*Számítógép hálózatok gyakorlat*

1. feladat

das1 = 1000m  
ds1s2 = 8000m  
Ras1 = 2 Mbit/s   
Rs1s2 = 8 Mbit/s  
L = 1000 byte  
s = 2\*108 m/s  
ta-> s1 -> s2 = ?

ta-> s1 -> s2 = t adasi a + t terjedési a->s1 + t fel. s1 + t sor s1 + t adási s2 + t terj s1->s2

t adási a = L / Ras1 = 8000 bit / (2 \* 108 m/s) = 4ms

t terj a-> s1 = das1 / s = 1000 m / (2 \* 108 m/s) = 5 µs

t sor s1 = L / Rs1s2 = 24 \* 103 bit / (8 \* 108 bit/s) = 3 ms

t adási s1 = L / Rs1s2 = 8 \* 103 bit / (8 \* 108 bit/s) = 1ms

t terj s1->s2 = ds1s2 / s = 8\*103 m / (2 \* 108 m/s) = 40 µs

ta-> s1 -> s2 = 9,045 ms

2. feladat

16 QAM  
N = 16  
k = 16  
b = log2N= log216 = 4  
L = b\*k = 64 bit = 8 byte

3. feladat

Hányszorosára nő a bitsebesség, ha QPSK-ról 64QAM-re váltunk, miközben a szimbólumsebesség a felére csökken?

QPSK -> 64QAM

NQPSK= 4 (mindig 4 !!!!!!!!!!!!!!!!!!!!!!)  
NQAM = 64

bQPSK = log24 = 2  
b64QAM = log264 = 6  
RQAM = 3 RQPSK -> felezés -> 1,5

**4. feladat**

8PSK esetén mekkora a nullától különböző második legkisebb pozitív fázistolás mértéke fokban? **90 fok**

3

2

1

90°

4

0

5

7

6

**5. feladat**

N = 16  
Pb = 10-7 W  
Ts = 10-2 s  
N0 = 4\*10-11 W/Hz

Pz = N0 / Ts = 4 \* 10-9 W  
Pj = log216 \* 10-7 W= 4 \* 10-7 W

SNR = Pj / Pz = 100  
SNR [dB] = 10 log10100= 20dB

**6. feladat**

Mikor kap nyugtát az adóállomás, amennyiben előzőleg, első alkalommal foglaltnak érzékelte a csatornát? (WLAN CDMA/CA)

TSIFS = 10 µs (802.11b)  
Tslot = 20 µs (DSSS)  
Tdata = 1 ms  
CWrand = 0.3

DATA

SIFS

ACK

DIFS

verseny ablak

DATA

SIFS

ACK

Tdelay

A

B

C

TDIFS = TSIFS + 2\* Tslot = 50 µs

Tcont = INT [CWrand \* 31]

Tslot = 9 \* 20 = 180 µs

TD = TDIFS + Tcont +Tdata + TSIFS =   
= 50 + 180 + 1000 + 10 = 1240 µs

**7. feladat**

CSMA/CD (vezetékes, ütközéses detektálás)  
L1=L2=L3= 200 m  
R = 16Mbit/adatátviteli ráta

Minimális kerethossz, hogy ütközést tudjunk detektálni.

*Résidő:* T = 2L / C = 2 \* 600 / (3\* 108) = 6 µs

Minimális kerethossz: dmin = R\*T = 1,6 \* 106 \* 6 \* 10-6 = 96bit = 12 byte

**8. feladat**

1)  
P (R=3 / K=4 ) = ? **1/16**  
R = RAND INT [0, 2K-1] = RAND INT [0,15]

2)  
TD = (3 \* 512 bit) / 107 = 153,6 µs

3)  
TMAX = 52,4 ms

**9. feladat**

1) Bellman-Ford

Állapotvektor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| B 1 | C 2 | D 3 | E 4 | F 1 | ∞ |

A:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A 1 | C 3 | D 2 | E 4 | H 4 |

B-től érkezik:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| B 1 | C 2 | D 3 | E 4 | F 1 | H 5 | ∞ |

A:

2) Dijkstra:

10

4

2

2

8

2

min(D->B) = ?

