Εργαστήριο Δικτύων Υπολογιστών

Ονοματεπώνυμο: Παναγιώτης Σταματόπουλος	Όνομα PC: TakisAsus
Ομάδα: 2	Ημερομηνία: 21/2/2024

Εργαστηριακή Άσκηση 3

Τοπικά δίκτυα και μεταγωγείς LAN

Άσκηση 1:

1.1: ifconfig 192.168.1.1/24 (PC1)
ifconfig 192.168.1.2/24 (PC2)
1.2: ifconfig em0 up
ifconfig em1 up
1.3: Με ping τα πακέτα φαίνεται να μην φτάνουν
1.4: tcpdump icmp
Δεν παράγονται πακέτα στο LAN1 και LAN2 γιατί τα PC1 και PC2 δεν έχουν επικοινωνήσει και άρα δεν έχει καταγραφεί η MAC address τους.¨
1.5: ifconfig bridge create
ifconfig bridge0 addm em0 addm em1
ifconfig bridge0 up
1.6: Ναι
1.7: TTL = 64 είναι η default τιμή που σημαίνει ότι συνδέονται άμεσα
1.8: arp -a
Έχει γίνει αντιστοίχιση των IP και των MAC address
1.9: alt+F1: tcpdump -i em0 -ev
alt+F2: tcpdump -i em1 -ev
Λαμβάνουμε και στις 2 κονσόλες τα ίδια πακέτα
1.10: Όχι
1.11: Αλλάζει μόνο η ώρα καταγραφής
1.12: Όχι, φαίνεται μόνο ένα hop, από το PC1 στο PC2
1.13: ping 192.168.1.2 (PC1)
tcpdump -i em1 -ev

1.14: ifconfig em0 192.168.2.1/24
Όχι
1.15: Στέλνει request αλλά δε λαμβάνει απάντηση, γιατί δεν βρίσκει host με την προηγούμενη IP
1.16: Όχι
1.17: ifconfig em2 up
ifconfig bridge0 addm em2
ifconfig bridge0 up
1.18: Ναι
1.19: tcpdump -i em1
Δεν εμφανίζονται, έχουν κρατηθεί στους ARP πίνακες των PC1 και PC2 οι MAC Addresses όλων των υπολογιστών και τα πακέτα στέλνονται μέσω των LAN1 και LAN3
1.20: arp -d -a
Καταγράφουμε ένα Broadcast ethertype ARP πακέτο για να βρει σε ποια MAC αντιστοιχεί η ζητούμενη IP
1.21: ifconfig bridge0
1.22: ifconfig bridge0 addr
1.23: em2: PC3, em1: PC2, em0: PC1
1.24: ifconfig bridge0 flush
1.25: ifconfig bridge0 deletem em2
1.26: ifconfig bridge0 destroy
1.27: ifconfig em0 delete

Άσκηση 2:

2.1: 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: ifconfig bridge1 create
ifconfig em0 up
ifconfig em1 up
ifconfig bridge1 addm em0 addm em1 up

2.3: ifconfig bridge2 create
ifconfig em0 up
ifconfig em1 up
ifconfig bridge2 addm em0 addm em1 up

2.4: ifconfig bridge3 create
ifconfig em0 up
ifconfig em1 up
ifconfig bridge3 addm em0 addm em1 up

2.5: PC1: 08:00:27:C7:92:79
PC2: 08:00:27:3D:85:1C
PC3: 08:00:27:5B:E8:AA
PC4: 08:00:27:9E:5D:C3
arp -d -a

2.6: ifconfig bridge1 flush
ifconfig bridge2 flush
ifconfig bridge3 flush

2.7: tcpdump -ve

2.8: ifconfig bridge1 addr, ifconfig bridge2 addr, ifconfig bridge3 addr
B1: 08:00:27:3D:85:1C em1
08:00:27:C7:92:79 em0
B2: 08:00:27:3D:85:1C em0
08:00:27:C7:92:79 em0
B3: 08:00:27:C7:92:79 em0

2.9: PC1: ARP Request για να βρει τη MAC του PC2 → B1: em0 MAC_{PC1}→ LNK1→ B2: em0

MAC_{PC1}→LNK2→B3: em0 MAC_{PC1}
PC2: ARP Reply → B2: em0 MAC_{PC2} & B1: em1 MAC_{PC2}
PC1: ICMP Echo Request → B1 → PC2
PC2: ICMP Echo Reply → B1 → PC1

