



Számítógépes Hálózatok

Laki Sándor

ELTE-Ericsson Kommunikációs Hálózatok Laboratórium

ELTE IK - Információs Rendszerek Tanszék

lakis@elte.hu

<http://lakis.web.elte.hu>

Eötvös Loránd
University



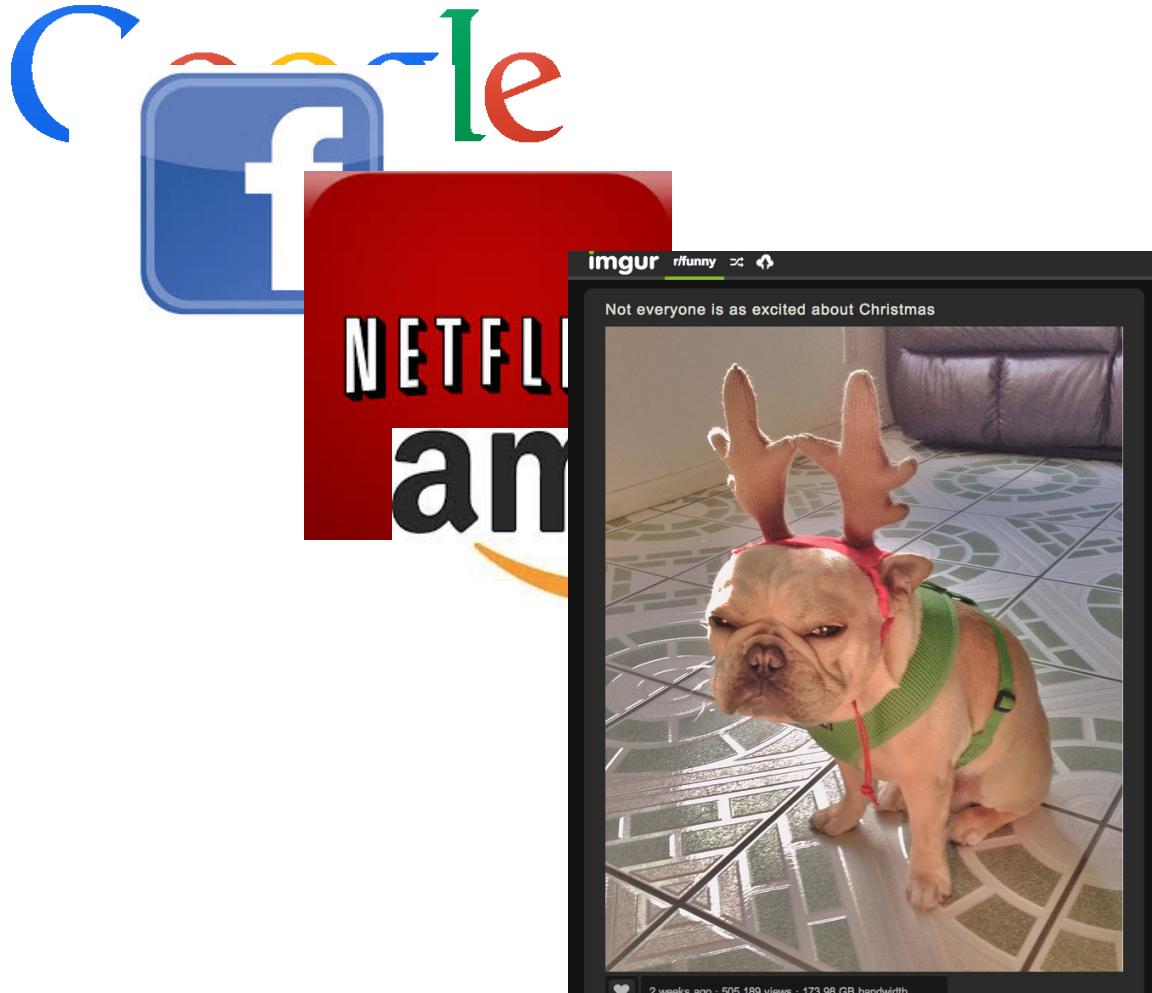
*Laurent Vanbever előadás alapján.
További inspiráció: Scott Shenker & Jennifer Rexford & Phillipa Gill*

Mi értelme ennek a tárgynak?

- Hányan nézték meg az e-mailjeiket, FB-ot, Twittert...
 - ma?
 - az elmúlt órában?
 - amióta elkezdtem beszélni?

A számítógépes hálózatok mindenhol jelen vannak

- A hálózatok az élet minden részét érintik
 - Web keresés
 - Közösségi hálók
 - Film nézés
 - Termékek rendelése
 - Időpocsékolás



A számítógépes hálózatok mindenhol jelen vannak

- A hálózatok az egyik legkritikusabb terület napjainkban
 - Hálózatok nélkül nem lenne...
 - Big Data
 - Cloud
 - Apps or Mobile Computing

Az Internet

egy igazán izgalmas hely...

17,1 milliárd

17,1 milliárd

az Internetre kötött eszközök száma 2016-ban (becslés*)

* Cisco Visual Networking Index 2016—2021

27,1 milliárd

az Internetre kötött eszközök száma 2021-ben (becslés*)

* Cisco Visual Networking Index 2016—2021

~3 exabájt

egy napi Internetes forgalom 2016-ban (becslés*)

[$1EB = 10^{18}$ Bájt = 1 000 000 000 000 000 000 000 Bájt]

* Cisco Visual Networking Index 2017

Ha  = 1 Gigabájt

An aerial photograph of the Great Wall of China, showing its winding path across a range of green, forested mountains. The wall itself is a light-colored, segmented structure that follows the contours of the hills.

~ 1 exabájt

~9 exabájt

egy napi Internetes forgalom 2021-ban (becslés*)

* Cisco Visual Networking Index 2017

~55%

video forgalom a teljes IP forgalomban 2016-ban (becslés*)

* Sandvine 2016 Global Internet Phenomena

Upstream		Downstream		Aggregate	
BitTorrent	18.37%	Netflix	35.15%	Netflix	32.72%
YouTube	13.13%	YouTube	17.53%	YouTube	17.31%
Netflix	10.33%	Amazon Video	4.26%	HTTP - OTHER	4.14%
SSL - OTHER	8.55%	HTTP - OTHER	4.19%	Amazon Video	3.96%
Google Cloud	6.98%	iTunes	2.91%	SSL - OTHER	3.12%
iCloud	5.98%	Hulu	2.68%	BitTorrent	2.85%
HTTP - OTHER	3.70%	SSL - OTHER	2.53%	iTunes	2.67%
Facebook	3.04%	Xbox One Games Download	2.18%	Hulu	2.47%
FaceTime	2.50%	Facebook	1.89%	Xbox One Games Download	2.15%
Skype	1.75%	BitTorrent	1.73%	Facebook	2.01%
	69.32%		74.33%		72.72%



* Sandvine 2016 Global Internet Phenomena

(<https://www.sandvine.com/hubfs/downloads/archive/2016-global-internet-phenomena-report-latin-america-and-north-america.pdf>)

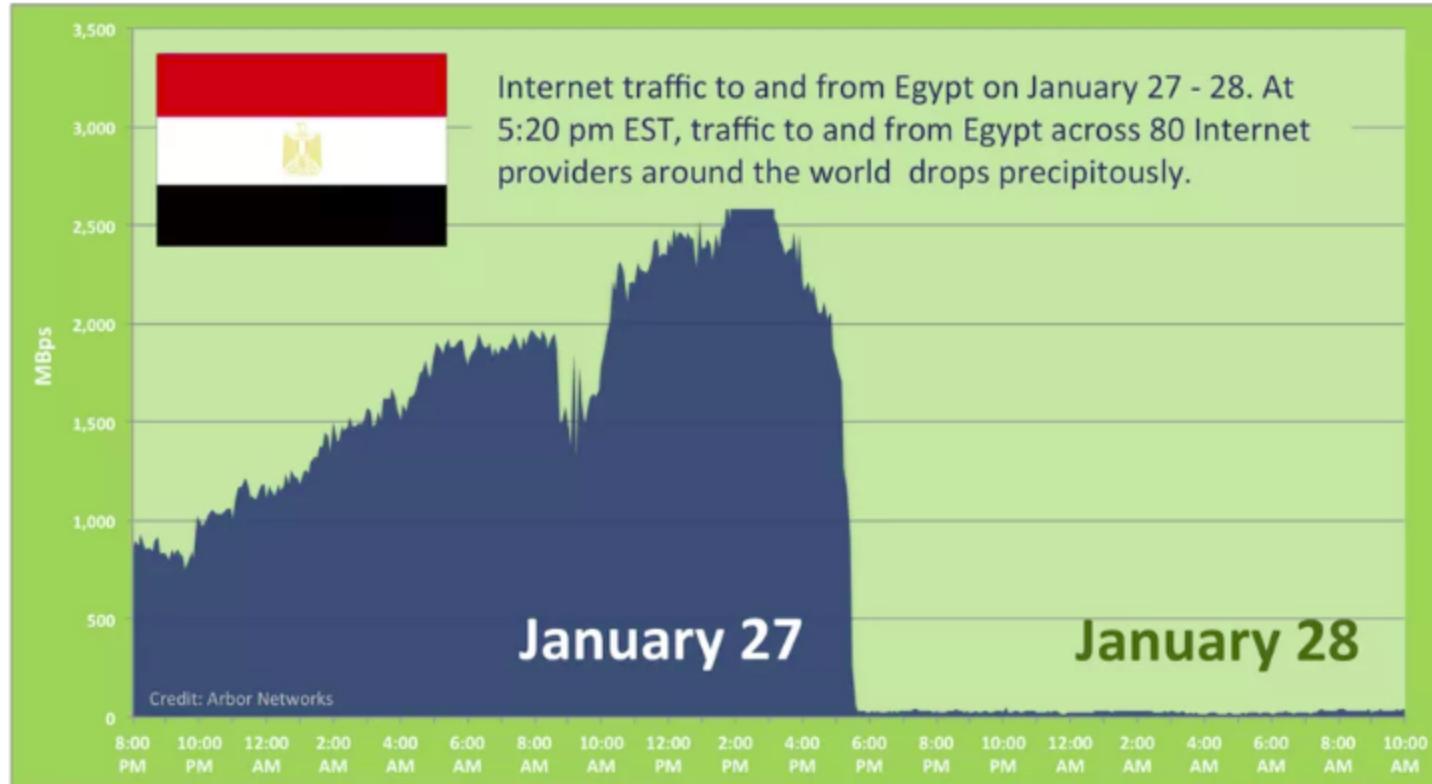
~80%

video forgalom a teljes IP forgalomban 2021-ben (becslés*)

* Cisco Visual Networking Index 2017

Van másik oldala is...

Internetes forgalom Egyiptomból/-ba (2011. január)



* <http://huff.to/1KxxoZF>

Szíria - 2013

NEWS / POLITICS

Syria cut off from global internet

International internet companies say Syria has been disconnected from "internet communication with the rest of world".

8 May 2013



* aljazeera.com

Governments shut down the internet more than 50 times in 2016

Economic impact alone was £1.9bn, with greater fears over human rights and freedom of speech



By MATT KAMEN

Tuesday 3 January 2017



Credit: Shutterstock

Governments around the world enacted internet shutdowns over 50 times in the last year, according to a report focusing on the impact of such draconian actions.

