Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 1
Εξοικείωση με το FreeBSD και το VirtualBox

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 08/03/2022

Άσκηση 1: Γνωριμία με το περιβάλλον εργασίας

1.1

Η διεύθυνση IPv4 του εικονικού VirtualBox Host-Only Ethernet Adapter είναι `192.168.56.1`.

1.2

Η μάσκα του τοπικού δικτύου είναι `255.255.255.0`.

1.3

Ναι, ο εξυπηρετητής DHCP είναι ενεργοποιημένος.

1.4

Η διεύθυνση IPv4 του εξυπηρετητή DHCP είναι `192.168.56.100` και η περιοχή διευθύνσεων που έχει διατεθεί για δυναμική παραχώρηση είναι `192.168.56.101-192.168.56.254`

1.5

Το prompt που εμφανίζεται για τον χρήστη `lab` είναι:

`lab@pc:~%`

1.6

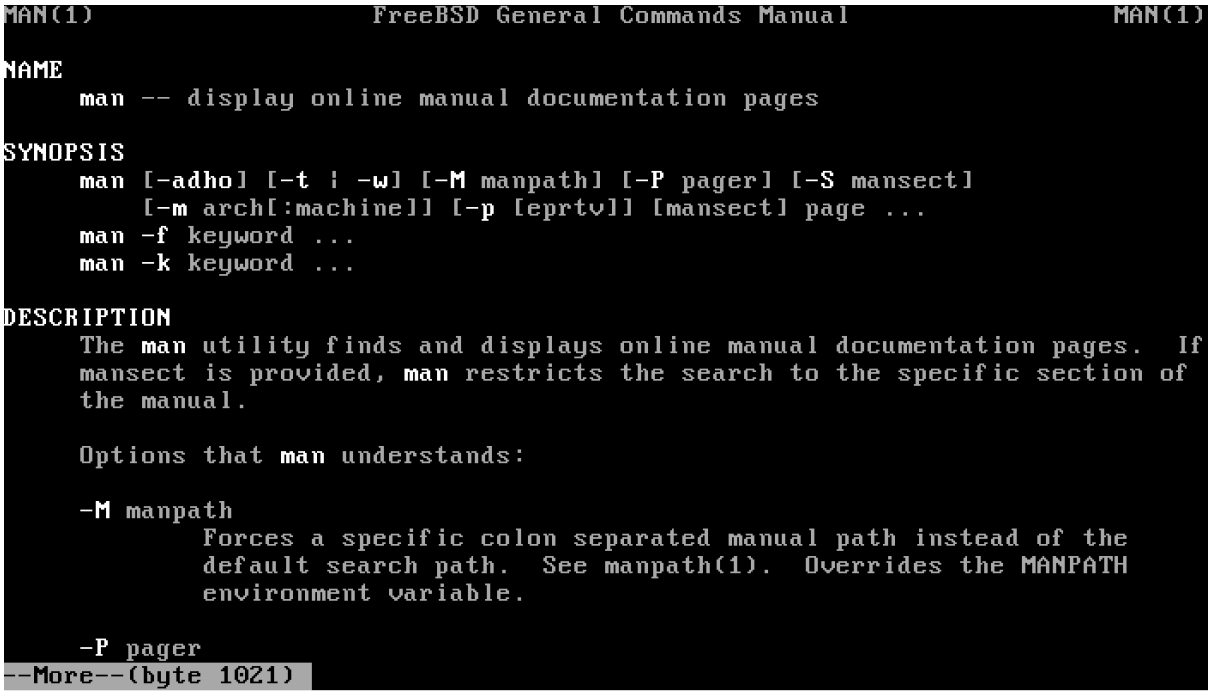
Το αποτέλεσμα της εντολής `man` είναι:

```
lab@pc:~ % man
What manual page do you want?
```

Αυτό συμβαίνει επειδή δεν έχουμε δώσει ως όρισμα στην man μια εντολή για να εμφανιστεί το κατάλληλο manpage.

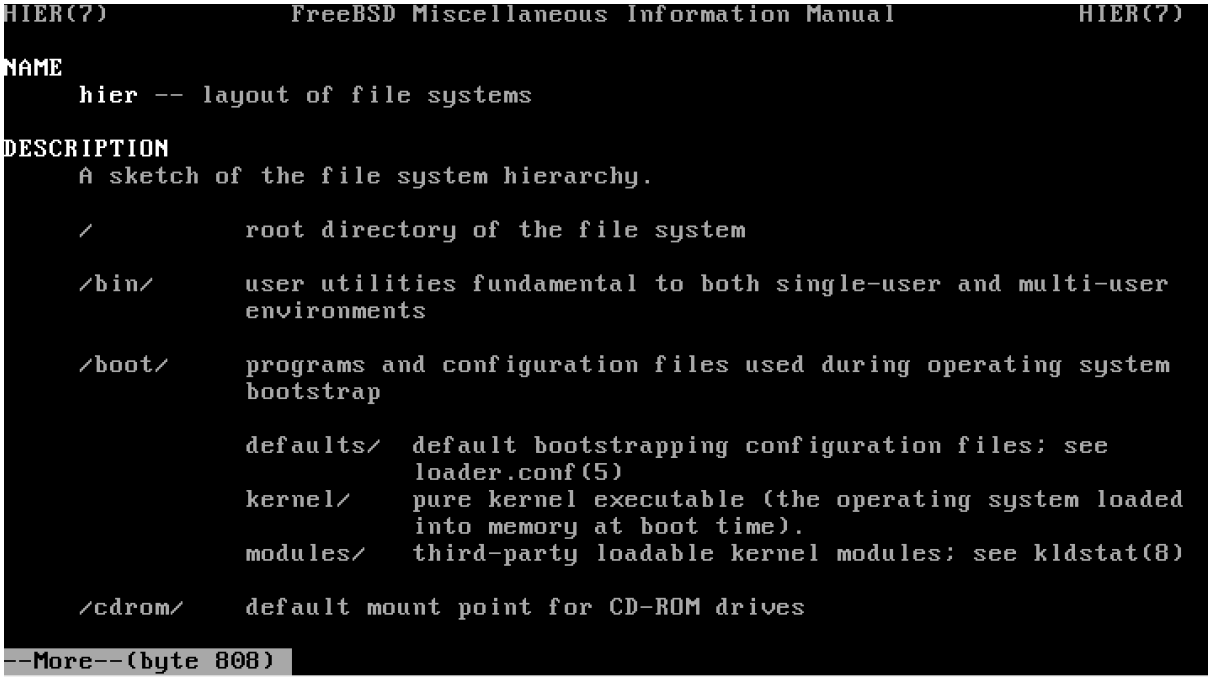
1.7

Το αποτέλεσμα της εντολής man man είναι το manual page της ίδιας της εντολής man, όπως φαίνεται στην εικόνα.



1.8

Η εντολή man hier (**hier**archy) μας εμφανίζει πληροφορίες για το layout του συστήματος αρχείων μας, όπως φαίνεται και στην εικόνα.



1.9

Σύμφωνα με το αποτέλεσμα της προηγούμενης εντολής, ο κατάλογος `/lib` περιέχει τις κρίσιμες βιβλιοθήκες που χρειάζονται για τα εκτελέσιμα στους καταλόγους `/bin` και `/sbin`.

1.10

Οι θυρίδες ηλεκτρονικού ταχυδρομείου των χρηστών βρίσκονται στον κατάλογο `/var/mail`.

1.11

Μπορούμε να μετακινηθούμε με τα πλήκτρα:

- Πάνω/κάτω βελάκι για μία γραμμή πάνω/κάτω αντίστοιχα
- `J/K` για μια γραμμή κάτω/πάνω αντίστοιχα
- `Space` για να κατέβουμε μία σελίδα κάτω
- `PgUp/PgDn` για να ανέβουμε/κατέβουμε μία σελίδα πάνω/κάτω αντίστοιχα.

Γενικά υπάρχουν πολλοί τρόποι. Άλλα πλήκτρα που μπορούμε να χρησιμοποιήσουμε για να μετακινηθούμε είναι `ENTER, e, d, b, w, y, u, f` και πολλά άλλα, καθώς και συνδυασμοί ορισμένων από αυτά με `CTRL` και `ESC`.

1.12

Για να αναζητήσουμε μια συγκεκριμένη λέξη-pattern, πατάμε `/` ακολουθούμενο από την λέξη που επιθυμούμε, δηλαδή `/pattern` και πατάμε `ENTER`. Μετά με `N` βρίσκουμε την επόμενη εμφάνιση της λέξης.

1.13

Το βασικό πλεονέκτημα της `less` σε σχέση με την `more` είναι ότι με την `less` μπορούμε να πάμε και πίσω στο αρχείο, όχι μόνο μπροστά.

1.14

Με την εντολή `hostname` βρίσκουμε ότι το όνομα του εικονικού μηχανήματος είναι `pc.ntua.lab`.

1.15

Με την εντολή `whoami` επιβεβαιώνουμε ότι το όνομα χρήστη με το οποίο έχουμε συνδεθεί είναι `lab`.

1.16

Με την εντολή `id` βρίσκουμε ότι ο αριθμός ταυτότητας του χρήστη `lab` είναι `1001`.

1.17

Πάλι από την εντολή `id` βλέπουμε ότι ο χρήστης `lab` ανήκει στις ομάδες `lab` και `wheel`.

1.18

Κάνουμε `cd ~` για να πάμε στον `home` φάκελο του χρήστη `lab`. Ύστερα γράφουμε `pwd` και βλέπουμε ότι ο φάκελος αυτός είναι ο `/usr/home/lab`.

1.19

Το prompt που εμφανίζεται για τον διαχειριστή `root` είναι:

```
root@pc:~#
```

1.20

Ο αριθμός ταυτότητας του `root` είναι 0 (εντολή `id`).

1.21

Ο διαχειριστής ανήκει στις ομάδες χρηστών `wheel` και `operator` (εντολή `id`).

1.22

Ο αριθμός ταυτότητας (gid) της ομάδας `wheel` είναι `0`.

1.23

Ο `home` φάκελος εργασίας του `root` είναι ο `/root`. (εντολή `pwd`).

1.24

Η διεύθυνση που αποδόθηκε στο εικονικό μας μηχάνημα από τον εξυπηρετητή DHCP είναι η `192.168.56.101`.

1.25

Με την εντολή `ifconfig` βρίσκουμε ότι το μηχάνημα διαθέτει τις διεπαφές `em0` και `lo0`.

1.26

Πάλι μέσω της εντολής `ifconfig` (πεδίο `ether`) βρίσκουμε ότι η διεύθυνση MAC της κάρτας δικτύου `em0` του εικονικού μηχανήματος είναι `08:00:27:e8:17:4a`.

1.27

Πάλι μέσω της εντολής `ifconfig` βρίσκουμε ότι η ταχύτητα της κάρτας δικτύου `em0` είναι `1 Gbps`.

1.28

Πάλι μέσω της εντολής `ifconfig` (πεδίο `inet`) βλέπουμε ότι η διεύθυνση IPv4 της διεπαφής που αντιστοιχεί στην κάρτα δικτύου `em0` είναι `192.168.56.101`.

1.29

Πάλι μέσω της εντολής `ifconfig` (πεδίο `netmask`), βρίσκουμε ότι η μάσκα υποδικτύου σε hex μορφή είναι `0xffffff00`, δηλαδή `255.255.255.0` σε δεκαδική μορφή.

1.30

Πάλι μέσω της εντολής `ifconfig` (πεδίο `mtu`), βρίσκουμε ότι η MTU είναι 1500 (bytes).

1.31

`lo0`:

- Διεύθυνση IPv4: `127.0.0.1`
- Μάσκα υποδικτύου: `255.0.0.0`
- MTU: 16384 (bytes)

1.32

Εκτελώντας `cat /etc/resolv.conf` παίρνουμε κενή έξοδο (το αρχείο δεν περιέχει τίποτα), άρα δεν έχουν οριστεί εξυπηρετητές DNS.

1.33

Αν κάνουμε από το φιλοξενούμενο μηχάνημα `ping 192.168.56.1`, τότε το φιλοξενούν μηχάνημα απαντάει.

1.34

Αν κάνουμε από το φιλοξενούν μηχάνημα `ping 192.168.56.101`, τότε το φιλοξενούμενο μηχάνημα απαντάει.

1.35

Η εντολή `ping` στέλνει για πάντα πακέτα, μέχρι να την διακόψουμε, ενώ η αντίστοιχη των Windows στέλνει μόνο 4 by default.

Άσκηση 2: Βασικές εντολές συστήματος αρχείων

2.1

Με την εντολή `pwd` βρίσκουμε ότι το home directory για τον χρήστη `lab` είναι `/usr/home/lab`.

2.2

Με την εντολή `mkdir tmp` δημιουργούμε έναν νέο φάκελο `tmp`.

2.3

Με την εντολή `mkdir tmp/el18175` δημιουργούμε έναν νέο φάκελο `el18175` κάτω από το `tmp`.

2.4

Με την εντολή `cd tmp/el18175` μετακινούμαστε στον φάκελο `el18175`.

2.5

Με την εντολή `find / -name hosts` βρίσκουμε ότι αρχείο με όνομα `hosts` βρίσκεται στις τοποθεσίες `/usr/share/examples/etc/hosts`, `/etc/bluetooth/hosts` και `/etc/hosts`.

2.6

Με την εντολή `cp /etc/hosts ~/tmp/el18175` αντιγράφουμε το αρχείο `/etc/hosts` στον φάκελο `~/tmp/el18175` που δημιουργήσαμε.

2.7

Με την εντολή `mv hosts hostsfile` αλλάζουμε το όνομα του αρχείου `hosts` σε `hostsfile`.

2.8

Με την εντολή `ls -l` βλέπουμε ότι το hostsfile έχει τα permission flags `-rw-r--r--`. Το πρώτο `r` σημαίνει ότι ο ιδιοκτήτης του αρχείου (user) έχει δικαίωμα ανάγνωσης στο συγκεκριμένο αρχείο, το `w` σημαίνει ότι ο ιδιοκτήτης του αρχείου έχει δικαίωμα εγγραφής, το δεύτερο `r` σημαίνει ότι οι χρήστες που ανήκουν στην ίδια ομάδα χρηστών (group) με τον ιδιοκτήτη έχουν δικαίωμα ανάγνωσης, και το τρίτο `r` σημαίνει ότι οι υπόλοιποι χρήστες (others) έχουν δικαίωμα ανάγνωσης.

2.9

Με την εντολή `touch test` δημιουργούμε ένα νέο άδειο αρχείο με όνομα `test`.

2.10

Με την εντολή `touch .hidden` δημιουργούμε ένα νέο κρυφό άδειο αρχείο με όνομα `.hidden` (είναι κρυφό λόγω της τελείας που βρίσκεται στην αρχή).

2.11

Με την εντολή `ls -l /etc/services` βρίσκουμε ότι το αρχείο `/etc/services` έχει μέγεθος 86674 (bytes).

2.12

Γενικά, η εντολή `df` εμφανίζει στατιστικά σχετικά με τον ελεύθερο χώρο της συσκευής. Η διαφορά των `df -H` και `df -h` είναι ότι η πρώτη εμφανίζει τα μεγέθη σε δυνάμεις του 1024, ενώ η δεύτερη σε δυνάμεις του 1000.

2.13

Με την εντολή `df -h ./` (βρισκόμαστε στον φάκελο `el18175`) βλέπουμε ότι έχουμε `6.5 Gigabyte` ελεύθερο χώρο, που είναι παραπάνω από αρκετός.

2.14

Με την εντολή `cp /etc/services ./` αντιγράφουμε το αρχείο `/etc/services` στον φάκελό μας.

2.15

Με την εντολή `gzip services` συμπιέζουμε το αρχείο `services`, και με την εντολή `ls -l` βρίσκουμε ότι το νέο μέγεθός του είναι `24577 bytes`.

2.16

Με την εντολή `ls -a` βλέπουμε τα περιεχόμενα του φακέλου μας συμπεριλαμβανομένων και των κρυφών αρχείων.

2.17

Με την εντολή `find /usr -user lab` βρίσκουμε όλα τα αρχεία του καταλόγου `/usr` που ανήκουν στον χρήστη `lab`. Είναι τα εξής:

```
/usr/home/lab
/usr/home/lab/.login
/usr/home/lab/.rhosts
/usr/home/lab/.mail_aliases
/usr/home/lab/.profile
/usr/home/lab/.cshrc
/usr/home/lab/.login_conf
/usr/home/lab/.shrc
/usr/home/lab/.mailrc
/usr/home/lab/.history
/usr/home/lab/.lesshst
/usr/home/lab/tmp
/usr/home/lab/tmp/el18175
/usr/home/lab/tmp/el18175/test
/usr/home/lab/tmp/el18175/hostsfile
/usr/home/lab/tmp/el18175/services.gz
```

2.18

Με την εντολή `rm ~/tmp/el18175/*` διαγράφουμε όλα τα αρχεία που περιέχει ο φάκελος με όνομα τον αριθμό μητρώου μας.

2.19

Με την εντολή `rm -r ~/tmp` διαγράφουμε τον φάκελο `tmp` που δημιουργήσαμε και ό,τι αυτός περιέχει.

Άσκηση 3: Επεξεργασία κειμένου, ανακατεύθυνση εντολών

3.1

Οι εντολές του `vi` που χρησιμοποιήσαμε είναι:

- `:%s /localhost/ntua-lab/g`
- `:q!`

3.2

Με την εντολή `ls -l /etc > filelist` δημιουργούμε ένα νέο αρχείο με όνομα `filelist` που περιέχει την έξοδο της εντολής `ls -l /etc`.

3.3

Με την εντολή `vi filelist` ανοίγουμε το αρχείο στον `vi` για επεξεργασία. Μέσα στον `vi`, εκτελούμε `dd` για να διαγράψουμε την πρώτη γραμμή του αρχείου, και ύστερα εκτελούμε `:wq` για να αποθηκεύσουμε τις αλλαγές και να επιστρέψουμε στην γραμμή εντολών. Ο `vi` μας ενημερώνει ότι το νέο αρχείο έχει `101` γραμμές και `5949` χαρακτήρες.

3.4

Η γραμμή που σβήσαμε δείχνει πόσο χώρο (μετρημένο σε filesystem blocks) πιάνουν τα αρχεία του φακέλου στον οποίο εκτελέσαμε `ls -l`.

3.5

Μπορούμε να εκτελέσουμε την εντολή `wc filelist`, η οποία μας επιστρέφει ότι το αρχείο `filelist` περιέχει `101` γραμμές, `917` λέξεις, και `5949` χαρακτήρες.

3.6

Μπορούμε να εκτελέσουμε `ls -l /etc | wc`, που θα μας επιστρέψει `102` γραμμές, οπότε αν αφαιρέσουμε μία γραμμή (λόγω του `total ...`) βρίσκουμε το ίδιο αποτέλεσμα με πριν.

3.7

Με την εντολή `ls -l /etc | grep "rc" | wc` βρίσκουμε ότι `14` αρχεία του καταλόγου `/etc` περιέχουν το κείμενο `"rc"` στο όνομά τους.

Άσκηση 4: Βασικές πληροφορίες συστήματος

4.1

Με την εντολή `grep -i cpu /var/run/dmesg.boot` παίρνουμε τις εξής πληροφορίες για τον επεξεργαστή:

```
CPU: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz (1991.76-MHz 686-class CPU)
```

4.2

Με την εντολή `grep -i memory /var/run/dmesg.boot` παίρνουμε τις εξής πληροφορίες για την μνήμη:

```
real memory  = 268369920 (255 MB)
avail memory = 239337472 (228 MB)
```

4.3

Με την εντολή `uname -v` παίρνουμε τις εξής πληροφορίες για τη έκδοση του λειτουργικού συστήματος:

```
FreeBSD 10.0-RELEASE #0 r260789: Fri Jan 17 01:46:25 UTC 2014
root@snap.freebsd.org:/usr/obj/usr/src/sys/GENERIC
```

4.4

Με την εντολή `service -e | wc` βρίσκουμε ότι υπάρχουν 16 ενεργοποιημένες υπηρεσίες.

4.5

Μπορούμε να δούμε τη λίστα όλων των διεργασιών που τρέχουν στο σύστημα με την εντολή `ps -aux`.

4.6

Με οποιαδήποτε από τις εντολές:

- `ps -aux | grep syslogd`
- `system -e | grep syslogd`
- `top | grep syslogd`

μπορούμε να δούμε αν τρέχει η υπηρεσία `syslogd`.

4.7

Με την εντολή `sockstat -4` μπορούμε να βρούμε τις υπηρεσίες που αναμένουν κίνηση `IPv4`, και τις αντίστοιχες θύρες `TCP` ή `UDP` όπου την περιμένουν.

4.8

Με την εντολή `top` βλέπουμε τις πιο κοστοβόρες διεργασίες. Αν θέλουμε να δούμε για κάποια συγκεκριμένη διεργασία, εκτελούμε `top | grep <process name>`.

4.9

Με την εντολή `iostat -d ada0 -w 1` μπορούμε να δούμε τη δραστηριότητα του δίσκου `ada0` ανά δευτερόλεπτο.

4.10

Με την εντολή `vmstat -w 2` μπορούμε να δούμε τη δραστηριότητα της μνήμης (μέση και ελεύθερη) ανά δύο δευτερόλεπτα.

Άσκηση 5: Πρόσβαση ως root

5.1

Η πρώτη προσπάθεια απέτυχε, διότι το μηχάνημα δεν επιτρέπει απομακρυσμένη σύνδεση ως `root` για λόγους ασφαλείας.

5.2

Ως χρήστης (lab), δεν μπορούμε να αλλάξουμε το όνομα του εικονικού μηχανήματος σε `virtualmachine`, διότι αυτή η εντολή απαιτεί δικαιώματα διαχειριστή (root). Η εντολή που προσπαθήσαμε να εκτελέσουμε είναι `hostname virtualmachine`.

5.3

Ο DHCP Server έχει IPv4 διεύθυνση `192.168.56.100`. Εκτελούμε λοιπόν:
`ping -c 5 -i 2 192.168.56.100`.

5.4

Η κλήση με διάστημα ενδιάμεσης παύσης `0.1 sec` θα αποτύχει, γιατί το διάστημα είναι πολύ μικρό, μόνο ο διαχειριστής μπορεί να ορίσει τόσο μικρό διάστημα.

5.5

Για να πετύχουν οι εντολές των ερωτημάτων 5.2 και 5.4, πρέπει να εκτελέσουμε `su` και ύστερα να εισάγουμε τον κωδικό `ntua`. Έτσι θα είμαστε πλέον σε mode διαχειριστή (root) και οι παραπάνω εντολές θα πετύχουν.
Αν θέλουμε να πετυχαίνει **και** η εντολή του 5.1, θα πρέπει να αλλάξουμε τις ρυθμίσεις `ssh`:

```
vi /etc/ssh/sshd_config

replace: #PermitRootLogin no
with: PermitRootLogin yes

/etc/rc.d/sshd restart
```

Πλέον μπορούμε να συνδεθούμε και απομακρυσμένα ως root κατευθείαν (χωρίς να χρειάζεται `su`).

5.6

Με τις εντολές `w` ή `who` μπορούμε να δούμε ποιοι χρήστες είναι συνδεδεμένοι στο σύστημα.

5.7

Αν κάποιος κοινός χρήστης λάβει δικαιώματα διαχειριστή, εμφανίζεται στην κονσόλα του εικονικού μηχανήματος προειδοποιητικό μήνυμα.

5.8

Με την εντολή `cat /var/log/auth.log` βλέπουμε όλα τα μηνύματα σχετικά με authentication/login κλπ. Το μήνυμα που είδαμε πριν στην κονσόλα του εικονικού μηχανήματος εμφανίζεται (μεταξύ άλλων σχετικών μηνυμάτων).

5.9

Με την εντολή `su -l lab` προσομοιώνουμε ένα πλήρες login (με κατάλληλη αλλαγή των μεταβλητών περιβάλλοντος) ως `lab`. Δεν χρειάζεται κωδικός αφού εκτελούμε την εντολή αυτή σε ρόλο διαχειριστή.

Άσκηση 6: Μεταφορά αρχείων

Αρχικά συνδεόμαστε μέσω `sftp` στο εικονικό μηχάνημα με την εντολή `sftp lab@192.168.56.101`. Όλες οι παρακάτω εντολές εκτελούνται μέσω `sftp`.

6.1

Αντιγράφουμε τα περιεχόμενα του φακέλου `/usr/home/lab` σε φάκελο `~/Downloads/tmp`, με τις εντολές:

```
Initial local directory: ~ (/home/nick)
Initial remote directory: /usr/home/lab

lcd Downloads (Local directory is now ~/Downloads)
!mkdir tmp (Directory ~/Downloads/tmp now exists)
lcd tmp (Local directory is now ~/Downloads/tmp)
get * (remote: /usr/home/lab/* --> local: ~/Downloads/tmp)
```

6.2

Βρισκόμαστε στα ίδια local/remote directories με πριν και εκτελούμε:

```
get /etc/hosts
get /etc/rc.conf
```

6.3

Με την εντολή `mkdir tmp` φτιάχνουμε έναν φάκελο `tmp` κάτω από τον φάκελο του χρήστη `lab`.

6.4

Με την εντολή `put *` αντιγράφουμε τα περιεχόμενα του φακέλου `tmp` του υπολογιστή μας στον φάκελο `tmp` του εικονικού μηχανήματος.

6.5

Με την εντολή `rm *` διαγράφουμε όλα τα περιεχόμενα του φακέλου `tmp` στο εικονικό μηχάνημα.

6.6

Βρισκόμαστε στον φάκελο `/usr/home/lab` και εκτελούμε `rmdir tmp` για να διαγράψουμε τον φάκελο `tmp` από το εικονικό μηχάνημα.

6.7

Με local directory `~/Downloads/tmp` εκτελούμε `get -r /etc/`.

6.8

Η μεταφορά αυτή δεν ολοκληρώνεται γιατί δεν έχουμε δικαίωμα πρόσβασης σε κάποια αρχεία.

6.9

Εκτελούμε `put -r etc`.

6.10

Από το τερματικό του εικονικού μηχανήματος εκτελούμε `mv etc tmp`.

6.11

Ναι, μπορούμε να διαγράψουμε τα περιεχόμενα του φακέλου `tmp`.

6.12

Ναι, μπορούμε να διαγράψουμε τον φάκελο `tmp`.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 2
Δικτύωση συστημάτων στο VirtualBox

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 08/03/2022

Άσκηση 1 (προετοιμασία): Δημιουργία εικονικού FreeBSD

1.1

Επισκεπτόμαστε την ιστοσελίδα της εκφώνησης και εντοπίζουμε τα VM images για επεξεργαστές `i386`.

1.2

Κατεβάζουμε το αντίστοιχο αρχείο και το αποσυμπιέζουμε.

1.3

Δημιουργούμε ένα νέο εικονικό μηχάνημα στο VirtualBox με τα ζητούμενα χαρακτηριστικά.

1.4

Από το γραφικό περιβάλλον του VirtualBox αλλάζουμε τις ζητούμενες παραμέτρους για μνήμη γραφικών, USB και Network.

1.5

Ξεκινάμε το εικονικό μηχάνημα και κάνουμε login ως root.

1.6

Με τη βοήθεια της εντολής `passwd` ορίζουμε συνθηματικό `ntua` για τον root.

1.7

Με τη βοήθεια της εντολής `adduser` δημιουργούμε έναν νέο χρήστη `lab` που έχει τα ζητούμενα χαρακτηριστικά.

1.8

Στο αρχείο `/etc/rc.conf` προσθέτουμε τις ζητούμενες γραμμές, διαγράφοντας ό,τι υπάρχει.

1.9

Δημιουργούμε το αρχείο `/boot/loader.conf`, το οποίο περιέχει τη ζητούμενη εντολή για επιτάχυνση της διαδικασίας εκκίνησης.

1.10

Επανεκκινούμε το FreeBSD χωρίς προβλήματα και με την εντολή `service -e | grep sshd` βεβαιωνόμαστε ότι η υπηρεσία `sshd` τρέχει.

1.11

Διαγράφουμε το ιστορικό των εντολών που δώσαμε μέχρι το σημείο αυτό με την εντολή `history -c`.

1.12

Διαγράφουμε όποιο αρχείο `/var/db/dhclient.leases.*` έχει δημιουργηθεί κατά την εκκίνηση με την εντολή `rm /var/db/dhclient.leases.*`.

1.13

Κλείνουμε το εικονικό μηχάνημα με την εντολή `poweroff`.

1.14

Από τη διαδρομή File→Export Appliance... δημιουργούμε ένα αρχείο `FreeBSD11.4.ova`.

1.15

Αποθηκεύουμε το αρχείο `.ova` για μελλοντική χρήση και διαγράφουμε το αρχείο `.vhd`, αφού δεν χρειάζεται πλέον.

Άσκηση 2: Ανάλυση δικτυακών πρωτοκόλλων με το TCPDUMP

2.1

Με την εντολή `ifconfig`.

2.2

Με τις εντολές `ifconfig em0 down` και ύστερα `ifconfig em0 up`.

2.3

Με τις εντολές `man tcpdump`, `man pcap` και `man pcap-filter`.

2.4

Η σύνταξη είναι: `tcpdump -i em0 -n`.

2.5

Η σύνταξη είναι: `tcpdump -i em0 -X`.

2.6

Η σύνταξη είναι: `tcpdump -e`.

2.7

Η σύνταξη είναι: `tcpdump -i em0 -s 68`.

2.8

Η σύνταξη είναι:
`tcpdump -v "ip && host 10.0.0.1"` ή `tcpdump -v "ip && (src or dst 10.0.0.1)"`.

2.9

Η σύνταξη είναι:
`tcpdump -i em0 "(src 10.0.0.1 && dst 10.0.0.2) || (src 10.0.0.2 && dst 10.0.0.1)"`.

2.10

Η σύνταξη είναι: `tcpdump -x "ip && net 1.1.0.0/16"`.

2.11

Η σύνταξη είναι: `tcpdump -e -x "ip && (!net 192.168.1.0/24)"`.

2.12

Η σύνταξη είναι: `tcpdump "ip && broadcast"`.

2.13

Η σύνταξη είναι: `tcpdump "ip[2:2] > 576"`.

2.14

Η σύνταξη είναι: `tcpdump "ip[8:1] < 5"`.

2.15

Η σύνταξη είναι: `tcpdump "ip[0:1] & 0x0f > 5`.

2.16

Η σύνταξη είναι: `tcpdump "icmp && (src 10.0.0.1)"`.

2.17

Η σύνταξη είναι: `tcpdump "tcp && (dst 10.0.0.2)"`.

2.18

Η σύνταξη είναι: `tcpdump "udp && (dst port 53)"`.

2.19

Η σύνταξη είναι: `tcpdump "tcp && (src or dst 10.0.0.10)"`.

2.20

Η τροποποιημένη σύνταξη είναι:
`tcpdump -w sample_capture "tcp && (src or dst 10.0.0.10) && (dst port 23)"`.

2.21

Η σύνταξη είναι: `tcpdump "tcp[13:1] & 0xff == 2"`.

2.22

Η σύνταξη είναι: `tcpdump "tcp[13:1] & 0xff == 2 || tcp[13:1] & 0xff == 18"`.

2.23

Η σύνταξη είναι: `tcpdump "tcp[13:1] & 0xff == 17"`.

2.24

Η παράσταση αυτή απομονώνει την τιμή του Data Offset, και λόγω τις δεξιάς ολίσθησης κατά 2 θέσεις μας δίνει τελικά το μέγεθος του TCP Header μετρημένο σε bytes.

2.25

Η σύνταξη είναι: `tcpdump "((tcp[12:1] & 0xf0) >> 2) > 20"`.

2.26

Η σύνταξη είναι: `tcpdump -A "port 80"`.

2.27

Η σύνταξη είναι: `tcpdump "dst edu-dy.cn.ntua.gr && dst port 23"`.

2.28

Η σύνταξη είναι: `tcpdump "ip6"`.

Άσκηση 3: Δικτύωση Host-only

3.1

Από τη διαδρομή: "File→Host Network Manager→Adapter" βρίσκουμε ότι η IPv4 διεύθυνση του Host-only Ethernet adapter είναι `192.168.56.1`.

3.2

Από τη διαδρομή "File→Host Network Manager→DHCP Server" βρίσκουμε ότι η IPv4 διεύθυνση του DHCP Server είναι `192.168.56.100`, ενώ η περιοχή διευθύνσεων που μπορεί να εκχωρήσει είναι `192.168.56.101-192.168.56.254`.

3.3

Με την εντολή `dhclient em0` αποδίδουμε διευθύνσεις μέσω DHCP στα εικονικά μηχανήματα.

3.4

Στο PC1 αποδόθηκε η διεύθυνση `192.168.56.102`, ενώ στο PC2 αποδόθηκε η διεύθυνση `192.168.56.103`.

3.5

Θα εκτελέσουμε `ping 192.168.56.102` από το PC2 ή `ping 192.168.56.103` από το PC1.

Εναλλακτικά για να έχουμε καλύτερη εποπτεία της επικοινωνίας θα μπορούσαμε να τρέξουμε και `nc -l <port>` από το ένα μηχάνημα (πχ το PC1), και από το άλλο να τρέξουμε `nc <PC1 ip> <port>`, όπου `port > 1024`. Ύστερα γράφουμε ένα μήνυμα (πχ hello) και πατάμε <enter> σε οποιαδήποτε από τις δύο κονσόλες. Το μήνυμα θα πρέπει να μεταφέρεται αυτούσιο στην κονσόλα του άλλου vm.

3.6

Θα εκτελέσουμε `ping 192.168.56.102` ή `ping 192.168.56.103` από το φιλοξενούν μηχάνημα.

Εναλλακτικά μπορούμε να ακολουθήσουμε την ίδια διαδικασία με το `nc` που περιγράψαμε παραπάνω.

3.7

Η σύνταξη της εντολής που δείχνει την προεπιλεγμένη πύλη είναι `netstat -rn`.

3.8

Όχι, δεν υπάρχει προεπιλεγμένη πύλη, αφού έχουμε Host-only networking, οπότε δεν υπάρχει επικοινωνία με εξωτερικά δίκτυα, αλλά μόνο με μηχανήματα που υπάρχουν στο εσωτερικό δίκτυο (συμπεριλαμβανομένου του host).

3.9

Όχι, δεν μπορούμε να κάνουμε ping στην φυσική κάρτα δικτύου του φιλοξενούντος μηχανήματος, αφού έχουμε λειτουργία Host-only networking και η φυσική κάρτα δικτύου θεωρείται ότι ανήκει σε εξωτερικό δίκτυο.

3.10

Το όνομα των μηχανημάτων όπως το αντιλαμβάνεται το λειτουργικό τους σύστημα είναι `PC.ntua.lab`, το οποίο βρέθηκε με την εντολή `hostname`.

3.11

Με την εντολή `hostname PC1` και την εντολή `hostname PC2` μετονομάζουμε τα εικονικά μηχανήματα σε `PC1` και `PC2` αντίστοιχα.

3.12

Με το που κάνουμε logout εμφανίζεται το εξής μήνυμα στον PC1 και τον PC2 αντίστοιχα, χωρίς εμείς να εκτελέσουμε κάποια επιπλέον εντολή:

Σε κάθε μηχάνημα, το νέο όνομα που αποδώσαμε φαίνεται στην παρένθεση.
Αφού κάνουμε login, αυτό φαίνεται και στο prompt:

```
root@PC1:~ #

root@PC2:~ #
```

3.13

Το `/etc/rc.conf` στο PC1 δεν περιέχει το νέο όνομα, οπότε σε περίπτωση επανεκκίνησής του το όνομα θα γίνει `PC.ntua.lab`.

3.14

Με `vi /etc/rc.conf` αντικαθιστούμε το `PC.ntua.lab` με `PC1`, `PC2` αντίστοιχα.

3.15

Πρέπει να προσθέσουμε στο `/etc/hosts` τις γραμμές:

```
192.168.56.102 PC1
192.168.56.103 PC2
```

3.16

Ένα σύνηθες παράδειγμα είναι η χρήση του `localhost`, πχ `ping localhost`.

3.17

Δύο τρόποι είναι:

- `tcpdump -l "icmp && host PC1" | tee test`
- `tcpdump -l "icpm && host PC1" > test & tail -f test`

3.18

Το μήκος των μηνυμάτων ICMP είναι 64 bytes και το TTL τους είναι 64.

3.19

Η τιμή του πεδίου TTL της απάντησης είναι 64.

3.20

Η σύνταξη της εντολής `tcpdump` που χρησιμοποιήσαμε είναι: `tcpdump -v "icmp"`. Θα μπορούσαμε να βάλουμε παραπάνω verbosity (πχ. `tcpdump -vv "icmp"`), αλλά δεν θα έκανε διαφορά στην συγκεκριμένη περίπτωση.

3.21

Το μήκος των μηνυμάτων ICMP που παράγει το φιλοξενούν μηχάνημα είναι 64 bytes, όπως και πριν.

3.22

Η τιμή του TTL είναι 64 και συμφωνεί με την τιμή που βρήκαμε προηγουμένως.

3.23

Όχι, δεν παρατηρήθηκε κάποια κίνηση σχετική με το ping, αφού το PC1 καταγράφει κίνηση που αφορά μόνο την κάρτα δικτύου του.

3.24

Αυτή τη φορά παρατηρούμε την σχετική με το ping κίνηση, αφού με το Allow VMs καταγράφεται στο PC1 και κίνηση που αφορά και τις άλλες εικονικές μηχανές που βρίσκονται στο ίδιο δίκτυο.

Άσκηση 4: Δικτύωση Internal

4.1

Η σύνταξη της εντολής που χρησιμοποιήσαμε για να ορίσουμε τις στατικές διευθύνσεις είναι: `ifconfig em0 192.168.56.102/24` (για το PC1) και `ifconfig em0 192.168.56.103/24` (για το PC2).

4.2

Το μήνυμα λάθους που εμφανίστηκε όταν ορίσαμε στατικές διευθύνσεις οφείλεται στο ότι ακόμα λειτουργεί ο dhcp client. Αν ξαναεκτελέσουμε την παραπάνω εντολή, δεν θα εμφανιστεί το μήνυμα λάθους διότι έχει πλέον απενεργοποιηθεί η υπηρεσία dhcp για το συγκεκριμένο μηχάνημα.

4.3

Με `tcpdump -v` στο PC1 ξεκινάμε τη ζητούμενη καταγραφή.

4.4

Όχι, δεν μπορούμε να κάνουμε `ping 192.168.56.103` (δηλαδή στο PC2).

4.5

Παρατηρούνται μόνο ARP Requests από το φιλοξενούν μηχάνημα, το οποίο θέλει να μάθει την φυσική διεύθυνση του PC2.

4.6

Όχι, δεν μπορούμε να κάνουμε `ping PC1`.

4.7

Όχι, δεν παρατηρούμε κίνηση σχετική με το ping προς το PC1.

4.8

Ναι, πλέον τα δύο εικονικά μηχανήματα επικοινωνούν, αφού βρίσκονται στο ίδιο εσωτερικό δίκτυο.

4.9

Όχι, δεν μπορούμε να επικοινωνήσουμε με κανένα από τα δύο εικονικά μηχανήματα, αφού στο mode ”Internal Network” δεν συμμετέχει καθόλου ο host.

4.10

Με την εντολή `tcpdump -n` ξεκινάμε μία νέα καταγραφή στο PC1, χωρίς επίλυση διευθύνσεων IPv4 σε ονόματα.

4.11

Με την εντολή `arp -d -a` αδειάζουμε τον πίνακα arp του PC2, και ύστερα κάνουμε ping από το PC2 προς την εικονική κάρτα δικτύου του host. Παρατηρούμε στην καταγραφή πλαίσια ARP Request που θέλουν να μάθουν την φυσική διεύθυνση της εικονικής κάρτας του host.

4.12

Επειδή τώρα ο host δεν συμμετέχει στο δίκτυο των εικονικών μηχανών, δεν απαντάει κανείς στο ping, εξού και το μήνυμα `host is down`, αφού θεωρείται ότι ο host είναι απενεργοποιημένος.

4.13

Με την εντολή `ifconfig em0 10.11.12.61/26` δίνουμε την προτελευταία διαθέσιμη διεύθυνση του υποδικτύου στο PC1, ενώ με την εντολή `ifconfig em0 10.11.12.62/26` δίνουμε την τελευταία διαθέσιμη διεύθυνση του υποδικτύου στο PC2. Η διεύθυνση `10.11.12.63/26` είναι η διεύθυνση broadcast, και άρα δεν είναι διαθέσιμη.

4.14

Ναι, τα δύο μηχανήματα επικοινωνούν μεταξύ τους με τις νέες διευθύνσεις που ορίσαμε προηγουμένως.

Άσκηση 5: Δικτύωση NAT

5.1

Με την εντολή `dhclient em0` σε καθένα από τα PC{1,2,3}, αποδίδουμε IPv4 διεύθυνση στην διεπαφή `em0` των εικονικών μηχανημάτων, με DHCP.

5.2

Και τα 3 μηχανήματα έχουν λάβει τη διεύθυνση IPv4 `10.0.2.15`, η οποία αποδόθηκε από τη διεύθυνση `10.0.2.2`, δηλαδή από τον host, ο οποίος λειτουργεί και ως DHCP Server.

5.3

Με την εντολή `netstat -rn` βλέπουμε ότι η προεπιλεγμένη πύλη είναι η διεύθυνση `10.0.2.2`, δηλαδή ο host.

5.4

Με `cat /etc/resolv.conf` βλέπουμε ότι το περιεχόμενο του αρχείου είναι:

```
# Generated by resolvconf
nameserver 10.0.2.3
```

5.5

Οι πληροφορίες αυτές έχουν καταγραφεί στο `/var/db/dhclient.leases.em0`.

5.6

Ναι, μπορούμε να κάνουμε `ping 10.0.2.2`.

5.7

Ναι, το νέο μηχάνημα επικοινωνεί με το Internet. Αν κάνουμε πχ `ping google.com`, βλέπουμε ότι στέλνονται επιτυχώς πακέτα. Αυτό συμβαίνει διότι στο NAT mode γίνεται μετάφραση των διευθύνσεων IP ώστε η προέλευση των πακέτων να φαίνεται ότι είναι η προκαθορισμένη πύλη (`10.0.2.2`).

5.8

Αν κάνουμε ping στις 4 αυτές διευθύνσεις, λαμβάνουμε απάντηση από όλες εκτός από την `10.0.2.1`. Οι διευθύνσεις αυτές παριστάνουν:

- `10.0.2.1` → Τίποτα
- `10.0.2.2` → Host/Default Gateway/DHCP Server
- `10.0.2.3` → Proxy DNS server
- `10.0.2.4` → TFTP Server για εκκίνηση του φιλοξενούμενου μηχανήματος από το δίκτυο

5.9

Όχι, δεν επικοινωνεί, διότι κάθε εικονικό μηχάνημα βρίσκεται σε δικό του δίκτυο, το οποίο έχει προκαθορισμένη πύλη τον host.

5.10

Η σημασία των παραμέτρων στην εντολή `traceroute` είναι:

- `-I` → χρήση πακέτων ICMP για τις ανάγκες τις εντολής
- `-n` → μη αντιστοίχιση των διευθύνσεων IP σε ονόματα
- `-q 1` → χρήση ενός πακέτου σε κάθε διαδοχική προσπάθεια της εντολής (αντί για 3 που είναι το default)
- `1.1.1.1` → η διεύθυνση-στόχος

5.11

Σύμφωνα με την καταγραφή του tcpdump, η διεύθυνση πηγής των μηνυμάτων ICMP που παράγει η `traceroute` είναι `10.0.2.15`, δηλαδή η διεύθυνση της διεπαφής του guest, και ο τύπος τους είναι `ICMP Echo Request`.

5.12

Σύμφωνα με την καταγραφή του Wireshark, η διεύθυνση πηγής των αντίστοιχων μηνυμάτων ICMP είναι `192.168.1.42`, δηλαδή η διεύθυνση της διεπαφής του host.

5.13

Οι διευθύνσεις πηγής των μηνυμάτων ICMP τύπου "TTL exceeded in transit" στην καταγραφή του Wireshark είναι `192.168.1.1` και `62.38.0.170`.

5.14

Η διεύθυνση προορισμού των μηνυμάτων αυτών είναι η `192.168.1.42`.

5.15

Σύμφωνα με την καταγραφή του tcpdump, οι διευθύνσεις IPv4 πηγής των μηνυμάτων ICMP τύπου "TTL exceeded in transit" είναι `10.0.2.2`, `192.168.1.1` και `62.38.0.170`.

5.16

Η διεύθυνση προορισμού των μηνυμάτων αυτών είναι η `10.0.2.15` .

5.17

Τα 2 από τα 3 μηνύματα "TTL exceeded in transit" της καταγραφής του tcpdump έχουν αντιστοίχιση σε αυτά της καταγραφής του Wireshark. Το πρώτο από τα τρία μηνύματα (αυτό με διεύθυνση πηγής `10.0.2.2`) όμως, εμφανίζεται μόνο στην καταγραφή του tcpdump, αφού γίνεται αντιληπτό μόνο από το φιλοξενούμενο μηχάνημα (επειδή μηδενίζεται το TTL στην προκαθορισμένη πύλη `10.0.2.2`).

5.18

Εκτελούμε την εντολή `traceroute -n 1.1.1.1` (η αντίστοιχη της `tracert -d 1.1.1.1` σε Linux) από το φιλοξενούν μηχάνημα. Το πλήθος των hops που προκύπτει είναι κατά 1 μικρότερο από αυτό που εμφάνισε η traceroute στο εικονικό μηχάνημα (7 αντί για 8), επειδή στο εικονικό μηχάνημα γίνεται και το επιπλέον hop guest $\rightarrow$ host.

Άσκηση 6: Δικτύωση NAT Network

6.1

Η διεύθυνση του δικτύου NAT που έχει οριστεί στο VirtualBox είναι `10.0.2.0/24`.

6.2

Με `ifconfig em0 delete` στα μηχανήματα PC{1,2} διαγράφουμε τη διεύθυνση IPv4 από την κάρτα δικτύου, ενώ διαγράφουμε (με `rm`) και το αρχείο `/var/db/dhclient.leases.em0`.

6.3

Με την εντολή `dhclient em0` δίνουμε μέσω DHCP διευθύνσεις IPv4 στα εικονικά μηχανήματα PC1, PC2.

6.4

Στο PC1 αποδόθηκε η διεύθυνση `10.0.2.15` (ίδια με πριν), ενώ στο PC2 αποδόθηκε η διεύθυνση `10.0.2.4` (διαφορετική απ' ό,τι πριν).

6.5

Η διεύθυνση του DHCP Server είναι `10.0.2.3`.

6.6

Κάνοντας `cat /etc/resolv.conf` βλέπουμε ότι το περιεχόμενο του αρχείου είναι:

```
# Generated by resolvconf
search home
nameserver 10.0.2.1
```

6.7

Κάνοντας `netstat -r` βλέπουμε ότι η προεπιλεγμένη πύλη στον πίνακα δρομολόγησης είναι η `10.0.2.1`

6.8

Ναι, μπορούμε να κάνουμε `ping` στην IPv4 διεύθυνση της προεπιλεγμένης πύλης από τα PC{1,2}.

6.9

Ναι, μπορούμε να κάνουμε `ping` στην IPv4 διεύθυνση του εξυπηρετητή DHCP και από τα δύο μηχανήματα.

6.10

Από τα PC{1,2} μπορούμε να κάνουμε `ping` στη διεύθυνση `10.0.2.2`, και απαντά το φιλοξενούν μηχάνημα, όπως φαίνεται και από τον πίνακα `arp`.

6.11

Ναι, τα δύο μηνύματα επικοινωνούν, γιατί αν πχ κάνουμε `ping google.com` θα είναι επιτυχές. Αυτό είναι λογικό, διότι στο mode NAT Network τα εικονικά μηχανήματα επικοινωνούν με το Internet μέσω του default gateway.

6.12

Ναι, τα PC1, PC2 επικοινωνούν μεταξύ τους, αφού στο mode NAT Network βρίσκονται στο ίδιο δίκτυο.

6.13

Όχι, δεν μπορούμε από το PC3 να κάνουμε `ping` στα άλλα δύο μηχανήματα, διότι το PC3 είναι ακόμη σε NAT Mode, και βρίσκεται απομονωμένο από το NAT Network των άλλων δύο.

6.14

Αφού κάνουμε `ping 10.0.2.15` (υποτίθεται στο PC1) και `ping 10.0.2.4` (υποτίθεται στο PC2), παίρνουμε απάντηση και στα δύο. Όμως, αν παρατηρήσουμε τον πίνακα `arp` του PC3, θα δούμε ότι οι φυσικές διευθύνσεις που αντιστοιχούν στις `10.0.2.4` και `10.0.2.15` δεν είναι ίδιες με αυτές των μηχανημάτων PC{1,2}. Συγκεκριμένα, η φυσική διεύθυνση που αντιστοιχεί στην `10.0.2.15` είναι του ίδιου του μηχανήματος PC3, ενώ η φυσική διεύθυνση που αντιστοιχεί στην `10.0.2.4` είναι αυτή του εξυπηρετητή TFTP.

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.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 4
Εισαγωγή στη δρομολόγηση

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 29/03/2022

Άσκηση 1 (προετοιμασία): Διευθύνσεις IP

1.1

Ο αριθμός δικτύου δεν είναι παρά ένα μέρος της διεύθυνσης IP. Συνολικά, η διεύθυνση IP αποτελείται από τον αριθμό δικτύου και τον αριθμό host.

1.2

Ο αριθμός δικτύου της διεύθυνσης `192.220.147.2/22` είναι `192.220.144` (προκύπτει από την σύζευξη της διεύθυνσης IP με τη μάσκα υποδικτύου).

1.3

Για 100 συσκευές χρειαζόμαστε 7 bits ($2^7 = 128$). Αυτό σημαίνει ότι μένουν για τα διαφορετικά υποδίκτυα $32 - 22 - 7 = 3$ bits, άρα μπορούν να δημιουργηθούν $2^3 = 8$ υποδίκτυα.

1.4

Η κλάση C παρέχει 254 διευθύνσεις για συσκευές.

1.5

a. Όχι.

b. Ναι, εμπίπτει στην περιοχή `10.0.0.0/8`.

c. Όχι.

d. Όχι.

e. Ναι, εμπίπτει στην περιοχή `192.168.0.0/16`.

1.6

Ο δρομολογητής καταλαβαίνει ότι μπορεί να στείλει απευθείας πακέτα σε κάποια συσκευή μέσω των διεπαφών του, αν δει ότι η συσκευή βρίσκεται στο ίδιο υποδίκτυο με αυτόν (δηλαδή στο ίδιο υποδίκτυο με κάποια από τις διεπαφές του). Ο έλεγχος γίνεται εφαρμόζοντας την κατάλληλη μάσκα υποδικτύου κάθε φορά.

1.7

Η διεύθυνση εκπομπής στο δίκτυο `10.50.10.0/23` είναι η `10.50.11.255/23`.

1.8

Η κλάση της διεύθυνσης `208.23.55.11` είναι C, επειδή τα πρώτα τρία bit είναι 110.

1.9

Το πλήθος των διευθύνσεων που είναι διαθέσιμες για συσκευές στο δίκτυο `147.102.0.0/17` είναι $2^{15} - 2$, γιατί έχουμε $32 - 17 = 15$ bits για τον αριθμό host, και το -2 προκύπτει επειδή δεν μετράμε τη διεύθυνση δικτύου και τη διεύθυνση εκπομπής.

1.10

Οι διευθύνσεις του ΕΜΠ ξεκινούν με $147 = 1001\ 0011$, οπότε τα 2 πρώτα bit είναι 10. Αυτό σημαίνει ότι ανήκουν στην κλάση B.

1.11

- Υποδίκτυο με 100 υπολογιστές: `10.11.12.0/25`.
- Υποδίκτυο με 60 υπολογιστές: `10.11.12.128/26`.
- Υποδίκτυο με 20 υπολογιστές: `10.11.12.192/27`.
- Υποδίκτυο με 10 υπολογιστές: `10.11.12.224/28`.

1.12

Ναι, υπάρχει χώρος για ένα ακόμη υποδίκτυο, το οποίο μπορεί να έχει μέχρι $2^4 - 2 = 14$ υπολογιστές.

1.13

Μπορούμε να τις συντμήσουμε ομαδοποιώντας τις ως εξής:

171.12.4.0/24, 171.12.5.0/24, 171.12.6.0/24, 171.12.7.0/24 → 171.12.4.0/22

171.12.8.0/24 → 171.12.8.0/24 (δεν μπορεί να γίνει κάποιου είδους σύντμηση)

Να σημειωθεί ότι δεν μπορούμε να κάνουμε μία σύντμηση του τύπου 171.12.0.0/20 που θα συμπεριλάμβανε και τις 5 παραπάνω διευθύνσεις, διότι έτσι συμπεριλαμβάνουμε ένα υπερσύνολο των παραπάνω διευθύνσεων, οι οποίες δεν είναι δεδομένες.

Άσκηση 2: Ένα απλό δίκτυο

2.1

Ναι, χρησιμοποιήσαμε την επιλογή αυτή, ώστε οι εικονικές κάρτες δικτύου να έχουν μοναδικές φυσικές διευθύνσεις, όπως συμβαίνει και με τις κανονικές κάρτες δικτύου.

2.2

Εκτελούμε ping από το PC1 στα PC2, PC3, PC4 και καταγράφουμε τα αποτελέσματα.

Οι εντολές είναι:

```
ping 192.168.1.2
ping 192.168.1.18
ping 192.168.1.29
```

Και τα αποτελέσματα:

- PC1 → PC2: Επιτυχές.
- PC1 → PC3: Επιτυχές.
- PC1 → PC4: Δεν λαμβάνεται ποτέ απάντηση.

2.3

Εκτελούμε ping από το PC2 στα PC3, PC4 και καταγράφουμε τα αποτελέσματα.

Οι εντολές είναι:

```
ping 192.168.1.18
ping 192.168.1.29
```

Και τα αποτελέσματα:

- PC2 → PC3: No route to host.
- PC2 → PC4: No route to host.

2.4

Εκτελούμε ping από το PC4 στα PC1, PC2, PC3 και καταγράφουμε τα αποτελέσματα.

Οι εντολές είναι:

```
ping 192.168.1.1
ping 192.168.1.2
ping 192.168.1.18
```

Και τα αποτελέσματα:

- PC4 → PC1: No route to host.
- PC4 → PC2: No route to host.
- PC4 → PC3: Επιτυχές.

2.5

Εκτελούμε ping από το PC3 στα PC1, PC2 και καταγράφουμε τα αποτελέσματα.

Οι εντολές είναι:

```
ping 192.168.1.1
ping 192.168.1.2
```

Και τα αποτελέσματα:

- PC3 → PC1: Επιτυχές.
- PC3 → PC2: Δεν λαμβάνεται ποτέ απάντηση.

2.6

Σε ορισμένα από τα προηγούμενα ping εμφανίζεται ”No route to host” διότι ο αποστολέας, εφαρμόζοντας τη μάσκα υποδικτύου τόσο στη δική του διεύθυνση, όσο και στου παραλήπτη, διαπιστώνει ότι αποστολέας και παραλήπτης δεν βρίσκονται στο ίδιο υποδίκτυο, και άρα απαιτείται η παρουσία δρομολογητή για να φτάσουν τα πακέτα του ping στον προορισμό τους. Αφού δεν υπάρχει δρομολογητής, δεν υπάρχει και διαδρομή προς τον παραλήπτη, εξού και το μήνυμα ”No route to host”.

