Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 3
Τοπικά δίκτυα και μεταγωγείς LAN

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 22/03/2022

Άσκηση 1: Γέφυρα - Διασύνδεση δύο LAN

1.1

Θα ορίσουμε IPv4 διευθύνσεις για τις διεπαφές των PC1, PC2 με τις εντολές:
`ifconfig em0 192.168.1.1/24` για το PC1
`ifconfig em0 192.168.1.2/24` για το PC2

1.2

Θα ενεργοποιήσουμε τις διεπαφές του B1 με τις εντολές:
`ifconfig em0 up`
`ifconfig em1 up`

1.3

Κάνουμε `ping 192.168.1.2` (από το PC1 στο PC2) και `ping 192.168.1.1` (από το PC2 στο PC1). Παρατηρούμε ότι κανένα από τα δύο `ping` δεν λαμβάνει απάντηση.

1.4

Με χρήση των `tcpdump -i em0` και `tcpdump -i em1` στο B1 βλέπουμε ότι δεν παράγονται πακέτα ICMP στα LAN1, LAN2, παρά μόνο ARP Requests, επειδή το PC1 προσπαθεί να μάθει την φυσική διεύθυνση του PC2 και αντίστροφα. Επειδή όμως τα PC1, PC2 βρίσκονται σε ξεχωριστά LANs, δεν θα έρθει ποτέ ARP Reply, με αποτέλεσμα να μην παράγονται ICMP πακέτα.

1.5

Με την εντολή `ifconfig bridge0 create` δημιουργούμε μία ψευδο-συσκευή γέφυρα `bridge0` στο B1 και με την εντολή `ifconfig bridge0 addm em0 addm em1 up` προσθέτουμε τις δύο διεπαφές του σαν μέλη στη γέφυρα, και ενεργοποιούμε τη γέφυρα `bridge0`.

1.6

Δοκιμάζουμε πάλι τις εντολές `ping` του ερωτήματος 1.3. Αυτή τη φορά υπάρχει επικοινωνία εξαιτίας της γέφυρας.

1.7

Βλέπουμε ότι το TTL είναι 64, άρα το PC2 δεν απέχει κανένα βήμα από το PC1.

1.8

Με την εντολή `arp -a` βλέπουμε τους πίνακες `arp` των PC1, PC2, και βλέπουμε ότι και οι δύο έχουν μόνο τις δύο εγγραφές των PC1 και PC2, και καθόλου της γέφυρας. Αυτό συμβαίνει διότι η γέφυρα είναι διαφανής, δηλαδή οι υπολογιστές αγνοούν την ύπαρξή της.

1.9

Με τις εντολές `tcpdump -evvv -i em0` και `tcpdump -evvv -i em1` στο B1, επιβεβαιώνουμε ότι γίνεται προώθηση μεταξύ LAN1, LAN2 από το B1, αφού οι φυσικές διευθύνσεις που αναγράφονται στα πλαίσια της καταγραφής είναι αυτές των PC1, PC2.

1.10

Όχι, δεν κάνει καμία αλλαγή στις διευθύνσεις MAC, ούτε στις διευθύνσεις IPv4.

1.11

Όχι, δεν αλλάζει κανένα άλλο πεδίο των πλαισίων.

1.12

Κάνουμε `traceroute 192.168.1.2`, δηλαδή από το PC1 στο PC2. Βλέπουμε ότι δεν υπάρχει καμία ένδειξη ύπαρξης του B1 στην έξοδο του `traceroute`. Αυτό συμβαίνει διότι όπως αναφέραμε και προηγουμένως, η λειτουργία της γέφυρας είναι διαφανής, και τα μηχανήματα την αγνοούν τελείως.

1.13

Εκτελούμε `ping 192.168.1.2` από το PC1 στο PC2 και με την εντολή `tcpdump -i em1` ξεκινάμε μια καταγραφή στο LAN2.

1.14

Με την εντολή `ifconfig em0 192.168.2.1/24` αλλάζουμε την διεύθυνση του PC2. Με το προηγούμενο ping να τρέχει βλέπουμε ότι η γέφυρα συνεχίζει να προωθεί τα πακέτα `ICMP` προς τη διεύθυνση `192.168.1.2`, αλλά όχι στο PC2, αφού αυτό πλέον άλλαξε διεύθυνση.

1.15

Το ping δεν είναι επιτυχές, γιατί πλέον το PC2 ανήκει σε διαφορετικό IP υποδίκτυο και απαιτείται η παρουσία router.

1.16

Όχι, δεν μπορούμε να κάνουμε `ping 192.168.1.1` από το PC3.

1.17

Προσθέτουμε τη διεπαφή `em2` του B1 στην `bridge0` με την εντολή `ifconfig bridge0 addm em2`, και ενεργοποιούμε την `em2` με την εντολή `ifconfig em2 up`.

1.18

Επαναλαμβάνουμε τώρα το `ping 192.168.1.1` από το PC3, και αυτή τη φορά λαμβάνουμε απάντηση.

1.19

Αν κάνουμε `ping` από το PC1 στο PC3 ή το αντίστροφο, δεν εμφανίζονται πακέτα ICMP στο LAN2, όπως μπορούμε να διαπιστώσουμε από tcpdump στο B1 (ή στο PC2). Αυτό συμβαίνει επειδή η γέφυρα έχει μάθει πλέον πού να προωθήσει το πακέτο ICMP, εξαιτίας του προηγούμενου ping που κάναμε, και δεν χρειάζεται να το στείλει στο LAN2.

