Εργαστήριο Δικτύων Υπολογιστών
Εργαστηριακή Άσκηση 3
Δημήτριος Κόγιος
03119220
Όνομα PC: lekog-HP-Laptop-15s-fq1xxx

Άσκηση 1:

1.1)PC1 : ifconfig em0 192.168.1.1/24
PC2: ifconfig em0 192.168.1.2/24

1.2) ifconfig em0 up
ifconfig em1 up
ifconfig em2 up
ifconfig em3 up

1.3) Παρατηρούμε μηνύματα host is down. Δηλαδή δεν υπάρχει επικοινωνία μεταξύ των PC1 και PC2.

1.4) Δεν βλέπουμε ICMP πακέτα αλλά μόνο ARP requests στα οποία δεν απαντάει κανείς. Αυτό συμβαίνει αφού τα PC1 και PC2 βρίσκονται σε διαφορετικό LAN.

1.5) ifconfig bridge0 create
ifconfig bridge0 addm em0 addm em1 up

1.6) Ναι πλέον επικοινωνούν.

1.7) Το time to live είναι 64 άρα μηδέν βήματα μεταξύ PC1 και PC2 αφού το 64 είναι η αρχική τιμή. Δηλαδή η γέφυρα δεν μετράει ως δρομολογητής.'

1.8) Στον πίνακα ARP του κάθε εικονικού μηχανήματος υπάρχει μόνο η διεύθυνση MAC του εαυτού του και του άλλου εικονικού μηχανήματος χωρίς να υπάρχει καταχώριση για τη γέφυρα.

1.9) Μέσω tcpdump -i em0 -e -vvv -l | tee LAN1 . Μετά ALT+F2 για νέο τερματικό και tcpdump -i em1 -e -vvv -l | tee LAN2. Επιβεβαιώνουμε ότι το B1 προωθεί πλαίσια Ethernet μεταξύ LAN1 και LAN2.

1.10) Όχι καμία αλλαγή. Διευθύνσεις MAC και IPv4 είναι ανέπαφες και ίδιες με αυτές που μας έδειξαν οι πίνακες ARP στο ερώτημα 1.8.

1.11) Όχι δεν παρατηρώ κάποια αλλαγή εκτός από τα timestamps στα οποία έγιναν capture τα πακέτα από το tcpdump.

1.12) Όχι δεν φαίνεται να παρεμβάλλεται τίποτα μεταξύ τους αφού επιτυγχάνει το πακέτο με TTL = 1.

1.13) Μέσω ping 192.168.1.2 στο PC1 και tcpdump -i em1 στο B1

1.14) Ναι η γέφυρα συνεχίζει να προωθεί τα ICMP echo requests προς το LAN2 αφού εκεί υπάρχει καταγραφή για τη MAC διεύθυνση που έχουν τα Ethernet frames.

1.15) Όχι, δεν βλέπουμε απάντηση στο PC1 αφού δεν υπάρχει μηχάνημα με 192.168.1.2 το οποίο να απαντάει .

1.16) Όχι αφού η διεπαφή em2 του B1 δεν έχει συνδεθεί στο bridge0.

1.17) ifconfig em2 up και μετά ifconfig bridge0 addm em2

1.18) Ναι το PC1 απαντάει.

1.19) Δεν παράγεται κίνηση στο LAN2. Αυτό συμβαίνει αφού η γέφυρα γνωρίζει σε ποια θύρα της βρίσκονται τα PC1 και PC3(το έμαθε λόγω του ping του προηγούμενου ερωτήματος) και προωθεί τα πλαίσια στη σωστή θύρα χωρίς να χρειάζεται να γίνει πλημμύρα.

1.20) Παρατηρούμε το ARP request που παράγεται γιατί αδειάσαμε τον πίνακα ARP. Το ARP request είναι broadcast και έτσι η γέφυρα το προωθεί σε όλα τα LAN.

1.21) ifconfig bridge0

1.22) ifconfig bridge0 addr

1.23) Στα μηχανήματα PC1 και PC3.

1.24) ifconfig bridge0 flush

1.25) ifconfig bridge0 deletem em2

1.26) ifconfig bridge0 destroy

1.27) ifconfig em0 delete και στα τρία εικονικά μηχανήματα.

Άσκηση 2:

2.1) PC1: ifconfig em0 192.168.1.1/24
PC2: ifconfig em0 192.168.1.2/24
PC3: ifconfig em0 192.168.1.3/24
PC4: ifconfig em0 192.168.1.4/24

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) Βλέπουμε τις διευθύνσεις MAC μέσω ifconfig:
PC1 : 08:00:27:55:bc:a2
PC2: 08:00:27:73:6d:03
PC3: 08:00:27:fc:47:6d
Pc4: 08:00:27:88:9f:75

Αδειάζουμε τις πίνακες ARP με arp -d -a.

