16. Számítógépes hálózatok és Internet eszközök

Hálózatok modelljei

TCP/IP modell

Transmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol. Röviden TCP/IP. A TCP/IP modell 1982-ben lett az amerikai hadászati célú számítógépes hálózatok standardja. 1985-től népszerűsítették kereskedelmi használatra.

Négy réteget különböztet meg, ezek fentről lefelé a következők:

4	Alkalmazási réteg
3	Szállítói réteg
2	Hálózati réteg
1	Kapcsolati réteg

OSI modell

Az **O**pen **S**ystems **I**nterconnection Reference Model, magyarul a *Nyílt rendszerek összekapcsolása referenciamodellje* (OSI-modell vagy OSI-referenciamodell) egy rétegekbe szervezett rendszer absztrakt leírása, amely a számítógépek kommunikációjához szükséges hálózati protokollt határozza meg, s amelyet az *Open Systems Interconnection* javaslatban foglalt össze.

Az OSI modellje a különböző protokollok által nyújtott funkciókat egymásra épülő rétegekbe sorolja. Minden réteg csak és kizárólag az alsóbb rétegek által nyújtott funkciókra támaszkodhat, és az általa megvalósított funkciókat pedig csak felette lévő réteg számára biztosíthat.

Hét réteget különböztet meg, ezek fentről lefelé a következők:

7	**Alkalmazási réteg** Alkalmazás szint hálózati eljárások	adat
6	**Megjelenítési réteg** Adat megjelenítés és kódolás/dekódolás	adat
5	**Munkamenet (viszony) réteg** Csomópontok közötti kommunikáció	adat
4	**Szállítási réteg** Végpontok közötti kapcsolat, megbízhatóság	szegmensek
3	**Hálózati réteg** Útvonalválasztás és IP (logikai címzés)	csomagok
2	**Adatkapcsolati réteg** MAC és LLC (fizikai címzés)	keretek
1	**Fizikai réteg** Médium, jelzések, bináris átvitel	bitek

1. Fizikai réteg

A fizikai réteg feladata a bitek továbbítása a kommunikációs csatornán keresztül:

- a ***korrekt bit átvitel*** biztosítása,
- a ***kapcsolat kezelése***, és
- az ***átvitelhez szükséges idő*** *és* ***egyéb részletek tisztázása***

Ez a réteg határoz meg minden, az eszközökkel kapcsolatos fizikai és elektromos specifikációt, beleértve az

- *érintkezők kiosztását*, a
- használatos *feszültség szinteket*, és a
- *kábel specifikációkat.*

Adatátvitel

Vezetékes

Adatátvitel vezeték esetén valamilyen fizikai jellemző változtatásával lehetséges (pl.: feszültség, áramerősség). Ezt egy $g(t)$ periodikus függvénnyel jellemezhetjük. A 19. század elején Jean-Baptiste Fourier francia matematikus bebizonyította, hogy bármely T periódusidejű, periodikus $g(t)$ függvény előállítható szinuszos és koszinuszos tagok (általában végtelen) összegeként:

$$g(t) = \frac{1}{2}c + \sum_{n=1}^{\infty} a_n sin(2\pi\ n\ ft) + \sum_{n=1}^{\infty} b_n cos(2\pi\ n\ ft) \tag{1}$$

ahol $f = \frac{1}{T}$ az alapfrekvencia, a_n és b_n pedig az n-edik harmonikus tag (azaz felharmonikus) szinuszos, illetve koszinuszos amplitúdója. Ezt a felbontást Fourier-sornak nevezzük. A Fourier-sor alapján az eredeti függvény visszaállítható, azaz a T periódusidő és az amplitúdók ismeretében az eredeti időfüggvény meghatározható a (1) összeg alapján. Fontos megjegyezni, hogy a fenti összefüggés feltételezi, hogy a teljes jelalak folyamatosan ismétlődik.

Ahhoz, hogy az a_n és b_n valamint c értékét megkaphassuk, azaz, hogy megtudjuk, hogy a kívánt jelalak előállításához az egyes felharmonikusokból mekkora amplitúdóra van szükség, a fenti képletből kiindulva eljuthatunk a következő három egyenletet kapjuk.

$$a_n = \frac{2}{T}\int_0^T g(t)sin(2\pi nft)dt, \quad b_n = \frac{2}{T}\int_0^T g(t)cos(2\pi nft)dt, \quad c = \frac{2}{T}\int_0^T g(t)dt$$

Vezeték nélküli

Vezeték nélküli adatátvitelre sok helyen használnak elektormágneses hullámokat.

- Rádió
- Mikrohullám
- Infravörös
- Látható
- Ultraibolya
- Röntgen sugarak
- Gamma sugarak

A hullámoknak van frekvenciája és hullámhossza.

- Frekvencia: A hullám másodpercenkénti rezgésszáma.
  - Jele: f,
  - Mértékegysége: Hz (Hertz)
- Hullámhossz: két egymást követő hullámcsúcs (v. hullámvölgy) közötti távolság. Jele: λ (*lambda*)

$$\lambda f = c$$

ahol c a fénysebesség, azaz az elektromágneses hullámok terjedési sebessége vákuumban.

Szimbólumok

Digitális hálózatokat az adatátviteli sebességükkel: az időegység alatt átvitt bitek számával jellemezhetjük. Ezt célszerű *bit/s*-ban mérni. Az átvitelt jellemezhetjük a felhasznált jel értékében egy másodperc alatt bekövetkezett változások számával is, amit jelzési sebességnek, vagy közismert néven **baud**-nak nevezünk.

$$1 \text{ baud} = log_2 P[bit/s]$$

Bitek helyett szimbólumokat küldünk át, ahol P a kódolásban használt jelszintek száma.
(Pl. 4 szimbólum: A - 00, B - 01, C - 10, D - 11)

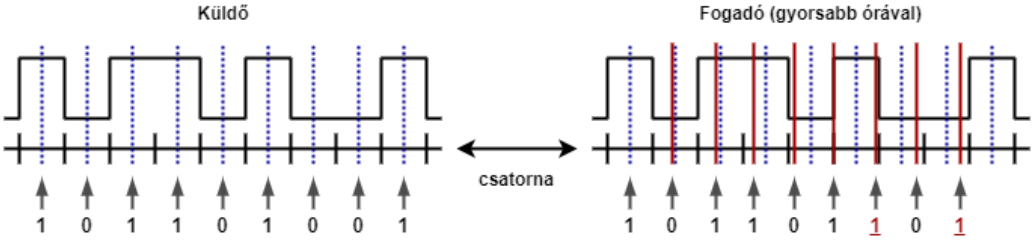
- Baud: szimbólum/másodperc
- Adatráta: bit/másodperc

Például olyan átvitelnél ahol ezt kétállapotú jelekkel valósítjuk meg, ott a *baud* és a bit/s azonos számértéket adnak, de ha a jelet négy szint felhasználásával visszük át, ott a *baud* számértéke már csak fele a bit/s-ban megadott valós adatátviteli sebességnek. Ezért mindig gondosan, ne egymás szinonimájaként használjuk a baud és bit/s mértékegységeket!

Szinkronizáció

Kérdés: Mikor kell szignálokat mérni, illetve mikor kezdődik egy szimbólum?
Ehhez szinkronizációra van szükséges a felek között.

- *Explicit órajel*: Párhuzamos átviteli csatornák használata, szinkronizált adatok, rövid átvitel esetén alkalmas.
- *Kritikus időpontok*: Szinkronizáljunk például egy szimbólum vagy blokk kezdetén, a kritikus időpontokon kívül szabadon futnak az órák, feltesszük, hogy az órák rövid ideig szinkronban futnak.
- *Szimbólum kódok*: Önütemező jel–külön órajel szinkronizáció nélkül dekódolható jel, a szignál tartalmazza a szinkronizáláshoz szükséges információt.



1. ábra. Szinkronizáció szükségessége

Átviteli közegek

Vezetékes

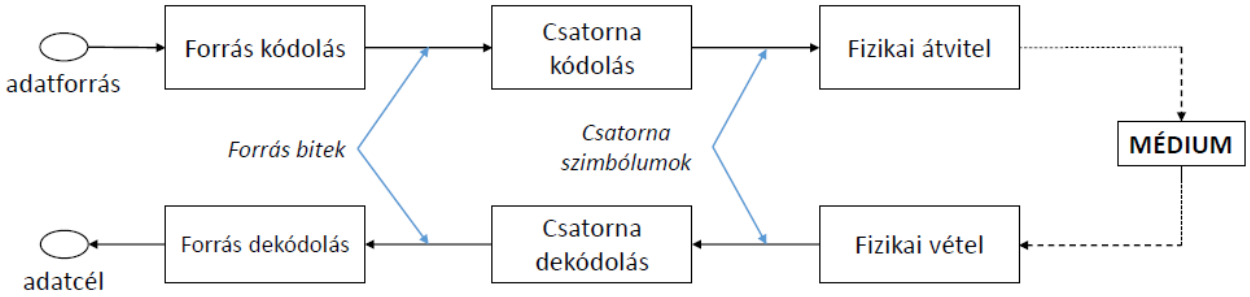
- *Mágneses adathordozók*
  - Nagy sávszélesség
  - Nagy késletetés
- *Sodort érpár*
  - Főként távbeszélőrendszerekben használatos
  - Dupla rézhuzal
  - Analóg és digitális jelátvitel
- *Koaxiális kábel*
  - Nagyobb sebesség és távolság érhető el, mint a sodorttal
  - Analóg és digitális jelátvitel
- *Fényvezető szálak (üvegszál)*
  - Fényforrás, átviteli közeg és detektor
  - Fényimpulzus 1-es bit, nincs fényimpulzus 0-s bit

Vezeték nélküli

- *Rádiófrekvenciás* átvitel
  - Egyszerűen előállíthatóak
  - Nagy távolságú átvitel (jel erősítés lehetséges)
  - Kültéri és beltéri alkalmazhatóság
  - Időjárás befolyásolhatja
  - Lehallgathatóság
  - Frekvenciakiosztás állami hatáskör
- *Mikrohullámú* átvitel
  - Egyenes vonal mentén terjed
  - Elhalkulás problémája
  - Olcsó
- *Műholdas átvitel*
  - 30.000 km felett keringő műholdak
  - Eltérő frekvenciájú továbbítás, mint fogadás
  - Időjárás függő
  - Lehallgatható
  - Nagy jel késés (a nagy távolság miatt)
- *Infravörös* és *milliméteres hullámú* átvitel
  - Kistávolságú átvitel esetén
  - Szilárd tárgyakon nem hatol át
- *Látható fényhullámú* átvitel
  - Lézerforrás + fényérzékelő
  - Nagy sávszélesség,
  - Nem engedélyköteles
  - Biztonságos (nehezen lehallgatható)
  - Olcsó
  - Időjárás erősen befolyásolhatja
- *Bluetooth*
  - Kis hatótávolság (méterek),
  - Adatcseréhez használható,
  - Alacsony energiaszükséglet

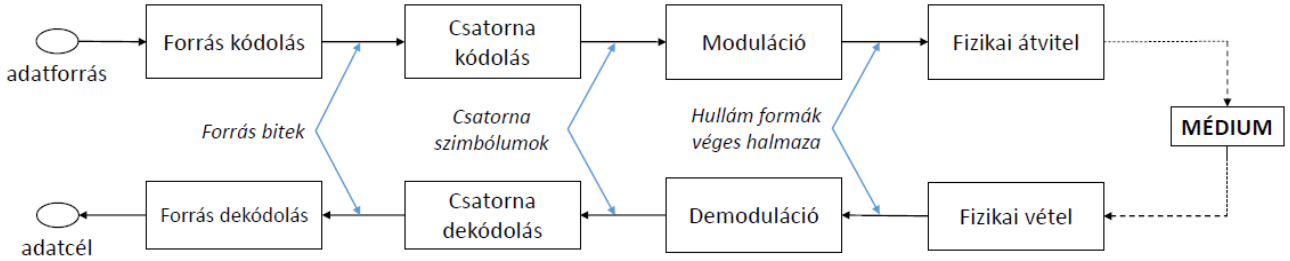
Jelátvitel

- **Alapsáv**: A digitális jel direkt árammá vagy feszültséggé alakul. A jel minden frekvencián átvitelre kerül. Átviteli korlátok



2. ábra. Digitális alapsávú átvitel struktúrája

- **Szélessáv**: Széles frekvencia tartományban történik az átvitel.



