ifconfig em2 up**
2) **ifconfig bridge0 addm em2**

1.18 **Ναι**, λαμβάνουμε απάντηση.

1.19 **Δεν εμφανίζονται πακέτα ICMP** στο LAN2. Αυτό είναι αναμενόμενο, καθώς και ο PC1 και ο PC3 έχουν ανταλλάξει πλαίσια Ethernet, οπότε η γέφυρα έχει μάθει τις διεπαφές στις οποίες πρέπει να προωθεί τα πλαίσια. Έτσι, δεν είναι απαραίτητη η χρήση πλημμύρας και συνεπώς δεν στέλνονται πλαίσια στο LAN2. Εντολή: **tcpdump -i em1 'icmp'** από το B1

1.20 Η προηγούμενη εντολή ανίχνευε μόνο τα πακέτα ICMP, όπως ζητείται. Επειδή τώρα αναζητούμε τα σχετικά με την ping πλαίσια, την τροποποιούμε ως εξής: **tcpdump -i em1 'icmp or arp'**.
Κάνοντας ping από τον PC1 στον PC3 παρατηρούμε στην καταγραφή ένα μόνο πακέτο ARP request από τον PC1 για τη διεύθυνση MAC του PC3. Αυτό είναι πλαίσιο broadcast (ως ARP request), για αυτό και στέλνεται παντού. Ο λόγος που παρατηρούμε τέτοιο πακέτο είναι επειδή εκκαθαρίσαμε το ARP table, οπότε ο PC1 δεν γνώριζε την MAC του PC3. (Εντολή για εκκαθάρηση ARP table: **arp -da**)

1.21 Εντολή: **ifconfig bridge0** Οι διεπαφές-μέλη χαρακτηρίζονται ως *member*.

1.22 Εντολή: **ifconfig bridge0 addr**

1.23 Οι εγγραφές για τις em0, em1, em2 αντιστοιχούν στα PC1, PC2, PC3. Επιβεβαιώνουμε με **ifconfig em0** στα 3 μηχανήματα.

1.24 Εντολή: **ifconfig bridge0 flush**

1.25 Εντολή: **ifconfig bridge0 deletem em2**

1.26 Εντολή: **ifconfig bridge0 destroy**

1.27 Εντολή: **ifconfig em0 delete** στα PC1, PC2, PC3

2

2.1 Εντολή: **ifconfig em0 [IPv4 address/24]** στα PC1,2,3,4

2.2 Εντολές: 1) **ifconfig bridge1 create addm em0 addm em1 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**

2.3 Εντολές: 1) **ifconfig bridge2 create addm em0 addm em1 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**

2.4 Εντολές: 1) **ifconfig bridge3 create addm em0 addm em1 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**

2.5 Διευθύνσεις MAC: PC1: **08:00:27:ec:73:82** PC2: **08:00:27:ad:5d:60**
(Εντολή: **ifconfig em0**) PC3: **08:00:27:23:2f:4f** PC4: **08:00:27:cb:2e:62**
Για την εκκαθάριση του ARP table εκτελούμε **arp -da**.

2.6 Εντολή: **ifconfig bridgeX flush** (με X={1,2,3})

2.7 Εντολή: **tcpdump -e**

2.8 Βεβαιωνόμαστε ότι οι πίνακες προώθησης των B1,2,3 είναι άδειοι με την εντολή: **ifconfig bridgeX addr** (με X={1,2,3}). Πράγματι, οι εντολές δεν εμφανίζουν κάτι.
Μετά την εκτέλεση της ping οι πίνακες προώθησης είναι οι εξής:
B1: 1) **08:00:27:ad:5d:60 – em1 (LNK1)** 2) **08:00:27:ec:73:82 – em0 (LAN1)**
B2: 1) **08:00:27:ad:5d:60 – em0 (LNK1)** 2) **08:00:27:ec:73:82 – em0 (LNK1)**
B3: 1) **08:00:27:ec:73:82 – em0 (LNK2)**