Για παράδειγμα, όταν ο PC2 κάνει ping στο PC3, έχουμε:

```
PC2:
192.168.1.2 & 255.255.255.240 = 192.168.1.0
192.168.1.18 & 255.255.255.240 = 192.168.1.16
192.168.1.0 != 192.168.1.16, so PC2 and PC3 are not in the same subnet.
```

2.7

Σε ορισμένα από τα προηγούμενα ping δεν λαμβάνουμε απάντηση, διότι ο αποστολέας, εφαρμόζοντας τη μάσκα υποδικτύου του τόσο στη δική του διεύθυνση, όσο και στου παραλήπτη, θεωρεί πως βρίσκεται στο ίδιο υποδίκτυο με τον παραλήπτη, όμως αυτή τη φορά ο παραλήπτης, εφαρμόζοντας τη δική του μάσκα υποδικτύου, διαπιστώνει ότι δεν βρίσκεται στο ίδιο υποδίκτυο με τον αποστολέα. Αυτό έχει ως συνέπεια να στέλνει μεν ο αποστολέας τα πακέτα του ping προς τον παραλήπτη, όμως ο παραλήπτης δεν στέλνει ποτέ απάντηση.
Για παράδειγμα, όταν ο PC1 κάνει ping στο PC4, έχουμε:

```
PC1:
192.168.1.1 & 255.255.255.0 = 192.168.1.0
192.168.1.29 & 255.255.255.0 = 192.168.1.0
192.168.1.0 == 192.168.1.0, so PC1 thinks PC4 is in the same subnet.

PC4:
192.168.1.1 & 255.255.255.240 = 192.168.1.0
192.168.1.29 & 255.255.255.240 = 192.168.1.16
192.168.1.0 != 192.168.1.16, so PC1 and PC4 are not in the same subnet.
```

2.8

Στα PC1 και PC3, εκτελούμε τις εντολές:

```
PC1: ifconfig em0 192.168.1.1/28
PC3: ifconfig em0 192.168.1.18/28
```

2.9

Δοκιμάζουμε τα προηγουμένως επιτυχή ping:

```
PC1 --> PC2: ping 192.168.1.2
PC1 --> PC3: ping 192.168.1.18
PC4 --> PC3: ping 192.168.1.18
PC3 --> PC1: ping 192.168.1.1
```

Βλέπουμε ότι αποτυγχάνουν τα PC1 → PC3 και PC3 → PC1.

2.10

Δοκιμάζουμε τα ping όπου πριν δεν λαμβάναμε απάντηση:

```
PC1 --> PC4: ping 192.168.1.29
PC3 --> PC2: ping 192.168.1.2
```

Βλέπουμε ότι τώρα εμφανίζεται το μήνυμα ”No route to host”.

Άσκηση 3: Ένα απλό δίκτυο με δρομολογητή

3.1

Αλλάξαμε το εσωτερικό δίκτυο που βρίσκονται τα PC3 και PC4 από τις ρυθμίσεις της κάθε μηχανής, ως εξής:

Machine → Settings → Network → Name: LAN2.

3.2

Κάνουμε `ping 192.168.1.14` από το PC1, και καταγράφουμε στον R1 τα πακέτα στο δίκτυο LAN1 με `tcpdump -i em0`. Παρατηρούμε τόσο πακέτα ARP, όσο και ICMP.

3.3

Κάνουμε `ping 192.168.1.17` από το PC3, και καταγράφουμε στον R1 τα πακέτα στο δίκτυο LAN2 με `tcpdump -i em1`. Παρατηρούμε τόσο πακέτα ARP, όσο και ICMP.

3.4

Δοκιμάζουμε να κάνουμε `ping 192.168.1.18` από το PC1 στο PC3. Εμφανίζεται το μήνυμα "No route to host". Δεν παράγονται καθόλου ARP/ICMP πακέτα σε κανένα από τα LAN1,2.

3.5

Δοκιμάζουμε να κάνουμε `ping 192.168.1.1` από το PC3 στο PC1. Εμφανίζεται πάλι το μήνυμα "No route to host", και πάλι δεν παράγονται καθόλου ARP/ICMP πακέτα σε κανένα από τα LAN1,2.

3.6

Το ping απέτυχε προηγουμένως διότι τα PC1 και PC3 βρίσκονται σε διαφορετικά υποδίκτυα, και δεν υπάρχει ακόμα δρομολογητής (το R1 δεν λειτουργεί ακόμα ως δρομολογητής).

3.7

Τα περιεχόμενα του πίνακα ARP στο PC1 είναι (`arp -a`):

```
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0 physical address
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1@em0 physical address
```

3.8

Τα περιεχόμενα του πίνακα ARP στο PC2 είναι (`arp -a`):

```
192.168.1.2 at 08:00:27:5f:02:9c on em0 --> PC2@em0 physical address
```

3.9

Τα περιεχόμενα του πίνακα ARP στο R1 είναι (`arp -a`):

```
192.168.1.17 at 08:00:27:8b:5b:ee on em1 --> R1@em1 physical address
192.168.1.18 at 08:00:27:a3:d0:ee on em1 --> PC3@em0 physical address
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0 physical address
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1@em0 physical address
```

3.10

Καθαρίζουμε τον πίνακα ARP στον R1 με την εντολή `arp -d -a` και ξαναβλέπουμε τα περιεχόμενα με `arp -a`. Παρατηρούμε ότι δεν έχουν διαγραφεί οι εγγραφές που αφορούν στην φυσική διεύθυνση των διεπαφών `em0` και `em1` του R1.

3.11

Ξεκινάμε ένα `tcpdump -i em0 "arp or icmp"` στο R1 σε νέα κονσόλα, και στην αρχική κονσόλα τρέχουμε τις εντολές:

```
ping -c 1 192.168.1.1
ping -c 1 192.168.1.2
```

3.12

Τα περιεχόμενα του πίνακα ARP στον R1 είναι τώρα: (`arp -a`)

```
192.168.1.17 at 08:00:27:8b:5b:ee on em1 --> R1@em1 physical address
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0 physical address
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1@em0 physical address
192.168.1.2 at 08:00:27:5f:02:9c on em0 --> PC2@em0 physical address
```

Όταν κάνουμε ping από το R1 στο PC1, στέλνεται ένα broadcast ARP Request στο LAN1, στο οποίο απαντάει ο PC1, γι᾽ αυτό και έχουμε την αντίστοιχη εγγραφή για την φυσική διεύθυνση του PC1 στον πίνακα ARP του R1. Το ίδιο ισχύει και όταν κάνουμε ping από το R1 στο PC2, γι᾽ αυτό και έχουμε την αντίστοιχη εγγραφή για την φυσική διεύθυνση του PC2 στον πίνακα ARP του R1.

3.13

Τα περιεχόμενα του πίνακα ARP στο PC1 είναι τώρα: (`arp -a`)

```
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0 physical address
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1@em0 physical address
```

Όταν κάνουμε ping από το R1 στο PC1, στέλνεται ένα broadcast ARP Request στο LAN1, το οποίο βλέπει ο PC1 και έτσι μαθαίνει την φυσική διεύθυνση της διεπαφής `em0` του R1.

3.14

Επαναλαμβάνουμε την καταγραφή στο LAN2 με την εντολή `tcpdump -i em1 "arp or icmp"` και αυτή τη φορά κάνουμε:

```
ping -c 1 192.168.1.18
ping -c 1 192.168.1.29
```

Παρατηρούμε ότι στον πίνακα ARP (`arp -a`) του R1 προστίθενται οι εξής γραμμές:

```
192.168.1.29 at 08:00:27:fe:70:e0 on em1 --> PC4@em0 physical address
192.168.1.18 at 08:00:27:a3:d0:ee on em1 --> PC3@em0 physical address
```

3.15

Με τη βοήθεια του πίνακα ARP βρίσκουμε όλες τις αντιστοιχίσεις:

```
PC1@em0 (192.168.1.1) --> 08:00:27:02:3e:c4
PC2@em0 (192.168.1.2) --> 08:00:27:5f:02:9c
PC3@em0 (192.168.1.18) --> 08:00:27:a3:d0:ee
PC4@em0 (192.168.1.29) --> 08:00:27:fe:70:e0
R1@em0 (192.168.1.14) --> 08:00:27:40:c2:01
R1@em1 (192.168.1.17) --> 08:00:27:8b:5b:ee
```

3.16

Στον R1 ξεκινάμε μία καταγραφή στο LAN1 με την εντολή `tcpdump -i em0` και από μία άλλη κονσόλα του κάνουμε `ping -c 3 192.168.1.5`, δηλαδή σε ένα ανύπαρκτο μηχάνημα. Παράγονται μόνο μηνύματα ARP, αφού ο R1 αναζητά την φυσική διεύθυνση του ανύπαρκτου μηχανήματος (το οποίο βρίσκεται θεωρητικά στο ίδιο υποδίκτυο), και επειδή δεν λαμβάνει απάντηση, δεν αποστέλλει το αντίστοιχο μήνυμα ICMP.

3.17

Εκτελώντας `arp -a`, βλέπουμε ότι δεν έχει γίνει κάποια καταχώρηση για το ανύπαρκτο μηχάνημα. Αυτό είναι λογικό, αφού δεν λήφθηκε κάποιο ARP Reply ώστε να γίνει η καταχώρηση.

3.18

Παρατηρούμε ότι μόλις εκτελέσουμε `ping -c 6 192.168.1.5`, τότε εμφανίζεται το μήνυμα ”Host is down”, αφού μετά από 6 δοκιμές ο R1 συμπεραίνει με αρκετά μεγάλη σιγουριά ότι ο παραλήπτης δεν είναι ενεργός.

Άσκηση 4: Προεπιλεγμένος δρομολογητής

4.1

Για να ενεργοποιήσουμε τη λειτουργία προώθησης πακέτων IPv4 στον R1 εκτελούμε την εντολή: `sysctl net.inet.ip.forwarding=1`

4.2

Για να παραμείνει η ρύθμιση μεταξύ επανεκκινήσεων, θα πρέπει να προσθέσουμε στο αρχείο `/etc/rc.conf` τη γραμμή:

```
gateway_enable="YES"
```

4.3

Δοκιμάζουμε την εντολή `ping 192.168.1.18` από το PC1 στο PC3. Εμφανίζεται και πάλι το μήνυμα ”No route to host”, αφού δεν έχει οριστεί προεπιλεγμένη πύλη ακόμα.

4.4

Με την εντολή `netstat -rn` βλέπουμε τον πίνακα δρομολόγησης IPv4 στο PC1. Δεν υπάρχει διαδρομή για το LAN2.

4.5

Στο PC1 ορίζουμε ως προεπιλεγμένη πύλη τον δρομολογητή R1 με την εντολή:

```
route add default 192.168.1.14
```

4.6

Βλέπουμε ότι στον πίνακα δρομολόγησης του PC1 (`netstat -rn`) προστίθεται η εγγραφή:

```
Destination | Gateway      | Flags | Netif | Expire
============|==============|=======|=======|=======
default     | 192.168.1.14 | UGS   | em0   |
```

4.7

Δοκιμάζουμε και πάλι την εντολή `ping 192.168.1.18` από το PC1 στο PC3. Βλέπουμε ότι τώρα δεν εμφανίζεται το μήνυμα "No route to host", αλλά παρολαυτά δεν λαμβάνεται απάντηση.

4.8

Με χρήση του tcpdump βλέπουμε ότι παράγονται πακέτα ICMP echo request τόσο στο LAN1 όσο και στο LAN2, αλλά όχι ICMP echo reply. Αυτό συμβαίνει επειδή ο PC1 έχει ορίσει προεπιλεγμένη πύλη, αλλά ο PC3 όχι. Επομένως, δεν μπορεί ο PC3 να απαντήσει στα ICMP echo request του PC1.

4.9

Στο PC3 ορίζουμε ως προεπιλεγμένη πύλη τον δρομολογητή R1 με την εντολή:

```
route add default 192.168.1.17
```

4.10

Δοκιμάζουμε πάλι την εντολή `ping 192.168.1.18` από το PC1 στο PC3. Τώρα το ping είναι επιτυχές. Αυτό συμβαίνει διότι τόσο το PC1 όσο και το PC3 έχουν ορίσει default gateway τον R1, ο οποίος αναλαμβάνει τη σωστή προώθηση των ICMP echo request/reply.

4.11

Εκτελούμε `traceroute 192.168.1.18` από το PC1 στο PC3. Βλέπουμε 2 βήματα.

4.12

Καθαρίζουμε τους πίνακες ARP σε όλα τα εικονικά μηχανήματα (PC1, PC3 και R1) με την εντολή `arp -d -a`.

4.13

Στον δρομολογητή R1 ξεκινάμε τις δύο ζητούμενες καταγραφές στα LAN1 και LAN2, με τις εντολές:

```
LAN1: tcpdump -evvv -i em0
LAN2: tcpdump -evvv -i em1
```

4.14

Κάνουμε `ping -c 1 192.168.1.18`.

4.15

ICMP echo request στο LAN1:

- Ethernet: 08:00:27:02:3e:c4 (PC1) → 08:00:27:40:c2:01 (R1@em0)
- IP: 192.168.1.1 (PC1) → 192.168.1.18 (PC3)

4.16

ICMP echo request στο LAN2:

- Ethernet: 08:00:27:8b:5b:ee (R1@em1) → 08:00:27:a3:d0:ee (PC3)
- IP: 192.168.1.1 (PC1) → 192.168.1.18 (PC3)

4.17

Οι διευθύνσεις IP δεν αλλάζουν κατά την προώθηση του πακέτου από τον δρομολογητή. Αντίθετα, οι διευθύνσεις πηγής και προορισμού αλλάζουν κατά την διαδρομή του πακέτου, ώστε οι διευθύνσεις πηγής και προορισμού να αντικατοπτρίζουν την φυσική προέλευση και τον φυσικό επόμενο προορισμό του πακέτου.

4.18

Εκτελούμε `ssh lab@192.168.1.18` από το PC1 στο PC3.

4.19

Χρησιμοποιούμε την `netstat -an | grep 192.168.1.1` και βλέπουμε ότι χρησιμοποιείται το TCP ως πρωτόκολλο μεταφοράς, η τοπική θύρα, δηλαδή του PC3, είναι η 22, και η απομακρυσμένη θύρα, δηλαδή του PC1, είναι η 56664.

4.20

Εκτελούμε την εντολή `netstat -p tcp` στον R1. Παρατηρούμε ότι η εντολή δεν εμφανίζει κάποια έξοδο. Αυτό συμβαίνει διότι ο R1 λειτουργεί μέχρι το στρώμα δικτύου με σκοπό την προώθηση των πακέτων, και δεν έχει καμία εμπλοκή στο στρώμα μεταφοράς.

Άσκηση 5: Προθέματα δικτύου και δρομολόγηση

5.1

Ορίζουμε σε όλα τα PC ως προεπιλεγμένη πύλη τον R1 με τις εντολές:

```
PC1: route add default 192.168.1.14
PC2: route add default 192.168.1.14
PC3: route add default 192.168.1.17
PC4: route add default 192.168.1.17
```

5.2

Καθαρίζουμε τους πίνακες ARP σε όλα τα μηχανήματα με την εντολή:

```
arp -d -a
```

5.3

Ξεκινάμε τη ζητούμενη καταγραφή στο R1 με την εντολή:

```
tcpdump -i em0 "icmp or arp"
```

5.4

Ξεκινάμε μία αντίστοιχη καταγραφή στο PC4 με την εντολή:

```
tcpdump -i em0 "icmp or arp"
```

5.5

Από το PC1 εκτελούμε:

```
ping -c 1 192.168.1.2
ping -c 1 192.168.1.18
ping -c 1 192.168.1.29
```

Τα ping είναι επιτυχή.

5.6

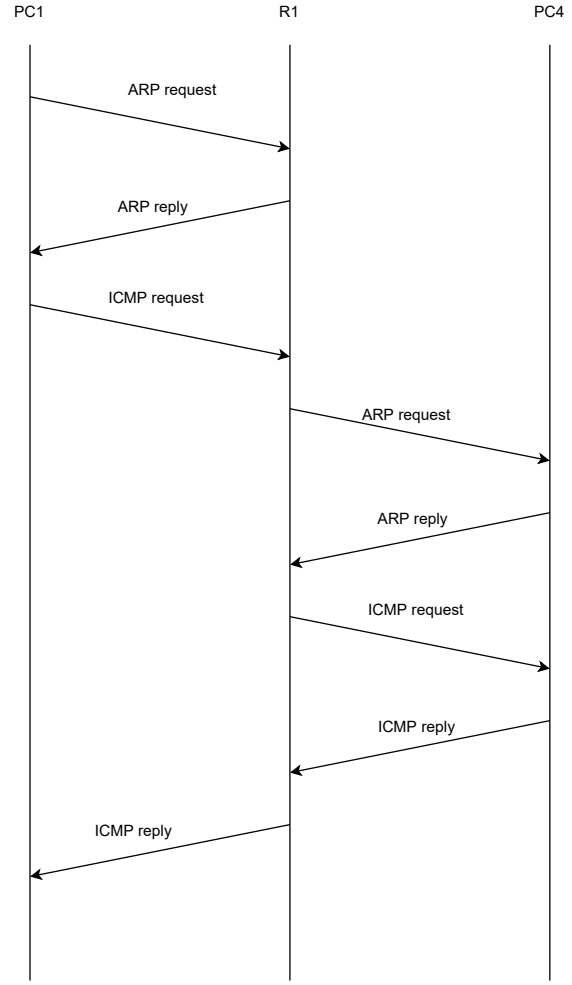
Σταματάμε τις καταγραφές και καταγράφουμε το περιεχόμενο των πινάκων ARP (`arp -a`) σε όλα τα μηχανήματα μετά την ολοκλήρωση των ping:

```
PC1
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1
192.168.1.2 at 08:00:27:5f:02:9c on em0 --> PC2

PC2
```

```
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1
192.168.1.2 at 08:00:27:5f:02:9c on em0 --> PC2

PC3
192.168.1.17 at 08:00:27:8b:5b:ee on em0 --> R1@em1
192.168.1.18 at 08:00:27:a3:d0:ee on em0 --> PC3

PC4
192.168.1.29 at 08:00:27:fe:70:e0 on em0 --> PC4
192.168.1.17 at 08:00:27:8b:5b:ee on em0 --> R1@em1

B1
192.168.1.29 at 08:00:27:fe:70:e0 on em1 --> PC4
192.168.1.17 at 08:00:27:8b:5b:ee on em1 --> R1@em1
192.168.1.18 at 08:00:27:a3:d0:ee on em1 --> PC3
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0
192.168.1.1 at 08:00:27:02:3e:c4 on em0 --> PC1
```

5.7

Η χρονική σειρά ανταλλαγής των μηνυμάτων έχει ως εξής:



5.8

Καθαρίζουμε τους πίνακες ARP σε όλα τα μηχανήματα με `arp -d -a`. Ξεκινάμε τις ζητούμενες καταγραφές στα PC3, PC4 και R1 για το LAN2, με τις εντολές:

```
PC3: tcpdump -e -i em0 "icmp or arp"
PC4: tcpdump -e -i em0 "icmp or arp"
R1: tcpdump -e -i em1 "icmp or arp"
```

5.9

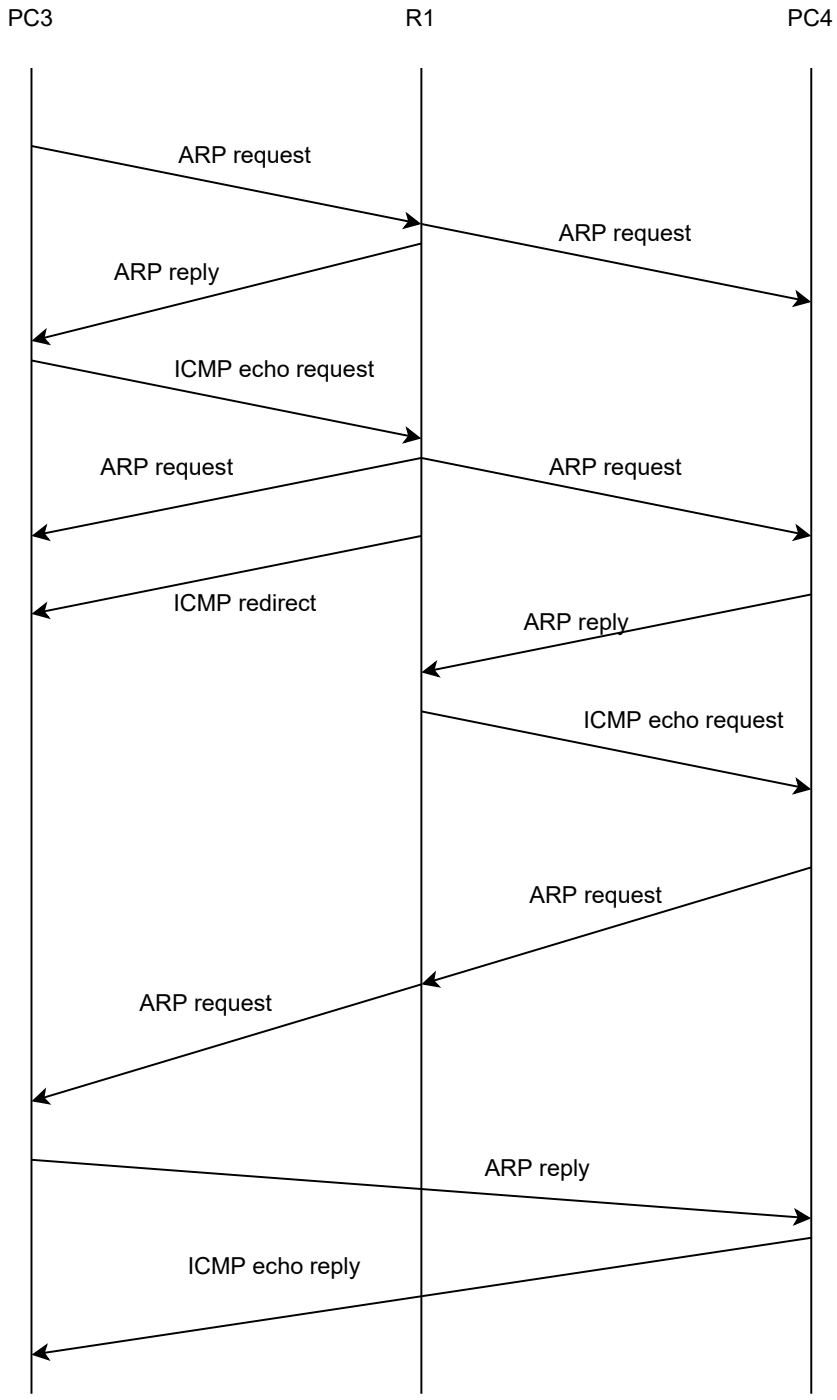
Από νέο παράθυρο στο PC3 κάνουμε ping στο PC4 με την εντολή `ping -c 1 192.168.1.29`. Το ping είναι επιτυχές. Επίσης, στην έξοδο του ping παρατηρούμε ότι ελήφθη ένα μήνυμα ICMP redirect (New addr: 192.168.1.29) από τον R1.

5.10

Σταματάμε τις καταγραφές και καταγράφουμε το περιεχόμενο των πινάκων ARP στα R1, PC3, PC4 (`arp -a`):

```
PC3
192.168.1.17 at 08:00:27:8b:5b:ee on em0 --> R1@em1
192.168.1.18 at 08:00:27:a3:d0:ee on em0 --> PC3

PC4
192.168.1.29 at 08:00:27:fe:70:e0 on em0 --> PC4
192.168.1.17 at 08:00:27:8b:5b:ee on em0 --> R1@em1
192.168.1.18 at 08:00:27:a3:d0:ee on em0 --> PC3

R1
192.168.1.29 at 08:00:27:fe:70:e0 on em1 --> PC4
192.168.1.17 at 08:00:27:8b:5b:ee on em1 --> R1@em1
192.168.1.18 at 08:00:27:a3:d0:ee on em1 --> PC3
192.168.1.14 at 08:00:27:40:c2:01 on em0 --> R1@em0
```

5.11



5.12

Το PC3 αναζητεί με ARP τη διεύθυνση του R1@em1, ενώ το PC4 αναζητεί τη διεύθυνση του PC3.

5.13

Το μήνυμα ICMP request του PC3 αποστέλλεται προς τον R1 και όχι απευθείας στον PC4 διότι ο PC3 βλέπει πως ο PC4 βρίσκεται σε διαφορετικό υποδίκτυο, οπότε στέλνει το μήνυμα στην

προεπιλεγμένη πύλη, δηλαδή στον R1.

5.14

Ο R1 χειρίζεται το προηγούμενο ICMP request στέλνοντας ένα μήνυμα ICMP redirect στον PC3, ώστε να τον ενημερώσει ότι το επόμενο σωστό βήμα ήταν ο PC4 και όχι ο R1.

5.15

Η απάντηση ICMP reply του PC4 στο PC3 στάλθηκε απευθείας.

5.16

Εκτελούμε:

```
PC3: tcpdump -e -i em0 "icmp"
PC4: tcpdump -e -i em0 "icmp"
R1: tcpdump -e -i em1 "icmp"
```

5.17

Κάνουμε `ping 192.168.1.29` από το PC3 στο PC4. Παρατηρούμε ότι ο PC3 λαμβάνει συνεχώς ICMP redirect από τον R1, ο οποίος προσπαθεί να τον ενημερώσει ότι το επόμενο σωστό βήμα δεν είναι ο R1 αλλά το PC4. Κατά τα άλλα η σχετική με το ping κίνηση κυκλοφορεί κανονικά. (ο R1 προωθεί το ICMP echo request στον PC4, και ο PC4 απαντά, στέλνοντας απευθείας στον PC3 ICMP echo reply).

5.18

Στο PC3 εκτελούμε `ifconfig em0 192.168.1.18/28`. Η προκαθορισμένη διαδρομή (`netstat -rn`) διαγράφεται από τον πίνακα δρομολόγησης, όπως ήταν αναμενόμενο.

5.19

Στο PC3, εκτελούμε `route add 192.168.1.24/29 192.168.1.17`. Ύστερα με την εντολή `netstat -rn` βλέπουμε ότι ο πίνακας δρομολόγησης του PC3 για προορισμούς σε υποδίκτυα του `192.168.0.0/16` είναι:

```
Destination        Gateway         Flags   Netif     Expire
192.168.1.16/28    link#1          U       em0
192.168.1.18       link#1          UHS     lo0
192.168.1.24/29    192.168.1.17    UGS     em0
```

5.20

Επαναλαμβάνουμε τις ζητούμενες καταγραφές στο LAN2:

```
PC3: tcpdump -e -i em0 "icmp"
PC4: tcpdump -e -i em0 "icmp"
R1: tcpdump -e -i em1 "icmp"
```

Εκτελούμε πάλι `ping 192.168.1.29`. Αυτή τη φορά το PC3 λαμβάνει μόνο μία φορά το μήνυμα ICMP redirect, επειδή πλέον επιλέγει το PC4 σαν επόμενο βήμα, όπως το συμβουλεύει ο R1, αφού PC3 και PC4 βρίσκονται στο ίδιο υποδίκτυο πλέον. Έτσι, μετά την αποστολή του ICMP redirect, ο R1 δεν εμπλέκεται πλέον στην επικοινωνία των PC3 και PC4 (αυτά επικοινωνούν μεταξύ τους απευθείας πλέον).

5.21

Πλέον έχει προστεθεί στον πίνακα δρομολόγησης του PC3 η εγγραφή:

```
Destination     Gateway         Flags   Netif    Expire
192.168.1.29    192.168.1.29    UGHD    em0
```

με τη βασική διαφορά ότι πρόκειται για εγγραφή με διεύθυνση προθέματος 32 bits, και άρα μεγαλύτερη προτεραιότητα.

5.22

Το PC3 δεν επικοινωνεί με τα μηχανήματα του LAN1, διότι δεν υπάρχει καμία εγγραφή στον πίνακα δρομολόγησης που να αφορά τα μηχανήματα του LAN1, και δεν υπάρχει ούτε default gateway.

5.23

Ορίζουμε ως προεπιλεγμένη πύλη τον R1 με την εντολή `route add default 192.168.1.17`. Η διαδρομή που θα επιλεχθεί αν κάνουμε ping από το PC3 στο PC4 είναι η απευθείας διαδρομή, διότι ο PC3 έχει λάβει υπόψιν του το ICMP redirect και έχει προσθέσει την κατάλληλη εγγραφή στον πίνακα δρομολόγησης του.

Άσκηση 6: Router on a stick

6.1

Στο R1, αλλάζουμε το Promiscuous mode σε ”Allow VMs”, και ύστερα με τις εντολές:

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

δημιουργούμε τη ζητούμενη γέφυρα.

6.2

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
```

6.3

Αντίστοιχα, στο PC2:

```
ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.2/24
```

6.4

Αντίστοιχα, στο PC3:

```
ifconfig em0.6 create vlan 6 vlandev em0 inet 192.168.6.18/24
```

6.5

Αντίστοιχα, στο PC4:

```
ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.29/24
```

6.6

Από το PC3: `ping 192.168.1.1` → επιτυχές
`ping 192.168.5.1` → ανεπιτυχές
`ping 192.168.6.1` → επιτυχές

6.7

Από το PC4: `ping 192.168.1.1` → επιτυχές
`ping 192.168.5.1` → επιτυχές
`ping 192.168.6.1` → ανεπιτυχές

6.8

Κάποια από τα ping απέτυχαν, διότι το PC3 δεν έχει εγγραφή στον πίνακα προώθησης σχετική με τη διεύθυνση `192.168.5.1`, ενώ το PC4 δεν έχει εγγραφή στον πίνακα προώθησης σχετική με τη διεύθυνση `192.168.6.1`.

6.9

Στο PC3 ορίζουμε ως νέα προεπιλεγμένη πύλη το PC1 με την εντολή:

```
route change default 192.168.6.1
```

Πλέον μπορούμε να κάνουμε ping σε όλες τις διεπαφές του PC1, αφού όταν κάνουμε `ping 192.168.5.1` προτιμάται η default gateway και το πακέτο δρομολογείται σωστά.

6.10

Ναι, μπορούμε να κάνουμε ping από το PC4 σε όλες τις διεπαφές του PC2.

6.11

Όχι, δεν μπορούμε να κάνουμε ping από το PC3 σε καμία από τις διεπαφές του PC2.

6.12

Με την εντολή `sysctl net.inet.ip.forwarding=1` ενεργοποιούμε τη λειτουργία προώθησης πακέτων IPv4 στο PC1, και στο PC2 ορίζουμε ως νέα προεπιλεγμένη πύλη την `192.168.1.1` με την εντολή `route change default 192.168.1.1`.

6.13

Ναι, τώρα τα ping από το PC3 προς τις διεπαφές του PC2 επιτυγχάνουν.

6.14

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

- PC1 → `08:00:27:02:3e:c4`
- PC2 → `08:00:27:5f:02:9c`
- PC3 → `08:00:27:a3:d0:ee`

Ύστερα, καθαρίζουμε τα ARP tables σε αυτά με την εντολή `arp -d -a`.

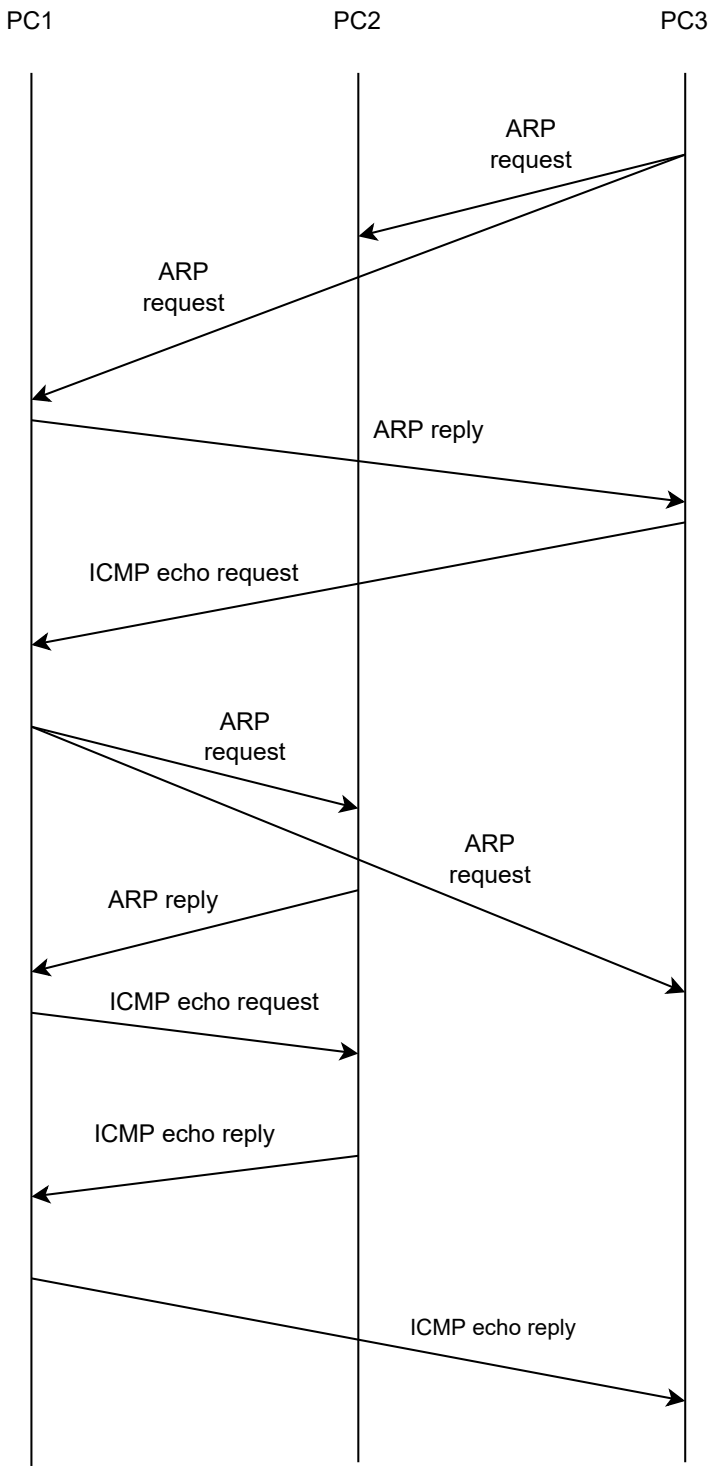
6.15

Ξεκινάμε τις ζητούμενες καταγραφές στα PC1,2,3 με τις εντολές:

```
PC1: tcpdump -e -i em0
PC2: tcpdump -e -i em0
PC3: tcpdump -e -i em0
```

6.16

Ανοίγουμε νέο παράθυρο εντολών στο PC3 και εκτελούμε την εντολή `ping -c 1 192.168.1.2`. Η ανταλλαγή των πακέτων έχει ως εξής:



6.17

Ξεκινάμε ένα `ping 192.168.5.29` από το PC3 προς τη διεπαφή του PC4 στο VLAN5 και το αφήνουμε να τρέχει. Το ping δεν είναι επιτυχές.

6.18

Ξεκινάμε τη ζητούμενη καταγραφή στο PC4 με την εντολή `tcpdump -e -i em0`. Επιβεβαιώνουμε ότι ο PC4 δεν απαντά, παρόλο που λαμβάνει τα ICMP echo requests. Αυτό συμβαίνει επειδή το PC4 δεν έχει εγγραφή στον πίνακα δρομολόγησης που να αντιστοιχεί στη διεύθυνση από την οποία προέρχεται το ICMP echo request.

6.19

Αλλάζουμε την προεπιλεγμένη πύλη με την εντολή `route change default 192.168.5.1`. Τώρα το προηγούμενο ping επιτυγχάνει, αφού το PC4 προωθεί στην προεπιλεγμένη πύλη τα ICMP echo replies, και το PC1 αναλαμβάνει να τα προωθήσει στο PC3.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 5
Στατική δρομολόγηση

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 29/03/2022

Άσκηση 1: Δρομολόγηση σε ένα βήμα

1.1

Με τις εντολές:

```
PC1:
ifconfig em0 192.168.1.2/24

PC2:
ifconfig em0 192.168.2.2/24

R1:
ifconfig em0 192.168.1.1/24
ifconfig em1 192.168.2.1/24
```

1.2

Προσθέσαμε τη γραμμή:

```
gateway_enable="YES"
```

1.3

Προσθέτουμε τη ζητούμενη εγγραφή με την εντολή:

```
route add -net 192.168.2.0/24 192.168.1.1
```

1.4

Εκτελούμε `netstat -rn`. Η σημαία ”U” σημαίνει ότι η διαδρομή είναι ενεργή, η σημαία ”G” σημαίνει ότι ο προορισμός είναι πύλη, που θα αποφασίσει για το πώς θα προωθήσει τα πακέτα περαιτέρω, ενώ η σημαία ”S” σημαίνει ότι η διαδρομή έχει οριστεί στατικά.

1.5

Δοκιμάζουμε `ping 192.168.2.2` από το PC1 στο PC2 και παρατηρούμε ότι το PC1 δε λαμβάνει απάντηση.

1.6

Με `tcpdump -i em0` στα PC1 και PC2, παρατηρούμε ότι παράγονται πακέτα ICMP τόσο στο LAN1 όσο και στο LAN2. Παρολαυτά, στο PC2 δεν υπάρχει η κατάλληλη εγγραφή στον πίνακα δρομολόγησης ώστε να μπορεί να στείλει το PC2 ICMP reply στο PC1.

1.7

Προσθέτουμε τη ζητούμενη εγγραφή με την εντολή:

```
route add -net 192.168.1.0/24 192.168.2.1
```

1.8

Δοκιμάζουμε πάλι `ping 192.168.2.2` από το PC1 στο PC2. Πλέον υπάρχει επικοινωνία.

1.9

Δεν χρειάζεται να αλλάξουμε τον πίνακα δρομολόγησης του R1, διότι ο R1 διαθέτει διεπαφές και στα δύο υποδίκτυα (την `em0` στο `192.168.1.0/24` και την `em1` στο `192.168.2.0/24`), και άρα οι απαιτούμενες εγγραφές υπάρχουν ήδη στον πίνακα δρομολόγησής του.

Άσκηση 2: Proxy ARP

2.1

Καταργούμε τη στατική εγγραφή στο PC1 με την εντολή `route del 192.168.2.0/24`.

2.2

Αλλάζουμε το μήκος προθέματος στο PC1 με την εντολή `ifconfig em0 192.168.1.2/20`.

2.3

Από την προοπτική του PC1, τα PC2 και PC3 βρίσκονται στο ίδιο υποδίκτυο με αυτόν, αφού:

```
PC1 IP & PC1 subnet mask = 192.168.1.2 & 255.255.240.0 = 192.168.0.0
PC2 IP & PC1 subnet mask = 192.168.2.2 & 255.255.240.0 = 192.168.0.0
PC3 IP & PC1 subnet mask = 192.168.2.3 & 255.255.240.0 = 192.168.0.0
```

2.4

Από το PC1 κάνουμε `ping 192.168.2.2` και `ping 192.168.2.3`. Και από τα δύο ping λαμβάνουμε "Host is down".
Στον δρομολογητή ενεργοποιούμε το proxy ARP με την εντολή:

```
sysctl net.link.ether.inet.proxyall=1
```

2.5

Επαναλαμβάνουμε το `ping -c 1 192.168.2.2`. Τώρα το ping είναι επιτυχές. Αυτό συμβαίνει διότι ενεργοποιήσαμε τη λειτουργία proxy ARP στον R1, οπότε ο R1 απαντά στο ARP request του PC1 -που θέλει να μάθει τη φυσική διεύθυνση του PC2- με τη δική του διεύθυνση MAC, προωθεί το ICMP echo request στον PC2, απαντά πάλι με τη δική του διεύθυνση MAC στο ARP request του PC2 -που θέλει να μάθει τη φυσική διεύθυνση του PC1-, και τέλος προωθεί το ICMP echo reply του PC2 στο PC1.

2.6

Επαναλαμβάνουμε το `ping 192.168.2.3`. Το ping αποτυγχάνει, διότι δεν υπάρχει εγγραφή στον πίνακα δρομολόγησης του PC3 σχετική με το PC1, ώστε το PC3 να μπορεί να στείλει ICMP echo reply, όπως έγινε με το PC2.

2.7

Προσθέτουμε τη στατική εγγραφή στο PC3 με την εντολή:

```
route add 192.168.1.0/24 192.168.2.1
```

2.8

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

2.9

Ξεκινάμε τις ζητούμενες καταγραφές στο R1 με τις εντολές:

```
tcpdump -e -i em0
tcpdump -e -i em1
```

Επαναλαμβάνουμε το προηγούμενο ping από το PC1 στο PC3: `ping -c 1 192.168.2.3`.

2.10

Παρατηρούμε ότι ο R1 απαντά στο ARP request του PC1 με τη δική του φυσική διεύθυνση, και συγκεκριμένα με τη φυσική διεύθυνση της διεπαφής του στο LAN1.

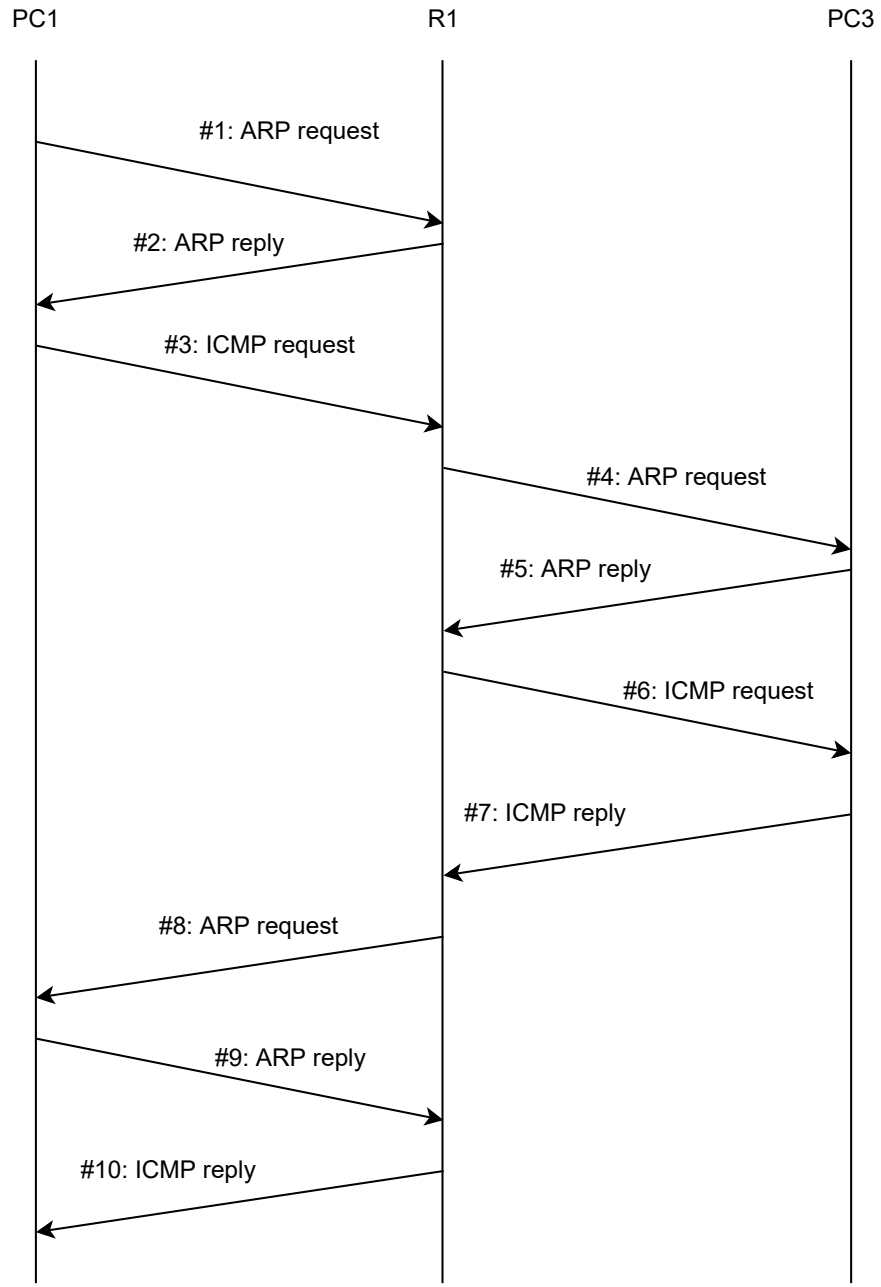
2.11

Ο PC1 στέλνει το ICMP request προς τη διεύθυνση `08:00:27:57:dd:d7`, δηλαδή τη φυσική διεύθυνση της διεπαφής του R1 στο LAN1.

2.12

Ο PC3 λαμβάνει το ICMP request από τη φυσική διεύθυνση της διεπαφής του R1 στο LAN2.

2.13



- #1: Επίλυση της φυσικής διεύθυνσης του PC3 για το PC1.
- #2: Proxy ARP, απάντηση στο ARP request #1. Ο R1 απαντά με τη δική του φυσική διεύθυνση αντί για τη διεύθυνση του PC3.
- #3: ICMP request από το PC1 με προορισμό το PC3.
- #4: Επίλυση της φυσικής διεύθυνσης του PC3 για το R1.

- #5: Απάντηση του ARP reply #4.
- #6: Προώθηση του ICMP request #3.
- #7: Απάντηση του ICMP request #6.
- #8: Επίλυση της φυσικής διεύθυνσης του PC1 για το R1. Να σημειωθεί ότι ο R1 δεν γνωρίζει τη φυσική διεύθυνση του PC1 λόγω του ARP request #1, αφού εκεί αναζητείται η φυσική διεύθυνση του PC3, και όχι του R1.
- #9: Απάντηση του ARP request #8.
- #10: Προώθηση του ICMP reply #7.

2.14

Ο πίνακας δρομολόγησης του PC1 (`netstat -rn`) έχει εγγραφές για τη loopback, το υποδίκτυο `192.168.0.0/20` και τη διεύθυνση `192.168.1.2`. Από αυτές η μόνη που μπορεί να φανεί χρήσιμη είναι αυτή του υποδικτύου `192.168.0.0/20`. Πρέπει λοιπόν το πρόθεμα δικτύου αυτής της εγγραφής να είναι τέτοιο ώστε να περιλαμβάνεται και το PC3 σε αυτό το υποδίκτυο.

```
PC3 IPv4:           11000000.10101000.00000010.00000011
PC1 subnet IPv4:    11000000.10101000.00000000.00000000
max prefix match == 22 bits
```

Επομένως το μέγιστο μήκος προθέματος για το PC1 είναι 22 bits.

2.15

Δίνουμε στο PC1 την εντολή `ifconfig em0 192.168.1.2/23`.

2.16

Στο PC1 εκτελούμε: `route add -net 192.168.2.0/24 -interface em0`

2.17

Στον πίνακα δρομολόγησης του PC1 εμφανίζεται:

```
Destination       Gateway              Netif
192.168.2.0/24    08:00:27:ca:53:74    em0
```

2.18

Τώρα το ping προς τον PC3 είναι επιτυχές. Αυτό συμβαίνει διότι ο PC1 βλέπει την εγγραφή `192.168.2.0/24` στον πίνακα δρομολόγησης και στέλνει το πακέτο στον R1, ο οποίος το προωθεί στον PC3, και ο PC3 βλέπει την εγγραφή `192.168.1.0` στον πίνακα δρομολόγησής του και μπορεί να απαντήσει. Για τα πακέτα ARP ισχύουν όσα έχουν αναφερθεί παραπάνω (proxy ARP κλπ).

2.19

Στον R1 ακυρώνουμε το proxy ARP με την εντολή: `sysctl net.link.ether.inet.proxyall=0`.

2.20

Στο PC1 εκτελούμε: `route change -net 192.168.2.0/24 192.168.1.1`.

2.21

Στο PC1 εκτελούμε: `ifconfig em0 192.168.1.2/24`.

2.22

Παρατηρούμε ότι η διαδρομή για το `192.168.2.0/24` δεν υπάρχει πλέον στον πίνακα δρομολόγησης του PC1. Την επαναπροσδιορίζουμε με την εντολή `route add -net 192.168.2.0/24 192.168.1.1`.

Άσκηση 3: Δρομολόγηση σε περισσότερα βήματα

3.1

Ορίζουμε διευθύνσεις IPv4 στον R1 με τις εντολές:

```
ifconfig em0 192.168.1.1/24
ifconfig em1 172.17.17.1/30
```

3.2

Ορίζουμε διευθύνσεις IPv4 στον R2 με τις εντολές:

```
ifconfig em0 172.17.17.2/30
ifconfig em1 192.168.2.1/24
```

3.3

Εκτελούμε `ping 192.168.2.2`. Εμφανίζεται μήνυμα ”Destination Host Unreachable”.

3.4

Με `tcpdump -e -i em0` και `tcpdump -e -i em1` στον R1 ελέγχουμε τα πακέτα που εκπέμπονται στα LAN1 και WAN1 αντίστοιχα. Στο LAN1 παρατηρούμε μηνύματα ”ICMP echo request” από το PC1 προς το R1, αλλά και μηνύματα ”ICMP host 192.168.2.2 unreachable” από το R1 στο PC1. Στο WAN1 δεν παρατηρούνται μηνύματα ICMP, αφού ο R1 δεν γνωρίζει πού να προωθήσει τα πακέτα που προορίζονται για το PC2 (δεν διαθέτει σχετική εγγραφή στον πίνακα δρομολόγησης).

3.5

Από το PC1 εκτελούμε `traceroute 192.168.2.2` και στην έξοδό της εντολής βλέπουμε 3 φορές το σύμβολο `!H`, που σημαίνει ότι ο PC1 έλαβε μήνυμα ”Destination host unreachable” από τον R1 (`192.168.1.1`).

3.6

Προσθέτουμε τη ζητούμενη εγγραφή μέσω της εντολής:

```
R1:
route add -net 192.168.2.0/24 172.17.17.2
```

3.7

Δοκιμάζουμε πάλι `ping 192.168.2.2` από το PC1 στο PC2. Δεν εμφανίζεται πλέον ”Destination Host Unreachable” αλλά πάλι δεν λαμβάνουμε απάντηση.

3.8

Με την εντολή `tcpdump -i em0` στο PC2 βλέπουμε ότι στο LAN2 παράγονται μηνύματα:

- ICMP echo request: Έχει προκύψει από την προώθηση του ICMP echo request του PC1 από τον R2.
- ICMP echo reply: Έχει προκύψει ως απάντηση στο ICMP echo request από τον PC2.
- ICMP host 192.168.1.2 unreachable: Έχει προκύψει από τον R2 και προορίζεται για τον PC2, επειδή ο R2 δεν γνωρίζει πού να προωθήσει το ICMP echo reply του PC2 που προορίζεται για τον PC1 (δεν έχει εγγραφή στον πίνακα δρομολόγησης σχετική με το PC1).

3.9

Δοκιμάζουμε ξανά `traceroute 192.168.2.2` από το PC1 προς το PC2. Δεν παρατηρούμε μηνύματα ICMP echo request στο WAN1 (εκτελέσαμε `tcpdump -i em1` στο R1). Αντ’ αυτού, παρατηρήσαμε UDP datagrams, επειδή η traceroute στέλνει UDP datagrams by default.

3.10

Στο LAN2 (εκτελέσαμε `tcpdump -i em1` στο R2) παρατηρούνται UDP datagrams και μηνύματα ”ICMP 192.168.2.2 udp port 3344x unreachable”, όπου $x = \{2, 3, 4, 5\}$.

3.11

Σύμφωνα με την ιστοσελίδα, δεν παράγεται το error message ”ICMP host unreachable” ως απάντηση στο error message ”ICMP 192.168.2.2 udp port 3344x unreachable”, διότι εξ’ ορισμού δεν παράγονται ICMP μηνύματα λάθους ως απάντηση σε άλλα ICMP μηνύματα λάθους (κάτι τέτοιο θα δημιουργούσε έναν κατακλυσμό από ICMP μηνύματα λάθους).

3.12

Στο R2 εκτελούμε την εντολή `route add 192.168.1.0/24 172.17.17.1`.

3.13

Πλέον μπορούμε να κάνουμε `traceroute 192.168.2.2` από το PC1 στο PC2. Στο WAN1 (εκτελέσαμε `tcpdump -i em1` στο R1) παράγονται μηνύματα "ICMP time exceeded in-transit", επειδή η traceroute ως γνωστόν στέλνει UDP datagrams με σταδιακά αυξανόμενο TTL (οπότε κάποια αναπόφευκτα θα αποκτήσουν TTL=0 όσο προωθούνται προς τον προορισμό) και "ICMP 192.168.2.2 udp port 3344x unreachable" ($x = \{2, 3, 4\}$), αφού δεν υπάρχει κάποια εφαρμογή που "ακούει" στις συγκεκριμένες πόρτες στο PC2.

3.14

Κάνουμε `ping 172.17.17.1` από το PC2. Παρατηρούμε ότι εμφανίζεται το μήνυμα "No route to host".

3.15

Στο PC2 εκτελούμε `route del 192.168.1.0/24`.

3.16

Στο PC2 εκτελούμε `route add default 192.168.2.1`.

3.17

Στο PC2 εκτελούμε πάλι `ping 172.17.17.1`. Το ping είναι επιτυχές αυτή τη φορά.

3.18

Ο λόγος που το ping αρχικά δεν πετυχαίνει, ενώ στη συνέχεια πετυχαίνει, είναι επειδή την πρώτη φορά, στον πίνακα δρομολόγησης του PC2 δεν υπήρχε εγγραφή σχετική με τη διεύθυνση `172.17.17.1`. Τη δεύτερη φορά όμως, επειδή προστίθεται εγγραφή για την προεπιλεγμένη πύλη, το ICMP request στέλνεται στη διεπαφή του R2 στο LAN2 και αυτός το προωθεί με επιτυχία στον προορισμό.

Άσκηση 4: Ένα πιο πολύπλοκο δίκτυο με εναλλακτικές διαδρομές

4.1

Στο PC3 εκτελούμε:

```
ifconfig em0 up
ifconfig em0 192.168.2.3/24
```

4.2

Στο PC3 εκτελούμε:

```
route add -net 192.168.1.0/24 192.168.2.1
```

4.3

Για το R1 έχουμε:

```
em0: LAN1
ifconfig em0 192.168.1.1/24

em1: WAN1
ifconfig em1 172.17.17.1/30

em2: WAN2
ifconfig em2 172.17.17.5/30
```

4.4

Για το R2 έχουμε:

```
em0: WAN1
ifconfig em0 172.17.17.2/30

em1: WAN3
ifconfig em1 172.17.17.9/30

em2: LAN2
ifconfig em2 192.168.2.1/24
```

4.5

Για το R3 έχουμε:

```
em0: WAN2
ifconfig em0 172.17.17.6/30

em1: WAN3
ifconfig em1 172.17.17.10/30
```

4.6

Στον R1 εκτελούμε:

```
route add -net 192.168.2.0/24 172.17.17.2
```

4.7

Στον R2 εκτελούμε:

```
route add -net 192.168.1.0/24 172.17.17.1
```

4.8

Στον R3 εκτελούμε:

```
route add -net 192.168.1.0/24 172.17.17.5
route add -net 192.168.2.0/24 172.17.17.9
```

4.9

Στον R1 εκτελούμε:

```
route add -host 192.168.2.3 172.17.17.6
```

Εκτελούμε `netstat -rn`. Η σημαία που δηλώνει ότι πρόκειται για υπολογιστή είναι η "H".

4.10

Δοκιμάζουμε `traceroute 192.168.2.2` από το PC1 στο PC2. Βλέπουμε 3 βήματα.

4.11

Δοκιμάζουμε `ping -c 1 192.168.2.2` από το PC1 στο PC2. Το ICMP reply έχει TTL = 62, οπότε με βάση αυτή την τιμή βλέπουμε 3 βήματα, αφού:

- PC2: το πακέτο φεύγει με TTL = 64
- 1ος δρομολογητής, 1ο βήμα: μειώνει το TTL κατά 1, και το πακέτο φεύγει με TTL = 63
- 2ος δρομολογητής, 2ο βήμα: μειώνει το TTL κατά 1, και το πακέτο φεύγει με TTL = 62
- PC1, 3ο βήμα: το πακέτο φτάνει έχοντας TTL = 62.

4.12

Δοκιμάζουμε `traceroute 192.168.2.3` από το PC1 στο PC3. Βλέπουμε 4 βήματα.

4.13

Δοκιμάζουμε `ping -c 1 192.168.2.3` από το PC1 στο PC3. Το ICMP reply έχει TTL = 62, οπότε με βάση την τιμή αυτή βλέπουμε 3 βήματα. Η αιτιολόγηση είναι ίδια με αυτή του ερωτήματος 4.11.

4.14

Με βάση την έξοδο της εντολής traceroute, το ICMP echo request προς το PC3 ακολουθεί τη διαδρομή:

```
PC1 --> R1 --> R3 --> R2 --> PC3
```

4.15

Το ICMP echo reply προς το PC1 ακολουθεί τη διαδρομή:

```
PC3 --> R2 --> R1 --> PC1
```

Η διαφορά των δύο διαδρομών έγκειται στο ότι στην περίπτωση του ερωτήματος 4.14, ο R1 έχει δύο εγγραφές σχετικές με το PC3 στον πίνακα δρομολόγησης, μία που αναφέρεται στη διεύθυνση του PC3 και μόνο, και μία που αναφέρεται στο LAN2. Από τις δύο, ο R1 επιλέγει το ταίριασμα μέγιστου προθέματος, δηλαδή την εγγραφή που αναφέρεται στη διεύθυνση του PC3, και με βάση αυτή την εγγραφή προωθεί το πακέτο στον R3. Αντίθετα, στον πίνακα δρομολόγησης του R2 βρίσκεται η στατική εγγραφή που προσθέσαμε προηγουμένως, ώστε ο R2 να προωθεί πακέτα για το LAN1 μέσω του R1, η οποία είναι και η μόνη έγκυρη εγγραφή, οπότε είναι και αυτή που επιλέγεται.

4.16

Απενεργοποιούμε την διεπαφή του R1 στο WAN1 και ξεκινάμε τη ζητούμενη καταγραφή στο R2 με την εντολή `tcpdump -i em2`.

4.17

Δοκιμάζουμε `traceroute 192.168.2.2` από το PC1 στο PC2 και αφήνουμε να ολοκληρωθούν τουλάχιστον 3 βήματα. Δεν παρατηρούμε να φτάνουν ή να παράγονται πακέτα UDP στο PC2.

4.18

Δοκιμάζουμε `traceroute 192.168.2.3` από το PC1 στο PC3 και αφήνουμε να ολοκληρωθούν τουλάχιστον 4 βήματα. Παρατηρούμε ότι φτάνουν και παράγονται πακέτα UDP στο PC3.

4.19

Η ζητούμενη σύνταξη της εντολής `route` είναι:

```
R1:
route change -net 192.168.2.0/24 172.17.17.6

R2:
route change -net 192.168.1.0/24 172.17.17.10
```

Με χρήση της traceroute επιβεβαιώνουμε ότι υπάρχει επικοινωνία μετά την αλλαγή.

4.20

Στον R1 εκτελούμε `route show 192.168.2.2` και `route show 192.168.2.3`. Παρατηρούμε ότι στην πρώτη περίπτωση το πεδίο "destination" αναφέρεται στο υποδίκτυο LAN2 -γι' αυτόν τον λόγο υπάρχει και η επιπλέον πληροφορία για τη μάσκα υποδικτύου-, ενώ στην δεύτερη περίπτωση το πεδίο "destination" αναφέρεται συγκεκριμένα στον host PC3 -γι' αυτόν τον λόγο υπάρχει και η επιπλέον σημαία "HOST".

4.21

Επιλέγεται η εγγραφή που αναφέρεται στον host `192.168.2.3` (πεδίο destination: `192.168.2.3`), επειδή αυτή η εγγραφή προσφέρει το μέγιστο ταίριασμα προθέματος.

Άσκηση 5: Βρόχοι κατά τη δρομολόγηση

5.1

Τροποποιούμε τη ζητούμενη εγγραφή στον R3 με την εντολή:

```
route change -net 192.168.2.0/24 172.17.17.5
```

5.2

Εκτελούμε `ping -c 1 192.168.2.2` από το PC1 στο PC2. Το ping δεν είναι επιτυχές, καθώς εμφανίζεται μήνυμα "Time to live exceeded".

5.3

Το μήνυμα λάθους προέρχεται από τη διεπαφή του R3 στο WAN2, με διεύθυνση `172.17.17.6`.

5.4

Πρέπει να καταγράψουμε στον R1 και στο δίκτυο WAN2, επειδή εκεί έχει δημιουργηθεί ο βρόχος λόγω της στατικής εγγραφής που τροποποιήσαμε στο ερώτημα 5.1.

5.5

Πρέπει να εφαρμόσουμε το φίλτρο `"icmp[icmptype] == icmp-echo"`.

5.6

Ξεκινάμε στον R1 την καταγραφή που περιγράφηκε παραπάνω με την εντολή:

```
tcpdump -e -i em2 "icmp[icmptype] == icmp-echo"
```

Παρατηρούμε ότι λήφθηκαν 64 πακέτα, ενώ καταγράφηκαν 63 πακέτα. Το ένα πακέτο που λήφθηκε αλλά δεν καταγράφηκε είναι τύπου ICMP time exceeded in-transit. Έτσι έχουμε ότι εμφανίσθηκαν στο WAN2 63 πακέτα ICMP echo request. Τέλος, από το PC1 παράχθηκε μόνο ένα πακέτο ICMP echo request, όπως εξάλλου ορίσαμε και στην εντολή ping με την παράμετρο `-c 1`.

5.7

Ξεκινάμε τη ζητούμενη καταγραφή στο R1 με την εντολή `tcpdump -l -i em0 | tee capture1`. Αντίστοιχα στο R3 με την εντολή `tcpdump -l -i em0 | tee capture2`.

5.8

Εκτελούμε `traceroute -I -q 1 192.168.2.2` από το PC1 στο PC2. Εμφανίζονται 64 βήματα μέχρι να ολοκληρωθεί η εκτέλεση της εντολής. Η διαδρομή που καταγράφουμε είναι η:

```
PC1 --> R1 --> R3 --> R1 --> R3 --> ... --> R1 --> R3
```

δηλαδή ο βρόχος που έχει σχηματιστεί.

5.9

Σταματάμε τις καταγραφές. Με την εντολή `grep "ICMP echo request" capture1 | wc -l` στο R1 βρίσκουμε ότι στάλθηκαν 64 πακέτα ICMP echo request από το PC1.

5.10

Με την εντολή `grep "ICMP echo request" capture2 | wc -l` στο R3 βρίσκουμε ότι εμφανίστηκαν 2016 πακέτα ICMP echo request στο WAN2.

Η εξήγηση είναι η εξής:

Έστω ότι από το PC1 ξεκινάει ένα πακέτο με TTL $= T$. Αυτό θα σταλεί στον R1, και εκεί θα αποκτήσει TTL $= T - 1$. Ύστερα, θα μπει στον βρόχο R1 $\rightarrow$ R3 $\rightarrow$ R1..., μέχρι το TTL του να μηδενιστεί, οπότε και θα απορριφθεί από τον αντίστοιχο δρομολογητή. Επομένως, ένα πακέτο με TTL $= T$, θα εμφανιστεί συνολικά $T - 1$ φορές στο WAN2. Επειδή η εντολή traceroute στέλνει ένα πακέτο ανά βήμα (παράμετρος `-q 1`), και σε κάθε βήμα το TTL αυξάνεται κατά 1, ο συνολικός αριθμός πακέτων ICMP echo request που θα εμφανιστούν στο WAN2 είναι:

$$1 + 2 + 3 + ... + 63 = \frac{63 \cdot (63 + 1)}{2} = 2016 \text{ πακέτα.}$$

5.11

Με την εντολή `grep "ICMP time exceeded in-transit" capture2 | wc -l` βρίσκουμε ότι εμφανίστηκαν 32 πακέτα ICMP time exceeded στο WAN2. Αυτό συμβαίνει διότι από τα 64 πακέτα ICMP echo request που παράχθηκαν συνολικά από το PC1, τα μισά θα απορριφθούν από τον R1 (αυτά με αρχικό TTL $= 1, 3, 5, 7, ..., 63$), οπότε θα σταλεί μήνυμα ICMP time exceeded από τον R1 στο PC1 μέσω του LAN1, ενώ τα άλλα μισά θα απορριφθούν από τον R2 (αυτά με αρχικό TTL $= 2, 4, 6, 8, ..., 64$), οπότε θα σταλεί μήνυμα ICMP time exceeded από τον R2 στον R1 μέσω του WAN2. Επομένως, τα 32 μηνύματα ICMP time exceeded που καταγράφηκαν αφορούν τα μηνύματα ICMP echo request με αρχικό TTL άρτιο.

5.12

Μπορούμε να αποφύγουμε την αποθήκευση σε αρχείο ως εξής:

```
Question 5.9:
R1: tcpdump -i em0 "icmp[icmptype] == icmp-echo"

Question 5.10:
R3: tcpdump -i em0 "icmp[icmptype] == icmp-echo"

Question 5.11:
R3: tcpdump -i em0 "icmp[icmptype] == icmp-timexceed"
```

Σε κάθε περίπτωση η απάντηση που ψάχνουμε δίνεται αφού τερματίσουμε την εντολή tcpdump με Ctrl-C, στη γραμμή `"xxxx packets captured"`.

5.13

Τα πακέτα ICMP echo request που παράχθηκαν από την εντολή ping έχουν μήκος 64 bytes και αρχικό TTL = 64, ενώ τα πακέτα ICMP echo request που παράχθηκαν από την εντολή traceroute έχουν μήκος 28 bytes και αρχικό TTL που κυμαίνεται από 1 έως 64.

5.14

Τα πακέτα ICMP echo request δεν κυκλοφορούν αενάως στο δίκτυο, διότι κάθε φορά που το πακέτο περνά από έναν δρομολογητή, αυτός μειώνει το TTL του κατά 1, και όταν το TTL γίνει μηδέν, τότε ο δρομολογητής απορρίπτει το πακέτο και στέλνει μήνυμα ICMP time exceeded.

Άσκηση 6: Χωρισμός σε υποδίκτυα

6.1

LAN1, 120 υπολογιστές, πρέπει:

$$2^n - 2 > 120 \implies n = 7 \implies \text{Μήκος προθέματος LAN1} = 32 - 7 = 25.$$

Λαμβάνουμε υπόψιν μας το γεγονός ότι το μπλοκ διευθύνσεων 172.17.17.129-172.17.17.138 είναι δεσμευμένο από τους δρομολογητές.

```
129 dec = 1000 0001 bin
138 dec = 1000 1010 bin
```

βλέπουμε ότι μπορούμε να διαθέσουμε το μπλοκ 172.17.17.0-172.17.17.127, επομένως η διεύθυνση υποδικτύου του LAN1 είναι **172.17.17.0/25**.

6.2

LAN2, 60 υπολογιστές, πρέπει:

$$2^n - 2 > 60 \implies n = 6 \implies \text{Μήκος προθέματος LAN2} = 32 - 6 = 26.$$

βλέπουμε ότι μπορούμε να διαθέσουμε το μπλοκ 172.17.17.192-172.17.17.255, επομένως η διεύθυνση υποδικτύου του LAN2 είναι **172.17.17.192/26**.

6.3

LAN3, 30 υπολογιστές, πρέπει:

$$2^n - 2 > 30 \implies n = 5 \implies \text{Μήκος προθέματος LAN3} = 32 - 5 = 27.$$

βλέπουμε ότι μπορούμε να διαθέσουμε το μπλοκ 172.17.17.160-172.17.17.191, επομένως η διεύθυνση υποδικτύου του LAN3 είναι **172.17.17.160/27**.

6.4

Στο PC1 εκτελούμε:

```
ifconfig em0 172.17.17.1/25
```

Ενώ στον R1 εκτελούμε:

```
ifconfig em0 172.17.17.126/25
```

6.5

Στο PC4 εκτελούμε:

```
ifconfig em0 172.17.17.161/27
```

Ενώ στον R3 εκτελούμε:

```
ifconfig em0 172.17.17.190/27
```

6.6

Στον R2 εκτελούμε:

```
ifconfig em2 172.17.17.193/26
```

Στο PC2 εκτελούμε:

```
ifconfig em0 172.17.17.253/26
```

Ενώ στο PC3 εκτελούμε:

```
ifconfig em0 172.17.17.254/26
```

6.7

Εκτελούμε τις εντολές:

```
PC1: route add default 172.17.17.126
PC2: route add default 172.17.17.193
PC3: route add default 172.17.17.193
PC4: route add default 172.17.17.190
```

6.8

Στον R1 εκτελούμε:

```
route add -net 172.17.17.192/26 172.17.17.130
route add -net 172.17.17.160/27 172.17.17.130
```

6.9

Στον R2 εκτελούμε:

```
route add -net 172.17.17.0/25 172.17.17.137
route add -net 172.17.17.160/27 172.17.17.137
```

6.10

Στον R3 εκτελούμε:

```
route add -net 172.17.17.0/25 172.17.17.133
route add -net 172.17.17.192/26 172.17.17.133
```

6.11

Για να επιβεβαιώσουμε ότι υπάρχει επικοινωνία ανάμεσα σε όλα τα LAN, εκτελούμε:

```
PC1: ping -c 1 172.17.17.253
PC2: ping -c 1 172.17.17.161
PC3: ping -c 1 172.17.17.1
```

Και τα τρία ping είναι επιτυχή.

Άσκηση 7: Ταυτόσημες διευθύνσεις IP

Επιβεβαιώνουμε ότι τα PC1, PC2 και PC3 επικοινωνούν μεταξύ τους εκτελώντας:

```
PC1 <-> PC2: ping -c 1 172.17.17.253
PC1 <-> PC3: ping -c 1 172.17.17.254
PC2 <-> PC3: ping -c 1 172.17.17.254
```

7.1

Με `ifconfig em0` στα PC2,3 βρίσκουμε ότι:

```
PC2 MAC address == 08:00:27:10:0c:23
PC3 MAC address == 08:00:27:be:32:ef
```

7.2

Στο PC2 εκτελούμε `ifconfig em0 172.17.17.254/26`.

7.3

Εμφανίστηκε στο PC2 η ένδειξη:

```
PC kernel: arp: 08:00:27:be:32:ef is using my IP address 172.17.17.254 on em0!
```

7.4

Ναι, εμφανίστηκε στο PC3 η ένδειξη:

```
PC kernel: arp: 08:00:27:10:0c:23 is using my IP address 172.17.17.254 on em0!
```

7.5

Ναι, με `ifconfig em0` στο PC2 βλέπουμε ότι η IP διεύθυνση έχει οριστεί κανονικά. Το νόημα των μηνυμάτων λάθους είναι απλώς να ενημερώσει τους χρήστες των δύο μηχανημάτων ότι μοιράζονται την ίδια διεύθυνση IPv4.

7.6

Όχι, ο R2 δεν παραμένει ως προεπιλεγμένη πύλη στο PC2 (`netstat -rn`), επειδή εκτελέσαμε `ifconfig` και αλλάξαμε την IP διεύθυνση του PC2. Όταν γίνεται αυτό, η προεπιλεγμένη πύλη διαγράφεται από τον πίνακα δρομολόγησης.

7.7

Στο PC2 ορίζουμε και πάλι ως προεπιλεγμένη πύλη τον δρομολογητή R2 με την εντολή:

```
route add default 172.17.17.193
```

7.8

Καθαρίζουμε τους πίνακες ARP στα PC2, PC3 και R2 με την εντολή `arp -d -a`.

7.9

Ξεκινάμε τη ζητούμενη καταγραφή στον R2 με την εντολή `tcpdump -i em2 "arp"`.

7.10

Ξεκινάμε τις ζητούμενες καταγραφές εκτελώντας την εντολή `tcpdump -n -i em0 "tcp"` στα PC2 και PC3.

7.11

Από το PC1 εκτελούμε `ssh lab@172.17.17.254`. Εμφανίζεται ένδειξη λάθους:

```
ssh_exchange_identification: read: Connection reset by peer
```

7.12

Σταματάμε τις καταγραφές και επαναλαμβάνουμε το `ssh lab@172.17.17.254`. Αυτή τη φορά η προσπάθεια είναι επιτυχής.

7.13

Με την εντολή `arp -a` στον R2 καταγράφουμε:

```
PC2: 172.17.17.254 at 08:00:27:10:0c:23 on em2
```

Δεν υπάρχει εγγραφή για το PC3.

7.14

Στην καταγραφή πακέτων arp, το PC3 απάντησε πρώτο στο ARP Request του R2, ενώ το PC2 απάντησε δεύτερο.

7.15

Η διεύθυνση MAC που περιέχει ο πίνακας ARP του R2 ανήκει στο PC2.

7.16

Την δεύτερη φορά συνδεθήκαμε στο PC2, επειδή το PC2 απάντησε δεύτερο και έτσι η πιο πρόσφατη εγγραφή στον πίνακα ARP έδειχνε ότι η κοινή IPv4 διεύθυνση αντιστοιχεί στην φυσική διεύθυνση του PC2.

7.17

Άλλοι τρόποι με τους οποίους μπορούμε να καταλάβουμε σε ποιο μηχάνημα έχουμε συνδεθεί, πέρα από τον τρόπο που χρησιμοποιήσαμε παραπάνω (δηλαδή μέσω της καταγραφής των πακέτων arp):

- Μπορούμε να εκτελέσουμε `ifconfig em0` και να δούμε τη διεύθυνση MAC που αναγράφεται στην έξοδο της εντολής.
- Μπορούμε να εκτελέσουμε την εντολή `w` στα PC2, PC3. Στο ένα από τα δύο θα υπάρχει μόνο ο χρήστης `root`, ενώ στο άλλο θα φαίνεται ότι έχει συνδεθεί και ένας χρήστης `lab`.
- Μππορούμε να εκτελέσουμε την εντολή `netstat | grep ssh` στα PC2, PC3. Στο ένα από τα δύο θα υπάρχει μια σύνδεση ssh που θα περιλαμβάνει τη διεύθυνση IPv4 του PC1 και τη διεύθυνση IPv4 του μηχανήματος που έχει γίνει η σύνδεση.

7.18

Στον R2 λήφθηκε πρώτα το ARP reply από το PC3 και ύστερα από το PC2. Στις δύο καταγραφές TCP τεμαχίων παρατηρούμε ότι τα τεμάχια έχουν τις εξής σημαίες:

- Καταγραφή PC3:

  - SYN: από το PC1 στο PC3, με σκοπό να ξεκινήσει η τριπλή χειραψία
  - SYN, ACK: η απάντηση του PC3 προς το PC1, το δεύτερο τεμάχιο της χειραψίας
  - SYN, ACK: επανάληψη του παραπάνω
  - SYN, ACK: επανάληψη του παραπάνω
  - SYN, ACK: επανάληψη του παραπάνω, τελικά εγκαταλείπεται η προσπάθεια, αφού δεν λαμβάνεται ποτέ το πακέτο με σημαία ACK (τρίτο τεμάχιο της χειραψίας)

- Καταγραφή PC2 (πλέον ο R2 έχει λάβει ARP reply από το PC2, και έτσι τα επόμενα TCP τεμάχια με προορισμό τη διεύθυνση 172.17.17.254 στέλνονται στο PC2, και γι' αυτό δημιουργείται η "σύγχυση" που προκαλεί την αποτυχία του πρώτου SSH):
  - ACK: από το PC1 στο PC2, είναι το τρίτο τεμάχιο της χειραψίας που δεν πάει ποτέ στο PC3, που έστειλε το δεύτερο τεμάχιο της χειραψίας
  - RST: από το PC2 στο PC1, αφού ο PC2 δεν έχει εγκατεστημένη κάποια σύνδεση με το PC1 και έτσι στέλνει τεμάχιο με σκοπό τον τερματισμό της παραπάνω επικοινωνίας.
  - PSH, ACK: από το PC1 στο PC3, δείχνει ότι τα δεδομένα πρέπει να προωθηθούν στην εφαρμογή
  - RST: από το PC2 στο PC1 με σκοπό τον τερματισμό της σύνδεσης
  - RST: από το PC1 στο PC2 με σκοπό τον τερματισμό της σύνδεσης
  - RST: από το PC1 στο PC2 με σκοπό τον τερματισμό της σύνδεσης
  - RST: από το PC1 στο PC2 με σκοπό τον τερματισμό της σύνδεσης

Αυτή η "σύγχυση" επιλύεται τη δεύτερη φορά που συνδεόμαστε με SSH, αφού πλέον υπάρχει η εγγραφή στον πίνακα ARP του R2 που αντιστοιχεί την IPv4 διεύθυνση 172.17.17.254 με την MAC διεύθυνση του PC2, και η σύνδεση με SSH μεταξύ PC1 και PC2 είναι επιτυχής.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 6
Εισαγωγή στο Quagga και FRRouting (FRR)

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 05/04/2022

Άσκηση 1 (προετοιμασία): Γνωριμία με το περιβάλλον του FRR

1

Ορίζουμε την πρώτη δικτυακή διεπαφή σε NAT.

2

Εκτελούμε `dhclient em0`.

3

Εκτελούμε `ping www.google.com`.

4

Εκτελούμε `pkg update`.

5

Εκτελούμε `pkg install frr7`.

6

Προσθέτουμε τη γραμμή `kern.ipc.maxsockbuf=16777216`.

7

Εκτελούμε `chown frr:frr /usr/local/etc/frr/`.

8

Εκτελούμε `touch /usr/local/etc/frr/{vtysh.conf,zebra.conf,staticd.conf}`.

9

Εκτελούμε `chown frr:frr /usr/local/etc/frr/{vtysh.conf,zebra.conf,staticd.conf}`

10

Στο αρχείο εκκίνησης `/etc/rc.conf` προσθέτουμε τις εντολές:

```
hostname="R0"
gateway_enable="YES"
frr_enable="YES"
frr_daemons="zebra staticd"
```

11

Στο αρχείο `/etc/csh.cshrc` προσθέτουμε τη γραμμή:

```
setenv VTYSH_PAGER "more -EFX"
```

12

Διαγράφουμε το αρχείο `/etc/resolv.conf` που δημιούργησε ο πελάτης DHCP και κλείνουμε το μηχάνημα με την εντολή `poweroff`.

13

Ενεργοποιούμε τις άλλες 3 κάρτες δικτύωσης, ρυθμίζουμε όλες τις κάρτες σε εσωτερική δικτύωση, επιλέγουμε Deny στο Promiscuous Mode και δίνουμε στα εσωτερικά δίκτυα διαφορετικά ονόματα LAN1,2,3,4.

14

Κάνουμε επανεκκίνηση του FreeBSD και βεβαιωνόμαστε ότι η υπηρεσία `sshd` και το `frr` τρέχουν με τις εντολές `ps aux | grep sshd` και `ps aux | grep frr`.

15

Εκτελούμε `history -c`, ύστερα `poweroff` και κάνουμε Export Appliance, δημιουργώντας ένα αρχείο `frr.ova`.

16

Αποθηκεύουμε το αρχείο `frr.ova` για μελλοντική χρήση.

1.1

Εκτελούμε `telnet localhost 2601`. Εμφανίζεται μήνυμα λάθους:

```
Vty password is not set.
Connection closed by foreign host.
```

1.2

Η εντολή `vtysh`.

1.3

Βλέπουμε 22 εντολές.

1.4

Παρατηρούμε ότι γίνεται αυτόματη συμπλήρωση της εντολής traceroute.

1.5

Πατώντας το tab εμφανίζονται οι εντολές που ξεκινάνε με ”co”, ενώ πατώντας το `"?"` εμφανίζονται επιπλέον σύντομες περιγραφές γι’ αυτές τις εντολές.

1.6

Είναι `show version`.

1.7

Είναι `w t`.

1.8

Με την εντολή `show running-config`.

1.9

Με την εντολή `configure terminal`

1.10

Εκτελούμε `configure terminal` και ύστερα `hostname R1`. Παρατηρούμε ότι αλλάζει το prompt, από `R0(config)` σε `R1(config)`.

1.11

Εκτελούμε `password ntua`.

1.12

Πρέπει να δώσουμε την εντολή ”exit” 2 φορές.

1.13

Δοκιμάζουμε πάλι `telnet localhost 2601`. Παρατηρούμε ότι αυτή τη φορά αντί να αποτυγχάνει το telnet, μας ζητείται να εισάγουμε το password που ορίσαμε.

1.14

Βρισκόμαστε σε επίπεδο User EXEC, όπως δείχνει και το prompt `R0>` αντί `R0#`.

1.15

Βλέπουμε 9 εντολές.

1.16

Σε επίπεδο User EXEC βλέπουμε 13 λιγότερες εντολές, όπως είναι λογικό, αφού το επίπεδο User EXEC έχει λιγότερες ελευθερίες από το επίπεδο Privileged EXEC.

1.17

Με την εντολή `show interface`.

1.18

Εκτελούμε `show ip forwarding`. Η προώθηση είναι ενεργοποιημένη.

1.19

Εκτελούμε `show ip route`.

1.20

Όχι, διότι η εντολή `show running-config / write terminal` ανήκει στο επίπεδο Privileged EXEC.

1.21

Με την εντολή `enable`.

1.22

Ναι, μπορούμε να δούμε την τρέχουσα παραμετροποίηση του FRR με `write terminal`. Το συνθηματικό εμφανίζεται στην έξοδο της εντολής.

1.23

Εκτελούμε `list`.

1.24

Σε Global Configuration Mode (δηλαδή αφού έχουμε εκτελέσει `configure terminal` σε λειτουργία Privileged EXEC), εκτελούμε `enable password ntua`.

1.25

Με την εντολή `service password-encryption`.

1.26

Θα προτιμούσαμε την επιλογή του ssh, καθώς ως γνωστόν το ssh αποτελεί πολύ ασφαλέστερη επιλογή από το telnet, για λόγους που έχουν συζητηθεί στο μάθημα των Δικτύων Υπολογιστών (το telnet στέλνει τα πακέτα χωρίς κανενός είδους κρυπτογράφηση -ακόμα και τους κωδικούς-, ενώ το ssh χρησιμοποιεί κρυπτογραφημένη επικοινωνία).

Άσκηση 2: Δρομολόγηση σε ένα βήμα

2.1

Με τις εντολές:

```
PC1: ifconfig em0 192.168.1.2/24
PC2: ifconfig em0 192.168.2.2/24
```

2.2

Εκτελούμε στον R1:

```
vtysh
configure terminal
hostname R1
interface em0
ip address 192.168.1.1/24
exit
interface em1
ip address 192.168.2.1/24
exit
```

2.3

Στον R1 εκτελούμε `do show interface`. Οι διευθύνσεις ορίστηκαν κανονικά.

2.4

Εκτελούμε `do show ip forwarding`. Αν δεν είναι ενεργοποιημένη, την ενεργοποιούμε με `ip forwarding`. Στη δική μας περίπτωση είναι ενεργοποιημένη.

2.5

Στο PC1 εκτελούμε `route add -net 192.168.2.0/24 192.168.1.1`.

2.6

Στο PC2 εκτελούμε `route add -net 192.168.1.0/24 192.168.2.1`.

2.7

Εκτελούμε `ping 192.168.2.2` από το PC1 στο PC2. Οι δύο υπολογιστές επικοινωνούν.

2.8

Στον R1 μέσω vtysh εκτελούμε:

```
interface em0
ip address 192.168.1.200/24
do show interface em0
```

Παρατηρούμε ότι η διεύθυνση `192.168.1.200/24` έχει οριστεί ως secondary, ενώ η προηγούμενη `192.168.1.1/24` δεν έχει σβηστεί.

2.9

Εκτελούμε `ifconfig em0` από τη γραμμή εντολών του R1. Βλέπουμε ότι η `em0` έχει όντως δύο διευθύνσεις IPv4, οι οποίες συμφωνούν με το προηγούμενο ερώτημα, ωστόσο σε αυτή την περίπτωση δεν αναγράφεται κάποια διεύθυνση ως secondary.

2.10

Στον R2 εκτελούμε:

```
no ip address 192.168.1.200/24
```

και ύστερα με `ifconfig em0` επιβεβαιώνουμε ότι η ζητούμενη διεύθυνση διαγράφτηκε.

2.11

Με την εντολή `do write memory` ή `do write file`.

2.12

Από την έξοδο της παραπάνω εντολής βλέπουμε ότι ενημερώνονται τα αρχεία `zebra.conf` και `staticd.conf` του φακέλου `/usr/local/etc/frr/`.

Άσκηση 3: Δρομολόγηση σε περισσότερα βήματα

3.1

Εκτελούμε:

```
PC1:
ifconfig em0 192.168.1.2/24
route add -net 192.168.2.0/24 192.168.1.1

PC2:
ifconfig em0 192.168.2.2/24
route add -net 192.168.1.0/24 192.168.2.1
```

3.2

Εκτελούμε στον R1:

```
vtysh
configure terminal
hostname R1
interface em0
ip address 192.168.1.1/24
interface em1
ip address 172.17.17.1/30
```

3.3

Εκτελούμε στον R2:

```
vtysh
configure terminal
hostname R2
interface em0
ip address 172.17.17.2/30
interface em1
ip address 192.168.2.1/24
```

3.4

Εκτελούμε στον R1 μέσω `vtysh`:

```
configure terminal
ip route 192.168.2.0/24 172.17.17.2
```

3.5

Εκτελούμε στον R2 μέσω `vtysh`:

```
configure terminal
ip route 192.168.1.0/24 172.17.17.1
```

3.6

Εκτελούμε `telnet 192.168.1.1 2601`. Η σύνδεση αποτυγχάνει με μήνυμα:

```
Vty password is not set.
Connection closed by foreign host.
```

Για να γίνει δυνατή η παραπάνω σύνδεση πρέπει να ορίσουμε κωδικό στον R1 για την πιστοποίηση απομακρυσμένων συνδέσεων (πχ `password ntua`).

3.7

Εκτελούμε ? και βλέπουμε ότι δεν υπάρχει διαθέσιμη εντολή σχετική με σύνδεση στην υπηρεσία zebra άλλου μηχανήματος, άρα δεν μπορούμε να συνδεθούμε στην υπηρεσία zebra του R2 μέσω του path PC1→R1→R2.

3.8

Θα κάναμε `telnet 192.168.2.1 2601` διότι το PC1 έχει εγγραφή στον πίνακα δρομολόγησης μόνο για το υποδίκτυο `192.168.2.0/24` και όχι για το `172.17.17.0/30`.

3.9

Με την εντολή `who`. Δεν εμφανίζεται ο χρήστης που έχει εισέλθει τοπικά μέσω `vtysh` στον R2, μόνο ο PC2 (`192.168.2.2`).

3.10

Όχι, διότι οι εντολές `ping` και `traceroute` δεν εμφανίζονται όταν πατάμε το πλήκτρο `"?"`. Αντίθετα, από την τοπική σύνδεση μπορούμε.

3.11

Εκτελούμε `traceroute 192.168.1.2` από τον R2 και `traceroute 192.168.2.2` από τον R1. Τα traceroute δεν ολοκληρώνονται διότι δεν έχει τεθεί στον πίνακα δρομολόγησης των PC1, PC2 εγγραφή το δίκτυο `172.17.17.0/30`, οπότε δεν μπορούν τα PC1,2 να απαντήσουν με ”ICMP udp port unreachable”.

3.12

Μπορούμε να κάνουμε:

```
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

Εκτελούμε:

```
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

Εκτελούμε στον R1:

```
cli
configure terminal
hostname R1
interface em0
ip address 192.168.1.1/24
interface em1
ip address 172.17.17.1/30
interface em2
ip address 172.17.17.5/30
```

4.3

Εκτελούμε στον R1 μέσω cli:

```
ip route 192.168.2.0/24 172.17.17.2
```

4.4

Εκτελούμε στον R1 μέσω cli:

```
do show ip route
```

Εμφανίζονται:

- 127.0.0.0/8 → lo0
- 172.17.17.0/30 → em1
- 172.17.17.4/30 → em2
- 192.168.1.0/24 → em0
- 192.168.2.0/24 → em1

4.5

Με τη φράση `is directly connected` στην έξοδο της εντολής.

4.6

Μέσω του flag ”S” στην αρχή της γραμμής:

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

4.7

Εκτελούμε `netstat -rn`. Οι εγγραφές συμφωνούν, απλώς η `netstat` έχει κάποιες επιπλέον εγγραφές που αφορούν την `lo0`.

4.8

Έχουν δηλωθεί οι σημαίες ”UG1”.

- ”U”: σημαίνει ότι η διαδρομή είναι ενεργή (up).
- ”G”: σημαίνει ότι ο προορισμός είναι πύλη, που θα αποφασίσει για το πώς θα προωθήσει τα πακέτα περαιτέρω.
- ”1”: μια σημαία που εξαρτάται από το πρωτόκολλο δρομολόγησης (Protocol specific routing flag #1)

4.9

Εκτελούμε στον R2 μέσω cli:

```
configure terminal
hostname R2
interface em0
ip address 172.17.17.2/30
interface em1
ip address 172.17.17.9/30
interface em2
ip address 192.168.2.1/24
```

4.10

Εκτελούμε στον R2 μέσω cli:

```
ip route 192.168.1.0/24 172.17.17.1
```

4.11

Εκτελούμε στον R3 μέσω cli:

```
hostname R3
interface em0
ip address 172.17.17.6/30
interface em1
ip address 172.17.17.10/30
```

4.12

Εκτελούμε στον R3 μέσω cli:

```
ip route 192.168.1.0/24 172.17.17.5
ip route 192.168.2.0/24 172.17.17.9
```

4.13

Εκτελούμε στον R3 μέσω cli:

```
do show ip forwarding
```

Εμφανίζεται μήνυμα ”IP forwarding is **on**”.

4.14

Στο PC1 εκτελούμε `traceroute 192.168.2.2`. Τα πακέτα ακολουθούν τη διαδρομή:

PC1 $\rightarrow$ R1 $\rightarrow$ R2 $\rightarrow$ PC2.

Άσκηση 5: Σφάλμα καλωδίου και αυτόματη αλλαγή στη δρομολόγηση

5.1

Μέσω cli στον R1:

```
ip route 192.168.2.0/24 172.17.17.6 2
```

5.2

Δώσαμε την τιμή 2, για να είναι μεγαλύτερη (και άρα λιγότερο προτιμητέα) από την προηγούμενη σχετική στατική εγγραφή, που έχει απόσταση 1.

5.3

Μέσω cli στον R2:

```
ip route 192.168.1.0/24 172.17.17.10 2
```

5.4

Στους R1 και R2 μέσω cli:

```
do show ip route
```

R1 σχετικά με το LAN2:

```
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 σχετικά με το LAN1:

```
S   192.168.1.0/24 [2/0] via 172.17.17.10, em1
S>* 192.168.1.0/24 [1/0] via 172.17.17.1, em0
```

5.5

Στον R1 είναι ενεργοποιημένη η διαδρομή μέσω του R2 (διεπαφή `172.17.17.2`). Αυτό φαίνεται από το σύμβολο `">"` στην αντίστοιχη εγγραφή του ερωτήματος `5.4`.

5.6

Η διαχειριστική απόσταση είναι ο πρώτος αριθμός μέσα σε αγκύλες (πχ `[2/0]` → distance = 2).

5.7

Η διαδρομή μέσω του R1 (διεπαφή `172.17.17.1`).

5.8

Εκτελούμε μέσω cli:

```
R1:
interface em1
link-detect

R2:
interface em0
link-detect
```

5.9

Κάνουμε δεξί κλικ στο εικονίδιο δικτύου του εικονικού μηχανήματος R1 και ύστερα αποεπιλογή του ”Connect Network Adapter 2”.

5.10

Η διαδρομή μέσω του R3 (διεπαφή `172.17.17.6`).

5.11

Ναι υπάρχει, είναι `inactive`.

5.12

Στον R1 εκτελούμε `netstat -rn` και παρατηρούμε ότι έχει γίνει η αντίστοιχη αλλαγή στον πίνακα δρομολόγησης:

```
192.168.2.0/24 172.17.17.6
```

5.13

Στον R2 εκτελούμε `do show ip route` και βλέπουμε ότι είναι ενεργοποιημένη η διαδρομή μέσω του R1 (`172.17.17.1`), διότι το αντίστοιχο καλώδιο δεν έχει αποσυνδεθεί στον R2 και έτσι αυτός δεν αντιλαμβάνεται την διακοπή της ζεύξης που εκτελέσαμε προηγουμένως.

5.14

Αποσυνδέουμε το ζητούμενο καλώδιο. Εκτελούμε `do show ip route` και διαπιστώνουμε ότι έγινε σωστά η μετάβαση στην εναλλακτική διαδρομή μέσω του R3 (`172.17.17.10`).

5.15

Στον PC1 εκτελούμε `traceroute 192.168.2.2`, και βλέπουμε ότι στα βήματα 2 και 3 της διαδρομής λαμβάνουμε απαντήσεις από τον R3 (`172.17.17.6`) και τον R2 (`172.17.17.9`) αντίστοιχα, επιβεβαιώνοντας όσα συζητήθηκαν παραπάνω.

5.16

Στον PC2 εκτελούμε `ssh lab@192.168.1.2` και ύστερα επανασυνδέουμε τα δύο καλώδια που αποσυνδέσαμε προηγουμένως. Παρατηρούμε ότι η σύνδεση SSH δεν χάνεται.

5.17

Εκτελώντας `do show ip route` στους R1, R2, διαπιστώνουμε ότι πλέον προτιμώνται οι αρχικές διαδρομές, δηλαδή:

- LAN1 → LAN2: R1 → R2
- LAN2 → LAN1: R2 → R1

Για να το εξακριβώσουμε, εκτελούμε `traceroute 192.168.2.2` από το PC1 ή `traceroute 192.168.1.2` από το PC2.

Άσκηση 6: Διευθύνσεις διαχείρισης (loopback)

6.1

Εκτελούμε μέσω cli:

```
R1:
interface lo0
ip address 172.22.22.1/32

R2:
interface lo0
ip address 172.22.22.2/32

R3:
interface lo0
ip address 172.22.22.3/32
```

6.2

Εκτελούμε "`ping 172.22.22.x`" $(x = 1, 2, 3)$ από τα PC1 και PC2. Μόνο τα ping από PC1 προς `172.22.22.1` και PC2 προς `172.22.22.2` επιτυγχάνουν. Τα υπόλοιπα ping όπως είναι αναμενόμενο αποτυγχάνουν, αφού δεν υπάρχουν αντίστοιχες εγγραφές στους πίνακες δρομολόγησης των R1,2,3.

6.3

Στον R1 μέσω cli:

```
ip route 172.22.22.2/32 172.17.17.2
ip route 172.22.22.3/32 172.17.17.6
```

6.4

Στον R2 μέσω cli:

```
ip route 172.22.22.1/32 172.17.17.1
ip route 172.22.22.3/32 172.17.17.10
```

6.5

Στον R3 μέσω cli:

```
ip route 172.22.22.1/32 172.17.17.5
ip route 172.22.22.2/32 172.17.17.9
```

6.6

Ξαναεκτελούμε `ping 172.22.22.x` από τα PC1,PC2. Αυτή τη φορά όλα τα ping επιτυγχάνουν.

6.7

Εκτελούμε:

```
PC1: tcpdump -i em0
PC2: tcpdump -i em0
```

Αν εκτελέσουμε `ping 192.168.1.2` και `ping 192.168.2.2` από την κονσόλα του R3:

```
ICMP source address for ping R3 --> PC1 == 172.17.17.6
ICMP source address for ping R3 --> PC2 == 172.17.17.10
```

6.8

Θα εκτελέσουμε `ping -S 172.22.22.3 192.168.1.2` και `ping -S 172.22.22.3 192.168.2.2`

6.9

Η δυσκολία έγκειται στο ότι κάθε φορά που δημιουργείται ένα πρόβλημα δρομολόγησης στα PC, πρέπει να ελέγξουμε όλον τον πίνακα δρομολόγησης για να διαπιστώσουμε μήπως το πρόβλημα οφείλεται στην απουσία κάποιας στατικής εγγραφής (που ενδεχομένως ξεχάσαμε να ορίσουμε ή διαγράφηκε με τον επαναπροσδιορισμό της διεύθυνσης IP του PC). Αντίθετα, αν έχουμε ορίσει default διαδρομή, αρκεί να κάνουμε ping σε αυτή για να καταλάβουμε αν το πρόβλημα δημιουργείται εξαιτίας του PC ή κάποιου άλλου κόμβου.

6.10

Όλα τα ping θα ήταν επιτυχή, εκτός από PC1 → loopback R2 και PC2 → loopback R1, διότι παρόλο που θα μπορούσαν τα πακέτα να προωθηθούν μέσω του R3, δεν έχει οριστεί διαδρομή στον πίνακα δρομολόγησης των R1,2 για τις loopback διευθύνσεις αυτές μέσω του R3.

6.11

Στον R1 μέσω cli:

```
ip route 172.22.22.2/32 172.17.17.6 2
ip route 172.22.22.3/32 172.17.17.2 2
```

6.12

Στον R2 μέσω cli:

```
ip route 172.22.22.1/32 172.17.17.10 2
ip route 172.22.22.3/32 172.17.17.1 2
```

6.13

Στον R3 μέσω cli:

```
ip route 172.22.22.1/32 172.17.17.9 2
ip route 172.22.22.2/32 172.17.17.5 2
```

6.14

Στον R1 μέσω cli:

```
do show ip route
```

Επιλεγμένη διαδρομή είναι αυτή μέσω του R2 `172.17.17.2`.

6.15

Προσομοιώνουμε τη βλάβη στο WAN1 και παρατηρούμε ότι οι διαδρομές που διέρχονται μέσω του WAN1 έχουν δηλωθεί ως `inactive`.

6.16

Αυτή τη φορά δεν εμφανίζονται inactive διαδρομές, αφού δεν έχουμε κάνει enable το `link-detect` στις διεπαφές του WAN2.

Άσκηση 7: Ένα εταιρικό δίκτυο

Υλοποίηση συνδεσμολογίας

Αντιστοίχιση Network Adapters σε LAN

- PC1:
  - Network Adapter 1 (em0): LAN1
- PC2:
  - Network Adapter 1 (em0): LAN2
- R1:
  - Network Adapter 1 (em0): LAN1
  - Network Adapter 2 (em1): WAN1
  - Network Adapter 3 (em2): WAN3
- R2:
  - Network Adapter 1 (em0): LAN2
  - Network Adapter 2 (em1): WAN2
  - Network Adapter 3 (em2): WAN4
- C1:
  - Network Adapter 1 (em0): CORE
  - Network Adapter 2 (em1): WAN1
  - Network Adapter 3 (em2): WAN2
- C2:
  - Network Adapter 1 (em0): CORE
  - Network Adapter 2 (em1): WAN3
  - Network Adapter 3 (em2): WAN4

Ορισμός ονομάτων και διευθύνσεων IP μέσω cli στους δρομολογητές

- R1:

```
cli
configure terminal

hostname R1

interface em0
ip address 192.168.1.1/24

interface em1
ip address 10.0.1.1/30

interface em2
ip address 10.0.1.5/30

interface lo0
ip address 172.22.1.1/32
```

- R2:

```
cli
configure terminal

hostname R2

interface em0
ip address 192.168.2.1/24

interface em1
ip address 10.0.2.1/30

interface em2
ip address 10.0.2.5/30

interface lo0
ip address 172.22.2.1/32
```

- C1:

```
cli
configure terminal

hostname C1

interface em0
```

```
ip address 10.0.0.1/30

interface em1
ip address 10.0.1.2/30

interface em2
ip address 10.0.2.2/30

interface lo0
ip address 172.22.1.2/32
```

- C2:

```
cli
configure terminal

hostname C2

interface em0
ip address 10.0.0.2/30

interface em1
ip address 10.0.1.6/30

interface em2
ip address 10.0.2.6/30

interface lo0
ip address 172.22.2.2/32
```

Ενεργοποίηση link-detect σε όλες τις διεπαφές WAN

- R1:

```
interface em1
link-detect

interface em2
link-detect
```

- R2:

```
interface em1
link-detect

interface em2
link-detect
```

- C1:

```
interface em1
link-detect

interface em2
link-detect
```

- C2:

```
interface em1
link-detect

interface em2
link-detect
```

Ορισμός διευθύνσεων IP και προεπιλεγμένης πύλης στα PC

- 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
```

7.1

Στον C1, μέσω cli:

```
ip route 192.168.1.0/24 10.0.1.1
ip route 192.168.1.0/24 10.0.0.2  2
ip route 192.168.2.0/24 10.0.2.1
ip route 192.168.2.0/24 10.0.0.2  2
```

7.2

Στον C2, μέσω cli:

```
ip route 192.168.1.0/24 10.0.1.5
ip route 192.168.1.0/24 10.0.0.1  2
ip route 192.168.2.0/24 10.0.2.5
ip route 192.168.2.0/24 10.0.0.1  2
```

7.3

Στον R1, μέσω cli:

```
ip route 192.168.2.0/24 10.0.1.2
ip route 192.168.2.0/24 10.0.1.6  2
```

7.4

Στον R2, μέσω cli:

```
ip route 192.168.1.0/24 10.0.2.2
ip route 192.168.1.0/24 10.0.2.6  2
```

7.5

Στο PC1 εκτελούμε `ping -c 1 192.168.2.2`. Το ping είναι επιτυχές.

7.6

Αποσυνδέουμε τη ζεύξη WAN2 και εκτελούμε `ping -c 1 192.168.2.2` το οποίο είναι επιτυχές, άρα το PC1 εξακολουθεί να επικοινωνεί με το PC2.

7.7

- PC1 → PC2:
  - PC1 → R1→ C1 → C2 → R2 → PC2
- PC2 → PC1:
  - PC2 → R2 → C2 → R1 → PC1

7.8

Στο PC1 εκτελούμε `traceroute 192.168.2.2`. Οι διευθύνσεις IP της διαδρομής είναι:

```
1   192.168.1.1 (R1)
2   10.0.1.2 (C1)
3   10.0.1.6 (C2)
4   10.0.2.5 (R2)
5   192.168.2.2 (PC2)
```

Παρατηρούμε ότι στην περίπτωση του C2, αντί να εμφανιστεί η διεύθυνση `10.0.0.2` από την οποία διέρχονται τα IP πακέτα του PC1, εμφανίζεται η `10.0.1.6`. Αυτό συμβαίνει διότι η traceroute καταγράφει τις διεπαφές που απαντάνε στον PC1 με μηνύματα ”Time to live exceeded”. Επειδή λοιπόν η απάντηση του C2 δρομολογείται από άλλη διεπαφή απ᾽ ό,τι τα πακέτα του PC1 -εξαιτίας της προτιμώμενης διαδρομής που έχει διαχειρστική απόσταση 1-, έχουμε αυτή τη συμπεριφορά.

7.9

Στον PC2 εκτελούμε `traceroute 192.168.1.2`. Οι διευθύνσεις IP της διαδρομής είναι:

```
1   192.168.2.1 (R2)
2   10.0.2.6 (C2)
3   10.0.1.1 (R1)
4   192.168.1.2 (PC1)
```

Παρατηρούμε ότι στην περίπτωση του R1, δεν εμφανίζεται η διεύθυνση `10.0.1.5`, αλλά η `10.0.1.1`, καθώς η απάντηση του R1 δρομολογείται μέσω του C1, επειδή η διαδρομή αυτή έχει μικρότερη απόσταση, και άρα μεγαλύτερη προτεραιότητα.

7.10

Προσομοιώνουμε βλάβη και στο WAN3, και στο PC1 εκτελούμε `traceroute 192.168.2.2`. Η έξοδος της εντολής είναι:

```
1   192.168.1.1 (R1)
2   10.0.1.2 (C1)
3   10.0.0.2 (C2)
4   10.0.2.5 (R2)
5   192.168.2.2 (PC2)
```

Ακολουθείται λοιπόν η διαδρομή: R1 $\rightarrow$ C1 $\rightarrow$ C2 $\rightarrow$ R2 $\rightarrow$ PC2.

7.11

Το ping θα ακολουθήσει τη διαδρομή:

PC1 $\rightarrow$ R1 $\rightarrow$ C1 $\rightarrow$ C2 $\rightarrow$ C1 $\rightarrow$ C2 $\rightarrow$ ... (βρόχος C1-C2)

Αυτό συμβαίνει επειδή ο C1 δεν μπορεί να προωθήσει το πακέτο μέσω του WAN2, και ο C2 δεν μπορεί να προωθήσει το πακέτο μέσω του WAN4. Έτσι, το πακέτο απλά θα επανεκπεμφθεί πολλές φορές μέχρι να μηδενιστεί το TTL του, και δεν θα φτάσει ποτέ στον προορισμό. Αντίθετα, ο PC1 θα λάβει μήνυμα "Time to live exceeded".

7.12

Το σημαντικότερο μειονέκτημα μιας τέτοιας τοπολογίας εταιρικού δικτύου είναι το διαχειριστικό overhead για τυχόν μελλοντικές αλλαγές της τοπολογίας, όπως για παράδειγμα η προσθήκη ενός επιπλέον δρομολογητή (έστω R3). Σε μια τέτοια περίπτωση, θα χρειαζόταν να ορίσουμε χειροκίνητα τον πίνακα προώθησης του R3, τυχόν διαδρομές από άλλα μηχανήματα μέσω αυτού, να ελέγξουμε μήπως υπάρχουν καλύτερες διαδρομές από τις ήδη υπάρχουσες λόγω της παρουσίας του R3, να ορίσουμε εναλλακτικές διαδρομές κλπ. Όπως είναι φανερό, κάτι τέτοιο κλιμακώνεται πολύ δύσκολα, ειδικά σε πραγματικές συνθήκες, όπου τα δίκτυα είναι πολύ μεγαλύτερα.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 7
Δυναμική δρομολόγηση RIP

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 12/04/2022

Άσκηση 1: Εισαγωγή στο RIP

1

Εκτελούμε `service frr stop`.

2

Εκτελούμε `touch /usr/local/etc/frr/ripd.conf`.

3

Εκτελούμε `chown frr:frr /usr/local/etc/frr/ripd.conf`.

4

Αλλάζουμε τη ζητούμενη γραμμή του `/etc/rc.conf` σε `frr_daemons="zebra staticd ripd"`.

5

Εκτελούμε `service frr start`.

6

Κλείνουμε το μηχάνημα και το κάνουμε export σε αρχείο με όνομα `RIP.ova`.

7

Αποθηκεύουμε το αρχείο `.ova` για μελλοντική χρήση.

1.1

Μέσω vtysh στο PC1:

```
configure terminal

hostname PC1

interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1
```

1.2

Μέσω vtysh στο PC2:

```
configure terminal

hostname PC2

interface em0
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1
```

1.3

Μέσω cli στο R1:

```
configure terminal

hostname R1

interface em0
ip address 192.168.1.1/24

interface em1
ip address 172.17.17.1/30
```

1.4

Μέσω cli στο R1:

```
do show ip route
```

Δεν βλέπουμε κάποια εγγραφή που να αρχίζει με ”S”, άρα δεν έχουμε στατικές εγγραφές.

1.5

Στον R1 σε global configuration mode γράφουμε `router` και μετά πατάμε `"?"`. Εμφανίζονται 7 διαθέσιμα πρωτόκολλα:

```
babel
bgp
isis
ospf
ospf6
rip
ripng
```

1.6

Στον R1 μέσω cli:

```
router rip
```

1.7

Στον R1 πατάμε `"?"`. Οι διαθέσιμες εντολές είναι 18.

1.8

Με την εντολή `version 2`.

1.9

Στον R1 εκτελούμε:

```
network 192.168.1.0/24
```

1.10

Στον R1 εκτελούμε:

```
network 172.17.17.0/30
```

1.11

Στον R1 εκτελούμε:

```
exit
do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα δρομολόγησης του R1.

1.12

Επαναλαμβάνουμε για τον R2:

```
1.3

configure terminal

hostname R2

interface em0
ip address 172.17.17.2/30

interface em1
ip address 192.168.2.1/24

1.4

do show ip route    # OK, no static records found

1.5

in global configuration mode:

router ?    # 7 available protocols

1.6

router rip

1.7

?    # 18 available commands

1.8

version 2

1.9

network 192.168.2.0/24

1.10

network 172.17.17.0/30
```

Στο PC1 εκτελούμε `ping 192.168.2.2`. Τα PC1 και PC2 επικοινωνούν μεταξύ τους.

1.13

Με την εντολή `do show ip route`. Βλέπουμε ότι έχει προστεθεί μία εγγραφή:

```
R>* 192.168.1.0/24 [120/2] via 172.17.17.1, em0, 00:05:08
```

1.14

Επιβεβαιώνουμε ότι τα PC επικοινωνούν με `ping 192.168.2.2` από το PC1. Στον R1 εκτελούμε `show ip rip`. Υπάρχουν εγγραφές για τα δίκτυα `172.17.17.0/30`, `192.168.1.0/24` και `192.168.2.0/24`.

1.15

Το νόημα είναι ότι σαν επόμενο βήμα επιλέγεται το ίδιο το μηχάνημα (δηλαδή ο R1), γιατί ο προορισμός είναι απευθείας συνδεδεμένος με αυτόν.

1.16

Η πηγή πληροφόρησης φαίνεται στο πεδίο "From". Έτσι οι πηγές πληροφόρησης είναι:

- `172.17.17.0/30` → `0.0.0.0`: από το ίδιο το μηχάνημα (self), Metric 1
- `192.168.1.0/24` → `0.0.0.0`: από το ίδιο το μηχάνημα (self), Metric 1
- `192.168.2.0/24` → `172.17.17.2`: από τον R2 (`172.17.17.2`), Metric 2

Το Metric δείχνει πόσα βήματα μακριά βρίσκεται ο προορισμός.

1.17

Στον R2 εκτελούμε `show ip route`. Εμφανίζονται 4 εγγραφές.

1.18

Ξεχωρίζουν επειδή στην αρχή υπάρχει ο χαρακτήρας `"R"`.

1.19

Δηλώνονται με το σύμβολο `">"`.

1.20

Δηλώνονται με το σύμβολο `"*"`.

1.21

Είναι 120, κάτι που φαίνεται στην αντίστοιχη εγγραφή μέσα σε αγκύλες: `[120/2]`, όπου το πρώτο νούμερο είναι η διαχειριστική απόσταση και το δεύτερο είναι το μήκος της διαδρομής, δηλαδή 2.

1.22

Με την εντολή `do show ip rip status`. Αποστέλλονται ενημερώσεις κάθε 30 δευτερόλεπτα, με απόκλιση $\pm 50\%$.

1.23

Είναι ενεργοποιημένο στις διεπαφές `em0` και `em1`, και στη δρομολόγηση μετέχουν τα δίκτυα `172.17.17.0/30` και `192.168.1.0/24`.

1.24

Λαμβάνει πληροφορία από τον R2 (`172.17.17.2`). Ο χρόνος τελευταίας ενημέρωσης δηλώνει πόσος χρόνος έχει περάσει από την τελευταία φορά που έλαβε ενημέρωση ο R1 από τον R2.

1.25

Εκτελούμε `do show ip rip`.

Το πεδίο "Time" αντιστοιχεί στον χρόνο "timeout" που αν λήξει, η διαδρομή παύει να ισχύει, όμως δεν αφαιρείται ακόμα από τον πίνακα δρομολόγησης (προεπιλεγμένη τιμή 180 sec). Κάθε φορά που λαμβάνεται μία ενημέρωση, το "Last Update" μηδενίζεται και το "Time" τίθεται στην προεπιλεγμένη τιμή. Η σχέση που τα συνδέει είναι: $\text{Time} = 180\ \text{sec} - \text{Last Update}$.

1.26

Στον R1 εκτελούμε:

```
exit
netstat -rn
```

Βλέπουμε ότι υπάρχει μια εγγραφή με σημαίες "UG1":

```
Destination      Gateway       Flags
192.168.2.0/24   172.17.17.2   UG1
```

Επειδή η σημαία "1" αντιστοιχεί σε "Protocol specific routing flag #1", μπορούμε να καταλάβουμε ότι είναι δυναμική αφού είναι ορισμένη από το RIP.

Άσκηση 2: Λειτουργία του RIP

2.1

Στον R1 εκτελούμε `tcpdump -vni em0` και περιμένουμε τουλάχιστον ένα λεπτό.

2.2

Βλέπουμε τόσο μηνύματα RIP Request όσο και RIP Response.

2.3

- Πηγή: `192.168.1.1`, θύρα `520`
- Προορισμός: `224.0.0.9`, θύρα `520`

Η διεύθυνση πηγής είναι αυτή του PC1. Η διεύθυνση προορισμού είναι αυτή που χρησιμοποιεί το RIPv2 για multicast μόνο στο τοπικό υποδίκτυο.