In purely monetary terms, the shutdowns resulted in economic slowdowns that cost a total of \$2.4bn (£1.9bn), according to research by The Brookings Institution, a Washington, DC-based

* <https://www.wired.co.uk/article/over-50-internet-shutdowns-2016>

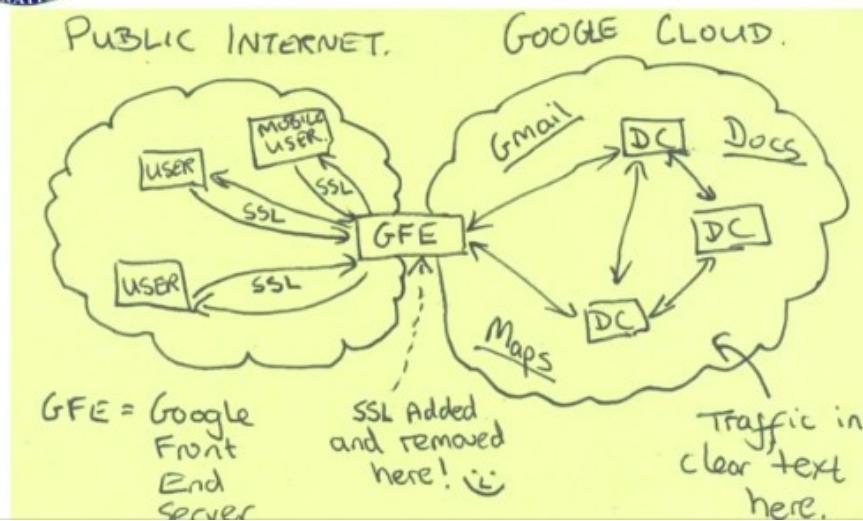
[National Security](#)

NSA infiltrates links to Yahoo, Google data centers worldwide, Snowden documents say

TOP SECRET//SI//NOFORN



Current Efforts - Google



* <http://wapo.st/1UVKamr>

The top-secret PRISM program

TOP SECRET//SI//ORCON//NOFORN

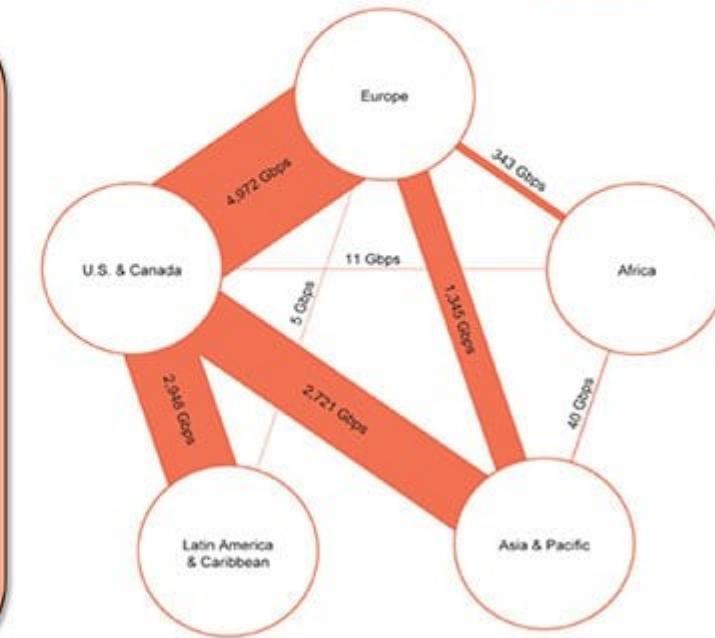


(TS//SI//NF) Introduction

U.S. as World's Telecommunications Backbone



- Much of the world's communications flow through the U.S.
- A target's phone call, e-mail or chat will take the **cheapest path, not the physically most direct** path – you can't always predict the path.
- Your target's communications could easily be flowing into and through the U.S.



International Internet Regional Bandwidth Capacity in 2011
Source: Telegeography Research

TOP SECRET//SI//ORCON//NOFORN

* <https://www.washingtonpost.com/wp-srv/special/politics/prism-collection-documents/>

„Netsemlegesség” – Network Neutrality

- Az Internet szolgáltató (ISP) szabadon eldöntheti-e, hogy mely forgalmakat lassítja?



The Federal Communications Commission is to take a more active role in regulating the Internet as a public utility, which is expected to provoke court cases from major broadband providers.

* <http://nyti.ms/2kZUnDA>



* <http://nyti.ms/2CkTbRR>



Netflix US

@netflix

Follow



We're disappointed in the decision to gut
#NetNeutrality protections that ushered in
an unprecedented era of innovation,
creativity & civic engagement. This is the
beginning of a longer legal battle. Netflix
stands w/ innovators, large & small, to
oppose this misguided FCC order.

10:26 AM - 14 Dec 2017

335,726 Retweets 831,986 Likes



7.1K

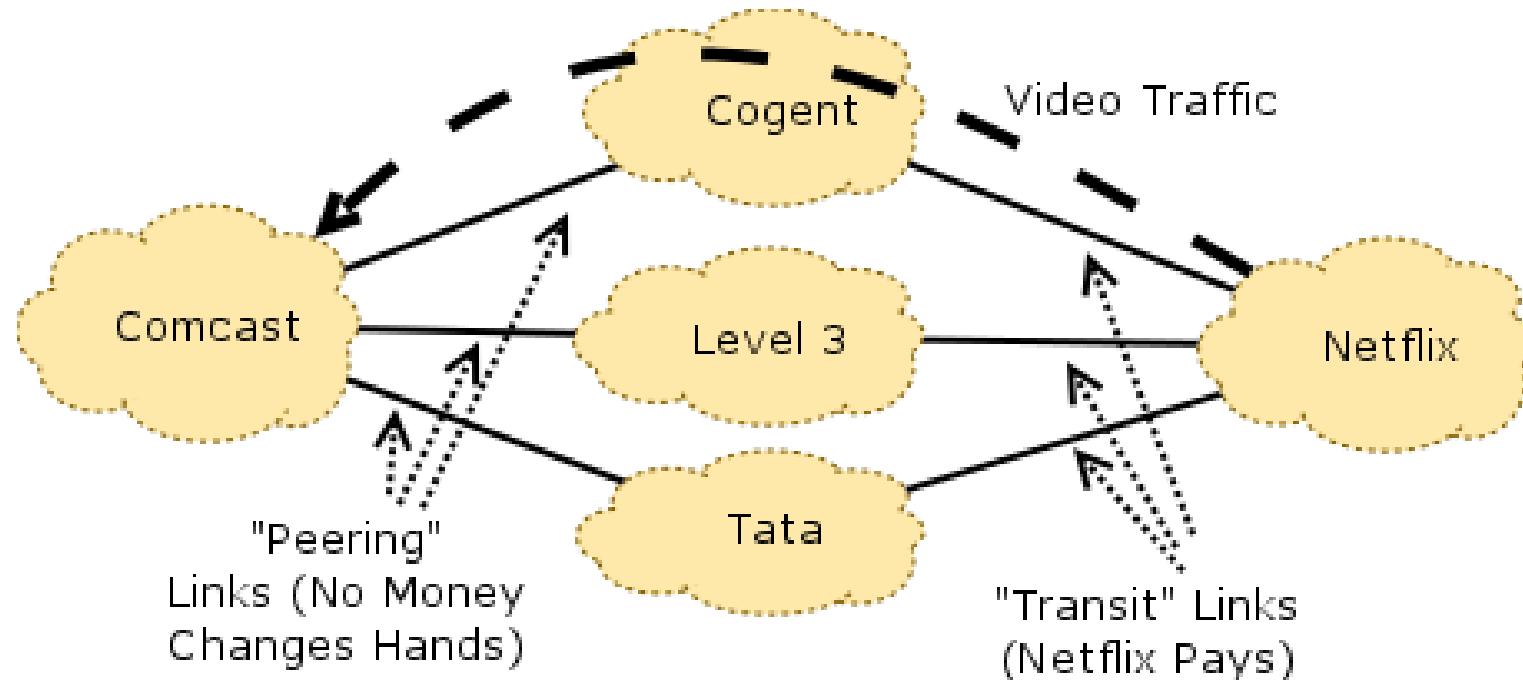
336K

832K

Kinek kell fizetni az Internet kapcsolatért?

Netflix VS ISPs

Előzmények – Comcast eset



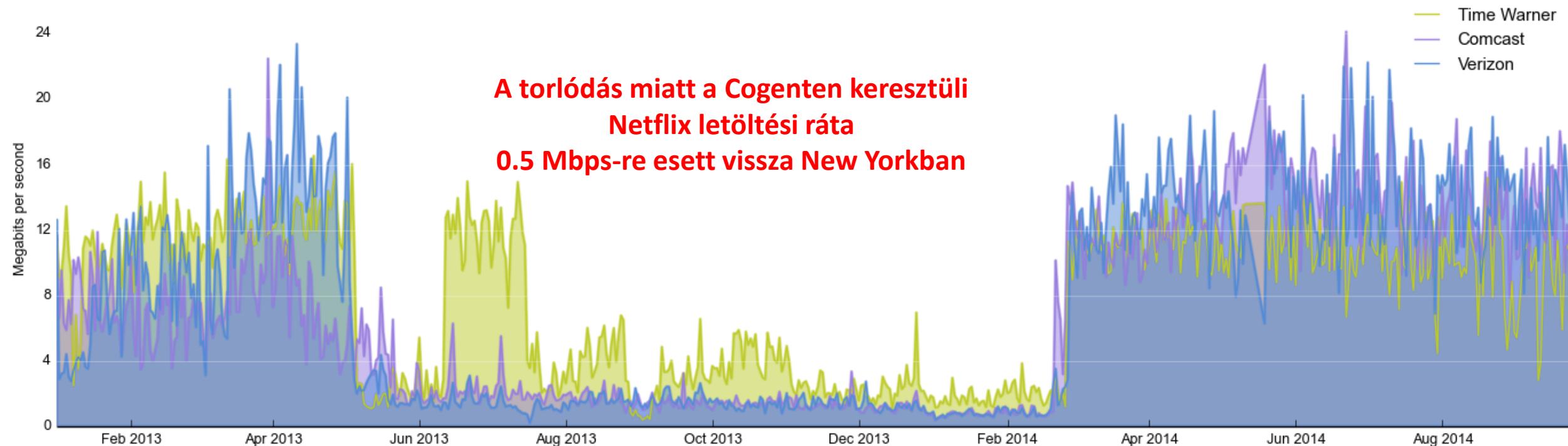
* <https://freedom-to-tinker.com/2015/03/25/why-your-netflix-traffic-is-slow-and-why-the-open-internet-order-wont-necessarily-make-it-faster/>



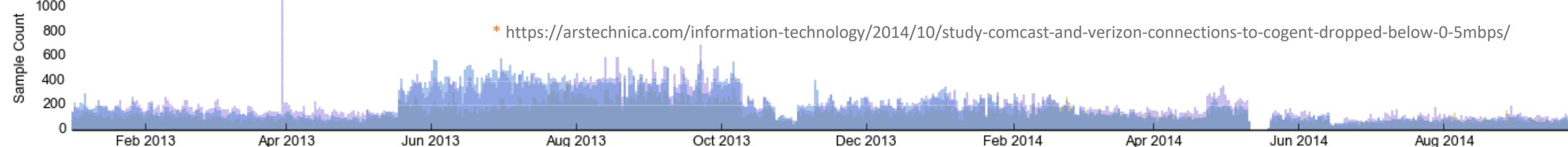
Kinek kell fizetni az Internet kapcsolatért?

Netflix VS ISPs

Median download throughput across Cogent in NYC over time from different ISPs (higher is better)



* <https://arstechnica.com/information-technology/2014/10/study-comcast-and-verizon-connections-to-cogent-dropped-below-0-5mbps/>



Kinek kell fizetni az Internet kapcsolatért?