2.10: Όχι, γιατί ο PC2 έχει καταγράψει την MAC του PC1 επομένως τα ICMP μηνύματα μεταφέρονται κατευθείαν

2.11: Ο PC2 δε γνωρίζει την MAC του PC4 επομένως κάνει Broadcast ARP Request μήνυμα για να την βρει. Η B3 προωθεί το ARP Reply του PC4 στην LNK2 και το παραλαμβάνει η B2 και αυτή με τη σειρά της το προωθεί στη LNK1 όπου το παραλαμβάνει η B1

2.12: Σε όλους τους πίνακες έχει προστεθεί η MAC address του PC3, καθώς όπως και προηγουμένως έγινε Broadcast του ARP Request μηνύματος σε όλες τις γέφυρες και στη συνέχεια έγινε Broadcast του ARP Reply σε όλες τις γέφυρες

2.13: ping 192.198.1.2

2.14: Συνεχίζει κανονικά και δέχεται απαντήσεις συντομότερα

2.15: Δεν δέχεται απάντηση από το PC2, γιατί οι γέφυρες έχουν αποθηκεύσει την παλιά διεύθυνση του PC2 στο LNK1 και στέλνουν τα πακέτα στο LNK1

2.16: Το ping πλέον λαμβάνει απάντηση, γιατί το PC2 έστειλε πακέτο στο B3 το οποίο με τη σειρά του έστειλε στο LNK2 και στο B2 το οποίο ενημέρωσε τη θέση του PC2 στο em1 αντί του em0

2.17: Στον B2 εκτελούμε ifconfig bridge2 και βλέπουμε ότι έχει timeout 1200 seconds δηλαδή μετά από 20 λεπτά θα ενημερωνόταν για τη σωστή θέση του PC2

Άσκηση 3:

3.1: ifconfig bridge1 create
ifconfig bridge1 addm em0 addm em1 up
ifconfig em0 up
ifconfig em1 up

3.2: ifconfig bridge2 create
ifconfig bridge2 addm em0 addm em1 up
ifconfig em0 up
ifconfig em1 up

3.3: PC1: 08:00:27:c7:92:79
PC2: 08:00:27:3d:85:1c
PC3: 08:00:27:5b:e8:aa
arp -d -a

3.4: tcpdump
ping 192.168.1.3 (Από PC2)
Εμφανίζεται κίνηση, γιατί το PC2 δεν γνωρίζει την MAC διεύθυνση του PC3 επομένως κάνει broadcast το ARP request στο LAN2 το οποίο λαμβάνει η B2, το στέλνει στο LNK1 όπου το λαμβάνει η B1 και τελικά το στέλνει στο LAN1 όπου το καταγράφει το PC1

3.5: ping 192.168.1.1

3.6: ifconfig bridgeX addm em2
Ifconfig em2 up

3.7: ifconfig bridgeX addr
Και στις δύο γέφυρες εμφανίζονται όλες οι MAC

3.8: B1: PC1 → em0
PC3 → em1
B2: PC1 → em0
PC3 → em1

3.9: tcpdump -ve

3.10: Όχι

3.11: Αποσυνδέουμε το καλώδιο του LNK1 στα B1 και B2 και στη B2 το PC1 εμφανίζεται στη διεπαφή em2 (LNK2) και το PC3 στην em0 (LNK1). Αυτό γίνεται γιατί το ARP request εκπέμπεται από το PC3 στο LAN2 και το

λαμβάνει η B2. Η B2 το προωθεί στα LNK1 και LNK2 και τα πακέτα λαμβάνει η B1 και καταγράφει τη διεύθυνση του PC3 ανάλογα με το τελευταίο πακέτο που στάλθηκε στη διεπαφή του LNK1 ή LNK2. Έπειτα προωθεί το ARP Request στο LAN1 αλλά και στην άλλη από τις δύο ζεύξεις LNK1 ή LNK2. Η B2 παραλαμβάνει το ARP Request και αλλάζει τη διεπαφή του PC3 σε αυτή από την οποία την παρέλαβε (em0 ή em2). Παράλληλα το ARP Reply του PC1 το οποίο λαμβάνει η B1 προωθείται στη B2 μέσω των LNK1 και LNK2, η B2 θέτει ως διεπαφή του PC1 την τελευταία από την οποία το παρέλαβε (em0 ή em2) και στέλνει στη λανθασμένη διεπαφή του PC3 το πακέτο. Έτσι δημιουργείται μία λούπα.