2.9 Λόγω της τοπολογίας και του ότι τα LAN1,2 και LNK1,2 είναι μέσα εκπομπής γνωρίζουμε ότι κάθε εγγραφή στον πίνακα προώθησης του B1 αντιστοιχεί στη MAC προέλευσης ενός πλαισίου που κατέγραψε ο PC1 ή ο PC2. Πλήρως αντίστοιχες παρατηρήσεις ισχύουν για τον B2 με τους PC2,3 και τον B3 με τους PC3,4. Συνεπώς μπορούμε να κατασκευάσουμε τον πίνακα του B1 παίρνοντας τις MAC προέλευσης όλων των πλαισίων των καταγραφών του PC1,2 (αντίστοιχα για τους B2,3). Παραθέτουμε παρακάτω τα περιεχόμενα των καταγραφών των PC1,2,3,4:
PC1 και PC2 (ταυτίζονται): ARP request/reply και ICMP echo request/reply → MAC προέλευσης: PC1, PC2
PC3 και PC4 (ταυτίζονται): ARP request → MAC προέλευσης: PC1
Με βάση τον κανόνα που περιγράψαμε προκύπτουν αμέσως τα αποτελέσματα του 2.8.

2.10 Δεν υπήρχαν αλλαγές στους πίνακες προώθησης των B1,2,3. Αυτό είναι λογικό, καθώς ο PC2 γνώριζε την MAC του PC1 από την προηγούμενη επικοινωνία τους, οπότε δεν υπήρξε ανάγκη για ARP πακέτα (δεν στάλθηκαν πλαίσια broadcast). Επιπλέον, οι B1,2 γνώριζαν τις σωστές διεπαφές για τα PC1,2, οπότε όλα τα πλαίσια ήταν περιορισμένα στο LAN1 και LNK1. Έτσι, ο B3 δεν έμαθε τον PC2.

2.11 Παρατηρούμε στην καταγραφή του PC2 πακέτο ARP reply από τον PC4. Το αντίστοιχο πλαίσιο είχε ως MAC προέλευσης αυτή του PC4. Επίσης, το PC2 και το B1 συνδέονται σε κοινό μέσο εκπομπής, το LNK1, οπότε αυτό το πλαίσιο σίγουρα ανιχνεύτηκε και από το B1, οδηγώντας στην δημιουργία εγγραφής για το PC4.

2.12 **Ναι**, στους πίνακες προώθησης των B1,2,3 προστέθηκε εγγραφή για τον PC3. Αυτό οφείλεται στο ο PC3 στέλνει ARP request (broadcaast) για την MAC του PC2.

2.13 Εντολές: **ping 192.168.1.2** από τους PC4 και PC1

2.14 Το ping αυτό **επιτυγχάνει**, καθώς οι δύο υπολογιστές βρίσκονται στο ίδιο LAN segment, οπότε έχουν απευθείας επικοινωνία άσχετα με τη διαμόρφωση των B1,2,3.

2.15 Το ping αυτό **αποτυγχάνει**, καθώς τα πλαίσια που φέρουν το ICMP echo request του PC1 δεν προωθούνται από τον B2 στο LNK2, οπότε δεν φτάνουν τον PC2. Αυτός λοιπόν δεν απαντά, με αποτέλεσμα τα B1,2,3 να μην ανανεώνουν τους πίνακες προώθησής τους και η κατάσταση να μην αλλάζει (βλ. 2.17).

2.16 Τώρα το ping **επιτυγχάνει**. Αυτό συμβαίνει διότι ο PC2 έστειλε πλαίσια προς τον PC3, τα οποία έφτασαν στον προορισμό τους, διότι οι καταχωρήσεις στους πίνακες προώθησης για τον PC3 είναι σωστές. Αυτά τα πλαίσια πέρασαν από τους B2 και B3 και είχαν ως MAC προέλευσης αυτή του PC2. Έτσι, οι προβληματικές εγγραφές στους B2 και B3 για τον PC2 διορθώθηκαν και τα μηνύματα ICMP του PC1 άρχισαν να φτάνουν στον προορισμό τους.