Netflix VS ISPs

Median download throughput across Cogent in NYC over time from different ISPs (higher is better)



* <https://arstechnica.com/information-technology/2014/10/study-comcast-and-verizon-connections-to-cogent-dropped-below-0-5mbps/>

Sérülékenység



Widespread impact caused by Level 3 BGP route leak

Research // Nov 7, 2017 // Doug Madory

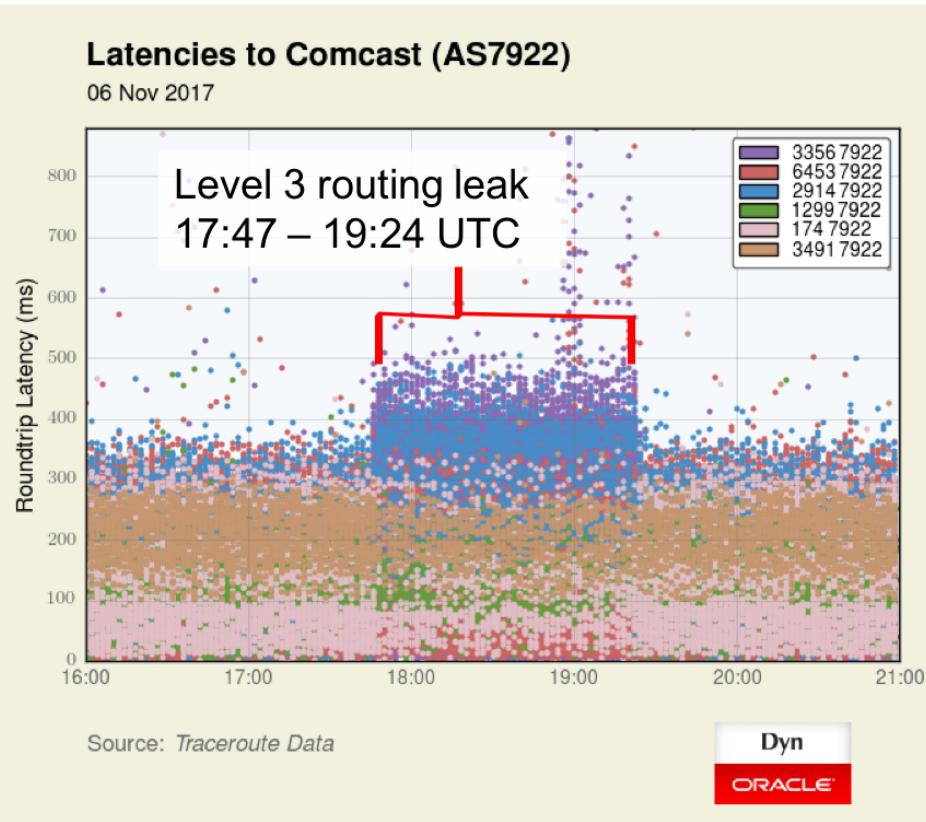
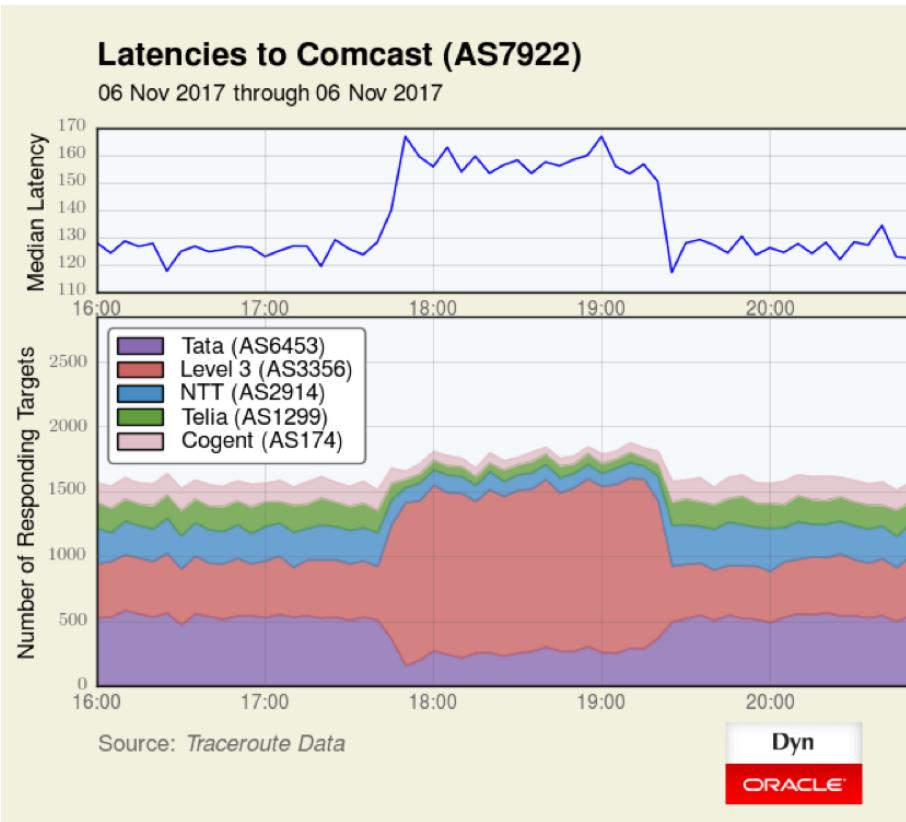
For a little more than 90 minutes yesterday, internet service for millions of users in the U.S. and around the world slowed to a crawl. Was this widespread service degradation caused by the latest botnet threat? Not this time. The cause was yet another BGP routing leak — a router misconfiguration directing internet traffic from its intended path to somewhere else.

* <https://dyn.com/blog/widespread-impact-caused-by-level-3-bgp-route-leak/>

Egy kis probléma...

- Egy kis időre kiesés történt
 - **több mint 90 perce**
- **Felhasználók millióit** érintette az USÁban és világszerte
- A probléma oka: BGP route szivárgás
 - **egy rosszul felkonfigurált router** az Internetes forgalmat nem az elvárt cél felé irányította, hanem valahova máshova





* <https://dyn.com/blog/widespread-impact-caused-by-level-3-bgp-route-leak/>

2017 augusztus



Data Centre ▶ Networks

Google routing blunder sent Japan's Internet dark on Friday

Another big BGP blunder

By Richard Chirgwin 27 Aug 2017 at 22:35

40 SHARE ▼

Last Friday, someone in Google fat-thumbed a border gateway protocol (BGP) advertisement and sent Japanese Internet traffic into a black hole.

The trouble began when The Chocolate Factory "leaked" a big route table to Verizon, the result of which was traffic from Japanese giants like NTT and KDDI was sent to Google on the expectation it would be treated as transit.

Since Google doesn't provide transit services, as BGP Mon explains, that traffic either filled a link beyond its capacity, or hit an access control list, and disappeared.

The outage in Japan only lasted a couple of hours, but was so severe

* https://www.theregister.co.uk/2017/08/27/google_routing_blunder_sent_japans_internet_dark/

A humán faktor

- Gyakran az ember a probléma...

11,353 views | Jul 8, 2015, 03:36pm

United Airlines Blames Router for Grounded Flights



Alexandra Talty Senior Contributor
Personal Finance

After a computer problem caused nearly two hours of grounded flights for United Airlines this morning and ongoing delays throughout the day, the airline announced the culprit: a **faulty router**.

Spokeswoman Jennifer Dohm said that the router problem caused "degraded network connectivity," which affected various applications.

A computer glitch in the airline's reservations system caused the Federal Aviation Administration to impose a groundstop at 8:26 a.m. E.T. Planes that were in the air continued to operate, but all planes on the ground were held. There were reports of agents writing tickets by hand. The ground stop was lifted around 9:47 a.m. ET.



Traders work on the floor of the New York Stock Exchange (NYSE) in July 2015.
(Photo by Spencer Platt/Getty Images)

DOWNTIME

UPDATED: "Configuration Issue" Halts Trading on NYSE

The article has been updated with the time trading resumed.

A second update identified the cause of the outage as a "configuration issue."

A third update added information about a software update that created the configuration issue.

A humán faktor

- Gyakran az ember a probléma...

11,353 views | Jul 8, 2015, 03:36pm

United Airlines Blames Router for Grounded Flights



Alexandra Talty Senior Contributor
Personal Finance

After a computer problem caused nearly two hours of grounded flights for United Airlines this morning and ongoing delays throughout the day, the airline announced the culprit: a faulty router.

Spokeswoman Jennifer Dohm said that the router problem caused "degraded network connectivity," which affected various applications.

A computer glitch in the airline's reservations system caused the Federal Aviation Administration to impose a groundstop at 8:26 a.m. E.T. Planes that were in the air continued to operate, but all planes on the ground were held. There were reports of agents writing tickets by hand. The ground stop was lifted around 9:47 a.m. ET.



„A hálózati kiesések 50-80%-át emberi tényező okozza.”

Jupiter Networks, What's Behind Network Downtime?, 2008



Traders work on the floor of the New York Stock Exchange (NYSE) in July 2015.
(Photo by Spencer Platt/Getty Images)

DOWNTIME

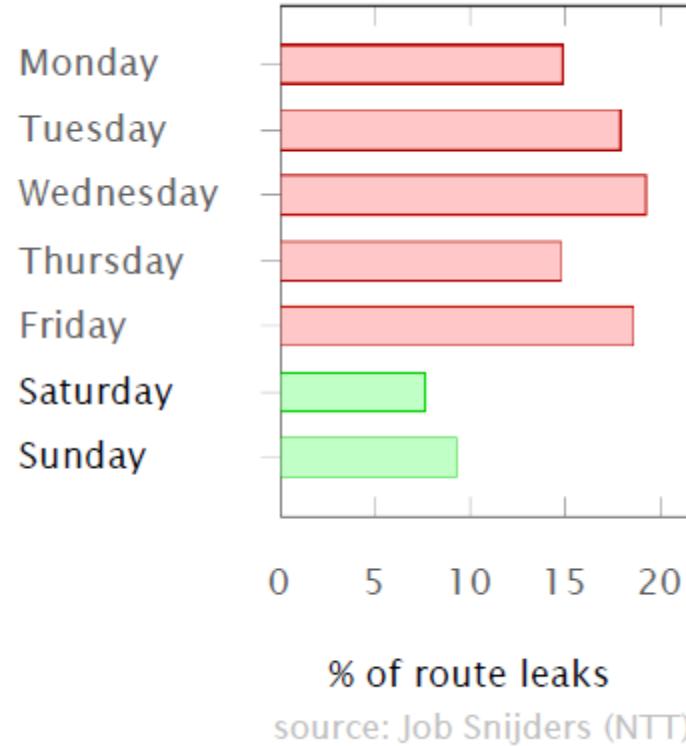
UPDATED: “Configuration Issue” Halts Trading on NYSE

The article has been updated with the time trading resumed.

A second update identified the cause of the outage as a “configuration issue.”

A third update added information about a software update that created the configuration issue.

Hétvégén minden jobban működik 😊



Számítógépes Hálózatok

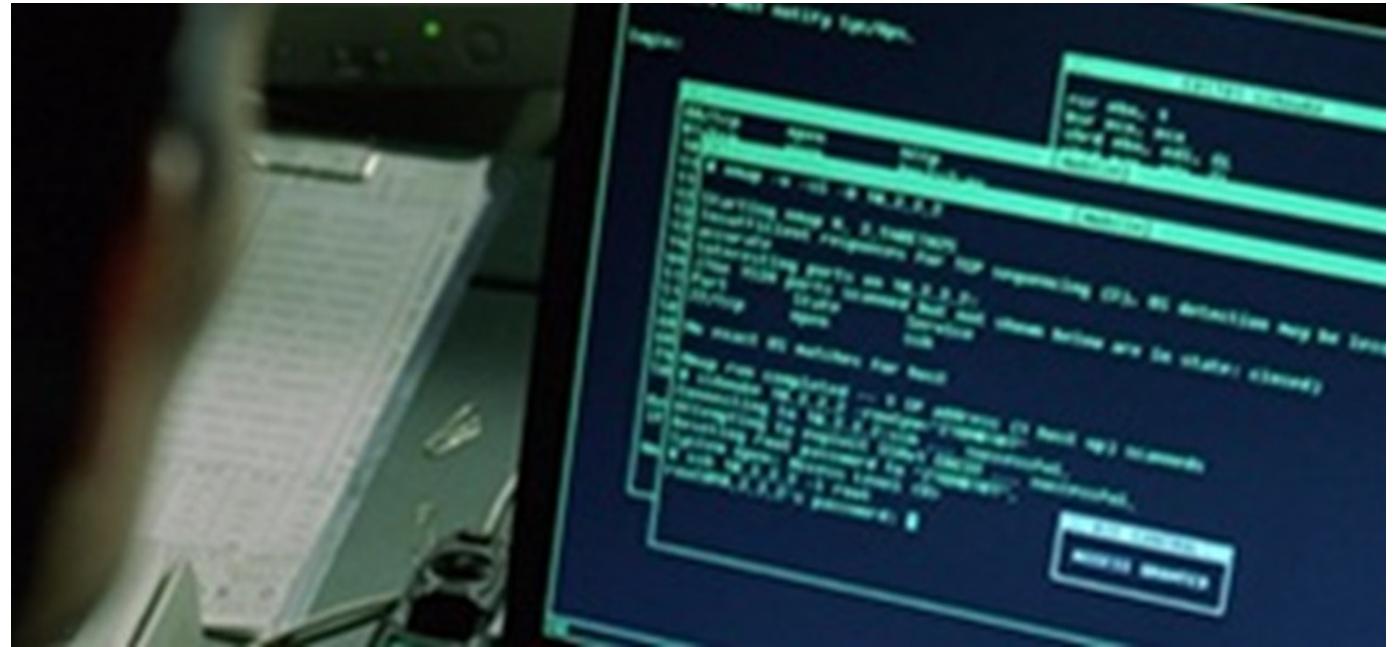
Egy kis logisztika

- Előadás
 - **Esti/Nappali:** *Kedd 16:15-17:45
Déli tömb, Mogyoródi terem*
- Előadó
 - Dr. Laki Sándor
 - Adjunktus, Információs Rendszerek Tanszék
 - lakis@inf.elte.hu
 - <http://lakis.web.elte.hu>
 - Iroda: Déli tömb, 2.506

A tárgy célja

- Megérteni, hogyan és egyáltalán miért működik az Internet...
- Főbb kérdések
 - Címzés
 - Hogyan címezhetők meg eszközök, szolgáltatások és protokollok?
 - Rétegek
 - Hogyan lehet a komplexitást kezelní?
 - Forgalomirányítás (routing)
 - Hogyan jutunk el A-ból B-be?
 - Megbízhatóság
 - Hogyan tudunk megbízhatóan üzenetet továbbítani megbízhatatlan közegeken (média) keresztül?
 - Erőforrás megosztás
 - Hogyan osszuk meg a korlátos hálózati erőforrásokat a versengő résztvevők között?