3. ábra. Digitális szélessávú átvitel struktúrája

Modulációs eljárást alkalmazunk akkor, mikor a rendelkezésünkre álló átviteli közegen az átvinni kívánt információt hordozó jel nem képes, vagy csak nagy csillapítással, torzulással képes eljuttatni az egyik pontból egy másikba. Ekkor választunk egy olyan jelet (*vivőjelet*), amelyet a rendelkezésre álló átviteli közeg az elvárásainknak megfelelően képes továbbítani.

A modulációs eljárásokat megkülönböztetjük aszerint, hogy a vivő jel mely jellemzője fogja az információt eljuttatni egyik ponttól a másikig. Az információ továbbításához a vivő jel valamely jellemzőjét (esetleg jellemzőit) az információt hordozó jellel (*moduláló jellel*) megváltoztatjuk. A fenti eljárást **modulációnak** nevezzük. A modulált jelet (vivőjel és moduláló jel segítségével előállított jel) eljuttatjuk az átviteli közegen keresztül a célpontig, majd leválasztjuk az információt hordozó jelet. A leválasztást nevezzük **demoduláció**nak.

Modulációk

Egy szinuszos rezgés ábrázolása T periódus idejű függvényre

$$g(t) = Asin(2\pi ft + \varphi),\ ahol\ A \text{ az amplitúdó}, f = \frac{1}{T} \text{ a frekvencia és } \varphi \text{ a fáziseltolás.}$$

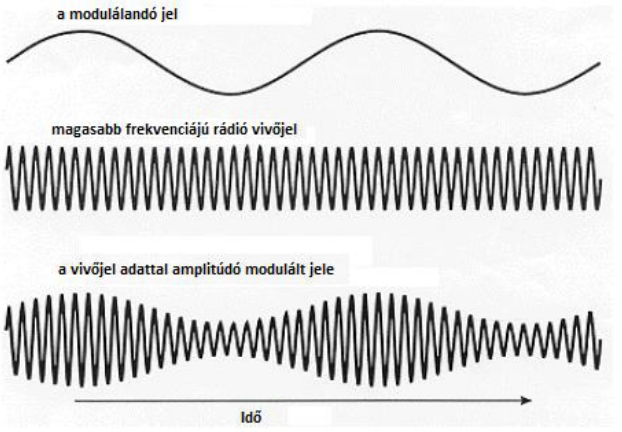
A jel modulálására az alábbi lehetőségeket használhatjuk.

Amplitúdó moduláció

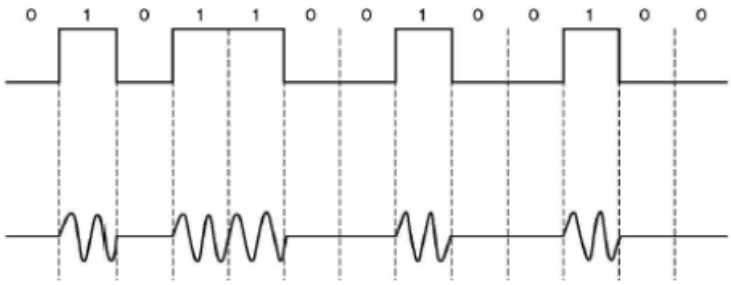
Az $s(t)$ szignált a szinuszgörbe amplitúdójaként kódoljuk, azaz:

$$f_A(t) = s(t) \cdot sin(2\pi ft + \varphi)$$

- Analóg szignál: amplitúdómoduláció
- Digitális szignál: amplitúdó keying
  (szignál erőssége egy diszkrét halmaz értékeinek megfelelően változik)



4. ábra. Amplitúdó moduláció



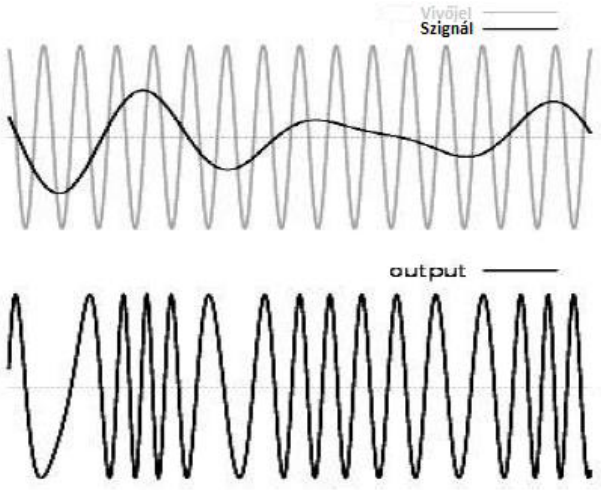
5. ábra. Amplitúdó keying

Frekvencia moduláció

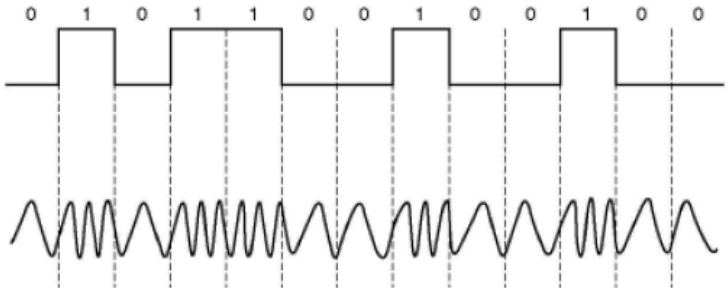
Az $s(t)$ szignált a szinuszgörbe frekvenciájában kódoljuk, azaz:

$$f_F(t) = a \cdot sin(2\pi s(t)t + \varphi)$$

- Analóg szignál: frekvenciamoduláció
- Digitális szignál: frekvencia-eltolás keying
  (például egy diszkrét halmaz szimbólumaihoz különböző frekvenciák hozzárendelésével)



6. ábra. Frekvencia moduláció



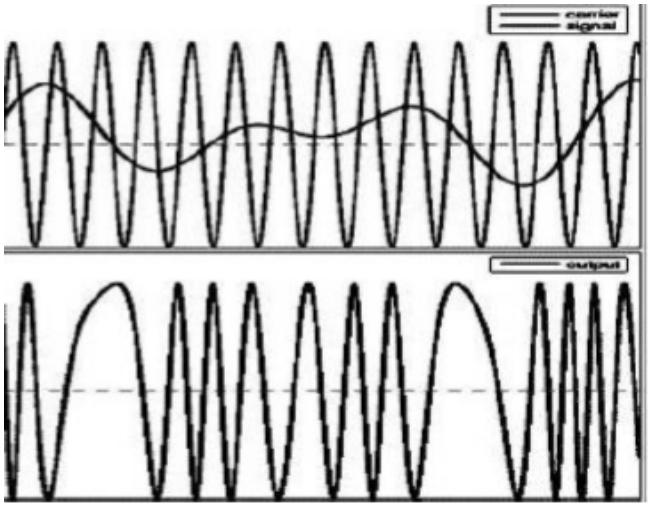
7. ábra. Frekvencia keying

Fázis moduláció

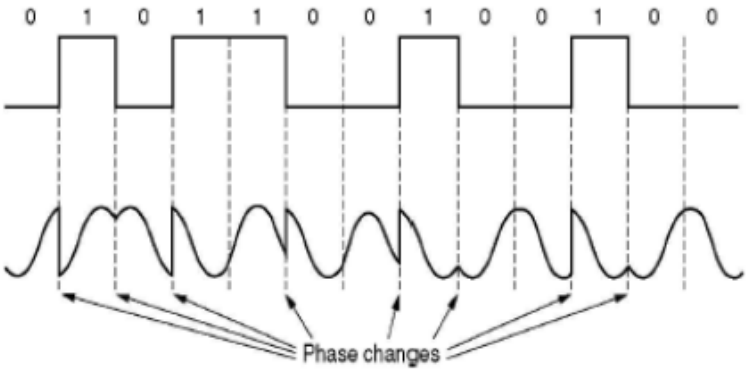
Az $s(t)$ szignált a szinuszgörbe fázisában kódoljuk, azaz:

$$f_P(t) = a \cdot sin(2\pi ft + s(t))$$

- Analóg szignál: fázis moduláció (nem igazán használják)
- Digitális szignál: fázis-eltolás keying
  (például egy diszkrét halmaz szimbólumaihoz különböző fázisok hozzárendelésével)



8. ábra. Fázis moduláció



9. ábra. Fázis-eltolás keying

- Digitális és analóg jelek összehasonlítása:

  - Digitális átvitel: Diszkrét szignálok véges halmazát használja.
    például: feszültség vagy áramerősség értékek.

  - Analóg átvitel: Szignálok folytonos halmazát használja.
    például: feszültség vagy áramerősség a vezetékben.

Digitális esetében lehetőség van a vételpontosság helyreállítására illetve az eredeti jel helyreállítására, míg az analógnál a fellépő hibák önmagukat erősíthetik.

2. Adatkapcsolati réteg

Az adatkapcsolati réteg feladata jól definiált szolgálati interfész biztosítása a hálózati rétegnek, melynek három fázisa van:

- *nyugtázatlan összeköttetés alapú* szolgálat
- *nyugtázott összeköttetés nélküli* szolgálat
- *nyugtázott összeköttetés alapú* szolgálat

Továbbá az átviteli hibák kezelése és az adatforgalom szabályozása (elárasztás elkerülése).

Keretképzés

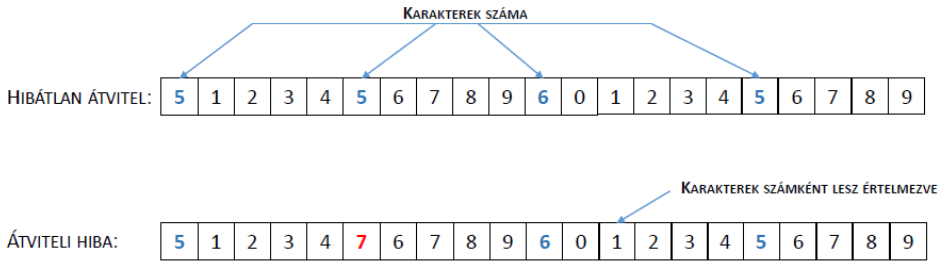
A fizikai réteg nem garantál hibamentességet, az adatkapcsolati réteg feladata a hibajelzés, illetve a szükség szerint javítás.