2.17 Θα έπρεπε να περιμένουμε να λήξει η εγγραφή στους πίνακες προώθησης του B2 και B3 για τον PC2 ή στον ARP πίνακα του PC1 για τον PC2 (όποιο συμβεί πρώτο αρκεί). Στην πρώτη περίπτωση οι B2 και B3 χρησιμοποιούν πλημμύρα και τα ICMP echo request φτάνουν στο LAN2. Στην δεύτερη περίπτωση ο PC1 στέλνει ARP request (broadcast), που φτάνουν το B2, οπότε αυτό απαντά με ARP reply, ενημερώνοντας τους πίνακες προώθησης του B2,3.

3

3.1 Εντολές: 1) **ifconfig bridge1 create addm em0 addm em1 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**

3.2 Εντολές: 1) **ifconfig bridge2 create addm em0 addm em1 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**

3.3 Διευθύνσεις MAC: PC1: **08:00:27:ec:73:82** PC2: **08:00:27:ad:5d:60**
(Εντολή: **ifconfig em0**) PC3: **08:00:27:23:2f:4f**
Για την εκκαθάριση του ARP table εκτελούμε **arp -da**.

3.4 Εντολές: **tcpdump -e** από το PC1 και **ping 192.168.1.3** από το PC2.
Στην καταγραφή του PC1 παρατηρούμε μόνο ένα πακέτο ARP request από τον PC2 για την διεύθυνση MAC του PC3. Αυτό είναι το πρώτο πλαίσιο που στέλνεται και είναι broadcast, οπότε στέλνεται παντού και δημιουργεί εγγραφές στους πίνακες προώθησης του B1,2 για τον PC2. Το πακέτο ARP reply που ακολουθεί από τον PC3 είναι προς τον PC2, οπότε ο B2, αφού δημιουργήσει καταχώρηση για τον PC3, δεν το προωθεί στον B1. Έτσι, αυτό δεν φτάνει τον PC1. Τα μηνύματα ICMP που ακολουθούν επίσης παραμένουν εντός του LAN2, καθώς ο B2 πλέον γνωρίζει τα PC2,3.

3.5 Εντολή: **ping 192.168.1.1**

3.6 Εντολές: 1) **ifconfig bridge1 addm em2**
2) **ifconfig em2 up** για το B1, αντίστοιχα με bridge2 για το B2

3.7 Εντολή: **ifconfig bridge1 addr** (αντίστοιχα bridge2 για B2)
Παρατηρούμε εγγραφές για τα PC1,2,3 και στους δύο πίνακες προώθησης.

3.8 B1: **PC1 στην em0** (LAN1) και **PC3 στην em1** (LNK1) Εντολή: **ifconfig bridge1 addr**
B2: **PC1 στην em0** (LNK1) και **PC3 στην em1** (LAN2) Εντολή: **ifconfig bridge2 addr**

3.9 Εντολή: **tcpdump -e**

3.10 Εντολή για εκκαθάριση του πίνακα ARP: **arp -da**
Το ping **δεν είναι επιτυχές**.

3.11 B2: **PC1 στην em0** (LNK1) και **PC3 στην em0** (LNK1) Εντολή: **ifconfig bridge2 addr**
Ο PC3 δεν γνώριζε τη MAC του PC1, οπότε έστειλε ένα broadcast πλαίσιο που έφερε ARP request. Αυτό προωθήθηκε από τον B2 στον B1 μέσω του LNK1 και LNK2 και από τον B1 στο LAN1, αλλά και στα LNK1,2. Έτσι, το ARP request προωθείται διαρκώς μεταξύ των B1 και B2, ανανεώνοντας την εγγραφή για τον PC3 στον B2 (em0 αντί για em1).

3.12 ARP request: Από τον PC3 για τη MAC του PC1 ARP reply: Από τον PC1 στον PC3

3.13 Η MAC διεύθυνση πηγής των ARP request είναι **08:00:27:23:2f:4f**, δηλαδή η MAC του PC3.

3.14 Τα πακέτα επαναλαμβάνονται συνεχώς σε αμφότερες τις καταγραφές, διότι ενθυλακώνονται σε πλαίσια Ethernet broadcast, και υπάρχει βρόχος μεταξύ των B1 και B2. Έτσι, οι B1 και B2 συνεχώς λαμβάνουν αντίγραφα του πλαισίου, τα οποία προωθούν στα LAN1 και LAN2, αλλά και μεταξύ τους, τροφοδοτώντας τον φαύλο κύκλο.

