Εργαστήριο Δικτύων Υπολογιστών

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ **4**

ΕΙΣΑΓΩΓΗ ΣΤΗ ΔΡΟΜΟΛΟΓΗΣΗ

Κουστένης Χρίστος | el20227 | 05/03/2024

Άσκηση 1 : Διευθύνσεις IP

1.1

Η IP είναι ο συνολικός αριθμός 32 bit που αποδίδεται σε κάθε διεπαφή για την αναπαράστασή του στο δίκτυο. Αριθμός δικτύου είναι το πρώτο μέρος της IP που χαρακτηρίζει το δίκτυο στο οποίο βρίσκεται η διεπαφή(δηλώνει τον χώρο διευθύνσεων αυτού του δικτύου).

1.2

Επειδή έχουμε το /22, θα κάνουμε πρακτικά τη λογική πράξη AND μεταξύ του 192.220.147.2 και του 255.255.252.0, επομένως αριθμός δικτύου είναι το 192.220.144.0.

1.3

Μας έχει δοθεί το μπλοκ διευθύνσεων 198.20.0.0/22. Επομένως, μάς είναι διαθέσιμα 10 bits για υποδίκτυα και hosts. Θέλουμε τουλάχιστον 100 συσκευές ανά υποδίκτυο, επομένως αναζητούμε n τέτοιο ώστε $2^n > 100 \Rightarrow n = 7$. Άρα από τα 10 διαθέσιμα bits, τα 7 θα χρησιμοποιηθούν για hosts, επομένως μας μένουν 3 bits για υποδίκτυα, άρα συνολικά $2^3 = 8$ υποδίκτυα.

1.4

Η κλάση C.

1.5

Είναι οι : b, d, e

1.6

Ελέγχει με χρήση subnet mask αν ανήκει η διεύθυνση προορισμού στο ίδιο υποδίκτυο.

1.7

Έχουμε το δίκτυο 10.50.10.0/23 → 00001010.00110010.00001010.00000000 με μάσκα 11111111.11111111.11111110.00000000, επομένως διεύθυνση εκπομπής είναι η 00001010.00110010.00001011.11111111 **(= OR operation μεταξύ network address και συμπληρώματος μάσκας)** ή αλλιώς 10.50.11.255.

1.8

Η κλάση C.

1.9

Το Ε.Μ.Π. έχει διευθύνσεις που έχουν ως πρώτο byte το 147, επομένως είναι κλάσης B.

1.10

Έχουν δεσμευτεί 17 bits για το δίκτυο, επομένως απομένουν 32-17 = 15 bits για συσκευές, δίνοντας μας συνολικά 2^{15} = 32768 διαθέσιμες διευθύνσεις. (Στην πραγματικότητα 32766, καθώς η πρώτη είναι δεσμευμένη για το δίκτυο και η τελευταία για broadcast).

1.11

Subnet 1 : 10.11.12.0/25 (10.11.12.0 - 10.11.12.127) → 100 devices

Subnet 2 : 10.11.12.128/26 (10.11.12.128 - 10.11.12.191) → 60 devices

Subnet 3 : 10.11.12.192/27 (10.11.12.192 - 10.11.12.223) → 20 devices

Subnet 4 : 10.11.12.224/28 (10.11.12.224 - 10.11.12.240) → 10 devices

1.12

Ναί υπάρχει, μπορεί να έχει μέχρι 14 υπολογιστές (δεν μπορούμε να χρησιμοποιήσουμε τη broadcast IP).

1.13

Αναζητούμε στην πραγματικότητα τη θέση του δεξιότερου κοινού bit στο 3ο byte των διευθύνσεων που δίνονται. Στο byte αυτό έχουμε τις εξής τιμές:

- 4 → 0000 0100
- 5 → 0000 0101
- 6 → 0000 0110
- 7 → 0000 0111
- 8 → 0000 1000

Παρατηρούμε πως εμφανίζεται το bit 0 στη 4η θέση (ξεκινώντας από τα αριστερά), επομένως τα συμπυκνώνουμε στο μπλοκ 171.12.0.0/20.

Άσκηση 2 : Ένα απλό δίκτυο

2.1

ifconfig em0 192.168.1.1/24 ---> PC1

ifconfig em0 192.168.1.2/28 ---> PC2

ifconfig em0 192.168.1.18/24 ---> PC3

ifconfig em0 192.168.1.29/28 ---> PC4

2.2

Χρησιμοποιήθηκε η επιλογή « Generate new MAC addresses for all network adapters », ούτως ώστε κάθε εικονική κάρτα δικτύου των μηχανών να έχει δική της MAC διεύθυνση. Αν αντιθέτως είχαμε για παράδειγμα 2 κάρτες δικτύου στο ίδιο δίκτυο με τις ίδιες MAC διευθύνσεις, τότε πρωτόκολλα όπως το DHCP για την απόδοση IP διευθύνσεων θα δυσλειτουργούσαν. Αυτό πιθανόν να συνέβαινε σε περίπτωση που χρησιμοποιούσαμε cloning, χωρίς να έχουμε ορίσει την παραπάνω επιλογή.