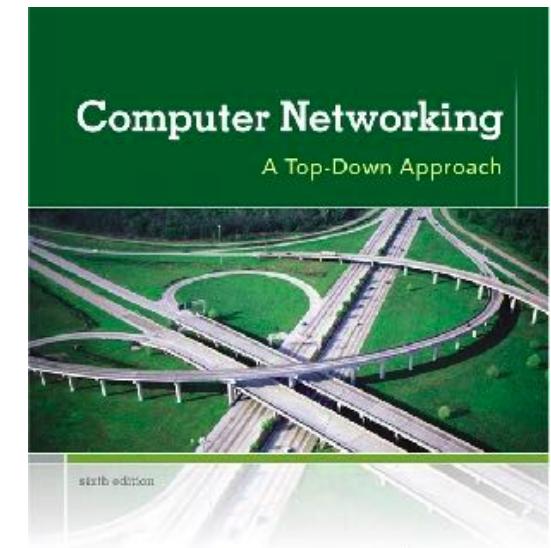
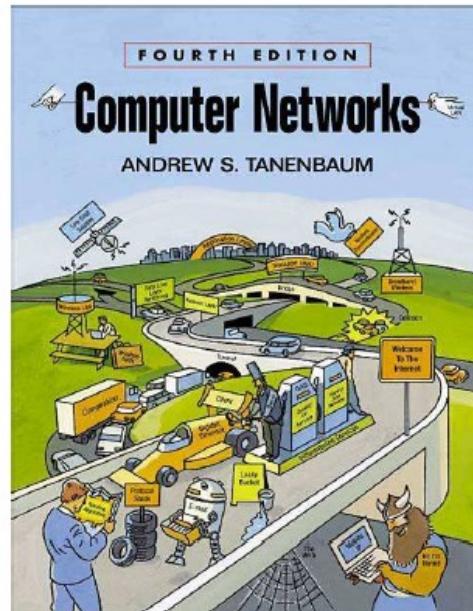
Skillek – amik a Mátrixban is elengedhetetlenek



Trinity a Matrix Reloaded-ből épp egy port szkent (**port scan**) futtat **nmap** segítségével...

Források

- A diák elérhetők:
 - <http://lakis.web.elte.hu>
- Könyvek → → → → →



Számonkérés - Vizsgajegy

- A vizsga előfeltétele a **legalább elégséges** gyakorlati jegy.
- A vizsga **írásbeli**, azaz az egész féléves anyagra épülő elméleti és gyakorlati feladatokból összeállított kérdéssor kitöltését jelenti. A vizsga időtartama **60 perc**.
- **Teszt részből és kifejtős részből áll.**
- A teszt rész esetén **60% minimum követelménnyel!** [beugró is egyben]
- A féléves anyag a fóliákon is szereplő fogalmakat, összefüggéseket és a belőlük levonható következtetéseket jelenti.
- **Értékelés**
 - [85%, 100%] – jeles(5)
 - [75%, 85%) – jó(4)
 - [60%, 75%) – közepes(3)
 - [50%, 60%) – elégséges(2)
 - [0%, 50%) – elégtelen(1)

Tegyük fel, hogy begépeled a böngészőbe, hogy www.google.com

Nyomd le az entert...

Mi történik ekkor?

Fel tudnád sorolni az összes

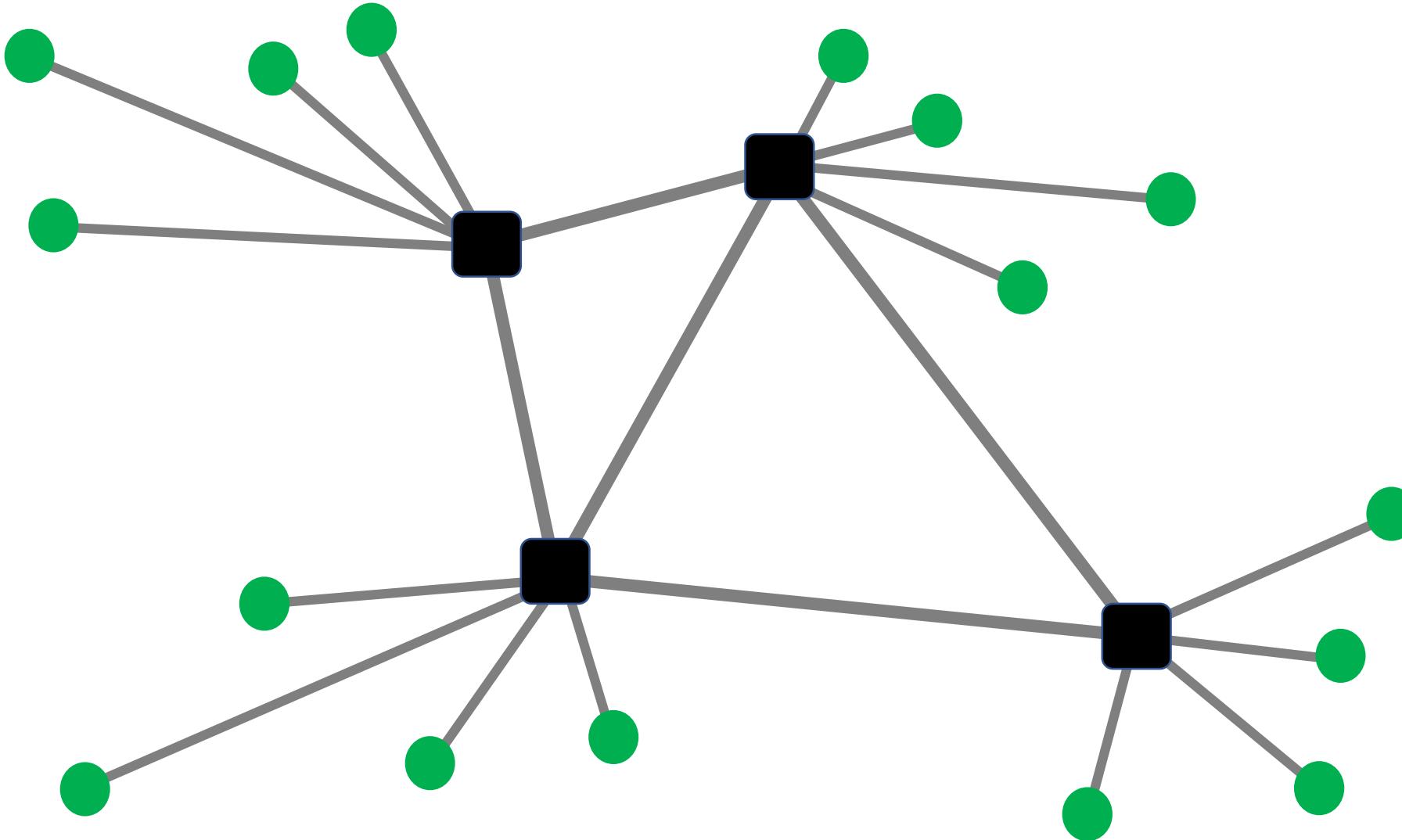
technológiát, alapelvet, protokollt, alkalmazást...

ami felhasználásra kerül???

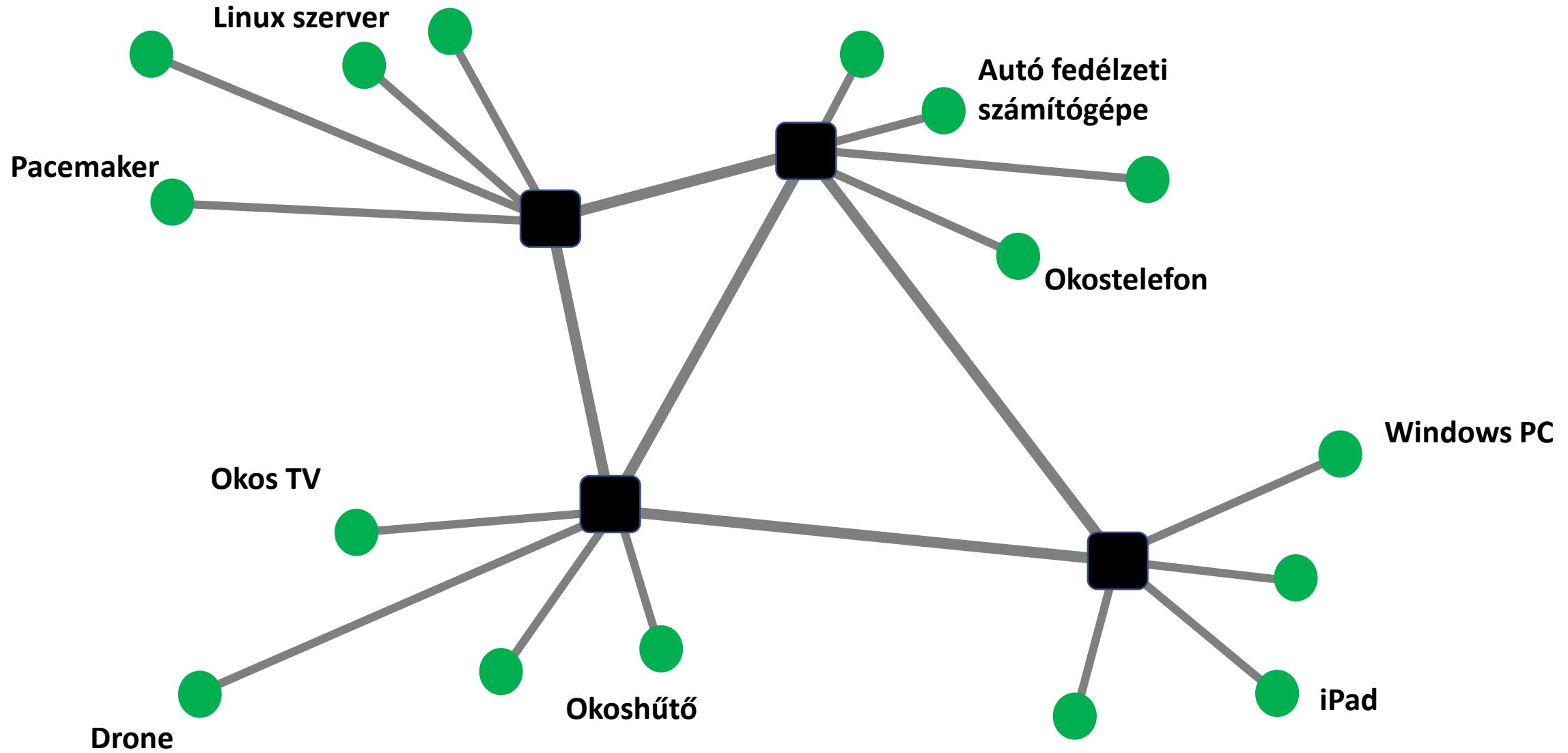
Áttekintés

Milyen elemekből épülnek fel a hálózatok?