2.6) Μέσω ifconfig bridge1 flush, ifconfig bridge2 flush, ifconfig bridge3 flush.

2.7) tcpdump -l | tee PCx όπου x = 1,2,3,4

2.8) y = 08:00:27

B1:
y:73:6d:03 em1
y:55:bc:a2 em0

B2:
y:73:6d:03 em0
y:55:bc:a2 em0

B3:
y:55:bc:a2 em0

2.9) Αρχικά γίνεται ARP request από το PC1 προς το PC2. Το B1 μαθαίνει ότι το PC1 βρίσκεται στο em0 και προωθεί το ARP reqeust στις υπόλοιπες εξόδους του εκτός από την em0. Τώρα το ARP request βρίσκεται στο LNK1. Το B2 μαθαίνει ότι το PC1 βρίσκεται στο em0 και προωθεί το ARP request στο em1 του. Το B3 μαθαίνει ότι το PC1 βρίσκεται στο em0 του. Το PC2 απαντάει στο ARP request του PC1. Το B1 μαθαίνει ότι το PC2 βρίσκεται στο em1 του και προωθεί το ARP reply στο PC1 που γνωρίζει ,εξαιτίας του ARP request, ότι βρίσκεται στη διεπαφή em0 του. Επίσης, το B2 μαθαίνει ότι το PC2 βρίσκεται στο em0 του. Μετά το PC1 στέλνει ICMP echo request προς το PC2 και η B1 γνωρίζει ότι το PC2 βρίσκεται στο em1 του οπότε το προωθεί εκεί. Το B2 γνωρίζει ότι το PC2 βρίσκεται στο em0 του οπότε δεν προωθεί το πακέτο. Το PC2 στέλνει ICMP echo reply προς το PC1. Το B2 γνωρίζει ότι το PC1 βρίσκεται στην em0 εκεί δηλαδή από όπου του ήρθε το πακέτο οπότε δεν το προωθεί. Αντιθέτως, το B1 προωθεί το πακέτο στο PC1. Εν τέλει, όλα τα bridges

μαθαίνουν που βρίσκεται το PC1 λόγω του ARP request. Ύστερα, τα bridge 1 και 2 μαθαίνουν τη θέση και του PC2 λόγω της συνομιλίας μεταξύ PC1 και PC2 ενώ το bridge 3 αγνοεί τη θέση του PC2.

2.10) Όχι, πλέον δεν υπάρχουν ARP πακέτα αφού τα PC1 και PC2 συνομίλησαν πριν λίγο και γνωρίζουν τις MAC διευθύνσεις τους οπότε στο δίκτυο θα υπάρχει μόνο ICMP κίνηση. Τα bridges B1 και B2 που παρεμβάλλονται μεταξύ PC1 και PC2 γνωρίζουν τις θέσεις τνω PC1 και PC2 και προωθούν τα πακέτα κατευθείαν εκεί που πρέπει.

2.11) y = 08:00:27

B1:
y:88:9f:75 em1
y:73:6d:03 em1
y:55:bc:a2 em0

B2:
y:88:9f:75 em1
y:73:6d:03 em0
y:55:bc:a2 em0

B3:
y:88:9f:75 em1
y:73:6d:03 em0
y:55:bc:a2 em0

Λόγω του ARP reply που προκαλείται από το ARP request του PC2 προς το PC4. Το ARP reply όταν φτάσει στο LNK1, θα

διαβαστεί από το B1 και έτσι το B1 θα μάθει την MAC διεύθυνση του PC4.

2.12) y = 08:00:27

B1:
y:fc:47:6d em1
y:88:9f:75 em1
y:73:6d:03 em1
y:55:bc:a2 em0

B2:
y:fc:47:6d em1
y:88:9f:75 em1
y:73:6d:03 em0
y:55:bc:a2 em0

B3:
y:fc:47:6d em0
y:88:9f:75 em1
y:73:6d:03 em0
y:55:bc:a2 em0

Πλέον όλες οι γέφυρες μαθαίνουν τη και τη θέση του PC3 λόγω του ARP request που στέλνει.

2.13) Done.

2.14) Τα PC2 και PC4 συνεχίζουν κανονικά να συνομιλούν.