2.3

```
root@PC:~ # ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=0.560 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=0.361 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=0.348 ms
^C
--- 192.168.1.2 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.348/0.423/0.560/0.097 ms
root@PC:~ #
```

```
root@PC:~ # ping 192.168.1.18
PING 192.168.1.18 (192.168.1.18): 56 data bytes
64 bytes from 192.168.1.18: icmp_seq=0 ttl=64 time=0.672 ms
64 bytes from 192.168.1.18: icmp_seq=1 ttl=64 time=0.333 ms
64 bytes from 192.168.1.18: icmp_seq=2 ttl=64 time=0.357 ms
64 bytes from 192.168.1.18: icmp_seq=3 ttl=64 time=0.342 ms
^C
--- 192.168.1.18 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.333/0.426/0.672/0.142 ms
root@PC:~ #
```

```
root@PC:~ # ping 192.168.1.29
PING 192.168.1.29 (192.168.1.29): 56 data bytes
^C
--- 192.168.1.29 ping statistics ---
7 packets transmitted, 0 packets received, 100.0% packet loss
root@PC:~ #
```

PC1 → PC2 OK

PC1 → PC3 OK

PC1 → PC4 NOT OK (Το PC1 μπορεί να στείλει στο PC4, το PC4 όμως δεν μπορεί να απαντήσει)

2.4

```
root@PC:~ # ping 192.168.1.18
PING 192.168.1.18 (192.168.1.18): 56 data bytes
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
^C
--- 192.168.1.18 ping statistics ---
4 packets transmitted, 0 packets received, 100.0% packet loss
root@PC:~ #
```

```
root@PC:~ # ping 192.168.1.29
PING 192.168.1.29 (192.168.1.29): 56 data bytes
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
```

PC2 → PC3 NOT OK (no route to host)

PC2 → PC4 NOT OK (no route to host)

2.5

```
root@PC:~ # ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
^C
--- 192.168.1.1 ping statistics ---
3 packets transmitted, 0 packets received, 100.0% packet loss
root@PC:~ #
```

```
root@PC:~ # ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
ping: sendto: No route to host
^C
--- 192.168.1.2 ping statistics ---
4 packets transmitted, 0 packets received, 100.0% packet loss
root@PC:~ #
```

```
root@PC:~ # ping 192.168.1.18
PING 192.168.1.18 (192.168.1.18): 56 data bytes
64 bytes from 192.168.1.18: icmp_seq=0 ttl=64 time=0.339 ms
64 bytes from 192.168.1.18: icmp_seq=1 ttl=64 time=0.361 ms
64 bytes from 192.168.1.18: icmp_seq=2 ttl=64 time=0.455 ms
64 bytes from 192.168.1.18: icmp_seq=3 ttl=64 time=0.333 ms
^C
--- 192.168.1.18 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.333/0.372/0.455/0.049 ms
root@PC:~ #
```

PC4 → PC1 NOT OK (no route to host)

PC4 → PC2 NOT OK (no route to host)

PC4 → PC3 OK

2.6

```
root@PC:~ # ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=0.648 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.338 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.336 ms
^C
--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.336/0.441/0.648/0.147 ms
root@PC:~ # ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
^C
--- 192.168.1.2 ping statistics ---
11 packets transmitted, 0 packets received, 100.0% packet loss
root@PC:~ #
```

PC3 → PC1 OK

PC3 → PC2 NOT OK (Το PC3 μπορεί να στείλει στο PC2, το PC2 όμως δεν μπορεί να απαντήσει)

2.7

Η διεύθυνση δεν ανήκει στο υποδίκτυο στο οποίο βρίσκεται ο αποστολέας των pings.

Το μήνυμα « No route to host » εμφανίστηκε στα εξής ping:

1) PC2 προς το PC3

2) PC2 προς το PC4

3) PC4 προς το PC1

4) PC4 προς το PC2

Ο λόγος που αυτά απέτυχαν, είναι πως εάν βάλουμε τη μάσκα της διεύθυνσης του αποστολέα στη διεύθυνση του παραλήπτη, θα δούμε πως τα υποδίκτυα είναι διαφορετικά μεταξύ τους. Αναλυτικότερα:

1) Αρχικά, βρίσκουμε το υποδίκτυο του PC2, εφαρμόζοντας τη μάσκα του στη διεύθυνση του και παίρνουμε το 192.168.1.0. Στη συνέχεια, βρίσκουμε του PC3 και παίρνουμε 192.168.1.16, άρα 192.168.1.0 ≠ 192.168.1.16.

2) Το υποδίκτυο του PC4 με τη μάσκα του PC2 γίνεται 192.168.1.16 ≠ 192.168.1.0.

3) Το υποδίκτυο του PC4 είναι το 192.168.1.16. Εφαρμόζοντας τη μάσκα του στη διεύθυνση του PC1 παίρνουμε το 192.168.1.0, άρα είναι επομένως σε διαφορετικά υποδίκτυα.

4) Εφαρμόζοντας τη μάσκα του υποδικτύου του PC4 στο PC2 παίρνουμε 192.168.1.0, το οποίο είναι διαφορετικό του 192.168.1.16.

2.8

Διότι η διεύθυνση παραλήπτη μπορεί να ανήκει στο ίδιο υποδίκτυο, αλλά αν δεν ανήκει αντίστοιχα η διεύθυνση αποστολέα στο υποδίκτυό της, τότε δεν θα μπορούν να σταλούν ICMP replies.