4

1

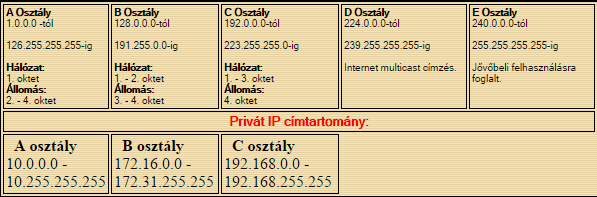
4

Aki akarja csinálja meg, egyszerű algel :D tessék itt egy táblázat is ;)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**10. feladat IPv4**

193.224.128.1 -> C osztály  
147.63.72.11 -> B osztály  
89.123.224.110 -> A osztály



**11. feladat**

193.224.130.172/27

27

193.224.130.10101100 -> 32-27=5 (utolsó 5bit maszkolva)  
255.255.255.11100000  
193.224.130.10100000 (Bc: 193.224.130.10111111)

**Hálózatcíme:** 83.79.60.0  
**Broadcast:** 83.79.63.255

**Hálózatcíme:** 193.224.130.160  
**Broadcast:** 193.224.130.191  
(Bc: csupa 0 helyett csupa 1 a vége)

&

83.79.60.11/22  
83.79.00111100.00001011

**12. feladat**

158.230.128.0/20  
IP-cím szám = 232-20 – 2 = 4094 ( -2 : hálózatcím+BC)  
4096 = 1111\_1111\_1111 -> Broadcast: 158.230. 10001111.11111111 = 153.231.143.255   
Tartomány:  
158.230.10000000.0 => 158.230.128.1 – 158.230.143.254 (legalsó és legfelső nem számít bele)

197.12.7.160/28  
IP-cím szám = 232-28 – 2 = 14   
Tartomány: 192.12.7.161 – 192.12.7.174

**13. feladat**

193.6.128.64/26   
193.6.128.01000000

26 => 28 lesz a maszkolás!!!

01  
10  
11

*193.6.128.64/28  
193.6.128.80/28  
193.6.128.96/28  
193.6.128.112/28*

152.66.192.0/22

*152.66.192.0/25 152.66.194.0/25  
152.66.192.128/25 152.66.194.128/25  
152.66.193.0/25 152.66.195.0/25  
152.66.193.128/25 152.66.195.128/25*

**14. feladat**

Az utolsó 4 bit a gépek címzése, mert legalább 10 géphez 3 bit nem elég!!!!

195.223.12.128/26  
195.223.12.10000000

*195.223.12.128/28  
195.223.12.144/28  
195.223.12.160/28  
195.223.12.176/28*

**15. feladat**

Vonja össze a lehető legnagyobb mértékben a következő hálózatokat:

10.1.0.0/23, 10.1.2.0/25, 10.1.2.128/25, 10.1.3.0./24, 10.1.4.0/24, 10.1.5.0/24

Egyformákat meghagyod az elejéről -> maszk balra tolódik eggyel.

|  |  |
| --- | --- |
| 10.1.00000101.0/24 | 10.1.0000100.0/23 **10.1.4.0/23** |
| 10.1.00000100.0/24 |
| 10.1.00000010.0/25 | 10.1.00000010.0/24  10.1.2.0/24 |
| 10.1.00000010.1/25 |
| 10.1.00000010.0/24 | 10.1.00000010.0/23  10.1.2.0/23 |
| 10.1.00000011.0/24 |
| 10.1.00000010.0/23 | 10.1.00000000.0/22 **10.1.0.0/22** |
| 10.1.00000000.0/23 |

**16. feladat**

Privát IP cím tartományokat lásd 10. feladat [képe](#PrivátIPcím) (CTRL + kattintás a szövegre és ott vagy ;) ).

10.0.0.0/8

172.16.0.0/16

192.168.0.0/24

**17. feladat**

84.2.36.102/26   
Hálózat címe: 84.2.36.64 (vége /26 után csupa nulla)  
Broadcast: 84.2.36.127 (vége /26 után csupa egy)  
Router legkisebb kiosztható => 84.2.36.65  
Többi gép: 84.2.36.66-126 kivéve 102!