1.20

Καθαρίζουμε τον πίνακα arp των PC1, PC3 με `arp -d -a`. Επαναλαμβάνουμε τη προηγούμενη καταγραφή στο LAN2 και κάνουμε ping από το PC1 στο PC3. Τώρα εμφανίζεται στην καταγραφή ένα ARP Request από το PC1 που θέλει να μάθει την φυσική διεύθυνση του PC3. Αυτό συμβαίνει επειδή το ARP Request είναι broadcast και προωθείται από την γέφυρα προς όλες τις διεπαφές της, εκτός από αυτήν που ήρθε.

1.21

Με την εντολή `ifconfig bridge0` μπορούμε να δούμε ποιες διεπαφές είναι μέλη της γέφυρας `bridge0` (αναγράφονται κάτω-κάτω, με επικεφαλίδα `member`).

1.22

Με την εντολή `ifconfig bridge0 addr` μπορούμε να δούμε το περιεχόμενο του πίνακα προώθησης της γέφυρας `bridge0`.

1.23

Οι αντιστοιχίσεις είναι οι εξής:

- `08:00:27:82:ed:cb` → PC1 (em0)

- `08:00:27:ce:04:8b` → PC2 (em1)
- `08:00:27:e7:87:37` → PC3 (em2)

1.24

Με την εντολή `ifconfig bridge0 flush` διαγράφουμε τις εγγραφές του πίνακα προώθησης.

1.25

Αφαιρούμε από τη γέφυρα `bridge0` τη διεπαφή της στο LAN3 με την εντολή:

```
ifconfig bridge0 deletem em2
```

1.26

Καταστρέφουμε τη γέφυρα `bridge0` με την εντολή `ifconfig bridge0 destroy`.

1.27

Αφαιρούμε τις διευθύνσεις IP από τις διεπαφές των PC1, PC2, PC3 με την εντολή:

```
ifconfig em0 delete
```

Άσκηση 2: Αυτο-εκπαίδευση γεφυρών

2.1

Ορίζουμε διευθύνσεις IPv4 στις διεπαφές των PC1,2,3,4 με τις εντολές:

```
ifconfig em0 192.168.1.1/24 (PC1)
ifconfig em0 192.168.1.2/24 (PC2)
ifconfig em0 192.168.1.3/24 (PC3)
ifconfig em0 192.168.1.4/24 (PC4)
```

2.2

Δημιουργούμε μια ψευδο-συσκευή γέφυρα `bridge1` στο B1 που να περιλαμβάνει τις διεπαφές στα LAN1 και LNK1 με τις εντολές:

```
ifconfig bridge1 create
ifconfig bridge1 addm em0 addm em1 up
```

και ενεργοποιούμε τις διεπαφές με τις εντολές:

```
ifconfig em0 up
ifconfig em1 up
```

2.3

Ομοίως:

```
ifconfig bridge2 create
ifconfig bridge2 addm em0 addm em1 up
```

και:

```
ifconfig em0 up
ifconfig em1 up
```

2.4

Ομοίως:

```
ifconfig bridge3 create
ifconfig bridge3 addm em0 addm em1 up
```

και:

```
ifconfig em0 up
ifconfig em1 up
```

2.5

- PC1: `08:00:27:82:ed:cb`
- PC2: `08:00:27:ce:04:8b`
- PC3: `08:00:27:e7:87:37`
- PC4: `08:00:27:e0:7c:b9`

Αδειάζουμε τους πίνακες arp με την εντολή `arp -d -a` σε όλα τα μηχανήματα.

2.6

Διαγράφουμε τις εγγραφές του πίνακα προώθησης των B1,2,3 με τις εντολές:

```
ifconfig bridge1 flush
ifconfig bridge2 flush
ifconfig bridge3 flush
```

2.7

Στα PC1,2,3,4 ξεκινάμε μία καταγραφή με tcpdump, με την εντολή `tcpdump -i em0`.

2.8

Βεβαιωνόμαστε ότι οι πίνακες προώθησης των B1,2,3 είναι κενοί με τις εντολές:

```
ifconfig bridge1 addr
ifconfig bridge2 addr
ifconfig bridge3 addr
```

Οι εντολές δεν παράγουν έξοδο, άρα οι πίνακες προώθησης είναι κενοί.

Εκτελούμε, από νέο παράθυρο στο PC1, την εντολή `ping -c 1 192.168.1.2`.

Το περιεχόμενο των πινάκων προώθησης είναι πλέον:

```
B1:
08:00:27:ce:04:8b Vlan1 em1 1190 flags=0<>
08:00:27:82:ed:cb Vlan1 em0 1190 flags=0<>

B2:
08:00:27:ce:04:8b Vlan1 em0 1187 flags=0<>
08:00:27:82:ed:cb Vlan1 em0 1187 flags=0<>

B3:
08:00:27:82:ed:cb Vlan1 em0 1185 flags=0<>
```

2.9

Σε κάθε μηχάνημα καταγράφηκαν τα εξής:

- PC1:
  - ICMP Echo Request
  - ICMP Echo Reply
  - ARP Request
  - ARP Reply
- PC2:
  - ICMP Echo Request
  - ICMP Echo Reply
  - ARP Request
  - ARP Reply
- PC3:
  - ARP Request
- PC4:
  - ARP Request

Η bridge1 μαθαίνει ότι το PC1 βρίσκεται στη διεπαφή em0 της, εξαιτίας του ARP request (αφού στα ARP request έχουμε broadcast), ενώ θα μάθει ότι το PC2 βρίσκεται στην em1 της, λόγω του ARP reply.