Ειδικότερα, δε λαμβάνουμε απάντηση στο ping στις εξής περιπτώσεις:

1) PC1 προς PC4

2) PC3 προς PC2

2.9

ifconfig em0 192.168.1.1/28

ifconfig em0 192.168.1.2/28

ifconfig em0 192.168.1.18/28

ifconfig em0 192.168.1.29/28

2.10

Αποτυγχάνουν πλέον τα ping:

1. Από το PC1 στο PC3 λαμβάνουμε πλέον « No route to host »

2. Από το PC3 στο PC1 λαμβάνουμε πλέον « No route to host »

2.11

Επιστρέφουν και αυτά « No route to host ».

Άσκηση 3 : Ένα απλό δίκτυο με δρομολογητή

3.1

ifconfig em0 192.168.1.14/28

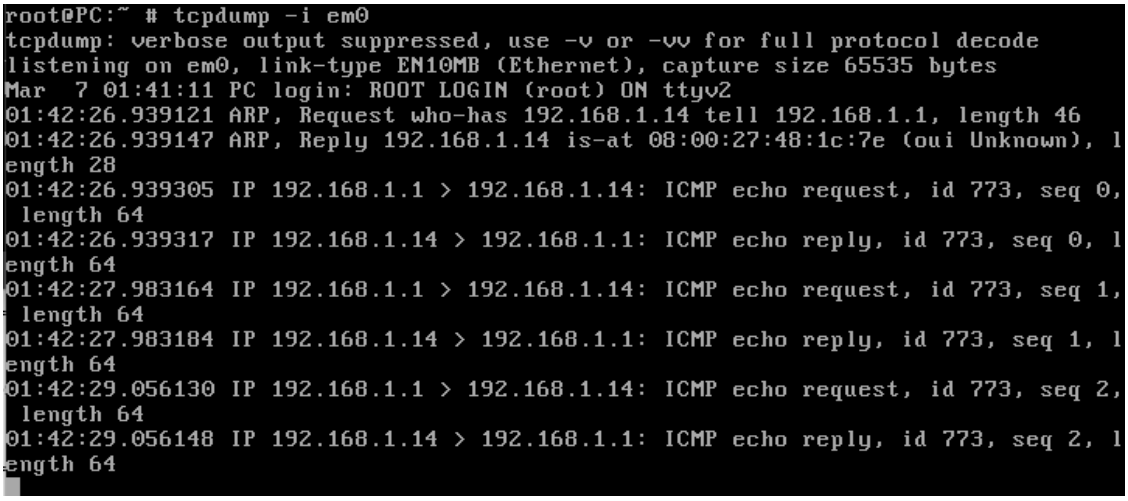
ifconfig em1 192.168.1.17/28

3.2

Μέσω των ρυθμίσεων του VirtualBox, στο πεδίο Network.

3.3

Όπως φαίνεται στην παρακάτω εικόνα παράγονται και ARP και ICMP πακέτα.



3.4

Όπως φαίνεται στην παρακάτω εικόνα παράγονται και ARP και ICMP πακέτα.

```
root@PC:~ # tcpdump -i em1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on em1, link-type EN10MB (Ethernet), capture size 65535 bytes
01:44:41.402298 ARP, Request who-has 192.168.1.17 tell 192.168.1.18, length 46
01:44:41.402331 ARP, Reply 192.168.1.17 is-at 08:00:27:da:e8:ec (oui Unknown),
ength 28
01:44:41.402485 IP 192.168.1.18 > 192.168.1.17: ICMP echo request, id 61956, se
 0, length 64
01:44:41.402497 IP 192.168.1.17 > 192.168.1.18: ICMP echo reply, id 61956, seq
, length 64
01:44:42.471339 IP 192.168.1.18 > 192.168.1.17: ICMP echo request, id 61956, se
 1, length 64
01:44:42.471442 IP 192.168.1.17 > 192.168.1.18: ICMP echo reply, id 61956, seq
, length 64
01:44:43.483190 IP 192.168.1.18 > 192.168.1.17: ICMP echo request, id 61956, se
 2, length 64
01:44:43.483215 IP 192.168.1.17 > 192.168.1.18: ICMP echo reply, id 61956, seq
, length 64
```

3.5

Παρατηρούμε πως λαμβάνουμε μήνυμα «No route to host» και δε καταγράφονται πακέτα ARP/ICMP ούτε στο LAN1 αλλά ούτε και στο LAN2.

3.6

Παρατηρούμε πως λαμβάνουμε μήνυμα «No route to host» και δε καταγράφονται πακέτα ARP/ICMP ούτε στο LAN1 αλλά ούτε και στο LAN2.

3.7

Στο πρώτο ping, από το PC1 στο PC3, εφαρμόζοντας τη μάσκα υποδικτύου στο PC1 βλέπουμε πως αυτό ανήκει στο δίκτυο **192.168.1.0**, επομένως όταν πάει να στείλει στο PC3, εφαρμόζει τη δική του μάσκα στο PC3 και βλέπει πως αυτό ανήκει στο υποδίκτυο **192.168.1.16**, το οποίο είναι διαφορετικό από το δικό του. Το ίδιο πράγμα συμβαίνει και στο αντίστροφο ping με ανεστραμμένους τους ρόλους των PC1 και PC3.