Erre ad megoldást a keretekre tördelése a bitfolyamnak, és ellenőrző összegek számítása.
A keretezés nem egyszerű feladat, mivel megbízható időzítésre nem nagyon van lehetőség.

Négy lehetséges módszer:

1. ***Karakterszámlálás***

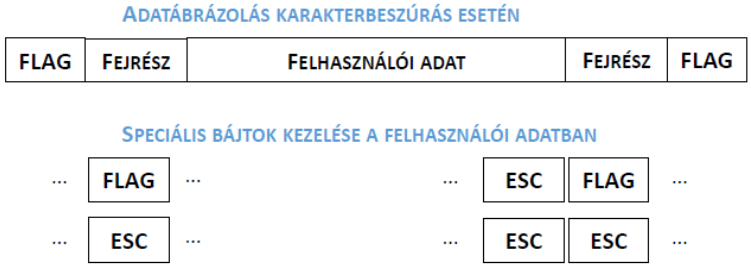
   A keretben lévő karakterek számát a keret fejlécében adjuk meg. Így a vevő adatkapcsolati rétege tudni fogja a keret végét. Probléma: nagyon érzékeny a hibára a módszer.



10. ábra. Karakterszámlálás

2. ***Kezdő és végkarakterek karakterbeszúrással***

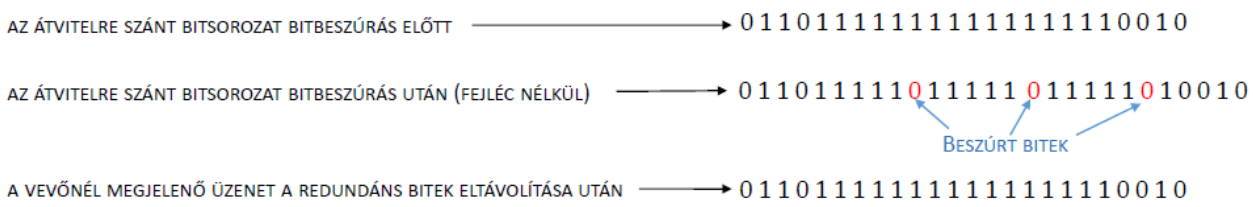
   Különleges bájtokat helyezünk el a keret elejének és végének jelzésére, aminek a neve jelző bájt (flagbyte). Az adatfolyamban szereplő speciális bájtokhoz ESC bájtot használnak.



11. ábra. Kezdő és végkarakterek karakterbeszúrással

3. ***Kezdő és végjelek bitbeszúrása***

   Minden keret egy speciális bitmintával kezdődik (flagbájt, 01111110) és minden egymást követő 5 hosszú folytonos 1-es bit sorozat után beszúr egy 0-át.



12. ábra. Kezdő és végjelek bitbeszúrásal

4. ***Fizikai rétegbeli kódolás-sértés***

   Olyan hálózatokban használható, ahol a fizikai rétegbeli kódolás redundanciát tartalmaz.

Hibakezelés

Hibakezelés szempontjából a következő két esetet kell vizsgálnunk. A keretek megérkeztek-e a célállomás hálózati rétegéhez, illetve helyes sorrendben érkeztek-e meg. Ehhez valamilyen visszacsatolás szükséges a vevő és az adó között. (például nyugták).

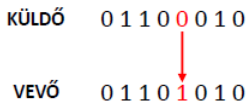
Időkorlátokat vezetünk be az egyes lépésekhez. Hiba estén a csomagot újraküldjük. Többszörös vétel lehet, amin segíthet a sorszámok használata.

Az adatkapcsolati réteg feladata a hibakezelés szempontjából, hogy az időzítőket és számlálókat úgy kezelje, hogy biztosítani tudja a keretek pontosan egyszeri (nem több és nem kevesebb) megérkezését a célállomás hálózati rétegéhez.

Bithibák

- ***Egyszerű bithiba***

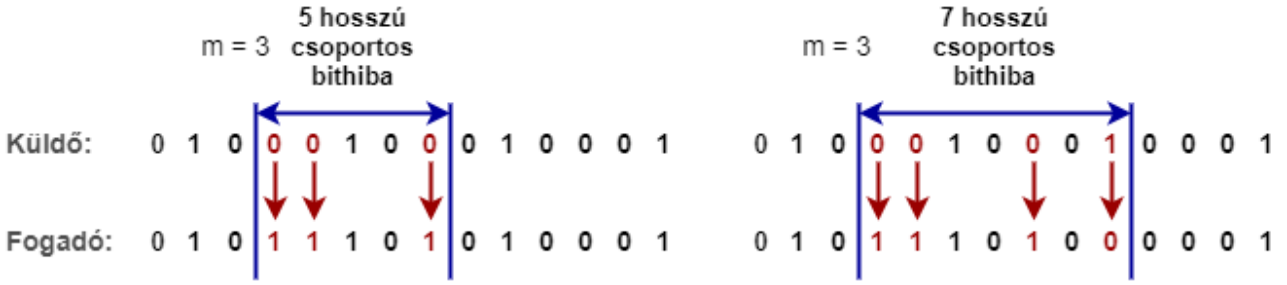
  Az adategység 1 bitje nulláról egyre avagy egyről nullára változik.



13. ábra. Egyszerű bithiba

- ***Csoportos bithiba***

  Egy olyan folytonos szimbólum sorozatot, amelynek az első és utolsó szimbóluma hibás, és nem létezik ezen két szimbólummal határolt részsorozatban olyan m hosszú részsorozat, amelyet helyesen fogadtunk, m hosszú csoportos bithibának nevezünk.



14. ábra. Egyszerű bithiba

Hiba jelzés és javítás

Kétféle hibakezelési stratégia létezik:

- Hibajelző (redundáns információ mellékelése)
- Hibajavító kódok (adatok közé iktatott redundancia).

Megjegyzés: Megbízható csatornákon a hibajelzés olcsóbb. (csomagot inkább újraküldjük). A kevésbé megbízható csatornákon a hibajavításos módszer célszerűbb.

Hamming-távolság, Hamming-korlát

- Küldendő keret m bitet tartalmaz.
- Redundáns bitek száma r.
- Az elküldött keret: $n = m + r$ bit.

Hamming-távolság

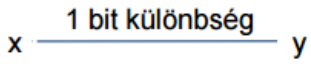
Az olyan bitpozíciók számát, amelyeken a két kódszóban különböző bitek állnak, a két kódszó *Hamming távolságának* nevezzük. Jelölés: $d(x, y)$

Legyen S az egyenlő hosszú bitszavak halmaza. S Hamming-távolsága:

$$d(S) := \min_{x,y,\in S \ \wedge \ x \neq y} d(x, y)$$

$d(S) = 1$ esetén:

Nincs hibafelismerés, ugyanis megengedett kódszóból 1 bit megváltoztatásával megengedett kódszó áll elő.



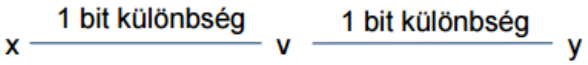
15. ábra. Kód Hamming-távolsága = 1

d(S) = 2 esetén:

Ha az x kódszóhoz létezik olyan v nem megengedett kódszó, amelyre $d(u, x) = 1$, akkor hiba történt. Ha x és y megengedett kódszavak (távolságuk minimális = 2), akkor a következő összefüggésnek teljesülnie kell:

$$2 = d(x, y) \leq d(x, v) + d(v, y)$$

Azaz egy bithiba felismerhető, de nem javítható.



16. ábra. Kód Hamming-távolsága = 2

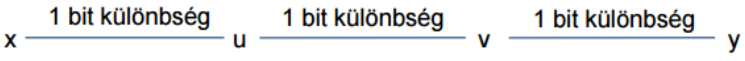
d(S) = 3 esetén:

Ekkor minden u, melyre $d(x, u) = 1$ és $d(u, y) > 1$ nem megengedett.

Ekkor három lehetőség áll fent:

- x került átvitelre és 1 bit hibával érkezett
- y került átvitelre és 2 bit hibával érkezett
- valami más került átvitelre és legalább 2 bit hibával érkezett

De valószínűbb, hogy x került átvitelre, tehát ez egy 1 bit hiba javító, 2 bit hiba felismerő kód.



17. ábra. Kód Hamming-távolsága = 3

Hamming-korlát

$C \subseteq \{0, 1\}^n$ és $d(C) = k$. Ekkor a kódszavak $\frac{k-1}{2}$ sugarú környezeteiben található bitszavak egymással diszjunkt halmazainak uniója legfeljebb az n-hosszú bitszavak halmazát adhatja ki.

Formálisan:

$$|C| \sum_{i=0}^{\lfloor \frac{k-1}{2} \rfloor} \binom{n}{i} \leq 2^n$$

- *Hibafelismerés*:

  d **bit hiba felismeréséhez** a keretek halmazában legalább **d+1** Hamming távolság szükséges.

- *Hibajavítás*:

  d **bit hiba javításához** a megengedett keretek halmazában legalább **2d+1** Hamming távolság szükséges.

- *Kód rátája*:

  $R_S = \frac{log_2|S|}{n}$ a kód rátája ($S \subseteq \{0,1\}^n$) - hatékonyságot karakterizálja

- *Kód távolsága*:

  $\delta_S = \frac{d(S)}{n}$ a kód távolsága ($S \subseteq \{0,1\}^n$) - hibakezelést karakterizálja

A jó kódnak a rátája és a távolsága is nagy.

Paritásbit

A paritásbit olyan bit, melyet a kódszóban lévő egyesek száma alapján választunk.

- **Odd Parity** (páratlan paritás): Az 1-esek száma páratlan. A kódszóban lévő 1-esek számát 1 vagy 0 hozzáadásával *páratlanra egészítjük ki.*
- **Even Parity** (páros paritás): Az 1-esek száma páros. A kódszóban lévő 1-esek számát 1 vagy 0 hozzáadásával *párosra egészítjük ki.*

Egy paritást használó módszer az ún. Hamming módszer:

- A biteket 1-től sorszámozzuk balról, jobbra.
- Minden kettő-hatvány (2^i) helyen paritásbit van.
- Az egyes paritásbitek nem az összes bitet "ellenőrzik".
- A paritásbit ellenőrzi a teljes kódszó i. bitjét, ha az i kettes számrendszerbeli alakjában szerepel a 2^i helyiérték.

Például: a 6. bitet a 4-es és 2-es paritásbit ellenőrzi.

A csoportok a következőképp alakulnak:

- 1. bit: Minden első egyhosszú bitsorozat az első bittől kezdve (tehát: 1,3,5,7,...)
- 2. bit: Minden első kéthosszú bitsorozat a második bittől kezdve (tehát: 2-3,6-7,10-11)
- 4. bit: Minden első négyhosszú bitsorozat a negyedik bittől kezdve (tehát: 4-7,12-15)
- stb.

