Ονοματεπώνυμο: Πυλιώτης Αθανάσιος	**Όνομα PC:** DESKTOP-5DLG3IF
Ομάδα: 1	**Ημερομηνία:** 30/03/23

Εργαστηριακή Άσκηση 6

Εισαγωγή στο Quagga και FRRouting (FRR)

Άσκηση 1: Γνωριμία με το περιβάλλον του FRR

1.1) **telnet localhost 2601**

vty password is not set.

Connection closed by foreign host.

1.2) **R0# vtysh** → Η εντολή που μας δίνει πρόσβαση διαχειριστή στο περιβάλλον FRR

1.3) **R0#** ? → Παρατηρούμε 22 διαθέσιμες εντολές προς εκτέλεση.

1.4) Παρατηρούμε πως συμπληρώνει αυτόματα την εντολή.

1.5) Δεν την συμπληρώνει. Αυτό συμβαίνει καθώς δεν υπάρχει μοναδικότητα της εντολής, επειδή υπάρχει κι η copy σαν εντολή και θέλει περισσότερα γράμματα.

Όταν πατάμε ? εμφανίζει όλες τις εντολές που ξεκινάνε με co, που είναι η copy και η configure.

1.6) **R0# sh ver (tab)→ show version**, είναι απλά συντομογραφία και τη συμπληρώνει αυτόματα λόγω μοναδικότητας των.

1.7) **R0# wr t** θα περίμενα, αλλά δούλευε και το **w t**. Συνεπώς αυτό το τελευταίο είναι το πιο σύντομο.

1.8) **R0# show running-config ή sh r**

1.9) **R0# configure (terminal)** → Πάμε απευθείας στο global configuration

1.10) **R0(config)# hostname R1** → άλλαξε το prompt από R0(config)# σε R1(config)#.

1.11) **R0(config)# password ntua** → θέλει πλέον συνθηματικό και χρήση enable για να εισέλθει κάποιος χρήστης.

1.12) 2 φορές έπρεπε να κάνουμε exit για να επιστρέψουμε.

1.13) **root@R0# telnet localhost 2601**

Μας εμφανίζει prompt για να βάλουμε κωδικό (δουλεύει το ntua)

1.14) Βρισκόμαστε στο επίπεδο User EXEC, το οποίο είναι προφανές από την έλλειψη εντολών και από το R0>.

1.15) Παρατηρούμε 9 εντολές διαθέσιμες σε εμάς.

1.16) Παρατηρούμε 13 λιγότερες εντολές, το οποίο είναι λογικό αφού δεν θέλουμε κάθε χρήστης να μπορεί να κάνει ό,τι αλλαγές θέλει στο σύστημα, όπως στα UNIX είναι ο root (SUDO) user και ο απλώς user. Έτσι εδώ στο privileged EXEC έχει περισσότερες εντολές.

1.17) **R0> show interface**

Δείχνει δεδομένα για όλες τις διεπαφές του δικτύου.

1.18) **R0> show ip forwarding**

Μας απαντάει IP forwarding is on

1.19) **R0> show ip route**

Εμφανίζει το routing table.

1.20) Όχι δεν μπορούμε, καθώς δεν είμαστε σε privileged EXEC. Δεν θέλουμε ο χρήστης να ξέρει όλες τις παραμέτρους του συστήματος. Πρέπει να είμαστε σε περιβάλλον privileged EXEC.

1.21) **R0> enable** (μπαίνουμε στο privileged EXEC)

1.22) **R0# show running-config**

Ναι μπορούμε να δούμε πλέον τη παραμετροποίηση. Κωδικό δεν μας ζήτησε κάπου πέρα από το telnet για να συνδεθούμε. Ο password είναι ntua.

1.23) **R0# list**

Δείχνει όλες τις διαθέσιμες εντολές που μπορούμε να χρησιμοποιήσουμε.

1.24) **R0(config)# enable password ntua**

Μετά μας ζητάει κωδικό και για το privileged EXEC, το οποίο είναι πιο ασφαλές.

1.25) **R0(config)# service password-encryption**

Αποτρέπει την εμφάνιση του κωδικού κατά την παραμετροποίηση.