3.8

arp -a

```
root@PC:~ # arp -a
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 expires in 1197 seconds [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 permanent [ethernet]
root@PC:~ #
```

R1, PC1

3.9

arp -a

```
root@PC:~ # arp -a
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 permanent [ethernet]
```

PC2

3.10

arp -a

```
root@PC:~ # arp -a
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em1 expires in 1195 seconds [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 732 seconds [ethernet]
root@PC:~ #
```

PC1, PC3, em0, em1

3.11

Παραμένουν μόνο οι MAC διευθύνσεις των καρτών δικτύου του στα LAN1 (em0) και LAN2 (em1).

3.12

tcpdump -i em0 arp or icmp

```
root@PC:~ # tcpdump -i em0 arp or icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on em0, link-type EN10MB (Ethernet), capture size 65535 bytes
02:16:34.892125 ARP, Request who-has 192.168.1.1 tell 192.168.1.14, length 28
02:16:34.892600 ARP, Reply 192.168.1.1 is-at 08:00:27:c2:25:85 (oui Unknown), le
ngth 46
02:16:34.892617 IP 192.168.1.14 > 192.168.1.1: ICMP echo request, id 54020, seq
0, length 64
02:16:34.893176 IP 192.168.1.1 > 192.168.1.14: ICMP echo reply, id 54020, seq 0,
 length 64
02:16:37.968060 ARP, Request who-has 192.168.1.2 tell 192.168.1.14, length 28
02:16:37.968616 ARP, Reply 192.168.1.2 is-at 08:00:27:aa:3e:5a (oui Unknown), le
ngth 46
02:16:37.968635 IP 192.168.1.14 > 192.168.1.2: ICMP echo request, id 54276, seq
0, length 64
02:16:37.969137 IP 192.168.1.2 > 192.168.1.14: ICMP echo reply, id 54276, seq 0,
 length 64
```

3.13

PC1, PC2, em0, em1

Παρατηρούμε πως προστέθηκαν οι εγγραφές για τις MAC διευθύνσεις των PC1 και PC2, τις οποίες το R1 έμαθε μέσω των ARP replies στα ARP request που έστειλε στο προηγούμενο ερώτημα λόγω του ping στα PC1/PC2.

```
root@PC:~ # arp -a
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 1186 seconds [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 expires in 1189 seconds [ethernet]
root@PC:~ #
```

3.14

PC1

Η διεύθυνση του PC1 και η διεύθυνση του R1 στο LAN1. Λόγω του arp request ο PC1 έμαθε τη διεύθυνση του router

```
root@PC:~ # arp -a
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 expires in 1155 seconds [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 permanent [ethernet]
root@PC:~ #
```

PC2

Η διεύθυνση του PC2 και η διεύθυνση του R1 στο LAN1. Λόγω του arp request ο PC2 έμαθε τη διεύθυνση του router

```
root@PC:~ # arp -a
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 expires in 1161 seconds [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 permanent [ethernet]
root@PC:~ #
```

3.15

```
root@PC:~ # tcpdump -i em1 arp or icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on em1, link-type EN10MB (Ethernet), capture size 65535 bytes
02:26:31.336455 ARP, Request who-has 192.168.1.18 tell 192.168.1.17, length 28
02:26:31.336965 ARP, Reply 192.168.1.18 is-at 08:00:27:eb:1f:4c (oui Unknown), l
ength 46
02:26:31.337009 IP 192.168.1.17 > 192.168.1.18: ICMP echo request, id 59140, seq
 0, length 64
02:26:31.337465 IP 192.168.1.18 > 192.168.1.17: ICMP echo reply, id 59140, seq 0
, length 64
02:26:36.264916 ARP, Request who-has 192.168.1.29 tell 192.168.1.17, length 28
02:26:36.265550 ARP, Reply 192.168.1.29 is-at 08:00:27:f2:08:42 (oui Unknown), l
ength 46
02:26:36.265571 IP 192.168.1.17 > 192.168.1.29: ICMP echo request, id 59396, seq
 0, length 64
02:26:36.266049 IP 192.168.1.29 > 192.168.1.17: ICMP echo reply, id 59396, seq 0
, length 64
```

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em1 expires in 1171 seconds [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em1 expires in 1166 seconds [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 570 seconds [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 expires in 573 seconds [ethernet]
root@PC:~ #
```

Έχει πλέον καταχωρημένες όλες τις διευθύνσεις IP

3.16

(PC1) 192.168.1.1 → 08:00:27:C2:25:85

(PC2) 192.168.1.2 → 08:00:27:AA:3E:5A

(PC3) 192.168.1.18 → 08:00:27:EB:1F:4C

(PC4) 192.168.1.29 → 08:00:27:F2:08:42

(R1-LAN1) 192.168.1.14 → 08:00:27:48:1C:7E

(R2-LAN2) 192.168.1.17 → 08:00:27:DA:E8:EC

3.17

Παράγονται μόνο ARP requests ώστε να βρεθεί αν υπάρχει κάποια συσκευή με αυτή την IP.