Három alapvető komponens

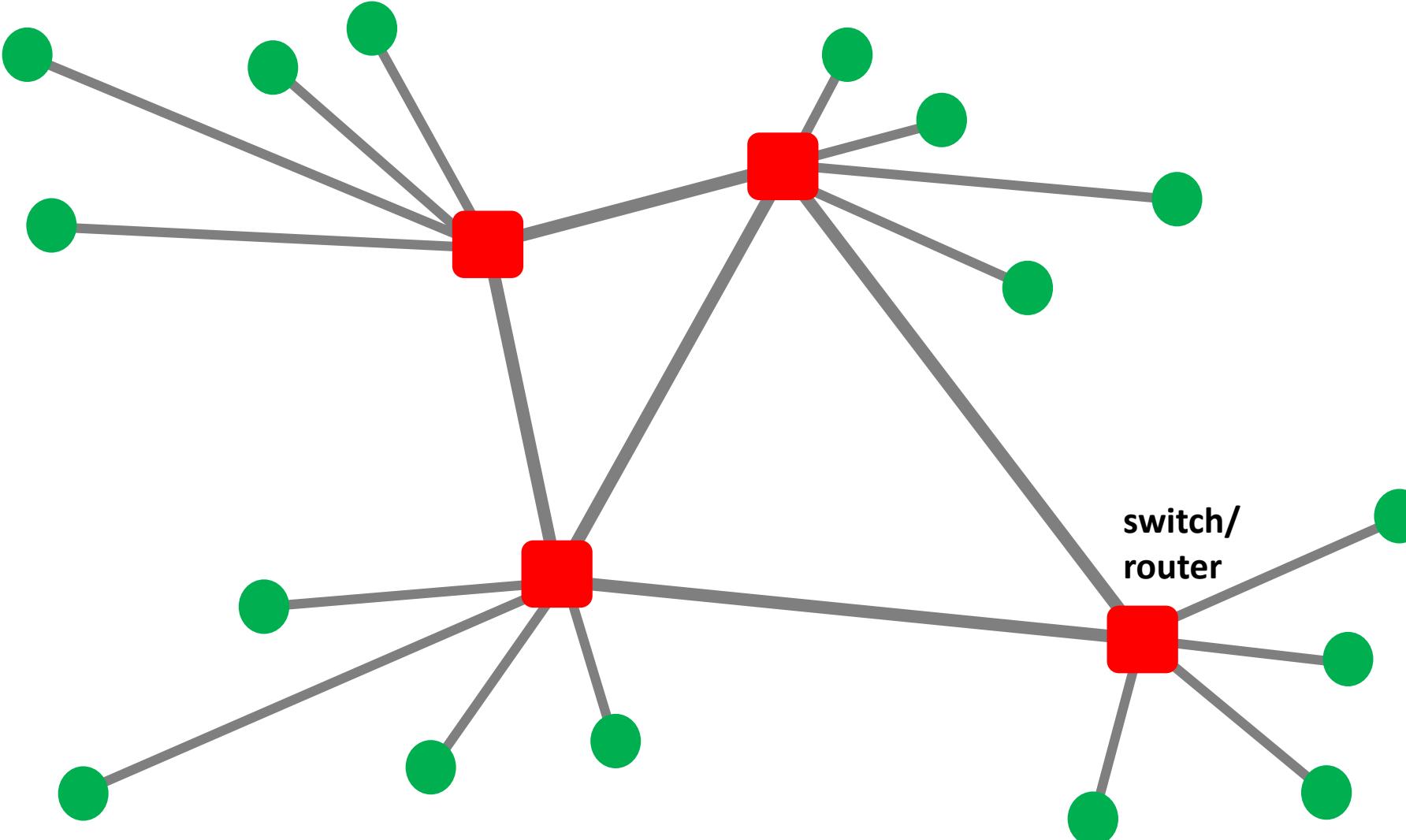


Adat küldő és fogadó **végpontok**



Adattovábbító **switchek** és **routerek**

Adat továbbítása a célállomás felé



A routerek mérete, képességei és felhasználása is változó

Otthoni router

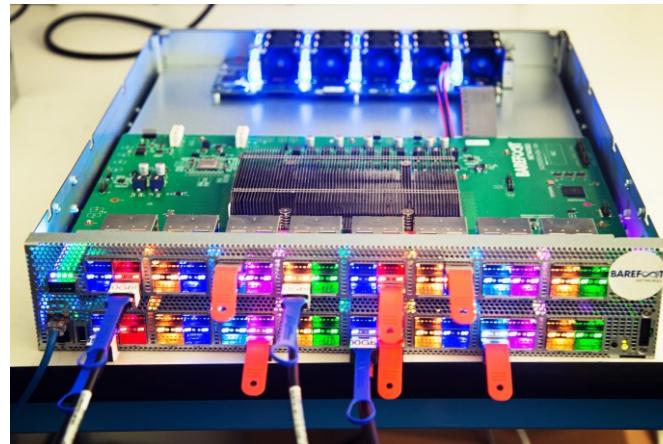


~20 cm

0,5 kg

1 Gbps

ToR switch/router



~50cm

~ 5 kg

1,8-6,5 Tbps

Internet core
router



>200 cm

~700 kg

12,8 Tbps

(up to 922 Tbps)

Kbps? Mbps? Tbps?

- **Hálózati sávszélesség**

Az adat átviteléhez elérhető vagy felhasznált kommunikációs erőforrás mérésére szolgáló mennyiség, amelyet bit per másodpercben szoktak kifejezni.

SI szabvány

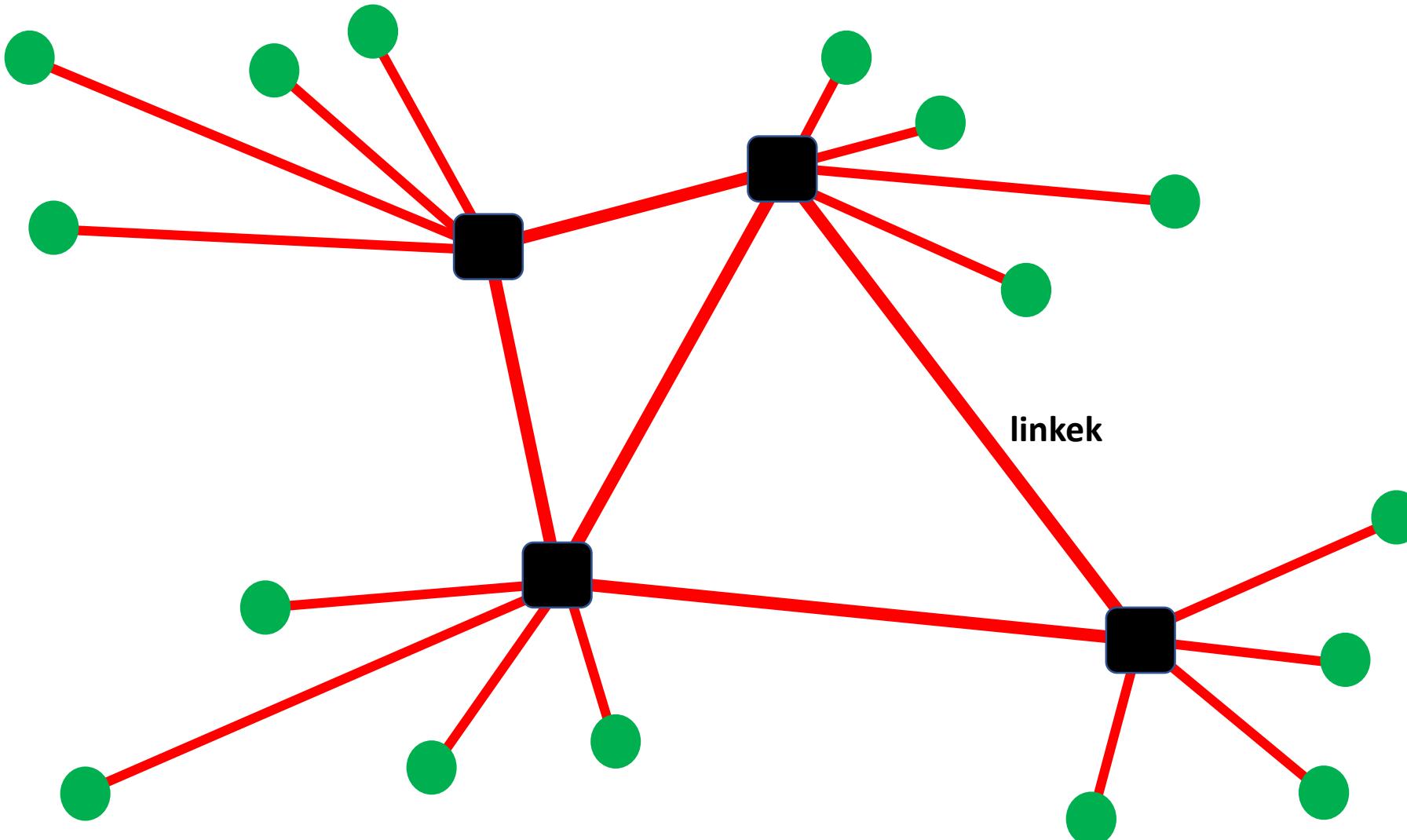
$8 \cdot 10^3$ bit/sec	1 KB/s	egy kiló-bájt
$8 \cdot 10^6$ bit/sec	1 MB/s	egy mega-bájt
$8 \cdot 10^9$ bit/sec	1 GB/s	egy giga-bájt
$8 \cdot 10^{12}$ bit/sec	1 TB/s	egy terra-bájt
$8 \cdot 10^{15}$ bit/sec	1 PB/s	egy peta-bájt
$8 \cdot 10^{18}$ bit/sec	1 EB/s	egy exa-bájt

IEC szabvány

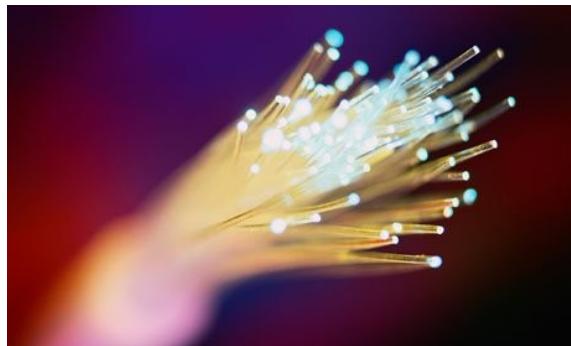
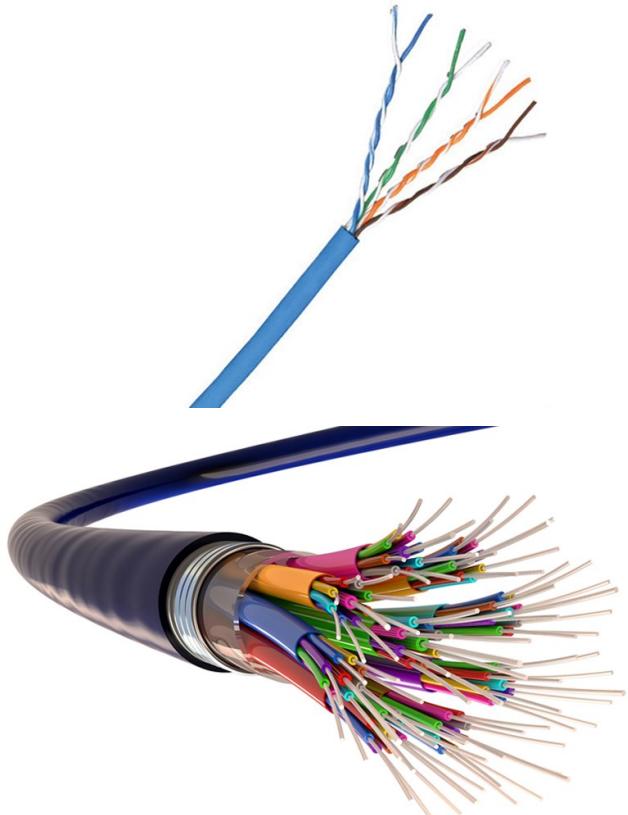
$8 \cdot 2^{10}$ bit/sec	1 KiB/s	egy kibi-bájt
$8 \cdot 2^{20}$ bit/sec	1 MiB/s	egy mebi-bájt
$8 \cdot 2^{30}$ bit/sec	1 GiB/s	egy gibi-bájt
$8 \cdot 2^{40}$ bit/sec	1 TiB/s	egy tebi-bájt
$8 \cdot 2^{50}$ bit/sec	1 PiB/s	egy pebi-bájt
$8 \cdot 2^{60}$ bit/sec	1 EiB/s	egy exbi-bájt