2.15) Το PC1 έχει σταματήσει να λαμβάνει replies από το PC2 γιατί η B1 τα στέλνει στο em1 της όμως η B2 συνεχίζει να πιστεύει ότι το PC2 βρίσκεται στο em0 της και δεν τα προωθεί πιο δεξιά.

2.16) Το PC1 ξαναξεκινάει να λαμβάνει ICMP echo replies από το PC2 καθώς με το που έστειλε ένα πακέτο το PC2 προς το PC3 ξαναεκπαιδεύτηκε η B2 και κατάλαβε ότι πλέον το PC2 βρίσκεται στο em1 της και όχι στο em0.

2.17) Μέχρι να λήξει η καταχώριση του PC2 στον πίνακα προώθησης της γέφυρας B2. Τότε θα γινόταν flooding και το πακέτο θα έφτανε στο B3 που θα το προωθούσε στο PC2.

Άσκηση 3:

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

3.2) ifconfig bridge2 create
ifconfig em0 up
ifconfig em2 up
ifconfig bridge2 addm em0 addm em2 up

3.3) PC1: 08:00:27:55:bc:a2
PC2: 08:00:27:73:6d:03
PC3: 08:00:27:fc:47:6d

3.4) Βλέπουμε μόνο το πρώτο ARP request. Με το ARP request η B2 μαθαίνει ότι ένα από τα PC2 ή PC3 (εξαρτάται από ποιο κάναμε ping) είναι στα δεξιά της ενώ με το ARP reply μαθαίνει

ότι και το άλλο PC είναι δεξιά της οπότε όποια μετέπειτα κίνηση μένει στο LAN2.

3.5) ping 192.168.1.1

3.6) Τις κάνω up και μετά addm.

3.7) y = 08:00:27

B1:
y:55:bc:a2 em0
y:fc:47:6d em1
y:73:6d:03 em1

B2:
y:55:bc:a2 em0
y:fc:47:6d em2
y:73:6d:03 em2

3.8) PC1:
Στο B1 το PC1 εμφανίζεται στο em0 (που είναι στο LAN1) ενώ στο B2 το PC1 εμφανίζεται στο em0 (που είναι στο LNK1).

PC3:
Στο B1 το PC3 εμφανίζεται στο em1 (που είναι στο LNK1) ενώ στο B2 το PC3 εμφανίζεται στο em2 (που είναι στο LNK2).

3.9) tcpdump -e

3.10) arp -d -a
Όχι το ping δεν είναι επιτυχές.
3.11) Αποσύνδεσα το LNK2 και από τα 2 bridges.

PC1 στο em0 (LNK1)
PC3 στο em1 (LNK2)

Το PC3 στέλνει ARP request , το B2 μαθαίνει ότι το PC3 βρίσκεται δεξία του και προωθεί το request στο B1 μέσω των LNK1 και LNK2.

Το B1 λαμβάνει τα ARP request στα LNK1 και LNK2 και μαθαίνει ότι το PC3 βρίσκεται δεξιά του. Ύστερα προωθεί το request που του ήρθε στο LNK1 και στο LAN1 το request που του ήρθε από το LNK2 οπότε το B2 αλλάζει και πλέον πιστεύει ότι το PC3 είναι αριστερά του. Το ίδιο γίνεται για το request που ήρθε στο LNK2.

Όταν το PC1 δημιουργήσει το ARP reply , το στέλνει στο B1 το οποίο λουπάρει με το B2 καθώς πιστεύουν ότι το PC3 βρίσκεται αριστερά από το B2 με αποτέλεσμα να λουπάρουν εκεί τα 2 ARP πακέτα.

3.12) Request who-has 192.168.1.1 tell 192.168.1.3

Reply 192.168.1.1 is-at 08:00:27:55:bc:a2

3.13) 08:00:27:fc:47:6d δηλαδή MAC του PC3.

3.14) Γιατί τα πακέτα που πάνε στο LNK1 του B1 προωθούνται στα LAN1 και LNK2 (αντίστοιχα αυτά που πάνε στο LNK2 του B1). Ενώ τα πακέτα που πάνε στο LNK1 του B2 προωθούνται στο LAN2 και LNK2 B2 (αντίστοιχα αυτά που πάνε στο LNK2 του B2). Αυτό συμβαίνει γιατί τα ARP requets είναι broadcast και γίνονται πάντα flooding.

3.15) Γιατί το B2 πιστεύει ότι το PC3 είναι αριστερά του όπως εξήγησα πιο πάνω λόγω της λούπας που δημιουργείται.