3.15 Η γέφυρα B2 αρχικά είχε τη σωστή εγγραφή στον πίνακα προώθησης, όμως λόγω της διαδικασίας που περιγράφηκε στο 3.14 έλαβε στη συνέχεια αντίγραφα του πακέτου του PC3 από τις διεπαφές των LNK1,2. Έτσι, η σωστή εγγραφή για τον PC3 αντικαταστάθηκε, με αποτέλεσμα τα πακέτα να ακολουθούν κυκλική πορεία μεταξύ των B1,2 και να μην φτάνουν ποτέ στον PC3.

4

4.1 Εντολές: 1) **ifconfig bridge1 destroy** (αντίστοιχα bridge2)
2) **ifconfig em0 down** 3) **ifconfig em1 down** 4) **ifconfig em2 down**
3) **ifconfig bridge create**

4.2 Εντολές: 1) **ifconfig em0 up** 2) **ifconfig em1 up** 3) **ifconfig em2 up**
4) **ifconfig lagg0 create**

4.3 Εντολή: **ifconfig lagg0 up laggport em1 laggport em2**

4.4 Εντολές: 1) **ifconfig em0 up**
2) **ifconfig em1 up**
3) **ifconfig em2 up**
4) **ifconfig lagg0 create**
5) **ifconfig lagg0 up laggport em0 laggport em2**

4.5 Εντολή: **ifconfig bridge0 addm em0 addm lagg0 up**

4.6 Εντολή: **ifconfig bridge0 addm em1 addm lagg0 up**

4.7 Εντολές: **tcpdump -e** από το PC1 και **ping 192.168.1.3** από το PC2.
Στην καταγραφή του PC1 παρατηρούμε μόνο ένα πακέτο ARP request από τον PC2 για την διεύθυνση MAC του PC3. Αυτό είναι το πρώτο πλαίσιο που στέλνεται και είναι broadcast, οπότε στέλνεται παντού και δημιουργεί εγγραφές στους πίνακες προώθησης του B1,2 για τον PC2. Το πακέτο ARP reply που ακολουθεί από τον PC3 είναι προς τον PC2, οπότε ο B2, αφού δημιουργήσει καταχώρηση για τον PC3, δεν το προωθεί στον B1. Έτσι, αυτό δεν φτάνει τον PC1. Τα μηνύματα ICMP που ακολουθούν επίσης παραμένουν εντός του LAN2, καθώς ο B2 πλέον γνωρίζει τα PC2,3.

4.8 Εντολή: **tcpdump**

4.9 Εντολή για εκκαθάριση του πίνακα ARP: **arp -da**
Το ping ήταν **επιτυχές**. **Παρατηρήσαμε πακέτα ARP request και reply** στην καταγραφή του PC1, όπως ήταν αναμενόμενο.

4.10 Εντολές: 1) **tcpdump -i em1** από τον B1
2) **tcpdump -i em2** από τον B2
3) **ping 192.168.1.1** από το PC2
Παρατηρούμε ότι τα πακέτα προωθούνται μόνο μέσω του LNK1, ενώ το LNK2 δεν χρησιμοποιείται καθόλου. Αυτό συμβαίνει διότι το προεπιλεγμένο πρωτόκολλο συνάθροισης είναι το failover, και έχει επιλεγεί το LNK1 ως master. Το LNK2 χρησιμοποιείται δηλαδή μόνο ως backup.

4.11 Παρατηρούμε ότι η ping **συνεχίζει να επιτυγχάνει**, παρόλο που αποσυνδέσαμε το LNK1. Επίσης παρατηρούμε ότι τώρα τα πακέτα διέρχονται από τη ζεύξη LNK2, αντί για την LNK1. Αυτό είναι λογικό, καθώς όπως προαναφέραμε η LNK2 λειτουργούσε ως backup, οπότε τώρα που η LNK1 έχει αποσυνδεθεί αναλαμβάνει η LNK2 χωρίς τα ακραία συστήματα να καταλαβαίνουν τη διαφορά.