Linkek

a végpontokat kapcsolják a switchekhez és a switcheket egymáshoz



Linkek - példák

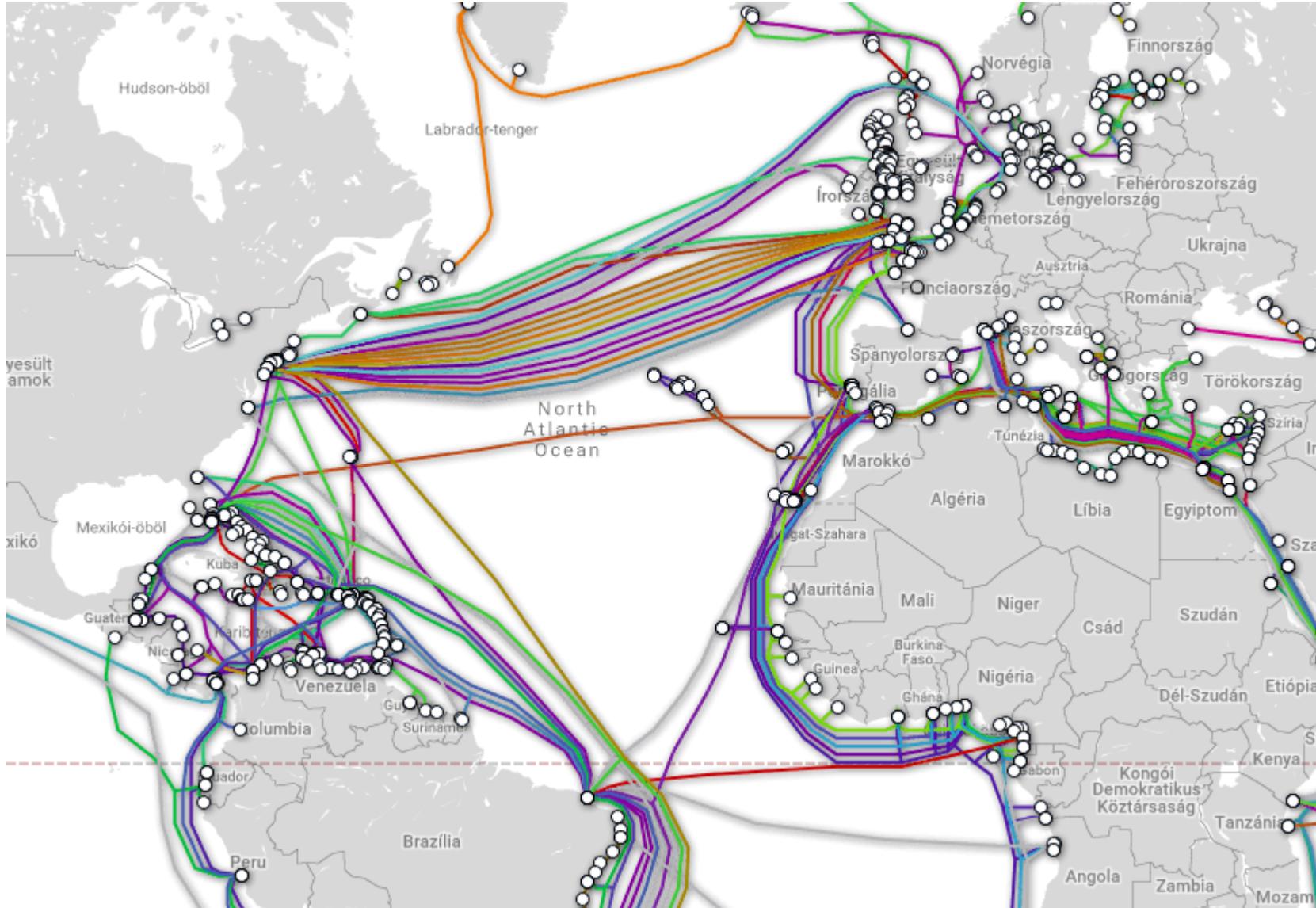


Rézvezeték
ADSL, RJ-45, Coax

Optikai szál



Vezetéknélküli



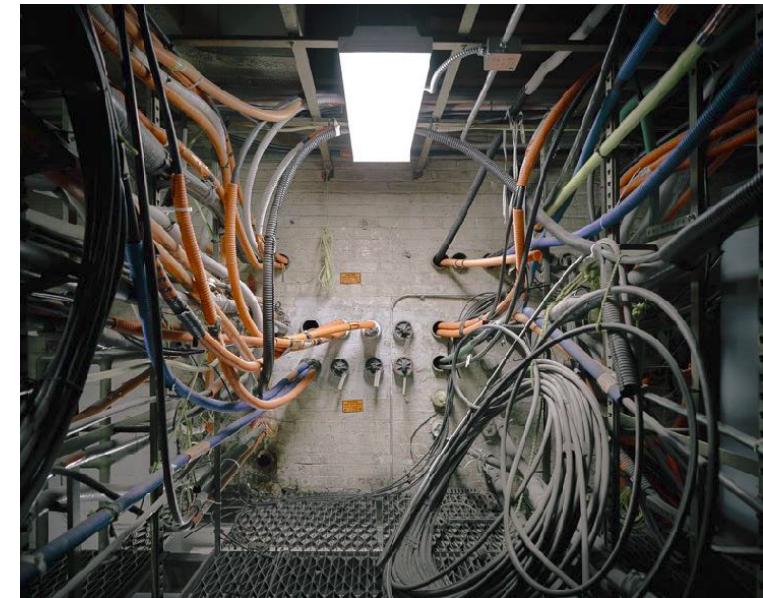
* <https://www.submarinecablemap.com/>



Mélytengeri kábelek
javítása



René Descart
mélytengeri kábelfektető hajó

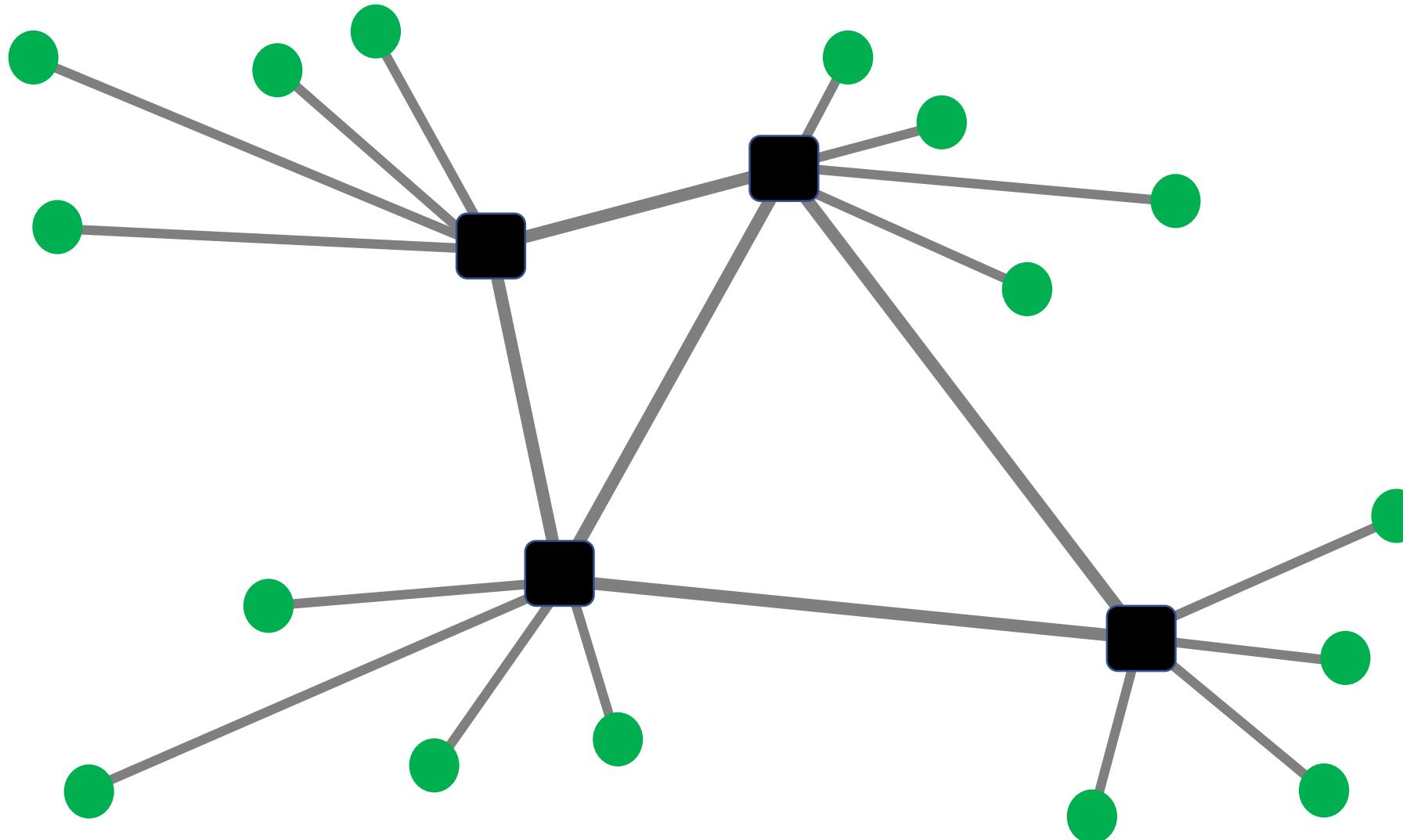


Valahol Manhattenben

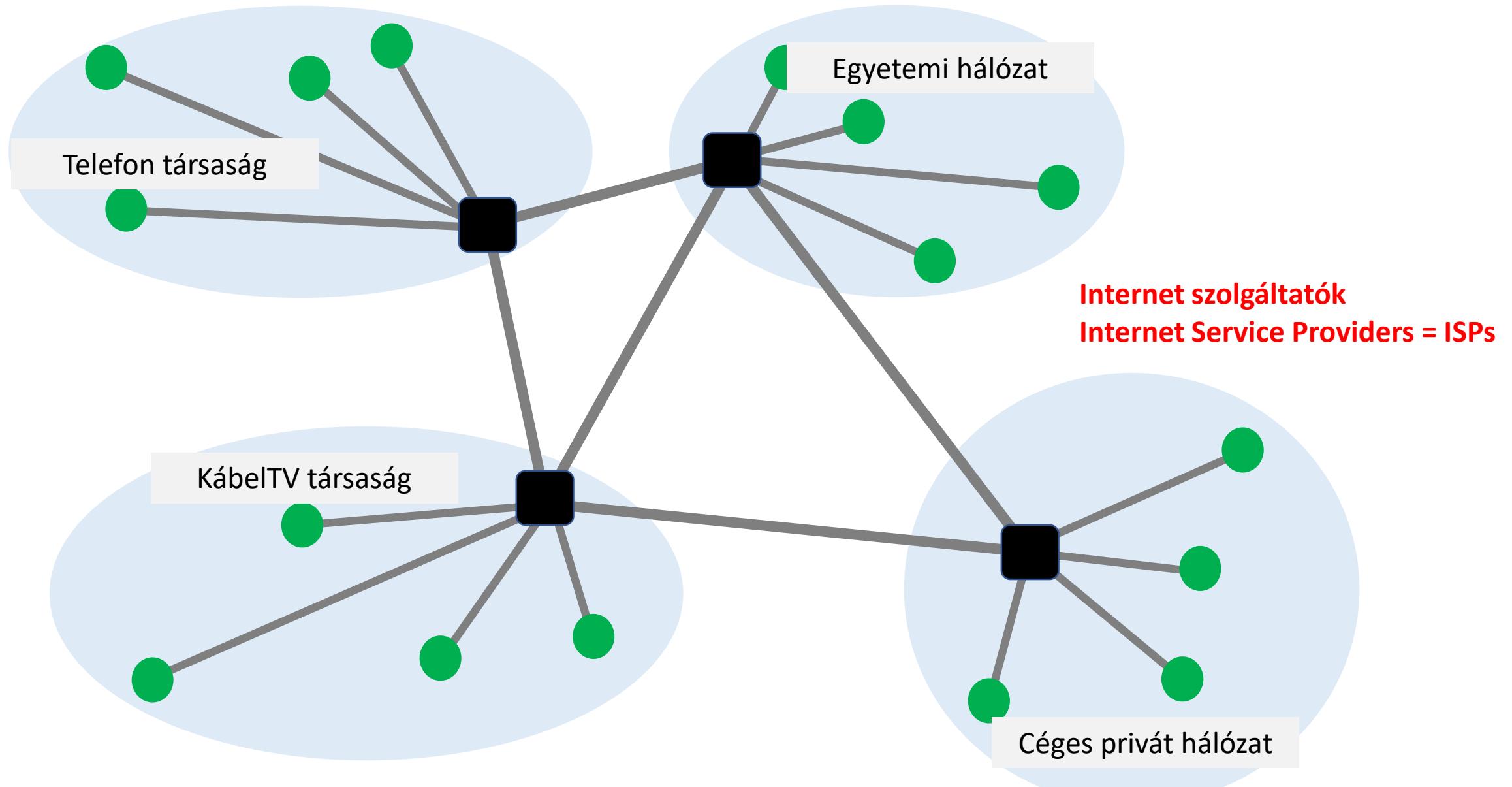
További fogalmak

- **Hálózati hoszt**
 - Olyan eszköz, amely egy számítógépes hálózattal áll összeköttetésben. Egy hoszt információkat oszthat meg, szolgáltatást és alkalmazásokat biztosíthat a hálózat további csomópontjainak. (Továbbiakban csak hosztként hivatkozunk rá.)
- **Átviteli csatorna, médium, fizikai közeg**
 - Az a közeg, amelyen a kommunikáció folyik a résztvevő hosztok között. Ez a közeg lehet egy koaxális kábel, a levegő, optikai kábel, stb.

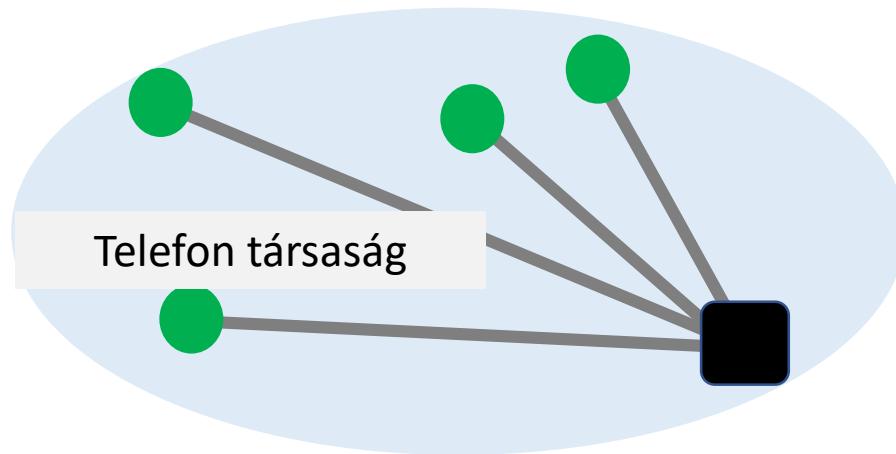
Internet = hálózatok hálózata



Internet = hálózatok hálózata



Internet kapcsolat telefon hálózat felett

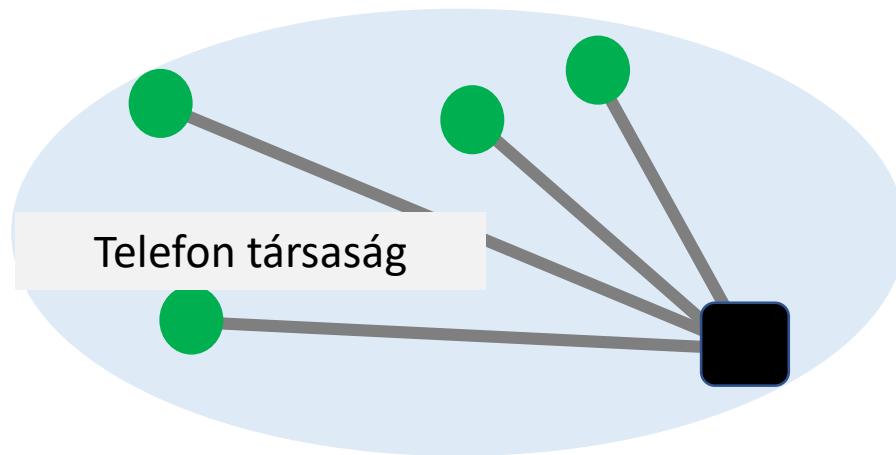


Digital Subscriber Line (DSL)

- Nagy sávszélességű hozzáférés biztosítása háztartások számára **telefon vonalon** keresztül

?????????????????

Internet kapcsolat telefon hálózat felett



Digital Subscriber Line (DSL)

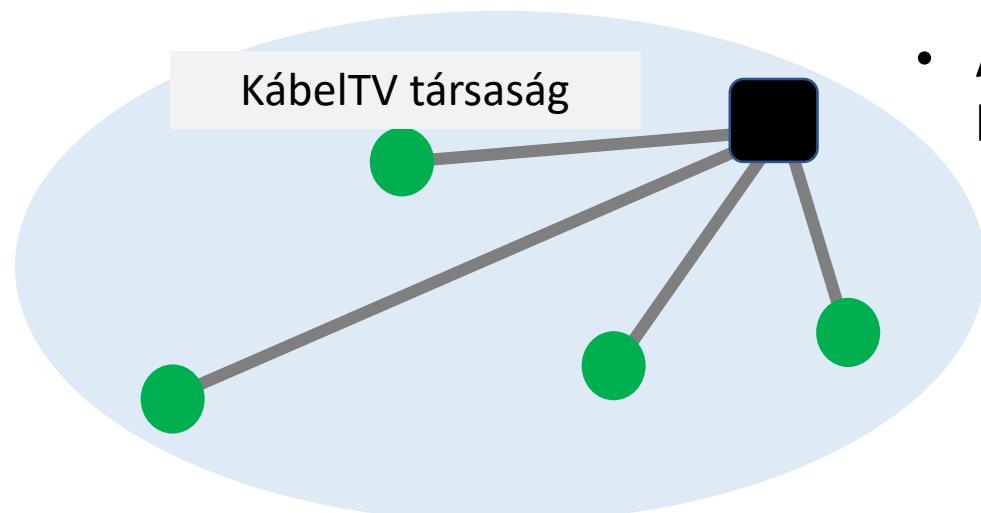
- Nagy sávszélességű hozzáférés biztosítása háztartások számára **telefon vonalon** keresztül
- 3 csatorna
 - **Letöltési csatorna** (downstream data channel)
 - Max. néhány száz Mbps
 - **Feltöltési csatorna** (upstream data channel)
 - Max. néhányszor tíz Mbps