Η bridge2 μαθαίνει ότι το PC1 βρίσκεται στη διεπαφή em0 της, εξαιτίας του ARP request, ενώ θα μάθει ότι το PC2 βρίσκεται στην em0 της, λόγω του ARP reply.

Η bridge3 μαθαίνει ότι το PC1 βρίσκεται στη διεπαφή em0 της, εξαιτίας του ARP request.
Τα μηνύματα ICMP echo request/reply θα περάσουν μόνο από την bridge1, διότι πλέον η bridge1 ξέρει ότι πρέπει να προωθήσει το ICMP echo request που προορίζεται για το PC2 στη διεπαφή em1, ενώ πρέπει να προωθήσει το ICMP echo reply που προορίζεται για το PC1 στη διεπαφή em0. Γι' αυτό το λόγο η bridge3 δεν μαθαίνει κάτι παραπάνω.

2.10

Από το PC2 εκτελούμε `ping -c 1 192.168.1.1`. Παρατηρούμε ότι οι πίνακες προώθησης των B1,2,3 δεν έχουν αλλάξει. Αυτό συμβαίνει επειδή πλέον το PC2 γνωρίζει την φυσική διεύθυνση του PC1, και επομένως δεν χρειάζεται να εκπέμψει ξανά ARP request (κάτι που θα σήμαινε πως η bridge3 θα μάθαινε πού βρίσκεται το PC2). Για τα ICMP echo request/reply ισχύουν τα παραπάνω.

2.11

Από το PC2 εκτελούμε `ping -c 1 192.168.1.4`.
Παρατηρούμε ότι όντως ο πίνακας προώθησης του B1 περιέχει τη διεύθυνση MAC του PC4.
Αυτό συμβαίνει επειδή το PC4 στέλνει ARP reply για να απαντήσει στο ARP request του PC2.
Έτσι, το ARP reply φτάνει στο LNK1, και άρα το βλέπει και η bridge1, οπότε καταχωρεί πως το B1 βρίσκεται στη διεπαφή em1.

2.12

Από το PC3 εκτελούμε `ping -c 1 192.168.1.2`.
Παρατηρούμε ότι σε όλους τους πίνακες προστέθηκε μία εγγραφή:

```
B1:
+++ 08:00:27:e7:87:37 Vlan1 em1 1163 flags=0<>

B2:
+++ 08:00:27:e7:87:37 Vlan1 em1 1160 flags=0<>

B3:
+++ 08:00:27:e7:87:37 Vlan1 em0 1158 flags=0<>
```

Αυτό συμβαίνει επειδή το PC3 στέλνει ARP request (το οποίο γίνεται broadcast) για να μάθει τη φυσική διεύθυνση του PC2.

2.13

Από νέο παράθυρο στο PC4 κάνουμε `ping 192.168.1.2`, και από νέο παράθυρο στο PC1 κάνουμε επίσης `ping 192.168.1.2`. Αφήνουμε και τα δύο ping να τρέχουν.

2.14

Μετακινούμε το PC2 από το LNK1 το LAN2, όπου βρίσκεται το PC4. Το ping από το PC4 προς το PC2 συνεχίζει κανονικά, αφού τα δύο μηχανήματα βρίσκονται στο ίδιο εσωτερικό δίκτυο (δεν χρειάζεται έτσι κι αλλιώς γέφυρα).
PC4->PC2 συνεχίζει κανονικά PC1->PC2 δεν λαμβάνει απάντηση πλέον Συνεχίζει πλέον κανονικά

2.15

Το ping από το PC1 προς το PC2 σταματάει να λαμβάνει απάντηση.

Η γέφυρα `bridge1` συνεχίζει να προωθεί τα ICMP echo request στην διεπαφή `em1`, όμως το PC2 δεν βρίσκεται πλέον στο LNK1. Επίσης, η γέφυρα `bridge2` δεν προωθεί τα ICMP echo request στην `em1` (όπως θα έπρεπε, αν θέλουμε να φτάσουν στο PC2), γιατί νομίζει ότι το PC2 βρίσκεται ακόμα στην `em0`. Γι' αυτό και το ping από το PC1 προς το PC2 δεν λαμβάνει απάντηση.

2.16

Στο PC2, εκτελούμε `ping -c 1 192.168.1.3`. Παρατηρούμε ότι πλέον το ping από το PC1 στο PC2 λαμβάνει απάντηση.

Αυτό συμβαίνει διότι το PC2 στέλνει ένα ICMP echo request προς τη `bridge3`, η οποία ξέρει ότι το PC3 βρίσκεται στην `em0`, οπότε το προωθεί εκεί. Έτσι, το ICMP echo request φτάνει στην `bridge2`, η οποία μαθαίνει έτσι ότι το PC2 πλέον βρίσκεται στην `em1`.

Με αυτόν τον τρόπο, η `bridge1` προωθεί το ICMP echo request του PC1 (που προορίζεται για το PC2) στην `em1`, η `bridge2` το προωθεί στην `em1`, και η `bridge3` το προωθεί στην `em1`, και πλέον το ping εκτελείται με επιτυχία.