4.12 Όπως ήταν αναμενόμενο, τώρα η LNK1 είναι πάλι διαθέσιμη, οπότε δεδομένου ότι αυτή έχει τον ρόλο του master η κίνηση διέρχεται πάλι από αυτή.

5

5.1 Εντολές: 1) **ifconfig bridge0 destroy**
2) **ifconfig lagg0 destroy**
3) **ifconfig em0 down**
4) **ifconfig em1 down**
5) **ifconfig em2 down**

5.2 Εντολές: 1) **ifconfig bridge1 create addm em0 addm em1 addm em2 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**
4) **ifconfig em2 up**

5.3 Εντολές: 1) **ifconfig bridge2 create addm em1 addm em0 addm em2 up**
2) **ifconfig em0 up**
3) **ifconfig em1 up**
4) **ifconfig em2 up**

5.4 Εντολή: **ifconfig bridge1 stp em0 stp em1 stp em2**

5.5 Εντολή: **ifconfig bridge2 stp em0 stp em1 stp em2**

5.6 Εντολές: B1) **ifconfig bridge1** id: **08:00:27:9f:b9:3f** priority: **32768**
B2) **ifconfig bridge2** id: **08:00:27:74:85:f4** priority: **32768**

5.7 Η ρίζα του επικαλύπτοντος δένδρου είναι η γέφυρα με το **μικρότερο Bridge ID**, δηλαδή η **B2**.
Εναλλακτικά μπορούμε να εκτελέσουμε **ifconfig bridge1** (ή 2) και να επιθεωρήσουμε το **root id**.

5.8 em2: Ο ρόλος είναι **designated** και η κατάσταση **forwarding**.
em0: Ο ρόλος είναι **designated** και η κατάσταση **forwarding**.
em1: Ο ρόλος είναι **designated** και η κατάσταση **forwarding**.
Εντολή: **ifconfig bridge2** και εστιάζουμε για κάθε διεπαφή **στα role και state**.

5.9 Είναι η θύρα 2, που αντιστοιχεί στα **em1 και LNK1**, καθώς αυτή έχει ρόλο **root**. Εντολή: **ifconfig bridge1**

5.10 Η θύρα 3, που αντιστοιχεί στα **em2 και LNK2**, έχει ρόλο **alternate** και κατάσταση **discarding**.

5.11 Η διεπαφή **em0 του B1** που αντιστοιχεί στο LAN1 έχει ρόλο **designated** και κατάσταση **forwarding**.

5.12 Εκπέμπονται BPDU κάθε περίπου **2 δευτερόλεπτα**. Εντολή: **tcpdump -i em1 -vvve** από τη B2 (για LAN2)

5.13 Χρησιμοποιείται ενθυλάκωση **IEEE 802.3**, το οποίο φαίνεται στην καταγραφή ("802.3").

5.14 Διεύθυνση MAC πηγής: **08:00:27:ce:39:a2** (B2)
Διεύθυνση MAC προορισμού: **01:80:c2:00:00:00**

5.15 Ανήκει στην διεπαφή **em1 της ρίζας (B2)**, η οποία είναι η διεπαφή της οποίας την κίνηση καταγράφουμε.
Εντολή: **ifconfig em1** στη B2

5.16 Το LSB του πρώτου byte της MAC προορισμού είναι 1, άρα είναι **multicast**.

5.17 Root ID: **8000.08:00:27:74:85:f4** Bridge ID: **8000.08:00:27:74:85:f4.8002** Root path cost: **0**

5.18 Εντολή: **tcpdump -i em0 -vvve** από τη B2. Τα πρώτα δύο bytes του Bridge ID είναι η προτεραιότητα, δηλαδή το 0x8000 = 32768, το οποίο φαίνεται και με εκτέλεση της **ifconfig bridge2**.

5.19 Το δεύτερο μέρος, δηλαδή τα 6 bytes **08:00:27:74:85:f4** είναι το id της γέφυρας, το οποίο αντιστοιχεί στη MAC της πρώτης διεπαφής της. Το τρίτο μέρος, δηλαδή τα 2 τελευταία bytes **0x8002** είναι το port ID, το οποίο διαφέρει και μεταξύ των δύο καταγραφών.