**18. feladat**

84.225.252.88/23  
Hálózat címe: 84.225.252.0 (vége /26 után csupa nulla)  
Broadcast: 84.225.253.255 (vége /26 után csupa egy)  
Router legnagyobb kiosztható => 84.225.252.254

**19. feladat**

Mekkora az IP fejrész mérete, ha az Internet Header Length mező értéke 6?

Internet Header Length mező értéke 6 -> 6\*4 = 24

**20. feladat**

Hány byte opció lehet a datagramban, ha az Internet Header Length mező értéke 7?  
5-8 byte

**21. feladat**

Hány byte helykitöltésre lehet szükség akkor, ha az Internet Header Length mező értéke 5, illetve akkor ha 8?  
mező értéke 5 -> 0  
mező értéke (6,7) 8 -> 0,1,2,3

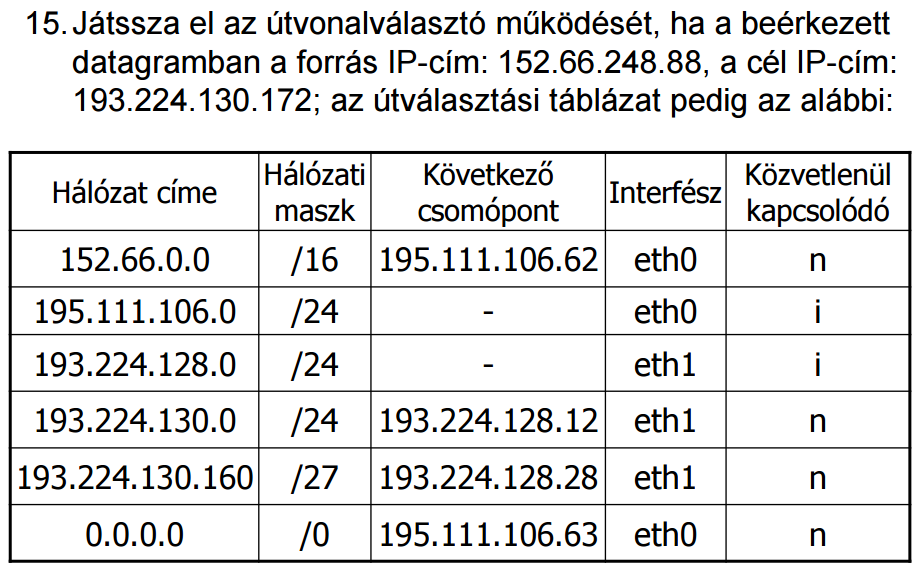
**23. feladat**

Az eredeti datagram adatmezőjében milyen pozícióban kezdődik annak a datagramnak az adatmezője, amelyben a Fragment Offset mező értéke 90?  
8\*90 = 720-tól

**24. feladat**

Mi a Time to Live mező lehetséges legnagyobb értéke?  
8bit -> 255

**25. feladat**



cél IP-címet kell figyelni (hálózati maszk -> cél IP-címe a maszk után csupa 0, ha egyezik a hálózat címével akkor jó )  
4., 5., 6. sor lehetséges útvonalválasztó -> leghosszabb kell (?) -> 5.sor

**26. feladat**

IHL = 6 => Header 24 byte

3 töredék (472+header, 472+header, 56+header)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | IHL | TL | ID | Flag | | | FO |
| 0 | MF | DF |
| 1 | 6 | 496 | 0x5fc3 | 0 | 1 | 0 | 0 |
| 2 | 6 | 496 | 0x5fc3 | 0 | 1 | 0 | 59 |
| 3 | 6 | 80 | 0x5fc3 | 0 | 0 | 0 | 118 |

**27. feladat**

Vond össze: 192.168.32.0/22, 192.168.36.0/23, 192.168.38.0/23, 192.168.40.0/21, 192.168.64.0/20

|  |  |
| --- | --- |
| 192.168.00100100.0/23 | 192.168.00100100.0  192.168.36.0/22 |
| 192.168.00100110.0/23 |
| 192.168.00100100.0/22 | 192.168.00100000.0  192.168.32.0/21 |
| 192.168.00100000.0/22 |
| 192.168.00100000.0/21 | 192.168.00100000.0  **192.168.32.0/20** |
| 192.168.00101000.0/21 |
| 192.168.00100000.0/20 | X |
| 192.168.01000000.0/20 |