1.26) Από πλευράς ασφάλειας θυμόμαστε πως το telnet δεν είναι καθόλου καλό και είναι πολύ εύκολο να εισέλθει κάποιος και να αποκλέψει τα δεδομένα της σύνδεσης. Προφανώς η έμμεση σύνδεση με ssh (**secure** shell) είναι καλύτερο, το λέει μέχρι και το όνομα του, έχει φτιαχτεί ώστε να συνδεόμαστε με ασφάλεια μέσω του command line σε απομακρυσμένους διακομιστές και για να αποκρυπτογραφεί τα δεδομένα κατά τη σύνδεση. Επίσης, προστατεύει καλύτερα τους κωδικούς μας και ό,τι είδους πληροφορία ανταλλάσσεται, αφού γίνεται κρυπτογραφημένα.

Άσκηση 2: Δρομολόγηση σε ένα βήμα

2.1) PC1: **ifconfig em0 192.168.1.2/24**

PC2: **ifconfig em0 192.168.2.2/24**

2.2) vtysh → R0(config): **hostname R1**

R1: **configure** → R1(config)# **interface em0** → R1(config-if)# **ip address 192.168.1.1/24**

R1(config-if)# **exit** → R1(config)# **interface em1** → R1(config-if)# **ip address 192.168.2.1/24**

2.3) R1(config)# **do show interface**

2.4) R1(config)# **do show ip forwarding** (είναι ON)

2.5) PC1: **route add -net 192.168.2.0/24 192.168.1.1**

2.6) PC2: **route add -net 192.168.1.0/24 192.168.2.1**

2.7) PC1: **ping -c 1 192.168.2.2**

Ναι επικοινωνούν οι υπολογιστές.

2.8) R1(config)# **interface em0** → R1(config-if)# **ip address 192.168.1.200/24** → R1(config-if)# **exit** → R1(config)# **do show interface em0**

Παρατηρούμε πως πλέον στη διεπαφή υπάρχουν 2 διευθύνσεις,

192.168.1.1/24

192.168.1.200/24 secondary

Άρα προστέθηκε και δευτερεύουσα διεύθυνση IP.

2.9) R1(config)# **exit** → R1# **exit** → root@R0# **ifconfig em0**

Φαίνεται να συμφωνούν, με μόνη εξαίρεση πως δεν γράφει ότι είναι secondary, αλλά οι διευθύνσεις είναι ίδιες κατά τα άλλα.

2.10) **vtysh** → R1# **configure** → R1(config)# **interface em0** → R1(config-if)# **no ip address 192.168.1.200/24** → R1(config-if)# **exit** → R1(config)# **do show interface em0**

Βλέπουμε πως την εμφανίζει ακόμα, αλλά αν βγούμε και κάνουμε ifconfig em0 παρατηρούμε πως όντως έχει αφαιρεθεί.

2.11) R1# **write file**

2.12) Τα αρχεία που ενημερώνονται είναι τα **/usr/local/etc/frr/zebra.conf** και **/usr/local/etc/frr/staticd.conf**

Άσκηση 3: Δρομολόγηση σε περισσότερα βήματα

3.1) **netstat -rn**, **ifconfig em0** και ελέγχουμε πως όλα είναι ορισμένα όπως θέλουμε (και είναι). Ελέγχουμε και τα IP forwarding. Κάνουμε και ping και όλα δουλεύουν.

3.2) R1: **vtysh** → R0# **configure** → R0(config)# **hostname R1** → R1(config)# **interface em1** → R1(config-if)# **no ip address 192.168.2.1/24** → R1(config-if)# **ip address 172.17.17.1/30**
Το em0 είναι ήδη ορισμένο σε 192.168.1.1/24 από πριν και για αυτό δεν αλλάζει.
Παράλληλα έχουμε αλλάξει και τα internal networks σε em0: LAN1, em1: WAN1

3.3) R2: **vtysh** → R0# **configure** → R0(config)# **hostname R2** → R2(config)# **interface em1** → R2(config-if)# **ip address 192.168.2.1/24** → R2(config)# **interface em0** → R2(config-if)# **ip address 172.17.17.2/30**
Παράλληλα έχουμε αλλάξει και τα internal networks σε em0: WAN1, em1: LAN2

3.4) R1(config)# **ip route 192.168.2.0/24 172.17.17.2**
Ελέγχουμε με **do show ip route** και βλέπουμε αριστερά να έχει S
Χρειάστηκε να κάνουμε και **write file** και μετά **reboot** για να δουλέψει (δεν θα έπρεπε).