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15	
Parity bit coverage	p1	X		X		X		X		X		X		X		X		X		X		
	p2		X	X			X	X			X	X			X	X			X	X		...
	p4				X	X	X	X					X	X	X	X					X	
	p8								X	X	X	X	X	X	X	X						
	p16																X	X	X	X	X	

18. ábra. Paritásbitek csoportjai

Példa:

Legyen az átküldendő üzenet: 1000101

Ekkor a kódszó a következőképp alakul:
♡♡1♡000♡101
A 8. bit a 8-11 bitsorozat paritását állítja be párosra:
♡♡1♡0000101
A 4. bit a 4-7 bitsorozat paritását állítja be párosra:
♡♡100000101
A 2. bit a 2-3, 6-7, 10-11 bitsorozat paritását állítja be párosra:
♡0100000101
Az 1. bit az 1,3,5,7,9,11 bitsorozat paritását állítja be párosra:
10100000101

Tehát a elküldendő bitsorozat: 10100000101

CRC - Polinom-kód, azaz ciklikus redundancia

A bitsorozatokat egy $\mathbb{Z}_2$ feletti polinom ($M(x)$) együtthatóinak tekintjük.
Definiálunk egy $G(x)$ r-ed fokú generátorpolinomot, melyet a vevő és küldő egyaránt ismer.

Algoritmus:

1. Fűzzünk r darab 0 bitet a keret alacsony helyi értékű végéhez. Azaz vegyük az $x^r M(x)$ polinomot (ez már $m + r$ fokú)
2. Osszuk el $x^r M(x)$-et $G(x)$-szel (mod 2).
3. A maradékot (mely mindig r vagy kevesebb bitet tartalmaz) vonjuk ki $x^r M(x)$-ből (mod 2). Így az eredeti keret végére egy r hosszú ellenőrző összeg kerül. Legyen ez a polinom $T(x)$.
4. A vevő egy $T(x) + E(x)$-nek megfelelő polinomot kap (ahol $E(x)$ a hiba polinom). Ezt elosztva a generátorpolinommal egy $R(x)$ polinomot kapunk. Ha ez a polinom nem nulla, akkor hiba történt.

Áttekintés:

- $G(x)$ legmagasabb és legalacsonyabb fokú tagjának együtthatója mindig 1.
- A $G(x)$ többszöröseinek megfelelő bithibákat nem ismeri fel, azaz, ha

$$\forall j \in \mathbb{N} : E(x) = x^j G(x)$$

- Szerkeszthető egy egyszerű, léptető regiszteres áramkör az ellenőrző összeg hardverben történő kiszámítására és ellenőrzésére.

Példa: Legyen az

- átküldendő üzenet: 11010011100
- $G(x) = x^3 + x + 1 = \mathbf{1}x^3 + \mathbf{0}x^2 + \mathbf{1}x^1 + \mathbf{1}x^0 = x^3 + x + 1 \quad \rightarrow \mathbf{r = 3}$

Minden lépésben a következő 1-es alá írom a generátorpolinomot első 1-sét, és ha két egyforma érték van egymás alatt, nullát írok alá, különben 1-et.

1 1 0 1 0 0 1 1 1 0 0	eredeti üzenet
1 1 0 1 0 0 1 1 1 0 0 0 0 0	r darab 0-val kiegészítve
1 0 1 1	generátor polinommal osztás
0 1 1 0 0 0 1 1 1 0 0 0 0 0	eredmény
1 0 1 1	generátor polinommal osztás
0 0 1 1 1 0 1 1 1 0 0 0 0 0	⋯
1 0 1 1	
0 0 0 1 0 1 1 1 1 0 0 0 0 0	
1 0 1 1	
0 0 0 0 0 0 0 1 1 0 0 0 0 0	
1 0 1 1	
0 0 0 0 0 0 0 0 1 1 1 0 0 0	
1 0 1 1	
0 0 0 0 0 0 0 0 0 1 0 1 0 0	
1 0 1 1	
0 0 0 0 0 0 0 0 0 0 0 0 1 0	

Az utolsó $r(=3)$ bitet hozzáfüzzük az eredeti üzenethez: 11010011100**010**

Ellenőrzés a vevő oldalon:

1 1 0 1 0 0 1 1 1 0 0 0 1 0	kódolt üzenet
1 0 1 1	generátor polinommal osztás
0 1 1 0 0 0 1 1 1 0 0 0 1 0	eredmény
1 0 1 1	generátor polinommal osztás
0 0 1 1 1 0 1 1 1 0 0 0 1 0	⋯
1 0 1 1	
0 0 0 1 0 1 1 1 1 0 0 0 1 0	
1 0 1 1	
0 0 0 0 0 0 0 1 1 0 0 0 1 0	
1 0 1 1	
0 0 0 0 0 0 0 0 1 1 1 0 1 0	
1 0 1 1	
0 0 0 0 0 0 0 0 0 1 0 1 1 0	
1 0 1 1	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	

Minden bit 0 lett, azaz az üzenet nem sérült.

Protokollok

Elemi adatkapcsolati protokollok

Feltevések:

- A fizikai, az adatkapcsolati és a hálózati réteg független folyamatok, amelyek üzeneteken keresztül kommunikálnak egymással.
- Az A gép megbízható, összeköttetés alapú szolgálat alkalmazásával akar a B gépnek egy hosszú adatfolyamot küldeni. (Adatok előállítására sosem kell várnia A gépnek.)
- A gépek nem fagynak le.
- Adatkapcsolati fejrészben vezérlési információk.
- Adatkapcsolati lábrészben ellenőrző összeg.

Kommunikáció fajtái:

- szimplex kommunikáció – a kommunikáció pusztán egy irányba lehetséges
- fél-duplex kommunikáció – mindkét irányba folyhat kommunikáció, de egyszerre csak egy irány lehet aktív.
- duplex kommunikáció – mindkét irányba folyhat kommunikáció szimultán módon

Az összes protokoll esetében

- Résztvevők: küldő és vevő

Korlátozás nélküli szimplex protokoll

A környezet

- Mind az adó, mind a vevő hálózati rétegei mindig készen állnak
- A feldolgozási időktől eltekintünk
- Végtelen puffer-területet feltételezünk
- Az adatkapcsolati rétegek közötti kommunikációs csatorna sosem rontja/veszíti el a kereteket

A protokoll

- Nincs sem sorszámozás, sem nyugta (szimplex)
- A küldő végtelen ciklusban küldi kifelé a kereteket folyamatosan
- A vevő kezdetben várakozik az első keret megérkezésére, keret érkezésekor a hardver puffer tartalmát változóba teszi és az adatrészt tovább küldi a hálózati rétegnek

Szimplex megáll és vár protokoll

A környezet

- Mind az adó, mind a vevő hálózati rétegei mindig készen állnak
- A vevőnek Δt időre van szüksége a bejövő keret feldolgozására

  nincs pufferelés és sorban állás sem
- Az adatkapcsolati rétegek közötti kommunikációs csatorna sosem rontja/veszíti el a kereteket

A protokoll

- A küldő egyesével küldi kereteket és addig nem küld újat, még nem kap nyugtát a vevőtől (fél-duplex)
- A vevő kezdetben várakozik az első keret megérkezésére, keret érkezésekor a hardver puffer tartalmát változóba teszi és az adatrészt továbbküldi a hálózati rétegnek, végül nyugtázza a keretet

Szimplex protokoll zajos csatornához

A környezet

- Mind az adó, mind a vevő hálózati rétegei mindig készen állnak
- A vevőnek Δt időre van szüksége a bejövő keret feldolgozására

  nincs pufferelés és sorban állás sem
- Az adatkapcsolati rétegek közötti kommunikációs csatorna hibázhat

  keret megsérülése vagy elvesztése

A protokoll

- A küldő egyesével küldi kereteket és addig nem küld újat, még nem kap nyugtát a vevőtől egy megadott határidőn belül, ha a határidő lejár, akkor ismételten elküldi az aktuális keretet.
- A vevő kezdetben várakozik az első keret megérkezésére, keret érkezésekor a hardver puffer tartalmát változóba teszi, leellenőrzi a kontroll összeget:
  - Nincs hiba: az adatrészt továbbküldi a hálózati rétegnek, végül nyugtázza a keretet.
  - Hiba történt: eldobja a keretet és nem nyugtáz (duplikátumok kellenek)

Csúszóablakos protokoll

Alapok (általános):

- Egy adott időpontban egyszerre több keret is átviteli állapotban lehet
- A fogadó n keretnek megfelelő méretű puffert allokál
- A küldőnek legfeljebb n, azaz ablak méretnyi, nyugtázatlan keret küldése engedélyezett
- A keret sorozatbeli pozíciója adja a keret címkéjét (sorozatszám)

Alapok (fogadó):

- A keret nyugtázója tartalmazza a következőnek várt keret sorozatszámát.
  - kumulatív nyugta – Olyan nyugta, amely több keretet nyugtáz egyszerre. Például, ha a 2,3 és 4 kereteket is fogadnánk, akkor a nyugtát 5 sorszám tartalommal küldenénk, amely nyugtázza mind a három keretet.
- A hibás kereteket el kell dobni.
- A nem megengedett sorozatszámmal érkező kereteket el kell dobni.

Jellemzők (általános):

- A küldő nyilvántartja a küldhető sorozatszámok halmazát. (adási ablak)
- A fogadó nyilvántartja a fogadható sorozatszámok halmazát. (vételi ablak)
- A sorozatszámok halmaza minden esetben véges.
  - K bites mező esetén: $[0..2K-1]$
- A adási ablak minden küldéssel szűkül, illetve nő egy nyugta érkezésével.

Jellemzők (gyakorlati alkalmazás esetén):

- A gyakorlatban kétirányú adatfolyamot kell kezelni (duplex csatorna)
  - két különböző szimplex csatorna használata (két áramkör használata)
  - egy csatorna használata (egy áramkör használata)
    - piggybacking módszer– a kimenő nyugtákat késleltetjük, hogy rá tudjuk akasztani a következő kimenő adatkeretre (ack mező használata)

Mi van ha egy hosszú folyam közepén történik egy keret hiba?

- **"visszalépés N-nel" stratégia**

  Stratégia:

  - Az összes hibás keret utáni keretet eldobja és nyugtát sem küld róluk.
  - Mikor az adónak lejár az időzítője, akkor újraküldi az összes nyugtázatlan keretet, kezdve a sérült vagy elveszett kerettel.

  Következmények:

  - Egy méretű vételi ablakot feltételez.
  - Nagy sávszéleséget pazarolhat el, ha nagy a hibaarány.

- **”szelektív ismétlés” stratégia**

  Stratégia:

  - A hibás kereteket eldobja, de a jó kereteket a hibás után puffereli.
  - Mikor az adónak lejár az időzítője, akkor a legrégebbi nyugtázatlan keretet küldi el újra.

  Következmények:

  - Javíthat a hatékonyságon a negatív nyugta használata. (NAK)
  - Egynél nagyobb méretű vételi ablakot feltételezünk.
  - Nagy memória igény, ha nagy vételi ablak esetén.