2.17

Αν δεν εκτελούσαμε το προηγούμενο `ping`, θα έπρεπε να περιμένουμε μέχρι:
Να λήξει η εγγραφή της `bridge2` για το πού βρίσκεται το PC2, ώστε το ICMP echo request να χρειαστεί να προωθηθεί στην `em1` (πλημμύρα), και από εκεί να φτάσει μέσω τις `bridge3` στο PC2. Με το ICMP echo reply από το PC2, όλες οι γέφυρες είναι πλέον ενημερωμένες και το ping λειτουργεί κανονικά.
Να λήξει η εγγραφή του πίνακα arp που αφορά τη διεύθυνση του PC2, στο PC1. Σε αυτή την περίπτωση, το PC1 στέλνει ARP request για να μάθει τη φυσική διεύθυνση του PC2, και το ARP reply του PC2 ενημερώνει όλους τους πίνακες προώθησης, οπότε το ping δουλεύει πλέον κανονικά.

Άσκηση 3: Καταιγίδα πλαισίων εκπομπής

3.1

Δημιουργούμε γέφυρα `bridge0` στο B1 με την εντολή:

```
ifconfig bridge0 create
```

Ύστερα ενεργοποιούμε την `bridge0`, προσθέτοντας ως μέλη τις διεπαφές `em0` (LAN1) και `em1` (LNK1) με την εντολή:

```
ifconfig bridge0 addm em0 addm em1 up
```

Τέλος, φροντίζουμε να ενεργοποιήσουμε τις διεπαφές που συμμετέχουν στη γέφυρα με τις εντολές:

```
ifconfig em0 up
ifconfig em1 up
```

3.2

Αντίστοιχα στο B2 (LAN2 → em2, LNK1 → em0):

```
ifconfig bridge1 create
ifconfig bridge1 addm em2 addm em0 up
ifconfig em2 up
ifconfig em0 up
```

3.3

Καταγράφουμε τις διευθύνσεις MAC των PC1,2,3 (εντολή `ifconfig em0`):

- PC1: `08:00:27:82:ed:cb`
- PC2: `08:00:27:ce:04:8b`
- PC3: `08:00:27:e7:87:37`

Αδειάζουμε τους πίνακες ARP με την εντολή `arp -d -a`.

3.4

Ξεκινάμε μια καταγραφή στο PC1 με `tcpdump -i em0` και από το PC2 κάνουμε `ping 192.168.1.3`. Παρατηρούμε μόνο το broadcast ARP request από το PC2 που θέλει να μάθει τη φυσική διεύθυνση του PC3, και επειδή ύστερα η γέφυρα B2 ξέρει ότι το PC3 βρίσκεται στη διεπαφή em2 του LAN2, η σχετική με το ping κίνηση δεν προωθείται προς τη B1, άρα ούτε και στο PC1, οπότε δεν παρατηρούμε κάποια άλλη κίνηση.

3.5

Ξεκινάμε στο PC3 ένα `ping 192.168.1.1` και το αφήνουμε να τρέχει.

3.6

Προσθέτουμε στις γέφυρες `bridge0` και `bridge1` τις διεπαφές τους στο LNK2 `em2`, και `em1` αντίστοιχα, με τις εντολές:

```
ifconfig bridge0 addm em2
ifconfig bridge1 addm em1
```

και ενεργοποιούμε τις διεπαφές αυτές με τις εντολές:

```
ifconfig em2 up
ifconfig em1 up
```

3.7

Σταματάμε το ping και ελέγχουμε το περιεχόμενο των πινάκων προώθησης των γεφυρών `bridge0` και `bridge1`, με τις εντολές:

```
ifconfig bridge0 addr
ifconfig bridge1 addr
```

Τα περιεχόμενα των πινάκων προώθησης είναι:

```
bridge0:
08:00:27:82:ed:cb em0
08:00:27:e7:87:37 em1

bridge1:
08:00:27:82:ed:cb em0
08:00:27:e7:87:37 em2
```

3.8

Έχουμε:

- B1:
  - PC1 → em0 (LAN1)
  - PC3 → em1 (LNK1)
- B2:
  - PC1 → em0 (LNK1)
  - PC3 → em2 (LAN2)

3.9

Ξεκινάμε μία καταγραφή στο PC1 με `tcpdump -i em0` και μία καταγραφή στο PC2 πάλι με `tcpdump -i em0`.

3.10

Κάνουμε εκκαθάριση του πίνακα arp στο PC3 με `arp -d -a` και δίνουμε την εντολή:

```
ping -c 1 192.168.1.1
```

Το ping είναι επιτυχές.

3.11

Στις καταγραφές όντως παρατηρούμε έναν κατακλυσμό από πακέτα ARP. Αυτό συμβαίνει διότι το B2 στέλνει το broadcast ARP request στις διεπαφές του με τα LNK1 και LNK2, το B1 στέλνει τα ARP requests που λαμβάνει από το LNK1 στο LNK2 και αυτά που λαμβάνει από το LNK2 στο LNK1, το B2 ξανά με τη σειρά του στέλνει τα ARP requests που λαμβάνει από το LNK1 στο LNK2 και αυτά που λαμβάνει από το LNK2 στο LNK1 κ.ο.κ. Έτσι κυκλοφορούν συνεχώς ARP requests στον βρόχο που δημιουργείται στις ζεύξεις LNK1,2. Επίσης, επειδή αυτά τα ARP requests φτάνουν στον PC1, αυτός αναγκάζεται να παράγει συνεχώς ARP replies. Για να σταματήσει αυτός ο καταιγισμός, αποσυνδέουμε το καλώδιο από τις κάρτες δικτύου των B1 και B2, σε μία από τις ζεύξεις LNK1 ή LNK2 (επιλέξαμε τη LNK2), για να σπάσει ο βρόχος. Τελικά, στο B2 οι MAC διευθύνσεις των PC1 και PC3 εμφανίζονται και οι δύο στη διεπαφή `em0` του LNK1, επειδή η B2 νομίζει ότι το ARP request του PC3 (που αρχικά παράχθηκε από τα "δεξιά" της) έρχεται από τα "αριστερά" της λόγω του βρόχου.