2.4

Όχι.

2.5

Έχει τιμή 1.

2.6

Το RIP χρησιμοποιεί UDP και τη θύρα 520.

2.7

Διαφημίζονται 2 δίκτυα, τα `172.17.17.0/30` και `192.168.2.0/24`. Δεν υπάρχει διαφήμιση για το δίκτυο του LAN1.

2.8

Τα βλέπουμε περίπου κάθε 30 sec (απόκλιση $\pm$15 sec), επιβεβαιώνοντας το ερώτημα 1.22.

2.9

Στον R1 εκτελούμε `tcpdump -vni em1`. Παρατηρούμε όντως μηνύματα από τον R1.

2.10

Διαφημίζεται ένα δίκτυο, το `192.168.1.0/24`, ενώ λείπουν τα `172.17.17.0/30` και `192.168.2.0/24`.

2.11

Ναι, παρατηρούμε μηνύματα από τον R2. Διαφημίζεται το δίκτυο `192.168.2.0/24`.

2.12

Όταν διαφημίζουν ένα δίκτυο έχουν μέγεθος 24 bytes, όταν διαφημίζουν δύο έχουν μέγεθος 44 bytes, ενώ το μέγεθος της κάθε εγγραφής RIP είναι 20 bytes.

2.13

Στον R1 εκτελούμε `tcpdump -vni em0 "udp port 520"`.

2.14

Στον R2 εκτελούμε:

```
router rip
no network 192.168.2.0/24
```

Αμέσως μετά τη διαγραφή εμφανίστηκε RIP Response σχετικό με το δίκτυο `192.168.2.0/24`. Σε αυτό το μήνυμα, η διαδρομή προς το `192.168.2.0/24` διαφημίζεται με κόστος 16 (άπειρο).

2.15

Στον R2 εκτελούμε:

```
network 192.168.2.0/24
```

Αμέσως μετά την αλλαγή εμφανίζεται μήνυμα RIP Response σχετικό με το δίκτυο `192.168.2.0/24`, το οποίο διαφημίζει κόστος 2 για τη διαδρομή μέσω του `192.168.2.0/24`.

2.16

Στον R2 εκτελούμε `tcpdump -vni em0 "udp port 520 && src 172.17.17.1`.

2.17

Στον R1 εκτελούμε:

```
router rip
no network 192.168.1.0/24
```

Παράγεται αμέσως μήνυμα RIP στο WAN1 σχετικό με τη διαγραφή.

2.18

Όχι, δεν παράχθηκε, διότι το `192.168.1.0/24` (LAN1) που διαγράφηκε δεν διαφημίζεται έτσι κι αλλιώς στην ίδια τη ζεύξη του.

2.19

Στον R2 εκτελούμε `no network 192.168.2.0/24` και αμέσως μετά στον R1 εκτελούμε `netstat -rn`. Όντως το `192.168.2.0/24` έχει διαγραφεί από τον πίνακα δρομολόγησης.

2.20

Στον R1 εκτελούμε `do show ip rip`. Αρχικά το `192.168.2.0/24` δεν έχει διαγραφεί από τον πίνακα διαδρομών RIP του R1, ύστερα όμως από δύο λεπτά διαγράφεται κι από εκεί, αφού πλέον έχει παρέλθει η προκαθορισμένη περίοδος ”garbage”.

2.21

Στον R1 εκτελούμε:

```
network 192.168.1.0/24
```

Στον R2 εκτελούμε:

```
network 192.168.2.0/24
```

2.22

Μπορούμε εκτελώντας στον R1:

```
passive-interface em0
```

Αντίστοιχα στον R2:

```
passive-interface em1
```

2.23

Πλέον επειδή οι διεπαφές των δρομολογητών στα LAN1 και LAN2 βρίσκονται σε παθητική κατάσταση, δεν παρατηρούνται RIP Responses στα δίκτυα αυτά.

Άσκηση 3: Εναλλακτικές διαδρομές

Επειδή κλείσαμε όλα τα μηχανήματα, στήνουμε το δίκτυο της άσκησης από την αρχή με τις παρακάτω εντολές. Αυτή η διαδικασία δεν αντιστοιχεί σε κάποιο ερώτημα και μπορεί να αγνοηθεί κατά τη διόρθωση.

```
--------------PC1-------------
vtysh
configure terminal

hostname PC1

interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1

--------------PC2-------------
vtysh
configure terminal

hostname PC2

interface em0
```

```
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1

--------------R1--------------
cli
configure terminal

hostname R1

LAN1: interface em0
ip address 192.168.1.1/24

WAN1: interface em1
ip address 172.17.17.1/30

router rip

version 2

network 192.168.1.0/24
network 172.17.17.0/30

passive-interface em0

--------------R2--------------
cli
configure terminal

hostname R2

WAN1: interface em0
ip address 172.17.17.2/30

LAN2: interface em2
ip address 192.168.2.1/24

router rip

version 2

network 192.168.2.0/24
network 172.17.17.0/30

passive-interface em2
```

3.1

Στον R1 εκτελούμε (σημειωτέον ότι em0 → LAN1, em1 → WAN1, em2 → WAN2):

```
cli
configure terminal

interface em2
ip address 172.17.17.5/30

router rip
network 172.17.17.4/30
```

3.2

Στον R2 εκτελούμε (σημειωτέον ότι em0 → WAN1, em1 → WAN3, em2 → LAN2):

```
cli
configure terminal

interface em1
ip address 172.17.17.9/30

router rip
network 172.17.17.8/30
```

3.3

Στον R3 εκτελούμε:

```
cli
configure terminal

hostname R3

interface em0
ip address 172.17.17.6/30

interface em1
ip address 172.17.17.10/30

router rip
network 172.17.17.4/30
network 172.17.17.8/30
```

3.4

Στον R1 εκτελούμε `do show ip route`. Ο R1 έχει μάθει δυναμικά τα δίκτυα:

```
172.17.17.8/30
193.168.2.0/24
```

3.5

Στον R2 εκτελούμε `do show ip route`. Ο R2 έχει μάθει δυναμικά τα δίκτυα:

```
172.17.17.4/30
192.168.1.0/24
```

3.6

Στον R3 εκτελούμε `do show ip route`. Ο R3 έχει μάθει δυναμικά τα δίκτυα:

```
172.17.17.0/30
192.168.1.0/24
192.168.2.0/24
```

3.7

Στο PC1 εκτελούμε `do ping 192.168.2.2`. Το ping είναι επιτυχές, οπότε μπορούμε να επικοινωνήσουμε από το PC1 με το PC2.

3.8

Στον R3 εκτελούμε:

```
interface em2
ip address 192.168.3.1/24
```

3.9

Στους R1 και R2 εκτελούμε `do show ip route`. Δεν έχουν αλλάξει οι δυναμικές εγγραφές.

3.10

Στον R3 εκτελούμε:

```
router rip
network 192.168.3.0/24
```

3.11

Στους R1 και R2 εκτελούμε `do show ip route`. Πλέον τόσο στον R1 όσο και στον R2 έχει προστεθεί μία νέα δυναμική εγγραφή που αφορά το δίκτυο `192.168.3.0/24`.

3.12

Ναι είναι, αφού στέλνονται κατευθείαν ενημερώσεις στους R1 και R2 σχετικές με την προσθήκη της νέας διαδρομής.

3.13

Στον R3 εκτελούμε:

```
router rip

no network 172.17.17.4/30
no network 172.17.17.8/30
no network 192.168.3.0/24

network 0.0.0.0/0
```

Το δίκτυο `0.0.0.0/0` υποδηλώνει ότι ενεργοποιείται το RIP σε όλα τα υποδίκτυα που είναι συνδεδεμένα στον R3.

3.14

Στον R3 εκτελούμε `do show ip rip status` και βλέπουμε ότι το RIP είναι ενεργοποιημένο στις διεπαφές `em0`, `em1`, `em2` και `lo0`. Στη δρομολόγηση συμμετέχει το δίκτυο `0.0.0.0/0`.

3.15

Στους R1 και R2 εκτελούμε `do show ip route`. Δεν υπάρχει κάποια αλλαγή.

3.16

Στον R3 εκτελούμε `tcpdump -vni em0` για να καταγράψουμε την κίνηση στο WAN2. Ο R3 διαφημίζει:

```
172.17.17.8/30
192.168.2.0/24
192.168.3.0/24
```

3.17

Όχι δεν υπάρχει, γιατί σύμφωνα με τον μηχανισμό του διαιρεμένου ορίζοντα, οι δρομολογητές δεν διαφημίζουν μία διαδρομή στη διεπαφή από όπου την έμαθαν. Συγκεκριμένα, ο R3 δεν διαφημίζει στο WAN2 τη διαδρομή που αφορά το LAN1, γιατί και ο ίδιος την έμαθε μέσω του WAN2.

3.18

Συμπεραίνουμε ότι όταν εισάγουμε το δίκτυο `0.0.0.0/0`, στα μηνύματα RIP περιλαμβάνονται όλα τα δίκτυα που είναι συνδεδεμένα με τον δρομολογητή, λαμβάνοντας υπόψιν τους περιορισμούς που αναφέρθηκαν στο 3.17.

3.19

Στον R1 εκτελούμε:

```
tcpdump -vni em1 "udp port 520 && src 172.17.17.2"      # for WAN1
tcpdump -vni em2 "udp port 520 && src 172.17.17.6"      # for WAN2
```

Τόσο ο R2 όσο και ο R3 διαφημίζουν κόστος 1 για τη διαδρομή προς το WAN3.

Στον R1 εκτελούμε `do show ip route` και βλέπουμε ότι έχει διαλέξει τη διαδρομή μέσω του R2.

3.20

Στον R1 εκτελούμε:

```
tcpdump -vni em1 "udp port 520 && src 172.17.17.1"      # for WAN1
tcpdump -vni em2 "udp port 520 && src 172.17.17.5"      # for WAN2
```

Διαφήμιση για το `172.17.17.8/30` περιέχεται στο WAN2, ενώ δεν περιέχεται στο WAN1, εξαιτίας του μηχανισμού διαιρεμένου ορίζοντα, αφού ο R1 έτυχε να μάθει τη διαδρομή προς το `172.17.17.8/30` πρώτα από τον R2 (τον οποίο και έχει επιλέξει σαν επόμενο βήμα). Ο λόγος που δεν αλλάζει η προτίμηση του R1 από τον R2 στον R3 είναι επειδή σύμφωνα με το RFC 2453, διαδρομές ίδιου κόστους στον ίδιο προορισμό από διαφορετικές πηγές δεν προκαλούν αλλαγή στον πίνακα δρομολόγησης. Σημειώνεται ότι υπάρχει μία αναφορά στο παραπάνω έγγραφο για προτίμηση άλλων διαδρομών αν η τρέχουσα εγγραφή έχει ξεπεράσει το ήμισυ της ζωής της, αλλά, εκτός του ότι είναι προαιρετική, επειδή ο R2 στέλνει συνεχώς ενημερώσεις στον R1, η τρέχουσα εγγραφή παραμένει πάντα επαρκώς πρόσφατη.

Άσκηση 4: Αλλαγές στην τοπολογία, σφάλμα καλωδίου και RIP

4.1

Στο PC3 εκτελούμε:

```
vtysh
configure terminal
hostname PC3
interface em0
ip address 192.168.3.2/24
ip route 0.0.0.0/0 192.168.3.1
```

4.2

Στο PC1 εκτελούμε `do ping 192.168.2.2`. Τα PC1 και PC2 επικοινωνούν κανονικά.

4.3

Στο PC2 εκτελούμε `do ping 192.168.3.2`. Τα PC1 και PC2 επικοινωνούν κανονικά.

4.4

Στο PC3 εκτελούμε `do ping 192.168.1.2`. Τα PC1 και PC2 επικοινωνούν κανονικά.

4.5

Στους R1,2,3 εκτελούμε `do show ip route`. Έχουμε:

```
R1 routing table:

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
R>* 172.17.17.8/30 [120/2] via 172.17.17.2, em1
C>* 192.168.1.0/24 is directly connected, em0
R>* 192.168.2.0/24 [120/2] via 172.17.17.2, em1
R>* 192.168.3.0/24 [120/2] via 172.17.17.6, em2

R2 routing table:

C>* 127.0.0.0/8 is directly connected, lo0
C>* 172.17.17.0/30 is directly connected, em0
C>* 172.17.17.4/30 [120/2] via 172.17.17.1, em0
R>* 172.17.17.8/30 is directly connected, em1
R>* 192.168.1.0/24 [120/2] via 172.17.17.1, em0
C>* 192.168.2.0/24 is directly connected, em2
R>* 192.168.3.0/24 [120/2] via 172.17.17.10, em1

R3 routing table:

C>* 127.0.0.0/8 is directly connected, lo0
R>* 172.17.17.0/30 [120/2] via 172.17.17.5, em0
C>* 172.17.17.4/30 is directly connected, em0
C>* 172.17.17.8/30 is directly connected, em1
R>* 192.168.1.0/24 [120/2] via 172.17.17.5, em0
R>* 192.168.2.0/24 [120/2] via 172.17.17.9, em1
C>* 192.168.3.0/24 is directly connected, em2
```

4.6

Εκτελούμε:

```
R1:

interface em1
link-detect

interface em2
link-detect
```

R2:

```
interface em0
link-detect

interface em1
link-detect
```

R3:

```
interface em0
link-detect

interface em1
link-detect
```

4.7

Αποσυνδέουμε τα άκρα του WAN1 και παρατηρούμε τις εξής αλλαγές:

- Σε όλους τους πίνακες δρομολόγησης έχουν αφαιρεθεί οι εγγραφές που αφορούν το δίκτυο `172.17.17.0/30`.
- Στον πίνακα του R1, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.2` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.6`, και αντίστοιχα η `em1` σε `em2`.
- Στον πίνακα του R2, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.1` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.10`, και αντίστοιχα η `em0` σε `em1`.
- Η διαδρομή για το `192.168.2.0/24` στον πίνακα του R1 έχει πλέον κόστος 3, όπως και η διαδρομή για το `192.168.1.0/24` στον πίνακα του R2.

4.8

Ελέγχουμε:

```
PC1 <--> PC2: do ping 192.168.2.2 (from PC1)
PC1 <--> PC3: do ping 192.168.3.2 (from PC1)
PC2 <--> PC3: do ping 192.168.3.2 (from PC2)
```

Τα PC1,2,3 επικοινωνούν μεταξύ τους.

4.9

Επανασυνδέουμε τα άκρα του WAN1, αποσυνδέουμε τα άκρα του WAN2 και παρατηρούμε τις εξής αλλαγές:

- Σε όλους τους πίνακες δρομολόγησης έχουν αφαιρεθεί οι εγγραφές που αφορούν το δίκτυο `172.17.17.4/30`.

- Στον πίνακα του R1, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.6` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.2`, και αντίστοιχα η `em2` σε `em1`.
- Στον πίνακα του R3, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.5` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.9`, και αντίστοιχα η `em0` σε `em1`.
- Η διαδρομή για το `192.168.3.0/24` στον πίνακα του R1 έχει πλέον κόστος 3, όπως και η διαδρομή για το `192.168.1.0/24` στον πίνακα του R3.

4.10

Ελέγχουμε:

```
PC1 <--> PC2: do ping 192.168.2.2 (from PC1)
PC1 <--> PC3: do ping 192.168.3.2 (from PC1)
PC2 <--> PC3: do ping 192.168.3.2 (from PC2)
```

Τα PC1,2,3 επικοινωνούν μεταξύ τους.

4.11

Επανασυνδέουμε τα άκρα του WAN2, αποσυνδέουμε τα άκρα του WAN3 και παρατηρούμε τις εξής αλλαγές:

- Σε όλους τους πίνακες δρομολόγησης έχουν αφαιρεθεί οι εγγραφές που αφορούν το δίκτυο `172.17.17.8/30`.
- Στον πίνακα του R2, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.10` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.1`, και αντίστοιχα η `em1` σε `em0`.
- Στον πίνακα του R3, όσες εγγραφές είχαν ως επόμενο βήμα τη διεπαφή `172.17.17.9` έχουν αλλάξει το επόμενο βήμα σε `172.17.17.5`, και αντίστοιχα η `em1` σε `em0`.
- Η διαδρομή για το `192.168.3.0/24` στον πίνακα του R2 έχει πλέον κόστος 3, όπως και η διαδρομή για το `192.168.2.0/24` στον πίνακα του R3.

4.12

Ελέγχουμε:

```
PC1 <--> PC2: do ping 192.168.2.2 (from PC1)
PC1 <--> PC3: do ping 192.168.3.2 (from PC1)
PC2 <--> PC3: do ping 192.168.3.2 (from PC2)
```

Τα PC1,2,3 επικοινωνούν μεταξύ τους.

4.13

Επανασυνδέουμε τα άκρα του WAN3 και εκτελούμε από τον PC1 `do ping 192.168.2.2`. Ύστερα αποσυνδέουμε τα άκρα του WAN1. Για να εγκατασταθεί η νέα διαδρομή χρειάστηκαν ~20 δευτερόλεπτα.

4.14

Επανασυνδέουμε τα άκρα του WAN1. Μπορούμε να καταλάβουμε ότι η νέα διαδρομή (PC1 → R1 → R3 → R2 → PC2) ξαναέδωσε τη θέση της στην παλιά (PC1 → R1 → R2 → PC2), επειδή πλέον η τιμή του TTL στα μηνύματα ICMP Reply είναι 62 αντί για 61. Αυτό ισχύει διότι η παλιά διαδρομή είναι μικρότερη σε μήκος από την καινούργια κατά ένα hop.

4.15

Στον R1 εκτελούμε `do show ip rip`. Έχουμε:

```
Network           Metric
172.17.17.0/30    1
192.168.2.0/24    2
```

4.16

Παριστάνει το `timeout` (default τιμή 180 δευτερόλεπτα), μετά τη λήξη του οποίου οι εγγραφές παύουν να ισχύουν, και το οποίο ανανεώνεται κάθε φορά που έρχεται ενημέρωση για τη συγκεκριμένη εγγραφή.

4.17

Αποσυνδέουμε τα άκρα του WAN1 και αμέσως ξαναεκτελούμε στον R1 `do show ip rip`. Έχουμε:

```
Network           Metric    Time
172.17.17.0/30    16        01:59
192.168.2.0/24    16        01:59
```

4.18

Μετά από λίγο η διαδρομή προς το `192.168.2.0/24` έχει πλέον:

- Στο πεδίο ”Next Hop” τη διεπαφή `172.17.17.6` αντί για `172.17.17.2`
- Το πεδίο ”Metric” ίσο με 3
- Στο πεδίο ”From” τη διεπαφή `172.17.17.6` αντί για `172.17.17.2`

4.19

Μετά από δύο λεπτά η διαδρομή προς το `172.17.17.0/30` διαγράφεται από τον πίνακα διαδρομών.

4.20

Παριστάνει το `garbage` (default τιμή 120 δευτερόλεπτα), με την λήξη του οποίου οι εγγραφές που δεν ισχύουν πλέον διαγράφονται από τον πίνακα διαδρομών.

4.21

Επανασυνδέουμε τα άκρα του WAN1 και εκτελούμε στον R1:

```
tcpdump -vni em1 "udp port 520 && src 172.17.17.1"   # for WAN1
tcpdump -vni em2 "udp port 520 && src 172.17.17.5"   # for WAN2
```

Διαφήμιση για το `172.17.17.8/30` περιλαμβάνεται στα μηνύματα RIP του WAN1. Αυτό συμβαίνει διότι η επαναφορά του WAN1 δεν αλλάζει τον πίνακα διαδρομών του R1, αφού η διαδρομή για το `172.17.17.8/30` μέσω του R2 έχει κόστος 2, όπως και η ισχύουσα διαδρομή μέσω του R3. Έτσι, σύμφωνα με το μηχανισμό του διαιρεμένου ορίζοντα, ο R1 δεν διαφημίζει στο WAN2 τη διαδρομή για το `172.17.17.8/30`, επειδή από εκεί την έμαθε εξαρχής.

Άσκηση 5: Τοπολογία με πολλαπλές WAN διασυνδέσεις

Κατασκευή του δικτύου

Αντιστοίχιση των network adapters σε LAN/WAN

```
-------------PC1-------------
Network Adapter 1 (em0): LAN1

-------------PC2-------------
Network Adapter 1 (em0): LAN2

-------------R1--------------
Network Adapter 1 (em0): LAN1
Network Adapter 2 (em1): WAN1
Network Adapter 3 (em2): WAN3

-------------R2--------------
Network Adapter 1 (em0): LAN2
Network Adapter 2 (em1): WAN2
Network Adapter 3 (em2): WAN4

-------------C1--------------
Network Adapter 1 (em0): CORE
Network Adapter 2 (em1): WAN1
Network Adapter 3 (em2): WAN2

-------------C2--------------
Network Adapter 1 (em0): CORE
Network Adapter 2 (em1): WAN3
Network Adapter 3 (em2): WAN4
```

Ορισμός ονομάτων και διευθύνσεων IP μέσω cli

```
-----------PC1-----------
vtysh
```

```
configure terminal

hostname PC1

interface em0
ip address 192.168.1.2/24

-----------PC2-----------
vtysh
configure terminal

hostname PC2

interface em0
ip address 192.168.2.2/24

-----------R1------------
cli
configure terminal

hostname R1

interface em0
ip address 192.168.1.1/24

interface em1
ip address 10.0.1.1/30

interface em2
ip address 10.0.1.5/30

interface lo0
ip address 172.22.1.1/32

-----------R2------------
cli
configure terminal

hostname R2

interface em0
ip address 192.168.2.1/24

interface em1
ip address 10.0.2.1/30

interface em2
ip address 10.0.2.5/30
```

```
interface lo0
ip address 172.22.2.1/32

-----------C1------------
cli
configure terminal

hostname C1

interface em0
ip address 10.0.0.1/30

interface em1
ip address 10.0.1.2/30

interface em2
ip address 10.0.2.2/30

interface lo0
ip address 172.22.1.2/32

-----------C2------------
cli
configure terminal

hostname C2

interface em0
ip address 10.0.0.2/30

interface em1
ip address 10.0.1.6/30

interface em2
ip address 10.0.2.6/30

interface lo0
ip address 172.22.2.2/32
```

Διαγραφή προκαθορισμένης διαδρομής στα PC

Δεν χρειάζεται, αφού δεν θα χρησιμοποιήσουμε τα PC από την προηγούμενη άσκηση.

5.1

Στους R1, R2, C1, C2 εκτελούμε:

```
router rip
```

```
network 0.0.0.0/0
```

5.2

Στον R1 εκτελούμε `do show ip route rip`. Εμφανίζονται 7 δυναμικές εγγραφές.

5.3

Στον R2 εκτελούμε `do show ip route rip`. Εμφανίζονται 7 δυναμικές εγγραφές.

5.4

Στον C1 εκτελούμε `do show ip route rip`. Εμφανίζονται 7 δυναμικές εγγραφές.

5.5

Στον C2 εκτελούμε `do show ip route rip`. Εμφανίζονται 7 δυναμικές εγγραφές.

5.6

Στον R1 εκτελούμε `do show ip rip status` και βλέπουμε ότι συμμετέχει με το δίκτυο `0.0.0.0`, δηλαδή όλα τα δίκτυα στα οποία έχει συνδεδεμένη διεπαφή.

5.7

Στον R1 εκτελούμε `tcpdump -vni em0 "udp port 520"`. Ο R1 διαφημίζει τα δίκτυα:

```
10.0.0.0/30
10.0.1.0/30
10.0.1.4/30
10.0.2.0/30
10.0.2.4/30
172.22.1.1/32
172.22.1.2/32
172.22.2.1/32
172.22.2.2/32
192.168.2.0/24
```

5.8

Στον PC1 εκτελούμε `do show ip route`. Δεν υπάρχουν δυναμικές εγγραφές.

5.9

Δεν υπάρχει προκαθορισμένη διαδρομή στο PC1. Εκτελούμε στο PC1:

```
router rip
network em0
```

5.10

Περιέχει 10 δυναμικές εγγραφές.

5.11

Δεν υπάρχει προκαθορισμένη διαδρομή στο PC2. Εκτελούμε στο PC2:

```
router rip
network em0
```

5.12

Υπάρχουν 2 διαδρομές ελαχίστου κόστους μεταξύ LAN1 και LAN2:

- **Διαδρομή Α:** PC1 → R1 → C1 → R2 → PC2 (και η αντίστροφή της)
- **Διαδρομή Β:** PC1 → R1 → C2 → R2 → PC2 (και η αντίστροφή της)

5.13

Στον PC1 εκτελούμε `do traceroute 192.168.2.2`. Βλέπουμε ότι ακολουθείται η διαδρομή **Α** (βλ. 5.12).

5.14

Στον PC2 εκτελούμε `do traceroute 192.168.1.2`. Βλέπουμε ότι ακολουθείται η διαδρομή **Α** (βλ. 5.12).

5.15

Ναι, χρησιμοποιείται η ίδια διαδρομή.

5.16

Εκτελούμε στο PC1:

```
do ping 172.22.1.1
do ping 172.22.1.2
do ping 172.22.2.1
do ping 172.22.2.2
```

Όλα τα ping είναι επιτυχή, οπότε μπορούμε να επικοινωνήσουμε με όλες τις loopback διαχείρισης.

5.17

Εκτελούμε στο PC2:

```
do ping 172.22.1.1
do ping 172.22.1.2
do ping 172.22.2.1
do ping 172.22.2.2
```

Όλα τα ping είναι επιτυχή, οπότε μπορούμε να επικοινωνήσουμε με όλες τις loopback διαχείρισης.

5.18

- WAN1: Μπορεί να αποκοπεί.
- WAN2: Μπορεί να αποκοπεί.
- WAN3: Μπορεί να αποκοπεί.
- WAN4: Μπορεί να αποκοπεί.
- CORE: Μπορεί να αποκοπεί.

5.19

- WAN1, WAN2 και CORE: Μπορούν να αποκοπούν.

5.20

- WAN1 και WAN3: Δεν μπορούν να αποκοπούν.

5.21

- WAN2 και WAN3: Μπορούν να αποκοπούν.

5.22

- WAN2 και WAN4: Δεν μπορούν να αποκοπούν.

5.23

- WAN3, WAN4 και CORE: Μπορούν να αποκοπούν.

5.24

- WAN1 και WAN4: Μπορούν να αποκοπούν.

5.25

Εκτελούμε:

```
R1:
interface em0
link-detect

interface em1
link-detect

interface em2
link-detect
```

```
interface lo0
link-detect

R2:
interface em0
link-detect

interface em1
link-detect

interface em2
link-detect

interface lo0
link-detect

C1:
interface em0
link-detect

interface em1
link-detect

interface em2
link-detect

interface lo0
link-detect

C2:
interface em0
link-detect

interface em1
link-detect

interface em2
link-detect

interface lo0
link-detect
```

Ύστερα εκτελούμε στο PC1:

```
do ping 172.22.2.2
```

Αποσυνδέουμε το CORE και μετά το WAN3. Στην έξοδο του ping παρατηρούμε να εμφανίζεται συνέχεια ”No route to host” για κάποιο διάστημα, μετά το οποίο το ping λειτουργεί και πάλι

κανονικά.

Αυτό συμβαίνει διότι με το που διακόπτεται η ζεύξη WAN3, η εγγραφή στον πίνακα δρομολόγησης του R1 με προορισμό την `172.22.2.2` διαγράφεται, αφού διέρχεται από το WAN3. Ο PC1 ενημερώνεται για την διακοπή της ζεύξης από τον R1, ο οποίος διαφημίζει ότι η διαδρομή αυτή έχει πλέον Metric = 16, οπότε την αφαιρεί κι αυτός από τον πίνακα δρομολόγησης του, εξού και το μήνυμα "No route to host". Αφού περάσει ένα μικρό χρονικό διάστημα, ο PC1 ενημερώνεται για την εναλλακτική διαδρομή (PC1 $\rightarrow$ R1 $\rightarrow$ C1 $\rightarrow$ C2), ενημερώνει τον πίνακα δρομολόγησής του και μπορεί ξανά να επικοινωνήσει με επιτυχία.

5.26

Περνούν ~25 δευτερόλεπτα μέχρι να επανέλθει η επικοινωνία.

Άσκηση 6: RIP και αναδιανομή διαδρομών

6.1

Στον C1 εκτελούμε:

```
ip route 4.0.0.0/8 172.22.1.2
```

6.2

Στον C1 εκτελούμε:

```
do show ip route
```

Η εγγραφή έχει τοποθετηθεί επιτυχώς.

6.3

Στους PC1, PC2, R1, R2, C2 εκτελούμε `do show ip route`. Η εγγραφή δεν έχει τοποθετηθεί.

6.4

Στον C1 εκτελούμε:

```
router rip
redistribute static

do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα δρομολόγησης του C1.

6.5

Στους PC1, PC2, R1, R2, C2 εκτελούμε `do show ip route`. Η εγγραφή έχει τοποθετηθεί στους πίνακες ως δυναμική.

6.6

Στον C2 εκτελούμε:

```
ip route 0.0.0.0/0 172.22.2.2
```

6.7

Στον C2 εκτελούμε `do show ip route`. Η εγγραφή έχει τοποθετηθεί στον πίνακα του C2.

6.8

Στους PC1, PC2, R1, R2, C1 εκτελούμε `do show ip route`. Η εγγραφή δεν έχει τοποθετηθεί.

6.9

Στον C2 εκτελούμε:

```
router rip
default-information originate

do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα του C2.

6.10

Στους PC1, PC2, R1, R2, C1 εκτελούμε `do show ip route`. Βλέπουμε ότι έχει προστεθεί μία δυναμική εγγραφή στον πίνακα κάθε μηχανήματος η οποία αφορά την προεπιλεγμένη διαδρομή, με επόμενο βήμα τη διεπαφή του επόμενου δρομολογητή της διαδρομής προς το C2.

6.11

Στον C2 εκτελούμε:

```
no default-information originate
```

Ύστερα στον C1 εκτελούμε:

```
ip route 0.0.0.0/0 10.0.0.2

router rip
default-information originate
```

6.12

Στον πίνακα του C2 προστίθεται μία μη-επιλεγμένη δυναμική εγγραφή σχετικά με την προεπιλεγμένη διαδρομή μέσω του C1, η οποία έχει προκύψει μέσω της διαφήμισης του C1 (εντολή default-information originate). Η εγγραφή δεν έχει επιλεγεί επειδή έχει μεγαλύτερη διαχειριστική απόσταση από την αντίστοιχη στατική (120 έναντι 1).

6.13

Εκτελούμε στον C2:

```
no ip route 0.0.0.0/0 172.22.2.2

do show ip route
```

Η δυναμική εγγραφή που αναφέραμε προηγουμένως είναι πλέον επιλεγμένη.

6.14

Εκτελούμε στα PC1, PC2:

```
do show ip route
```

Ο πίνακας δρομολόγησης έχει 13 εγγραφές.

6.15

Στον PC1 εκτελούμε `do ping 4.4.4.4`. Λαμβάνουμε ”Time to live exceeded”. Αυτό συμβαίνει διότι το πακέτο, με βάση τις εγγραφές του πίνακα δρομολόγησης με προορισμό το `4.0.0.0/8` προωθείται από τον PC1 στον R1, από τον R1 στον C1 και από τον C1 στη loopback του C1, με αποτέλεσμα να εγκλωβίζεται στον C1 και να μηδενίζεται το TTL χωρίς το μήνυμα να φτάνει στον προορισμό του, εξού και το μήνυμα ”Time to live exceeded” που λαμβάνει ο PC1.

6.16

Στον PC1 εκτελούμε `do ping 5.5.5.5`. Λαμβάνουμε ”Time to live exceeded”. Αυτό συμβαίνει διότι το πακέτο, με βάση τις εγγραφές του πίνακα δρομολόγησης με προορισμό το `0.0.0.0/0` (προεπιλεγμένη διαδρομή), προωθείται από τον PC1 στον R1, από τον R1 στον C1. Από ’κει και πέρα εγκλωβίζεται στον βρόχο C1-C2, αφού ο C1 προωθεί στον C2 και ο C2 στον C1, με αποτέλεσμα να μηδενίζεται το TTL χωρίς το μήνυμα να φτάνει στον προορισμό του, εξού και το μήνυμα ”Time to live exceeded” που λαμβάνει ο PC1.

6.17

Στον R1 εκτελούμε:

```
access-list private permit 192.168.0.0/16
access-list private deny any
```

6.18

Στον R1 εκτελούμε:

```
password ntua
exit
exit
```

6.19

Στο PC2 εκτελούμε `telnet 10.0.1.5 2602`.

6.20

Εκτελούμε στο PC2 που είναι συνδεμένο μέσω telnet στον R1:

```
enable
configure terminal
router rip
distribute-list private out em0
```

6.21

Αρχικά εκτελούμε στον PC1 `do show ip route` και δεν παρατηρούμε κάποια διαφορά. Ύστερα από τρία λεπτά όμως, οι περισσότερες εγγραφές έχουν διαγραφεί, και έχουν μείνει μόνο δύο εγγραφές, μία που αφορά το ίδιο το `192.168.1.0/24`, και μία δυναμική που αφορά το `192.168.2.0/24`.

6.22

Στον PC1 εκτελούμε `do show ip rip`. Αρχικά οι εγγραφές παραμένουν, αλλά ύστερα από δύο λεπτά διαγράφονται όλες εκτός από τις διαδρομές προς τα `192.168.1.0/24` και `192.168.2.0/24`, αφού έχει περάσει το διάστημα `garbage` (προεπιλεγμένη τιμή 2 λεπτά).

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 8
Δυναμική δρομολόγηση OSPF

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 04/05/2022

Προετοιμασία στο σπίτι

1

Εκτελούμε `service frr stop`.

2

Εκτελούμε `touch /usr/local/etc/frr/ospfd.conf`.

3

Εκτελούμε `chown frr:frr /usr/local/etc/frr/ospfd.conf`.

4

Αλλάζουμε τη ζητούμενη γραμμή του `/etc/rc.conf` σε:

```
frr_daemons="zebra staticd ripd ospfd"
```

5

Εκτελούμε `service frr start`.

6

Εκτελούμε `poweroff` και κάνουμε Export Appliance σε ένα αρχείο `OSPF.ova` για μελλοντική χρήση.

Άσκηση 1: Εισαγωγή στο OSPF

1.1

Στο PC1 εκτελούμε:

```
vtysh
configure terminal

hostname PC1

interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1
```

1.2

Στο PC2 εκτελούμε:

```
vtysh
configure terminal

hostname PC2

interface em0
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1
```

1.3

Στον R1 εκτελούμε:

```
cli
configure terminal

hostname R1

interface em0
ip address 192.168.1.1/24

interface em1
ip address 172.17.17.1/30
```

1.4

Στον R1 εκτελούμε `do show ip route`. Δεν εμφανίζεται καμία στατική εγγραφή.

1.5

Στον R1 εκτελούμε:

```
exit     # To enter global configuration mode
router ?
```

Το πρωτόκολλο OSPF είναι διαθέσιμο.

1.6

Στον R1 εκτελούμε `router ospf`.

1.7

Στον R1 πατάμε το πλήκτρο `"?"`. Εμφανίζονται 24 διαθέσιμες εντολές.

1.8

Στον R1 εκτελούμε `network 192.168.1.0/24 area 0`.

1.9

Στον R1 εκτελούμε `network 172.17.17.0/30 area 0`.

1.10

Στον R1 εκτελούμε:

```
exit
do show ip route
```

Βλέπουμε ότι έχουν προστεθεί δύο εγγραφές στον πίνακα δρομολόγησης:

```
O   172.17.17.0/30 [110/10] is directly connected, em1
O   192.168.1.0/24 [110/10] is directly connected, em0
```

1.11

Στον R2 εκτελούμε:

```
1.3

cli
configure terminal

hostname R2

interface em0
ip address 172.17.17.2/30
```

```
interface em1
ip address 192.168.2.1/24
```

1.4

```
do show ip route                # OK, no static records found
```

1.5

```
exit # To enter global configuration mode
router ?                        # OPSF protocol is available
```

1.6

```
router ospf
```

1.7

```
"?"                             # 24 commands available
```

1.8

```
network 192.168.2.0/24 area 0
```

1.9

```
network 172.17.17.0/30 area 0
```

Τέλος από τον PC1 εκτελούμε `do ping 192.168.2.2`. Τα PC1,2 επικοινωνούν κανονικά.

1.12

Χαρακτηρίζονται ως εσωτερικοί δρομολογητές και ως δρομολογητές κορμού, αφού έχουν όλες τις διεπαφές τους στην περιοχή 0.

1.13

Με την εντολή `do show ip route`. Έχουν προστεθεί τρεις νέες εγγραφές:

```
O   172.17.17.0/30 [110/10] is directly connected, em0
O>* 192.168.1.0/24 [110/20] via 172.17.17.1, em0
O   192.168.2.0/24 [110/10] is directly connected, em1
```

1.14

Ξεχωρίζουν από το γράμμα "O" στην αρχή της γραμμής.

1.15

Δηλώνονται με το σύμβολο `"*"` στην αρχή της γραμμής.

1.16

Η διαχειριστική απόσταση των διαδρομών OSPF είναι 110, και εμφανίζεται μέσα σε αγκύλες στη μορφή `[administrative distance/route length]`.

1.17

Έχει επιλεχθεί διότι το WAN1 είναι άμεσα συνδεδεμένο στον R2 μέσω της διεπαφής em0, οπότε αυτή η επιλογή υπερισχύει όλων των υπολοίπων (διαχειριστική απόσταση 0).

1.18

Στον R2 εκτελούμε:

```
exit
exit
netstat -rn
```

Μπορούμε να καταλάβουμε ότι η εγγραφή:

```
Destination        Gateway         Flags     Netif
192.168.1.0/24     172.17.17.1     UG1       em0
```

είναι δυναμική, επειδή στις σημαίες υπάρχει το σύμβολο `"1"`.

1.19

Στον R1 σε νέο παράθυρο εκτελούμε:

```
tcpdump -vni em0
```

και περιμένουμε τουλάχιστον μισό λεπτό.

1.20

Η διεύθυνση πηγής είναι `192.168.1.1` (R1@em0)

1.21

Η διεύθυνση προορισμού είναι `224.0.0.5` μία multicast διεύθυνση που χρησιμοποιείται από το OSPF. Σε αυτήν ακούνε όλοι οι OSPF δρομολογητές, και σε αυτήν αποστέλλονται τα πακέτα Hello.

1.22

- Πρωτόκολλο στρώματος δικτύου: IPv4
- Αριθμός πρωτοκόλλου ανωτέρου στρώματος: 89

1.23

Έχει τιμή 1.

1.24

Είναι πακέτα Hello, τα οποία ανήκουν στην περιοχή Backbone (0).

1.25

Τα βλέπουμε κάθε 10 δευτερόλεπτα, όσο δηλαδή και η τιμή του Hello Timer. Το Dead Timer έχει τιμή 40 δευτερόλεπτα (τυπικά είναι 4 φορές η τιμή του Hello Timer).

1.26

Είναι `192.168.1.1`. Προκύπτει ως εξής:

- Ο R1 δεν έχει διεπαφή loopback.
- Έτσι, επιλέγει την τιμή της υψηλότερης διεύθυνσης από τις φυσικές του διεπαφές.
- Η em0 έχει την υψηλότερη διεύθυνση, οπότε επιλέγεται το `192.168.1.1` σαν Router-ID.

1.27

Είναι ο R1 (`192.168.1.1`). Δεν υπάρχει Backup Designated Router.

1.28

Στον R1 εκτελούμε `tcpdump -vni em1` και περιμένουμε τουλάχιστον μισό λεπτό. Παρατηρούμε αποστολή μηνυμάτων OSPF Hello από τον R1 με IP διεύθυνση πηγής την `172.17.17.1`.

1.29

Ναι, παρατηρούμε, με διεύθυνση πηγής την `172.17.17.2`. Το Router-ID του R2 είναι `192.168.2.1`.

1.30

Αφορά τη διεύθυνση της διεπαφής από την οποία προήλθε το μήνυμα Hello.

1.31

Περιλαμβάνουν επιπλέον πληροφορία για τον Backup Designated Router και για το Neighbor List. Αυτό συμβαίνει διότι στη ζεύξη WAN1 υπάρχουν δύο δρομολογητές, οπότε δημιουργείται σχέση γειτνίασης, ενώ στη ζεύξη LAN1 υπάρχει μόνο ένας.

1.32

Όχι, δεν περιλαμβάνονται διαφημίσεις δικτύων όπως στο RIP.

1.33

Και οι δύο δρομολογητές δίνουν προτεραιότητα 1 στα πακέτα OSPF Hello.

1.34

```
WAN1
Designated Router: 172.17.17.1
Backup Designated Router: 172.17.17.2
```

Οι τιμές είναι οι αναμενόμενες, αφού ενεργοποιήσαμε το OSPF στο WAN1 πρώτα για τον R1, οπότε επιλέχθηκε ως Designated Router. Παρά το γεγονός ότι ο R2 έχει ίδια προτεραιότητα και μεγαλύτερο Router-ID, δεν επιλέγεται ως DR. Για να επιλεγεί ως DR, θα πρέπει να έχουμε απώλεια της αντίστοιχης διεπαφής του R1. Έτσι ο R2 επιλέγεται ως BDR.

1.35

Στον R1 εκτελούμε:

```
router ospf
passive-interface em0
```

Αντίστοιχα στον R2:

```
router ospf
passive-interface em1
```

1.36

Εκτελούμε:

```
R1:
tcpdump -vni em0

R2:
tcpdump -vni em1
```

και περιμένουμε τουλάχιστον μισό λεπτό. Επιβεβαιώνουμε ότι έχει σταματήσει η αποστολή πακέτων στα LAN1 και LAN2.

1.37

Η λειτουργία του δικτύου δεν επηρεάζεται, αφού οι μόνοι δρομολογητές του δικτύου είναι οι R1 και R2, δηλαδή στις ζεύξεις LAN1 και LAN2 υπάρχει μόνο ένας δρομολογητής κάθε φορά, οπότε δεν γίνεται καμία σύναψη γειτνίασης.

Άσκηση 2: Λειτουργία του OSPF

2.1

Με την εντολή:

```
router-id <IPv4 address>
```

2.2

Εκτελούμε:

```
R1:
router ospf
router-id 0.0.0.1

R2:
router ospf
router-id 0.0.0.2
```

2.3

Στον R1 εκτελούμε `do show ip ospf`. Το Router-ID του είναι `0.0.0.1`, ανήκει σε μία περιοχή, την Backbone με Area-ID `0.0.0.0` και η LSDB του περιέχει 3 LSA.

2.4

Στον R1 εκτελούμε `do show ip ospf neighbor`. Το OSPF έχει συγκλίνει επειδή βλέπουμε ότι το State είναι `Full`, και επιπλέον ο γείτονας είναι DR.

2.5

Εκτελούμε διαδοχικά `do show ip ospf neighbor` στον R1. Το Dead Time είναι ο χρόνος που αν λήξει, ο δρομολογητής θεωρεί ότι ο γείτονας είναι ανενεργός, και παύει η γειτνίαση με τον γείτονα αυτόν. Ανανεώνεται στην προεπιλεγμένη τιμή των 40 δευτερολέπτων κάθε φορά που ο δρομολογητής λαμβάνει ένα μήνυμα Hello από τον γείτονα, και επειδή μηνύματα Hello στέλνονται κάθε 10 δευτερόλεπτα (προεπιλεγμένη τιμή), το Dead Time δεν προλαβαίνει να φτάσει κάτω από 30 δευτερόλεπτα, γι' αυτό και κυμαίνεται μεταξύ 30 και 40 δευτερολέπτων.

2.6

Με την σύνταξη `do show ip ospf neighbor detail`.

2.7

Εκτελούμε στον R1:

```
do show ip ospf interface em1
```

Έχουμε:

- Είδος δικτύου: Broadcast
- DR: R2 (`0.0.0.2`)
- BDR: R1 (`0.0.0.1`)

Βλέπουμε ότι DR είναι πλέον ο R2, και όχι ο R1 όπως στο ερώτημα 1.34, αφού με την αλλαγή των Router-ID το OSPF ανιχνεύει δύο δρομολογητές με ίδιο priority στη ζεύξη WAN1, οπότε αφού ο R2 έχει μεγαλύτερο Router-ID επιλέγεται αυτός ως DR και ο R2 ως BDR.

2.8

Στις ομάδες `OSPFAllRouters` και `OSPFDesignatedRouters`.

2.9

Στους R1 και R2 εκτελούμε `do show ip ospf database`. Βλέπουμε 2 Router και 1 Network LSA. Το αποτέλεσμα είναι ίδιο και στους δύο δρομολογητές.

2.10

Τα Router LSA έχουν Link ID `0.0.0.1` ή `0.0.0.2`, τα οποία ταυτίζονται με το Router ID του δρομολογητή που τα παράγει.

2.11

Το Link ID των Network LSA είναι `172.17.17.2`. Δεν είναι το Router ID του δρομολογητή που τα παράγει, αλλά η IP διεύθυνση της διεπαφής του DR στο WAN1.

2.12

Με την εντολή:

```
do show ip ospf database router adv-router 0.0.0.1
```

2.13

Το LAN1 χαρακτηρίζεται ως Stub Network, επειδή έχει μόνο έναν OSPF δρομολογητή, ενώ το WAN1 ως Transit Network, επειδή έχει δύο δρομολογητές OSPF.

2.14

Με την εντολή:

```
do show ip ospf network adv-router 0.0.0.2
```

2.15

Εμφανίζεται:

```
Attached Router: 0.0.0.1
Attached Router: 0.0.0.2
```

2.16

Στους R1 και R2 εκτελούμε `do show ip ospf route`. Βλέπουμε 3 εγγραφές που ανήκουν στην περιοχή `0.0.0.0`.

2.17

Είναι:

```
R1:
172.17.17.0/30, Cost 10
192.168.1.0/24, Cost 10
192.168.2.0/24, Cost 20

R2:
172.17.17.0/30, Cost 10
192.168.1.0/24, Cost 20
192.168.2.0/24, Cost 10
```

Ύστερα εκτελούμε στους R1 και R2:

```
do show ip route ospf
```

Παρατηρούμε ότι τα κόστη ταυτίζονται (υπενθυμίζουμε ότι το κόστος είναι ο δεύτερος αριθμός μέσα σε αγκύλες).

2.18

Στον R1 εκτελούμε:

```
interface em1
bandwidth 100000
```

2.19

Στον R1 εκτελούμε `do show ip ospf interface em1` και βλέπουμε το πεδίο Cost, που τώρα έχει γίνει ίσο με 1.

2.20

Εκτελούμε στον R1:

```
do show ip route
```

Παρατηρούμε ότι έχουν αλλάξει τα κόστη:

```
172.17.17.0/30, Cost 10 --> 1
192.168.2.0/24, Cost 20 --> 11
```

2.21

Εκτελούμε στον R2:

```
do show ip route
```

Το κόστος προς το LAN1 παραμένει 20, γιατί δεν έχουμε αλλάξει την ταχύτητα της διεπαφής του R2 στο WAN1.

2.22

Στον R2 εκτελούμε:

```
interface em0
bandwidth 100000
```

2.23

Στον R1 εκτελούμε `tcpdump -vni em1`.

2.24

Στον R2 εκτελούμε:

```
router ospf
no network 192.168.2.0/24 area 0
```

2.25

Παρατηρούμε να παράγονται πακέτα LS-Update από τον R2 και πακέτα LS-Ack από τον R1, χωρίς καθυστέρηση.

2.26

Στους R1 και R2 εκτελούμε `do show ip ospf route`. Παρατηρούμε ότι αφαιρέθηκε η εγγραφή για το δίκτυο `192.168.2.0/24` και στους δύο δρομολογητές.

Στον PC1 εκτελούμε `do ping 192.168.2.2`. Δεν υπάρχει επικοινωνία μεταξύ PC1 και PC2, αφού εμφανίζεται μήνυμα Destination Host Unreachable.

2.27

Στον R1 εκτελούμε `tcpdump -vni em1`. Δεν έχει σταματήσει η αποστολή μηνυμάτων OSPF το WAN1, αφού οι R1 και R2 συνεχίζουν να διατηρούν τη γειτνίασή τους (το OSPF είναι ακόμα ενεργοποιημένο στους R1, R2).

2.28

Στον R2 εκτελούμε `network 192.168.2.0/24 area 0`. Παρατηρούμε πάλι μήνυμα LS-Update από τον R2, που ενημερώνει για το δίκτυο που επανεισήχθη στο OSPF, και αντίστοιχο μήνυμα LS-Ack από τον R1.

Άσκηση 3: Εναλλακτικές διαδρομές, σφάλμα καλωδίου και OSPF

3.1

Στον R3 εκτελούμε:

```
cli
configure terminal

hostname R3

interface em0
ip address 172.17.17.6/30

interface em1
ip address 172.17.17.10/30
```

3.2

Στον R1 εκτελούμε:

```
interface em2
ip address 172.17.17.5/30
```

Ενώ στον R2 εκτελούμε:

```
interface em2
ip address 172.17.17.9/30
```

3.3

Εκτελούμε:

```
R1:
interface em1
link-detect

interface em2
link-detect

R2:
interface em0
link-detect

interface em2
link-detect

R3:
interface em0
link-detect

interface em1
link-detect
```

3.4

Εκτελούμε:

```
R1:
interface em1
ospf network point-to-point

interface em2
ospf network point-to-point

R2:
interface em0
ospf network point-to-point

interface em2
ospf network point-to-point

R3:
interface em0
ospf network point-to-point

interface em1
ospf network point-to-point
```

3.5

Στον R1 εκτελούμε:

```
router ospf
network 172.17.17.4/30 area 0
```

3.6

Στον R2 εκτελούμε:

```
router ospf
network 172.17.17.8/30 area 0
```

3.7

Στον R3 εκτελούμε:

```
router ospf
router-id 0.0.0.3

network 0.0.0.0/0 area 0
```

3.8

Στον R1 εκτελούμε `do show ip ospf route`. Έχουμε:

```
Network          Cost
127.0.0.1/32     20
172.17.17.0/30   1
172.17.17.4/30   10
172.17.17.8/30   11
192.168.1.0/24   10
192.168.2.0/24   11
```

3.9

Στον R2 εκτελούμε `do show ip ospf route`. Έχουμε:

```
Network          Cost
127.0.0.1/32     20
172.17.17.0/30   1
172.17.17.4/30   11
172.17.17.8/30   10
192.168.1.0/24   11
192.168.2.0/24   10
```

3.10

Στον R3 εκτελούμε `do show ip ospf route`. Έχουμε:

```
Network          Cost
172.17.17.0/30   11
172.17.17.4/30   10
172.17.17.8/30   10
192.168.1.0/24   20
192.168.2.0/24   20
```

3.11

Με την εντολή `network 0.0.0.0/0 area 0` ενεργοποιείται το OSPF σε όλες τις διεπαφές του R3, οπότε αυτός διαφημίζει τα υποδίκτυα `172.17.17.4/30 (em0)`, `172.17.17.8/30 (em1)` και `127.0.0.1/32 (lo0)`.

3.12

Η πηγή αυτής της πληροφορίας είναι ο R3, όπως μπορούμε να δούμε και από την εκτέλεση της εντολής:

```
do show ip ospf database router
```

3.13

Στον R1 κάνουμε `do ping 127.0.0.1`. Όπως φαίνεται και από το TTL=64, απαντά ο ίδιος ο R1, αφού η διεύθυνση `127.0.0.1` αναφέρεται στον localhost, ο οποίος προτιμάται ως άμεσα συνδεδεμένος στον R1. Αυτό μπορούμε να το επιβεβαιώσουμε είτε εκτελώντας:

```
do traceroute 127.0.0.1
```

είτε βλέποντας τον πίνακα δρομολόγησης με:

```
do show ip route
```

όπου βλέπουμε επιλεγμένη διαδρομή για το `127.0.0.0/8` μέσω της `lo0`.

3.14

Εκτελούμε στον R3:

```
do show ip route ospf
```

Έχει 2 διαδρομές, από τις οποίες έχει επιλεχθεί για τον πίνακα προώθησης η διαδρομή μέσω του R1.

3.15

Στον R3 εκτελούμε:

```
do show ip ospf neighbor
```

Βλέπουμε ότι οι R1 και R2 είναι DROther.

3.16

Στους R1,2,3 εκτελούμε:

```
do show ip ospf database
```

Περιέχονται μόνο Router LSA. Δεν υπάρχει πληροφορία για Network LSA γιατί έχουμε δηλώσει λειτουργία point-to-point σε όλες τις διεπαφές των δρομολογητών στα WAN.

3.17

Εκτελούμε `do show ip ospf database router router-adv 0.0.0.1`. Η σύνδεσή του στο WAN1 περιγράφεται ως Stub Network.

3.18

Στο PC2 εκτελούμε `do ping 192.168.1.2`. Η τιμή του TTL είναι 62.

3.19

Στον R2 εκτελούμε σε νέο παράθυρο `tcpdump -vni em2 "not icmp"`.

3.20

Αποσυνδέουμε το καλώδιο της διεπαφής του R1 στο WAN1. Το ping δεν διακόπτεται καθόλου, επανέρχεται κατευθείαν. Σταματάμε το ping. Δεν υπάρχουν καθόλου χαμένα πακέτα, ενώ το TTL αρχικά ήταν 62 και μετά τη διακοπή της σύνδεσης έγινε 61.

3.21

Κρίνοντας και από τα αποτελέσματα του ping, ο χρόνος αντίδρασης του OSPF σε αλλαγές της τοπολογίας του δικτύου είναι άμεσος (μικρότερος από 1 δευτερόλεπτο στην συγκεκριμένη περίπτωση), κατά πολύ μικρότερος σε σχέση με το RIP (~30 δευτερόλεπτα, βλ. Εργαστηριακή Άσκηση 7)

3.22

Σταματάμε το tcpdump. Πλην των μηνυμάτων Hello, ανταλλάχθηκαν 3 μηνύματα LS-Update και 3 αντίστοιχα μηνύματα LS-Ack.

3.23

Διήρκησε ~34 δευτερόλεπτα.

3.24

Στον R1 εκτελούμε:

```
do show ip route
```

Έχουμε:

```
Network                 Cost
WAN1 (172.17.17.0/30)   21
WAN3 (172.17.17.8/30)   20
LAN2 (192.168.2.0/24)   30
```

3.25

Στον R2 εκτελούμε:

```
do show ip route
```

Έχουμε:

```
Network                 Cost
WAN1 (172.17.17.0/30)   1
WAN2 (172.17.17.4/30)   20
LAN1 (192.168.1.0/24)   30
```

3.26

Στον R3 εκτελούμε:

```
do show ip route
```

Πλέον έχει διαγραφεί μία από τις δύο διαδρομές, αυτή μέσω της διεπαφής `172.17.17.5` (R1), και ως εκ τούτου έχει επιλεχθεί η διαδρομή μέσω της διεπαφής `172.17.17.9` (R2), η οποία έχει μπει και στο FIB.

3.27

Επειδή ο R1 έχει άμεσα συνδεδεμένη την `em1` στο WAN1, και η διαδρομή αυτή έχει μηδενική διαχειριστική απόσταση, σε αντίθεση με την διαδρομή OSPF που έχει διαχειριστική απόσταση 110.

3.28

Αποσυνδέουμε το καλώδιο της διεπαφής του R2 στο WAN1 και εκτελούμε `do show ip route` στους R1,2,3. Βλέπουμε ότι από τους πίνακες δρομολόγησης έχουν διαγραφεί οι εγγραφές που αφορούν το `172.17.17.0/30`

3.29

Ξεκινάμε πάλι `do ping 192.168.1.2` από το PC2. Επανασυνδέουμε τα καλώδια του WAN1. Η ενημέρωση των πινάκων δρομολόγησης δεν είναι άμεση, κάτι που φαίνεται και από το ότι αργεί να αλλάξει το TTL από 61 σε 62 στην έξοδο της εντολής ping.