 - **Két-irányú telefon csatorna**
 - Csak hangátvitel

Miért aszimmetrikus a kapcsolat?

Kábel TV hálózaton keresztül



Réz koax cábel



Cable Access Technology (CATV)

- Nagy sávszélességű hozzáférés biztosítása háztartások számára **kábel TV hálózaton** keresztül
- **Letöltési csatorna** (downstream data channel)
 - Max. néhány száz Mbps
- **Feltöltési csatorna** (upstream data channel)
 - Max. néhányszor tíz Mbps
- Az ADSL-lel ellentétben a közeg **meg van osztva** a háztartások között.

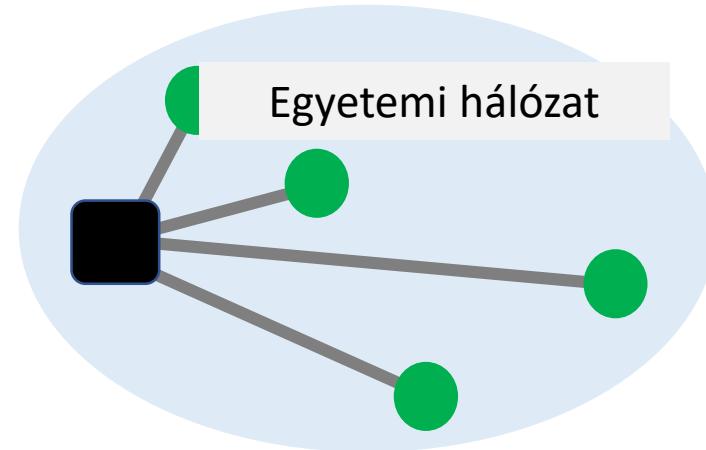
Az Ethernet a leggyakrabban használt Helyi Hálózati - Local Area Network (LAN) technológia



Sodort érpár
(Twisted pair copper)



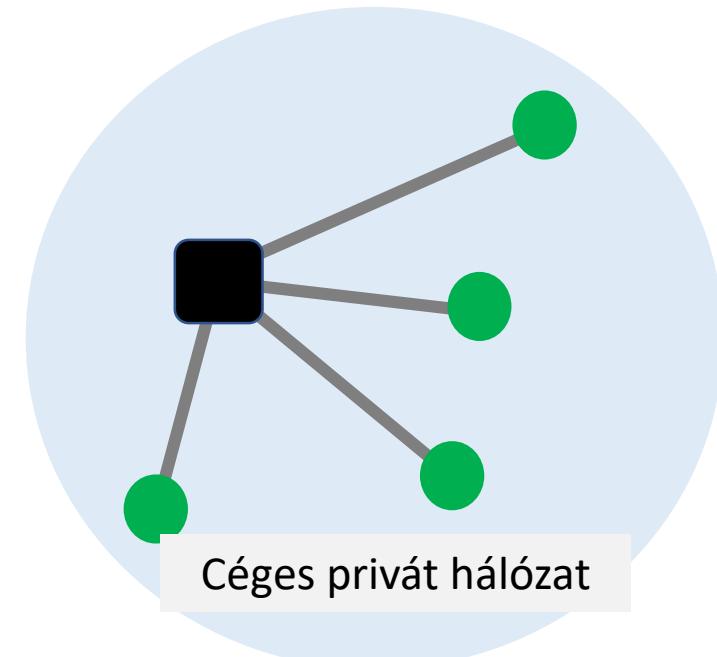
Optikai kábel
(SFP+ Active Optical)



1 Gbps, 10 Gbps, 40 Gbps, 100 Gbps, ...
Szimmetrikus – full-duplex



100 Gbps hálókártya (NIC)



Vannak további technológiák is

- **Mobil** – okostelefonok
- **Műholdas** – távoli elérés ott is, ahol nincs infrastruktúra
- **FTTH** – háztartások
- **Optikai kábelek** (fibers, dark fibers) – Internet gerinchálózatok
- **Infiniband** – HPC klaszterek
- ...

Áttekintés

Hogyan osszuk meg az erőforrásokat?

Hogy néz ki az Internet „belseje”?

3 fontos követelmény a hálózat topológiájával kapcsolatban

- **Hibatolerancia**

- Több útvonal a források és célok között

- **Rugalmasság**

- Erőforrásmegosztás költséghatékonyúság és megvalósíthatóság érdekében
- Azaz a linkek száma nem lehet túl nagy

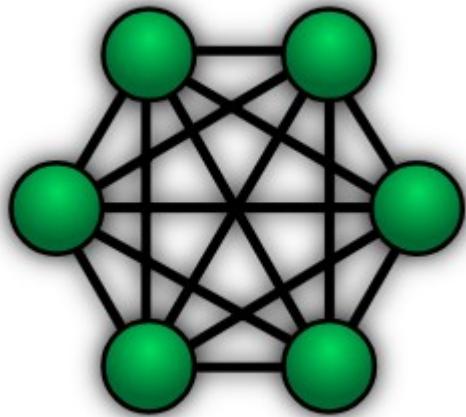
- **Megfelelő csomópont-kapacitás**

- Azaz a linkek száma nem lehet túl kicsi

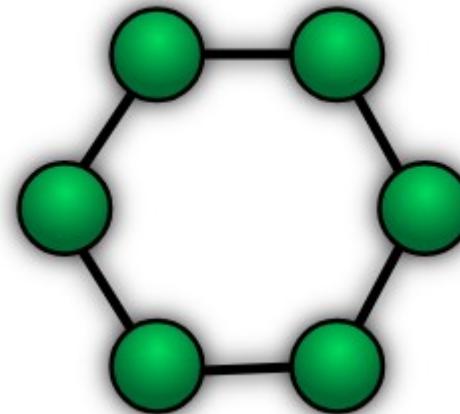
Nézzünk néhány speciális topológiát...