3.12

Στον κατακλυσμό της καταγραφής του PC1, γίνεται η ερώτηση:

```
Who has 192.168.1.1? Tell 192.168.1.3.
```

και δίνεται η απάντηση:

```
192.168.1.1 is at 08:00:27:82:ed:cb.
```

3.13

Στην καταγραφή στο PC2 παρατηρούμε έναν κατακλυσμό από ARP request, διότι το B2 λαμβάνει τα προαναφερθέντα ARP request που προκύπτουν λόγω του βρόχου, και επειδή τα ARP request είναι broadcast, τα προωθεί σε όλες τις διεπαφές εκτός από εκείνη από όπου τα έλαβε. Άρα όσα ARP request έρχονται από τις διεπαφές της B2 με τα LNK1 και LNK2 προωθούνται στην διεπαφή με το LAN2.

3.14

Τα πακέτα ARP request επαναλαμβάνονται συνεχώς σε αμφότερες τις καταγραφές για τον λόγο που εξηγήθηκε ακριβώς από πάνω για την καταγραφή του PC2. Ακριβώς η ίδια διαδικασία ισχύει και για το LAN1, και άρα για την καταγραφή του PC1.

3.15

Τα ARP reply δεν προωθούνται στο LAN2 διότι, ενώ στην αρχή το B2 ήξερε πού βρίσκεται το PC3, στη συνέχεια, κατέγραψε (λανθασμένα) ότι το PC3 βρίσκεται από την "αριστερή" μεριά (στην διεπαφή με το LNK1 ή με το LNK2), λόγω του βρόχου που δημιουργήθηκε.

Άσκηση 4: Συνάθροιση ζεύξεων

4.1

Καταστρέφουμε τις γέφυρες στα B1,2 με τις εντολές:

```
ifconfig bridge0 destroy
ifconfig bridge1 destroy
```

και απενεργοποιούμε τις διεπαφές τους με τις εντολές:

```
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down
```

Στη συνέχεια δημιουργούμε νέες γέφυρες (χωρίς να προσθέσουμε προς το παρόν διεπαφές) με τις εντολές:

```
ifconfig bridge0 create
ifconfig bridge1 create
```

4.2

Ενεργοποιούμε όλες τις κάρτες δικτύου στο B1 με τις εντολές:

```
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
```

και δημιουργούμε μία ψευδο-συσκευή συνάθροισης `lagg0` με την εντολή:

```
ifconfig lagg0 create
```

4.3

Εντάσσουμε στην `lagg0` τις διεπαφές της B1 στα LNK1 (`em1`) και LNK2 (`em2`) και ενεργοποιούμε την ψευδο-συσκευή με την εντολή:

```
ifconfig lagg0 up laggport em1 laggport em2
```

4.4

Επαναλαμβάνουμε τα δύο προηγούμενα βήματα για το B2, δηλαδή:

```
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up

ifconfig lagg0 create

ifconfig lagg0 up laggport em0 laggport em1
```

Να σημειωθεί ότι στο B2 ισχύει η αντιστοιχία `LNK1:em0` και `LNK2:em1`.

4.5

Στο B1, εντάσσουμε στη γέφυρα `bridge0` που δημιουργήσαμε τη διεπαφή του στο LAN1 και την `lagg0`, και ενεργοποιούμε τη γέφυρα, με την εντολή:

```
ifconfig bridge0 addm em0 addm lagg0 up
```

4.6

Ομοίως, στο B2:

```
ifconfig bridge1 addm em2 addm lagg0 up
```

4.7

Με `tcpdump -i em0` ελέγχουμε αν εμφανίζεται κίνηση στο PC1 όταν κάνουμε `ping 192.168.1.3` από το PC2. Βλέπουμε ότι στην καταγραφή εμφανίζεται μόνο ένα πλαίσιο ARP request από το PC2 που θέλει να μάθει την φυσική διεύθυνση του PC3. Το B2 γνωρίζει εξαιτίας των ARP request/reply ότι τα PC2,3 βρίσκονται στο LAN2, και έτσι η κίνηση περιορίζεται εκεί, οπότε δεν παρατηρούμε άλλη κίνηση στο PC1.

4.8

Ξεκινάμε μία καταγραφή στο PC1 με `tcpdump -i em0`.

4.9

Εκκαθαρίζουμε τον πίνακα arp του PC3 με την εντολή `arp -d -a` και δίνουμε την εντολή `ping -c 1 192.168.1.1`. Το ping επιτυγχάνει, και παρατηρούμε ένα ARP request από το PC3 που θέλει να μάθει την φυσική διεύθυνση του PC1, και το αντίστοιχο ARP reply.

4.10

Ξεκινάμε μία καταγραφή στη διεπαφή του B1 στο LNK1 (`em1`) και μία άλλη στη διεπαφή του B2 στο LNK2 (`em1`) με τις εντολές `tcpdump -i em1` (στο B1) και `tcpdump -i em1` (στο B2).