3.5) R2(config)# **ip route 192.168.1.0/24 172.17.17.1**

3.6) Πρέπει να έχει προηγηθεί
R1(config)# **password ntua**
PC1: **telnet 192.168.1.1 2601**
Δουλεύει αν έχουμε δώσει κωδικό, αλλιώς αποτυγχάνει.

3.7) Πρώτα θέτουμε κωδικό και στο R2. Ωστόσο, παρατηρούμε στο PC1 πως δεν υπάρχει καμία κατάλληλη εντολή στο list και συνεπώς δεν μπορούμε να συνδεθούμε στην αντίστοιχη υπηρεσία του R2 από τη zebra του R1. Πρακτικά το zebra δεν μας παρέχει την κατάλληλη υπηρεσία.

3.8) Θα στέλναμε στην 192.168.2.1 γιατί αυτή γνωρίζουμε γενικά και σε αυτή θα καλούσαμε. Ωστόσο, θα μπορούσαμε να προσπαθήσουμε και στην 172.17.17.2 αλλά θεωρητικά δεν θα ξέραμε την ύπαρξη της.
PC1: **telnet 192.168.2.1 2601**
Το δοκιμάσαμε και δούλεψε.

3.9) PC2: **telnet 192.168.2.1 2601**
R0> **who**
Δείχνει και τους 2
Όχι δεν τον εμφανίζει τον τοπικό, λογικά για ασφάλεια

3.10) Από τις πομακρυσμένες διευθύνσεις δεν ήταν διαθέσιμα τα ping και τα traceroute και δεν τα έδειχνε στο list ακόμα και με όλα τα privileges. Από τη τοπική μπορούμε να τα εκτελέσουμε

αλλά αποτυγχάνουν.

3.11) Αποτυγχάνουν επειδή δεν υπάρχουν στα routing tables των PC1, PC2 εγγραφές που να οδηγούν στο WAN1.

3.12) Για να πετύχει μπορούμε να προσθέσουμε στατικές εγγραφές στους πίνακες προώθησης, είτε για τον host που δεν μπορούν να προσεγγίσουν απευθείας είτε για το δίκτυο στο οποίο ανήκει (το 172.17.17.0/30).

Θα κάναμε:

PC1: **route add -net 172.17.17.0/30 192.168.1.1**

PC2: **route add -net 172.17.17.0/30 192.168.2.1**

Άσκηση 4: Εναλλακτικές διαδρομές

4.1) PC1: **ifconfig em0 192.168.1.2/24**
route add default 192.168.1.1
PC2: **ifconfig em0 192.168.2.2/24**
route add default 192.168.2.1

4.2) R1: **cli** → route.ntua.lab# **configure terminal** → router.ntua.lab(config)# **hostname R1** → R1(config)# **interface em0** → R1(config-if)# **ip address 192.168.1.1/24** → R1(config-if)# **exit** → R1(config)# **interface em1** → R1(config-if)# **ip address 172.17.17.1/30** → R1(config-if)# **exit** → R1(config)# **interface em2** → R1(config-if)# **ip address 172.17.17.5/30** → R1(config-if)# **exit**
em0: LAN1, em1: WAN1, em2: WAN2

4.3) R1(config)# **ip route 192.168.2.0/24 172.17.17.2**

4.4) R1(config)# do show ip route
C>* 127.0.0.0/8 is directly connected, lo0
C>* 172.17.17.0/30 is directly connected, em1
C>* 172.17.17.4/30 is directly connected, em2
C>* 192.168.1.0/24 is directly connected, em0
S>* 192.168.2.0/24 [1/0] via 172.17.17.2, em1

4.5) Οι διαδρομές προς τα απευθείας δηλώνονται ως “directly connected” και έχουν ένα C>* αριστερά (λογικά από το Connected)

4.6) Φαίνεται πως έχει οριστεί στατικά επειδή έχει το S>* στα αριστερά από το static.

4.7) Ναι συμφωνούν οι δύο πίνακες (με το netstat -rn βλέπουμε και την ip στις διεπαφές μας). Φαίνονται επίσης ποια είναι host και ποια είναι statically set.