Άσκηση 4:

4.1) ifconfig bridge1 destroy
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down
ifconfig bridge1 create

ifconfig bridge2 destroy
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down
ifconfig bridge2 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 em1

4.5) ifconfig bridge1 addm em0 addm lagg0 up

4.6) ifconfig bridge2 addm em2 addm lagg0 up

4.7) Βλέπουμε μόνο το broadcast ARP request.

4.8) tcpdump

4.9) arp -d -a για να αδειάσει ο ARP πίνακας του PC3.

Ναι, το ping είναι επιτυχές και παρατηρούμε ARP πακέτα (request και reply).

4.10) tcpdump -i em1 και στο B1 και στο B2.

Η κίνηση εμφανίζεται στην καταγραφή του B1 δηλαδή στο LNK1. Ο λόγος είναι ότι το προεπιλεγμένο πρωτόκολλο συνάθροισης είναι failover το οποίο ορίζει μία ζεύξη ως master και τις υπόλοιπες ως backup για να χρησιμοποιηθούν άμα η master port γίνει unavailable.

4.11) Πλέον βλέπουμε κίνηση στην καταγραφή του B2 δηλαδή στο LNK2 αφού η ζεύξη LNK1 έγινε unavailable.

4.12) Η κίνηση ξαναμεταφέρθηκε στο LNK1 αφού είναι master.

Άσκηση 5:

5.1) ifconfig bridge1 destroy
ifconfig lagg0 destroy
ifconfig em0 down
ifconfig em1 down
ifconfig em2 down

ifconfig bridge2 destroy
ifconfig lagg0 destroy
ifconfig em0 down
ifconfig em1 down

ifconfig em2 down

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

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

5.4) ifconfig bridge1 stp em0 stp em1 stp em2

5.5) ifconfig bridge2 stp em0 stp em1 stp em2

5.6) B1: 32768.08:00:27:42:09:19
B2: 32768.08:00:27:1f:cc:54
της μορφής priority.mac-address

5.7) Είναι η B2, το βλέπουμε από το root id του ifconfig.

5.8) Και οι 3 έχουν role designated και state forwarding

5.9) Είναι η διεπαφή LNK1 αφού έχει role root.

5.10) role alternate (εναλλακτική της ριζικής θύρας προς τη γέφυρα ρίζα) state discarding

5.11) role designated state forwarding

5.12) Επειδή είναι root η B2 που δεν συνδέεται με LAN1 θα κάνω καταγραφή για LAN2.
tcpdump -i em2 -e -vvv

Εκπέμπονται κάθε 2 δευτερόλεπτα αφού τόσο είναι το hello-time.

5.13) IEEE 802.3

5.14) Πηγή: 08:00:27:84:fa:bd (που είναι η διεπαφή για την οποία κάνουμε capture)
Προορισμός: 01:80:c2:00:00:00 (multicast address για Spanning Tree Protocol)

5.15) Στην em2 που κάνουμε capture.

5.16) Multicast, είναι η multicast address για Spanning Tree Protocol.

5.17) root ID: 8000.08:00:27:1f:cc:54
bridge ID: 8000.08:00:27:1f:cc:54.8003
root path cost: 0

5.18) tcpdump -i em0 -e -vvv (δηλαδή καταγραφή στο LNK1 της root bridge)

Η προτεραιότητα είναι το πρώτο μέρος του Bridge ID αφού 0x8000 = 32768.

5.19) Το δεύτερο μέρος είναι η MAC διεύθυνση της γέφυρας ενώ το τρίτο είναι το portID.

5.20) Όχι.

5.21) Στο em0 δηλαδή στη διεπαφή που είναι στο LAN1.

5.22) root-id 8000.08:00:27:1f:cc:54
bridge-id: 8000.08:00:27:42:09:19.8001
root-pathcost: 20000

5.23) Ναι είναι επιτυχές.

5.24) Περίπου 8 δευτερόλεπτα.
Αυτό συμβαίνει γιατί τα οι αλλαγές τοπολογίας μεταφέρονται από non root bridges προς τη root η οποία τους δίνει την εντολή να να αλλάξουν τα forwarding tables τους όσο πιο γρήγορα μπορούν. Δεδομένου ότι BPDUs στέλνονται κάθε 2 δευτερόλεπτα ο χρόνος είναι αναμενόμενος.

5.25) Όχι δεν υπάρχει διακοπή.

Άσκηση 6:

6.1) ifconfig em3 up
ifconfig bridge1 addm em3
ifconfig bridge1 stp em3

6.2) ifconfig em3 up
ifconfig bridge1 addm em3
ifconfig bridge1 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 , X = {1,2,3}
Ναι, το ping είναι επιτυχές μετά από μερικές προσπάθειες.