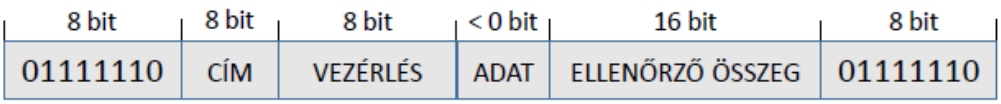
Példák adatkapcsolati protokollokra

- **HDLC - High-level Data Link Control**
  A HDLC protokoll 3 bites csúszó-ablak protokollt alkalmaz a sorszámozáshoz.
  Három típusú keretet használ:

  - információs
  - felügyelő
    - nyugtakeret (RECIEVE READY)
    - negatív nyugta keret (REJECT)
    - vételre nem kész (RECIEVE NOT READY) - nyugtáz minden keretet a következőig
    - szelektív elutasítás (SELECTIVE REJECT) - egy gy adott keret újraküldésére szólít fel
  - Számozatlan

  Általános keretfelépítése:

  - FLAG bájt a keret határok jelzésére
  - *cím* mező - több vonallal rendelkező terminálok esetén van jelentősége
  - *vezérlés* mező - sorszámozás, nyugtázás és egyéb feladatok ellátására
  - *adat* mező - tetszőleges hosszú adat lehet
  - *ellenőrző összeg* mező - CRC kontrollösszeg (CRC-CCITT generátor polinom felhasználásával)

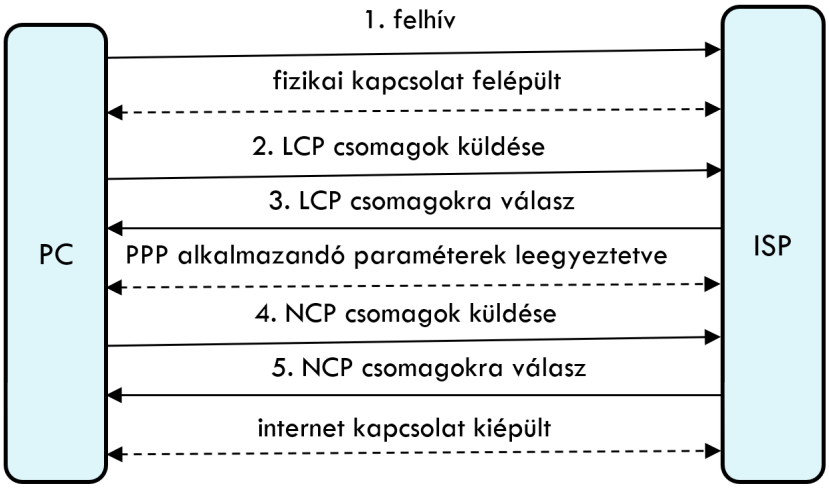


19. ábra. HDLC keret felépítése

- **PPP - Point-to Point Protocol**

  A PPP protokoll három dolgot biztosít:

  - Keretezési módszert (egyértelmű kerethatárok)
  - Kapcsolatvezérlő protokollt (a vonalak felélesztésére, tesztelésére, az opció egyeztetésére és a vonalak elengedésére.)
  - Olyan módot a hálózati réteg-opciók megbeszélésére, amely független az alkalmazott hálózati réteg-protokolltól.



20. ábra

Bájt alapú keretszerkezet használ (azaz a legkisebb adategység a bájt). Vezérlő mező alapértéke a számozatlan keretet jelzi. Protokoll mezőben protokoll kód lehet az LCP, NCP, IP, IPX, AppleTalk vagy más protokollhoz.

1	1	1	1 vagy 2	változó	2 vagy 4	1
Jelző 01111110	Cím 1111111	Vezérlő 00000011	Protokoll	Adatmező	Ellenőrző összeg	Jelző 01111110

21. ábra. PPP keret felépítése

MAC - Media Access Control

Egyetlen üzenetszórásos csatorna megosztása több egymással versengő felhasználó (állomás) között. A csatorna kiosztás történhet statikus vagy dinamikus módon.

Statikus csatornamegosztási módszerek

Statikus esetben vagy frekvenciaosztásos nyalábolást vagy időosztásos nyalábolást használnak.

- Frekvenciaosztásos (FDM) esetben
  - A sávszélességet osztják n egyenlő részre, és mindegyik felhasználó egy sávot kap.
  - Mindenkinek külön frekvenciasávja van, így nincs interferencia.

- Időosztásos (TDM) esetben
  - Az időegységet osztják n egyenlő időrésre.
  - Minden felhasználóhoz statikusan egy időrést rendel.
    (Az i. részben az i. felhasználó a teljes sávszélességet használhatja)

Jellemzőik:

- kötött a felhasználószám
- a keretek átlagos késleltetése n-szerese annak, mintha az egész csatorna egy felhasználóé lenne
  - FDM-nél kisebb (n-ed része) sávszélesség,
  - TDM-nél nagyobb várakozási idő van
    (a ciklusidő n-ed része csak az övé).
- a keretek késleltetése állandó
- a csatornakihasználtság
  - rossz kis és nem egyenletes forgalom esetén
    (ha foglalt csatorna üres, más akkor sem veheti át)
  - jó folytonos, egyenletes terheléskor
    (pl. telefonközpontközi trönk – kevés rögzített számú felhasználó, nagy egyenletes terhelés).

Dinamikus csatornamegosztási módszerek

Verseny protokollok

N független állomás van, amelyeken egy program vagy egy felhasználó továbbítandó kereteket generál. Ha egy állomás generált egy keretet, akkor blokkolt állapotban marad mindaddig, amíg a keretet sikeresen nem továbbította. Egyetlen csatorna van, melyen mindenféle kommunikáció zajlik. Minden állomás tud adatot küldeni és fogadni ezen a csatornán. Ha két keret egy időben kerül átvitelre, akkor átlapolódnak, és az eredményül kapott jel értelmezhetetlenné válik. Ezt nevezzük ütközésnek. Ez minden állomás számára felismerhető. Az ütközésben érintett kereteket később újra kell küldeni. (Ezen a hibán kívül egyéb hiba nem történhet.)

Kétféle időmodellt különböztetünk meg:

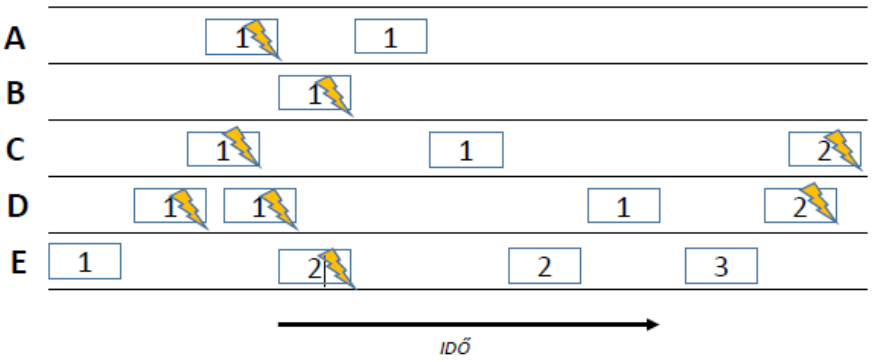
1. **Folytonos** – Mindegyik állomás tetszőleges időpontban megkezdheti a küldésre kész keretének sugárzását.
2. **Diszkrét** – Az időt diszkrét résekre osztjuk. Keret továbbítás csak időrés elején lehetséges. Az időrés lehet üres, sikeres vagy ütközéses

Az egyes állomások vagy rendelkeznek vivőjel érzékeléssel vagy nem.

- Ha nem, akkor az állomások nem tudják megvizsgálni a közös csatorna állapotát, ezért egyszerűen elkezdenek küldeni, ha van rá lehetőségük.
- Ha igen, akkor az állomások meg tudják vizsgálni a közös csatorna állapotát a küldés előtt. A csatorna lehet: foglalt vagy szabad. Ha a foglalt a csatorna, akkor nem próbálják használni az állomások, amíg fel nem szabadul

Egyszerű ALOHA

- A felhasználó akkor vihet át adatot, amikor csak szeretne.
- Ütközés esetén véletlen ideig várakozik az állomás, majd újra próbálkozik.
- Keret idő–egy szabványos, fix hosszúságú keret átviteléhez szükséges idő (T).
- Egy keret akkor nem szenved ütközést, ha elküldésének első pillanatától kezdve egy keret-ideig nem próbálkozik más állomás keretküldéssel.
- 2T ideig ütközésveszélyes.



22. ábra. Egyszerű ALOHA keret ütközések

Réselt ALOHA

- Az időt keretidőnyi (T) résekre osztják
  - külön meg kell oldani az állomások szinkronizációját, pl. egy külön állomás küld egy speciális jelet minden rés elején.
- Adást csak az időrés elején lehet kezdeni!
- Az ütközésveszélyes kritikus időszakasz 2T-ről T-re csökken az egyszerű ALOHA-hoz képest.

1-prezisztens CSMA

Vivőjel érzékelés van, azaz minden állomás belehallgathat a csatornába. Folytonos időmodellt használ a protokoll.

Algoritmus:

1. Keret leadása előtt belehallgat a csatornába:
   (a) Ha foglalt, akkor addig vár, amíg fel nem szabadul. Szabad csatorna esetén azonnal küld. (perzisztens)
   (b) Ha szabad, akkor küld.
2. Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

Nem-prezisztens CSMA

Vivőjel érzékelés van, azaz minden állomás belehallgathat a csatornába. Folytonos időmodellt használ a protokoll. Mohóságot kerüli, azaz nem küld azonnal, ha foglalt.

Algoritmus:

1. Keret leadása előtt belehallgat a csatornába:
   (a) Ha foglalt, akkor véletlen ideig vár (nem figyeli a forgalmat), majd kezdi előről a küldési algoritmust. (nem-perzisztens)
   (b) Ha szabad, akkor küld.
2. Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

p-prezisztens CSMA

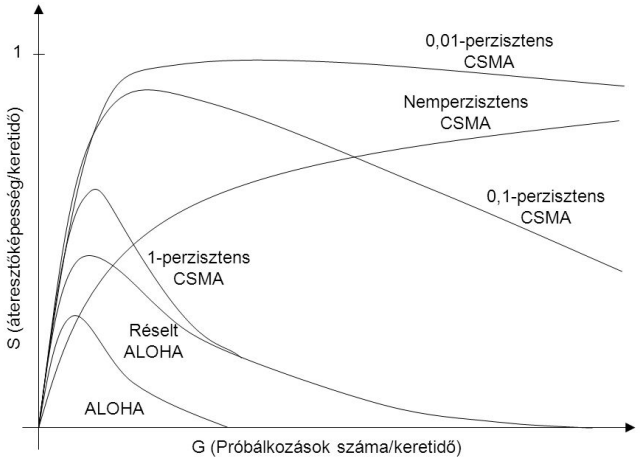
Vivőjel érzékelés van, azaz minden állomás belehallgathat a csatornába. Diszkrét időmodellt használ a protokoll.