```
root@PC:~ # tcpdump -i em0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on em0, link-type EN10MB (Ethernet), capture size 65535 bytes
02:31:55.730944 ARP, Request who-has 192.168.1.5 tell 192.168.1.14, length 28
02:31:56.741661 ARP, Request who-has 192.168.1.5 tell 192.168.1.14, length 28
02:31:57.748475 ARP, Request who-has 192.168.1.5 tell 192.168.1.14, length 28
```

3.18

Παρουσιάζεται το παρακάτω, όπου μας ενημερώνει πως είναι ανολοκλήρωτη η αναζήτης της MAC διεύθυνση της συγκεκριμένης IP.

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em1 expires in 780 seconds [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em1 expires in 775 seconds [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 179 seconds [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 expires in 182 seconds [ethernet]
? (192.168.1.5) at (incomplete) on em0 expired [ethernet]
root@PC:~ #
```

3.19

Παρατηρούμε πως αποστέλλεται ένα ARP request ανά ping. Όταν, επιπλέον, φτάσουμε τα 6 αντί να μη λάβουμε καμία απάντηση όπως στα προηγούμενα λαμβάνουμε το μήνυμα « ping: sendto: Host is down ».

Άσκηση 4: Προεπιλεγμένος δρομολογητής

4.1

sysctl net.inet.ip.forwarding=1

4.2

Να προσθέσουμε τη γραμμή « **gateway_enable="YES"** » στο αρχείο /etc/rc.conf του R1

4.3

Προς το παρόν δε βλέπουμε καμία διαφορά, λαμβάνουμε πάλι το μήνυμα « No route to host ».

4.4

Παρατηρούμε πως δεν υπάρχει διαδρομή για το LAN2.

```
root@PC:~ # netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags     Netif Expire
127.0.0.1          link#2             UH          lo0
192.168.1.0/28     link#1             U           em0
192.168.1.1        link#1             UHS         lo0
```

4.5

route add default 192.168.1.14 --> Στο PC1

4.6

Προστέθηκε το default gateway.

```
root@PC:~ # netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags     Netif Expire
default            192.168.1.14       UGS         em0
127.0.0.1          link#2             UH          lo0
192.168.1.0/28     link#1             U           em0
192.168.1.1        link#1             UHS         lo0
```

4.7

Πάλι δεν ανταποκρίνεται, αλλα αυτή τη φορά δεν υπάρχει το μήνυμα « no route to host ».

4.8

Εκτελούμε σε διαφορετικές κονσόλες του R1 τις εντολές « **tcpdump -i em0 icmp** » και « **tcpdump -i em1 icmp** » για να καταγράψουμε ICMP πακέτα στα LAN1 και LAN2 αντίστοιχα. Αυτό που παρατηρούμε είναι πως καταγράφονται κανονικά ICMP Request προς τον προορισμό, χωρίς ωστόσο να καταγράφεται κάποιο Reply, αφού στον PC3 δεν έχει οριστεί προκαθορισμένη πύλη.

4.9

route add default 192.168.1.14 --> Στο PC3

4.10

Πλέον, όχι μόνο έχουμε επιτυχής αποστολή του ICMP Request, αλλά λαμβάνουμε κανονικά και το ICMP Reply. Τα βήματα που συμβαίνουν είναι τα εξής:

1) Το PC1 πάει να στείλει πακέτο στο PC3, ωστόσο βλέπει πως είναι σε διαφορετικά δίκτυα, επομένως το στέλνει στο default gateway, εν προκειμένω στο em0 του R1.

2) Το R1, έχει καταγραφή για τη διεύθυνση IP του παραλήπτη (του PC3) στην em1 του, επομένως στέλνει το πακέτο από αυτή την κάρτα δικτύου απευθείας στο PC3.

3) Το PC3 λαμβάνει το μήνυμα και επειδή προφανώς το PC1 είναι επίσης σε διαφορετικό δίκτυο από αυτό, προωθεί την απάντησή του στο default gateway του, δηλαδή στο em1 του R1.

4) Με τη σειρά του, το R1 ακολουθεί πλέον αντίστροφη πορεία και στέλνει το πακέτο από το em0 απευθείας στο PC1.

4.11

2 βήματα, υπάρχει στη μέση ο R1 ο οποίος μετράει σαν έξτρα βήμα.

```
root@PC:~ # traceroute 192.168.1.18
traceroute to 192.168.1.18 (192.168.1.18), 64 hops max, 40 byte packets
 1  192.168.1.14 (192.168.1.14)  0.318 ms  0.262 ms  0.207 ms
 2  192.168.1.18 (192.168.1.18)  0.445 ms  0.829 ms  0.435 ms
root@PC:~ #
```

4.12

arp -ad

4.13

tcpdump -i em0 -vvv -e

tcpdump -i em1 -vvv -e

4.14

ping -c 1 192.168.1.18

4.15

LAN1

MAC source : 08:00:27:C2:25:85 (PC1)

MAC destination : 08:00:27:48:1C:7E (R1 em0)

IPv4 source : 192.168.1.1 (PC1)

IPv4 destination : 192.168.1.18 (PC3)

4.16

LAN2

MAC source : 08:00:27:DA:E8:EC (R1 em1)

MAC destination : 08:00:27:EB:1F:4C (PC3)

IPv4 source : 192.168.1.1 (PC1)

IPv4 destination : 192.168.1.18 (PC3)

4.17

Οι διευθύνσεις IP μένουν σταθερές, αλλά μεταβάλλονται οι διευθύνσεις MAC οσο το πακέτο προωθείται από δρομολογητή σε δρομολογητή. Από τη στιγμή που η διεύθυνση IP δεν ανήκει στο τοπικό δίκτυο, αυτή προωθείται κατευθείαν στο default gateway ο οποίος είναι υπέυθυνος να βρεί αυτός τη MAC του προορισμού, διατηρώντας ωστόσο τη διεύθυνση IP, γιατί αλλιώς το πακέτο θα χανόταν.