3.30

Διότι όταν έχουμε αποσύνδεση του καλωδίου, στέλνεται ένα LS-Update κατευθείαν με σκοπό να είναι όσο το δυνατόν πιο άμεση η αποκατάσταση της επικοινωνίας. Αντίθετα, όταν ήδη υπάρχει επικοινωνία:

- Δεν υπάρχει λόγος να γίνει άμεση αλλαγή στην ζεύξη που επανήλθε.
- Είναι συνετό να υπάρχει κάποια αναμονή ώστε να επιβεβαιωθεί η σταθερότητα της ζεύξης που επανήλθε.

Άσκηση 4: Περιοχές OSPF

4.1

Εκτελούμε:

```
### PC1 ###

vtysh
configure terminal
```

```
hostname PC1

interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1

### PC2 ###

vtysh
configure terminal

hostname PC2

interface em0
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1
```

4.2

Εκτελούμε:

```
### R1 ###

cli
configure terminal

hostname R1

interface lo0
ip address 172.22.22.1/32

### R2 ###

cli
configure terminal

hostname R2

interface lo0
ip address 172.22.22.2/32

### R3 ###

cli
configure terminal
```

```
hostname R3

interface lo0
ip address 172.22.22.3/32

### R4 ###

cli
configure terminal

hostname R4

interface lo0
ip address 172.22.22.4/32

### R5 ###

cli
configure terminal

hostname R5

interface lo0
ip address 172.22.22.5/32
```

4.3

Εκτελούμε:

```
### R1 ###

interface em0
link-detect

interface em1
link-detect

### R2 ###

interface em0
link-detect

interface em1
link-detect

### R3 ###

interface em0
link-detect
```

```
interface em1
link-detect

### R4 ###

interface em0
link-detect

### R5 ###

interface em0
link-detect
```

4.4

Στον R1 εκτελούμε:

```
interface em0
ip address 10.1.1.1/30

interface em1
ip address 10.1.1.5/30

router ospf

network 10.1.1.0/30 area 0
network 10.1.1.4/30 area 0
```

4.5

Στον R2 εκτελούμε:

```
interface em0
ip address 10.1.1.2/30

interface em1
ip address 10.1.1.9/30

router ospf

network 10.1.1.0/30 area 0
network 10.1.1.8/30 area 1
```

4.6

Στον R3 εκτελούμε:

```
interface em0
```

```
ip address 10.1.1.6/30

interface em1
ip address 10.1.1.13/30

router ospf

network 10.1.1.4/30 area 0
network 10.1.1.12/30 area 2
```

4.7

Στον R4 εκτελούμε:

```
interface em0
ip address 10.1.1.10/30

interface em1
ip address 192.168.1.1/24

router ospf

network 192.168.1.0/24 area 1
network 10.1.1.8/30 area 1
```

4.8

Στον R5 εκτελούμε:

```
interface em0
ip address 10.1.1.14/30

interface em1
ip address 192.168.2.1/24

router ospf

network 192.168.2.0/24 area 2
network 10.1.1.12/30 area 2
```

4.9

Στον PC1 εκτελούμε `do ping 192.168.2.2`. Το ping είναι επιτυχές.

4.10

Στους R1,2,3,4,5 εκτελούμε `do show ip ospf`. Έχουμε:

```
R1 Router-ID: 172.22.22.1
R2 Router-ID: 172.22.22.2
R3 Router-ID: 172.22.22.3
R4 Router-ID: 172.22.22.4
R5 Router-ID: 172.22.22.5
```

4.11

Εκτελούμε `do show ip ospf neighbor` στους R1,2,3,4,5. Έχουμε:

```
### WAN1 ###

Designated: R1
Backup: R2

### WAN2 ###

Designated: R1
Backup: R3

### WAN3 ###

Designated: R2
Backup: R4

### WAN4 ###

Designated: R3
Backup: R5
```

Το αποτέλεσμα δεν είναι το αναμενόμενο με βάση το προηγούμενο ερώτημα (δηλαδή δεν είναι DR οι δρομολογητές με μεγαλύτερο Router-ID), επειδή το OSPF ενεργοποιήθηκε στους δρομολογητές με την σειρά R1 $\rightarrow$ R2 $\rightarrow$ R3 $\rightarrow$ R4 $\rightarrow$ R5. Επομένως ο R1 μπήκε ως DR στις ζεύξεις WAN1 και WAN2, ο R2 στην WAN3, και ο R3 στην WAN4. Αφού οριστεί ένας DR, δεν αντικαθίσταται από έναν καταλληλότερο DR (με μεγαλύτερο Router-ID), εκτός κι αν υπάρξει απώλεια του DR.

4.12

Στους R1,2,3,4,5 εκτελούμε:

```
do show ip ospf border-routers
```

Βρίσκουμε τους ABR για κάθε περιοχή:

```
Area 0: R2, R3
Area 1: R2
Area 2: R3
```

4.13

Στον R1 εκτελούμε `do show ip ospf database`. Σε σχέση με την Άσκηση 2, βλέπουμε επιπλέον τα Summary Link States.

4.14

Έχει 9 LSA, εκ των οποίων 3 είναι Router LSA, 2 είναι Network LSA, και 4 είναι Summary LSA. Τα Router LSA είναι 3 και όχι 5 διότι ο κάθε δρομολογητής καταγράφει τα Router LSA της περιοχής του.

4.15

Εκτελούμε `do show ip ospf database self-originate` στον R1. Από τον R1 πηγάζουν τα ακόλουθα LSA:

```
Router Link States

Link ID
172.22.22.1

Net Link States

Link ID
10.1.1.1
10.1.1.5
```

4.16

Στον R1 εκτελούμε `do show ip ospf database router` και παρατηρούμε ότι το Link ID και για τα τρία Router LSA προκύπτει από το Router-ID του δρομολογητή που τα παρήγαγε (`172.22.22.1, 172.22.22.2, 172.22.22.3`).

4.17

Στον R2 εκτελούμε `do show ip ospf database`. Η LSDB του R2 περιέχει LSA για τις περιοχές 0 και 1.

4.18

Στον R2 εκτελούμε `do show ip ospf database`. Έχουμε:

```
Total LSA: 16

### Area 0 ###

Router LSA: 3
Network LSA: 2
Summary LSA: 4
```

```
### Area 1 ###

Router LSA: 2
Network LSA: 1
Summary LSA: 4
```

Στην περιοχή 0 υπάρχουν 2 Network LSA, διότι υπάρχουν 2 ζεύξεις μεταξύ δρομολογητών, οπότε οι διεπαφές που είναι DR στέλνουν Network LSA διαφημίζοντας τη λίστα των δρομολογητών που είναι συνδεδεμένοι στην ίδια ζεύξη με αυτούς.

Αντίστοιχα, στην περιοχή 1 υπάρχει 1 Network LSA, διότι υπάρχει 1 ζεύξη μεταξύ δρομολογητών, οπότε η διεπαφή που είναι DR στέλνει Network LSA διαφημίζοντας τη λίστα των δρομολογητών που είναι συνδεδεμένοι στην ίδια ζεύξη με αυτόν.

4.19

Στον R2 εκτελούμε `do show ip ospf database network` και βλέπουμε ότι το Link ID των Network LSA ταυτίζεται με τη διεύθυνση IP της αντίστοιχης διεπαφής του DR στην εκάστοτε ζεύξη:

```
### Area 0 ###

Link-ID: 10.1.1.1
DR of 10.1.1.0/30: R1 (10.1.1.1)

Link-ID: 10.1.1.5
DR of 10.1.1.4/30: R1 (10.1.1.5)

### Area 1 ###

Link-ID: 10.1.1.9
DR of 10.1.1.8/30: R2 (10.1.1.9)
```

4.20

Στον R3 εκτελούμε `do show ip ospf database`. Έχουμε:

```
Total LSA: 16

### Area 0 ###

Router LSA: 3
Network LSA: 2
Summary LSA: 4

### Area 2 ###

Router LSA: 2
```

```
Network LSA: 1
Summary LSA: 4
```

Στις περιοχές 0 και 2 υπάρχουν 4 Summary LSA, επειδή σε κάθε περιοχή υπάρχουν 4 ζεύξεις/δίκτυα εκτός της περιοχής αυτής. Συγκεκριμένα:

```
Area 0: LAN1, LAN2, WAN3, WAN4
Area 2: LAN1, WAN1, WAN2, WAN3
```

Για αυτές τις ζεύξεις ο R3 ενημερώνεται μέσω των ABR R2, R3.

4.21

Στον R3 εκτελούμε `do show ip ospf database summary` και παρατηρούμε ότι το Link ID των Summary LSA ταυτίζεται με τον αριθμό δικτύου προορισμού.

```
### Area 0 ###

Link-ID: 10.1.1.8 == WAN3 Network Number

Link-ID: 10.1.1.12 == WAN4 Network Number

Link-ID: 192.168.1.0 == LAN1 Network Number

Link-ID: 192.168.2.0 == LAN2 Network Number

### Area 2 ###

Link-ID: 10.1.1.0 == WAN1 Network Number

Link-ID: 10.1.1.4 == WAN2 Network Number

Link-ID: 10.1.1.8 == WAN3 Network Number

Link-ID: 192.168.1.0 == LAN1 Network Number
```

4.22

Στον R1 εκτελούμε:

```
do show ip ospf database
```

Τα Router LSA πηγάζουν από τους R1, R2, R3, ενώ τα Network LSA πηγάζουν από τον R1.

4.23

Στον R2 εκτελούμε `do show ip ospf database`. Οι πηγές διαφήμισης των Summary LSA της LSDB του R2 για την περιοχή 0 είναι οι R2 και R3, ενώ για την περιοχή 1 είναι ο R2.

4.24

Στον R1 εκτελούμε `do show ip ospf route`. Για τις διαδρομές μεταξύ περιοχών υπάρχει η ένδειξη `"IA"` (Inter-Area).

4.25

Στον R1 εκτελούμε `do show ip route ospf`. Δεν υπάρχει αντίστοιχη ένδειξη.

4.26

Στον R1 εκτελούμε `do show ip ospf route`. Εκτός από διαδρομές προς δίκτυα, υπάρχουν και διαδρομές προς δρομολογητές (ένδειξη `"R"`).

4.27

Ναι, υπάρχει η ένδειξη `"ABR"`.

Άσκηση 5: OSPF και αναδιανομή διαδρομών

5.1

Στον R3 εκτελούμε:

```
ip route 5.5.5.0/24 lo0
ip route 6.6.6.0/24 lo0
```

5.2

Στον R3 εκτελούμε `do show ip route`. Οι εγγραφές έχουν τοποθετηθεί στον πίνακα δρομολόγησης του R3. Ύστερα στον R3 εκτελούμε `do show ip ospf route`. Οι εγγραφές δεν εμφανίζονται στον πίνακα διαδρομών OSPF.

5.3

Στους R1,2,4,5 εκτελούμε `do show ip route`. Οι εγγραφές δεν έχουν τοποθετηθεί στον πίνακα δρομολόγησης των άλλων δρομολογητών.

5.4

Στον R3 εκτελούμε:

```
router ospf
redistribute static

do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα δρομολόγησης του R3.

5.5

Στους R1,2,4,5 εκτελούμε `do show ip route`. Βλέπουμε ότι οι εγγραφές για τα δίκτυα `5.5.5.0/24` και `6.6.6.0/24` έχουν προστεθεί στους άλλους δρομολογητές ως OSPF διαδρομές.

5.6

Στους R1,2,4,5 εκτελούμε `do show ip ospf route`. Ο πίνακας διαδρομών OSPF περιέχει επίσης εξωτερικές (external) διαδρομές.

5.7

Είναι είδους E2, όπως φαίνεται και από την αντίστοιχη ένδειξη. Από τις δύο τιμές κόστους στον πίνακα διαδρομών OSPF (μέσα σε αγκύλες), η πρώτη είναι το κόστος εντός του δικτύου OSPF, ενώ η δεύτερη είναι το κόστος προς προορισμό.

5.8

Στους R1,2,4,5 εκτελούμε `do show ip ospf route`. Έχουμε:

```
### R1 ###

R3: ABR, ASBR

### R2 ###

R3: ABR, ASBR

### R4 ###

R3: ASBR

### R5 ###

R3: ABR, ASBR
```

5.9

Στον R1 εκτελούμε `do show ip ospf database`. Αυτή τη φορά εμφανίζονται επιπλέον AS External LSA.

5.10

Στον R1 εκτελούμε `do show ip ospf database external`. Παρατηρούμε ότι το Link-ID των External LSA ταυτίζεται με τον αριθμό δικτύου του εξωτερικού δικτύου προορισμού. Συγκεκριμένα:

```
Link-ID: 5.5.5.0 == 5.5.5.0/24 Network Number

Link-ID: 6.6.6.0 == 6.6.6.0/24 Network Number
```

5.11

Στον R4 εκτελούμε `do show ip ospf database`. Αυτή τη φορά εμφανίζονται όχι μόνο AS External LSA, αλλά και ASBR-Summary LSA.

5.12

Στον R4 εκτελούμε `do show ip ospf database asbr-summary`. Παρατηρούμε ότι το Link-ID του LSA ταυτίζεται με το Router-ID του αντίστοιχου ASBR που διαφημίζεται. Συγκεκριμένα:

`Link-ID: 172.22.22.3 == R3 Router-ID`

5.13

Είναι ο R2, όπως φαίνεται και από το πεδίο `Advertising Router: 172.22.22.2`.

5.14

Διότι ο R3 (ASBR) και ο R5 βρίσκονται στην ίδια περιοχή, ενώ τα ASBR Summary LSA εκπέμπονται από τον ABR για να διαφημίσουν την παρουσία ASBR προς τις *άλλες* περιοχές, και όχι στην ίδια περιοχή.

5.15

Στον R2 εκτελούμε:

```
ip route 0.0.0.0/0 172.22.22.2
```

5.16

Στον R2 εκτελούμε:

```
do show ip route
do show ip ospf route
```

Η διαδρομή εμφανίζεται στον πίνακα δρομολόγησης, αλλά όχι στον πίνακα διαδρομών OSPF.

5.17

Στους R1,3,4,5 εκτελούμε `do show ip route`. Η εγγραφή δεν έχει τοποθετηθεί στους άλλους δρομολογητές.

5.18

Στον R2 εκτελούμε:

```
router ospf
default-information originate

do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα δρομολόγησης του R2.

5.19

Στους R1,3,4,5 εκτελούμε `do show ip route`. Σε κάθε δρομολογητή έχει τοποθετηθεί OSPF διαδρομή με προορισμό το `0.0.0.0/0` και επόμενο βήμα τέτοιο ώστε να ακολουθείται διαδρομή προς τον R2.

5.20

Στους R1,3,4,5 εκτελούμε `do show ip ospf route`. Η προκαθορισμένη διαδρομή χαρακτηρίζεται στον πίνακα διαδρομών OSPF ως external.

5.21

Είναι είδους E2, όπως φαίνεται και από την σχετική ένδειξη. Από τις δύο τιμές κόστους στον πίνακα διαδρομών OSPF (μέσα σε αγκύλες), η πρώτη είναι το κόστος εντός του δικτύου OSPF, ενώ η δεύτερη είναι το κόστος προς τον προορισμό.

5.22

Στους R1,3,4,5 εκτελούμε `do show ip ospf route`. Έχουμε:

```
### R1 ###

R2: ABR, ASBR

### R3 ###

R2: ABR, ASBR

### R4 ###

R2: ABR, ASBR

### R5 ###

R2: ASBR
```

5.23

Στον R5 εκτελούμε `do show ip ospf database`. Υπάρχουν πλέον ASBR-Summary LSA, στην LSDB του R5, επειδή διαφημίζεται ο ASBR R2, που δεν ανήκει στην ίδια περιοχή με τον R5.

5.24

Στους R1,2,3,4,5 εκτελούμε `do show ip ospf database`. Σε όλους τους δρομολογητές υπάρχουν 3 εγγραφές External LSA, διότι διαφημίζονται 3 εξωτερικά δίκτυα από τους ASBR, τα `0.0.0.0/0`, `5.5.5.0/24` και `6.6.6.0/24`.

5.25

Στον R1 εκτελούμε `do show ip ospf database external`. Η τιμή του κόστους για τις εξωτερικές διαδρομές είναι ίση με 20.

5.26

Το Metric Type έχει τιμή 2, που αντιστοιχεί σε Ext type 2 (E2). Στο E2, ο ASBR καθορίζει το κόστος της διαδρομής προς τον προορισμό, και το κόστος εντός του δικτύου OSPF αγνοείται. Αυτό εξηγεί και αυτά που παρατηρήσαμε στα ερωτήματα 5.7 και 5.21, όπου τα κόστη προς τον προορισμό ήταν σταθερά για όλους τους δρομολογητές.

5.27

Στον R4 εκτελούμε `do show ip ospf route` και βλέπουμε ότι το κόστος της διαδρομής OSPF από τον R4 στον R3 είναι 30.

5.28

Στον R4 εκτελούμε `do show ip ospf database asbr-summary`. Το κόστος που παρατηρούμε αφορά τη διαδρομή R2 $\rightarrow$ R1 $\rightarrow$ R3.

Άσκηση 6: OSPF και περιοχές απόληξης

6.1

Στον PC1 εκτελούμε `do ping 192.168.2.2`.

6.2

Στον R3 εκτελούμε `do show ip route ospf`. Έχουμε:

```
O>* 0.0.0.0/0 [110/10] via 10.1.1.5, em0
O>* 10.1.1.0/30 [110/20] via 10.1.1.5, em0
O   10.1.1.4/30 [110/10] is directly connected, em0
O>* 10.1.1.8/30 [110/30] via 10.1.1.5, em0
O   10.1.1.12/30 [110/10] is directly connected, em1
O>* 192.168.1.0/24 [110/40] via 10.1.1.5, em0
O>* 192.168.2.0/24 [110/20] via 10.1.1.14, em1
```

6.3

Στον R5 εκτελούμε `do show ip route ospf`. Έχουμε:

```
O>* 0.0.0.0/0 [110/10] via 10.1.1.13, em0
O>* 5.5.5.0/24 [110/20] via 10.1.1.13, em0
O>* 6.6.6.0/24 [110/20] via 10.1.1.13, em0
O>* 10.1.1.0/30 [110/30] via 10.1.1.13, em0
O>* 10.1.1.4/30 [110/20] via 10.1.1.13, em0
```

```
O>* 10.1.1.8/30 [110/40] via 10.1.1.13, em0
O   10.1.1.12/30 [110/10] is directly connected, em0
O>* 192.168.1.0/24 [110/50] via 10.1.1.13, em0
O   192.168.2.0/24 [110/10] is directly connected, em1
```

6.4

Στον R5 εκτελούμε `do show ip ospf database router self-originate`. Το δίκτυο του LAN2 χαρακτηρίζεται ως Stub, ενώ του WAN4 ως Transit.

6.5

Στον R3 εκτελούμε `area 2 stub` και περιμένουμε να διαδοθεί η αλλαγή. Μετά από περίπου 30 δευτερόλεπτα εμφανίζεται μήνυμα "Time to live exceeded".

6.6

Στον R3 εκτελούμε `do show ip route`. Παρατηρούμε ότι έχει διαγραφεί η εγγραφή με προορισμό του δίκτυο `192.168.2.0/24`.

6.7

Στον R5 εκτελούμε `do show ip route`. Ο πίνακας δρομολόγησης του R5 περιέχει διαδρομές για τα δίκτυα:

```
10.1.1.12/30
127.0.0.0/8
172.22.22.5/32
192.168.2.0/24
```

6.8

Στους R1,2,3,4 εκτελούμε `do show ip route`. Δεν υπάρχει διαδρομή για το LAN2 στον πίνακα δρομολόγησης κανενός δρομολογητή.

6.9

Στον PC1 εκτελούμε `do traceroute 192.168.2.2`. Η έξοδος της εντολής είναι:

```
1   192.168.1.1
2   10.1.1.9
3   10.1.1.9
4   10.1.1.9
...
63  10.1.1.9
64  10.1.1.9
```

Βλέπουμε ότι το ICMP Request ξεκινά από τον PC1, φτάνει στον R4 και ύστερα ανακυκλώνεται στον R2, επειδή -ελλείψει άλλης διαδρομής προς το LAN2- επιλέγεται συνεχώς η προκαθορισμένη διαδρομή μέσω της loopback του R2. Έτσι δεν φτάνει ποτέ στον PC2.

6.10

Για τον ίδιο λόγο με το 6.9, το ICMP Request ανακυκλώνεται στον R2 μέχρι να μηδενιστεί το TTL του, εξού και το μήνυμα λάθους "Time to live exceeded".

6.11

Στον R3 εκτελούμε `do show ip ospf database router`. Παρατηρούμε ότι στην περιοχή 2, το E-bit είναι ίσο με 0 στο Router LSA του R3, ενώ είναι ίσο με 1 στο Router LSA του R5. Αυτό συμβαίνει επειδή έχουμε ορίσει την περιοχή 2 ως stub στον R3 αλλά όχι στον R5.

6.12

Στον R3 εκτελούμε `do show ip ospf`. Έχουμε:

```
Area ID: 0.0.0.2 (Stub)
```

6.13

Στον R5 εκτελούμε `area 2 stub` και μετά από λίγο το ping λειτουργεί κανονικά.

6.14

Στον R3 εκτελούμε `do show ip route` και βλέπουμε ότι έχει προστεθεί OSPF εγγραφή με προορισμό το `192.168.2.0/24`.

6.15

Στον R5 εκτελούμε `do show ip ospf database router` και βλέπουμε ότι το E-bit είναι 0 και στον R5 πλέον, εξαιτίας της αλλαγής που κάναμε στο 6.13.

6.16

Στον R5 εκτελούμε `do show ip route`. Υπάρχει εγγραφή για την προκαθορισμένη διαδρομή μέσω του R3 (`10.1.1.13`).

6.17

Όχι, δεν υπάρχουν εγγραφές προς τα `5.5.5.0/24` και `6.6.6.0/24`.

6.18

Περιέχει διαδρομές προς τα δίκτυα:

```
10.1.1.0/30        inter-area
10.1.1.4/30        inter-area
10.1.1.8/30        inter-area
10.1.1.12/30       intra-area
192.168.1.0/24     inter-area
192.168.2.0/24     intra-area
```

6.19

Στους R1,2,3,4 εκτελούμε: `do show ip route`. Παρατηρούμε ότι έχει προστεθεί ξανά η εγγραφή με προορισμό το `192.168.2.0/24`.

6.20

Όταν ορίσαμε στον R3 την περιοχή 2 ως απόληξη, στον R5 ακόμα ήταν ορισμένη ως κανονική, οπότε οι δρομολογητές R3 και R5 δεν μπορούσαν να φτάσουν σε κατάσταση *2-way*, ώστε να ανταλλάξουν δεδομένα δρομολόγησης OSPF, και έτσι η διαδρομή στο LAN2 δεν γινόταν γνωστή στον R3. Λογικό είναι λοιπόν το ping να αποτυγχάνει αρχικά. Ύστερα όμως, όταν ορίζουμε και στον R5 την περιοχή 2 ως απόληξη, η πληροφορία δρομολόγησης μεταξύ R3 και R5 ανταλλάσεται επιτυχώς και όλο το δίκτυο μαθαίνει για την παρουσία του LAN2, οπότε το ping είναι και πάλι επιτυχές.

6.21

Αυτό συμβαίνει διότι στην περίπτωση του R4, η προκαθορισμένη διαδρομή πρόκειται γι' αυτήν που διένειμε ο R2, οπότε αναγράφεται ως εξωτερική, ενώ στην περίπτωση του R5, η προκαθορισμένη διαδρομή πρόκειται γι' αυτήν που ορίστηκε επειδή η περιοχή 2 είναι πλέον απόληξη, και έχει σκοπό την δρομολόγηση όλων των πακέτων που προορίζονται για εξωτερικά δίκτυα μέσω της προκαθορισμένης πύλης.

Πιο συγκεκριμένα, στον R4 αυτή η διαδρομή διανεμήθηκε από τον R2, στον οποίο είχε οριστεί ως στατική (γι' αυτό και χαρακτηρίζεται ως εξωτερική), ενώ στον R5 αυτή η διαδρομή διαφημίστηκε από τον R3 (ως ABR) μέσω Summary LSA.

6.22

Στον R3 εκτελούμε `do show ip ospf database self-originate` και βλέπουμε ότι τη διαφημίζει με κόστος 1.

6.23

Στον R5 εκτελούμε `do show ip ospf route`. Η προκαθορισμένη διαδρομή έχει κόστος 11. Αυτό συμβαίνει επειδή το προκαθορισμένο κόστος για την προκαθορισμένη διαδρομή στις περιοχές απολήξεις είναι 1, επομένως για την προκαθορισμένη διαδρομή ο R5 "βλέπει" κόστος:

```
(R3 advertised cost for default gateway) + (Cost from R5 to R3) = 1 + 10 = 11.
```

6.24

Στον R5 εκτελούμε `do show ip ospf database`. Εμφανίζονται εγγραφές για εξωτερικές εγγραφές, οι οποίες θα διαγραφούν μετά την παρέλευση 3600 δευτερολέπτων από την τελευταία ενημέρωση τους.

6.25

Στους R3 και R5 εκτελούμε `no area 2 stub`.

6.26

Πρέπει να προσθέσουμε το `no-summary` στο τέλος της εντολής, ως εξής:

```
area 2 stub no-summary
```

6.27

Στον R3 εκτελούμε `area 2 stub no-summary` ενώ στον R5 `area 2 stub`.

6.28

Στον R5 εκτελούμε `do show ip ospf route`. Ο πίνακας του R5 περιέχει εγγραφές για τα δίκτυα:

```
0.0.0.0/0 (default route)
10.1.1.12/30
192.168.2.0/24
```

6.29

Στο PC2 εκτελούμε:

```
no ip route 0.0.0.0/0 192.168.2.1

router ospf

network 192.168.2.0/24 area 2

area 2 stub
```

6.30

Στο PC2 εκτελούμε `do show ip route`. Περιέχονται οι δυναμικές εγγραφές:

```
O>* 0.0.0.0/0 [110/111] via 192.168.2.1, em0
O>* 10.1.1.12/30 [110/110] via 192.168.2.1, em0
O   192.168.2.0/24 [110/100] is directly connected, em0
```

6.31

Στον R5 εκτελούμε `do show ip ospf database router self-originate`. Πλέον το LAN2 χαρακτηρίζεται ως Transit, επειδή έχουμε δύο OSPF δρομολογητές στο δίκτυο `192.168.2.0/24`.

6.32

Συμπέρασμα των παραπάνω είναι ότι δίκτυο απόληξη και περιοχή απόληξη διαφέρουν μεταξύ τους. Δίκτυο απόληξη είναι αυτό που έχει μόνο έναν OSPF δρομολογητή και ως εκ τούτου τα πακέτα OSPF είτε πηγάζουν είτε καταλήγουν σε αυτό, ενώ περιοχή απόληξη είναι αυτή στην οποία οι πίνακες των δρομολογητών περιέχουν όλες τις εσωτερικές διαδρομές και μία προκαθορισμένη

διαδρομή για όλους τους προορισμούς εκτός δικτύου OSPF.

Επίσης, η έννοια του δικτύου απόληξη είναι κάτι που προκύπτει από την κατασκευή του δικτύου (εξαρτάται από το πόσοι δρομολογητές υπάρχουν σε ένα δίκτυο και σε πόσους από αυτούς το OSPF είναι ενεργοποιημένο), ενώ η έννοια της περιοχής απόληξη είναι κάτι που ορίζεται χειροκίνητα σε όλους τους δρομολογητές (αρκεί να εκτελεστεί μια εντολή σε όλους τους δρομολογητές της περιοχής).

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 9
Δυναμική δρομολόγηση BGP

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 10/05/2022

Άσκηση 1: Εισαγωγή στο BGP

1

Εκτελούμε `service frr stop`.

2

Εκτελούμε `touch /usr/local/etc/frr/bgpd.conf`.

3

Εκτελούμε `chown frr:frr /usr/local/etc/frr/bgpd.conf`.

4

Αλλάζουμε τη ζητούμενη γραμμή σε `frr_daemons="zebra staticd ripd ospfd bgpd`.

5

Εκτελούμε `service frr start`.

1.1

Εκτελούμε:

```
### PC1 ###

vtysh
configure terminal

hostname PC1
```

```
interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1

### PC2 ###

vtysh
configure terminal

hostname PC2

interface em0
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1
```

1.2

Εκτελούμε:

```
### R1 ###

cli
configure terminal

hostname R1

interface em0
ip address 192.168.1.1/24

interface em1
ip address 10.1.1.1/30

### R2 ###

cli
configure terminal

hostname R2

interface em0
ip address 10.1.1.2/30

interface em1
ip address 10.1.1.5/30

### R3 ###
```

```
cli
configure terminal

hostname R3

interface em0
ip address 10.1.1.6/30

interface em1
ip address 192.168.2.1/24
```

1.3

Στον R1 εκτελούμε `do show ip route`. Δεν υπάρχει καμία στατική εγγραφή.

1.4

Στον R1 εκτελούμε:

```
exit       # To enter global configuration mode
router ?   # BGP is available
```

1.5

Στον R1 εκτελούμε `router bgp 65010`.

1.6

Στον R1 πατάμε το πλήκτρο `"?"`. Εμφανίζονται 14 διαθέσιμες εντολές.

1.7

Στον R1 εκτελούμε `neighbor 10.1.1.2 remote-as 65020`.

1.8

Στον R1 εκτελούμε `network 192.168.1.0/24`.

1.9

Στον R1 εκτελούμε:

```
exit
# wait for ~1 min
do show ip route
```

Δεν έχει αλλάξει κάτι στον πίνακα δρομολόγησης του R1.

1.10

Στους R1 και R2 εκτελούμε `do show ip bgp`. Στον R1 έχει προστεθεί μία εγγραφή στον πίνακα διαδρομών BGP με δίκτυο προορισμού το `192.168.1.0`, ενώ στον R2 εμφανίζεται μήνυμα λάθους "No BGP process is configured".

1.11

Στον R2 εκτελούμε `router bgp 65020`.

1.12

Στον R2 εκτελούμε:

```
neighbor 10.1.1.1 remote-as 65010
neighbor 10.1.1.6 remote-as 65030
```

1.13

Στον R2 εκτελούμε `exit` και περιμένουμε περίπου ένα λεπτό. Ύστερα στους R1 και R2 εκτελούμε `do show ip bgp`. Στον R1 δεν έχει αλλάξει κάτι, το οποίο είναι λογικό, αφού ο R2 δεν διαφημίζει κάποιο δίκτυο, ενώ στον R2 προστέθηκε μία εγγραφή με δίκτυο προορισμού το `192.168.1.0/24`, την οποία ο R2 έμαθε από τον R1.

1.14

Στον R3 εκτελούμε `do show ip route`. Δεν υπάρχει διαδρομή για το `192.168.1.0/24`.

1.15

Στον R3 εκτελούμε `router bgp 65030`.

1.16

Στον R3 εκτελούμε `neighbor 10.1.1.5 remote-as 65020`.

1.17

Στον R3 εκτελούμε `network 192.168.2.0/24`.

1.18

Στον R3 εκτελούμε `exit` και περιμένουμε περίπου ένα λεπτό. Ύστερα στους R1,2,3 εκτελούμε `do show ip bgp`. Πλέον και στους τρεις δρομολογητές υπάρχουν εγγραφές για τους προορισμούς `192.168.1.0/24` και `192.168.2.0/24`.

1.19

Στον R2 εκτελούμε `do show ip route`. Οι εγγραφές BGP ξεχωρίζουν από το γράμμα `"B"` στην αρχή της γραμμής.

1.20

Δηλώνονται με το σύμβολο "*" στην αρχή της γραμμής.

1.21

Είναι 20, όπως αναγράφεται και στις εγγραφές του πίνακα δρομολόγησης, στον πρώτο αριθμό μέσα σε αγκύλες.

1.22

Στον R1 εκτελούμε `do show ip route bgp`. Βλέπουμε μία εγγραφή BGP.

1.23

Στον R1 εκτελούμε `do show ip bgp`. Βλέπουμε δύο εγγραφές, ενώ σε σχέση με τον πίνακα δρομολόγησης εμφανίζεται η επιπλέον πληροφορία `Local Preference`, `Weight`, και `Path`.

1.24

```
NETWORK          NEXT_HOP   WEIGHT   AS_PATH
192.168.1.0/24   0.0.0.0    32768    i
192.168.2.0/24   10.1.1.2   0        65020 65030 i
```

1.25

Στους δρομολογητές Quagga/FRR οι διαδρομές που πηγάζουν από τον δρομολογητή έχουν προκαθορισμένη τιμή `WEIGHT` ίση με 32768, ενώ όλες οι άλλες έχουν βάρος 0. Επομένως, η διαδρομή για το δίκτυο `192.168.1.0/24` πηγάζει από τον R1 και έχει βάρος 32768, ενώ η διαδρομή για το δίκτυο `192.168.2.0/24` *δεν* πηγάζει από τον R1 και έχει βάρος 0.

1.26

Παριστάνει τον τύπο πηγής `ORIGIN (IGP)`, αφού οι διαδρομές έγιναν γνωστές μέσω της εντολής `network` στο αντίστοιχο AS.

1.27

Στον R1 εκτελούμε:

```
exit
exit
netstat -rn
```

Υπάρχει μία δυναμική εγγραφή:

```
Destination      Gateway    Flags   Netif
192.168.2.0/24   10.1.1.2   UG1     em1
```

Αυτό φαίνεται από το πεδίο `Flags (UG1)` και συγκεκριμένα από τη σημαία:

```
"1" (Protocol specific routing flag #1)
```

1.28

Στο PC1 εκτελούμε `do ping 192.168.2.2`. Το ping είναι επιτυχές, άρα το PC1 επικοινωνεί με το PC2.

Άσκηση 2: Λειτουργία του BGP

2.1

Στο τέλος της πρώτης γραμμής της εξόδου της εντολής `do show ip bgp neighbors` εμφανίζεται η φράση ”external link” ή ”internal link” αντίστοιχα.

2.2

Στην τρίτη γραμμή εμφανίζεται `BGP state = ...`, για παράδειγμα `BGP state = Established`.

2.3

Σε νέο παράθυρο στον R1 εκτελούμε `tcpdump -vni em1` και περιμένουμε τουλάχιστον ένα λεπτό.

2.4

Παρατηρούμε μηνύματα BGP KEEPALIVE.

2.5

Χρησιμοποιεί το TCP και τη θύρα 179. Η εντολή `do show ip bgp neighbors` δείχνει μόνο την θύρα που χρησιμοποιείται (αναγράφεται ως ”Local port”).

2.6

Τα βλέπουμε κάθε ένα λεπτό, όπως αναγράφεται και στην εντολή `do show ip bgp neighbors`:

`keepalive interval is 60 seconds`

2.7

Το TTL είναι 1.

2.8

Στον R2 εκτελούμε `do show ip bgp summary`. Το Router-ID του R2 είναι `10.1.1.5`, όπως αναγράφεται στην έξοδο της εντολής (`BGP router identifier 10.1.1.5`). Επιλέγεται αυτή η τιμή, ως η μεγαλύτερη IP διεύθυνση, εφόσον δεν έχει οριστεί διεύθυνση IP για την loopback.

2.9

Στην έξοδο της εντολής `do show ip bgp summary` βλέπουμε ότι οι 3 εγγραφές RIB καταναλώνουν 192 bytes, οπότε μία εγγραφή RIB καταναλώνει 64 bytes στη μνήμη.

2.10

Είναι `10.1.1.1` και το βρήκαμε δίνοντας την εντολή `do show ip bgp summary` στον R1.

2.11

Στον R1 εκτελούμε:

```
interface lo0
ip address 172.17.17.1/32

do show ip bgp summary
```

Τώρα το Router-ID του R1 είναι `172.17.17.1`.

2.12

Στον R1 εκτελούμε:

```
interface lo0
no ip addr 172.17.17.1/32

do show ip bgp summary
```

Το προηγούμενο Router-ID (`10.1.1.1`) επανέρχεται.

2.13

Με την εντολή:

```
bgp router-id <IPaddr>
```

όπου `<IPaddr>` η τιμή του επιθυμητού Router-ID.

2.14

Σε νέο παράθυρο εντολών στον R2 εκτελούμε `tcpdump -vni em1`.

2.15

Στον R3 εκτελούμε:

```
router bgp 65030
no network 192.168.2.0/24
```

2.16

Παρατηρούμε μήνυμα BGP UPDATE.

2.17

Όχι, τόσο η παραγωγή του μηνύματος όσο και η ενημέρωση του πίνακα δρομολόγησης στον R1 έγιναν αμέσως.

2.18

Στον R3 εκτελούμε `network 192.168.2.0/24`.

2.19

Ναι, υπήρξε καθυστέρηση.

2.20

Στον R1 εκτελούμε `do show ip bgp neighbors`. Παρατηρούμε ότι στάλθηκε ένα παραπάνω μήνυμα `BGP UPDATE` σε σχέση με πριν, με σκοπό την ενημέρωση για το νέο δίκτυο `192.168.2.0/24` που διαφημίζεται.

2.21

Έγινε με μήνυμα BGP UPDATE.

2.22

Χαρακτηριστικά:

```
ORIGIN: IGP
AS_PATH: 65020 65030
NEXT_HOP: 10.1.1.2
UPDATED ROUTES: 192.168.2.0/24
```

Άσκηση 3: Χαρακτηριστικά διαδρομών BGP

3.1

Εκτελούμε:

```
### R1 ###

interface em2
ip address 10.1.1.9/30

### R3 ###

interface em2
ip address 10.1.1.10/30
```

3.2

Στον PC1 εκτελούμε `do traceroute 192.168.2.2`. Βλέπουμε ότι ακολουθείται η διαδρομή PC1 → R1 → R2 → R3 → PC2.

3.3

Στον R1 εκτελούμε:

```
interface lo0
ip address 172.17.17.1/32
```

3.4

Στον R2 εκτελούμε:

```
interface lo0
ip address 172.17.17.2/32
```

3.5

Στον R3 εκτελούμε:

```
interface lo0
ip address 172.17.17.3/32
```

3.6

Εκτελούμε:

```
### R1 ###

router bgp 65010
network 172.17.17.1/32

### R2 ###

router bgp 65020
network 172.17.17.2/32

### R3 ###

router bgp 65030
network 172.17.17.3/32
```

3.7

Στον R1 εκτελούμε `do show ip bgp neighbors`. Βλέπουμε ότι ο R1 έχει γείτονα BGP τον R2 (`10.1.1.2`).

3.8

Στον R1 εκτελούμε `do show ip bgp`. Έχουμε:

```
NETWORK          NEXT_HOP
172.17.17.1/32   0.0.0.0
172.17.17.2/32   10.1.1.2
172.17.17.3/32   10.1.1.2
192.168.1.0      0.0.0.0
192.168.2.0      10.1.1.2
```

3.9

Στον R2 εκτελούμε `do show ip bgp neighbors`. Βλέπουμε ότι ο R2 έχει γείτονες τους R1 (`10.1.1.1`) και R3 (`10.1.1.6`).

3.10

Στον R2 εκτελούμε `do show ip bgp`. Έχουμε:

```
NETWORK          NEXT_HOP
172.17.17.1/32   10.1.1.1
172.17.17.2/32   0.0.0.0
172.17.17.3/32   10.1.1.6
192.168.1.0/24   10.1.1.1
192.168.2.0/24   10.1.1.6
```

3.11

Στον R3 εκτελούμε `do show ip bgp neighbors`. Βλέπουμε ότι ο R3 έχει γείτονα τον R2 (`10.1.1.5`).

3.12

Στον R3 εκτελούμε `do show ip bgp`. Έχουμε:

```
NETWORK          NEXT_HOP
172.17.17.1/32   10.1.1.5
172.17.17.2/32   10.1.1.5
172.17.17.3/32   0.0.0.0
192.168.1.0/24   10.1.1.5
192.168.2.0/24   0.0.0.0
```

3.13

Στον R1 σε νέο παράθυρο εκτελούμε `tcpdump -vni em2`.

3.14

Στον R1 εκτελούμε `neighbor 10.1.1.10 remote-as 65030`.

3.15

Στους R1 και R3 εκτελούμε `do show ip bgp neighbors`. Ο γείτονας R3 έχει προστεθεί μόνο στον R1, ενώ στον R3 δεν έχει αλλάξει κάτι.

3.16

Στον R1 εκτελούμε `do show ip bgp/route`. Η διαδρομή *δεν* είναι διαθέσιμη.

3.17

Στον R1 εκτελούμε `do show ip bgp neighbor`. Η σύνοδος βρίσκεται σε κατάσταση `Active`.

3.18

Στον R1 εκτελούμε `do show ip bgp summary`. Στην έξοδο της εντολής εμφανίζεται η εγγραφή:

```
Neighbor    AS      Up/Down   State
10.1.1.10   65030   never     Active
```

3.19

Βλέπουμε μηνύματα BGP OPEN.

3.20

Επαναλαμβάνεται κάθε δύο λεπτά, και ο R3 απαντά με πακέτο τερματισμού της σύνδεσης TCP (σημαία FIN ενεργοποιημένη).

3.21

Σταματάμε την καταγραφή. Μέχρι να εκτελέσουμε τα παραπάνω βήματα έγιναν προσπάθειες για να εγκατασταθεί σύνδεση TCP από τον R1, οι οποίες πετύχαιναν μόνο προσωρινά, καθώς ο R3 έστελνε αμέσως πακέτο για τερματισμό της σύνδεσης.

3.22

Στον R1 εκτελούμε `tcpdump -vni em2`.

3.23

Στον R3 εκτελούμε `neighbor 10.1.1.9 remote-as 65010`.

3.24

Στον R1 εκτελούμε `do show ip bgp neighbors`. Η σύνοδος βρίσκεται σε κατάσταση `Established`.

3.25

Στον R1 εκτελούμε `do show ip bgp`. Τώρα η διαδρομή είναι διαθέσιμη.

3.26

Στον R3 εκτελούμε `do show ip bgp`. Προστέθηκαν οι διαδρομές (επισημαίνονται με +++ στην αρχή της γραμμής):

```
    NETWORK          NEXT_HOP
+++ 172.17.17.1/32   10.1.1.9
    172.17.17.1/32   10.1.1.5
+++ 172.17.17.2/32   10.1.1.9
    172.17.17.2/32   10.1.1.5
    172.17.17.3/32   0.0.0.0
+++ 192.168.1.0/24   10.1.1.9
    192.168.1.0/24   10.1.1.5
    192.168.2.0/24   0.0.0.0
```

3.27

Στο PC1 εκτελούμε `do traceroute 192.168.2.2`. Βλέπουμε ότι τώρα το PC1 επικοινωνεί με το PC2 μέσω της διαδρομής PC1 → R1 → R3 → PC2.

3.28

Σταματάμε την καταγραφή. Αυτή τη φορά ο R3 στέλνει μήνυμα BGP OPEN, στο οποίο ο R1 απαντάει πάλι με BGP OPEN, ενώ πριν ο R3 απλά τερμάτιζε την σύνδεση TCP.

3.29

Παρατηρήσαμε μηνύματα BGP KEEPALIVE.

3.30

```
NETWORKS: 172.17.17.1/32, 192.168.1.0/24
AS_PATH: 65010

NETWORKS: 172.17.17.2/32
AS_PATH: 65010 65020

NETWORKS: 172.17.17.3/32, 192.168.2.0/24
AS_PATH: 65010 65020 65030
```

3.31

Στον R3 εκτελούμε `do show ip bgp`. Αγνοήθηκαν οι διαδρομές προς τα δίκτυα `172.17.17.3/32` και `192.168.2.0/24`, αφού αυτά τα δίκτυα ανήκουν στο τοπικό αυτόνομο σύστημα AS 65030 (το οποίο αντιλαμβάνεται ο R3 επειδή βλέπει το τοπικό αυτόνομο σύστημα AS 65030 στο `AS_PATH`).

3.32

Στον R1 εκτελούμε `do show ip bgp 172.17.17.2/32`. Υπάρχουν 2 διαδρομές προς τον προορισμό `172.17.17.2/32`, με καλύτερη αυτή που έχει επόμενο βήμα τη διεύθυνση `10.1.1.2`, δηλαδή τον R2.

3.33

Έχουμε:

Path	NEXT_HOP	ORIGIN	AS_PATH	Local Preference
#1	10.1.1.10	IGP	65030 65020	100
#2	10.1.1.2	IGP	65020	100

3.34

Ξεκινάμε από τα ισχυρότερα κριτήρια και πηγαίνουμε προς τα πιο αδύναμα, μέχρι να βρούμε κάποιο κριτήριο στο οποίο η μία διαδρομή να υπερισχύει της άλλης, οπότε θα είναι και η βέλτιστη. Τα κριτήρια είναι:

- Η διαδρομή με το υψηλότερο βάρος. Και οι δύο διαδρομές έχουν βάρος 0, καθώς δεν πηγάζουν από τον R1.
- Η διαδρομή με την υψηλότερη τιμή `Local Preference`. Και οι δύο διαδρομές έχουν τιμή 100.
- Η διαδρομή που ορίζεται τοπικά στο AS σε σχέση με μία που έγινε γνωστή μέσω γείτονα eBGP. Και οι δύο διαδρομές έγιναν γνωστές μέσω γείτονα BGP.
- Η διαδρομή με μικρότερο μήκος `AS_PATH`. Η διαδρομή `#2` έχει μικρότερο μήκος `AS_PATH` (ίσο με 1) από την διαδρομή `#1` (ίσο με 2), οπότε τελειώσαμε.

Επομένως το κριτήριο για την επιλογή είναι το μήκος του `AS_PATH`.

3.35

Στον R1 εκτελούμε `tcpdump -vni em2 "tcp && src 10.1.1.10 && port 179"`.

3.36

Στον R3 εκτελούμε `tcpdump -vni em0 "tcp && src 10.1.1.5 && port 179"`.

3.37

Στον R2 εκτελούμε `no network 172.17.17.2/32`.

3.38

Παράχθηκε μήνυμα BGP UPDATE με την πληροφορία:

```
Withdrawn routes: 5 bytes
```

3.39

Στον R2 εκτελούμε `network 172.17.17.2/32`.

3.40

Είναι:

```
ORIGIN: IGP
AS_PATH: 65020
NEXT_HOP: 10.1.1.5
```

3.41

Στον R2 εκτελούμε `ip route 5.5.5.0/24 lo0`.

3.42

Στον R2 εκτελούμε:

```
router bgp 65020
redistribute static
```

3.43

Είναι `ORIGIN: Incomplete`.

3.44

Στους R1,2,3 εκτελούμε `do show ip bgp`. Η πληροφορία για τιμή `ORIGIN: Incomplete` δηλώνεται με το σύμβολο `"?"` στο τέλος της γραμμής.

Άσκηση 4: Εφαρμογή πολιτικών στο BGP

4.1

Στον R1 εκτελούμε:

```
do show ip bgp 192.168.2.0/24
```

Έχουμε:

```
NEXT_HOP      METRIC     LOCPRF     AS_PATH         ORIGIN
10.1.1.10     0          100        65030           IGP
10.1.1.2                 100        65020 65030     IGP
```

4.2

Στον R3 εκτελούμε:

```
do show ip bgp 192.168.1.0/24
```

Έχουμε:

```
NEXT_HOP      METRIC      LOCPRF      AS_PATH          ORIGIN
10.1.1.9      0           100         65010            IGP
10.1.1.5                  100         65020 65010      IGP
```

4.3

Στον R2 εκτελούμε:

```
do show ip bgp 192.168.1.0/24
do show ip bgp 192.168.2.0/24
```

Έχουμε:

```
### 192.168.1.0/24 ###

NEXT_HOP      METRIC      LOCPRF      AS_PATH          ORIGIN
10.1.1.6                  100         65030 65010      IGP
10.1.1.1      0           100         65010            IGP

### 192.168.2.0/24 ###

NEXT_HOP      METRIC      LOCPRF      AS_PATH          ORIGIN
10.1.1.1                  100         65010 65030      IGP
10.1.1.6      0           100         65030            IGP
```

4.4

Με την εντολή `do show ip bgp neighbors 10.1.1.10 advertised-routes` στον R1. Έχουμε:

```
   Network            Next Hop      Metric      Weight      Path
*> 5.5.5.0/24         10.1.1.9                  0           65020 ?
*> 172.17.17.1/32     10.1.1.9      0           32768       i
*> 172.17.17.2/32     10.1.1.9                  0           65020 i
*> 192.168.1.0/24     10.1.1.9      0           32768       i
```

4.5

Με την εντολή `do show ip bgp neighbors 10.1.1.10 routes` στον R1. Έχουμε:

```
   Network            Next Hop       Metric      Weight      Path
*  5.5.5.0/24         10.1.1.10                  0           65030 65020 ?
*  172.17.17.2/32     10.1.1.10                  0           65030 65020 i
*> 172.17.17.3/32     10.1.1.10      0           0           65030 i
*> 192.168.2.0/24     10.1.1.10      0           0           65030 i
```

4.6

Στον R1 εκτελούμε:

```
ip prefix-list geitones_in deny 192.168.2.0/24
```

4.7

Στον R1 εκτελούμε:

```
ip prefix-list geitones_in permit any
```

4.8

Στον R1 εκτελούμε:

```
router bgp 65010
neighbor 10.1.1.10 prefix-list geitones_in in
```

4.9

Στον R1 εκτελούμε `do show ip bgp 192.168.2.0/24`. Δεν παρατηρούμε κάποια αλλαγή.

4.10

Στον R1 εκτελούμε `do clear ip bgp 10.1.1.10`. Εάν δεν χρησιμοποιούσαμε το `do`, θα έπρεπε να είχαμε εκτελέσει `exit` για να εισέλθουμε σε global configuration mode, και ύστερα ξανά `exit` για να εισέλθουμε σε privileged exec mode.

4.11

Στον R1 εκτελούμε `do show ip bgp neighbors 10.1.1.10 routes`. Έχει διαγραφεί η διαδρομή προς το δίκτυο `192.168.2.0/24`.

4.12

Στον R1 εκτελούμε `do show ip bgp neighbors 10.1.1.10 advertised-routes`. Έχει προστεθεί η εγγραφή:

```
   Network          Next Hop     Metric     Weight     Path
*> 192.168.2.0      10.1.1.9                0          65020 65030 i
```

4.13

Στον R1 εκτελούμε `do show ip bgp`. Πλέον υπάρχει μόνο μία διαδρομή για το δίκτυο `192.168.2.0/24` μέσω του R2.

4.14

Στον R2 εκτελούμε `do show ip bgp`. Πλέον υπάρχει μόνο μία διαδρομή για το δίκτυο `192.168.2.0/24` μέσω του R3.

4.15

Στον PC1 εκτελούμε `ping -R 192.168.2.2`. Ακολουθείται η διαδρομή:

```
192.168.1.2 (PC1)     # ICMP request starts
10.1.1.1 (R1)
10.1.1.5 (R2)
192.168.2.1 (R3)
192.168.2.2 (PC2)     # ICMP request is received, ICMP reply starts
10.1.1.10 (R3)
192.168.1.1 (R1)
192.168.1.2 (PC1)     # ICMP reply is received
```

4.16

Δεν την επηρεάζει, αφού ο R1 συνεχίζει να διαφημίζει το δίκτυο `192.168.1.0/24` στον R3.

4.17

Στον R1 εκτελούμε `ip prefix-list geitones_out deny 192.168.1.0/24`.

4.18

Στον R1 εκτελούμε `ip prefix-list geitones_out permit any`.

4.19

Στον R1 εκτελούμε:

```
router bgp 65010
neighbor 10.1.1.10 prefix-list geitones_out out
```

4.20

Στον R1 εκτελούμε `do clear ip bgp 10.1.1.10`.

4.21

Στον R1 εκτελούμε `do show ip bgp neighbor 10.1.1.10 advertised-routes`. Έχει διαγραφεί η διαδρομή προς το δίκτυο `192.168.1.0/24`.

4.22

Στον R1 εκτελούμε `do show ip bgp neighbor 10.1.1.10 routes`. Δεν παρατηρούμε κάποια αλλαγή.

4.23

Στον R3 εκτελούμε `do show ip bgp`. Πλέον προς το `192.168.1.0/24` υπάρχει μόνο μία διαδρομή που έχει Next Hop τη διεύθυνση `10.1.1.5`, δηλαδή τον R2.

4.24

Στον R2 εκτελούμε `do show ip bgp`. Πλέον προς το `192.168.2.0/24` υπάρχει μόνο μία διαδρομή που έχει Next Hop τη διεύθυνση `10.1.1.1`, δηλαδή τον R1.

4.25

Στον PC1 εκτελούμε `ping -R 192.168.2.2`. Ακολουθείται η διαδρομή:

```
192.168.1.2 (PC1) # ICMP request starts
10.1.1.1 (R1)
10.1.1.5 (R2)
192.168.2.1 (R3)
192.168.2.2 (PC2) # ICMP request is received, ICMP reply starts
10.1.1.6 (R3)
10.1.1.2 (R2)
192.168.1.1 (R1)
192.168.1.2 (PC1) # ICMP reply is received
```

4.26

Στον R1 εκτελούμε:

```
no neighbor 10.1.1.10 prefix-list geitones_in in
no neighbor 10.1.1.10 prefix-list geitones_out out
do clear ip bgp 10.1.1.10
```

Άσκηση 5: iBGP

5.1

Στον R4 εκτελούμε:

```
cli
configure terminal

hostname R4

interface em0
ip address 192.168.0.2/24

interface em1
ip address 10.1.1.13/30
```

5.2

Στον R4 εκτελούμε:

```
interface lo0
ip address 172.17.17.4/32
```

5.3

Στον R1 εκτελούμε:

```
interface em3
ip address 192.168.0.1/24
```

5.4

Στον R3 εκτελούμε:

```
interface em3
ip address 10.1.1.14/30
```

5.5

Στον R4 εκτελούμε:

```
router bgp 65010
```

5.6

Στον R4 εκτελούμε:

```
neighbor 192.168.0.1/24 remote-as 65010
network 172.17.17.4/32
```

5.7

Στον R1 εκτελούμε:

```
neighbor 192.168.0.2/24 remote-as 65010
```

5.8

Στον R1 εκτελούμε `do show ip bgp neighbors 192.168.0.2`. Η σύνοδος είναι internal, αφού στην πρώτη γραμμή αναγράφεται `"internal link"`.

5.9

Στον R4 εκτελούμε `do show ip bgp neighbors 192.168.0.1 routes` και βλέπουμε ότι οι διαδρομές που έμαθε από τον R1 και συμπεριέλαβε στην RIB είναι (αναγράφεται ακριβώς δίπλα και το `NEXT_HOP`):

```
Network           Next Hop
5.5.5.0/24        10.1.1.2
172.17.17.1/32    192.168.0.1
172.17.17.2/32    10.1.1.2
172.17.17.3/32    10.1.1.10
192.168.1.0       192.168.0.1
192.168.2.0       10.1.1.10
```

5.10

Αντίστοιχα, στον R1 εκτελούμε `do show ip bgp neighbors 192.168.0.2 routes` και έχουμε:

```
Network           Next Hop
172.17.17.4/32    192.168.0.2
```

5.11

Τις διακρίνουμε επειδή έχουν το σύμβολο `"i"` στην αρχή της γραμμής (διαφορετικό από το `"i"` στο τέλος της γραμμής -βλ. ερώτημα 1.26-).

5.12

Ναι, έχουν τεθεί ρητά οι τιμές Metric = 0 και Local Preference = 100.

5.13

Στον R4 εκτελούμε `do show ip route`. Από τα δίκτυα της ερώτησης 5.9 έχουν εισαχθεί στον πίνακα δρομολόγησης τα:

```
172.17.17.1/32
192.168.1.0/24
```

5.14

Δεν έχουν εισαχθεί τα:

```
5.5.5.0/24
172.17.17.2/32
172.17.17.3/32
192.168.2.0/24
```

Αυτό συμβαίνει επειδή τα `NEXT_HOP` που αφορούν τα παραπάνω δίκτυα δεν είναι προσβάσιμα, δηλαδή δεν υπάρχει εγγραφή γι' αυτά στον πίνακα δρομολόγησης του R4.

5.15

Στον R4 εκτελούμε `ip route 10.1.1.8/30 192.168.0.1`.