5.20 **Όχι**, δεν παρατηρούμε.

5.21 Στην **designated θύρα** , που αντιστοιχεί στο **em0 και το LAN1**. Εντολή: **tcpdump -i em0 -vvve** στη B1

5.22 Root ID: **8000.08:00:27:74:85:f4** Bridge ID: **8000.08:00:27:9f:b9:3f.8001** Root path cost: **20000**

5.23 **Ναι**, είναι επιτυχές. Εντολή: **ping 192.168.1.2**

5.24 Περνάνε **περίπου 6 δευτερόλεπτα** για την αποκατάσταση της επικοινωνίας. Το πρωτόκολλο RSTP έχει την ικανότητα αντίδρασης σε αλλαγές στην τοπολογία σε περίπου **3 φορές το hello time**, δηλαδή 6 δευτερόλεπτα στην περίπτωσή μας. Τα δύο νούμερα ταυτίζονται απόλυτα.

5.25 **Ναι**, υπάρχει διακοπή, αλλά πολύ μικρότερης χρονικής διάρκειας (~1 second).

6

6.1 Εντολές: 1) **ifconfig em3 up**
2) **ifconfig bridge1 addm em3**
3) **ifconfig bridge1 stp em3**

6.2 Εντολές: 1) **ifconfig em3 up**
2) **ifconfig bridge2 addm em3**
3) **ifconfig bridge2 stp em3**

6.3 Εντολές: 1) **ifconfig bridge3 create**
2) **ifconfig em0 up** 3) **ifconfig em1 up** 4) **ifconfig em2 up**
5) **ifconfig bridge3 up addm em2 addm em0 addm em1**
6) **ifconfig bridge3 stp em0 stp em1 stp em2**

6.4 Εντολή για διαγραφή των πινάκων προώθησης: **ifconfig bridgeX flush** (με X={1,2,3})
Τα ping προς τους PC2 και PC3 **επιτυγχάνουν**.

6.5 Εντολή: **ifconfig bridge1 priority 0**

6.6 Εκτελούμε την εντολή **ifconfig bridge2** και εστιάζουμε στο **path cost** κάθε διεπαφής-μέλους. Και οι τρεις διεπαφές έχουν κόστος 20,000. Εκτελώντας **ifconfig emX** (με X={0,1,2,3}) παρατηρούμε ότι όλες οι διεπαφές έχουν **ταχύτητα 1Gbps** (1000baseT). Ο τύπος υπολογισμού του path cost είναι: path cost = 200,000,000/link speed (σε 100kbps), ο οποίος δίνει το αποτέλεσμα που παρατηρήσαμε.

6.7 Από την B1: **root path cost = 0** αφού η B1 είναι η ρίζα.
Από την B2: **root path cost = 20,000** αφού η B2 απέχει από τη ρίζα 20,000 (οι διεπαφές της προς τη B1 έχουν κόστος 20,000).

6.8 Η θύρα που αντιστοιχεί στην **em0 και το LNK3**, καθώς από το 6.7 έχει το μικρότερο root path cost (τα κόστη των διεπαφών της B3 στα LNK3 και LNK4 είναι ίσα, οπότε μπορούμε απλώς να συγκρίνουμε τα νούμερα του 6.7).

6.9 Ο ρόλος της είναι **designated** και η κατάστασή της **forwarding**. Η αντίστοιχη θύρα στο B2 έχει ρόλο **alternate** και κατάσταση **discarding**.

6.10 Το root path cost είναι ίσο με **20,000**. Εντολή: **ifconfig -i em2 -vvve**

6.11 Εντολή: **ping 192.168.1.3**

6.12 Εντολή: **ifconfig bridge3 ifpathcost em0 100000**
Διαλέξαμε την τιμή 100,000 γιατί χρειαζόμαστε μία τιμή μεγαλύτερη του κόστους του εναλλακτικού μονοπατιού προς τη ρίζα, το οποίο είναι 20,000 + 20,000 = 40,000.

6.13 Πέρασαν περίπου **4 δευτερόλεπτα** μέχρι την αποκατάσταση της επικοινωνίας.