Algoritmus:

1. Adás kész állapotban az állomás belehallgat a csatornába:
   (a) Ha foglalt, akkor vár a következő időrésig, majd megismétli az algoritmust.
   (b) Ha szabad, akkor p valószínűséggel küld, illetve $1 - p$ valószínűséggel visszalép a szándékától a következő időrésig. Várakozás esetén a következő időrésben megismétli az algoritmust. Ez addig folytatódik, amíg el nem küldi a keretet, vagy amíg egy másik állomás el nem kezd küldeni, mert ilyenkor úgy viselkedik, mintha ütközés történt volna.
2. Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

CSMA/CD

Ütközés érzékelés esetén meg lehessen szakítani az adást. Minden állomás küldés közben megfigyeli a csatornát, ha ütközést tapasztalna, akkor megszakítja az adást, és véletlen ideig várakozik, majd újra elkezdi leadni a keretét.



Verseny mentes protokollok

Motiváció: Az ütközések hátrányosan hatnak a rendszer teljesítményére, és a CSMA/CD nem mindenhol alkalmazható.

N állomás van. Az állomások 0-ától N-ig egyértelműen sorszámozva vannak. Réselt időmodellt feltételezünk.

- **Egy helyfoglalásos protokoll**
  Ha az i-edik állomás küldeni szeretne, akkor az i-edik versengési időrésben egy 1-es bit elküldésével jelezheti. Így a versengési időszak végére minden állomás ismeri a küldőket. A küldés a sorszámok szerinti sorrendben történik meg.

- **Bináris visszaszámlálás protokoll**
  Minden állomás azonos hosszú bináris azonosítóval rendelkezik. A forgalmazni kívánó állomás elkezdi a bináris címét bitenként elküldeni a legnagyobb helyi értékű bittel kezdve. Az azonos pozíciójú bitek logikai VAGY kapcsolatba lépnek ütközés esetén. Ha az állomás nullát küld, de egyet hall vissza, akkor feladja a küldési szándékát, mert van nála nagyobb azonosítóval rendelkező küldő.

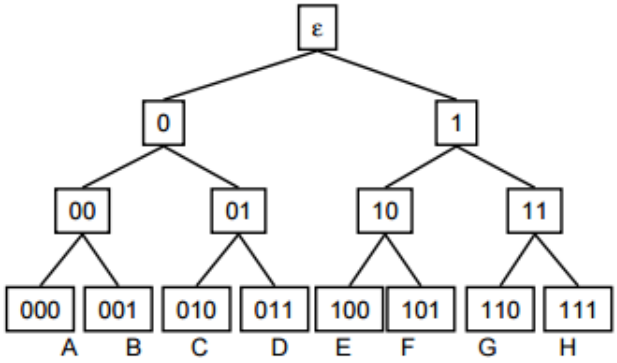
Korlátozott verseny protokollok

Olyan protokoll, amely kis terhelés esetén versenyhelyzetes technikát használ a kis késleltetés érdekében, illetve nagy terhelés mellett ütközésmentes technikát alkalmaz a csatorna jó kihasználása érdekében.

Adaptív fabejárás

- Minden állomást egy egyértelmű, bináris ID reprezentál.
- Az ID-k egy (bináris) fa leveleinek felelnek meg.
- Az időrések a fa egyes csomópontjaihoz vannak rendelve.
- Minden időrésben megvizsgáljuk az adott csomópont alatti részfát.
- A fa egy u csomópontjánál 3 esetet különböztethetünk meg:
  - Egy állomás sem küld az u részfában.
  - Pontosan egy állomás küld az u részfában.
  - Több állomás küld az u részfában. Ezt nevezzük kollíziónak.

Kollízió esetén hajtsuk végre az ellenőrzést u bal, és jobb oldali gyerekére egyaránt. Ezzel a módszerrel könnyen megállapítható, hogy melyik állomás küldhet az adott időszeletben.



23. ábra. Adaptív fabejárás protokoll bináris fája

3. Hálózati réteg

A hálózati réteg fő feladata a csomagok továbbítása a forrás és a cél között. Ez a legalacsonyabb olyan réteg, amely két végpont közötti átvitellel foglalkozik. Ismernie kell a kommunikációs alhálózat topológiáját. Ügyelni kell, hogy ne terheljen túl se bizonyos kommunikációs útvonalakat, se bizonyos routereket úgy, hogy mások tétlen maradnak.

A szállítási réteg felé nyújtott szolgálatok:

- Függetlenek az alhálózatok kialakításától
- Eltakarják a jelen lévő alhálózatok számát, típusát és topológiáját
- A szállítási réteg számára rendelkezésre bocsájtott hálózati címek egységes számozási rendszert kell alkotnak

Forgalom irányítás típusai

- **Hierarchikus forgalomirányítás**

  Routereket tartományokra osztjuk. A saját tartományát az összes router ismeri, de a többi belső szerkezetéről nincs tudomása. Nagy hálózatok esetén többszintű hierarchia lehet szükséges.

- **Adatszóró forgalomirányítás**

  Egy csomag mindenhová történő egyidejű küldése.

- **Többküldéses forgalomirányítás**

  Egy csomag meghatározott csoporthoz történő egyidejű küldése.

Forgalom irányító algoritmusok

A hálózati réteg szoftverének azon része, amely azért a döntésért felelős, hogy a bejövő csomag melyik kimeneti vonalon kerüljön továbbításra. A folyamat két lépésre bontható:

1. Forgalomirányító táblázatok feltöltése és karbantartása.
2. Továbbítás.

A forgalomirányító algoritmusok osztályai:

1. Adaptív algoritmusok
   (a) távolság alapú
   (b) kapcsolat alapú

   A topológia és rendszerint a forgalom is befolyásolhatja a döntést.
2. Nem-adaptív algoritmusok
   Offline meghatározás, betöltés a routerekbe induláskor

Dijkstra algoritmus

A Dijkstra algoritmus egy statikus algoritmus, melynek célja két csomópont közötti legrövidebb út meghatározása.

Minden csomópontot felcímkézünk a kezdőpontból az addig megtalált legrövidebb út távolságával. Kezdetben minden távolság végtelen, mivel nem ismerünk útvonalat.

Az algoritmus működése során a címkék változhatnak az utak megtalálásával. Két fajta címkét különböztetünk meg: ideiglenes és állandó. Kezdetben minden címke ideiglenes. A legrövidebb út megtalálásakor a címke állandó címkévé válik, és továbbá nem változik.

Elárasztás algoritmus

Elárasztás algoritmusa egy statikus algoritmus.

Minden bejövő csomagot minden kimenő vonalon továbbítunk kivéve azon, amin érkezett. Így azonban nagyon sok duplikátum keletkezne. Ezért

- Ugrásszámlálót vezetünk be, melyet minden állomás eggyel csökkent. Ha 0-ra csökken, eldobják.
- Az állomások nyilvántartják a már kiküldött csomagokat. Így egy csomagot nem küldenek ki többször.

Elosztott Bellman-Ford algoritmus

Az Elosztott Bellman-Ford algoritmus adaptív, távolság alapú forgalomirányító algoritmus. Minden csomópont csak a közvetlen szomszédjaival kommunikálhat. Minden állomásnak van saját távolság vektora. Ezt periodikusan elküldi a direkt szomszédoknak. Minden router ismeri a közvetlen szomszédjaihoz a költséget. A kapott távolság vektorok alapján minden csomópont aktualizálja a saját vektorát.

Kapcsolatállapot alapú forgalomirányítás

A kapcsolatállapot alapú forgalomirányító algoritmusnak a motivációja, hogy a távolság alapú algoritmusok lassan konvergáltak, illetve az eltérő sávszélek figyelembevétele.

A kapcsolatállapot alapú forgalomirányító algoritmus lépései:

1. Szomszédok felkutatása, és hálózati címeik meghatározása
2. Megmérni a késleltetést vagy költséget minden szomszédhoz
3. Egy csomag összeállítása a megismert információkból
4. Csomag elküldése az összes többi routernek
5. Kiszámítani a legrövidebb utat az összes többi routerhez.

Hálózati réteg az Interneten

A hálózati réteg szintjén az internet autonóm rendszerek összekapcsolt együttesének tekinthető. Nincs igazi szerkezete, de számos főbb gerinchálózata létezik.

A gerinchálózatokhoz csatlakoznak a területi illetve regionális hálózatok. A regionális és területi hálózatokhoz csatlakoznak az egyetemeken, vállalatoknál és az internet szolgáltatóknál lévő LAN-ok. Az internet protokollja, az IP.

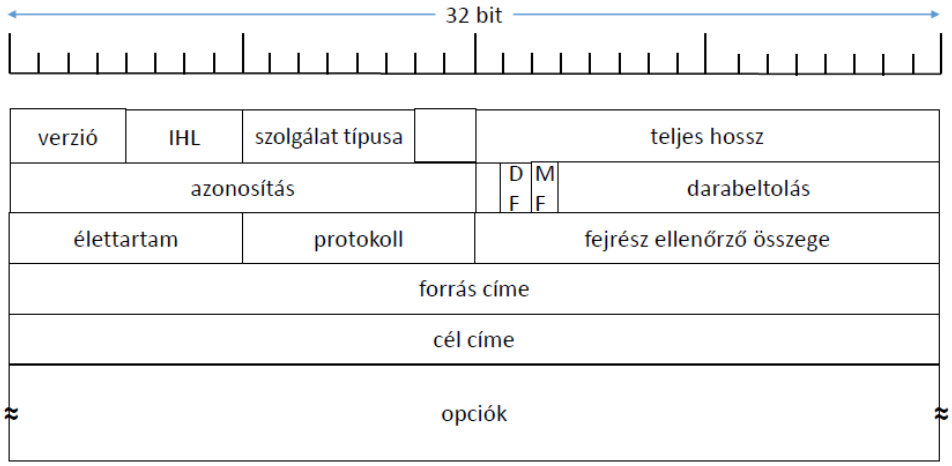
Az Interneten a kommunikáció az alábbi módon működik:

1. A szállítási réteg viszi az adatfolyamokat és datagramokra tördeli azokat.
2. Minden datagram átvitelre kerül az Interneten, esetleg menet közben kisebb egységekre darabolva.
3. A célgép hálózati rétege összeállítja az eredeti datagramot, majd átadja a szállítási rétegének.
4. A célgép szállítási rétege beilleszti a datagramot a vételi folyamat bemeneti adatfolyamába.