4.8) Οι σημαίες είναι οι UG1 και σημαίνουν: U → up, G → Gateway that decides how the packets will be forwarded, 1 → routing flag#1, η σημασία του εξαρτάται από το πρωτόκολλο δρομολόγησης.

4.9) R2: **cli** → route.ntua.lab# **configure terminal** → router.ntua.lab(config)# **hostname R2** → R2(config)# **interface em1** → R2(config-if)# **ip address 192.168.2.1/24** → R2(config-if)# **exit** → R2(config)# **interface em0** → R2(config-if)# **ip address 172.17.17.2/30** → R2(config-if)# **exit** → R2(config)# **interface em2** → R2(config-if)# **ip address 172.17.17.9/30** → R2(config-if)# **exit**
em0: WAN1, em1: LAN2, em2: WAN3

4.10) R2(config)# **ip route 192.168.1.0/24 172.17.17.1**

4.11) R3: **cli** → route.ntua.lab# **configure terminal** → router.ntua.lab(config)# **hostname R3** → R3(config)# **interface em0** → R3(config-if)# **ip address 172.17.17.6/30** → R3(config-if)# **exit** → R3(config)# **interface em1** → R3(config-if)# **ip address 172.17.17.10/30** → R3(config-if)# **exit**

em0: WAN2, em1: WAN3

4.12) R3(config)# **ip route 192.168.1.0/24 172.17.17.5**

ip route 192.168.2.0/24 172.17.17.9

NOTE: πριν πετύχουν τα ip route έπρεπε να γίνει write file και reboot στα μηχανήματα, δεν ξέρουμε γιατί.

4.13) R3(config)# **do show ip forwarding**

Και δούλευε κανονικά, IP forwarding is ON.

4.14) PC1: **traceroute 192.168.2.2**

192.168.1.2 (PC1) → 192.168.1.1 (R1) → 172.17.17.2 (R2) → 192.168.2.2 (PC2)

Ακολουθούν τη διαδρομή LAN1 → WAN1 → LAN2, όπως και περιμέναμε.

Άσκηση 5: Σφάλμα καλωδίου και αυτόματη αλλαγή στη δρομολόγηση

5.1) R1(config)# **ip route 192.168.2.0/24 172.17.17.6 2**

5.2) Βάλαμε 2 στο distance έτσι ώστε να έχει χαμηλότερη προτεραιότητα από την επιλογή μέσω του WAN1, η οποία έχει by default distance 1 ως στατική εγγραφή.

5.3) R2(config)# **ip route 192.168.1.0/24 172.17.17.10 2**

5.4) Στον πίνακα δρομολόγησης βλέπουμε 2 εγγραφές για τα LAN1, LAN2 στα R1,R2.

R1(config)# **do show ip route**

R1:

C>* 192.168.1.0/24 is directly connected em0

S 192.168.2.0/24 [2/0] via 172.17.17.6, em2

S>* 192.168.2.0/24 [1/0] via 172.17.17.2, em1

R2(config)# **do show ip route**

R2:

C>* 192.168.2.0/24 is directly connected em1

S 192.168.1.0/24 [2/0] via 172.17.17.10, em2

S>* 192.168.1.0/24 [1/0] via 172.17.17.1, em0

5.5) S>* 192.168.2.0/24 [1/0] via 172.17.17.2, em1

Η διαδρομή που είναι ενεργοποιημένη είναι η S>* και φαίνεται από το σύμβολο «>*» το οποίο βλέπουμε απουσιάζει από την άλλη εγγραφή. Άρα μέσω του WAN1 που πάει στο R2.

5.6) Η διαχειριστική απόσταση φαίνεται μετά από το υποδίκτυο στις αγκύλες [] και δείχνει γράφει τον αριθμό της απόστασης [x/0] όπου x η απόσταση.

5.7) S>* 192.168.1.0/24 [1/0] via 172.17.17.1, em0

Άρα μέσω του WAN1, R1.