4.18

ssh 192.168.1.18 -l lab

4.19

netstat -an | grep 192.168.1.18

Protocol : TCP

Local port (PC1) : 18403

Remote port (PC3) : 22

```
root@PC:~ # netstat -an | grep 192.168.1.18
tcp4       0      0 192.168.1.1.18403      192.168.1.18.22        ESTABLISHED
```

4.20

netstat -p tcp --> Δεν παρατηρούμε κάτι γιατί ο R1 δεν μπορεί να δει δεδομένα που βρίσκονται στο στρώμα μεταφοράς(transport layer).

Άσκηση 5: Προθέματα δικτύου και δρομολόγηση

5.1

ifconfig em0 192.168.18/29

route add default 192.168.1.17

5.2

arp -ad

5.3

tcpdump -i em0 arp or icmp → R1

5.4

tcpdump -i em0 arp or icmp → PC4

5.5

ping -c 1 192.168.1.X , X = {2, 18, 29}

Τα ping είναι επιτυχή.

```
root@PC:~ # ping -c 1 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=0.686 ms

--- 192.168.1.2 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.686/0.686/0.686/0.000 ms
root@PC:~ # ping -c 1 192.168.1.18
PING 192.168.1.18 (192.168.1.18): 56 data bytes
64 bytes from 192.168.1.18: icmp_seq=0 ttl=63 time=1.034 ms

--- 192.168.1.18 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 1.034/1.034/1.034/0.000 ms
root@PC:~ # ping -c 1 192.168.1.29
PING 192.168.1.29 (192.168.1.29): 56 data bytes
64 bytes from 192.168.1.29: icmp_seq=0 ttl=63 time=0.849 ms

--- 192.168.1.29 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.849/0.849/0.849/0.000 ms
root@PC:~ #
```

5.6

PC1 : PC1, PC2, R1(em0)

```
root@PC:~ # arp -a
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 expires in 1185 seconds [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 permanent [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 expires in 1174 seconds [ethernet]
root@PC:~ #
```

PC2 : PC1, PC2

```
root@PC:~ # arp -a
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 1164 seconds [ethernet]
? (192.168.1.2) at 08:00:27:aa:3e:5a on em0 permanent [ethernet]
root@PC:~ #
```

PC3 : PC3, R1(em1)

```
root@PC:~ # arp -a
? (192.168.1.17) at 08:00:27:da:e8:ec on em0 expires in 1180 seconds [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em0 permanent [ethernet]
root@PC:~ #
```

PC4 : PC4, R1(em1)

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em0 permanent [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em0 expires in 1173 seconds [ethernet]
root@PC:~ #
```

R1 : PC1, PC3, PC4, R1(em0), R2(em1)

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em1 expires in 1185 seconds [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em1 expires in 1178 seconds [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
? (192.168.1.1) at 08:00:27:c2:25:85 on em0 expires in 1178 seconds [ethernet]
root@PC:~ #
```

5.7

i. *ARP, Request who-has 192.168.1.14 tell 192.168.1.1*, όπου το PC1 ρωτάει για τη MAC της IP 192.168.1.14

ii. *ARP, Reply 192.168.1.14 is-at 08:00:27:48:1c:7e*, όπου το R1 υπό τη διεύθυνση 192.168.1.14 απαντάει με τη MAC του στο PC1

iii. *ICMP echo request, IP 192.168.1.1 > 192.168.1.29*, όπου το PC1 στέλνει το ICMP echo request του με τελικό προορισμό τη διεύθυνση 192.168.1.29, αλλά απευθείας στη MAC address του em0 του R1.

iv. *ARP, Request who-has 192.168.1.29 tell 192.168.1.17*, όπου το R1 ρωτάει για τη MAC του PC4

v. *ARP, Reply 192.168.1.29 is-at 08:00:27:f2:08:42*, όπου το PC4 απαντάει στο R1 πληροφορώντας το για τη MAC του

vi. *ICMP echo request, IP 192.168.1.1 > 192.168.1.29*, όπου το R1 προωθεί το πακέτο του PC1 στο PC4

vii. *ICMP echo reply, IP 192.168.1.29 > 192.168.1.1*, όπου το PC4 απαντάει στη διεύθυνση του PC1

viii. *ICMP echo reply, IP 192.168.1.29 > 192.168.1.1*, όπου το R1 προωθεί την απάντηση του PC4 στο PC1

5.8

Εκτελούμε « **arp -ad** » σε κάθε μηχάνημα και στη συνέχεια « tcpdump -e icmp or arp » στα PC3, PC4, ενώ « **tcpdump -e -i em1 icmp or arp** » στο R1.