6.5) ifconfig bridge1 priority 28672

6.6) Όλα τα κόστη είναι 20000. Αυτό σημαίνει ότι έχουμε bandwidth 1 Gbps. ((20 Tbit/s) / bandwidth)

6.7) Από τη B1 λαμβάνει root-pathcost = 0 αφού η bridge1 είναι root.
Από τη B2 λαμβάνει 20000 όπως είπαμε στο προηγούμενο ερώτημα.

6.8) Η LNK3 αφού από εκεί βλέπει χαμηλότερο root path cost.

6.9) Στο B3 , η LNK4 έχει role alternate και state discarding.

Στο B2, η LNK4 έχει role designated και state forwarding.

6.10) 20000

6.11) ping 192.168.1.3

6.12) ifconfig bridge3 ifpathcost em0 40001 γιατί θέλω η ζεύξη LNK3 του B3 να έχει μεγαλύτερο κόστος από το άρθοισμα

κόστων της ζεύξης LNK4 + LNK1/2. Δεδομένου ότι κόστος κάθε ζεύξης είναι 20000 , επιλέγουμε οποιαδήποτε τιμή πάνω από 40000.

6.13) Περίπου 4 δευτερόλεπτα.

6.14) B3 στο LNK3 έχει role alternate και state discarding
B2 στο LNK4 έχει role designated και state forwarding.

6.15) Όχι, συνεχίζει να λαμβάνει ο και 2000.

6.16) Ναι, πλέον έχει root path cost = 40000 αφού προηγούνται 2 ζεύξεις αντί για μία.

6.17) Περίπου 8 δευτερόλεπτα.

6.18) Περίπου 7-8 δευτερόλεπτα.

6.19) em2 είναι designated και state forwarding
em3 είναι backup και state discarding

6.20) ifconfig bridge3 ifpathcost em0 20000

Το επαναφέρουμε στο 20000.

Άσκηση 7:

7.1) ifconfig em0.5 create
ifconfig em0.5 192.168.5.1/24
ifconfig em0.6 create
ifconfig em0.6 192.168.6.1/24

7.2) ifconfig em0.5 create
ifconfig em0.6 create

7.3) ifconfig em1.6 create (em1 του B1 είναι στο LNK1)
ifconfig em3.5 create (em3 του B1 είναι στο LNK3)

7.4) ifconfig em0.6 create
ifconfig em0.6 192.168.6.2/24

7.5) ifconfig em0.6 create (em0 του B2 είναι στο LNK1)
ifconfig em2.6 create (em2 του B2 είναι στο LAN2)
7.6) ifconfig em0.5 create
ifconfig em0.5 192.168.5.3/24

7.7) ifconfig em0.5 create (em0 του B3 είναι στο LNK3)
ifconfig em2.5 create (em2 του B3 είναι στο LAN3)

7.8) Ναι και ναι.

7.9) ifconfig bridge1 -stp em0

7.10) tcpdump -i em0 -vvv -xx -e

7.11) ARP -> 0x0806
IPv4 -> 0x0800

7.12) Πριν το ether type έχουν το 802.1Q VLAN Tag το οποίο είναι 4 bytes : 8100 0006

7.13) Το tcpdump εμφανίζει και για τα δύο ethertype = 8100 όμως πιο μετά λέει και τα ethertypes που βρήκαμε στο ερώτημα 7.11.

ARP -> 0x806

IPv4 -> 0x0800

Το ether type βρίσκεται μετά το 802.1Q VLAN tag

7.14) Μέσα στο 802.1Q VLAN tag το οποίο βρίσκεται μέσα στο Ethernet header μετά τη διεύθυνση πηγής (4 bytes η διεύθυνση προορισμού + 4 bytes η διεύθυνση πηγής). Συγκεκριμένα, το 802.1Q VLAN Tag έχει μήκος 4 bytes, τα πρώτα 2 είναι πάντα 0x8100 ενώ τα τελευταία 12 bits των επόμενων 2 bytes είναι το VLAN ID.

7.15) tcpdump -i em0.5 -vvv- xx -e

7.16) ARP -> 0x0806

IPv4 -> 0x0800

Δεν υπάρχει πληροφορία για το VLAN.

7.17) ifconfig bridge1 stp em0

tcpdump -i em0 -vvv -xx -e

7.18) Είναι IEEE 802.3 οπότε στη θέση του Ethertype έχουν length δηλαδή το μήκος του payload.

7.19) tcpdump '(not stp)' (και μετά όποιο άλλο φίλτρο θέλουμε)