3.12: ARP Request who-has 192.168.1.1
ARP Reply 192.168.1.1 is-at 08:00:27:c7:92:79

3.13: Η MAC του PC3

3.14: Λόγω της λούπας οι B2 και B1 στέλνουν τα ARP Request στα LAN2 και LAN1 αντίστοιχα

3.15: Η διεπαφή του PC3 στη B2 είναι καταγεγραμμένη στον πίνακα προώθησης επομένως δεν γίνεται broadcast σε όλες τις συνδέσεις παρά μόνο στη διεπαφή αυτή που είναι είτε η LNK1 ή η LNK2

Άσκηση 4:

4.1: ifconfig bridgeX destroy
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down
ifconfig bridgeX create

4.2: ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
ifconfig lagg0 create

4.3: ifconfig lagg0 up laggport em1 laggport em2

4.4: ifconfig lagg0 up laggport em0 laggport em2

4.5: ifconfig bridge1 addm em0 addm lagg0 up

4.6: ifconfig bridge1 addm em1 addm lagg0 up

4.7: Εμφανίζεται ARP Request για τη διεύθυνση του PC3, αφού έχει γίνει εκκαθάριση των πινάκων προώθησης στις γέφυρες B1 και B2. Το PC2 προωθεί το πακέτο στο LAN2 και το λαμβάνει η B2. Αυτή το προωθεί μέσω της lagg0 στη B1 και αυτή με τη σειρά της το προωθεί στο PC1

4.8: tcpdump

4.9: Επιτυγχάνει, στην αρχή στέλνεται πακέτο ARP Request για να βρει τη MAC του PC1, το PC1 στέλνει πακέτο ARP Reply με τη διεύθυνσή του και έπειτα στέλνονται κανονικά τα ICMP Echo Request και Reply

4.10: B1: tcpdump -i em1
B2: tcpdump -i em2
Εμφανίζονται μόνο στη LNK1 λόγω του default πρωτοκόλλου failover, στο οποίο τα πακέτα στέλνονται στο master port (το πρώτο port που προσθέτουμε στο lagg0) εκτός αν δεν είναι διαθέσιμο, οπότε στέλνονται στο επόμενο

4.11: Στέλνονται πλέον στην LNK2 γιατί master είναι πλέον τα άλλα ports (em2)

4.12: Τα πακέτα στέλνονται πάλι στα master port (B1: em1, B2: em0) στο LNK1

Άσκηση 5:

5.1: ifconfig bridgeX destroy
ifconfig lagg0 destroy
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down

5.2: ifconfig bridge1 create
ifconfig bridge1 addm em0 addm em1 addm em2 up
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up

5.3: ifconfig bridge2 create
ifconfig bridge2 addm em0 addm em1 addm em2 up
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up

5.4: ifconfig bridge1 stp em0 stp em1 stp em2

5.5: ifconfig bridge2 stp em0 stp em1 stp em2

5.6: ifconfig bridge1
B1: 08:00:27:17:f3:a2
ifconfig bridge2
B2: 08:00:27:29:55:1d

5.7: H B1

5.8: State: Forwarding, Role: Designated για όλες

5.9: em0: role root (LNK1)

5.10: em2: role alternate state discarding

5.11: em1: role designated state forwarding (LAN2)

5.12: tcpdump -i em0 –ve (B1)
Κάθε 2 δευτερόλεπτα

5.13: IEEE 802.3

5.14: Source: 08:00:27:be:84:a8
Destination: 01:80:c2:00:00:00

5.15: em0 (LAN1)

5.16: Πρώτο byte είναι 01 άρα LSB = 1 είναι multicast

5.17: Root ID: 8000.08:00:27:17:f3:a2
Bridge ID: 8000.08:00:27:17:f3:a2.8001
Root path cost: 0

5.18: tcpdump -i em1 -ve (LNK1)
Bridge ID: 8000.08:00:27:17:f3:a2.8002
Προτεραιότητα είναι το πρώτο μέρος 0x8000 = 32768 (πολλαπλάσιο του 4096)