5.9

Ναι, ήταν επιτυχές. Δεν παρατήρησα κάτι διαφορετκό

```
root@PC:~ # ping -c 1 192.168.1.29
PING 192.168.1.29 (192.168.1.29): 56 data bytes
64 bytes from 192.168.1.29: icmp_seq=0 ttl=64 time=0.820 ms

--- 192.168.1.29 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.820/0.820/0.820/0.000 ms
root@PC:~ #
```

5.10

PC3 → PC3, R1(em1)

```
root@PC:~ # arp -a
? (192.168.1.17) at 08:00:27:da:e8:ec on em0 expires in 1153 seconds [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em0 permanent [ethernet]
root@PC:~ #
```

PC4 → PC3, PC4, R1(em1)

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em0 permanent [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em0 expires in 1161 seconds [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em0 expires in 1161 seconds [ethernet]
root@PC:~ #
```

R1 → PC3, PC4, R1(em1), R1(em0)

```
root@PC:~ # arp -a
? (192.168.1.29) at 08:00:27:f2:08:42 on em1 expires in 1157 seconds [ethernet]
? (192.168.1.17) at 08:00:27:da:e8:ec on em1 permanent [ethernet]
? (192.168.1.18) at 08:00:27:eb:1f:4c on em1 expires in 1157 seconds [ethernet]
? (192.168.1.14) at 08:00:27:48:1c:7e on em0 permanent [ethernet]
root@PC:~ #
```

5.11

Έχουμε την παρακάτω αλληλουχία μηνυμάτων:

i. Το PC3 κάνει broadcast ένα ARP request ρωτώντας τη MAC διεύθυνση της IP 192.168.1.17

ii. Το R1 λαμβάνει το broadcast και απαντάει στο PC3 με τη MAC του(08:27:00:DA:E8:EC)

iii. Το PC3(192.168.1.18) στέλνει στην παραπάνω MAC του R1 το ICMP echo request με προορισμό τη διεύθυνση 192.168.1.29(=IP_{PC4})

iv. ARP Request who-has 192.168.1.29 tell 192.168.1.17 .Το R1 κάνει broadcast ένα ARP Request, όπου ρωτάει για τη MAC της διεύθυνσης 192.168.1.29

v. Προτού λάβει απάντηση από το broadcast, απαντάει στο ICMP Echo request του PC3 με ICMP redirect 192.168.1.29 to host 192.168.1.29.

vi. ARP Reply 192.168.1.29 is-at 08:00:27:F2:08:42. Το R1 μαθαίνει τη MAC του PC4.

vii. Το R1 προωθεί στο PC4 το ICMP Echo request του PC3.

viii. Το PC4 κάνει broadcast ένα ARP Request με σκοπό να μάθει τη MAC της διεύθυνσης 192.168.1.18 για να στείλει το ICMP reply.

ix. Το PC3 απαντάει (χωρίς διαμεσολάβηση) στο ARP request του PC4 ενημερώνοντάς το με τη MAC του

x. 08:00:27:F2:08:42(MAC_{PC4}) > 08:00:27:EB:1F:4C(MAC_{PC3}), 192.168.1.29 > 192.168.1.18, ICMP echo reply. Τέλος, ο PC4 απαντάει στο ICMP echo request του PC3.

5.12

Το PC3 αναζητά του R1 (αφού το PC4 δεν ανήκει στο ίδιο υποδίκτυο). Το PC4 ψάχνει του PC3.

5.13

Γιατί δεν ανήκει στο ίδιο υποδίκτυο. Το PC3 « βλέπει » τις διευθύνσεις 192.168.1.16 - 192.168.1.23 στο LAN2.

5.14

Το κάνει redirect στο PC4.

5.15

Απευθείας, γιατί ο PC3 ανήκει στο υποδίκτυο του PC4.

5.16

Εκτελούμε στα PC3, PC4 « tcpdump -e icmp », ενώ στο R1 « tcpdump -i em1 icmp »

5.17

Αυτό που βλέπουμε να συμβαίνει, είναι πως σε κάθε ICMP echo request μήνυμα, το PC3 στέλνει το ping στην IP_{PC4}, αυτό μεταφέρεται στο R1, το R1 στέλνει στο PC3 Icmp redirect, προωθεί το request στο PC4 και τέλος εκείνο απαντάει κατευθείαν στο PC3. Ο PC3 ουσιαστικά αγνοεί τα redirects, καθώς η προτεινόμενη διεύθυνση παράκαμψης δεν ανήκει στο υποδίκτυο της διεπαφής του στο LAN2, διαφορετικά θα ενημέρωνε κατάλληλα τον πίνακα δρομολόγησής του.

5.18

Διαγράφηκε η προκαθορισμένη διαδρομή.

5.19

netstat -rn

```
root@PC:~ # netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags     Netif Expire
127.0.0.1          link#2             UH          lo0
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

Παρατηρούμε πως στο πρώτο Ping, το ταίριασμα μεγαλύτερου προθέματος στέλνει το πακέτο στο R1, καθώς η μάσκα /29 στη διεύθυνση του PC4 το κάνει να ανήκει στο υποδίκτυο 192.168.1.24/29, επομένως προωθείται στην 192.168.1.17 για να γίνει rerouting. Στη συνέχεια, μετά το πρώτο ping, το PC3 λαμβάνει το icmp redirect, αλλά αυτή τη φορά δε το αγνοεί, αφού αν εφαρμόσει το subnet mask του στη διεύθυνση του PC4 βλέπει πως είναι στο ίδιο LAN οπότε και το προωθεί πλέον κατευθείαν εκεί.