5.16

Στον R4 εκτελούμε `do show ip route`. Τώρα το `192.168.2.0/24` έχει τοποθετηθεί στον πίνακα δρομολόγησης του R4. Για το επόμενο βήμα αναγράφεται:

```
via 10.1.1.10 (recursive via 192.168.0.1)
```

5.17

Από τα:

```
5.5.5.0/24
172.17.17.2/32
172.17.17.3/32
192.168.2.0/24
```

δεν έχουν εισαχθεί τα:

```
5.5.5.0/24
172.17.17.2/32,
```

επειδή το `NEXT_HOP` τους (`10.1.1.2`) εξακολουθεί να μην είναι προσβάσιμο από τον R4.

5.18

Στον R1 εκτελούμε `neighbor 192.168.0.2 next-hop-self`.

5.19

Στον R4 εκτελούμε `do show ip route`. Παρατηρούμε ότι πλέον έχουν εισαχθεί και τα δίκτυα

```
5.5.5.0/24
172.17.17.2/32
```

στον πίνακα δρομολόγησης. Επίσης δεν αναγράφεται πλέον το μήνυμα (`recursive via 192.168.0.1`) ενώ ως `NEXT_HOP` για όλες τις διαδρομές iBGP είναι το `192.168.0.1`, δηλαδή ο R1.

5.20

Στον R4 εκτελούμε `do show ip route`. Η διαχειριστική απόσταση είναι 200, γιατί αυτή τη φορά πρόκειται για Internal BGP, ενώ στην ερώτηση 1.21 επρόκειτο για External BGP (διαχειριστική απόσταση 20).

5.21

Στον R4 εκτελούμε `do ping 10.1.1.9`. Το ping είναι επιτυχές.

5.22

Στον R4 εκτελούμε `do ping 10.1.1.10`. Δεν λαμβάνουμε καμία απάντηση. Αυτό συμβαίνει διότι, ενώ το ICMP request φτάνει στον R3, δεν μπορεί να σταλεί ICMP reply πίσω στον R4, επειδή δεν υπάρχει εγγραφή στον πίνακα δρομολόγησης του R3 σχετική με τον R4 (είτε εγγραφή για το δίκτυο `192.168.0.0/24` είτε προκαθορισμένη διαδρομή)

5.23

Στον R1 εκτελούμε `network 192.168.0.0/24`.

5.24

Στον R4 εκτελούμε πάλι `do ping 10.1.1.10`. Πλέον το ping είναι επιτυχές.

5.25

Στον R1 εκτελούμε `aggregate-address 192.168.0.0/23`.

5.26

Στον R3 εκτελούμε `do show ip bgp`. Πλέον βλέπουμε 6 εγγραφές σχετικές με το `192.168.0.0/23` και τα υποδίκτυά του:

```
Network          Next Hop
192.168.0.0/23   10.1.1.9
192.168.0.0/23   10.1.1.5
192.168.0.0      10.1.1.5
192.168.0.0      10.1.1.9
192.168.1.0      10.1.1.9
192.168.1.0      10.1.1.5
```

5.27

Στον R1 εκτελούμε `aggregate-address 192.168.0.0/23 summary-only`.

5.28

Στον R3 εκτελούμε `do show ip bgp`. Πλέον βλέπουμε 2 εγγραφές σχετικές με το `192.168.0.0/23`:

```
Network          Next Hop
192.168.0.0/23   10.1.1.9
192.168.0.0/23   10.1.1.5
```

5.29

Στον R1 εκτελούμε `no aggregate-address 192.168.0.0/23 summary-only`.

5.30

Στον R4, σε νέο παράθυρο, εκτελούμε `tcpdump -vni em0`.

5.31

Το TTL των πακέτων που μεταφέρουν τα μηνύματα BGP είναι ίσο με 64 και όχι 1, όπως είχαμε βρει στην ερώτηση 2.7. Αυτό συμβαίνει γιατί αυτή τη φορά έχουμε iBGP (default TTL 64), και όχι eBGP (default TTL 1).

Άσκηση 6: Περισσότερα περί πολιτικών στο BGP

6.1

Εκτελούμε:

```
### R4 ###
neighbor 10.1.1.14 remote-as 65030

### R3 ###
neighbor 10.1.1.13 remote-as 65010
```

6.2

Στον R4 εκτελούμε `neighbor 192.168.0.1 next-hop-self`.

6.3

Στον R1 εκτελούμε `do show ip bgp`. Υπάρχουν 3 διαδρομές προς το `192.168.2.0/24`:

```
   Network          Next Hop
#1 192.168.2.0/24    10.1.1.14
#2 192.168.2.0/24    10.1.1.10
#3 192.168.2.0/24    10.1.1.2
```

Από αυτές έχει επιλεχθεί η διαδρομή #2, με `NEXT_HOP` τη διεύθυνση `10.1.1.10`, δηλαδή τον R3.

6.4

Ελέγχουμε ένα-ένα τα κριτήρια ξεκινώντας από το σημαντικότερο και σταματάμε όταν μόνο μία διαδρομή υπερτερεί σε κάποιο:

Συν.: σημαίνει ότι η διαδρομή συνεχίζει και στο επόμενο κριτήριο.
Απ.: σημαίνει ότι η διαδρομή αποκλείεται με βάση το τρέχον κριτήριο.
—: σημαίνει ότι η διαδρομή έχει αποκλειστεί πιο πριν και δεν λαμβάνεται πλέον υπόψιν.
Επ.: σημαίνει ότι η διαδρομή επιλέγεται ως καλύτερη και η αναζήτηση σταματά.

Κριτήριο	**#1**	**#2**	**#3**	**Αιτιολόγηση**
`WEIGHT`	Συν.	Συν.	Συν.	Ίδιο `WEIGHT` για όλες
`LOCAL_PREF`	Συν.	Συν.	Συν.	Ίδιο `LOCAL_PREF` για όλες
`AS_PATH`	Συν.	Συν.	Απ.	Οι #1 και #2 έχουν μήκος 1, ενώ η #3 έχει μήκος 2
ORIGIN	Συν.	Συν.	—	Όλες είναι `IGP (i)`
MED	Συν.	Συν.	—	Δεν έχει οριστεί MED σε καμία διαδρομή
eBGP vs iBGP	Απ.	Επ.	—	#1 γνωστή μέσω γείτονα iBGP, ενώ #2 μέσω γείτονα eBGP.

Τελικά η επιλογή βασίστηκε στο ότι η διαδρομή #2 (`NEXT_HOP 10.1.1.10`) έγινε γνωστή μέσω γείτονα eBGP, ενώ η διαδρομή #1 μέσω γείτονα iBGP, γι' αυτό και επιλέχθηκε η διαδρομή #2.

6.5

Στον R4 εκτελούμε `do show ip bgp`. Υπάρχουν 2 διαδρομές προς το `192.168.2.0/24`:

```
   Network            Next Hop
#1 192.168.2.0/24     10.1.1.14
#2 192.168.2.0/24     192.168.0.1
```

Από αυτές έχει επιλεχθεί η διαδρομή `#1`, με `NEXT_HOP` τη διεύθυνση `10.1.1.14`, δηλαδή τον R3.

6.6

Αντίστοιχα με το ερώτημα 6.4, ελέγχουμε τα κριτήρια ξεκινώντας από το σημαντικότερο. Και πάλι το καθοριστικό κριτήριο είναι ότι η διαδρομή `#1` (`NEXT_HOP 10.1.1.14`) έγινε γνωστή μέσω γείτονα eBGP, ενώ η διαδρομή `#2` μέσω γείτονα iBGP, οπότε επιλέχθηκε η διαδρομή `#1`.

6.7

Στον R4 εκτελούμε `do show ip bgp`. Υπάρχουν 2 διαδρομές προς το `172.17.17.2/32`:

```
   Network             Next Hop
#1 172.17.17.2/32      10.1.1.14
#2 172.17.17.2/32      192.168.0.1
```

Από αυτές έχει επιλεχθεί η διαδρομή `#2`, με `NEXT_HOP` τη διεύθυνση `192.168.0.1`, δηλαδή τον R1.

6.8

Αντίστοιχα με τα ερωτήματα 6.4 και 6.6, ελέγχουμε τα κριτήρια ξεκινώντας από το σημαντικότερο. Αυτή τη φορά καθοριστικό κριτήριο είναι ότι η διαδρομή `#2` (`NEXT_HOP 192.168.0.1`) έχει μικρότερο μήκος `AS_PATH` (ίσο με 1) από τη διαδρομή `#1` (ίσο με 2), οπότε επιλέγεται η διαδρομή `#2`.

6.9

Στον R3 εκτελούμε `do show ip bgp`. Υπάρχουν 3 διαδρομές προς το `192.168.1.0/24`:

```
   Network             Next Hop
#1 192.168.1.0/24      10.1.1.13
#2 192.168.1.0/24      10.1.1.5
#3 192.168.1.0/24      10.1.1.9
```

Από αυτές έχει επιλεχθεί η διαδρομή `#3`, με `NEXT_HOP` τη διεύθυνση `10.1.1.9`, δηλαδή τον R1.

6.10

Σύμφωνα με την υπόδειξη της ερώτησης, εκτελούμε `do show ip bgp 192.168.1.0/24`. Αρχικά η διαδρομή `#2` αποκλείεται επειδή έχει μεγαλύτερο μήκος `AS_PATH` (ίσο με 2) από τις `#1` και `#3` (ίσο με 1). Ύστερα, μεταξύ των διαδρομών `#1` και `#3` επιλέγεται η `#3` επειδή είναι αρχαιότερη, όπως δείχνει και η τιμή του πεδίου `Last Update`.

6.11

Στον R1 εκτελούμε `do clear ip bgp 10.1.1.10` και ύστερα από λίγο εκτελούμε στον R3 `do show ip bgp 192.168.1.0/24`. Παρατηρούμε ότι πλέον έχει επιλεχθεί η διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.13`, δηλαδή τον R4.

6.12

Στον R4 εκτελούμε `do clear ip bgp 10.1.1.14` και ύστερα από λίγο εκτελούμε στον R3 `do show ip bgp 192.168.1.0/24`. Παρατηρούμε ότι έχει επιλεχθεί η διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.9`, δηλαδή τον R1.

6.13

Στον R1 εκτελούμε:

```
exit      # To enter global configuration mode

route-map set-locpref permit 10
```

6.14

Στον R1 εκτελούμε:

```
set local-preference 90

exit
```

6.15

Στον R1 εκτελούμε:

```
router bgp 65010

neighbor 10.1.1.10 route-map set-locpref in
```

6.16

Στον R1 εκτελούμε `do clear ip bgp *` και περιμένουμε περίπου ένα λεπτό. Ύστερα εκτελούμε `do show ip bgp`. Η τιμή του local-preference έχει αλλάξει στις διαδρομές:

```
Network             Next Hop       LocPrf
5.5.5.0/24          10.1.1.10      90
172.17.17.2/32      10.1.1.10      90
172.17.17.3/32      10.1.1.10      90
192.168.2.0/24      10.1.1.10      90
```

6.17

Στον R1 εκτελούμε `do show ip route`. Έχει επιλεχθεί η διαδρομή με επόμενο βήμα τη διεύθυνση `192.168.0.2`, δηλαδή τον R4, επειδή έχει μεγαλύτερο local-preference (ίσο με 100) από τη διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.10` (ίσο με 90), και επίσης έχει μικρότερο μήκος `AS_PATH` (ίσο με 1) από τη διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.2` (ίσο με 2).

6.18

Στον R4 εκτελούμε `do show ip bgp`. Παρατηρούμε ότι πλέον υπάρχει μόνο μία διαδρομή προς το δίκτυο `192.168.2.0/24` με επόμενο βήμα τη διεύθυνση `10.1.1.14`, δηλαδή τον R3. Επιπλέον, υπάρχει μόνο μία διαδρομή προς το δίκτυο `172.17.17.3/32` με επόμενο βήμα τη διεύθυνση `10.1.1.14`, δηλαδή τον R3.

6.19

Στον R1 εκτελούμε:

```
do show ip bgp neighbors 192.168.0.2 advertised-routes
```

Εμφανίζονται διαδρομές για τα δίκτυα:

```
5.5.5.0/24
172.17.17.1/32
172.17.17.2/32
192.168.0.0/24
192.168.1.0/24
```

Παρατηρούμε ότι δεν εμφανίζονται διαδρομές για τα δίκτυα του AS 65030.

6.20

Οι αλλαγές στην RIB του R4 που παρατηρήσαμε προηγουμένως οφείλονται στο ότι ο R1 δεν διαφημίζει πλέον διαδρομές προς τα δίκτυα του AS 65030, όπως είδαμε παραπάνω. Αυτό συμβαίνει επειδή η αλλαγή του local-preference στις διαδρομές που διαφημίζει ο R3 στον R1 ωθεί τον R1 να προτιμά διαδρομές με επόμενο βήμα τον R4, και όχι τον R3.

6.21

Στο PC1 εκτελούμε `ping -R 192.168.2.2`. Ακολουθείται η διαδρομή:

```
192.168.1.2 (PC1)      # ICMP request starts
192.168.0.1 (R1)
10.1.1.13   (R4)
192.168.2.1 (R3)
192.168.2.2 (PC2)      # ICMP request is received, ICMP reply starts
10.1.1.10   (R3)
192.168.1.1 (R1)
192.168.1.2 (PC1)      # ICMP reply is received
```

6.22

Στον R1 εκτελούμε:

```
exit      # To enter global configuration mode

route-map set-MED permit 15
```

6.23

Στον R1 εκτελούμε:

```
set metric 1
exit
```

6.24

Στον R1 εκτελούμε:

```
router bgp 65010
neighbor 10.1.1.10 route-map set-MED out
```

6.25

Στον R1 εκτελούμε `do clear ip bgp 10.1.1.10` και ύστερα από λίγο εκτελούμε στον R3 `do show ip bgp`. Η τιμή του Metric έχει αλλάξει στις διαδρομές:

```
Network             Next Hop      Metric
5.5.5.0/24          10.1.1.9      1
172.17.17.1/32      10.1.1.9      1
172.17.17.2/32      10.1.1.9      1
172.17.17.4/32      10.1.1.9      1
192.168.0.0/24      10.1.1.9      1
192.168.1.0/24      10.1.1.9      1
```

6.26

Στον R3 εκτελούμε `do show ip bgp`. Έχει επιλεχθεί η διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.13`, δηλαδή τον R4, επειδή έχει μικρότερο μήκος `AS_PATH` από την διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.5` (1 έναντι 2), και επιπλέον επειδή έχει μικρότερο MED από τη διαδρομή με επόμενο βήμα τη διεύθυνση `10.1.1.9` (0 έναντι 1). Δίνεται έμφαση στο ότι οι διαδρομές με επόμενο βήμα `10.1.1.9` και `10.1.1.13` έχουν ίδιο πρώτο βήμα AS, αλλιώς δεν θα είχε νόημα η σύγκριση του MED.

6.27

Στον PC1 εκτελούμε `ping -R 192.168.2.2`. Αυτή τη φορά ακολουθείται η διαδρομή:

```
192.168.1.2 (PC1)      # ICMP request starts
192.168.0.1 (R1)
10.1.1.13   (R4)
192.168.2.1 (R3)
192.168.2.2 (PC2)      # ICMP request is received, ICMP reply starts
10.1.1.14   (R3)
192.168.0.2 (R4)
192.168.1.1 (R1)
192.168.1.2 (PC1)      # ICMP reply is received
```

6.28

Στον R1 εκτελούμε:

```
exit #      To enter global configuration mode

route-map set-prepend permit 5
```

6.29

Στον R1 εκτελούμε:

```
set as-path prepend 65010 65010

exit
```

6.30

Στον R1 εκτελούμε:

```
router bgp 65010
neighbor 10.1.1.2 route-map set-prepend out
```

6.31

Στον R1 εκτελούμε `do clear ip bgp 10.1.1.2` και ύστερα από λίγο στον R2 εκτελούμε `do show ip bgp`. Παρατηρούμε ότι στο `AS_PATH` των εγγραφών που έχουν ως επόμενο βήμα τη διεύθυνση `10.1.1.1` εμφανίζεται 3 φορές το ASN 65010, εξαιτίας του route-map που εφαρμόσαμε, σύμφωνα με το οποίο ο R1 προσθέτει 2 παραπάνω φορές το 65010 στο `AS_PATH` των διαδρομών που διαφημίζει στον R2.

6.32

Στον R2 εκτελούμε `do show ip route bgp`. Το επόμενο βήμα σε όλες τις διαδρομές είναι η διεύθυνση `10.1.1.6`, δηλαδή ο R3.

6.33

Στον R3 εκτελούμε `do show ip bgp`. Παρατηρούμε ότι έχουν διαγραφεί οι εγγραφές:

```
Network             Next Hop
172.17.17.1/32      10.1.1.5
172.17.17.4/32      10.1.1.5
192.168.0.0/24      10.1.1.5
192.168.1.0/24      10.1.1.5
```

όσες δηλαδή είχαν ως επόμενο βήμα τον R2.

6.34

Στον R4 εκτελούμε `do show ip bgp` και επιβεβαιώνουμε ότι δεν έχει αλλάξει τίποτα. Αυτό συμβαίνει διότι οι διαδρομές που διαγράφηκαν στον R3 δεν διαφημίζονταν έτσι κι αλλιώς από τον R3, αφού δεν είχαν επιλεχθεί στον πίνακα δρομολόγησης.

Άσκηση 7: Περισσότερα για το iBGP και την προκαθορισμένη διαδρομή

7.1

Στο PC1 εκτελούμε:

```
no ip route 0.0.0.0/0 192.168.1.1

router bgp 65010

neighbor 192.168.1.1 remote-as 65010
```

7.2

Στον R1 εκτελούμε `neighbor 192.168.1.2 remote-as 65010`.

7.3

Στον PC1 εκτελούμε:

```
do show ip bgp
do show ip route
```

Παρατηρούμε ότι στον πίνακα δρομολόγησης έχουν εισαχθεί μόνο τα δίκτυα:

```
172.17.17.1/32
192.168.0.0/24
192.168.1.0/24
```

δηλαδή αυτά που ανήκουν στο AS 65010. Αυτό συμβαίνει διότι οι υπόλοιποι προορισμοί έχουν ως επόμενο βήμα (`NEXT_HOP`) διευθύνσεις οι οποίες δεν είναι προσβάσιμες από τον PC1, δηλαδή δεν υπάρχουν στον πίνακα δρομολόγησής του.

7.4

Στον R1 εκτελούμε `neighbor 192.168.1.2 next-hop-self` και ύστερα από λίγο στο PC1 εκτελούμε `do show ip route`. Τώρα ο PC1 γνωρίζει διαδρομές προς τα δίκτυα:

```
5.5.5.0/24          # new
172.17.17.1/32
172.17.17.2/32      # new
192.168.0.0/24
192.168.1.0/24
```

δηλαδή προς όλους τους προορισμούς που έμαθε από τον R1.

7.5

Διότι ο R1 δεν διαφημίζει διαδρομές προς τα δίκτυα:

`172.17.17.3/32, 172.17.17.4/32, 192.168.2.0/24`

στον PC1, διότι αυτές έγιναν γνωστές από άλλο εσωτερικό συνομιλητή (τον R4), και επομένως δεν προωθούνται σε άλλους εσωτερικούς συνομιλητές.

7.6

Εκτελούμε:

```
### PC1 ###

neighbor 192.168.0.2 remote-as 65010

### R4 ###

neighbor 192.168.1.2 remote-as 65010
```

7.7

Πρέπει να εκτελέσουμε στον R4 `neighbor 192.168.1.2 next-hop-self`.

7.8

Στο PC1 εκτελούμε:

```
do ping 192.168.1.2               # Network: LAN1 (192.168.1.0/24)
do ping 192.168.2.2               # Network: LAN2 (192.168.2.0/24)
do ping 192.168.0.2               # Network: LAN3 (192.168.0.0/24)
do ping 10.1.1.2                  # Network: WAN1 (10.1.1.0/30)
do ping 10.1.1.6                  # Network: WAN2 (10.1.1.4/30)
do ping 10.1.1.10                 # Network: WAN3 (10.1.1.8/30)
do ping 10.1.1.14                 # Network: WAN5 (10.1.1.12/30)
do ping 172.17.17.1               # Network: 172.17.17.1/32
```

```
do ping 172.17.17.2                # Network: 172.17.17.2/32
do ping 172.17.17.3                # Network: 172.17.17.3/32
do ping 172.17.17.4                # Network: 172.17.17.4/32
```

Τα ping προς τα `LAN1,2,3` και προς τα `172.17.17.{1,2,3,4}/32` επιτυγχάνουν, ενώ τα ping προς τα `WAN1,2,3,5` αποτυγχάνουν.

7.9

Στο PC1 εκτελούμε `ping -R 192.168.2.2`. Ακολουθείται η διαδρομή:

```
192.168.1.2 (PC1)      # ICMP request starts
192.168.0.1 (R1)
10.1.1.13   (R4)
192.168.2.1 (R3)
192.168.2.2 (PC2)      # ICMP request is received, ICMP reply starts
10.1.1.14   (R3)
192.168.0.2 (R4)
192.168.1.1 (R1)
192.168.1.2 (PC1)      # ICMP reply is received
```

7.10

Εκτελούμε:

```
### PC1 ###

traceroute 5.5.5.1          # LAN1 --> 5.5.5.0/24

### R2 ###

ping -R -S 172.17.17.2      # 5.5.5.0/24 --> LAN1
```

- Διαδρομή από το LAN1 προς το `5.5.5.0/24`

  ```
  0  192.168.1.2 (PC1)
  1  192.168.1.1 (R1)
  2  10.1.1.5    (R2)
  ...
  64 10.1.1.5    (R2)
  ```

  το ICMP request εγκλωβίζεται στην loopback του R2.

- Διαδρομή από το `5.5.5.0/24` προς το LAN1:

  ```
  172.17.17.2 (R2)
  10.1.1.14   (R3)
  192.168.0.2 (R4)
  192.168.1.1 (R1)
  192.168.1.2 (PC1)
  ```

7.11

Στα PC1,2 εκτελούμε `do ping 10.1.1.9`. Επιβεβαιώνουμε ότι το ping επιτυγχάνει στο PC2, ενώ αποτυγχάνει στο PC1. Αυτό συμβαίνει διότι στον PC2 έχει οριστεί προκαθορισμένη διαδρομή μέσω του R3, οπότε το ICMP request προωθείται στον R3 και από εκεί στην άμεσα συνδεδεμένη διεύθυνση `10.1.1.9` του R1. Αντίθετα, στον PC1 δεν υπάρχει προκαθορισμένη διαδρομή ή σχετική με τη διεύθυνση `10.1.1.9` εγγραφή στον πίνακα δρομολόγησης.

7.12

Στον R2 εκτελούμε `network 0.0.0.0/0`.

7.13

Στον R2 εκτελούμε:

```
do show ip bgp
do show ip route
```

Η προκαθορισμένη διαδρομή έχει προστεθεί στην RIB του R2, παρά το γεγονός ότι δεν έχει προστεθεί στον πίνακα δρομολόγησης, κάτι που δεν είναι αναμενόμενο με βάση τη θεωρία και την λειτουργία του BGP που έχουμε παρατηρήσει έως τώρα στη διάρκεια της άσκησης.

7.14

Στους R1, R3, R4 και PC1 εκτελούμε `do show ip route`. Η προκαθορισμένη διαδρομή έχει προστεθεί στον πίνακα δρομολόγησης των άλλων δρομολογητών και του PC1.

7.15

Στους R1, R3, R4 και PC1 εκτελούμε `do show ip bgp`. Ο τύπος πηγής ORIGIN είναι IGP.

7.16

Στο PC1 εκτελούμε:

```
do ping 10.1.1.2
do ping 10.1.1.6
do ping 10.1.1.10
```

Ναι, μπορούμε να κάνουμε ping στις διευθύνσεις IP και των τριών αυτών WAN.

7.17

Αν στο PC1 εκτελέσουμε `do ping 10.1.1.14` θα λάβουμε μήνυμα λάθους ”Destination Host Unreachable” από τον R2. Ακολουθεί η εξήγηση:

- Ο PC1 δεν διαθέτει σχετική εγγραφή για τη διεύθυνση `10.1.1.14`, οπότε προωθεί το ICMP request στον R1 μέσω της προκαθορισμένης διαδρομής.

- Ο R1, για τον ίδιο λόγο, προωθεί το ICMP request στον R2 μέσω της προκαθορισμένης διαδρομής.
- Ο R2 δεν διαθέτει ούτε σχετική με την `10.1.1.14` εγγραφή, ούτε προκαθορισμένη διαδρομή, οπότε απαντά με μήνυμα λάθους "Destination Host Unreachable".

7.18

Στον R2 εκτελούμε:

```
no network 0.0.0.0/0
ip route 0.0.0.0/0 172.17.17.2
```

7.19

Στους R1,2,3,4 εκτελούμε `do show ip bgp`. Βλέπουμε ότι πλέον ο τύπος πηγής ORIGIN είναι `incomplete`.

7.20

Στον R2 εκτελούμε `do show running-config`. Βλέπουμε ότι η εντολή `redistribute static` έχει ήδη δοθεί σε προηγούμενο ερώτημα (συγκεκριμένα στο ερώτημα 3.42).

7.21

Αν στον PC1 εκτελέσουμε `do ping 10.1.1.14` θα λάβουμε μήνυμα λάθους "Time to live exceeded" από τον R2. Αυτό συμβαίνει διότι ο PC1 θα προωθήσει το ICMP request μέσω της προκαθορισμένης διαδρομής στον R1, ο R1 θα το προωθήσει μέσω της προκαθορισμένης διαδρομής στον R2, και ο R2 θα το προωθεί συνεχώς μέσω της προκαθορισμένης διαδρομής στην loopback του, μέχρι να μηδενιστεί το TTL και να σταλεί μήνυμα λάθους TTL exceeded.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 10
Τείχη προστασίας (Firewalls) και NAT

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 17/05/2022

Άσκηση 1: Ένα απλό τείχος προστασίας

Προετοιμασία στο σπίτι

Στο νέο FreeBSD 11.4 εκτελούμε:

```
sysrc ifconfig_em0="192.168.1.1/24"
sysrc ifconfig_em1="192.0.2.1/30"
sysrc defaultrouter="192.0.2.2"
sysrc gateway_enable="YES"
sysrc firewall_enable="YES"
sysrc firewall_nat_enable="YES"
sysrc firewall_logif="YES"
```

1.1

Στο PC1 εκτελούμε `kldload ipfw`.

1.2

Με την εντολή `kldstat | grep ipfw` ή `kldstat -m ipfw`.

1.3

Στο PC1 εκτελούμε:

```
ping 127.0.0.1
ping 192.168.1.2
```

Και τα δύο ping αποτυγχάνουν, ενώ εμφανίζεται μήνυμα λάθους ”Permission denied”.

1.4

Στο PC1 εκτελούμε `ipfw list`. Έχουμε:

```
65535 deny ip from any to any
```

1.5

Στο PC1 εκτελούμε `ipfw show`. Έχουμε:

```
65535 2 168 deny ip from any to any
```

Οι μετρητές είναι: `2 168`.

1.6

Με την εντολή `ipfw zero`.

1.7

Στο PC1 εκτελούμε:

```
ipfw add 100 allow all from any to any via lo0
```

1.8

Στο PC1 εκτελούμε:

```
ping 127.0.0.1
ping 192.168.1.2
```

Τώρα τα ping είναι επιτυχή.

1.9

Στο PC1 εκτελούμε `ping 192.168.1.3`. Το ping αποτυγχάνει με μήνυμα "Permission denied".

1.10

Στο PC1 εκτελούμε `ipfw add allow icmp from any to any`.

1.11

Έλαβε αύξοντα αριθμό 200, όπως φαίνεται από την έξοδο της παραπάνω εντολής (εναλλακτικά θα μπορούσαμε να εκτελέσουμε και `ipfw list`).

1.12

Εκτελούμε:

```
### PC1 ###

ping 192.168.1.3

### PC2 ###

ping 192.168.1.2
```

Και τα δύο ping είναι επιτυχή.

1.13

Στο PC1 εκτελούμε `traceroute 192.168.1.3` και επιβεβαιώνουμε ότι δεν μπορούμε, αφού εμφανίζεται μήνυμα λάθους "Permission denied". Αυτό συμβαίνει διότι by default η εντολή traceroute στέλνει πακέτα UDP, και δεν υπάρχει κανόνας στο firewall που επιτρέπει να περάσουν. Αλλάζοντας την εντολή σε:

```
traceroute -I 192.168.1.3
```

πλέον στέλνουμε ICMP πακέτα, τα οποία επιτρέπει ο κανόνας υπ' αριθμόν 200 που προσθέσαμε προηγουμένως.

1.14

Το traceroute χρησιμοποιεί τις θύρες με αριθμούς από

```
port + 1
```

μέχρι και

```
port + (max_ttl - first_ttl + 1) * nprobes
```

Αν αντικαταστήσουμε τις default τιμές:

```
port = 33434
max_ttl = 64
first_ttl = 1
nprobes = 3
```

βρίσκουμε ότι η κλήση της traceroute χωρίς παραμέτρους χρησιμοποιεί τις θύρες 33435-33626. Επομένως εκτελούμε στο PC1:

```
ipfw add allow udp from me to any 33435-33626
```

1.15

Στο PC1 εκτελούμε `ssh lab@192.168.1.3`. Το ssh αποτυγχάνει με μήνυμα λάθους "Permission denied".

1.16

Στο PC1 εκτελούμε:

```
ipfw add allow tcp from me to any setup
ipfw add allow tcp from any to any established
```

1.17

Στο PC1 εκτελούμε:

```
ipfw zero
ssh lab@192.168.1.3
ls
exit
```

1.18

Έχουμε:

```
No.          Rule                                    Times applied
00100        allow ip from any to any via lo0        0
00200        allow icmp from any to any              0
00300        allow udp from me to any 33435-33626    0
00400        allow tcp from me to any setup          1
00500        allow tcp from any to any established   68
```

- Ο κανόνας 100 δεν χρησιμοποιήθηκε καμία φορά, αφού κατά τη σύνδεση με ssh δεν χρησιμοποιείται η `lo0` από το PC1.
- Ο κανόνας 200 δεν χρησιμοποιήθηκε καμία φορά, αφού κατά τη σύνδεση με ssh δεν στέλνονται πακέτα ICMP.
- Ο κανόνας 300 δεν χρησιμοποιήθηκε καμία φορά, αφού κατά τη σύνδεση με ssh δεν στέλνονται πακέτα UDP.
- Ο κανόνας 400 χρησιμοποιήθηκε 1 φορά, στο πρώτο τεμάχιο της TCP χειραψίας, που προέρχεται από το PC1 (με σημαία SYN, όχι σημαία ACK).
- Ο κανόνας 500 χρησιμοποιήθηκε 68 φορές για όλα τα υπόλοιπα τεμάχια TCP που ανήκουν στη σύνδεση που εγκαταστάθηκε (με σημαία ACK ή RST).

1.19

Στο PC2 εκτελούμε `ssh lab@192.168.1.2`. Η εντολή αρχικά φαίνεται να ”κολλάει” και ύστερα από κάποιο χρονικό διάστημα εμφανίζεται μήνυμα ”Operation timed out”. Αυτό είναι λογικό, αφού κανένας από τους κανόνες που έχουμε ορίσει στο firewall του PC1 δεν επιτρέπουν TCP τεμάχια με σημαίες SYN=1, ACK=0 από απομακρυσμένα μηχανήματα, παρά μόνο από το PC1.

1.20

Στο PC2 εκτελούμε `service ftpd onestart`.

1.21

Στο PC1 εκτελούμε:

```
ftp lab@192.168.1.3
get /usr/bin/grep
exit
```

Μπορούμε να κατεβάσουμε το αρχείο με επιτυχία.

Άσκηση 2: Ένα πιο σύνθετο τείχος προστασίας

2.1

Στο PC2 εκτελούμε `kldload ipfw`.

2.2

Στο PC2 εκτελούμε `ping 192.168.1.2`. Δεν μπορούμε, καθώς εμφανίζεται μήνυμα λάθους "Permission denied".

2.3

Στο PC2 εκτελούμε:

```
ipfw add allow all from any to any via lo0
```

2.4

Στο PC2 εκτελούμε:

```
ipfw add allow icmp from me to any icmptypes 8
```

2.5

Στο PC2 εκτελούμε `ping 192.168.1.2`. Δεν λαμβάνεται κάποια απάντηση.

2.6

Στο PC2 εκτελούμε:

```
ipfw zero
ping -c 1 192.168.1.2
ipfw show
```

Ο σχετικός μετρητής στον παραπάνω κανόνα αυξήθηκε κατά 1, οπότε τα πακέτα ICMP περνούν το τείχος προστασίας του PC2.

2.7

Στο PC2 εκτελούμε:

```
ipfw delete 200
ipfw add allow icmp from me to any icmptypes 8 keep-state
ping 192.168.1.2
```

Πλέον το ping είναι επιτυχές.

2.8

Εκτελούμε:

```
### PC2 ###

ping 192.168.1.3

### PC1 ###

ping 192.168.1.2
```

Μπορούμε να κάνουμε ping από το PC1 στο PC2.

2.9

Σταματάμε τα ping και ύστερα από λίγο εκτελούμε `ping 192.168.1.3` στο PC1. Πλέον το ping από το PC1 στο PC2 αποτυγχάνει, γιατί διακόπηκε η κίνηση που ανανέωνε τη διάρκεια του δυναμικού κανόνα που δημιουργήθηκε λόγω του ping από το PC2 στο PC1, και ο οποίος επέτρεπε την αμφίδρομη επικοινωνία μεταξύ PC1 και PC2. Έτσι, τα πακέτα ICMP request του PC1 πλέον απορρίπτονται από το τείχος προστασίας του PC2.

2.10

Στο PC2 εκτελούμε:

```
ipfw add allow icmp from any to me icmptypes 8 keep-state
```

2.11

Στο PC1 εκτελούμε `ping 192.168.1.3`. Στο PC2 εκτελούμε `ipfw -d show`. Βλέπουμε ότι, εκτός από τους στατικούς κανόνες, εμφανίζονται και οι δυναμικοί κανόνες του τείχους προστασίας:

```
## Dynamic rules (1 136):
00300   34   2856 (5s) STATE icmp 192.168.1.2 0 <-> 192.168.1.3 0 :default
```

2.12

Σταματάμε το ping από το PC1 και εκτελούμε ξανά `ipfw -d show` στο PC2. Βλέπουμε ότι ο δυναμικός κανόνας έχει διαγραφεί.

2.13

Στο PC2 εκτελούμε:

```
ipfw add allow udp from any to me 33435-33626
ipfw add allow icmp from me to any icmptypes 3
```

2.14

Στο PC2 εκτελούμε:

```
ipfw add allow udp from me to any 33435-33626
ipfw add allow icmp from any to me icmptypes 3
```

2.15

Πρέπει να προσθέσουμε τον κανόνα (στο PC1):

```
ipfw add allow udp from any to me 33435-33626
```

2.16

Στο PC2 εκτελούμε:

```
ipfw add allow tcp from 192.168.1.0/24 to me 22 keep-state
```

2.17

Με την εντολή `ssh lab@192.168.1.3`.

2.18

Στο PC2 εκτελούμε:

```
ipfw add allow tcp from me to any 22 keep-state
```

2.19

Πρέπει να προσθέσουμε στο PC1 τον κανόνα:

```
ipfw add allow tcp from 192.168.1.0/24 to me 22 setup keep-state
```

2.20

Στο PC1 εκτελούμε:

```
sftp lab@192.168.1.2
get /etc/rc.conf
```

Μπορούμε να κατεβάσουμε το `/etc/rc.conf`.

2.21

Στο PC1 εκτελούμε `ftp lab@192.168.1.2`. Λαμβάνουμε μήνυμα λάθους "Connection refused", οπότε εκτελούμε στο PC2:

```
ipfw add allow tcp from 192.168.1.0/24 to me 21 keep-state
```

2.22

Στο PC1 εκτελούμε:

```
ftp lab@192.168.1.3
cd /usr
ls
```

Η `cd /usr` εκτελείται επιτυχώς, αφού υπάρχει κανόνας στο firewall του PC2 για τη θύρα 21, ενώ η `ls` αποτυγχάνει, γιατί εκτελώντας την μπαίνουμε Extended Passive Mode, στο οποίο επιλέγεται δυναμικά μια θύρα για τη σύνδεση δεδομένων, για την οποία δεν υπάρχει αντίστοιχος κανόνας στο firewall του PC2.

2.23

Σύμφωνα με την ιστοσελίδα της υπόδειξης, για να μπορεί ο FTP server (PC2) να υποστηρίξει παθητικό FTP, πρέπει εκτελέσουμε στον PC2:

```
ipfw add allow tcp from 192.168.1.0/24 to me 1024-65535 keep-state
```

2.24

Στο PC1 εκτελούμε:

```
ftp lab@192.168.1.3
get /usr/bin/grep
exit
```

Μπορούμε να κατεβάσουμε το αρχείο με επιτυχία.

2.25

Πρέπει να προσθέσουμε τους κανόνες:

```
### PC1 ###

ipfw add allow tcp from 192.168.1.0/24 20 to me keep-state

### PC2 ###

ipfw add allow tcp from me 20 to 192.168.1.0/24 keep-state
```

2.26

Φαίνεται ότι σε πρωτόκολλα όπως το FTP χρειάζεται να ορίσουμε πολλούς κανόνες, επειδή χρησιμοποιούνται πολλές θύρες ανάλογα τον τρόπο λειτουργίας (σύνδεση δεδομένων, σύνδεση εντολών/ελέγχου, ενεργητικός/παθητικός τρόπος λειτουργίας), ενώ άλλες φορές η σύνδεση εκκινείται από τον πελάτη και άλλες από τον εξυπηρετητή. Όλα αυτά κάνουν τη διαχείριση των κανόνων κάπως δύστροπη και επιρρεπή σε λάθη.

2.27

Εκτελούμε:

```
### PC1 ###

kldunload ipfw
kldstat

### PC2 ###

kldunload ipfw
kldstat
```

Επιβεβαιώνουμε ότι το `ipfw` έχει απενεργοποιηθεί και στα δύο μηχανήματα.

Άσκηση 3: Απλό Network Address Translation

3.1

Εκτελούμε:

```
### PC1 ###

hostname PC1
ifconfig em0 192.168.1.2/24
route add default 192.168.1.1

### PC2 ###

hostname PC2
ifconfig em0 192.168.1.3/24
route add default 192.168.1.1
```

3.2

Στον R1 εκτελούμε:

```
cli
configure terminal
```

```
hostname R1

interface em0
ip address 192.0.2.2/30

interface em1
ip address 192.0.2.6/30
```

3.3

Στον SRV1 εκτελούμε:

```
hostname SRV1

ifconfig em0 192.0.2.5/30

route add default 192.0.2.6
```

3.4

Εκτελούμε:

```
### PC2 ###

service ftpd onestart

### SRV1 ###

service ftpd onestart
```

3.5

Στο FW1 εκτελούμε `kldstat`. Έχουν φορτωθεί τα modules:

```
kernel
ipfw
ipfw_nat
libalias
```

3.6

Ενεργοποιήθηκε το IPFW.

3.7

Στο FW1 εκτελούμε `sysrc firewall_type` και έχουμε:

```
firewall_type: UNKNOWN
```

3.8

Στο FW1 εκτελούμε `ipfw list`. Βλέπουμε 11 κανόνες. Ο τελευταίος είναι ο default κανόνας (αύξων αριθμός 65535):

```
65535 deny ip from any to any
```

3.9

Με την εντολή `ip nat show config`. Βλέπουμε ότι δεν έχει οριστεί κανένας πίνακας in-kernel NAT.

3.10

Στο PC1 εκτελούμε:

```
ping 192.168.1.1
ping 192.0.2.1
```

Σε κανένα από τα δύο ping δε λαμβάνουμε απάντηση.

3.11

Στον SRV1 εκτελούμε `ping 192.0.2.1`. Δε λαμβάνουμε απάντηση.

3.12

Στο FW1 εκτελούμε:

```
ipfw nat 123 config unreg_only if em1 reset
```

3.13

Στο FW1 εκτελούμε:

```
ipfw add nat 123 ip4 from any to any
```

3.14

Στο PC1 εκτελούμε:

```
ping 192.168.1.1
ping 192.0.2.1
```

Πλέον και τα δύο ping είναι επιτυχή.

3.15

Στον R1 εκτελούμε `tcpdump -i em0`.

3.16

Στο FW1 εκτελούμε `ipfw show && ipfw zero`.

3.17

Στο PC1 εκτελούμε `ping -c 3 192.0.2.2`. Η IP διεύθυνση πηγής των ICMP echo request είναι `192.0.2.1` (FW1@em1).

3.18

Είναι πάλι η `192.0.2.1` (FW1@em1).

3.19

Εκτελούμε `ipfw show` στο FW1. Υπεύθυνος για την επιτυχία του ping είναι ο κανόνας:

```
01100 nat 123 ip4 from any to any
```

3.20

Εφαρμόστηκε 124 φορές. Αυτό συμβαίνει διότι στάλθηκαν συνολικά 31 (δηλαδή $124 \div 4$) ICMP echo request από το PC1, και κάθε φορά που στέλνεται ένα ICMP echo request συμβαίνουν τα εξής:

- Εφαρμόζεται ο κανόνας για το ICMP echo request με κατεύθυνση PC1 $\rightarrow$ FW1
- Εφαρμόζεται ο κανόνας για το ICMP echo request με κατεύθυνση FW1 $\rightarrow$ R1
- Εφαρμόζεται ο κανόνας για το αντίστοιχο ICMP echo reply με κατεύθυνση R1 $\rightarrow$ FW1
- Εφαρμόζεται ο κανόνας για το αντίστοιχο ICMP echo reply με κατεύθυνση FW1 $\rightarrow$ PC1

3.21

Στον SRV1 εκτελούμε `ping 192.0.2.1`. Το ping είναι επιτυχές.

3.22

Εκτελούμε:

```
### FW1 ###

ipfw zero

### SRV1 ###

ping -c 1 192.0.2.1

### FW1 ###

ipfw show
```

Βλέπουμε ότι υπεύθυνος για την αποδοχή της κίνησης είναι ο ίδιος κανόνας:

```
01100  2  168 nat 123 ip4 from any to any
```

3.23

Η κίνηση δεν ωθείται προς μετάφραση διευθύνσεων διότι η διεύθυνση `192.0.2.5` του SRV1 δεν είναι ιδιωτική. Υπενθυμίζουμε ότι στην προηγούμενη εντολή ορισμού του πίνακα NAT έχουμε δώσει τη ρύθμιση `unreg_only`.

3.24

Στο PC2 εκτελούμε `ssh lab@192.0.2.5`. Μπορούμε να συνδεθούμε κανονικά.

3.25

Στον SRV1 εκτελούμε `ssh lab@192.168.1.3`. Η εντολή φαίνεται αρχικά να μπλοκάρει, και ύστερα από λίγο εμφανίζεται μήνυμα "No route to host". Ο λόγος που δεν μπορεί να επιτευχθεί η σύνδεση είναι διότι ο SRV1 προωθεί αρχικά τα πακέτα στον R1 μέσω της προκαθορισμένης διαδρομής, αλλά ο R1 δεν διαθέτει εγγραφή σχετική με την (ιδιωτική) διεύθυνση `192.168.1.3`. Μπορούμε να επιβεβαιώσουμε τα παραπάνω αν εκτελέσουμε `traceroute 192.168.1.3` στον SRV1. Η έξοδος της εντολής είναι:

```
1  192.0.2.6 (192.0.2.6) 0.336 ms  0.150 ms  0.115 ms
2  192.0.2.6 (192.0.2.6) 0.114 ms !H  0.192 ms !H  0.089 ms !H
```

όπου το `"!H"` υποδηλώνει ότι ο προορισμός δεν είναι προσβάσιμος από τον R1.

3.26

Στο FW1 εκτελούμε:

```
ipfw nat 123 config unreg_only if em1 reset redirect_addr 192.168.1.3 192.0.2.1
```

3.27

Στον SRV1 εκτελούμε `ssh lab@192.0.2.1`. Η προσπάθεια είναι επιτυχής. Έχουμε συνδεθεί στο PC2, όπως φαίνεται και από το prompt στη γραμμή εντολών:

```
lab@PC2:~ %
```

Μπορούμε επίσης να εκτελέσουμε `ifconfig em0` και να δούμε ότι η διεύθυνση IP της `em0` είναι `192.168.1.3`.

3.28

Στο FW1 εκτελούμε (όλο μαζί μία εντολή):

```
ipfw nat 123 config unreg_only if em1 reset \
redirect_addr 192.168.1.3 192.0.2.1 \
redirect_port tcp 192.168.1.2:22 22
```

3.29

Στον SRV1 εκτελούμε πάλι `ssh lab@192.0.2.1`. Αυτή τη φορά συνδεθήκαμε στο PC1, όπως φαίνεται από το prompt:

```
lab@PC1:~ %
```

ή από την έξοδο της εντολής `ifconfig em0`.

3.30

Στον SRV1 εκτελούμε `ftp lab@192.0.2.1`. Στην προτροπή για κωδικό εμφανίζεται:

```
220 PC2 FTP server (Version 6.00LS) ready.
```

οπότε καταλαβαίνουμε ότι έχουμε συνδεθεί από το PC2. Εναλλακτικά, μπορούμε μέσω ftp στον SRV1 να εκτελέσουμε `rstatus`. Στην πρώτη γραμμή της εξόδου της `rstatus` εμφανίζεται:

```
211- PC2 FTP server status:
```

Επίσης μπορούμε να εκτελέσουμε:

```
### PC1 ###

netstat | grep ftp

### PC2 ###

netstat | grep ftp
```

Στο PC1 η έξοδος της εντολής είναι κενή, ενώ στο PC2 εμφανίζεται:

```
tcp4    0    0    192.0.2.5.52490    192.0.2.1.ftp ESTABLISHED
```

3.31

Στον SRV1 εκτελούμε:

```
ls /etc
get /etc/rc.conf
```

Και οι δύο εντολές εκτελούνται με επιτυχία.

3.32

Στο PC1 εκτελούμε `ftp lab@192.0.2.1`. Με τους ίδιους τρόπους (βλ. 3.30) μπορούμε να εξακριβώσουμε ότι απαντά το PC2.

3.33

Στο PC2 εκτελούμε `ssh lab@192.0.2.1`. Συνδεόμαστε στο PC1, όπως φαίνεται από το prompt:

```
lab@PC1:~ %
```

ή από την έξοδο της εντολής `ifconfig em0`.

Άσκηση 4: Τείχος προστασίας και NAT

4.1

Εκτελούμε:

```
### FW1 ###

ipfw disable one_pass

### PC1 ###

ping 192.168.1.1

### SRV1 ###

ping 192.0.2.1
```

Σε κανένα από τα δύο ping δεν λαμβάνουμε απάντηση.

4.2

Στο FW1 εκτελούμε επανειλημμένα `ipfw show` και βλέπουμε ότι όσο το ping εκτελείται, οι μετρητές χρήσης του κανόνα `01100 nat 123 ip4 from any to any` αυξάνονται. Ωστόσο, επειδή μόλις απενεργοποιήσαμε τη λειτουργία one-pass, ύστερα από τη μετάφραση διευθύνσεων, η επεξεργασία των πακέτων συνεχίζει με τον επόμενο κανόνα του τείχους προστασίας. Έτσι, τα πακέτα περνούν από τον default κανόνα (αριθμός 65535), ο οποίος απορρίπτει όλη την κίνηση.

4.3

Στο FW1 εκτελούμε:

```
ipfw delete 1100
ipfw add 1100 allow all from any to any via em0
```

4.4

Στο PC1 εκτελούμε:

```
ping 192.168.1.1
ping 192.0.2.1
```

Και τα δύο ping είναι επιτυχή.

4.5

Στο PC2 εκτελούμε `ssh lab@192.0.2.1`. Με τις εντολές `hostname` ή/και `ifconfig em0` μπορούμε να διαπιστώσουμε ότι έχουμε συνδεθεί στο FW1.

4.6

Στο FW1 εκτελούμε `ipfw show`. Οι μετρητές που είναι μη-μηδενικοί δείχνουν και τους κανόνες που χρησιμοποιήθηκαν, οι οποίοι είναι:

```
01100 allow ip from any to any via em0
65535 deny ip from any to any
```

4.7

Στο FW1 εκτελούμε:

```
ipfw add 3000 nat 123 all from any to any xmit em1
```

4.8

Στο FW1 εκτελούμε:

```
ipfw add 3001 allow all from any to any
```

4.9

Στο FW1 εκτελούμε:

```
ipfw add 2000 nat 123 all from any to any recv em1
```

4.10

Στο FW1 εκτελούμε:

```
ipfw add 2001 check-state
```

4.11

Στο PC1 εκτελούμε `ping 192.0.2.1`. Απαντά το FW1.

4.12

Στον SRV1 εκτελούμε `ping 192.0.2.1`. Απαντά το PC2.

4.13

Στο PC1 εκτελούμε `ssh lab@192.0.2.1`. Συνδεόμαστε στο FW1.

4.14

Στον SRV1 εκτελούμε `ssh lab@192.0.2.1`. Συνδεόμαστε στο PC1.

4.15

Στον SRV1 εκτελούμε `ftp lab@192.0.2.1`. Συνδεόμαστε στο PC2.

4.16

Στο PC1 εκτελούμε `ping 192.0.2.5`. Το ping είναι επιτυχές.

4.17

Στο PC1 εκτελούμε `ssh lab@192.0.2.5`. Μπορούμε να συνδεθούμε κανονικά.

4.18

Στο PC1 εκτελούμε:

```
ftp lab@192.0.2.5
ls /etc
get /etc/rc.conf
```

Μπορούμε να εκτελέσουμε όλες αυτές τις ενέργειες.

4.19

Στο FW1 εκτελούμε:

```
ipfw add 2999 deny all from any to any via em1
```

4.20

Εκτελούμε:

```
### 4.11 ###

PC1: ping 192.0.2.1          # SUCCESSFUL

### 4.12 ###

SRV1: ping 192.0.2.1         # FAILED

### 4.13 ###

PC1: ssh lab@192.0.2.1       # SUCCESSFUL

### 4.14 ###
```

```
SRV1: ssh lab@192.0.2.1      # FAILED

### 4.15 ###

SRV1: ftp lab@192.0.2.1      # FAILED

### 4.16 ###

PC1: ping 192.0.2.5          # FAILED

### 4.17 ###

PC1: ssh lab@192.0.2.5       # FAILED

### 4.18 ###

PC1: ftp lab@192.0.2.5       # FAILED
```

Βλέπουμε ότι μόνο οι εντολές των ερωτημάτων 4.11 και 4.13 επιτυγχάνουν.

4.21

Στο FW1 εκτελούμε:

```
ipfw add 2500 skipto 3000 icmp from any to any xmit em1 keep-state
```

4.22

Στο PC1 εκτελούμε `ping 192.0.2.5`. Το ping είναι επιτυχές.

4.23

Στο FW1 εκτελούμε:

```
ipfw add 2600 skipto 3000 tcp from any to any 22 out via em1 keep-state
```

4.24

Στο PC1 εκτελούμε `ssh lab@192.0.2.5`. Μπορούμε να συνδεθούμε κανονικά.

4.25

Στο FW1 εκτελούμε:

```
ipfw add 2100 skipto 3000 icmp from any to any in via em1 keep-state
```

4.26

Στον SRV1 εκτελούμε `ping 192.0.2.1`. Απαντά το PC2.

4.27

Στο FW1 εκτελούμε:

```
ipfw add 2200 skipto 3000 tcp from any to any 22 recv em1 keep-state
```

4.28

Στον SRV1 εκτελούμε `ssh lab@192.0.2.1`. Συνδεόμαστε στο PC1.

4.29

Στον SRV1 εκτελούμε `ftp lab@192.0.2.1`. Το ftp αποτυγχάνει.

4.30

Πρέπει να προσθέσουμε τους κανόνες:

```
ipfw add 2700 skipto 3000 tcp from any to any 21 recv em1 keep-state
ipfw add 2701 skipto 3000 tcp from any 20 to any out via em1 keep-state
```

Άσκηση 5: Τείχος προστασίας με γραφικό περιβάλλον διαχείρισης

5.1

Από το γραφικό περιβάλλον:

```
Interfaces - LAN - Primary configuration - IP address
```

Έχει ρυθμιστεί η διεύθυνση `192.168.1.1/24`.

5.2

Από το γραφικό περιβάλλον:

```
Interfaces - WAN - Static IP configuration - IP address
```

Έχει ρυθμιστεί η διεύθυνση `10.0.0.1/30`.

5.3

Από το γραφικό περιβάλλον:

```
Status - System - Memory Usage
```

Χρησιμοποιείται το 34% της μνήμης, επομένως το ποσοστό της ελεύθερης μνήμης είναι 66%.

5.4

Από το γραφικό περιβάλλον:

`Interfaces (assign)`

Έχουμε:

```
Interface Network Port
LAN       em0 (Intel(R)...
WAN       em1 (Intel(R)...
MNG       em2 (Intel(R)...
DMZ       em3 (Intel(R)...
```

οπότε συνολικά βλέπουμε 4 διεπαφές. Επιβεβαιώνουμε ότι στο VirtualBox οι κάρτες δικτύου είναι σωστά ρυθμισμένες.

5.5

Από το γραφικό περιβάλλον:

`Interfaces - DMZ - IP configuration - IP address`

Έχει ρυθμιστεί η διεύθυνση `172.22.1.1/24`.

5.6

Από το γραφικό περιβάλλον:

`System - General setup - Hostname`

Το όνομα είναι `fw`.

5.7

Από το γραφικό περιβάλλον:

`System - General setup - Hostname`

Αλλάζουμε την τιμή σε ”fw1” και πατάμε ”Save”.

5.8

Από το γραφικό περιβάλλον:

`Firewall - Rules - WAN`

Δεν υπάρχουν κανόνες για το WAN.

5.9

Από το γραφικό περιβάλλον:

```
Interfaces - WAN
```

Στο υπο-μενού ”Static IP configuration” αλλάζουμε τις τιμές των πεδίων ”IP address” και ”Gateway” σε `192.0.2.1/30` και `192.0.2.2` αντίστοιχα, και ύστερα στο κάτω μέρος ενεργοποιούμε την επιλογή ”Block private networks”. Τέλος, πατάμε ”Save” για την εφαρμογή των αλλαγών.

5.10

Από το γραφικό περιβάλλον:

```
Firewall - Rules - WAN
```

Πλέον εμφανίζεται ο κανόνας:

```
Action  Proto   Source              Port   Destination   Port   Description
Block   *       RFC 1918 networks   *      *             *      Block private
                                                                 networks
```

5.11

Από το γραφικό περιβάλλον:

```
Services - DNS forwarder
Services - Dynamic DNS
Services - DHCP server
Services - DHCP relay
Services - SNMP
Services - Proxy ARP
Services - Captive portal
Services - Wake on LAN
Services - Scheduler
VPN - IPsec
VPN - PPTP
```

Καμία υπηρεσία δεν είναι ενεργοποιημένη.

5.12

Από το γραφικό περιβάλλον:

```
Services - DNS forwarder
```

Ενεργοποιούμε την επιλογή ”Enable DNS forwarder” και πατάμε ”Save”.

5.13

Από το γραφικό περιβάλλον:

`Services - DHCP server - LAN`

Τσεκάρουμε το κουτί ”Enable” και αλλάζουμε το πεδίο ”Range”:

`Range: 192.168.1.2 to 192.168.1.3`

Τέλος, πατάμε ”Save”.

5.14

Στο PC1 εκτελούμε:

```
dhclient em0
ifconfig em0 | grep inet
netstat -rn | grep default
cat /etc/resolv.conf | grep nameserver
```

Έχουμε:

```
IP address: 192.168.1.2
Default gateway: 192.168.1.1
DNS server address: 192.168.1.1
```

5.15

Χρειάστηκε προκειμένου η υπηρεσία DHCP να λειτουργεί και ως εξυπηρετητής DNS για τους πελάτες DHCP στο LAN, αλλιώς θα έπρεπε να ορίσουμε εξυπηρετητή DNS χειροκίνητα.