Internet Protokoll - IP

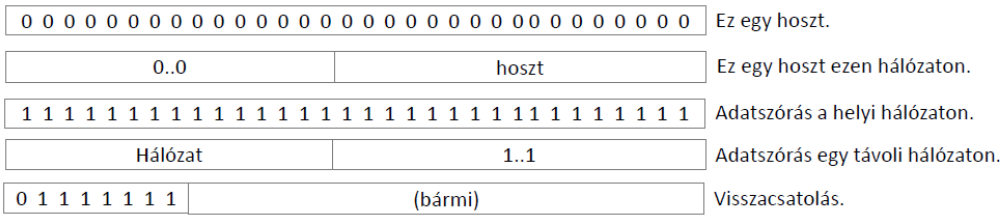
- Az IP fejrésze:
  - verzió: IP melyik verzióját használja
  - IHL: a fejléc hosszát határozza meg
  - szolgálat típusa: szolgálati osztályt jelöl
  - teljes hossz: fejléc és adatrész együttes hossza bájtokban
  - azonosítás: egy datagram minden darabja ugyanazt az azonosítósértéket hordozza.
  - DF: ”ne darabold” flag a routereknek

- MF: ”több darab” flag minden darabban be kell legyen állítva, kivéve az utolsót.
- darabeltolás: a darab helyét mutatja a datagramon belül.
- élettartam: másodpercenként kellene csökkenteni a mező értékét, minden ugrásnál csökkentik eggyel az értékét
- protokoll: szállítási réteg protokolljának azonosítóját tartalmazza
- ellenőrző összeg: a routereken belüli rossz memóriaszavak által előállított hibák kezelésére használt ellenőrző összeg a fejrészre, amelyet minden ugrásnál újra kell számolni
- forrás cím és cél cím: IP cím
- opciók: következő verzió bővíthetősége miatt hagyták benne.



24. ábra. IPv4 fejléce

- Az IP cím
  Minden hoszt és minden router az Interneten rendelkezik egy IP-címmel, amely a hálózat számát és a hoszt számát kódolja. 4 bájton ábrázolják az IP-címet. Az IP-t pontokkal elválasztott decimális rendszerben írják. (Például: 192.168.0.1)
  Van pár speciális cím (ábra 25).



25. ábra. Speciális IP címek

Alhálózatok

Az azonos hálózatban lévő hosztok ugyanazzal a hálózatszámmal rendelkeznek. Egy hálózat belső felhasználás szempontjából több részre osztódhat, de a külvilág számára egyetlen hálózatként jelenik meg. Azonosításnál az alhálózati maszk ismerete kell a routernek. A forgalomirányító táblázatba a routereknél (hálózat,0) és (saját hálózat, hoszt) alakú bejegyzések. Ha nincs találat, akkor az alapértelmezett router felé továbbítják a csomagot.

IP címek fogyása

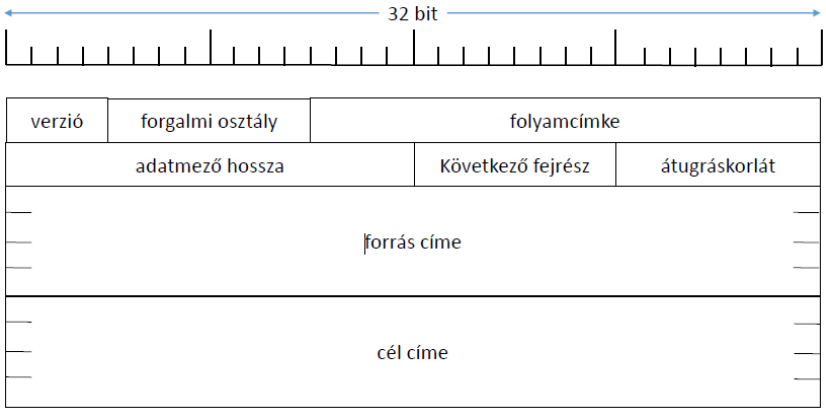
Az IP címek gyorsan fogytak. Megoldás: osztályok nélküli környezetek közötti forgalomirányítás (CIDR). A forgalomirányítás megbonyolódik: Minden bejegyzés egy 32-bites maszkkal egészül ki. Egy bejegyzés innentől egy hármassal jellemezhető: (ip-cím, alhálózati maszk, kimeneti vonal). Új csomag esetén a cél címből kimaszkolják az alhálózati címet, és találat esetén a leghosszabb illeszkedés felé továbbítják.

Másik módszer a NAT, ami gyors javítás az IP címek elfogyásának problémájára. Az internet forgalomhoz minden cégnek egy vagy legalábbis kevés IP-címet adnak, míg vállalaton belül minden számítógéphez egyedi IP-címet használnak a belső forgalomirányításra:

10.0.0.0	10.255.255.255	16.777.216 egyedi cím
172.16.0.0	172.31.255.255	1.048.576 egyedi cím
192.168.0.0	192.168.255.255	65.536 egyedi cím

IPv6:

Az IPv4-gyel szemben 16 bájt hosszú címeket használ; 8 darab, egyenként
négy-négy hexadecimális számjegyből álló csoportként írjuk le.
(Például: 8000:0000:0000:0000:0123:4567:89AB:CDEF) Az IP fejléc egyszerűsödött, amely lehetővé teszi a routereknek a gyorsabb feldolgozást. A biztonság irányába jelentős lépés történt.



26. ábra. IPv6 fejléce

Protokollok

- **Internet Control Message Protocol - ICMP**
  Feladata a váratlan események jelentése. Többféle ICMP-üzenetet definiáltak:
  - Elérhetetlen cél
  - Időtúllépés
  - Paraméterprobléma
  - Forráslefojtás
  - Visszhang kérés
  - Visszhang válasz
  - etc.
- **Address Resolution Protocol - ARP**

  Feladata az IP cím megfeleltetése egy fizikai címnek. Egy "Kié a 192.60.34.12-es IP-cím?" csomagot küld ki az Ethernet-re adatszórással az alhálózaton. Minden egyes host ellenőrzi, hogy övé-e a kérdéses IP-cím. Ha egyezik az IP a hoszt saját IP-jével, akkor a saját Ethernet címével válaszol.
- **Reverse Address Resolution Protocol - RARP**

  Feladatat a fizikai cím megfeleltetése egy IP címnek. Az újonnan indított állomás adatszórással csomagot küld ki az Ethernetre: "A 48-bites Ethernet-címem 14.04.05.18.01.25. Tudja valaki az IP címemet?" Az RARP-szerver pedig válaszol a megfelelő IP címmel, mikor meglátja a kérést.
- **Open Shortest Path First - OSPF**

  Az OSPF az AS-eken (Autonomus System) belüli forgalomirányításért felel. A hálózat topológiáját térképezi fel, és érzékeli a változásokat. A topológiát egy súlyozott irányított gráffal reprezentálja, melyben legolcsóbb utakat keres.
- **Border Gateway Protocol - BGP**

  Feladata hogy a politikai szempontok szerepet játsszanak az AS-ek közötti forgalomirányítási döntésekben.

  Például: Az IBM-nél kezdődő illetve végződő forgalom ne menjen át a Microsoft-on vagy Csak akkor haladjunk át Albánián, ha nincs más út a célhoz.

  A BGP router szempontjából a világ AS-ekből és a közöttük átmenő vonalakból áll. (Két AS összekötött, ha van köztük a BGP-routereiket összekötő vonal.)
  Az átmenő forgalom szempontjából 3 féle hálózat van:
  - Csonka hálózatok, amelyeknek csak egyetlen összeköttetésük van a BGP gráffal
  - Többszörösen bekötött hálózatok, amelyeket használhatna az átmenő forgalom, de ezek ezt megtagadják
  - Tranzit hálózatok, amelyek némi megkötéssel, illetve általában fizetség ellenében, készek kezelni harmadik fél csomagjait

4. Szállítói réteg

- A szállítási réteg biztosítja, hogy a felhasználók közötti adatátvitel transzparens (átlátszó) legyen.
- A réteg biztosítja, és ellenőrzi egy adott kapcsolat megbízhatóságát.
- Az alkalmazási rétegtől kapott adat elejére egy úgynevezett fejlécet csatol, mely jelzi, hogy melyik szállítási rétegbeli protokollal küldik az adatot.
- Néhány protokoll kapcsolat orientált. Ez azt jelenti, hogy a réteg nyomon követi az adatcsomagokat, és hiba esetén gondoskodik a csomag vagy csomagok újraküldéséről.

Kapcsolat nélküli és kapcsolatorientált

A kapcsolatorientált protokoll elfedi az alkalmazások előtt az átvitel esetleges hibáit, nem kell törődnünk az elveszett, vagy duplán megérkezett, illetve sérült csomagokkal, és azzal sem, hogy milyen sorrendben érkeztek meg. Viszont ez rontja a teljesítményét.
Kapcsolat nélküli esetben nincs szükség az adat keretekre bontására, és nincs csomagújraküldés.

Megbízhatóság

A megbízhatóság ismérvei:

- Minden csomag megérkezése nyugtázásra kerül.
- A nem nyugtázott adatcsomagokat újraküldik.
- A fejléchez és a csomaghoz ellenőrzőösszeg van rendelve.
- A csomagokat számozza, és a fogadónál sorba rendezésre kerülnek a csomagok a sorszámaik alapján.
- Duplikátumokat törli.

Torlódásfelügyelet

Minden hálózaton korlátos az átviteli sávszélessége. Ha több adatot vezetünk a hálózatba, akkor az torlódáshoz (congestion) vezet, vagy akár a hálózat összeomlásához (congestive collapse). Következmény: Az adatcsomagok nem érkeznek meg.

Lavina jelenség: A hálózat túlterhelése csomagok elvesztését okozza, ami csomag újraküldését eredményezi. Az újraküldés tovább növeli a hálózat terhelését így még nagyobb lesz csomagvesztés. Ez még több újraküldött csomagot eredményez. ...
A torlódás felügyelet feladata a lavina jelenség elkerülése.

Követelmények a torlódásfelügyelettel szemben

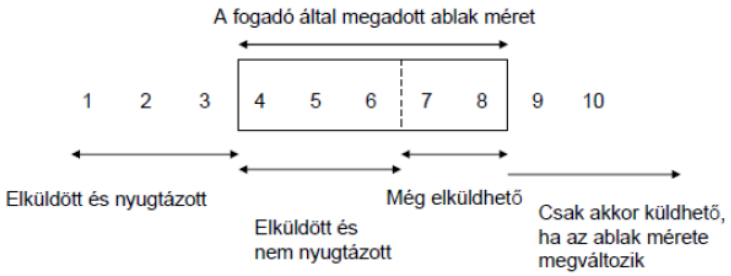
- Hatékonyság: Az átvitel nagy, míg a késés kicsi.
- Fairness: Minden folyamat megközelítőleg azonos részt kap a sávszélességből. (Priorizálás lehetősége fennáll)

A torlódásfelügyelet eszközei

- Kapacitásnövelés
- Erőforrás foglalás és hozzáférés szabályzás
- Terhelés csökkentése és szabályzása

Stratégiák

- **Csúszóablak**
  Adatráta szabályozása ablak segítségével. A fogadó határozza meg az ablak (wnd) méretét. Ha a fogadási puffere tele van, lecsökkenti 0-ra, egyébként >0-t küld. A küldő nem küld több csomagot, ha az elküldött még nem nyugtázott csomagok száma elérte az ablak méretét.