**ARP Feladatok**

1. **feladat**

ARP Request

1. **feladat**

ARP reply

1. **feladat**

ARP Probe

1. **feladat**

ARP Announcment

1. **feladat**

ARP Reply

**29. feladat IPv6**

Címek száma számológép nélkül:  
2128= 28 \* (210)12 ≈ 256 \* 1036

**30. feladat**

6\*1023 (Avogadro állandó :O MAGIC…)  
(Miért becsapós? … nem érdekel)

**31. feladat**

0-k a négyes csoport elejéről eltűnnek.

|  |  |
| --- | --- |
| 2000:0000:0000:0002:00C0:C000:000C:000A | 2000::2:C0:C000:C:A |
| 0000:0000:ABCD:EF01:00FA:A000:000B:0000 | ::ABCD:EF01:A000:B:0 |
| 0000:ABBA:BABA:0000:0000:CACA:DADA:0000 | 0:ABBA:BABA::CACA:DADA:0 |
| FE00:0000:0000:0B0B:BAB0:0000:0000:CABA | FE00::B0B:BAB0:0:0:CABA |

**32. feladat**

|  |  |  |
| --- | --- | --- |
|  | /56 | /64 |

264-56 = 28 = 256

**33. feladat**

/48-as prefix-szel ugyanaz  
264-48 = 216 = 65536

**34. feladat**

|  |  |  |  |
| --- | --- | --- | --- |
|  | /56 | /60 | /64 |

2bit kell 3 hálózathoz, 4 bites a határ  
64-4 = 60 -> /60 prefix

**35. feladat**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | /48 | 6bit | 10bit (1000db) | /64 |

26 = 64 db hálózat

/54

/48

2002:A0B0:8010::0000:0000:0000:0000

0000 00  
0000 01  
0000 10  
…  
1111 11

2002:A0B0:8010::/54  
2002:A0B0:8010:0400::/54  
2002:A0B0:8010:0800::/54  
2002:A0B0:8010:0C00::/54  
…  
2002:A0B0:8010:FC00::/54

**36-37. feladat**

00:C0:C1:0B:0C:1B, 88:00:00:CC:00:EE

Felezed -> FF:FE közé; +2 az első taghoz  
37-eshez -> 4-es csoportokat csinálsz a kettes tagokból -> vezér nullák HUSSSS!!!

|  |  |  |
| --- | --- | --- |
|  | 36. feladat | FE80::/64 (37.feladat) |
| 00:C0:C1:0B:0C:1B | 02:C0:C1:FF:FE:0C:1B | FE80::2C0:C1FF:FE08:C1B |
| 88:00:00:CC:00:EE | 8A:00:00:FF:FE:CC:00:EE | FE80::8A00:FF:FECC:EE |

ZH utáni feladatok

**1. feladat(TCP/IP feladatok)**

A 193.224.130.172 IPv4 címhez írja fel az alábbiakat:

* *IPv4-Compatible IPv6 Address* (ez már nem használatos!)

::193.224.130.174

* az *IPv4-Mapped IPv6 Address*

*::FFFF:192.224.130.172*

**2. feladat**

A 152.66.148.88 IPv4 címhez írja fel az *IPv4-Embedded IPv6 Address*t a *NAT64 Well-Known Prefix*-szel!

64:FF9B::152.66.148.88

**3. feladat**

Az alábbi prefixek esetén válasszon alkalmas hálózat specifikus prefixet, és a 16.32.64.128 IPv4 címhez írja fel *IPv4-Embedded IPv6 Address*t!

* 2A00:1878::/32
* 2001:738:2001::/48
* 2001:738:2C01:8001::/64

**Órai megoldás:**

Dec: 16.32.64.128 => Hex: 10.20.40.80

**IPv6 címtartomány => IPv4-Embedded IPv6 Address**

2A00:1878::/32 => 2A00:1878:0:1020:0040:8000::/48

2001:738:2001::/48 => 2001:738:2001::16.32.64.128

Választott prefix: 2001:738:2001::/96

Magyarázat:

IPv4-Embedded Ipv6 címek (Ipv6-transition 22.dia, ábra 23.dia)

* IPv6 hálózati prefix mérete szigorúan csak 32,40,48,56,64,96 lehet
* IPv6 címben 64-71 biteknek 0-nak kell lenniük.