5.19: Είναι το ID της γέφυρας όπως είδαμε στο ερώτημα 5.6

5.20: Προκύπτει από το άθροισμα της προτεραιότητας (8000) και του port (ifconfig bridge1) π.χ. το em0 έχει port 1 ενώ το em1 έχει port 2

5.21: Ναι παρατηρούμε και στα 2

5.22: LNK1: em1 της B1 = Source της B2
LNK2: em2 της B1 = Source της B2

5.23: LNK1:
root ID: 8000.08:00:27:17:f3:a2
bridge ID: 8000.08:00:27:17:f3:a2.8002
root path cost: 0
LNK2:
root ID: 8000.08:00:27:17:f3:a2
bridge ID: 8000.08:00:27:17:f3:a2.8003
root path cost: 0

5.24: Ναι

5.25: Χρειάζεται περίπου 6 δευτερόλεπτα, δηλαδή 3 hello-time στην default τιμή τους, όπως ορίζει το RSTP

5.26: Όχι, συνεχίζει κανονικά

Άσκηση 6:

6.1: ifconfig em3 up
ifconfig bridge1 addm em3
ifconfig bridge1 stp em3

6.2: ifconfig em3 up
ifconfig bridge2 addm em3
ifconfig bridge2 stp em3

6.3: ifconfig bridge3 create
ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
ifconfig bridge3 addm em0 addm em1 addm em2 up
ifconfig bridge3 stp em0 stp em1 stp em2

6.4: ifconfig bridgeX flush
Είναι επιτυχές

6.5: ifconfig bridge1 priority 0

6.6: pathcost = 20000
1 Gbit/s = 20000

6.7: B1: root-pathcost = 0
B2: root-pathcost = 20000
Η B1 είναι γέφυρα ρίζα επομένως το κόστος είναι 0, ενώ από τη B2 έχουμε 1 Gbit/s ταχύτητα Ethernet

6.8: ifconfig bridge3
em0 (LNK3) role root το οποίο περιμένουμε αφού είναι συνδεδεμένη με τη ριζική γέφυρα B1

6.9: Alternate discarding

6.10: 20000

6.11: ping 192.168.1.3

6.12: Αναμένουμε κόστος 40000 από την LNK4 (20000 LNK4 + 20000 LNK1) άρα θα δώσουμε κόστος λίγο πάνω από 40000 στην LNK3
ifconfig bridge3 ifpathcost em0 40001

6.13: 4 seconds

6.14: B3: LNK3 em0: role alternate state discarding
B2: LNK4 em2: role designated state forwarding

6.15: Όχι
6.16: root-pathcost: 40000
6.17: Περίπου 10 δευτερόλεπτα
6.18: Περίπου 4 δευτερόλεπτα
6.19: em2: role designated state forwarding
em3: role backup state discarding
6.20: ifconfig bridge3 ifpathcost em0 0
Θέτουμε το κόστος σε 0 όπως ήταν αρχικά

Άσκηση 7:

7.1: 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: ifconfig em0.5 create up
ifconfig em0.6 create up

7.3: ifconfig em1.6 create up
ifconfig em3.5 create up

7.4: ifconfig em create vlan 6 vlandev em0 inet 192.168.6.2/24

7.5: ifconfig em0.6 create up
ifconfig em3.6 create up

7.6: ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.3/24

7.7: ifconfig em0.5 create up
ifconfig em2.5 create up

7.8: Ναι

7.9: B1: ifconfig bridge1 -stp em0

7.10: tcpdump -vvvex

7.11: ethertype ARP (0x0806)
ethertype IPv4 (0x0800)

7.12: Περιλαμβάνουν τα ίδια πλαίσια με τα προηγούμενα και επιπλέον πεδία ethertype και vlan

7.13: 802.1Q (0x8100) μέσα στην επικεφαλίδα υπάρχει το πεδίο ethertype που είναι ARP ή IPv4

7.14: Μετά το πρώτο πεδίο length, στο πεδίο vlan

7.15: tcpdump -i em0.5 -vvvex

7.16: ARP 0x0806
IPv4 0x0800
Δεν υπάρχει πλέον το πεδίο vlan

7.17: ifconfig bridge1 stp em0
tcpdump -i em0 -vvvex

7.18: Δεν είναι ίδιου τύπου γιατί λείπει το πεδίο Ethertype (Ethernet II) και έχει αντικατασταθεί από το πεδίο length (IEEE 802.3)

7.19: not stp