5.8) Πάμε στα R1, R2 και στα interfaces em1,em2 και em0,em2 αντίστοιχα

R1(config)# **interface em1** → R1(config-if)# **link-detect**

R2(config)# **interface em0** → R2(config-if)# **link-detect**

5.9) Θα πάμε στο **R1 → Settings → Network settings → Adapter 2 → Cable Connected off**

5.10) Πλέον είναι η διαδρομή μέσω του R3:

R1(config)# **do show ip route**

R1:

S>* 192.168.2.0/24 [2/0] via 172.17.17.6, em2

S 192.168.2.0/24 [1/0] via 172.17.17.2 inactive

5.11) Ναι. Γράφει inactive πλέον.

5.12) Ναι υπάρχει αλλαγή όπως φαίνεται παραπάνω.

5.13) R2(config)# **do show ip route**

R2:

S 192.168.1.0/24 [2/0] via 172.17.17.10, em2

S>* 192.168.1.0/24 [1/0] via 172.17.17.1, em0

Παρατηρούμε πως δεν έχει αλλάξει. Αυτό συμβαίνει επειδή το καλώδιο από τη μεριά του δεν έχει κοπεί και δεν έχουν σταλεί πακέτα προκειμένου να το ανιχνεύσει.

5.14) Θα πάμε στο **R2 → Settings → Network settings → Adapter 1 → Cable Connected off**

Ναι άλλαξε όπως αναμέναμε:

R2(config)# **do show ip route**

R2:

S>* 192.168.1.0/24 [2/0] via 172.17.17.10, em2

S 192.168.1.0/24 [1/0] via 172.17.17.1 inactive

5.15) PC1: **traceroute 192.168.2.2**

Ναι δείχνει διαδρομή 192.168.1.2 (PC1) → 192.168.1.1 (R1) → 172.17.17.6 (R3) → 172.17.17.9 (R2) → 192.168.2.2 (PC2), δηλαδή LAN1 → WAN2 → WAN3 → LAN2

5.16) PC2: **ssh lab@192.168.1.2**

Παρατηρούμε πως ακριβώς μετά που τα επανασυνδέσουμε κολλάει για λίγο, αλλά μετά είναι κανονικά συνδεδεμένο χωρίς κανένα πρόβλημα, άρα δεν επηρεάζεται η σύνδεση. Εκπληκτικό.

5.17) PC1: **traceroute 192.168.2.2**

Κάνουμε το παραπάνω traceroute και εξακριβώνουμε αυτό που περιμέναμε, πως πλέον η σύνδεση είναι μέσω του WAN1 και άλλαξε αυτόματα από μόνο του. 192.168.1.2 (PC1) → 192.168.1.1 (R1) → 172.17.17.2 (R2) → 192.168.2.2 (PC2)

Άσκηση 6: Διευθύνσεις διαχείρισης (loopback)

6.1) R1(config)# **interface lo0** → R1(config-if)# **ip address 172.22.22.1/32**
R2(config)# **interface lo0** → R2(config-if)# **ip address 172.22.22.2/32**
R3(config)# **interface lo0** → R3(config-if)# **ip address 172.22.22.3/32**

6.2) PC1: **ping 172.22.22.1(2,3)**
Παρατηρούμε πως επιτυγχάνει μόνο το ping στο loopback του R1 για το PC1 και του R2 για το PC2, και επιστρέφει no route to host για τα υπόλοιπα. Λογικό αφού δεν είναι συνδεδεμένα τα loopback, δεν υπάρχουν διαδρομές στα routing tables των R1, R2, R3 για τα υπόλοιπα loopbacks και δεν ξέρει που να στείλει για τα άλλα loopback.

6.3) R1(config)# **ip route 172.22.22.2/32 172.17.17.2**
R1(config)# **ip route 172.22.22.3/32 172.17.17.6**

6.4) R2(config)# **ip route 172.22.22.1/32 172.17.17.1**
R2(config)# **ip route 172.22.22.3/32 172.17.17.10**

6.5) R3(config)# **ip route 172.22.22.1/32 172.17.17.5**
R3(config)# **ip route 172.22.22.2/32 172.17.17.9**

6.6) PC1/2: **ping 172.22.22.1(2,3)**
Πετυχαίνουν όλα τα pings πλέον προς όλα τα loopbacks.