27. ábra. Csúszóablak

- **Slow-start**
  A küldőnek nem szabad a fogadó által küldött ablakméretet azonnal kihasználni. Meghatároz egy másik ablakot (cwnd - Congestion Window), melyet ő választ. Ezután végül amiben küld: min{wnd, cwnd}. Kezdetben cwnd = MSS (Maximum Segment Size). Minden csomagnál a megkapott nyugta után növeli: cwnd = cwnd + MSS (azaz minden RTT után duplázódik). Ez addig megy, míg a nyugta egyszer kimarad.
- **TCP-Nagle**
  Biztosítani kell, hogy a kis csomagok időben egymáshoz közel kerüljenek kiszállításkor, illetve hogy sok adat esetén a nagy csomagok részesüljenek előnyben.
  - A kis csomagok nem kerülnek kiküldésre, míg nyugták hiányoznak (egy csomag kicsi, ha az adathossz <MSS).
  - Ha a korábban küldött csomag nyugtája megérkezik, küldi a következőt.
- **TCP Tahoe és Reno**
  A TCP csúszóablakot és a Slow-start mechanizmusát is használja. Habár a kezdő ráta kicsi, az ablak mérete rohamosan nő. Amikor a cwnd eléri az ssthresh (slow start threshold) értéket átvált torlódás elkerülési állapotba. A TCP Tahoe és Reno torlódás elkerülési algoritmusok. A két algoritmus abban különbözik, hogy hogyan detektálják és kezelik a csomag vesztést.

  TCP Tahoe: A torlódás detektálására egy időzítőt állít a várt nyugta megérkezésére.
  - Kapcsolatfelvételkor:
    - cwnd = MSS,

        - ssthresh = 2^{16}
    - Csomagvesztésnél : Multiplicative decrease
        - cwns = MSS,
        - ssthresh = max $\left\{2MSS, \frac{min\{cwnd,wnd\}}{2}\right\}$
    - cwnd $\leq$ ssthresh : Slow-start
      cwnd = cwnd + MSS
    - cwnd $>$ ssthresh : Additive Increase
      cwnd = cwnd + MSS$\frac{MSS}{cwnd}$

    TCP Reno: A torlódás detektálásához időzítőt és gyors újraadást is használ. [Gyors újraadás: ugyanazon csomaghoz 3 nyugta duplikátum érkezik (4 azonos nyugta), akkor újraküldi a csomagot és Slow-start fázisba lép.]

    Gyors újraadás után:

    - ssthresh = max$\{\frac{min\{wnd,cwnd\}}{2}$, 2MSS$\}$,
    - cwnd = sstresh + 3MSS.

    Gyors visszaállítás a gyors újraadás után minden további nyugta után növeli a rátát:

    - cwnd = cwnd + MSS.

Hatékonyság és Fairness:
Az átvitel maximális, ha a terhelés a hálózat kapacitását majdnem eléri. Ha a terhelés tovább nő, túlcsordulnak a pufferek, csomagok vesznek el, újra kell küldeni, drasztikusan nő a válaszidő. Ezt a torlódásnak nevezzük. Ezért a maximális terhelés helyett, ajánlatos a hálózat terhelését a könyök közelében beállítani. Itt a válaszidő csak lassan emelkedik, míg az adatátvitel már a maximum közelében van.

Egy jó torlódáselkerülési (angolul congestion avoidance) stratégia a hálózat terhelését a könyök közelében tartja: *hatékonyság.* Emellett fontos, hogy minden résztvevőt egyforma rátával szolgáljunk ki: *fairness*

Jelölje az i-edik résztvevő adatrátáját a t időpontban $x_i(t)$. Minden résztvevő aktualizálja az adatrátáját a $t+1$-ik fordulóban:

$$x_i(t+1) = f_0(t) \quad ha \ \sum_{i=1}^{n} x_i(t) \leq K$$

$$x_i(t+1) = f_1(t) \quad ha \ \sum_{i=1}^{n} x_i(t) > K$$

ahol $f_0(x) = a_I + b_I x$ a növelési, $f_1(x) = a_D + b_D x$ a csökkentési stratégia.

Speciális esetek

- **M**ultiplcative **I**ncrease **M**ultiplcative **D**ecrease - MIMD:

$$f_0(x) = b_I x \quad (b_I > 1) \qquad f_1(x) = b_D x \quad (b_D < 1)$$

- **A**dditive **I**ncrease **A**dditive **D**ecrease - AIAD:

$$f_0(x) = a_I + x \quad (a_I > 0) \qquad f_1(x) = a_D + x \quad (a_D < 0)$$

- **A**dditive **I**ncrease **M**ultiplcative **D**ecrease - AIMD:

$$f_0(x) = a_I + x \quad (a_I > 0) \qquad f_1(x) = b_D x \quad (b_D < 1)$$

Multiplexálás, demultiplexálás

Multiplexelés alatt a telekommunikációban azt az eljárást értik, amikor két vagy több csatornát összefognak egy csatornába úgy, hogy az inverz multiplexelés művelettel, vagy demultiplexeléssel, vagy demuxálással elő tudják állítani az eredeti csatornákat. Az eredeti csatornák egy úgynevezett kódolási sémával azonosíthatóak.

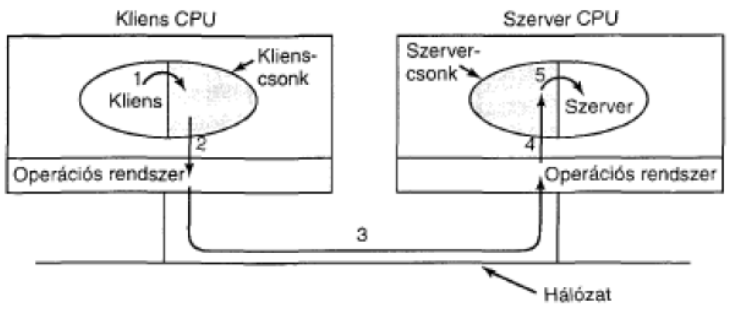
Interakciós modellek

- **Kétirányú bájtfolyam**
  Az adatok két egymással ellentétes irányú bájt-sorozatként kerülnek átvitelre. A tartalom nem interpretálódik Az adatcsomagok időbeli viselkedése megváltozhat: átvitel sebessége növekedhet, csökkenhet, más késés, más sorrendben is megérkezhetnek. Megpróbálja az adatcsomagokat időben egymáshoz közel kiszállítani. Megpróbálja az átviteli közeget hatékonyan használni.

- **RPC**
  A távoli gépen futtatandó eljárás eléréséhez hálózati kommunikációra van szükség, ezt az eljáráshívási mechanizmust az RPC (Remote Procedure Call) fedi el.

  A hívás lépései:

  1. A kliensfolyamat lokálisan meghívja a kliencsonkot.
  2. Az becsomagolja az eljárás azonosítóját és paramétereit, meghívja az OS-t.
  3. Az átküldi az üzenetet a távoli OS-nek.
  4. Az átadja az üzenetet a szervercsonknak.
  5. Az kicsomagolja a paramétereket, átadja a szervernek.
  6. A szerver lokálisan meghívja az eljárást, megkapja a visszatérési értéket.
  7. Ennek visszaküldése a klienshez hasonlóan zajlik, fordított irányban.



28. ábra. RPC

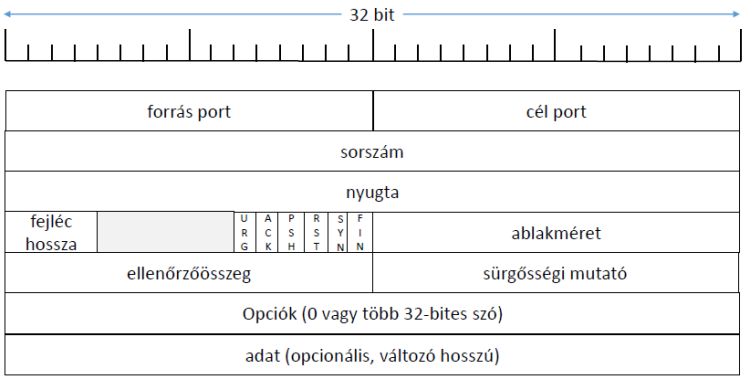
Protokollok

TCP

- Megbízható adatfolyam létrehozása két végpont között
- Az alkalmazási réteg adatáramát osztja csomagokra
- A másik végpont a csomagok fogadásról nyugtát küld

A TCP fejléc tartalma:

Küldő port	16 bit	A küldő folyamatot azonosítja
Cél port	16 bit	A címzett folyamat azonosítója
Sorszám	32 bit	Az első adatbájt sorszáma az aktuális szegmensen belül. Ha a SYN jelzőbit értéke 1, akkor ez a sorszám a kezdeti sorszám, azaz az első adatbájt sorszáma a kezdeti sorszám + 1 lesz.
Nyugtaszám	32 bit	Ha az ACK jelzőbit értéke 1, akkor a fogadó által következőnek fogadni kívánt sorszámot tartalmazza. Minden kapcsolat felépítés esetén elküldésre kerül.
Fejléc hossza	4 bit	A TCP fejléc hossza 32-bites egységekben.
Ablak	16 bit	A nyugtázott bájttal kezdődően hány bájtot lehet elküldeni. (A 0 érték is érvényes.)
Ellenőrzőösszeg	16 bit	Az adat-, fej-, és pszeudofejrész ellenőrzésére.
Opciók	0-40 bájt	A szabványos fejlécen kívüli lehetőségekre tervezték. Legfontosabb ilyen lehetőség az MSS, azaz a legnagyobb szegmens méret megadása. További opciók: MD5-aláírás, TCP-AO, "usertimeout", stb.
Sürgősségi mutató	16 bit	A sürgős adat bájtban mért helyét jelzi a jelenlegi bájtsorszámhoz viszonyítva.
Jelző bitek		URG – Sürgős jelzőbit. ACK – nyugta jelzés. PSH – Az jelzi, hogy gyors adattovábbítás kell a felhasználói rétegnek. RST – Kapcsolat egyoldalú bontását jelzi. SYN – Sorszám szinkronizációtjelez. FIN – Adatfolyam végét jelzi.



29. ábra. TCP Fejléc

TCP jellemzői:

- Kapcsolatorientált
- Megbízható
- Kétirányú bájtfolyam

UDP

Az UDP egy kapcsolat nélküli protokoll. A kommunikáció UDP csomagokkal (packet) történik, és a csomagok megérkezése nem garantált (elveszhetnek útközben). Továbbá, mivel az egyes csomagok akár más-más útvonalon is eljuthatnak a célhoz, a csomagok megérkezésének sorrendje is eltérhet a küldési sorrendtől.

- Egyszerű, nem megbízható szolgáltatás csomagok küldésére
- Az alkalmazási réteg határozza meg a csomag méretét
- Az inputot egy datagrammá alakítja

Összeköttetés nélküli protokoll. Olyan szegmenseket használ az átvitelhez, amelyek egy 8 bájtos fejrészből, valamint a felhasználói adatokból állnak.

A fejrész tartalmaz:

- egy forrásportot(2 bájt)
- egy célportot(2 bájt)
- egy UDP szegmens hossz értéket (2 bájt)
- egy UDP ellenőrzőösszeget (2 bájt)

Az UDP nem végez forgalomszabályozást, hibakezelést vagy újraküldést egy rossz szegmens fogadása után.

Kliens-szerver alkalmazások esetén kifejezetten hasznos lehet az UDP a rövid üzenetek miatt.

UDP-t alkalmaznak például olyan esetekben, ahol egy csomagot érdemes inkább eldobni, mint várni annak újraküldésére.

Ilyen lehet például: video-streaming, voice-over-IP alkalmazások (pl. internetes telefon), illetve bizonyos online játékok.