5.16

Στο μενού:

`Diagnostics - DHCP Leases`

5.17

Από το γραφικό περιβάλλον:

`Diagnostics - ARP Table`

Βλέπουμε 9 εγγραφές:

IP address	MAC address	Interface
172.22.1.2	08:00:27:62:6b:12	DMZ
172.22.1.1	08:00:27:2b:a4:d8	DMZ
192.168.56.1	0a:00:27:00:00:00	MNG
192.168.56.2	08:00:27:78:b0:f7	MNG
192.0.2.2	08:00:27:b1:c8:2f	WAN
192.0.2.1	08:00:27:cc:e3:72	WAN
192.168.1.1	08:00:27:03:6b:29	LAN
192.168.1.3	08:00:27:5e:eb:81	LAN
192.168.1.2	08:00:27:75:42:4f	LAN

5.18

Στο PC1 εκτελούμε `ping 192.168.1.1`. Δεν λαμβάνουμε απάντηση.

5.19

Από το γραφικό περιβάλλον:

`Diagnostics - Logs - Firewall`

Βλέπουμε τις 50 πιο πρόσφατες καταγραφές πακέτων που έχουν περάσει από το firewall, και την αντίστοιχη ενέργεια που εκτελέστηκε για κάθε πακέτο. Βλέπουμε ότι οι πρώτες 3 γραμμές του πίνακα αφορούν το ping που εκτελέσαμε προηγουμένως:

```
Act      If       Source           Destination                 Proto
X        LAN      192.168.1.2      192.168.1.1, type echo/0    ICMP
X        LAN      192.168.1.2      192.168.1.1, type echo/0    ICMP
X        LAN      192.168.1.2      192.168.1.1, type echo/0    ICMP
```

Καθαρίζουμε το αρχείο καταγραφών από το γραφικό περιβάλλον:

`Diagnostics - Logs - Firewall - Clear log`

5.20

Από το γραφικό περιβάλλον:

`Diagnostics - Firewall states`

Βλέπουμε 2 firewall states.

5.21

Από το γραφικό περιβάλλον:

`Firewall - Rules - LAN`

Δεν βλέπουμε κανέναν κανόνα.

5.22

Από το γραφικό περιβάλλον:

`Firewall - Rules - LAN`

Πατάμε το κουμπί ”+” και τροποποιούμε τα πεδία ως εξής:

```
Action: Pass
Interface: LAN
Protocol: any
Source:
    Type: LAN subnet
Destination:
    Type: any
```

Τέλος πατάμε ”Save” και ύστερα ”Apply changes”.

5.23

Στο PC1 εκτελούμε:

```
ping 192.168.1.1    # LAN1, successful
ping 192.0.2.1      # WAN1, successful
ping 172.22.1.1     # DMZ, successful
```

Και τα τρία ping είναι επιτυχή.

5.24

Στον R1 εκτελούμε `do ping 192.0.2.1`. Δεν λαμβάνουμε απάντηση.

5.25

Στον R1 εκτελούμε `arp -a`. Έχουμε:

```
IP address       MAC address             Interface
192.0.2.2        08:00:27:b1:c8:2f       em0
192.0.2.1        08:00:27:cc:e3:72       em0
```

Ναι, η δεύτερη εγγραφή αφορά τη ζητούμενη διεύθυνση MAC.

5.26

Από το γραφικό περιβάλλον:

`Firewall - Rules - WAN`

Πατάμε το κουμπί ”+” και τροποποιούμε τα πεδία ως εξής:

```
Action: Pass
Interface: WAN
Protocol: ICMP
ICMP type: any
Source:
    Type: any
Destination:
    Type: WAN address
```

Πατάμε ”Save” και ”Apply changes”.

5.27

Στον R1 εκτελούμε `do ping 192.0.2.1`. Το ping είναι επιτυχές.

5.28

Στον R1 εκτελούμε `do ping 192.168.1.2`. Εμφανίζεται μήνυμα λάθους ”No route to host”, αφού δεν υπάρχει εγγραφή στον πίνακα δρομολόγησης σχετική με την (ιδιωτική) διεύθυνση `192.168.1.2`.

5.29

Στο PC1 εκτελούμε `ping 192.0.2.2`. Το ping είναι επιτυχές. Συμπεραίνουμε ότι στο FW1, κατά την κατεύθυνση PC1 → R1 αντικαθίσταται η εσωτερική τοπική διεύθυνση IP του PC1 με την διεύθυνση της διεπαφής του FW1 στο WAN1, γι' αυτό και ο R1 μπορεί να απαντήσει κανονικά με ICMP reply. Επομένως έχουμε παραδοσιακό/απερχόμενο NAT (Traditional/Outbound NAT), .

5.30

Έχουμε ήδη τοποθετήσει τον SRV1 στο DMZ και έχουμε ήδη ορίσει τη διεύθυνση IP του με την εντολή:

```
ifconfig em0 172.22.1.2/24
```

Στο PC1 εκτελούμε `ping 172.22.1.2`. Δε λαμβάνουμε απάντηση. Αυτό συμβαίνει επειδή δεν γίνεται μετάφραση διευθύνσεων προς το DMZ και δεν υπάρχει εγγραφή σχετική με την εσωτερική τοπική διεύθυνση `192.168.1.2` του PC1 στον πίνακα δρομολόγησης του SRV1, επομένως δεν μπορεί να απαντήσει στα ICMP request του PC1, τα οποία όμως λαμβάνει κανονικά.

5.31

Στον SRV1 εκτελούμε `route add default 172.22.1.1`.

5.32

Στο PC1 εκτελούμε `ping 172.22.1.2`. Πλέον το ping είναι επιτυχές.

5.33

Στον SRV1 εκτελούμε `ping 172.22.1.1`. Δε λαμβάνουμε απάντηση. Αυτό συμβαίνει επειδή δεν έχουμε ορίσει κανέναν κανόνα στο FW1 που να επιτρέπει την εξερχόμενη κίνηση από το DMZ, οπότε τα ICMP reply απορρίπτονται.

5.34

Στον SRV1 εκτελούμε:

```
ping 192.168.1.2
ping 192.0.2.2
```

Σε κανένα από τα δύο ping δε λαμβάνουμε απάντηση. Αυτό συμβαίνει πάλι επειδή δεν έχουμε ορίσει κανέναν κανόνα στο FW1 που να επιτρέπει την εξερχόμενη κίνηση από το DMZ, οπότε τα ICMP reply απορρίπτονται.

5.35

Από το γραφικό περιβάλλον:

```
Firewall - Rules - DMZ
```

Πατάμε το ”+” και τροποποιούμε τα πεδία ως εξής (το `[v]` παρακάτω σημαίνει ότι το κουτάκι είναι τσεκαρισμένο):

```
Action: Pass
Interface: DMZ
Protocol: any
Source:
    Type: DMZ subnet
Destination: [v] not
    Type: LAN subnet
```

Πατάμε ”Save” και ”Apply changes”.

5.36

Στον SRV1 εκτελούμε `ping 172.22.1.1`. Το ping είναι επιτυχές.

5.37

Στον SRV1 εκτελούμε `ping 192.0.2.1`. Το ping είναι επιτυχές.

5.38

Στον R1 εκτελούμε `do ping 172.22.1.2`. Το ping αποτυγχάνει με μήνυμα ”No route to host”. Αυτό είναι λογικό, καθώς δεν υπάρχει εγγραφή σχετική με τη διεύθυνση `172.22.1.2` στον πίνακα δρομολόγησης του R1.

5.39

Στον SRV1 εκτελούμε `ping 192.0.2.2`. Το ping είναι επιτυχές. Αυτό συμβαίνει διότι τα πακέτα περνούν από το FW1 και εκεί γίνεται μετάφραση διευθύνσεων, στην οποία η διεύθυνση πηγής των ICMP request αλλάζει από `172.22.1.2` σε `192.0.2.1`, και έτσι ο R1 μπορεί πλέον να απαντήσει με ICMP reply, αφού διαθέτει εγγραφή σχετική με το δίκτυο `192.0.2.0/30`.

5.40

Στο PC2 εκτελούμε:

```
dhclient em0
ifconfig em0 | grep inet
netstat -rn | grep default
cat /etc/resolv.conf | grep nameserver
```

Έχουμε:

```
IP address: 192.168.1.3
Default gateway: 192.168.1.1
DNS server address: 192.168.1.1
```

5.41

Από το γραφικό περιβάλλον:

`Firewall - Rules - LAN`

Πατάμε το ”+” και τροποποιούμε κατάλληλα τα πεδία:

```
Action: Block
Interface: LAN
Protocol: any
Source:
    Type: Single host or alias
    Address: 192.168.1.3
Destination:
    Type: Single host or alias
    Address: 172.22.1.2
```

Πατάμε ”Save” και ”Apply changes”.

5.42

Πρέπει να τοποθετηθεί πριν από τον ήδη υπάρχοντα, αλλιώς δεν θα χρησιμοποιηθεί ποτέ, αφού ο ήδη υπάρχοντας επιτρέπει όλη την κίνηση από το LAN.

5.43

Στο PC2 εκτελούμε `ping 172.22.1.2`. Δεν λαμβάνουμε απάντηση.

5.44

Στο PC2 εκτελούμε `ping 172.22.1.1`. Το ping είναι επιτυχές. Αυτό συμβαίνει γιατί ο τελευταίος κανόνας που ορίσαμε σταματά μόνο την κίνηση από το PC2 προς τον SRV1, και όχι από το PC2 προς το FW1, και έτσι ο έλεγχος περνά στον επόμενο κανόνα που επιτρέπει όλη την κίνηση από το LAN.

Άσκηση 6: Τείχος προστασίας και προχωρημένο NAT

6.1

Στον R1 εκτελούμε `ip route 203.0.118.0/24 192.0.2.1`.

6.2

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Outbound`

Τσεκάρουμε το κουτί ”Enable advanced outbound NAT” και πατάμε ”Save”.

6.3

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Outbound`

Πατάμε το ”+” και τροποποιούμε τα πεδία:

```
Interface: WAN
Source: 192.168.1.2/32
Destination:
    Type: any
Target: 203.0.118.14
```

Πατάμε ”Save” και ”Apply changes”.

6.4

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Outbound`

Πατάμε το ”+” και τροποποιούμε τα πεδία:

```
Interface: WAN
Source: 192.168.1.3
Destination:
    Type: any
Target: 203.0.118.15
```

Πατάμε ”Save” και ”Apply changes”.

6.5

Στον R1 εκτελούμε `tcpdump -i em0`.

6.6

Στο PC1 εκτελούμε `ping -c 1 192.0.2.2`. Το ping επιτυγχάνει, και τα πακέτα φτάνουν με διεύθυνση IP την `203.0.118.14`.

6.7

Στο PC2 εκτελούμε `ping -c 1 192.0.2.2`. Το ping επιτυγχάνει, και τα πακέτα φτάνουν με διεύθυνση IP την `203.0.118.15`.

6.8

Από νέο παράθυρο εντολών στον R1 εκτελούμε `ping 203.0.118.14`. Το ping όντως αποτυγχάνει. Αυτό συμβαίνει διότι δεν υπάρχει κανόνας στο firewall που να επιτρέπει την ICMP κίνηση από το WAN σε άλλη διεύθυνση εκτός από την WAN Address, δηλαδή την `192.0.2.1`.

6.9

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Server NAT`

Πατάμε το "+" και τροποποιούμε το πεδίο:

`External IP address: 203.0.118.18`

Πατάμε "Save" και "Apply changes".

6.10

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Inbound`

Πατάμε το "+" και τροποποιούμε τα πεδία (το `[v]` παρακάτω σημαίνει ότι το αντίστοιχο κουτί είναι τσεκαρισμένο):

```
Interface: WAN
External address: 203.0.118.18
Protocol: TCP
External port range:
    from: SSH
    to: SSH
NAT IP: 172.22.1.2
Local port: SSH
[v] Auto-add a firewall rule to permit traffic through this NAT rule
```

Πατάμε "Save" και "Apply changes".

6.11

Από το γραφικό περιβάλλον του FW1:

`Firewall - Rules - WAN`

Προστίθεται ο κανόνας:

Action	Proto	Source	Port	Destination	Port	Description
Allow	TCP	*	*	172.22.1.2	22 (SSH)	NAT

Ο παραπάνω κανόνας προστέθηκε επειδή τσεκάραμε το κουτί "Auto-add a firewall rule to permit traffic through this NAT rule" καθώς ορίζαμε τον παραπάνω κανόνα NAT.

6.12

Στον R1 εκτελούμε `ssh lab@203.0.118.18`. Συνδεόμαστε στον SRV1, όπως μπορούμε να δούμε και από την έξοδο της εντολής `hostname`.

6.13

Στο R1 εκτελούμε `do ping 203.0.118.18`. Δεν λαμβάνουμε απάντηση. Αυτό συμβαίνει διότι από τη μία δεν υπάρχει κάποιος NAT κανόνας για την κίνηση ICMP που παράγεται από το ping, και από την άλλη δεν υπάρχει κάποιος κανόνας που να επιτρέπει την ICMP κίνηση προς τον προορισμό `203.0.118.18`, οπότε τα πακέτα απορρίπτονται από το firewall.

6.14

Στο PC2 εκτελούμε:

```
ssh lab@203.0.118.18
```

Μπορούμε να συνδεθούμε κανονικά. Εκτελώντας καταγραφές στις διεπαφές:

```
PC2@em0, R1@em0, SRV1@em0
```

παρακολουθούμε την πορεία ενός συγκεκριμένου TCP τεμαχίου (επιλέγουμε ένα που έχει συγκεκριμένη τιμή `seq` και το παρατηρούμε). Διαπιστώνουμε ότι ακολουθείται η πορεία:

- PC2 → SRV1:
  - PC2 → FW1 → R1 → FW1 → SRV1
- SRV1 → PC2:
  - SRV1 → FW1 → R1 → FW1 → PC2

Αναλύουμε τα βήματα της διαδρομής, καταγράφοντας κάθε φορά τις διευθύνσεις πηγής και προορισμού του πακέτου και σχολιάζοντας τι γίνεται σε κάθε βήμα:

- PC2 → SRV1:
  - PC2 → FW1: `192.168.1.3` → `203.0.118.18`
    Το πακέτο προωθήθηκε με βάση την προκαθορισμένη διαδρομή του PC2 μέσω του FW1.

    Στο FW1 γίνεται μετάφραση NAT στη διεύθυνση πηγής: `192.168.1.3` → `203.0.118.15`, γιατί το πακέτο πρόκειται να εξέλθει στο WAN.

    Δεν γίνεται μετάφραση NAT στη διεύθυνση προορισμού, γιατί το πακέτο δεν προέρχεται από το WAN.

  - FW1 → R1: `203.0.118.15` → `203.0.118.18`
    Το πακέτο προωθήθηκε με βάση την προκαθορισμένη διαδρομή του FW1 μέσω του R1.

  - R1 → FW1: `203.0.118.15` → `203.0.118.18`
    Το πακέτο προωθήθηκε με βάση την εγγραφή για το δίκτυο `203.0.118.0/24` στον πίνακα δρομολόγησης του R1 μέσω του FW1.

Στο FW1 γίνεται μετάφραση NAT στη διεύθυνση προορισμού: `203.0.118.18` → `172.22.1.2`, γιατί το πακέτο προέρχεται από το WAN.

Δεν γίνεται μετάφραση NAT στη διεύθυνση πηγής, γιατί το πακέτο δεν πρόκειται να εξέλθει στο WAN.

- FW1 → SRV1: `203.0.118.15` → `172.22.1.2`
  Το πακέτο προωθείται τελικά στον προορισμό με επιτυχία.

6.15

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Outbound`

Επιλέγουμε τον κανόνα για το PC1 (Source `192.168.1.2`) και πατάμε "Delete selected mappings". Ύστερα στο PC1 εκτελούμε `ping 192.0.2.2`. Δεν λαμβάνουμε απάντηση. Με tcpdump στον R1 βλέπουμε ότι τα ICMP request έχουν διεύθυνση πηγής την `192.168.1.2` (δεν έχουν υποστεί δηλαδή μετάφραση, επειδή διαγράψαμε τον σχετικό κανόνα Outbound NAT), για την οποία δεν υπάρχει σχετική εγγραφή στον πίνακα δρομολόγησης του R1, επομένως ο R1 δεν μπορεί να απαντήσει με ICMP reply.

6.16

Από το γραφικό περιβάλλον του FW1:

`Firewall - NAT - Outbound`

Αποεπιλέγουμε το κουτί "Enable advanced outbound NAT", πατάμε "Save" και εκτελούμε ξανά στο PC1 `ping 192.0.2.2`. Αυτή τη φορά το ping είναι επιτυχές.

6.17

Εκτελούμε:

```
### R1 ###

ssh lab@203.0.118.18

### PC2 ###

ssh lab@203.0.118.18
```

Μπορούμε να συνδεθούμε κανονικά από τον R1, αλλά όχι από το PC2.

6.18

Στον SRV1 εκτελούμε `tcpdump -i em0` και στο PC2 εκτελούμε `ssh lab@203.0.118.18`. Στην καταγραφή φαίνεται ότι γίνονται συνεχώς προσπάθειες για σύναψη τριμερούς TCP χειραψίας, οι οποίες αποτυγχάνουν. Συγκεκριμένα, ενώ αποστέλλονται τα δύο πρώτα τεμάχια (πρώτο τεμάχιο -σημαία SYN- από τη διεύθυνση `192.0.2.1`, δεύτερο τεμάχιο -σημαίες SYN, ACK- από τη διεύθυνση `172.22.1.2`), αμέσως μετά αποστέλλεται από τη διεύθυνση `192.0.2.1` πακέτο για τερματισμό της σύνδεσης -σημαία RST-. Αυτό επαναλαμβάνεται μέχρι να γίνει timeout από τη μεριά του PC2.

6.19

Για την παραπάνω συμπεριφορά δεν ευθύνονται οι δύο κανόνες που ορίσαμε παραπάνω, πρώτον επειδή ο PC2 προσπαθεί να επικοινωνήσει με τον SRV1 μέσω της διεύθυνσης `203.0.118.18` και όχι μέσω της `172.22.1.2`, και δεύτερον επειδή ο SRV1 χρησιμοποιεί τη διεύθυνση `192.0.2.1` και όχι το LAN net `192.168.1.0/24` για να αποστείλει τα τεμάχια TCP.

Λαμβάνοντας υπόψιν τη σημείωση στην καρτέλα για το Inbound NAT ("It is not possible to access NATed services using the WAN IP address from within LAN"), βλέπουμε πως το πρόβλημα έγκειται στο γεγονός ότι προσπαθούμε να χρησιμοποιήσουμε τη διεύθυνση `203.0.118.18` (NATed service) μέσα από το LAN, χρησιμοποιώντας την WAN IP διεύθυνση `192.0.2.1`. Αυτός είναι και ο λόγος που τερματίζεται συνεχώς η σύνδεση στο ερώτημα 6.18.

Άσκηση 7: IPSec site-to-site VPN

7.1

Από το VirtualBox (το `[ ]` σημαίνει ότι αποεπιλέγουμε το αντίστοιχο κουτί):

`FW1 - Network - Adapter 3 - Advanced - [ ] Cable connected`

7.2

Πηγαίνουμε στη διεύθυνση `http://192.168.56.2` και από το γραφικό περιβάλλον (του FW2):

`Interfaces - MNG - Primary configuration - IP configuration - IP address`

Αλλάζουμε το πεδίο σε `192.168.56.3/24` και πατάμε "Save".

7.3

Από το VirtualBox (το `[v]` σημαίνει ότι τσεκάρουμε το αντίστοιχο κουτί):

`FW1 - Network - Adapter 3 - Advanced - [v] Cable connected`

7.4

Πηγαίνουμε στις διευθύνσεις `http://192.168.56.2` και `http://192.168.56.3`. Και οι δύο σελίδες φορτώνουν κανονικά, και εμφανίζονται τα hostnames `fw1.lab.ntua.gr` και `fw.lab.ntua.gr`, οπότε έχουμε συνδεθεί ταυτόχρονα και στα δύο τείχη προστασίας.

7.5

Από το γραφικό περιβάλλον του FW2:

`System - General setup - Hostname`

Αλλάζουμε την τιμή του πεδίου σε `fw2` και πατάμε "Save".

7.6

Από το γραφικό περιβάλλον του FW2:

`Interface - WAN - Static IP configuration`

Αλλάζουμε τα πεδία ”IP address” και ”Gateway” σε `192.0.2.5/30` και `192.0.2.6` αντίστοιχα, επιλέγουμε το κουτί ”Block private networks” και πατάμε ”Save”.

7.7

Από το γραφικό περιβάλλον του FW2:

`Interfaces - LAN - Primary configuration`

Αλλάζουμε την τιμή του πεδίου ”IP address” σε `192.168.2.1/24` και πατάμε ”Save”.

7.8

Από το γραφικό περιβάλλον του FW2:

`Diagnostics - Reboot system - Yes`

7.9

Από το γραφικό περιβάλλον του FW2:

`Firewall - Rules - LAN`

Πατάμε το ”+” και τροποποιούμε κατάλληλα τα πεδία:

```
Action: Pass
Interface: LAN
Protocol: any
Source:
    Type: LAN subnet
Destination:
    Type: any
```

Πατάμε ”Save” και ”Apply changes”.

7.10

Από το γραφικό περιβάλλον του FW2:

`Firewall - Rules - WAN`

Πατάμε το ”+” και τροποποιούμε κατάλληλα τα πεδία:

```
Action: Pass
Interface: WAN
Protocol: ICMP
ICMP type: any
Source:
    Type: any
Destination:
    Type: WAN address
```

Πατάμε ”Save” και ”Apply changes”.

7.11

Έχουμε ήδη μετακινήσει το PC2 στο LAN2. Ορίζουμε διεύθυνση και προεπιλεγμένη πύλη στο PC2 με τις εντολές:

```
ifconfig em0 192.168.2.2/24
route add default 192.168.2.1
```

7.12

Στο PC1 εκτελούμε `ping 192.0.2.5`. Το ping είναι επιτυχές.

7.13

Στο PC2 εκτελούμε `ping 192.0.2.1`. Το ping είναι επιτυχές.

7.14

Εκτελούμε:

```
### PC1 ###

ping 192.168.2.2

### PC2 ###

ping 192.168.1.2
```

Και τα δύο ping αποτυγχάνουν με μήνυμα λάθους ”Destination Host Unreachable”, το οποίο έρχεται από τη διεύθυνση `192.0.2.2`, δηλαδή τον R1. Αυτό συμβαίνει διότι ο R1 δεν διαθέτει εγγραφή στον πίνακα δρομολόγησής του (`do show ip route`) σχετική με τις διευθύνσεις `192.168.1.2` και `192.168.2.2`, οπότε δεν μπορεί να προωθήσει τα πακέτα στον προορισμό.

7.15

Από το γραφικό περιβάλλον του FW1:

```
VPN - IPsec - Tunnels - [v] Enable IPsec - Save
```

Ύστερα πατάμε το ”+” και τροποποιούμε τα πεδία:

```
Local subnet:
    Type: LAN subnet
Remote subnet: 192.168.2.0/24
Remote gateway: 192.0.2.5
Pre-Shared Key: nick
```

Πατάμε ”Save” και ”Apply changes”.

7.16

Από το γραφικό περιβάλλον του FW1:

`Firewall - Rules - IPsec VPN`

Εμφανίζεται ο κανόνας:

Action	Proto	Source	Port	Destination	Port	Description
Allow	*	*	*	*	*	Default IPsec VPN

7.17

Από το γραφικό περιβάλλον του FW1:

`Diagnostics - IPsec - SAD`

Δεν έχουν ορισθεί σχέσεις.

7.18

Από το γραφικό περιβάλλον του FW1:

`Diagnostics - IPsec - SPD`

Έχουν ορισθεί 2 πολιτικές:

Source	Destination	Direction	Protocol	Tunnel endpoints
192.168.2.0/24	192.168.1.0/24	-->	ESP	192.0.2.5-192.0.2.1
192.168.1.0/24	192.168.2.0/24	<--	ESP	192.0.2.1-192.0.2.5

7.19

Από το γραφικό περιβάλλον του FW2:

`VPN - IPsec - Tunnels - [v] Enable IPsec - Save`

Ύστερα πατάμε το ”+” και τροποποιούμε τα πεδία:

```
  Local subnet:
      Type: LAN subnet
  Remote subnet: 192.168.1.0/24
  Remote gateway: 192.0.2.1
  Pre-Shared Key: nick
```

Πατάμε ”Save” και ”Apply changes”.

7.20

Από το γραφικό περιβάλλον του FW2:

`Diagnostics - IPsec - SAD`

Δεν έχουν ορισθεί σχέσεις.

7.21

Από το γραφικό περιβάλλον του FW2:

`Diagnostics - IPsec - SPD`

Έχουν ορισθεί 2 πολιτικές:

Source	Destination	Direction	Protocol	Tunnel endpoints
192.168.1.0/24	192.168.2.0/24	-->	ESP	192.0.2.1-192.0.2.5
192.168.2.0/24	192.168.1.0/24	<--	ESP	192.0.2.5-192.0.2.1

7.22

Στο PC1 εκτελούμε `ping 192.168.2.2`. Το ping είναι επιτυχές.

7.23

Στο PC2 εκτελούμε `ping 192.168.1.2`. Το ping είναι επιτυχές.

7.24

Από το γραφικό περιβάλλον του FW1:

`Diagnostics - IPsec - SAD`

Πλέον έχουν ορισθεί 2 σχέσεις:

Source	Destination	Protocol	SPI	Enc. alg.	Auth. alg.
192.0.2.1	192.0.2.5	ESP	08911c49	3des-cbc	hmac-sha1
192.0.2.5	192.0.2.1	ESP	0a631651	3des-cbc	hmac-sha1

7.25

Από το γραφικό περιβάλλον του FW2:

`Diagnostics - IPsec - SAD`

Πλέον έχουν ορισθεί 2 σχέσεις:

Source	Destination	Protocol	SPI	Enc. alg.	Auth. alg.
192.0.2.5	192.0.2.1	ESP	0a631651	3des-cbc	hmac-sha1
192.0.2.1	192.0.2.5	ESP	08911c49	3des-cbc	hmac-sha1

7.26

Στον R1 εκτελούμε `tcpdump -vi em0`.

7.27

Εκτελούμε:

```
### PC1 ###

ping 192.168.2.2

### PC2 ###

ping 192.168.1.2
```

Δεν παρατηρούμε πακέτα ICMP.

7.28

Εμφανίζονται πακέτα ESP, κάποια με πηγή `192.0.2.1` και προορισμό `192.0.2.5`, και τα υπόλοιπα με πηγή `192.0.2.5` και προορισμό `192.0.2.1`.

7.29

Όχι, δεν υπάρχει κάπου η πληροφορία για τις διευθύνσεις IP των PC1 και PC2.

7.30

Στο PC2 εκτελούμε `ssh lab@203.0.118.18`. Μπορούμε να συνδεθούμε κανονικά. Ο λόγος είναι ότι παρόλο που ακόμα προσπαθούμε να χρησιμοποιήσουμε NATed υπηρεσίες μέσα από ένα LAN, πλέον ως διεύθυνση πηγής δεν χρησιμοποιείται η WAN IP του FW1 (`192.0.2.1`), αλλά η WAN IP του FW2 `192.0.2.5`, οπότε μπορούν να χρησιμοποιηθούν οι NATed services.

7.31

Παρατηρούμε πακέτα TCP, κάποια με πηγή `192.0.2.5` και προορισμό `203.0.118.18`, και τα υπόλοιπα με πηγή `203.0.118.18` και προορισμό `192.0.2.5`.

7.32

Ναι, είναι κρυπτογραφημένα με το IPsec.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 11
Το πρωτόκολλο IPv6

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 24/05/2022

Άσκηση 1: Εισαγωγή στο IPv6

Προετοιμασία στο σπίτι

1.

Εκτελούμε `service frr stop`.

2.

Εκτελούμε `touch /usr/local/etc/frr/ripngd.conf`.

3.

Εκτελούμε `touch /usr/local/etc/frr/ospf6d.conf`.

4.

Εκτελούμε `chown frr:frr /usr/local/etc/frr/{ripngd.conf,ospf6d.conf}`

5.

Αλλάζουμε τη ζητούμενη γραμμή σε:

```
frr_daemons="zebra staticd ripd ripngd ospfd ospf6d bgpd"
```

6.

Εκτελούμε:

```
reboot
service sshd status
service frr status
```

Τρέχουν κανονικά.

7.

Εκτελούμε `service frr start`.

8.

Εκτελούμε:

```
poweroff
File - Export Appliance...
```

και δημιουργούμε ένα αρχείο `frr.ova`.

9.

Αποθηκεύουμε το `frr.ova` για μελλοντική χρήση.

1.1

Εκτελούμε:

```
### PC1 ###

sysrc ifconfig_em0_ipv6="inet6 accept_rtadv"

### PC2 ###

sysrc ifconfig_em0_ipv6="inet6 accept_rtadv"
```

1.2

Εκτελούμε:

```
### PC1 ###

service netif restart

### PC2 ###

service netif restart
```

1.3

Στο PC1 εκτελούμε `ifconfig em0`. Έχει αποδοθεί η διεύθυνση `fe80::a00:27ff:fe96:db`.

1.4

Στο PC2 εκτελούμε `ifconfig em0`. Έχει αποδοθεί η διεύθυνση `fe80::a00:27ff:fe25:d99f`.

1.5

Είναι είδους link-local (`fe80::/10`). Προκύπτει ως εξής:

```
### PC1 ###

08 00 27 96 00 db                                     (MAC address)

08 00 27 ff fe 96 00 db                               (ff fe in the middle)

0a 00 27 ff fe 96 00 db                               (Flip 7th bit of 1st byte)

fe 80 00 00 00 00 00 00 0a 00 27 ff fe 96 00 db       (Insert fe80::/10 prefix)

fe80:0000:0000:0000:0a00:27ff:fe96:00db               (IP address with leading 0s)

fe80::a00:27ff:fe96:db                                (Final IP address)

### PC2 ###

08 00 27 25 d9 9f                                     (MAC address)

08 00 27 ff fe 25 d9 9f                               (ff fe in the middle)

0a 00 27 ff fe 25 d9 9f                               (Flip 7th bit of 1st byte)

fe 80 00 00 00 00 00 00 0a 00 27 ff fe 25 d9 9f       (Insert fe80::/10 prefix)

fe80:0000:0000:0000:0a00:27ff:fe25:d99f               (IP address with leading 0s)

fe80::a00:27ff:fe25:d99f                              (Final IP address)
```

1.6

Στο PC1 εκτελούμε `netstat -rn6`. Εμφανίζονται 9 εγγραφές:

```
Destination                       Gateway       Flags       Netif
::/96                             ::1           UGRS        lo0
::1                               link#2        UH          lo0
::ffff:0.0.0.0/96                 ::1           UGRS        lo0
fe80::/10                         ::1           UGRS        lo0
fe80::%em0/64                     link#1        U           em0
fe80::a00:27ff:fe96:db%em0        link#1        UHS         lo0
fe80::%lo0/64                     link#2        U           lo0
fe80::1%lo0                       link#2        UHS         lo0
ff02::/16                         ::1           UGRS        lo0
```

1.7

Μία εγγραφή αφορά την em0:

```
Destination          Gateway       Flags       Netif
fe80::%em0/64        link#1        U           em0
```

1.8

Περιέχει τις εγγραφές:

```
Destination                       Gateway       Flags       Netif
fe80::%em0/64                     link#1        U           em0
fe80::a00:27ff:fe96:db%em0        link#1        UHS         lo0
fe80::%lo0/64                     link#2        U           lo0
fe80::1%lo0                       link#2        UHS         lo0
```

1.9

Στο PC1 εκτελούμε `ping6 ::1`. Απαντά το ίδιο το PC1.

1.10

Στο PC1 εκτελούμε `ping6 fe80::a00:27ff:fe96:db`. Εμφανίζεται μήνυμα λάθους "UDP connect: Network is unreachable". Πρέπει να προσθέσουμε τον δείκτη ζώνης `%em0` στο τέλος της διεύθυνσης ως εξής:

```
ping6 fe80::a00:27ff:fe96:db%em0
```

Τώρα το ping είναι επιτυχές.

1.11

Στο PC1 εκτελούμε `ping6 fe80::a00:27ff:fe25:d99f`. Αποτυγχάνει όπως και πριν με μήνυμα λάθους "UDP connect: Network is unreachable". Προσθέτουμε τον δείκτη ζώνης `%em0`:

```
ping6 fe80::a00:27ff:fe25:d99f%em0
```

Το ping είναι επιτυχές.

1.12

Στο PC1 εκτελούμε `ping6 -I em0 ff01::1`. Απαντά το PC1, αφού η διεύθυνση `ff01::1` παριστάνει όλους τους κόμβους στη διεπαφή.

1.13

Στο PC1 εκτελούμε `ping6 -I em0 ff02::1`. Απαντούν τα PC1 και PC2, αφού η διεύθυνση `ff02::1` παριστάνει όλους τους κόμβους στην τοπική ζεύξη.

1.14

Στο PC1 εκτελούμε `ifconfig em0 inet6 fd00:1::2/64`.

1.15

Στο PC2 εκτελούμε `ifconfig em0 inet6 fd00:1::3/64`.

1.16

Είναι μοναδικές τοπικές διευθύνσεις (`fd00::/8`), οι ανάλογες των διευθύνσεων ιδιωτικής χρήσης `10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16` στο IPv4.

1.17

Στα PC1 και PC2 εκτελούμε `ifconfig em0`. Υπάρχουν 2 διευθύνσεις.

1.18

Στο PC1 εκτελούμε `netstat -rn6`. Προστέθηκαν 2 νέες εγγραφές (επισημαίνονται με `+++` στην αρχή της γραμμής):

```
    Destination                    Gateway      Flags      Netif
    ::/96                          ::1          UGRS       lo0
    ::1                            link#2       UH         lo0
    ::ffff:0.0.0.0/96              ::1          UGRS       lo0
+++ fd00:1::/64                    link#1       U          em0
+++ fd00:1::2                      link#1       UHS        lo0
    fe80::/10                      ::1          UGRS       lo0
    fe80::%em0/64                  link#1       U          em0
    fe80::a00:27ff:fe96:db%em0     link#1       UHS        lo0
    fe80::%lo0/64                  link#2       U          lo0
    fe80:1%lo0                     link#2       UHS        lo0
    ff02::/16                      ::1          UGRS       lo0
```

1.19

Πρέπει και στα δύο μηχανήματα να προσθέσουμε (μέσω `vi /etc/hosts`) τις γραμμές:

```
fd00:1::2 PC1
fd00:1::3 PC2
```

1.20

Στο PC1 εκτελούμε `ping6 PC2`. Το ping είναι επιτυχές.

1.21

Στο PC1 εκτελούμε `arp -a`. Δεν παρατηρούμε καμία εγγραφή.

1.22

Στο PC1 εκτελούμε `man ndp`.

1.23

Είναι `npd -a`.

1.24

Βλέπουμε 4 εγγραφές:

```
Neighbor                        Linklayer Addr     Netif  Expire     S  Flags
PC1                             08:00:27:96:00:db  em0    permanent  R
PC2                             08:00:27:25:d9:9f  em0    23h58m7s   S
fe80::a00:27ff:fe96:db%em0      08:00:27:96:00:db  em0    permanent  R
fe80::a00:27ff:fe25:d99f%em0    08:00:27:25:d9:9f  em0    22h7m44s   S
```

1.25

Στο PC1 εκτελούμε `ndp -p`. Έχουμε:

```
fd00:1::/64 if=em0
flags=LO vltime=infinity, pltime=infinity, expire=Never, ref=1
  No advertising router

fe80::%em0/64 if=em0
flags=LAO vltime=infinity, pltime=infinity, expire=Never, ref=0
  No advertising router

fe80::%lo0/64 if=lo0
flags=LAO vltime=infinity, pltime=infinity, expire=Never, ref=0
  No advertising router
```

Οι εγγραφές έχουν άπειρη διάρκεια ζωής. Από το SLAAC μπορούν να χρησιμοποιηθούν η 2η και η 3η εγγραφή, που έχουν τη σημαία ”A” ενεργοποιημένη.

1.26

Σε νέο παράθυρο στο PC2 εκτελούμε `tcpdump -vn`.

1.27

Στα PC1 και PC2 εκτελούμε `ndp -c`.

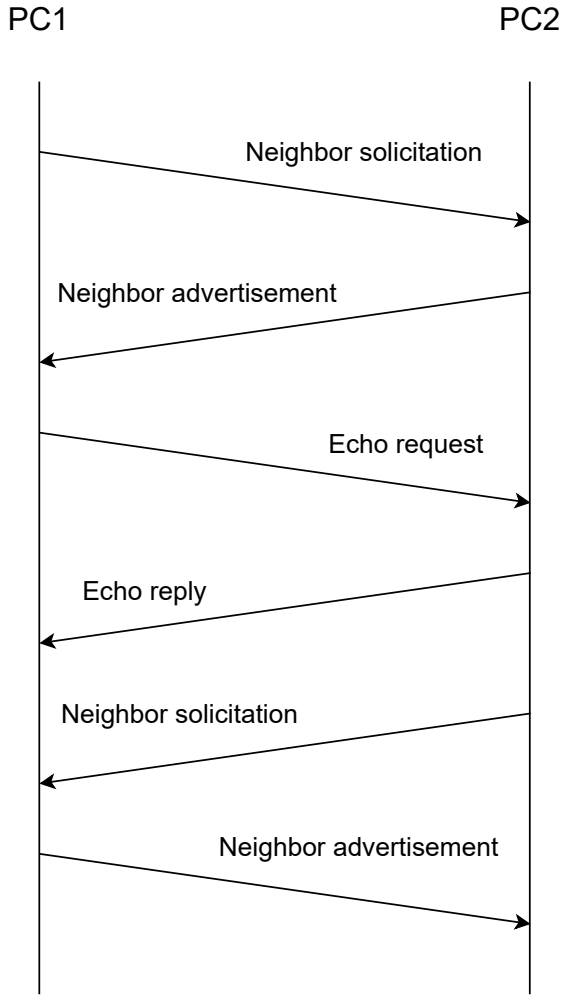
1.28

Στο PC1 εκτελούμε `ping6 -c 1 PC2` και σταματάμε την καταγραφή. Βλέπουμε 6 πακέτα IPv6.

1.29

Μεταφέρουν μηνύματα του ICMPv6. Η τιμή του Next header είναι 58.

1.30



1.31

Η διεύθυνση προορισμού του πρώτου πακέτου NS (`ff02::1:ff00:3`) είναι multicast Solicited Node και προκύπτει από τα τελευταία 24 bit της διεύθυνσης unicast του PC2, αν σε αυτά προσθέσουμε το πρόθεμα `ff02:0:0:0:0:1:ff00:0/104`:

```
    ff02:0:0:0:0:1:ff00:0000    (Special prefix)
+                     00:0003    (Last 24 bits of PC2 unicast address)
----------------------------
    ff02:0:0:0:0:1:ff00:0003    (Multicast Solicited Node address with leading 0s)

               ff02::1:ff00:3    (Final multicast Solicited Node address)
```

1.32

Η διεύθυνση προορισμού του δεύτερου πακέτου NS (`fd00::2`) είναι μοναδική τοπική διεύθυνση, και έχει προκύψει από ανάθεση (που έγινε σε προηγούμενο ερώτημα).

1.33

Στο PC2 εκτελούμε `ndp -a`. Η κατάσταση της εγγραφής για το PC1 είναι ”S” (Stale). Η διάρκεια ζωής της σχετικής εγγραφής είναι `22h1m44s`.

1.34

Στο PC1 εκτελούμε `ping6 PC2` και ύστερα με διαδοχικά `ndp -a` στο PC2 παρατηρούμε συνεχή εναλλαγή μεταξύ των καταστάσεων ”R” και ”S” (Reachable και Stale).

1.35

Φαίνεται να είναι ≈43 δευτερόλεπτα, και μόλις λήξει, η αντίστοιχη εγγραφή μπαίνει σε κατάσταση Stale.

1.36

Είναι 24 ώρες.

1.37

Σταματάμε το ping στο PC1 και ύστερα με διαδοχικά `ndp -a` στο PC2 παρατηρούμε αλλαγή από την κατάσταση ”R” (Reachable) στην κατάσταση ”S” (Stale), και αυτή τη φορά η κατάσταση παραμένει ”S” (Stale).

1.38

Στο PC1 εκτελούμε `ping6 PC2`. Στο PC2 εκτελούμε `tcpdump -n`. Παρατηρούμε επιπλέον πακέτα ICMPv6 neighbor solicitation (NS), με σκοπό να επιβεβαιωθεί η προσβασιμότητα του γείτονα, και ICMPv6 neighbor advertisement (NA), ως απάντηση στα πακέτα NS. Παράγονται περίπου κάθε 23 δευτερόλεπτα.

Άσκηση 2: SLAAC και Στατική δρομολόγηση IPv6

2.1

Στους R1 και R2 εκτελούμε:

```
sysrc ipv6_gateway_enable="YES"
service routing restart
```

2.2

Στο PC2 εκτελούμε:

```
ifconfig em0 inet6 fd00:1::3/64 delete
ifconfig em0 inet6 fd00:2::2/64
```

2.3

Στον R1 εκτελούμε:

```
vtysh
configure terminal

interface em0

ipv6 address fd00:1::1/64
```

Σημείωση: Μπορούμε να δώσουμε και την εντολή `"ip address"` αντί για `"ipv6 address"`. Και στις δύο περιπτώσεις η εντολή θα καταχωρηθεί ως `"ipv6 address"` στο running configuration. Εμείς προτιμούμε την εντολή `"ipv6 address"` για να είμαστε απολύτως ακριβείς.

2.4

Στον R1 εκτελούμε:

```
interface em1
ipv6 address fd00:3::1/126
```

2.5

Στον R2 εκτελούμε:

```
vtysh
configure terminal

interface em1
ipv6 address fd00:2::1/64
```

2.6

Στον R2 εκτελούμε:

```
interface em0
ipv6 address fd00:3::2/126
```

2.7

Στο PC1 εκτελούμε `route -6 add default fd00:1::1`.

2.8

Στο PC2 εκτελούμε `route -6 add default fd00:2::1`.

2.9

Στον R1 εκτελούμε `tcpdump -i em0`.

2.10

Στο PC1 εκτελούμε:

```
ndp -c
ping6 -c 1 fd00:2::2
```

Δε λαμβάνουμε απάντηση. Αυτό συμβαίνει διότι ο R1 δεν διαθέτει εγγραφή στον πίνακα δρομολόγησης σχετική με τη διεύθυνση `fd00:2::2` και έτσι στέλνει μήνυμα λάθους ICMPv6 destination unreachable.

2.11

Παράγονται (καταγράφουμε και την διεύθυνση πηγής για πληρότητα):

ICMPv6 Message	IPv6 source address	IPv6 destination address
Neighbor solicitation	fd00:1::2	ff02::1:ff00:1
Neighbor advertisement	fd00:1::1	fd00:1::2
Echo request	fd00:1::2	fd00:2::2
Destination unreachable	fd00:1::1	fd00:1::2
Neighbor solicitation	fd00:1::1	fd00:1::2
Neighbor advertisement	fd00:1::2	fd00:1::1

2.12

Στον R1 εκτελούμε:

```
ipv6 route fd00:2::/64 fd00:3::2
```

Σημείωση: Όπως και πριν, επιλέγουμε την εντολή `"ipv6"`, αλλά και η `"ip"` θα είχε τα ίδια αποτελέσματα (στο running configuration η εντολή καταχωρείται έτσι κι αλλιώς ως `"ipv6"`).

2.13

Στο PC1 εκτελούμε `ping6 -c 1 fd00:2::2`. Δεν λαμβάνουμε απάντηση, διότι ο R2 δεν έχει εγγραφή στον πίνακα δρομολόγησης σχετική με τη διεύθυνση `fd00:1::2` προκειμένου να στείλει το ICMPv6 echo reply.

2.14

Στον R2 εκτελούμε:

```
ipv6 route fd00:1::/64 fd00:3::1
```

2.15

Στο PC1 εκτελούμε `ping6 -c 1 fd00:2::2`. Το ping είναι επιτυχές.

2.16

Στον R1 εκτελούμε:

```
interface em0
no ipv6 nd suppress-ra
```

2.17

Στον R1 εκτελούμε:

```
interface em0
ipv6 nd prefix fd00:1::/64
```

2.18

Στον R2 εκτελούμε:

```
interface em1
no ipv6 nd suppress-ra
```

2.19

Στον R2 εκτελούμε:

```
interface em1
ipv6 nd prefix fd00:2::/64
```

2.20

Στο PC1 εκτελούμε:

```
route -6 delete default
```

2.21

Στον R1 εκτελούμε `tcpdump -eni em0 icmp6`.

2.22

Στο PC1 εκτελούμε `service netif restart`.

2.23

Ανταλλάσονται τα μηνύματα:

Type	MAC source	MAC destination	IPv6 source	IPv6 destination
NS	08:00:27:96:00:DB	33:33:FF:96:00:DB	::	FF02::1:FF96:DB
RS	08:00:27:96:00:DB	33:33:00:00:00:02	FE80::A00:27FF:FE96:DB	FF02::2
RA	08:00:27:03:C1:F6	33:33:00:00:00:01	FE80::A00:27FF:FE03:C1F6	FF02::1
NS	08:00:27:96:00:DB	33:33:FF:96:00:DB	::	FF02::1:FF96:DB

2.24

Παράγει 2 μηνύματα NS. Το πρώτο από αυτά χρειάζεται για την αυτόματη απόδοση διεύθυνσης και το δεύτερο για την ανίχνευση ταυτόσημων διευθύνσεων.

2.25

Χρησιμοποιεί την `::` (Μη ορισμένη/Unspecified) και στα δύο, στο πρώτο NS επειδή ο PC1 δεν έχει αποκτήσει ακόμα διεύθυνση, και στο δεύτερο NS επειδή κατά τη διαδικασία DAD χρησιμοποιείται πάντα η ακαθόριστη διεύθυνση `::`.

2.26

Χρησιμοποιεί την `fe80::a00:27ff:fe96:db`.

2.27

Είναι:

```
Type      IPv6 destination address
NS        ff02::1:ff96:db
RS        ff02::2
RA        ff02::1
NS        ff02::1:ff96:db
```

- Το NS έχει αυτή τη διεύθυνση επειδή αναφέρεται στην τοπική ζεύξη → στον PC1
- Το RS έχει αυτή τη διεύθυνση επειδή αναφέρεται σε όλους τους δρομολογητές στην τοπική ζεύξη.
- Το RA έχει αυτή τη διεύθυνση επειδή αναφέρεται σε όλους τους κόμβους της τοπικής ζεύξης.
- Τέλος, το δεύτερο NS έχει διεύθυνση ίδια με το παραπάνω NS.

2.28

Είναι:

```
Type       MAC destination address
NS         33:33:ff:96:00:db
RS         33:33:00:00:00:02
RA         33:33:00:00:00:01
NS         33:33:ff:96:00:db
```

Οι παραπάνω διευθύνσεις MAC προορισμού προκύπτουν από την παράθεση του προθέματος `33:33:` να ακολουθείται από τα τελευταία 32 bit της IPv6 διεύθυνσης προορισμού, επομένως:

```
### NS ###
ff 02 00 00 00 00 00 00 00 00 00 01 ff 96 00 db    (IPv6 destination address)
                                    ff 96 00 db    (Last 32 bits)
                                33 33 ff 96 00 db    (Add 33 33 prefix)
                                33:33:ff:96:00:db    (Final MAC address)
```

```
### RS ###
ff 02 00 00 00 00 00 00 00 00 00 00 00 02    (IPv6 destination address)
                                  00 00 02    (Last 32 bits)
                               33 33 00 00 02    (Add 33 33 prefix)
                               33:33:00:00:02    (Final MAC address)


### RA ###
ff 02 00 00 00 00 00 00 00 00 00 00 00 01    (IPv6 destination address)
                                  00 00 01    (Last 32 bits)
                               33 33 00 00 01    (Add 33 33 prefix)
                               33:33:00:00:01    (Final MAC address)


### NS ###
ff 02 00 00 00 00 00 00 00 00 01 ff 96 00 db    (IPv6 destination address)
                                  ff 96 00 db    (Last 32 bits)
                               33 33 ff 96 00 db    (Add 33 33 prefix)
                               33:33:ff:96:00:db    (Final MAC address)
```

2.29

Στο PC1 εκτελούμε `ndp -p`. Έχουμε:

```
fd00:1::/64 if=em0
flags=LAO vltime=2592000, pltime=604800, expire=29d23h51m54s, ref=1
  advertised by
    fe80::a00:27ff:fe03:c1f6%em0 (reachable)

fe80::%em0/64 if=em0
flags=LAO vltime=infinity, pltime=infinity, expire=Never, ref=0
  No advertising router

fe80::%lo0/64 if=lo0
flags=LAO vltime=infinity, pltime=infinity, expire=Never, ref=0
  No advertising router
```

Πλέον έχει μαθευτεί το πρόθεμα `fd00:1::/64` που διαφημίζει ο R1 (`fe80::a00:27ff:fe03:c1f6`).

2.30

Στο PC1 εκτελούμε `ifconfig em0`. Το PC1 έχει λάβει αυτόματα μέσω του SLAAC τη διεύθυνση `fd00:1::a00:27ff:fe96:db`.

2.31

Στο PC1 εκτελούμε `netstat -rn6`. Έχουμε:

```
Destination                 Gateway                         Flags  Netif
::/96                       ::1                             UGRS   lo0
default                     fe80::a00:27ff:fe03:c1f6%em0    UG     em0
::1                         link#2                          UH     lo0
::ffff:0.0.0.0/96           ::1                             UGRS   lo0
fd00:1::/64                 link#1                          U      em0
fd00:1::a00:27ff:fe96:db    link#1                          UHS    lo0
fe80::/10                   ::1                             UGRS   lo0
fe80::%em0/64               link#1                          U      em0
fe80::a00:27ff:fe96:db%em0  link#1                          UHS    lo0
fe80::%lo0/64               link#2                          U      lo0
fe80::1%lo0                 link#2                          UHS    lo0
ff02::/16                   ::1                             UGRS   lo0
```

Υπάρχει προκαθορισμένη διαδρομή (είναι η δεύτερη εγγραφή στον παραπάνω πίνακα). Προέκυψε από τα μηνύματα Router advertisement που έστειλε ο R1.

2.32

Μπορούμε να χρησιμοποιήσουμε τις διευθύνσεις:

```
### Ping PC2 --> PC1 ###

fd00:1::a00:27ff:fe96:db

### Ping R1 --> PC1 ###

fd00:1::a00:27ff:fe96:db
fe80::a00:27ff:fe96:db%em0
```

Άσκηση 3: Δυναμική δρομολόγηση IPv6

3.1

Εκτελούμε:

```
### R1 ###

no ipv6 route fd00:2::/64 fd00:3::2

### R2 ###

no ipv6 route fd00:1::/64 fd00:3::1
```

3.2

Εκτελούμε:

```
### R1 ###

router ripng
network em0
network em1

### R2 ###

router ripng
network em0
network em1
```

3.3

Στον R1 εκτελούμε `do show ipv6 route ripng`. Βλέπουμε μια εγγραφή:

```
R>* fd00:2::/64 [120/2] via fe80::a00:27ff:fed1:fb31, em1
```

3.4

Είναι `fe80::a00:27ff:fed1:fb31` και είναι τοπική στη ζεύξη διεύθυνση (link-local).

3.5

Στο PC1 εκτελούμε `ping6 fd00:2::2`. Το ping είναι επιτυχές.

3.6

Στον R1 εκτελούμε `tcpdump -vni em1 ip6`.

3.7

Παρατηρούμε πακέτα ripng-resp. Η διεύθυνση προορισμού τους είναι η multicast διεύθυνση `ff02::9` (η αντίστοιχη της `224.0.0.9` για το RIPv2).

3.8

Έχει τιμή 255, προκειμένου να μην μπορούν τα ripng-resp πακέτα να διέλθουν από δρομολογητές.

3.9

Χρησιμοποιεί το UDP και τη θύρα 521, ενώ το RIP χρησιμοποιεί πάλι το UDP, αλλά τη θύρα 520.

3.10

Εκτελούμε:

```
### R1 ###

no router ripng

### R2 ###

no router ripng
```

3.11

Εκτελούμε:

```
### R1 ###

do write file

### R2 ###

do write file
```

3.12

Εκτελούμε:

```
### R1 ###

service frr restart

### R2 ###

service frr restart
```

3.13

Εκτελούμε:

```
### R1 ###

router ospf6
router-id 1.1.1.1
ospf router-id 1.1.1.1

### R2 ###

router ospf6
router-id 2.2.2.2
ospf router-id 2.2.2.2
```

3.14

Στον R1 εκτελούμε:

```
router ospf6
interface em0 area 0.0.0.0
interface em1 area 0.0.0.0
```

3.15

Στον R2 εκτελούμε:

```
router ospf6
interface em0 area 0.0.0.0
interface em1 area 0.0.0.0
```

3.16

Στον R2 εκτελούμε `do show ipv6 route ospf`. Βλέπουμε 2 εγγραφές:

```
O>* fd00:1::/64 [110/200] via fe80::a00:27ff:febb:4046, em0
O   fd00:2::/64 [110/100] is directly connected, em1
```

Το κόστος τους προέκυψε ως το άθροισμα του κόστους όλων των ζεύξεων της εκάστοτε διαδρομής. Το κόστος μιας ζεύξης υπολογίζεται ως:

$$\frac{\text{Reference bandwidth}}{\text{Interface bandwidth}} = \frac{100\ \text{Mbps}}{1\ \text{Mbps}} = 100$$

όπου ως Reference bandwidth χρησιμοποιείται η προκαθορισμένη τιμή (εφόσον δεν την αλλάξαμε με κάποιο τρόπο), ενώ το Interface bandwidth μπορεί να βρεθεί από την εκτέλεση της "do show interface" στους R1 και R2. Έτσι, το κόστος προς το δίκτυο `fd00:1::/64` (πρώτη εγγραφή) είναι 200 (διαδρομή R2 $\rightarrow$ R1 $\rightarrow$ PC1, 2 ζεύξεις), ενώ το κόστος προς το δίκτυο `fd00:2::/64` (δεύτερη εγγραφή) είναι 100 (διαδρομή R2 $\rightarrow$ PC2, 1 ζεύξη).

3.17

Είναι η `fe80::a00:27ff:febb:4046` (R1@em1), και είναι διεύθυνση τοπική στη ζεύξη (link-local).

3.18

Στον R2 εκτελούμε `tcpdump -vni em0 ip6`.

3.19

Παρατηρούμε πακέτο OSPFv3 Hello, με προορισμό τη διεύθυνση `ff02::5` (η αντίστοιχη της `224.0.0.5` για το OSPFv2).

3.20

Έχει τιμή 1.

3.21

Χρησιμοποιεί τον αριθμό 89 (OSPF), ίδιο με του OSPFv2.

3.22

Στο PC2 εκτελούμε `ping6 fd00:1::a00:27ff:fe96:db`. Το ping είναι επιτυχές.

3.23

Εκτελούμε:

```
### R1 ###

no router ospf6

### R2 ###

no router ospf6
```

3.24

Εκτελούμε:

```
### R1 ###

service frr restart

### R2 ###

service frr restart
```

3.25

Στον R1 εκτελούμε:

```
router-id 1.1.1.1
router bgp 65010
```

3.26

Στον R1 εκτελούμε:

```
no bgp ebgp-requires-policy
```

3.27

Στον R1 εκτελούμε `no bgp default ipv4-unicast`.

3.28

Στον R1 εκτελούμε `neighbor fd00:3::2 remote-as 65020`.

3.29

Στον R1 εκτελούμε `address-family ipv6`.

3.30

Στον R1 εκτελούμε `network fd00:1::/64`.

3.31

Στον R1 εκτελούμε `neighbor fd00:3::2 activate` και εξερχόμαστε με `exit`.