**Megoldás 23. dia szerint:**

2A00:1878::/32 => 2A00:1878:1020:4080::/72

2001:738:2C01::/48 => 2001:738:2C01:1020:0040:8000::/88

2001:738:2C01:8001::/64 => 2001:738:2C01:8001:0010:2040:8000::/104

**4. feladat**

Mit tud a 2002:5EF8:964D::/64 prefixről?

6 to 4 prefix: 2002:5EF8:964D::/64 mert: 5EF8:964D == 94.248.150.77

**5. feladat**

A 128.130.12.10 IPv4 címhez írjon fel

* egy /48 méretű *6to4* prefixet

2002:8082:C0A::/48

* egy a 00:C0:AA:BB:CC:DD MAC című gép esetén lehetséges *6to4*  
  *IPv6 cím*et!

2002:8082:C0A::2C0:AAFF:FEBB:CCDD

**6. feladat**

Mi lenne a válasza, ha valaki a 192.168.1.1 címhez tartozó *6to4 cím*et kérdezné?

Nem lehet képezni, mert privát IP cím

**7. feladat**

10.0.0.0 - 10.255.255.255

172.16.0.0 – 172.31.255.255

192.168.0.0 – 192.168.255.255

**8. feladat**

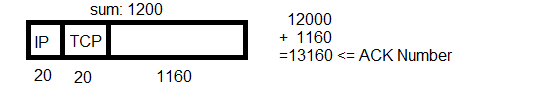
Egy ICMP hibaüzenetben az üzenetet kiváltó IP datagram fejrészén túl az adatrészből hány oktett szerepel? Hogyan található meg, hogy milyen alkalmazáshoz kell a hibaüzenetnek megérkeznie?

ICMP – 64 bit = 8 oktett

Traceroute

**9. feladat**

Egy IP datagramban: Total Length=1200, IHL=5, a benne található TCP szegmensben Data Offset=5, Sequence Number=12000. A szegmensre adott nyugtában mennyi lesz az Acknowledgement Number mező értéke?



**10. feladat**

Az „A” állomás a „B”-től egy olyan TCP szegmenst kapott, amelyben Window=1000, Acknowledgement Number=8000. Ezután „A” elküldött egy szegmenst, melyben Sequence Number=8000, és az adat oktettek száma 800 volt. Ha ezt „B” megkapja, mekkora lehet a válaszában a Window legkisebb értéke?

Window = 1000 800-at elküldtem

ACK Number = 8000 küldhet még 200-at

=> 200-nál kisebbre nem csökkenthető

**11. feladat(Forgalomszabályozás, TCP)**

Stop-and-wait protokollt alkalmazunk, ahol a linken egyszerre egy nyugtázatlan csomag lehet. A csomaghossz 2500 byte, a távolság a két végpont között 6000 km, az adatátviteli sebesség pedig 1 Mbit/s? A nyugtaüzenet adási id őtartama elhanyagolható, optikai szálat használunk.

Mekkora lesz a kihasználtság?

R= 1 Mbit/s

csomag = 2500 byte = 20000 bit

L = 6000 km = 6\*106m

jelterjedés = 2\*108 m/s

adási idő = csomag / adatátviteli sebesség = 20000 / 106= 0.02 s

RTT = 2 \* L / jelterjedési idő = 1.2 \* 107  / 2\*108 = 0.06 s

W hatásfok = (adási idő) / (RTT+adási idő) = 0.02/ (0.06 + 0.02) = 0.25 => csak 25%-ban van kihasználva a kapcsolat.

**12. feladat**

Az “A” és „B” végpont közötti kommunikáció során „A” végpont utolsóként elküldött TCP PDU-jában a sorszám (sequence number) 3970, a hasznos adatrész 70 byte. „B” válaszként küldött TCP PDU-jában az ACK-szám 3740.