Από το PC2 κάνουμε `ping 192.168.1.1` στο PC1 και το αφήνουμε να τρέχει. Παρατηρούμε ότι τα πακέτα ICMP εμφανίζονται στη ζεύξη LNK1 (αφού εμφανίστηκαν πακέτα μόνο στην καταγραφή του B1). Αυτό συμβαίνει γιατί το default aggregation mode είναι `failover`, το οποίο σημαίνει ότι τα πακέτα στέλνονται μόνο μέσω του primary/master interface, δηλαδή του interface που αντιστοιχεί στο LNK1, και το LNK2 θα χρησιμοποιηθεί μόνο σε περίπτωση αποτυχίας του LNK1.

4.11

Αποσυνδέουμε τα καλώδια από τις κάρτες δικτύου των B1 και B2 στη ζεύξη LNK1 (όπου παρατηρήσαμε πακέτα προηγουμένως). Παρατηρούμε ότι η εντολή ping συνεχίζει κανονικά, ενώ τώρα τα πακέτα ICMP μεταφέρονται μέσω της ζεύξης LNK2, αφού η master interface δεν λειτουργεί.

4.12

Επανασυνδέουμε τις κάρτες δικτύων των B1 και B2, και βλέπουμε ότι τα πακέτα ICMP μεταφέρονται μέσω της LNK1 αυτή τη φορά, αφού η ζεύξη (και άρα η master interface) είναι και πάλι λειτουργική.

Άσκηση 5: Αποφυγή βρόχων

5.1

Στα B1 και B2, καταστρέφουμε τις γέφυρες με τις εντολές:

```
ifconfig bridge0 destroy
ifconfig bridge1 destroy
```

τις ψευδο-συσκευές συνάθροισης με την εντολή:

```
ifconfig lagg0 destroy
```

και απενεργοποιούμε τις κάρτες δικτύου με τις εντολές:

```
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down
```

5.2

Δημιουργούμε γέφυρα `bridge1` στο B1 που να περιλαμβάνει τις διεπαφές του στα LAN1, LNK1 και LNK2, και ενεργοποιούμε αυτήν και όλες τις διεπαφές με τις εντολές:

```
ifconfig bridge1 create
ifconfig bridge1 addm em0 addm em1 addm em2 up
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
```

5.3

Ομοίως για το B2:

```
ifconfig bridge2 create
ifconfig bridge2 addm em0 addm em1 addm em2 up
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
```

5.4

Ενεργοποιούμε το STP για όλες τις διεπαφές που μετέχουν στην `bridge1` με την εντολή:

```
ifconfig bridge1 stp em0 stp em1 stp em2
```

5.5

Ομοίως για την bridge2:

```
ifconfig bridge2 stp em0 stp em1 stp em2
```

5.6

Με τις εντολές:

```
ifconfig bridge1
ifconfig bridge2
```

βρίσκουμε ότι οι γέφυρες bridge1 και bridge2 έχουν Bridge ID ίσο με 8000.08:00:27:6e:b3:91 και 8000.08:00:27:16:2d:45 αντίστοιχα (μορφή priority.mac-address, προτεραιότητα 32768 δεκαδικό → 8000 δεκαεξαδικό).

5.7

Η γέφυρα ρίζα του επικαλύπτοντος δέντρου είναι η bridge2, αφού γι' αυτήν ισχύει: id == root.id.

5.8

Είναι role designated και state forwarding, και για τις 3 διεπαφές em0,1,2 της bridge2.

5.9

Η ριζική θύρα της μη-ριζικής γέφυρας bridge1 είναι η διεπαφή στο LNK1.

5.10

Για την διεπαφή LNK2, η κατάσταση είναι discarding και ο ρόλος είναι alternate.

5.11

Για την διεπαφή στο LAN1 της μη-ριζικής γέφυρας bridge1, η κατάσταση είναι forwarding και ο ρόλος είναι designated.

5.12

Ξεκινάμε μία καταγραφή στην διεπαφή της γέφυρας ρίζας bridge2 στο LNK1 με την εντολή:

```
tcpdump -evvv -i em0
```

Βλέπουμε ότι εκπέμπονται BPDU κάθε 2 δευτερόλεπτα.

5.13

Χρησιμοποιείται ενθυλάκωση IEEE 802.3.

5.14

Η διεύθυνση MAC πηγής των BPDU είναι `08:00:27:50:45:8b`, ενώ η διεύθυνση MAC προορισμού είναι `01:80:c2:00:00:00`.

5.15

Η διεύθυνση MAC πηγής ανήκει στη διεπαφή της γέφυρας ρίζας στο LNK1.

5.16

Η διεύθυνση MAC προορισμού είναι multicast, αφού το LSBit του MSByte της διεύθυνσης είναι ίσο με 1.

5.17

Στα πλαίσια BPDU που καταγράψαμε προηγουμένως η root-id είναι `8000.08:00:27:16:2d:45`, η bridge-id είναι `8000.08:00:27:16:2d:45.8001` και το root-path-cost είναι `0`.