3.32

Επαναλαμβάνουμε τα προηγούμενα για τον R2:

```
router-id 2.2.2.2
router bgp 65020
no bgp ebgp-requires-policy
no bgp default ipv4-unicast
neighbor fd00:3::1 remote-as 65010
address-family ipv6
network fd00:2::/64
neighbor fd00:3::1 activate
exit
```

3.33

Στον R1 εκτελούμε `do show ipv6 route bgp`. Βλέπουμε 1 εγγραφή:

```
B>* fd00:1::/64 [20/0] via fe80::a00:27ff:febb:4046, em0
```

3.34

Είναι η `fe80::a00:27ff:fed1:fb31` (R2@em0), και είναι διεύθυνση τοπική στη ζεύξη (link-local).

3.35

Στον R1 εκτελούμε `tcpdump -vni em1`.

3.36

Παρατηρούμε μηνύματα BGP KEEPALIVE. Χρησιμοποιείται το TCP και η θύρα 179, όπως και στην περίπτωση του IPv4.

3.37

Έχει τιμή 1.

3.38

Στο PC1 εκτελούμε `ping6 fd00:2::a00:27ff:fe25:d99f`. Το ping είναι επιτυχές.

3.39

Στο PC1 εκτελούμε:

```
reboot

vtysh
configure terminal

interface em0
ipv6 address fd00:1::2/64
```

3.40

Στο PC1 εκτελούμε:

```
router-id 1.1.0.0
router bgp 65010
```

3.41

Στο PC1 εκτελούμε `no bgp default ipv4-unicast`.

3.42

Στο PC1 εκτελούμε `neighbor fd00:1::1 remote-as 65010`.

3.43

Στο PC1 εκτελούμε:

```
address-family ipv6
neighbor fd00:1::1 activate
exit
```

3.44

Στον R1 εκτελούμε:

```
router bgp 65010
neighbor fd00:1::2 remote-as 65010
```

3.45

Στον R1 εκτελούμε:

```
address-family ipv6
neighbor fd00:1::2 activate
neighbor fd00:1::2 next-hop-self
exit
```

3.46

Με την εντολή `do show ip bgp neighbors fd00:1::2`. Στο τέλος της πρώτης γραμμής αναγράφεται ”internal link”, ενώ το BGP state είναι Established.

3.47

Στο PC1 εκτελούμε `do show ipv6 route bgp`. Βλέπουμε 2 εγγραφές:

```
B   fd00:1::/64 [200/0] via fe80::a00:27ff:fe03:c1f6, em0
B>* fd00:2::/64 [200/0] via fd00:1::1, em0
```

3.48

Δεν είναι επιλεγμένη επειδή υπάρχει η προτιμότερη εγγραφή στον πίνακα δρομολόγησης:

```
C>* fd00:1::/64 [0/1] is directly connected, em0
```

3.49

Είναι η `fd00:1::1` (R1@em0), και είναι μοναδική τοπική διεύθυνση.

3.50

Στο PC2 εκτελούμε `ping6 fd00:1::2`. Το ping είναι επιτυχές.

Άσκηση 4: Μηχανισμός μετάβασης 464 XLAT

4.1

Στον R1 εκτελούμε:

```
vtysh
configure terminal

interface em0
ip address 192.168.1.1/24
```

4.2

Στον R2 εκτελούμε:

```
vtysh
configure terminal

interface em1
ip address 192.168.2.1/24
```

4.3

Στο PC1 εκτελούμε:

```
vtysh
configure terminal

interface em0
ip address 192.168.1.2/24

ip route 0.0.0.0/0 192.168.1.1
```

4.4

Στο PC2 εκτελούμε:

```
vtysh
configure terminal

interface em0
ip address 192.168.2.2/24

ip route 0.0.0.0/0 192.168.2.1
```

4.5

Στον R1 εκτελούμε:

```
sysrc firewall_enable="YES"
sysrc firewall_nat64_enable="YES"
sysrc firewall_type="open"
sysrc firewall_logif="YES"
```

4.6

Στον R1 εκτελούμε:

```
service ipfw start
```

4.7

Στον R1 εκτελούμε:

```
ipfw list
```

Εμφανίζονται 12 κανόνες:

```
00100 allow ip from any to any via lo0
00200 deny ip from any to 127.0.0.0/8
00300 deny ip from 127.0.0.0/8 to any
00400 deny ip from any to ::1
00500 deny ip from ::1 to any
00600 allow ipv6-icmp from :: to ff02::/16
00700 allow ipv6-icmp from fe80::/10 to fe80::/10
00800 allow ipv6-icmp from fe80::/10 to ff02::/16
00900 allow ipv6-icmp from any to any icmp6types 1
01000 allow ipv6-icmp from any to any icmp6types 2,135,136
65000 allow ip from any to any
65535 deny ip from any to any
```

4.8

Στο PC1 εκτελούμε `do ping ipv6 fd00:2::a00:27ff:fe25:d99f/64`. Το ping είναι επιτυχές.

4.9

Στον R1 εκτελούμε (όλο μαζί μία εντολή):

```
ipfw nat64clat nat64 create \
clat_prefix fd00:3:1::/96   \
plat_prefix 64:ff9b::/96    \
allow_private log
```

4.10

Στον R1 εκτελούμε:

```
ipfw add 2000 nat64clat nat64 ip4 from any to not me recv em0
```

4.11

Στον R1 εκτελούμε:

```
ipfw add 3000 nat64clat nat64 ip6 from 64:ff9b::/96 to fd00:3:1::/96 recv em1
```

4.12

Στον R1 εκτελούμε:

```
ipv6 route 64:ff9b::/96 fd00:3::2
```

4.13

Στον R2 εκτελούμε:

```
sysrc firewall_enable="YES"
sysrc firewall_nat64_enable="YES"
sysrc firewall_type="open"
sysrc firewall_logif="YES"
service ipfw start
```

4.14

Στον R2 εκτελούμε (όλο μαζί μία εντολή):

```
ipfw nat64lsn nat64 create \
prefix4 2.2.2.0/24         \
prefix6 64:ff9b::/96       \
allow_private log
```

4.15

Στον R2 εκτελούμε:

```
ipfw add 2000 nat64lsn nat64 ip6 from fd00:3:1::/96 to 64:ff9b::/96 recv em0
```

4.16

Στον R2 εκτελούμε:

```
ipfw add 3000 nat64lsn nat64 ip4 from any to 2.2.2.0/24 recv em1
```

4.17

Στον R2 εκτελούμε:

```
ipv6 route fd00:3:1::/96 fd00:3::1
```

4.18

Στον R2 εκτελούμε:

```
ip route 0.0.0.0/0 192.168.2.2
```

4.19

Στο PC1 εκτελούμε:

```
do ping 192.168.1.1
do ping 192.168.2.2
```

Τα ping είναι επιτυχή.

4.20

Στον R1 εκτελούμε:

```
ifconfig ipfwlog0 create
tcpdump -i ipfwlog0
```

4.21

Στον R2 εκτελούμε:

```
ifconfig ipfwlog0 create
tcpdump -i ipfwlog0
```

4.22

Στο PC1 εκτελούμε `ping -c 1 192.168.2.2`. Παρατηρούμε τα πακέτα:

```
### R1 capture ###

Type                    Source                  Destination
ICMP echo request       192.168.1.2             192.168.2.2
ICMPv6 echo request     fd00:3:1::c0a8:102      64:ff9b::c0a8:202
ICMPv6 echo reply       64:ff9b::c0a8:202       fd00:3:1::c0a8:102
ICMP echo reply         192.168.2.2             192.168.1.2

### R2 capture ###

Type                    Source                  Destination
ICMPv6 echo request     fd00:3:1::c0a8:102      64:ff9b::c0a8:202
ICMP echo request       2.2.2.158               192.168.2.2
ICMP echo reply         192.168.2.2             2.2.2.158
ICMPv6 echo reply       64:ff9b::c0a8:202       fd00:3:1::c0a8:102
```

4.23

Στο PC2 εκτελούμε:

```
interface em0
ip address 172.17.17.2/24
ip address 10.0.0.2/24
```

4.24

Στο PC1 εκτελούμε:

```
do ping 172.17.17.2
do ping 10.0.0.2
```

Και τα δύο ping είναι επιτυχή.

4.25

Στον R2 εκτελούμε `ipfw nat64lsn nat64 show states`.

4.26

Εκτελούμε:

```
### PC1 ###

ping -c 1 172.17.17.2
ping -c 1 10.0.0.2

### R2 ###

ipfw nat64lsn nat64 show states
```

Εμφανίζονται οι εγγραφές:

```
fd00:3:1::c0a8:102 2.2.2.65 ICMPv6 <time> 172.17.17.2
fd00:3:1::c0a8:102 2.2.2.65 ICMPv6 <time> 10.0.0.2
```

όπου `<time>` είναι ο χρόνος που πέρασε από την καταχώρηση της εγγραφής. Αυτές οι εγγραφές αφορούν τις μεταφράσεις που σχετίζονται με τα δύο παραπάνω ping. Με διαδοχικές εκτελέσεις της:

```
ipfw nat64lsn nat64 show states
```

διαπιστώνουμε ότι οι εγγραφές διαρκούν 62 δευτερόλεπτα.

Άσκηση 5: Μηχανισμός μετάβασης Teredo

5.1

Στα PC1 και PC2 εκτελούμε:

```
dhclient em0            # Got 10.0.2.15
ping www.google.com     # To test internet access
```

5.2

Στα PC1 και PC2 εκτελούμε:

```
pkg install miredo
```

5.3

Στα PC1 και PC2 εκτελούμε:

```
sysrc miredo_enable="YES"
```

5.4

Στα PC1 και PC2 εκτελούμε:

```
service miredo start
```

5.5

Στο PC1 εκτελούμε `ifconfig`. Εμφανίζεται η νέα διεπαφή:

```
Name: teredo
IPv6 address: 2001:0:c38c:c38c:14d3:5860:b07d:1bdc/128
```

5.6

Στο PC1 εκτελούμε `tcpdump -ni em0`.

5.7

Είναι η `195.140.195.140`.

5.8

Χρησιμοποιείται το UDP, ενώ στον εξυπηρετητή Teredo αντιστοιχεί η θύρα 3544.

5.9

Ξεκινάμε μία καταγραφή με Wireshark στην φυσική κάρτα του υπολογιστή μας (διεπαφή `enp2s0`), με φίλτρο απεικόνισης `teredo`. Παρατηρούμε μηνύματα του πρωτοκόλλου ICMPv6, συγκεκριμένα Router Solicitation/Advertisement.

5.10

Στο PC1 εκτελούμε:

```
ping6 www.ntua.gr
ping6 www.ibm.com
ping6 www.google.com
```

Μπορούμε να κάνουμε ping μόνο στο `www.ibm.com`.

5.11

Στο PC1 εκτελούμε `ping6 www.ibm.com` και το αφήνουμε να τρέχει.

5.12

Παρατηρούμε μηνύματα IPv6 no next header.

5.13

Όχι, δεν παρατηρούμε.

5.14

Παρατηρούμε το IPv4, ενώ η θύρα που αντιστοιχεί στον αναμεταδότη Teredo είναι η 3545.

5.15

Σταματάμε τις καταγραφές και ύστερα στο PC1 εκτελούμε `tcpdump -ni teredo`.

5.16

Ναι, παράγονται ICMPv6 echo request/reply.

5.17

Στο PC1 εκτελούμε:

```
ping6 2001:0:c38c:c38c:2cdb:2a24:b07d:1bdc
```

Δεν λαμβάνουμε απάντηση.

5.18

Ναι, παράγονται ICMPv6 echo request.

5.19

Στο PC1 εκτελούμε `tcpdump -ni em0`. Δεν παράγονται δεδομενογράμματα UDP που να αντιστοιχούν στα ICMPv6 μηνύματα.

5.20

Στο PC1 εκτελούμε `ping6 www.hp.com` και μετά `ping6 www.f5.com`. Επιλέγεται διαφορετικός teredo relay κάθε φορά:

```
ping www.hp.com: 185.218.193.138
ping www.f5.com: 216.66.80.238
```

5.21

Στο PC1 εκτελούμε `ping6 one.one.one.one`. Παρατηρούμε ότι ο teredo relay που επιλέγεται είναι ίδιος με τον teredo server που επιλέχθηκε στο ερώτημα 5.7.

Εργαστήριο Δικτύων Υπολογιστών

Εργαστηριακή Άσκηση 12
Υπηρεσίες στο Διαδίκτυο

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Όνομα PC:** nick-ubuntu
Ομάδα: 1 (Τρίτη 10:45)	**Ημερομηνία Εξέτασης:** Τρίτη 31/05/22

Άσκηση 1: Εγκατάσταση DHCP server

Κατασκευάζουμε ένα εικονικό μηχάνημα με βάση ένα νέο FreeBSD 12.3 με **3** κάρτες δικτύου:

1.

Ορίζουμε τη διεπαφή `em1` σε NAT.

2.

Εκτελούμε `dhclient em1`.

3.

Εκτελούμε `ping www.google.com`. Το ping είναι επιτυχές.

4.

Εκτελούμε `pkg update`.

5.

Εκτελούμε `poweroff` και ύστερα από τη διαδρομή File→Export Appliance... δημιουργούμε ένα αρχείο `new.ova`.

Στη συνέχεια δημιουργούμε ένα νέο μηχάνημα NS1 βασισμένο στο `new.ova`.

1.

Εκτελούμε `pkg install isc-dhcp44-server`.

2.

Κατασκευάζουμε ένα δικό μας `dhcpd.conf` με περιεχόμενα:

```
default-lease-time 60;                              # (e)
max-lease-time 120;                                 # (f)

subnet 192.168.2.0 netmask 255.255.255.240 {        # (a)
  range 192.168.2.5 192.168.2.6;                    # (b)
  option routers 192.168.2.1;                       # (c)
  option broadcast address 192.168.2.15;            # (d)
}
```

3.

Εκτελούμε:

```
sysrc ifconfig_em0="192.168.2.1/28"                 # (a)
sysrc ifconfig_em1="DHCP"                           # (b)
sysrc dhcpd_enable="YES"                            # (c)
sysrc dhcp_ifaces="em0"                             # (d)
sysrc hostname="ns1.ntua.lab"                       # (e)
```

4.

Εκτελούμε `reboot`.

5.

Εκτελούμε `service isc-dhcpd status` και επιβεβαιώνουμε ότι η υπηρεσία dhcpd τρέχει κανονικά.

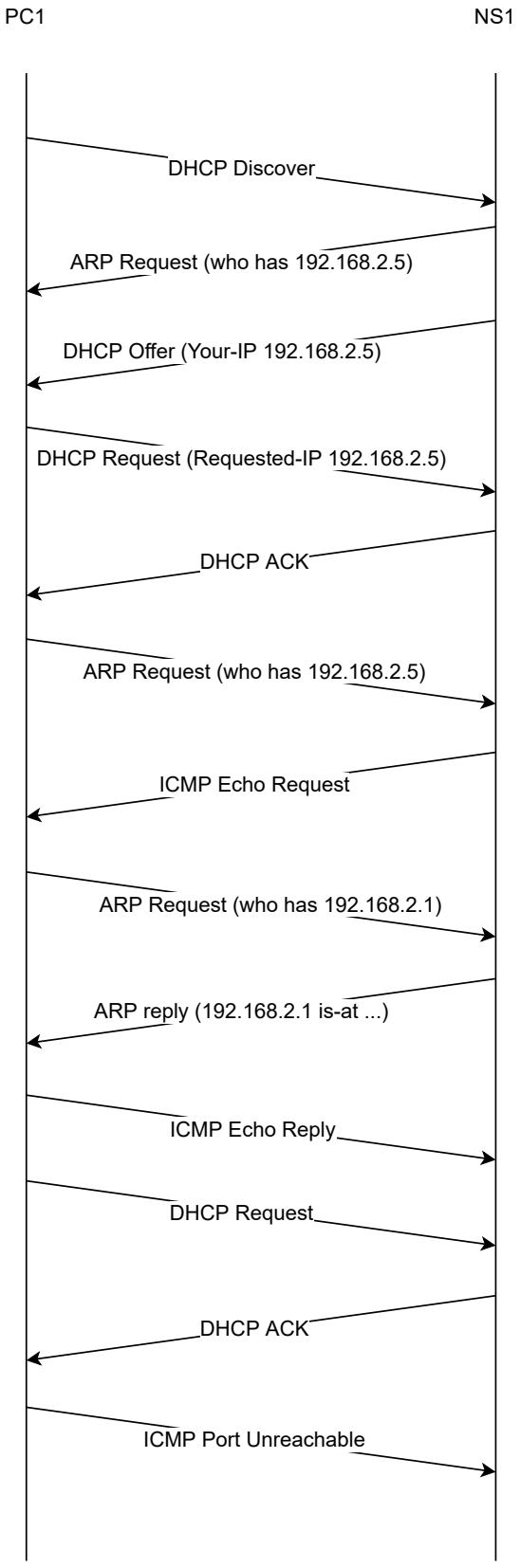
1.1

Στο NS1 εκτελούμε `tcpdump -veni em0`.

1.2

Στο PC1 εκτελούμε `dhclient em0` και περιμένουμε τουλάχιστον δύο λεπτά.

1.3


PC1
NS1
DHCP Discover
ARP Request (who has 192.168.2.5)
DHCP Offer (Your-IP 192.168.2.5)
DHCP Request (Requested-IP 192.168.2.5)
DHCP ACK
ARP Request (who has 192.168.2.5)
ICMP Echo Request
ARP Request (who has 192.168.2.1)
ARP reply (192.168.2.1 is-at ...)
ICMP Echo Reply
DHCP Request
DHCP ACK
ICMP Port Unreachable

1.4

Ανταλλάσσονται:

```
DHCPDISCOVER on em0 to 255.255.255.255
DHCPDISCOVER on em0 to 255.255.255.255
DHCPOFFER from 192.168.2.1
DHCPREQUEST on em0 to 255.255.255.255
DHCPACK from 192.168.2.1
```

1.5

Αποδόθηκε η `192.168.2.5`, ενώ η IP του εξυπηρετητή είναι `192.168.2.1`.

1.6

Μετά από 60 δευτερόλεπτα, όπως φαίνεται στη γραμμή:

```
bound to 192.168.2.5 -- renewal in 60 seconds.
```

1.7

Βλέπουμε ότι χρησιμοποιείται το UDP.

1.8

Οι θύρες πηγής και προορισμού είναι οι 67 (PC1) και 68 (NS1).

1.9

Έχουμε:

Message	Source IP	Destination IP
DHCP DISCOVER	0.0.0.0	255.255.255.255
DHCP OFFER	192.168.2.1	192.168.2.5
DHCP REQUEST	0.0.0.0	255.255.255.255
DHCP ACK	192.168.2.1	192.168.2.5

1.10

Έχουμε:

Message	Source MAC	Destination MAC
DHCP DISCOVER	08:00:27:cc:ef:48	ff:ff:ff:ff:ff:ff
DHCP OFFER	08:00:27:64:52:f6	08:00:27:cc:ef:48
DHCP REQUEST	08:00:27:cc:ef:48	ff:ff:ff:ff:ff:ff
DHCP ACK	08:00:27:64:52:f6	08:00:27:cc:ef:48

1.11

Μπορεί να το κάνει χρησιμοποιώντας την διεύθυνση IP `0.0.0.0`, τις διευθύνσεις MAC, και την ευρυεκπομπή (όπως φαίνεται στα μηνύματα DHCP DISCOVER/REQUEST).

1.12

Ναι, παρατηρήσαμε πλαίσια ARP από τον NS1, ο οποίος θέλει να εξακριβώσει αν χρησιμοποιεί κάποιος τη διεύθυνση IP που σκοπεύει να προσφέρει.

1.13

Όχι, δεν παρατηρήσαμε.

1.14

Με αυτό το πλαίσιο ARP το PC1 προσπαθεί να επιβεβαιώσει ότι δεν υπάρχει άλλος υπολογιστής με αυτή τη διεύθυνση IP.

1.15

Ναι, παρατηρήσαμε ένα ICMP request από τον NS1, και το αντίστοιχο ICMP reply από το PC1. Αυτή η ανταλλαγή γίνεται προκειμένου να επιβεβαιωθεί ότι η απόδοση διεύθυνσης έγινε χωρίς κάποιο πρόβλημα.

1.16

Διαρκεί 120 δευτερόλεπτα (`Lease-Time: 120`).

1.17

Περιέχει τα επιπλέον options:

```
Server-ID Option 54, length 4: 192.168.2.1
Requested_IP Option 50, length 4: 192.168.2.5
```

1.18

Στο δεύτερο DHCP request:

- Δεν χρησιμοποιείται πλέον η MAC προορισμού `ff:ff:ff:ff:ff:ff` (broadcast), αλλά η `08:00:27:64:52:f6` (NS1)
- Δεν χρησιμοποιείται η IP πηγής `0.0.0.0`, αλλά η `192.168.2.5` (PC1)
- Δεν χρησιμοποιείται η IP προορισμού `255.255.255.255` (broadcast), αλλά η `192.168.2.1` (NS1).
- Εμφανίζεται το επιπλέον option `"Client IP: 192.168.2.5"`
- **Δεν** εμφανίζεται το option `"Server-ID Option 54, length 4: 192.168.2.1"`.
- **Δεν** εμφανίζεται το option `"Requested-IP Option 50, length 4: 192.168.2.5"`.

1.19

Αυτό συμβαίνει διότι ο πελάτης DHCP (PC1) ανανέωσε τη διεύθυνση IPv4 του, οπότε η σύνδεση στη θύρα 68 δεν χρειάζεται πλέον (σε αντίθεση με αυτή του εξυπηρετητή DHCP στη θύρα 67, που πρέπει να παραμένει ανοιχτή για να ακούσει τυχόν αιτήματα από πελάτες).

1.20

Στο αρχείο `/var/db/dhcpd/dhcpd.leases`.

1.21

Γίνονται κάθε ένα λεπτό.

1.22

Περιέχει τις πληροφορίες:

```
starts ...
ends ...
cltt ...
binding state ...
next binding state ...
rewind binding state ...
hardware ethernet ...
uid ...
client-hostname ...
```

1.23

Στο αρχείο `/var/db/dhclient.leases.em0`.

1.24

Περιέχει τις πληροφορίες:

```
interface ...
fixed-address ...
option subnet-mask ...
option routers ...
option broadcast-address ...
option dhcp-lease-time ...
option dhcp-message-type ...
option dhcp-server-identifier ...
renew ...
rebind ...
expire ...
```

1.25

Ο ζητούμενος χρόνος είναι ίσος με `rebind - renew = 45 sec`.

1.26

Ζήτησε 10 παραμέτρους:

```
Subnet-Mask
BR
Time-Zone
Classless-Static-Route
Default-Gateway
Domain-Name
Domain-Name-Server
Hostname
Option 119 (Domain Search List)
MTU
```

1.27

Προσδιορίζει τις παραμέτρους `Subnet-Mask, BR`, και `Default-Gateway`.

1.28

Στον NS1 εκτελούμε `tcpdump -ni em0`.

1.29

Σε δεύτερη κονσόλα στον NS1 εκτελούμε `service isc-dhcpd stop`.

1.30

Στο PC1 εκτελούμε συνεχώς `ifconfig em0` μέχρι η διεπαφή `em0` να μην έχει πλέον διεύθυνση IP. Μόλις συμβεί αυτό, εκτελούμε `service isc-dhcpd start` στον NS1.

1.31

Στο PC1 εκτελούμε συνεχώς `ifconfig em0` μέχρι η διεπαφή `em0` να ξαναποκτήσει διεύθυνση IP. Μόλις συμβεί αυτό, σταματάμε την καταγραφή στον NS1.

1.32

Στάλθηκαν 4 μηνύματα DHCP:

```
#1 (3 sec) #2 (5 sec) #3 (11 sec) #4 (26 sec) #5
```

1.33

Λαμβάνει απάντηση "ICMP udp port 67 unreachable", που σημαίνει ότι ο εξυπηρετητής DHCP δεν ακούει στην θύρα 67, το οποίο είναι λογικό, αφού τον απενεργοποιήσαμε, και έτσι δεν προσφέρεται η υπηρεσία DHCP.

1.34

Είναι η διεύθυνση `255.255.255.255` (broadcast).

1.35

Χρησιμοποιείται η `255.255.255.255` ως διεύθυνση προορισμού, επειδή αφού λήξει ο χρόνος επανασύνδεσης (rebind), το PC1 προσπαθεί να δανειστεί μία νέα διεύθυνση από οποιονδήποτε άλλο εξυπηρετητή, όχι υποχρεωτικά από αυτόν που δανείστηκε πριν.

1.36

Είναι:

```
MAC Destination: ff:ff:ff:ff:ff:ff
IP Destination: 255.255.255.255
```

Το πεδίο του μηνύματος που δείχνει ότι έχει απολεσθεί η διεύθυνση IP είναι το:

```
Requested-IP: 192.168.2.5
```

1.37

Επειδή ο NS1 προσπαθεί να επιβεβαιώσει ότι δεν χρησιμοποιείται από κάποιον άλλον η διεύθυνση που πρόκειται να προσφέρει.

1.38

Στο PC1 εκτελούμε `cat /var/db/dhclient.leases.em0`. Παρατηρούμε ότι με κάθε ανανέωση της IP διεύθυνσης προστίθεται ένα νέο δάνειο στο αρχείο αυτό.

1.39

Αυτό συμβαίνει διότι το DHCP πρέπει να είναι συμβατό με το BOOTP (RFC 2131, "DHCP must provide service to existing BOOTP clients") και ως εκ τούτου, ακολουθούνται αυτά που περιγράφονται στο RFC 951:

We could not simply allow the client to pick a 'random' port number for the UDP source port field; **since the server reply may be broadcast, a randomly chosen port number could confuse other hosts that happened to be listening on that port.**

Άσκηση 2: Εγκατάσταση εξυπηρετητή DNS

Εγκαθιστούμε έναν εξυπηρετητή DNS στο NS1:

1.

Εκτελούμε `pkg install unbound`.

2.

Εκτελούμε `sysrc unbound_enable="YES"`.

3.

Δημιουργούμε ένα προσωρινό αρχείο `/var/tmp/unbound.conf` με περιεχόμενο:

```
server:
interface: 0.0.0.0
do-ip4: yes
do-ip6: yes
do-udp: yes
do-tcp: yes
access-control: 192.168.2.0/24 allow
private-domain: "ntua.lab"
local-zone: "ntua.lab." static
local-data: "ntua.lab. 360 IN SOA ns1.ntua.lab. admin.ntua.lab.
                                          20200501 3600 1200 604800 10800"
local-data: "ntua.lab. 360 IN NS ns1.ntua.lab."
local-data: "ntua.lab. IN MX 10 192.168.2.1"
local-data: "ntua.lab. IN A 192.168.2.1"
local-data: "ns1.ntua.lab. IN A 192.168.2.1"
local-data: "www.ntua.lab. IN CNAME ntua.lab"
local-zone: "2.168.192.in-addr.arpa." static
local-data-ptr: "192.168.2.1 ns1.ntua.lab."
forward-zone:
name:"."
forward-addr: 1.1.1.1
forward-addr: 8.8.8.8
forward-addr: 9.9.9.9
```

4.

Εκτελούμε `unbound-checkconf /var/tmp/unbound.conf`. Δεν υπάρχουν λάθη, οπότε εκτελούμε `cp /var/tmp/unbound.conf /usr/local/etc/unbound/unbound.conf`.

5.

Εκτελούμε `cat /etc/resolv.conf`. Το αρχείο υπάρχει, οπότε εκτελούμε `rm /etc/resolv.conf`. Δημιουργούμε νέο `/etc/resolv.conf` με περιεχόμενα:

```
search ntua.lab
nameserver 192.168.2.1
```

6.

Προσθέτουμε στην αρχή του `/usr/local/etc/dhcpd.conf` τις γραμμές:

```
option domain-name "ntua.lab";
option domain-name-servers 192.168.2.1;
```

7.

Εκτελούμε `service isc-dhcpd restart`. Δεν υπάρχουν λάθη.

8.

Εκτελούμε `poweroff` και ύστερα δημιουργούμε έναν κλώνο του NS1, τον NS2.

Πριν ξεκινήσουμε την άσκηση, εκτελούμε:

```
### PC1 ###

ifconfig em0 192.168.2.5/28
rm /etc/resolv.conf

### PC2 ###

ifconfig em0 192.168.2.6/28
rm /etc/resolv.conf
```

Επίλυση ονομάτων μέσω του αρχείου /etc/hosts

2.1

Στο PC1, τροποποιούμε το `/etc/hosts` ως εξής: (`"---"` σημαίνει διαγραφή γραμμής, ενώ `"+++"` σημαίνει προσθήκη γραμμής):

```
--- ::1             localhost localhost.my.domain
+++ ::1             localhost localhost.ntua.lab

--- 127.0.0.1       localhost localhost.my.domain
+++ 127.0.0.1       localhost localhost.ntua.lab

+++ 192.168.2.5     PC1       PC1.ntua.lab
+++ 192.168.2.6     PC2       PC2.ntua.lab
```

2.2

Στο PC1 εκτελούμε:

```
ping PC2
ping pc2
ping pc2.NTUA.LAB
```

Και στις 3 περιπτώσεις απαντά το PC2, ενώ δεν έχει σημασία η χρήση μικρών ή κεφαλαίων γραμμάτων.

2.3

Στο PC2, τροποποιούμε το `/etc/hosts` ως εξής:

```
--- ::1             localhost localhost.my.domain
+++ ::1             localhost localhost.ntua.lab

--- 127.0.0.1       localhost localhost.my.domain
+++ 127.0.0.1       localhost localhost.ntua.lab

+++ 192.168.2.5     PC1       PC1.ntua.lab
+++ 192.168.2.6     PC2       PC2.ntua.lab
```

Ύστερα εκτελούμε `ping PC1` και επιβεβαιώνουμε ότι όντως απαντά το PC1.

2.4

Στο PC2 διαγράφουμε την εγγραφή του `/etc/hosts`:

```
192.168.2.5       PC1       PC1.ntua.lab
```

Ύστερα εκτελούμε `ping PC1`. Αυτή τη φορά λαμβάνουμε μήνυμα λάθους:

```
ping: cannot resolve PC1: Host name lookup failure
```

Επίλυση ονομάτων μέσω του εξυπηρετητή DNS

2.5

Ξεκινάμε τον NS1 και προσθέτουμε στο `/var/tmp/unbound.conf` τις γραμμές:

```
local-data: "PC1.ntua.lab. IN A 192.168.2.5"
local-data: "PC2.ntua.lab. IN A 192.168.2.6"
```

2.6

Στο NS1, στο `/var/tmp/unbound.conf`, προσθέτουμε τις γραμμές:

```
local-data-ptr: "192.168.2.5 PC1.ntua.lab."
local-data-ptr: "192.168.2.6 PC2.ntua.lab."
```

2.7

Στον NS1 εκτελούμε:

```
unbound-checkconf /var/tmp/unbound.conf      # No errors found.
cp /var/tmp/unbound.conf /usr/local/etc/unbound/unbound.conf
service unbound restart
```

2.8

Στον NS1 εκτελούμε `tcpdump -vni em0`.

2.9

Στο PC1 εκτελούμε:

```
ifconfig em0 delete
dhclient em0
```

2.10

Σταματάμε την καταγραφή. Το PC1 έλαβε την `192.168.2.5`.

2.11

Απέδωσε επιπλέον τις παραμέτρους `"Domain-Name"` και `"Domain-Name-Server"`.

2.12

Στο PC1 εκτελούμε `cat /etc/resolv.conf`. Το αρχείο έχει δημιουργηθεί και έχει περιεχόμενο:

```
search ntua.lab
nameserver 192.168.2.1
```

2.13

Στο PC1 εκτελούμε:

```
host 192.168.2.5
```

OR

```
drill -x 192.168.2.5
```

Στην `192.168.2.5` αντιστοιχεί το όνομα `PC1.ntua.lab`.

2.14

Στο PC1 εκτελούμε `host NS1` και έχουμε:

```
NS1.ntua.lab has address 192.168.2.1
```

2.15

Στο PC1 εκτελούμε `ping ns1`. Το ping είναι επιτυχές.

2.16

Στο PC2 εκτελούμε:

```
ifconfig em0 delete
dhclient em0
```

2.17

Έλαβε τη διεύθυνση `192.168.2.6`.

2.18

Στο PC2 εκτελούμε `ping PC1`. Το ping είναι επιτυχές.

2.19

Την έλαβε από τον εξυπηρετητή DNS, αφού προηγουμένως είχαμε διαγράψει την σχετική εγγραφή του `/etc/hosts` που αφορά το PC1 (ερώτημα 2.4). Μπορούμε να το επιβεβαιώσουμε και στην πράξη, κάνοντας καταγραφή στον NS1 στη διεπαφή `em0` πριν εκτελέσουμε το ping.

2.20

Στον PC1, αλλάζουμε τη διεύθυνση IP σε `192.168.2.7` στο αρχείο `/etc/hosts` και ύστερα εκτελούμε `ping PC2`. Το ping αποτυγχάνει με μήνυμα λάθους "Host is down".

2.21

Συμπεραίνουμε ότι πρώτα ελέγχεται το αρχείο `/etc/hosts`, και αν δεν υπάρχει σχετική εγγραφή καλείται ο εξυπηρετητής DNS.

2.22

Στο PC1 εκτελούμε `cat /etc/nsswitch.conf`. Είναι:

```
hosts: files dns
```

Δηλαδή πρώτα ελέγχεται το αρχείο `/etc/hosts` και ύστερα καλούνται οι εξυπηρετητές DNS. Η σειρά αυτή συμφωνεί με αυτή που είδαμε προηγουμένως.

2.23

Στο PC1 εκτελούμε `host PC2`. Η έξοδος της εντολής είναι:

```
PC2.ntua.lab has address 192.168.2.6
```

2.24

Σύμφωνα με την εντολή `man host`, το `host` είναι εργαλείο που εκτελεί DNS lookups, οπότε δεν ασχολείται καθόλου με το περιεχόμενο του αρχείου `/etc/hosts`.

2.25

Στο PC1 εκτελούμε:

```
rm /etc/resolv.conf
resolvconf -u
cat /etc/resolv.conf
```

Τώρα το περιεχόμενο του `/etc/resolv.conf` είναι:

```
search ntua.lab
nameserver 192.168.2.1
```

Πρωτόκολλο DNS

2.26

Στο NS1 εκτελούμε `tcpdump -vni em0 "not port 67 and not port 68"`.

2.27

Στο PC1 εκτελούμε `host ntua.lab`.

2.28

Ναι, υπάρχει.

2.29

Χρησιμοποιήθηκε το UDP.

2.30

Οι θύρες προέλευσης και προορισμού είναι οι 53, 16104, 21507 και 45142.

2.31

Η θύρα 53.

2.32

Στο NS1 εκτελούμε `tcpdump -vni em0 "udp port 53"`.

2.33

Στο PC1 εκτελούμε `host NS1`.

2.34

Ανταλλάχθηκαν 6 μηνύματα.

2.35

Αντιστοιχούσαν σε ερωτήματα είδους `A`, `AAAA` και `MX` για το όνομα `NS1.ntua.lab`.

2.36

Δόθηκε απάντηση μόνο στο ερώτημα είδους `A` (δηλαδή για την διεύθυνση IPv4).

2.37

Στο PC1 εκτελούμε:

```
drill ns1
drill ns1.ntua.lab
```

2.38

Έγιναν οι ερωτήσεις για τα ονόματα:

```
### drill ns1 ###
---> name: ns1

### drill ns1.ntua.lab ###
---> name: ns1.ntua.lab
```

και λήφθηκαν οι απαντήσεις:

```
### drill ns1 ###
---> <no answer>

### drill ns1.ntua.lab ###
---> ns1.ntua.lab.   3600    IN      A       192.168.2.1
```

2.39

Συμπεραίνουμε ότι στην εντολή `host` δεν απαιτείται η χρήση του επιθέματος `ntua.lab`, ενώ στην εντολή `drill` χρειάζεται.

2.40

Στο PC1 εκτελούμε:

```
ping localhost
ping pc1
```

Δεν παράγονται ερωτήσεις προς τον εξυπηρετητή DNS σε καμία περίπτωση.

2.41

Στο PC1 εκτελούμε `ping -c 1 ns1`.

2.42

Ανταλλάχθηκαν 2 μηνύματα DNS, που αφορούσαν το ερώτημα `A? ns1.ntua.lab`, δηλαδή είδους `A` (IPv4 διεύθυνση) για το όνομα `ns1.ntua.lab`.

2.43

Στο PC1 εκτελούμε:

```
ping -c 1 ns1
ping -c 1 ns1
ping -c 1 ns1
```

Παρατηρούμε να παράγονται 3 επιπλέον ερωτήματα προς τον εξυπηρετητή DNS, όσα και τα ping που εκτελέσαμε.

2.44

Συμπεραίνουμε ότι οι απαντήσεις του εξυπηρετητή DNS δεν αποθηκεύονται προσωρινά στο PC1.

Άσκηση 3: Εγκατάσταση εξυπηρετητή HTTP

Για την εγκατάσταση του εξυπηρετητή HTTP στον SRV κάνουμε τα εξής:

1.

Επιβεβαιώνουμε ότι η διεπαφή `em1` είναι σε NAT.

2.

Εκτελούμε `dhclient em1`.

3.

Εκτελούμε `ping www.google.com`. Το ping είναι επιτυχές.

4.

Εκτελούμε `pkg install lighttpd`.

5.

Απενεργοποιούμε τις διεπαφές πλην της `em0` και εκτελούμε `rm /etc/resolv.conf`.

3.1

Στον SRV εκτελούμε:

```
sysrc hostname="SRV"
sysrc lighttpd_enable="YES"
```

3.2

Στον SRV εκτελούμε:

```
mkdir /usr/local/www/data
```

3.3

Στον SRV εκτελούμε:

```
echo "Hello World!" > /usr/local/www/data/index.html
```

3.4

Στον SRV εκτελούμε:

```
reboot
rm /etc/resolv.conf
```

3.5

Μπορούμε να εκτελέσουμε `service lighttpd status`.

3.6

Μπορούμε να εκτελέσουμε την εντολή `netstat -an | grep 80` ή την `netstat -a | grep http`. Αν υπάρχει εξυπηρετητής http θα εμφανιστεί έξοδος της μορφής:

```
tcp6    0    0    *.http            *.*            LISTEN
tcp4    0    0    *.http            *.*            LISTEN
```

3.7

Τοποθετούμε τη διεπαφή `em0` του SRV στο LAN1 και ύστερα εκτελούμε:

```
ifconfig em0 192.168.2.3/28
```

3.8

Στο `/var/tmp/unbound.conf` του NS1 προσθέτουμε τη γραμμή:

```
local-data: "SRV.ntua.lab. IN A 192.168.2.3"
```

3.9

Στο `/var/tmp/unbound.conf` του NS1 προσθέτουμε τη γραμμή:

```
local-data-ptr: "192.168.2.3 SRV.ntua.lab."
```

3.10

Στον NS1 εκτελούμε:

```
unbound-checkconf /var/tmp/unbound.conf   # No errors found.
cp /var/tmp/unbound.conf /usr/local/etc/unbound/unbound.conf
service unbound restart
```

3.11

Στον SRV εκτελούμε `tcpdump -ni em0`.

3.12

Στο PC1 εκτελούμε `fetch http://srv.ntua.lab`.

3.13

Χρησιμοποιήθηκε το TCP, ενώ ο εξυπηρετητής http ακούει στη θύρα 80.

3.14

Αποθηκεύτηκε στο `srv.ntua.lab`.

Άσκηση 4: Εγκατάσταση ιδιωτικού δρομολογητή και Firewall

4.1

Στον NS1 εκτελούμε `sysrc gateway_enable="YES"`.

4.2

Στον NS1 εκτελούμε `sysrc firewall_enable="YES"`.

4.3

Στον NS1 εκτελούμε `sysrc firewall_type="open"`.

4.4

Στον NS1 εκτελούμε `sysrc firewall_nat_enable="YES"`.

4.5

Στον NS1 εκτελούμε `sysrc ifconfig_em2="192.168.2.17/28"`.

4.6

Στον NS1 εκτελούμε:

```
cat /etc/rc.conf        # Everything OK.
```

4.7

Στο NS1 εκτελούμε `poweroff`, τοποθετούμε τη διεπαφή `em2` του NS1 στο DMZ και το επανεκκινούμε. Ύστερα εκτελούμε `netstat -rn` και βλέπουμε ότι η προκαθορισμένη πύλη είναι σωστά ρυθμισμένη.

4.8

Στο NS1 εκτελούμε `vi /etc/resolv.conf` και αλλάζουμε τα περιεχόμενα του αρχείου σε:

```
search ntua.lab
nameserver 192.168.2.1
```

Επιβεβαιώνουμε ότι η επίλυση ονομάτων λειτουργεί.

4.9

Στο PC1 εκτελούμε:

```
sysrc ifconfig_em0="DHCP"
service netif restart
```

4.10

Στο PC2 εκτελούμε:

```
sysrc ifconfig_em0="192.168.2.4/28"
sysrc defaultrouter="192.168.2.1"
```

4.11

Στο PC2 εκτελούμε:

```
service netif restart
service routing restart
vi /etc/resolv.conf
```

Αλλάζουμε το περιεχόμενο του `/etc/resolv.conf` σε:

```
nameserver 192.168.2.1
```

Τέλος επιβεβαιώνουμε ότι η επίλυση ονομάτων λειτουργεί με οποιαδήποτε από τις παρακάτω εντολές:

```
ping PC1
---> Successful

host PC1.ntua.lab
---> PC1.ntua.lab has address 192.168.2.5

host 192.168.2.5
---> 5.2.168.192.in-addr.arpa domain name pointer PC1.ntua.lab
```

4.12

Τοποθετούμε την διεπαφή em0 του SRV στο τοπικό δίκτυο DMZ και ύστερα στον SRV εκτελούμε:

```
sysrc ifconfig_em0="192.168.2.18/28"
sysrc defaultrouter="192.168.2.17"
service netif restart
service routing restart
```

4.13

Στον NS1 εκτελούμε `vi /var/tmp/unbound.conf` και διορθώνουμε τις γραμμές
("---" → παλιά γραμμή, "+++" → καινούργια γραμμή):

```
--- local-data: "PC2.ntua.lab. IN A 192.168.2.6"
+++ local-data: "PC2.ntua.lab. IN A 192.168.2.4"

--- local-data: "SRV.ntua.lab. IN A 192.168.2.3"
+++ local-data: "SRV.ntua.lab. IN A 192.168.2.18"

--- local-data-ptr: "192.168.2.6 PC2.ntua.lab."
+++ local-data-ptr: "192.168.2.4 PC2.ntua.lab."

--- local-data-ptr: "192.168.2.3 SRV.ntua.lab."
+++ local-data-ptr: "192.168.2.18 SRV.ntua.lab."
```

Ύστερα εκτελούμε:

```
unbound-checkconf /var/tmp/unbound.conf      # No errors found.
cp /var/tmp/unbound.conf /usr/local/etc/unbound/unbound.conf
service unbound restart
```

4.14

Στον SRV εκτελούμε:

```
ping 192.168.2.5   # PC1
ping 192.168.2.4   # PC2
ping 192.168.2.1   # NS1
```

Ναι, μπορούμε να κάνουμε ping στα μηχανήματα του LAN1 χρησιμοποιώντας την IP διεύθυνσή τους.

4.15

Στο NS1 εκτελούμε:

```
ipfw add 2000 deny all from any to 192.168.2.0/28 in via em2
```

4.16

Στον SRV εκτελούμε `ping 192.168.2.5`. Δεν λαμβάνουμε απάντηση.

4.17

Στον NS1 εκτελούμε (όλο μαζί μία εντολή):

```
ipfw add 1900 allow all from 192.168.2.0/28 to 192.168.2.16/28 \
in recv em0 keep-state
```

4.18

Στο PC1 εκτελούμε `ping SRV`. Το ping είναι επιτυχές.

4.19

Στο NS1 εκτελούμε `ping 147.102.1.1`. Το ping είναι επιτυχές.

4.20

Στο PC1 εκτελούμε `ping 147.102.1.1`. Δεν λαμβάνουμε απάντηση.

4.21

Στον NS1 εκτελούμε:

```
ipfw nat 111 config unreg_only if em1 reset
```

4.22

Στον NS1 εκτελούμε:

```
ipfw add 3000 nat 111 ip4 from any to any via em1
```

4.23

Στο PC1 εκτελούμε `ping 147.102.1.1`. Το ping είναι επιτυχές.

4.24

Στο PC1 εκτελούμε `host 147.102.1.1`. Το όνομα του μηχανήματος με αυτή τη διεύθυνση IP είναι `theseas.softlab.ece.ntua.gr`.

4.25

Στον NS1 εκτελούμε `tcpdump -ni em1`.

4.26

Στο PC1 εκτελούμε `ping -c 2 www.ntua.gr`. Τα πακέτα που παράγει το PC1 εμφανίζονται με διεύθυνση πηγής `10.0.3.15`.

4.27

Είναι η `147.102.224.101`.

4.28

Έγινε προς τον `9.9.9.9 (dns9.quad9.net)`.

4.29

Στον NS1 εκτελούμε `tcpdump -ni em1 "udp port 53"`.

4.30

Στο PC2 εκτελούμε:

```
#1: ping -c 1 www.google.com
#2: ping -c 1 www.cnn.com
#3: ping -c 1 www.yahoo.com
#4: ping -c 1 www.mit.edu
```

Κάθε φορά καλείται ένας εξυπηρετητής από αυτούς που έχουμε ορίσει, δηλαδή `1.1.1.1, 8.8.8.8` και `9.9.9.9`.

4.31

Σε νέο παράθυρο στον NS1 εκτελούμε `tcpdump -ni em0 "udp port 53"`.

4.32

Στο PC1 εκτελούμε `ping -c 1 courses.cn.ntua.gr`. Το canonical name του `courses.cn.ntua.gr` είναι `courses.cn.ece.ntua.gr`.

4.33

Το PC1 έκανε ερώτημα `"A"`:

```
A? courses.cn.ntua.gr.
```

και έλαβε απάντηση `"CNAME", "A"` από τον NS1:

```
CNAME courses.cn.ece.ntua.gr, A 147.102.40.10
```

Επίσης, ο NS1 έκανε ερωτήματα είδους "A" στους εξωτερικούς εξυπηρετητές DNS:

```
#1   A? courses.cn.ntua.gr.
#2   A? courses.cn.ece.ntua.gr
```

και έλαβε απαντήσεις είδους "CNAME", "A" και "A" αντίστοιχα:

```
#1   CNAME courses.cn.ece.ntua.gr, A 147.102.40.10
#2   A 147.102.40.10
```

4.34

Στον NS1 εκτελούμε `tcpdump -vvvni em1 "udp port 53"`.

4.35

Στο PC1 εκτελούμε:

```
drill www.cn.ece.ntua.gr
drill www.cn.ece.ntua.gr
```

Παρατηρήσαμε μόνο ένα ερώτημα DNS:

```
A? www.cn.ece.ntua.gr
```

Οι απαντήσεις DNS ισχύουν για 20 λεπτά.

4.36

Στον NS1 εκτελούμε `tcpdump -vvvni em0 "udp port 53"`. Ύστερα στο PC1 εκτελούμε:

```
drill www.cn.ece.ntua.gr
drill www.cn.ece.ntua.gr
```

Παράγονται μηνύματα DNS κάθε φορά που εκτελούμε την εντολή `drill`. Παρατηρούμε ότι η χρονική διάρκεια ισχύος των απαντήσεων DNS μειώνεται συνεχώς.

4.37

Συμπεραίνουμε ότι οι απαντήσεις που λαμβάνει ο τοπικός εξυπηρετητής DNS στο NS1 αποθηκεύονται προσωρινά.

4.38

Στον SRV εκτελούμε `ping 147.102.224.101`. Το ping είναι επιτυχές.

4.39

Στον SRV εκτελούμε `ping www.ntua.gr`. Το ping αποτυγχάνει με μήνυμα λάθους:

```
ping: cannot resolve www.ntua.gr: Host name lookup failure
```

Αυτό συμβαίνει διότι:

- Δεν υπάρχει εγγραφή σχετική με τον `www.ntua.gr` στο αρχείο `/etc/hosts`.
- Δεν έχει οριστεί εξυπηρετητής DNS, αφού δεν υπάρχει το αρχείο `/etc/resolv.conf`.

4.40

Στον SRV εκτελούμε:

```
echo "nameserver 192.168.2.17" > /etc/resolv.conf
```

4.41

Στον SRV εκτελούμε `ping www.ntua.gr`. Το ping είναι επιτυχές.

4.42

Στο PC1 εκτελούμε `host www.ntua.lab` και παίρνουμε την απάντηση:

```
www.ntua.lab is an alias for ntua.lab
```

Για να πάρουμε τη διεύθυνση IP πρέπει να ξαναεκτελέσουμε την `host`, αλλά αυτή τη φορά να βάλουμε σαν όρισμα την έξοδο του προηγούμενου `host`, δηλαδή:

```
host ntua.lab
```

Οπότε παίρνουμε ως απάντηση τη διεύθυνση `192.168.2.1`.

Στο PC1 εκτελούμε `ping www.ntua.lab`. Το ping αποτυγχάνει με μήνυμα λάθους:

```
ping: cannot resolve www.ntua.lab: Unknown server error
```

4.43

Στον NS1 εκτελούμε `vi /usr/local/etc/unbound/unbound.conf` και προσθέτουμε πριν από την εγγραφή CNAME τη γραμμή:

```
local-data: "www.ntua.lab. IN A 192.168.2.18"
```

Ύστερα εκτελούμε `service unbound restart`.

4.44

Στο PC1 εκτελούμε `ping www.ntua.lab`. Απαντά το SRV (`192.168.2.18`).

Άσκηση 5: Εγκατάσταση δημόσιου δρομολογητή και DNS

Πριν ξεκινήσουμε, τοποθετούμε τις διεπαφές του NS2 ως εξής:

```
em0: LAN2
em1: NAT
em2: WAN
```

5.1

Στον NS2 εκτελούμε `sysrc hostname="ns2.ntua.lab"`.

5.2

Στον NS2 εκτελούμε:

```
sysrc ifconfig_em0="192.0.2.1/29"
sysrc ifconfig_em2="192.0.2.9/29"
```

5.3

Στον NS2 εκτελούμε `sysrc ifconfig_em1="DHCP"`.

5.4

Στον NS2 εκτελούμε `sysrc gateway_enable="YES"`.

5.5

Στον NS2 εκτελούμε `sysrc firewall_enable="YES"`.

5.6

Στον NS2 εκτελούμε `sysrc firewall_type="open"`.

5.7

Στον NS2 εκτελούμε `sysrc firewall_nat_enable="YES"`.

5.8

Στον NS2 εκτελούμε:

```
sysrc -a     # To see DHCP server related settings

# DHCP server related:
# dhcp_ifaces
# dhcpd_enable

sysrc -x dhcp_ifaces
sysrc -x dhcpd_enable
```

5.9

Στον NS2 εκτελούμε `sysrc -a`. Επιβεβαιώνουμε ότι υπάρχει (`unbound_enable: YES`).

5.10

Στον NS2 εκτελούμε `vi /var/tmp/unbound.conf` και αλλάζουμε τις γραμμές
("---" → γραμμή που διαγράφηκε, "+++" → γραμμή που προστέθηκε):

```
### NTUA.LAB RELATED LINES ###

--- private-domain: "ntua.lab"
--- local-zone: "ntua.lab." static
--- local-data: "ntua.lab. 360 IN SOA ns1.ntua.lab. admin.ntua.lab.
                                    20200501 3600 1200 604800 10800"
--- local-data: "ntua.lab. 360 IN NS ns1.ntua.lab."
--- local-data: "ntua.lab. IN MX 10 192.168.2.1"
--- local-data: "ntua.lab. IN A 192.168.2.1"
--- local-data: "ns1.ntua.lab. IN A 192.168.2.1"
--- local-data: "www.ntua.lab. IN CNAME ntua.lab."
--- local-data-ptr: "192.168.2.1 ns1.ntua.lab."

### ACCESS-CONTROL ###

--- access-control: 192.168.2.0/24 allow

### NEW LINES ###

+++ access-control: 192.0.2.0/24 allow
+++ local-zone: "ntua.lab." redirect
+++ local-data: "ntua.lab. IN A 192.0.2.10"
```

Ύστερα εκτελούμε:

```
unbound-checkconf /var/tmp/unbound.conf    # No errors found.
cp /var/tmp/unbound.conf /usr/local/etc/unbound/unbound.conf
```

5.11

Στον NS2 εκτελούμε:

```
reboot
netstat -rn     #  default gateway exists, OK.
```

5.12

Στον NS2 εκτελούμε:

```
ipfw nat 222 config if em1 reset same_ports
```

5.13

Στον NS2 εκτελούμε:

```
ipfw add 1100 nat 222 ip4 from any to any via em1
```

5.14

Στο PC2 εκτελούμε:

```
sysrc ifconfig_em0="192.0.2.2/29"
sysrc defaultrouter="192.0.2.1"
```

5.15

Συνδέουμε το PC2 στο LAN2 και εκτελούμε:

```
service netif restart
service routing restart
echo "nameserver 192.0.2.1" > /etc/resolv.conf
host google.com              # Name resolution OK.
```

5.16

Στο PC2 εκτελούμε `www.ntua.gr`. Το ping είναι επιτυχές.

5.17

Στον NS1 εκτελούμε:

```
sysrc ifconfig_em1="192.0.2.10/29"
sysrc defaultrouter="192.0.2.9"
```

5.18

Μετακινούμε την `em1` του NS1 στο WAN και ύστερα εκτελούμε:

```
service netif restart
service routing restart
```

5.19

Εκτελούμε:

```
### NS1 ###
ipfw zero

### PC1 ###
ping www.ntua.gr
```

```
### NS1 ###
ipfw show          # NAT 111 rule used
ipfw zero

### SRV ###
ping www.ntua.gr

### NS1 ###
ipfw show          # NAT 111 rule used
```

Και τα δύο ping είναι επιτυχή, ενώ παραμένει η λειτουργία του πίνακα nat 111.

5.20

Εκτελούμε:

```
### PC1 ###
host www.ntua.lab

### PC2 ###
host www.ntua.lab
```

Στο PC1 δίνει διεύθυνση `192.168.2.18`, ενώ στο PC2 `192.0.2.10`.

5.21

Στο PC2 εκτελούμε `fetch http://www.ntua.lab`. Η εντολή αποτυγχάνει με μήνυμα λάθους "Connection refused".

5.22

Στον NS1 εκτελούμε:

```
ipfw nat 111 config unreg_only if em1 reset redirect_port tcp 192.168.2.18:80 80
```

5.23

Στο PC2 εκτελούμε `fetch http://www.ntua.lab`. Πλέον μπορούμε να κατεβάσουμε κανονικά την ιστοσελίδα.

5.24

Στο PC2 εκτελούμε `ping www.ntua.lab`. Απαντά το NS1 (`192.0.2.10`).

5.25

Στο PC1 εκτελούμε `ssh lab@www.ntua.lab`. Συνδεόμαστε στο SRV, όπως φαίνεται και από το prompt.

5.26

Στο PC2 εκτελούμε `ssh lab@www.ntua.lab`. Συνδεόμαστε στο μηχάνημα NS1 (όπως φαίνεται από το prompt), αφού στο PC2 το όνομα `www.ntua.lab` αντιστοιχεί στη διεύθυνση `192.0.2.10`, όπως μπορούμε να επιβεβαιώσουμε και με την εντολή `host www.ntua.lab`.

5.27

Στο NS1 εκτελούμε (όλο μαζί μία εντολή):

```
ipfw nat 111 config unreg_only if em1 reset \
redirect_port tcp 192.168.2.18:80 80        \
redirect_port tcp 192.168.2.18:22 22
```

5.28

Στο PC2 εκτελούμε `ssh lab@www.ntua.lab`. Συνδεόμαστε στον SRV, όπως μπορούμε να επιβεβαιώσουμε από το prompt, την εντολή `hostname`, ή -αν θέλουμε να είμαστε απολύτως σίγουροι- διασταυρώνοντας τις διευθύνσεις MAC στην έξοδο της `ifconfig`.