6.7) PC1: **tcpdump -i em0 -vvv -en icmp**
PC2: **tcpdump -i em0 -vvv -en icmp**
R3: **ping 192.168.1(2).2**
Τα ICMP πακέτα φτάνουν με διεύθυνση 172.17.17.6 στο PC1 και 172.17.17.10 στο PC2 τελικά, όπως περιμέναμε.

6.8) R3: **ping -S 172.22.22.3 -c 2 192.168.1.2**
Με αυτό το τρόπο επιβάλουμε τη πηγή, όχι απαραίτητα τη loopback, αλλά την πηγή από την οποία φαίνονται να στέλνονται.

6.9) Δεν θα μπορούσαν να επικοινωνήσουν με τα loopback καθόλου, αφού δεν έχουν εγγραφές στον πίνακα προώθησης τους. Επειδή τώρα έχουν default gateway, στέλνουν στο αντίστοιχο router και μετά εκείνο έχει εγγραφές για τα άλλα υποδίκτυα. Για να υπάρχει αμφίδρομη επικοινωνία θα επρεπε να προσθέταμε χειροκίνητα στη τοπολογία κάθε διαδρομή και, ανάλογα με το μέγεθος του δικτύου μας, θα αυξανόταν και η πιθανότητα να υπάρχουν λάθη.

6.10) Θα ήταν επιτυχή από το PC1 στα R1,R3 και από το PC2 στα R2,R3 γιατί θα προωθούσαν σε μη λειτουργική γραμμή το ping για το loopback από R1 σε R2 και αντίθετα, οπότε και θα αποτύχει.

6.11) R1(config) **ip route 172.22.22.2/32 172.17.17.6 2**

R1(config) **ip route 172.22.22.3/32 172.17.17.2 2**

6.12) R2(config) **ip route 172.22.22.1/32 172.17.17.10 2**

R2(config) **ip route 172.22.22.3/32 172.17.17.1 2**

6.13) R3(config) **ip route 172.22.22.1/32 172.17.17.9 2**

R3(config) **ip route 172.22.22.2/32 172.17.17.5 2**

6.14) R1(config)# **do show ip route**

S 172.22.22.2/32 [2/0] via 172.17.17.6, em2

S>* 172.22.22.2/32 [1/0] via 172.17.17.2, em1

Παρατηρούμε πως έχει επιλεχθεί η διαδρομή μέσω του WAN1 όπως είναι αναμενόμενο.

6.15) Παρατηρούμε πως άλλαξε όπως φαίνεται παρακάτω:

R1(config)# **do show ip route**

S>* 172.22.22.2/32 [2/0] via 172.17.17.6, em2

S 172.22.22.2/32 [1/0] via 172.17.17.2 inactive

Άλλαξε αυτόματα και δουλεύει κανονικά. Οι διαδρομές που ήταν μέσω WAN1 έγιναν inactive και τέθηκαν σε ισχύ οι δευτερεύουσες μέσω του WAN2.

6.16) Παρατηρούμε πως η βλάβη δεν έγινε αντιληπτή στα R1,R3 καθώς δεν είχαμε ενεργοποιημένο το link-detect στις διεπαφές στα άκρα της ζεύξης.

Άσκηση 7: Ένα εταιρικό δίκτυο

R1: em0 → LAN1, em1 → WAN1, em2 → WAN3

R2: em0 → LAN2, em1 → WAN2, em2 → WAN4

C1: em0 → CORE, em1 → WAN1, em2 → WAN2

C2: em0 → CORE, em1 → WAN3, em2 → WAN4

7.1) C1: **cli** → C1# **configure terminal** →

C1(configure)# **ip route 192.168.1.0/24 10.0.1.1**

C1(configure)# **ip route 192.168.1.0/24 10.0.0.2 2**

C1(configure)# **ip route 192.168.2.0/24 10.0.2.1**

C1(configure)# **ip route 192.168.2.0/24 10.0.0.2 2**

7.2) C2: **cli** → C2# **configure terminal** →

C2(configure)# **ip route 192.168.1.0/24 10.0.1.5**

C2(configure)# **ip route 192.168.1.0/24 10.0.0.1 2**

C2(configure)# **ip route 192.168.2.0/24 10.0.2.5**

C2(configure)# **ip route 192.168.2.0/24 10.0.0.1 2**

7.3) R1: **cli** → R1# **configure terminal** →

R1(configure)# **ip route 192.168.2.0/24 10.0.1.2**

R1(configure)# **ip route 192.168.2.0/24 10.0.1.6 2**

7.4) R2: **cli** → R2# **configure terminal** →

R2(configure)# **ip route 192.168.1.0/24 10.0.2.2**

R2(configure)# **ip route 192.168.1.0/24 10.0.2.6 2**

7.5) PC1: **ping -c 1 192.168.2.2**

Δούλεψε οπότε συνεχίζουμε.