6.14 bridge3 – LNK3 (em0): ρόλος = **alternate** και κατάσταση **= discarding**
bridge2 – LNK4 (em3): ρόλος = **designated** και κατάσταση **= forwarding**.

6.15 **Όχι**, δεν υπάρχει αλλαγή.

6.16 **Ναι**, έχει αλλάξει το root path cost από 20,000 σε **40,000**.

6.17 Χρειάστηκαν περίπου **7 δευτερόλεπτα** για την αποκατάσταση της επικοινωνίας.

6.18 Χρειάστηκαν περίπου 6 **δευτερόλεπτα** για την αποκατάσταση της επικοινωνίας.

6.19 Εντολές: 1) **ifconfig bridge3 addm em3** 2) **ifconfig bridge3 stp em3**
em2 (παλιά): ρόλος = **designated** κατάσταση = **forwarding**
em3 (νέα): ρόλος = **backup** κατάσταση = **discarding**

6.20 Αρκεί να θέσουμε οποιαδήποτε τιμή **μικρότερη του 40,000**, που είναι το κόστος της εναλλακτικής διαδρομής. Επιλέγουμε **20,000**, καθώς αυτήν την τιμή είχε η θύρα πριν την αλλάξουμε, και ήταν ριζική.

7

7.1 Εντολές: 1) **ifconfig em0.5 create inet 192.168.5.1/24**
2) **ifconfig em0.6 create inet 192.168.6.1/24**

7.2 Εντολές: 1) **ifconfig em0.5 create**
2) **ifconfig em0.6 create**

7.3 Εντολές: 1) **ifconfig em1.6 create**
2) **ifconfig em3.5 create**

7.4 Εντολή: **ifconfig em0.6 create inet 192.168.6.2/24**

7.5 Εντολές: 1) **ifconfig em1.6 create**
2) **ifconfig em0.6 create**

7.6 Εντολή: **ifconfig em0.5 create inet 192.168.5.3/24**

7.7 Εντολές: 1) **ifconfig em2.5 create**
2) **ifconfig em0.5 create**

7.8 **Ναι**, μπορούμε. Εντολές: **ping 192.168.6.2** και **ping 192.168.5.3**

7.9 Εντολή: **ifconfig bridge1 -stp em0**

7.10 Εντολή: **tcpdump -vvvexx -i em0**

7.11 Εντολή για εκκαθάριση πίνακα ARP: **arp -da**
Η τιμή του Ethertype είναι **0x0806** για τα ARP και **0x0800** για τα IPv4.

7.12 Τα πλαίσια Ethernet τώρα έχουν 4 παραπάνω bytes από τα αντίστοιχα που παρατηρήσαμε πριν.

7.13 Η τιμή του πεδίου Ethertype είναι **0x8100 (802.1Q)**. Σύμφωνα με το πρωτόκολλο 802.1Q μπορούμε να αντλήσουμε την πληροφορία για το πρωτόκολλο που ενθυλακώνει το πλαίσιο όπως πριν, αγνοώντας τα 4 bytes **0x8100xxxx**. Παρατηρώντας τα επόμενα δύο bytes στην καταγραφή λαμβάνουμε τις τιμές Ethertype του 7.11, οι οποίες φέρουν τη ζητούμενη πληροφορία.

7.14 Η πληροφορία για το VLAN εμφανίζεται **στα bytes #14-15 (VLAN tag)** της επικεφαλίδας Ethernet.

7.15 Εντολή: **tcpdump -vvvexx -i em0.5**

7.16 Εντολή για εκκαθάριση πίνακα ARP: **arp -da**
Η τιμή του Ethertype είναι **0x0806** για τα ARP και **0x0800** για τα Ipv4.
Δεν υπάρχει πεδίο σχετικό με VLAN.

7.17 Εντολές: 1) **ifconfig bridge1 stp em0**
2) **tcpdump -vvvexx -i em0**

7.18 **Όχι**, δεν είναι ίδιου τύπου. Είναι πλαίσια **Ethernet 802.3**. Στη θέση του Ethertype υπάρχει το μήκος του πλαισίου σε bytes, σύμφωνα με το πρωτόκολλο 802.3.

7.19 Φίλτρο: **'not stp'**