5.18

Ξεκινάμε μία αντίστοιχη καταγραφή στη διεπαφή με το LNK2 με την εντολή:

```
tcpdump -evvv -i em1
```

Συγκρίνοντας με την προηγούμενη καταγραφή, βρίσκουμε ότι αν το bridge-id έχει την μορφή `xxxx.yyyyyy.zzzz`, τότε `xxxx` είναι η προτεραιότητα (τιμή 8000 δεκαεξαδικό) και `zzzz` είναι ο αριθμός της θύρας (8001 για το LNK1, 8002 για το LNK2)

5.19

Στο B1, εκτελούμε `tcpdump -evvv -i em1` και `tcpdump -evvv -i em2`. Δεν παρατηρούμε κάποια κίνηση που να προέρχεται από αυτές τις διεπαφές.

5.20

Με `tcpdump -evvv -i em0` παρατηρούμε παραγωγή κίνησης BPDU από την διεπαφή της `bridge1` στο LAN1.

5.21

Στα προηγούμενα BPDU, είναι:

```
root-id: 8000.08:00:27:16:2d:45
bridge-id: 8000.08:00:27:16:2d:45.8002
root-pathcost: 0
```

5.22

Εκτελούμε `ping 192.168.1.2` από το PC1 στο PC2. Το ping είναι επιτυχές.

5.23

Αποσυνδέουμε το κατάλληλο καλώδιο και παρατηρούμε ότι η επικοινωνία σταματά στο πακέτο με `icmp_seq = 286`, ενώ αποκαθίσταται στο πακέτο με `icmp_seq = 295`, δηλαδή η επικοινωνία έκανε 8-9 δευτερόλεπτα να αποκατασταθεί. Κανονικά για να ανιχνευθεί ότι το καλώδιο αποσυνδέθηκε χρειάζονται 3 hello times (6 δευτερόλεπτα), ωστόσο (ίσως εξαιτίας επιπλέον καθυστερήσεων μέχρι να παραχθεί το ping και να ληφθεί η απάντηση) χρειάστηκαν 8-9 δευτερόλεπτα για να επανεγκατασταθεί η επικοινωνία.

5.24

Όχι, δεν υπάρχει διακοπή στην επικοινωνία.

Άσκηση 6: Ένα πιο πολύπλοκο δίκτυο με εναλλακτικές διαδρομές

6.1

Προσθέτουμε στην `bridge1` τη διεπαφή του B1 στο LNK3 και την ενεργοποιούμε με τις εντολές:

```
ifconfig bridge1 addm em3
ifconfig em3 up
```

Ενεργοποιούμε και το STP για τη διεπαφή που προσθέσαμε με την εντολή:

```
ifconfig bridge1 stp em3
```

6.2

Αντίστοιχα για την `bridge2`:

```
ifconfig bridge2 addm em3
ifconfig em3 up
ifconfig bridge2 stp em3
```

6.3

Δημιουργούμε γέφυρα `bridge3` στο B3 με την εντολή `ifconfig bridge3 create`. Προσθέτουμε στη γέφυρα και ενεργοποιούμε τις διεπαφές του B3 στα LAN3, LNK3 και LNK4 με τις εντολές:

```
ifconfig bridge3 addm em0 addm em1 addm em2 up
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
```

Τέλος, ενεργοποιούμε το STP σε όλες τις διεπαφές που μετέχουν στην `bridge3` με την εντολή:

```
ifconfig bridge3 stp em0 stp em1 stp em2
```

6.4

Διαγράφουμε τους πίνακες προώθησης σε όλες τις γέφυρες με τις εντολές:

```
ifconfig bridge1 flush
ifconfig bridge2 flush
ifconfig bridge3 flush
```

Εκτελούμε `ping 192.168.1.2` και `ping 192.168.1.3`. Είναι και τα δύο επιτυχή.

6.5

Βλέπουμε ότι η `bridge1` δεν είναι ρίζα του επικαλύπτοντος δέντρου. Για να γίνει, χρησιμοποιούμε την εντολή:

```
ifconfig bridge1 priority 28672
```

Ο αριθμός 28672 επιλέχθηκε επειδή είναι κατά 4096 μικρότερος από το 32768.

6.6

Το root path cost για τις ζεύξεις αυτές είναι 20000. Αυτό προκύπτει από τον τύπο:

$$20 \text{ Tbps} \,/\, \text{bandwidth} = 20 \text{ Tbps} \,/ \mathbf{1\ Gbps} = 20000.$$

6.7

Ξεκινάμε δύο καταγραφές στο B3 με τις εντολές (em1 → LNK3, em2 → LNK4):

```
tcpdump -evvv -i em1
tcpdump -evvv -i em2
```

Το root path cost στα πλαίσια BPDU από την `bridge1` είναι 0, αφού η `bridge1` είναι η ρίζα, ενώ από την `bridge2` είναι 20000, αφού η `bridge2` συνδέεται απευθείας με την `bridge1` με κόστος 20000.

6.8

Η θύρα στο `em1` (LNK3) είναι ριζική, αφού συνδέεται απευθείας με την γέφυρα-ρίζα `bridge1` με κόστος 20000, ενώ το μονοπάτι μέσω της θύρας LNK4 έχει κόστος 40000.

6.9

Η άλλη θύρα (στο LNK4) έχει ρόλο designated και κατάσταση forwarding.

6.10

Με `tcpdump -evvv -i em0` στο B3 βρίσκουμε ότι το root path cost στα πλαίσια BPDU που παράγει η `bridge3` στο LAN3 είναι 20000.

6.11

Ξεκινάμε ένα `ping 192.168.1.3` από το PC1 στο PC3 και το αφήνουμε να τρέχει.