7.6) Αποσυνδέουμε το em2 από το C1 και το em1 από το R2. Ακόμα επικοινωνούν μεταξύ τους χάρη στο link-detect.

7.7) Από το PC1 στο PC2 ακολουθούν στη διαδρομή LAN1 → WAN1 → CORE → WAN4 → LAN2 ή PC1 → R1 → C1 → C2 → R2 → PC2

Από το PC2 στο PC1 ακολουθούν τη διαδρομή LAN2 → WAN4 → WAN3 → LAN1

PC2 → R2 → C2 → R1 → PC1

7.8) PC1 → PC2: **traceroute 192.168.2.2**

PC1 (192.168.2.1) → R1 (192.168.1.1) → C1 (10.0.1.2) → C2 (10.0.1.6) → R2 (10.0.2.5) → PC2 (192.168.2.2)

PC2 → PC1: **traceroute 192.168.1.2**
PC2 (192.168.2.2) → R2 (192.168.2.1) → C2 (10.0.2.6) → R1 (10.0.1.1) → PC1 (192.168.1.2)

Οι δρομολογητές είναι οι σωστοί αλλά οι διεπαφές όχι. Ο τρόπος με τον οποίο λειτουργεί η traceroute είναι να στέλνει ping με αυξανόμενο TTL και να παίρνει απάντηση time to live exceeded και έτσι βρίσκει τις διεπαφές. Για αυτό βλέπουμε παραπάνω πως λάβαμε απάντηση από την 10.0.1.6 και όχι την 10.0.0.2 που περιμέναμε, επειδή ο C2 δρομολογεί για τον PC1 μέσω της WAN3 by default, και επειδή αυτή λειτουργεί το στέλνει μέσω αυτής. Αντίστοιχα από το PC2 στο PC1 είναι προς τα πίσω.

7.9) PC2: **ping -c 1 192.168.1.2**

Παρατηρούμε πως το TTL είναι 60 όταν φτάνει στο PC1, δηλαδή το PC1 απέχει 5 βήματα από το PC2, ενώ πριν είχαμε 4 για τη διαδρομή R2, C2, R1, PC1. Αυτό συμβαίνει επειδή το TTL μετριέται στη διαδρομή της επιστροφής, από το PC1 στο PC2 και συνεπώς πηγαίνει από τη διαδρομή PC1 → R1 → C1 → C2 → R2 → PC2.

7.10) Ναι τα PC επικοινωνούνε. Η επικοινωνία τους πλέον γίνεται αποκλειστικά μέσω του CORE υποδικτύου που είναι συνδεδεμένα τα C1, C2. Άρα η διαδρομή είναι PC1 ←→ R1 ←→C1 ←→ C2 ←→ R2 ←→ PC2 (δηλαδή είναι η ίδια διαδρομή πρακτικά από PC1 σε PC2 και αντίστροφα).

7.11) Δεν θα φτάσει ποτέ. Παρατηρούμε πως δεν λαμβάνουμε απάντηση. Το PC1 στέλνει στο R1, εκείνο θα στείλει στο C1, το οποίο θα δει το WAN2 πεσμένο και θα στείλει στο C2, το οποίο θα δει το WAN4 πεσμένο και θα στείλει στο C1 ή απλά δεν θα μπορέσει να το δρομολογήσει, οπότε απλά δεν θα λάβει απάντηση.

7.12) Ένα βασικό μειονέκτημα της σύνδεσης αυτής είναι η σπατάλη του C2, καθώς όσο το σύστημα λειτουργεί σωστά, το C2 δεν κάνει τίποτα και απλά υπάρχει, έτσι δεν είναι αποδοτικό. Επίσης, είναι εύκολο να αποκοπεί ένα υποδίκτυο αν κοπούν 2 γραμμές μονάχα.