5.21

Έχει προστεθεί πλέον η παρακάτω εγγραφή, η οποία διαφέρει στο ότι η δρομολόγηση σε αυτήν γίνεται μέσω του στρώματος ζεύξης δεδομένων, σε αντίθεση με τα άλλα υποδίκτυα του 192.168.0.0/16 που γίνεται μέσω του R1.

%%

Διαφέρει από τις άλλες εγγραφές γιατί το είναι εγγραφή προς υπολογιστή και όχι προς υποδίκτυο. Έχει δηλαδή prefix = 32 ενώ οι άλλες εγγραφές μικρότερο.

```
root@PC:~ # netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags     Netif Expire
127.0.0.1          link#2             UH          lo0
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
192.168.1.29       192.168.1.29       UGHD        em0 1709819086
```

5.22

Δε μπορεί να επικοινωνήσει με μηχανήματα του LAN1, καθώς δεν υπάρχει προεπιλεγμένη πύλη προκειμένου να κάνει το κατάλληλο routing.

5.23

traceroute 192.168.1.29 → επιλέγεται η διαδρομή κατευθείαν στο PC4 διότι πλέον με τη καινούργια μάσκα υποδικτύου το PC4 ανήκει στο υποδίκτυο του PC3 και έτσι δεν υπάρχει η ανάγκη για ενδιάμεσο δρομολογητή.

Άσκηση 6: Router on a stick

6.1

R1

ifconfig bridge0 create addm em0 addm em1 up

6.2

PC1

ifconfig em0.5 create vlan 5 vlandev em0 inet 192.168.5.1/24 up

ifconfig em0.6 create vlan 6 vlandev em0 inet 192.168.6.1/24 up

6.3

PC2

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

6.4

PC3

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

6.5

PC4

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

6.6

ifconfig em0.5 create vlan 5 vlandev em0 up

ifconfig em0.6 create vlan 6 vlandev em0 up

ifconfig em1.5 create vlan 5 vlandev em1 up

ifconfig em1.6 create vlan 6 vlandev em1 up

6.7

Όχι.

6.8

Όχι.

6.9

Γιατί δεν ανήκουν στο ίδιο vlan.

6.10

Ναι.

6.11

Ναι.

6.12

Όχι, δε μπορούμε να κάνουμε σε καμία.

6.13

PC1 → **sysctl net.inet.ip.forwarding=1**

PC2 → **route change default 192.168.1.1**

6.14

Ναι επιτυγχάνουν.

6.15

(PC1) 192.168.1.1 → 08:00:27:C2:25:85

(PC2) 192.168.1.2 → 08:00:27:AA:3E:5A

(PC3) 192.168.1.18 → 08:00:27:EB:1F:4C

arp -ad

6.16

tcpdump -e

6.17

PC3 → PC1 : arp request (broadcast)

PC1 → PC3 : arp reply

PC3 → PC1 : icmp request

PC1 → PC2 : arp request (broadcast)

PC2 → PC1 : arp reply

PC1 → PC2 : icmp request

PC2 → PC1 : icmp reply

PC1 → PC3 : icmp reply

6.18

Όχι, δεν ειναι επιτυχές.

6.19

Ενώ το PC4 απαντάει κανονικά στα ping request που λαμβάνει, τα στέλνει στο R1, δηλαδή στην προκαθορισμένη του πύλη(192.168.1.17), το οποίο δε ξέρει που να τα προωθήσει ώστε να πάνε στη διεπαφή 192.168.6.18, επομένως απλά τα απορρίπτει.

6.20

```
Destination        Gateway            Flags     Netif Expire
default            192.168.1.17       UGS         em0
127.0.0.1          link#2             UH          lo0
192.168.1.16/28    link#1             U           em0
192.168.1.29       link#1             UHS         lo0
192.168.5.0/24     link#3             U         em0.5
192.168.5.29       link#3             UHS         lo0

Internet6:
Destination                       Gateway                       Flags     Netif
Expire
::/96                             ::1                           URS         lo0
::1                               link#2                        UHS         lo0
::ffff:0.0.0.0/96                 ::1                           URS         lo0
fe80::/10                         ::1                           URS         lo0
fe80::%lo0/64                     link#2                        U           lo0
fe80::1%lo0                       link#2                        UHS         lo0
ff02::/16                         ::1                           URS         lo0
root@PC:~ # route add default 192.168.5.1
add net default: gateway 192.168.5.1
root@PC:~ # route change default 192.168.5.1
change net default: gateway 192.168.5.1
root@PC:~ # route del default 192.168.1.17
del net default: gateway 192.168.1.17
root@PC:~ #
```

Ναι επιτυγχάνει.