6.12

Με την εντολή `ifconfig bridge3 ifpathcost em1 50000` ορίζουμε το κόστος της διεπαφής της `bridge3` στο LNK3, έτσι ώστε να γίνει ριζική θύρα η διεπαφή της στο LNK4. Η τιμή 50000 επιλέχθηκε επειδή είναι μεγαλύτερη από το κόστος μέχρι τη ρίζα μέσω του LNK4 (40000), οπότε με αυτόν τον τρόπο προτιμάται η διαδρομή μέσω του LNK4.

6.13

Για να αποκατασταθεί η επικοινωνία πέρασαν περίπου 18 δευτερόλεπτα.

6.14

Η κατάσταση της διεπαφής της `bridge3` στο LNK3 είναι `discarding` ενώ ο ρόλος της είναι `alternate`.

6.15

Με `tcpdump -evvv -i em1` και `tcpdump -evvv -i em2` βλέπουμε ότι δεν άλλαξε η τιμή κάποιας παραμέτρου.

6.16

Με `tcpdump -evvv -i em0` βλέπουμε ότι άλλαξε η τιμή του root path cost σε 40000 από 20000.

6.17

Παρατηρήσαμε ότι η επικοινωνία αποκαθίσταται 10 δευτερόλεπτα αφότου αποσυνδέσαμε το καλώδιο.

6.18

Παρατηρήσαμε ότι η επικοινωνία αποκαθίσταται 6 δευτερόλεπτα αφότου αποσυνδέσαμε το καλώδιο.

Άσκηση 7: Εικονικά τοπικά δίκτυα (VLAN)

7.1

Στο PC1 δημιουργούμε τις ζητούμενες διεπαφές με τις εντολές:

```
ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.1/24
ifconfig em0.6 create vlan 6 vlandev em0 inet 192.168.6.1/24
```

7.2

Ομοίως στο PC2:

```
ifconfig em0.6 create vlan 6 vlandev em0 inet 192.168.6.2/24
```

7.3

Ομοίως στο PC3:

```
ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.3/24
```

7.4

Ναι, μπορούμε να κάνουμε `ping 192.168.6.2` και `ping 192.168.5.3` από το PC1.

7.5

Όχι, δεν μπορούμε να κάνουμε `ping 192.168.5.3` από το PC2, διότι το PC2 δεν έχει διεπαφή που να ανήκει στο εικονικό δίκτυο `192.168.5.0/24`.

7.6

Όχι, δεν μπορούμε να κάνουμε `ping 192.168.6.2` από το PC3, διότι το PC3 δεν έχει διεπαφή που να ανήκει στο εικονικό δίκτυο `192.168.6.0/24`.

7.7

Αφαιρούμε από το επικαλύπτον δέντρο τη διεπαφή της `bridge1` στο LAN1 με την εντολή `ifconfig bridge1 -stp em0`.

7.8

Ξεκινάμε τη ζητούμενη καταγραφή με την εντολή `tcpdump -v -e -x -i em0`.

7.9

Στο PC2 καθαρίζουμε τον πίνακα ARP με `arp -d -a` και εκτελούμε την εντολή `ping -c 1 192.168.1.1`. Η τιμή του πεδίου Ethertype των πλαισίων για τα πακέτα ARP είναι `0x0806`, ενώ για τα πακέτα IPv4 είναι `0x0800`.

7.10

Στο PC2 εκτελούμε την εντολή `ping -c 1 192.168.6.1`. Τα πλαίσια Ethernet που παράγονται τώρα είναι κατά 4 bytes μεγαλύτερα, επειδή προστέθηκε το VLAN tag 802.1Q.

7.11

Η τιμή του πεδίου Ethertype είναι τώρα `0x8100`, που δηλώνει ετικέτα 802.1Q. Τώρα τα πακέτα ARP ξεχωρίζουν από τα IP από το 17ο και το 18ο byte, που περιέχουν πλέον το Ethertype.

7.12

Πληροφορία για το LAN εμφανίζεται στο πεδίο ”Tag Control Information”.

7.13

Στο PC1 ξεκινάμε την ζητούμενη καταγραφή με την εντολή `tcpdump -e -vvv -xx -i em0.5`.

7.14

Στο PC3 καθαρίζουμε τον πίνακα ARP με `arp -d -a` και εκτελούμε την εντολή `ping -c 1 192.168.5.1`. Αυτή τη φορά το πεδίο Ethertype έχει τιμή `0x0806` για τα πακέτα ARP και `0x0800` για τα πακέτα που μεταφέρουν τα μηνύματα ICMP. Βλέπουμε ότι δεν υπάρχει πεδίο σχετικό με το VLAN, αφού η καταγραφή γίνεται εντός του εικονικού δικτύου (διεπαφή `em0.5`).

7.15

Προσθέτουμε τη ζητούμενη διεπαφή στο επικαλύπτον δέντρο με την εντολή `ifconfig bridge1 stp em0`. Ξεκινάμε μία νέα καταγραφή στο `em0` αυτή τη φορά, με την εντολή `tcpdump -e -vvv -xx -i em0`.

7.16

Όχι, αφού πλαίσια που μεταφέρουν BPDU είναι IEEE 802.3, και στη θέση του Ethertype έχουν το μήκος (length).

7.17

Εάν δεν είχαμε αφαιρέσει τη ζητούμενη διεπαφή από το επικαλύπτον δέντρο, θα χρησιμοποιούσαμε το φίλτρο ”not stp” για να μην συλλαμβάνουμε πλαίσια BPDU.