Hány byte-nyi adatot küldhet még „A” a következő nyugta megérkezéséig, ha a B vételi ablakmérete 500?

A -> B Seq= 3970 Data = 70

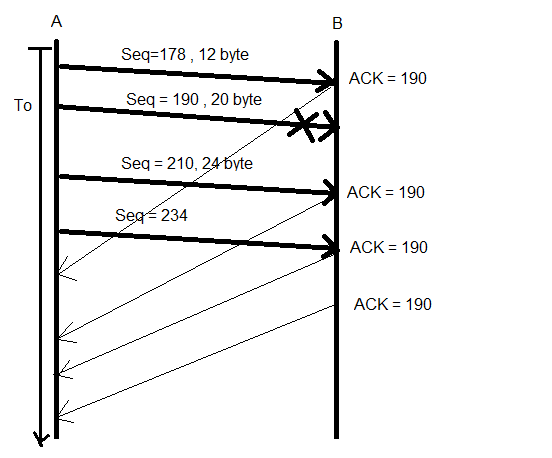
B -> A ACK = 3740

A következő Seq = 4040 kezdetű. 4040-ig mindent elküldött, de B-hez csak 3740-ig érkeztek meg. Úton van 300 byte. Tehát 200 byte adatot küldhet még A.

**13. feladat(Fast retransmit)**

Az “A” és „B” végpont közötti kommunikáció során „A” végpont által küldött utolsó 4 szegmens TCP PDU-jában a sorszám (sequence number) sorban 178, 190, 210, 234. A 178-as sorszámú szegmens lenyugtázása után „B” válaszként küldött 3 szegmens TCP PDU-jában az ACK-szám 190, 190, 190.

Erre válaszként az „A” terminal milyen szekvenciájú és hány bájtos szegmenst fog küldeni fast retransmit esetén, amennyiben nem járt még le a time-out idő?

Fast restransmit => vételi oldalon érzékelik, hogy hiányos csomag érkezett, ezt onnan tudjuk, hogy 3 azonos nyugtát kapunk küldő oldalon(A) fogadótól(B).

**Válasz:**

A terminál Seq=190 20 byte-os szegmenst küldi újra.

To = Timeout

**13. feladat(AIMD)**

Számolja ki a pillanatnyi adatsebességet Mbit/s-ban TCP átvitel során egy csomagvesztés után, amennyiben AIMD  
torlódásvezérlési eljárást használunk és a körülfordulási id ő 200 ms, a csomagvesztési id őpont el őtti torlódási ablak (congestion window) pedig 200 kByte volt!

AIMD => Additive Increase Multiplicative Decrease

RTT = 200 ms RAIMD1= LCW / RTT = 8\*106 bit/s = 8Mbit/s

CW = 200 kByte RAIMD2= RAIMD1 /2 = 4 Mbit/s

**14. feladat (TCP-IP socket)**

Képzeljünk el egy DNS kiszolgálót, ami UDP fölött kéréseket fogad és válaszokat küld vissza. Működése során ilyen socket kezelő függvényeket hív meg, és milyen sorrendben?

1. socket
2. bind
3. recvfrom
4. sendto
5. close

**15. feladat**

A fenti DNS kiszolgáló kliense egy lekérdezés érdekében milyen függvényeket hív meg, és milyen sorrendben?

1. socket
2. bind
3. sendto
4. recvfrom
5. close

**16. feladat**

Nagyobb mennyiségű adatot szeretnénk letölteni TCP használatával. A szerver programunk milyen socket kezelő függvényeket használ, és milyen sorrendben?

1. socket
2. bind
3. listen
4. accept
5. write/send
6. close

**17. feladat**

A fenti feladathoz tartozó kliens milyen socket kezelő függvényeket használ, és milyen sorrendben?

Mi a logikai hiba az alábbi programrészletben?

if ( (s=socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 )  
{

perror("socket()");

if ( errno == NOMEM )

fprintf(stderr, "Out of memory! \n");

}

Kliens:

1. socket
2. bind
3. connect
4. read/recv
5. close

**18. feladat**

Javítsa ki a fenti hibát!

perror állítja az errno értékét, előtte el kell menteni.