Az Internet nem ilyen

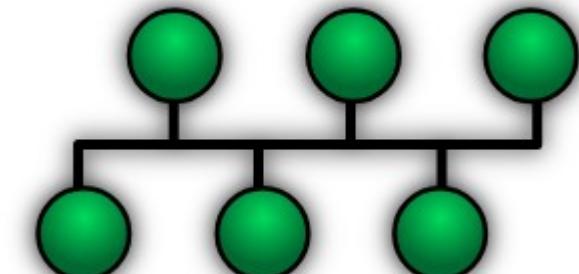
Teljesen
összkötött
(Full-mesh)



Lánc/Gyűrű



Busz

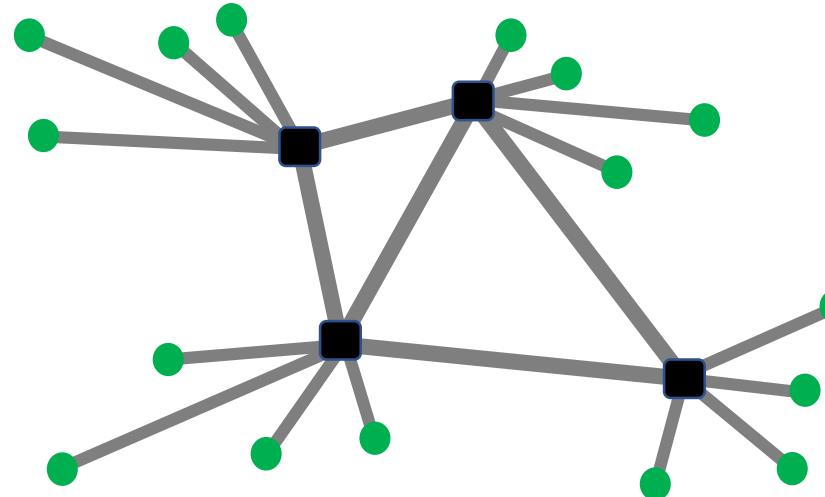


Előnyök

Hátrányok

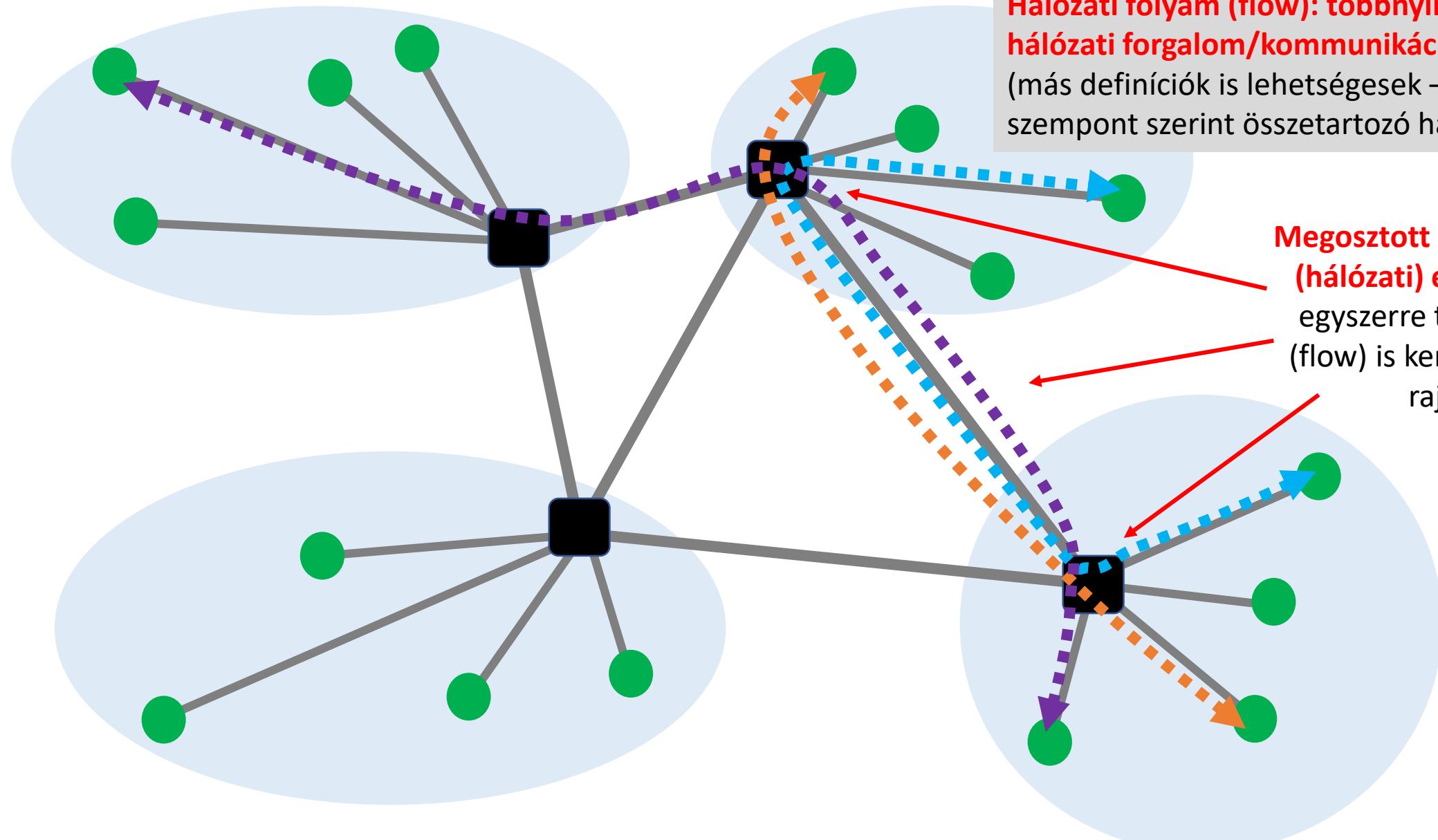
A kompromisszumos megoldás: switchelt/kapcsolt hálózatok (switched networks) ésszerűség és rugalmasság

Switchelt/kapcsolt kialakítás



- | | |
|------------------|--|
| Előnyök | Az erőforrás megosztás és csomópontok kapacitása úgy alakítható, hogy megfeleljön a hálózati igényeknek. |
| Hátrányok | Okos eszközöket igényel, melyek támogatják a csomagtovábbítást , forgalomirányítást és az erőforrás kiosztást . |

Linkek és switchek megosztott használata



Hálózati folyam (flow): többnyire két fél közötti hálózati forgalom/kommunikáció

(más definíciók is lehetségesek – valamilyen szempont szerint összetartozó hálózati forgalom)

Megosztott link és switch (hálózati) erőforrások:
egyszerre több folyam (flow) is keresztül halad rajtuk

Erőforrások kezelése

Két alapvetően eltérő megközelítés hálózati

Előre foglalással

**Előre lefoglalja a szükséges
sávszélességet.**

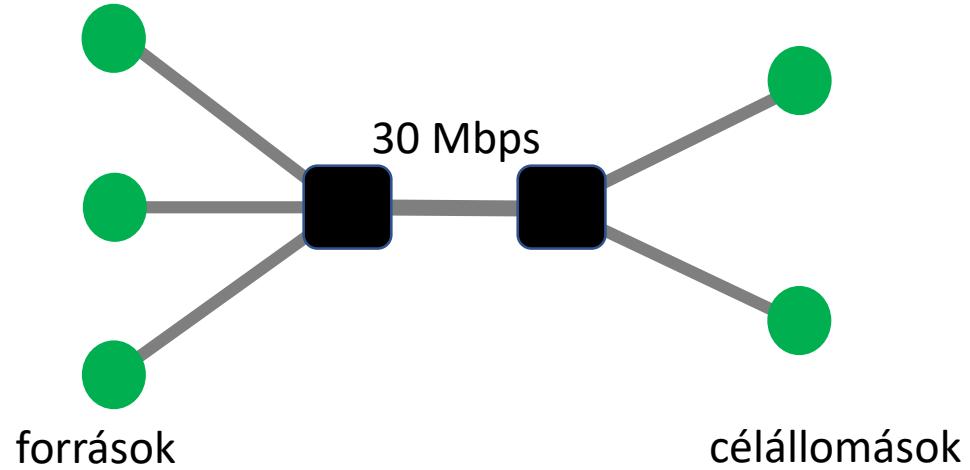
Folyam szintű multiplexálás

Igény szerinti

**Akkor küld adatot, amikor
szükséges.**

Csomag szintű multiplexálás

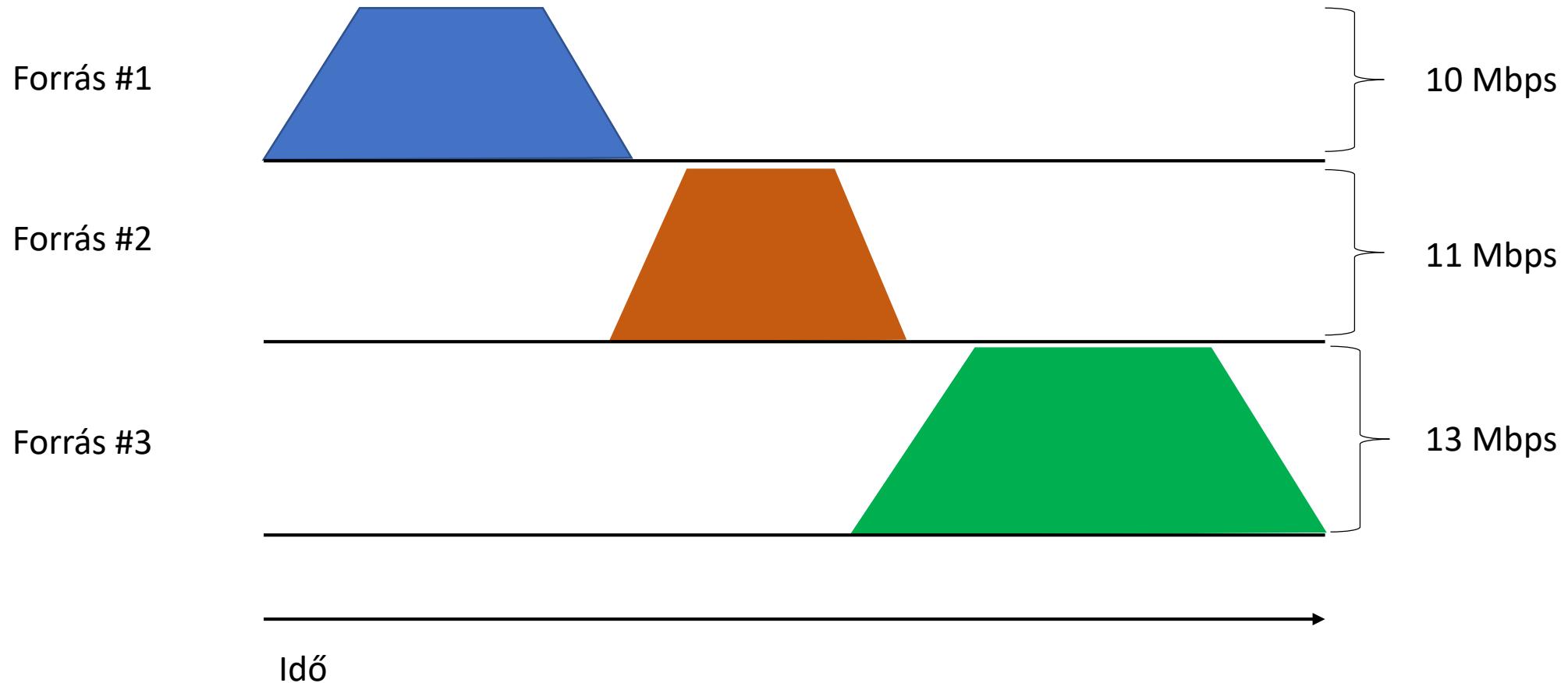
Melyik a jobb?



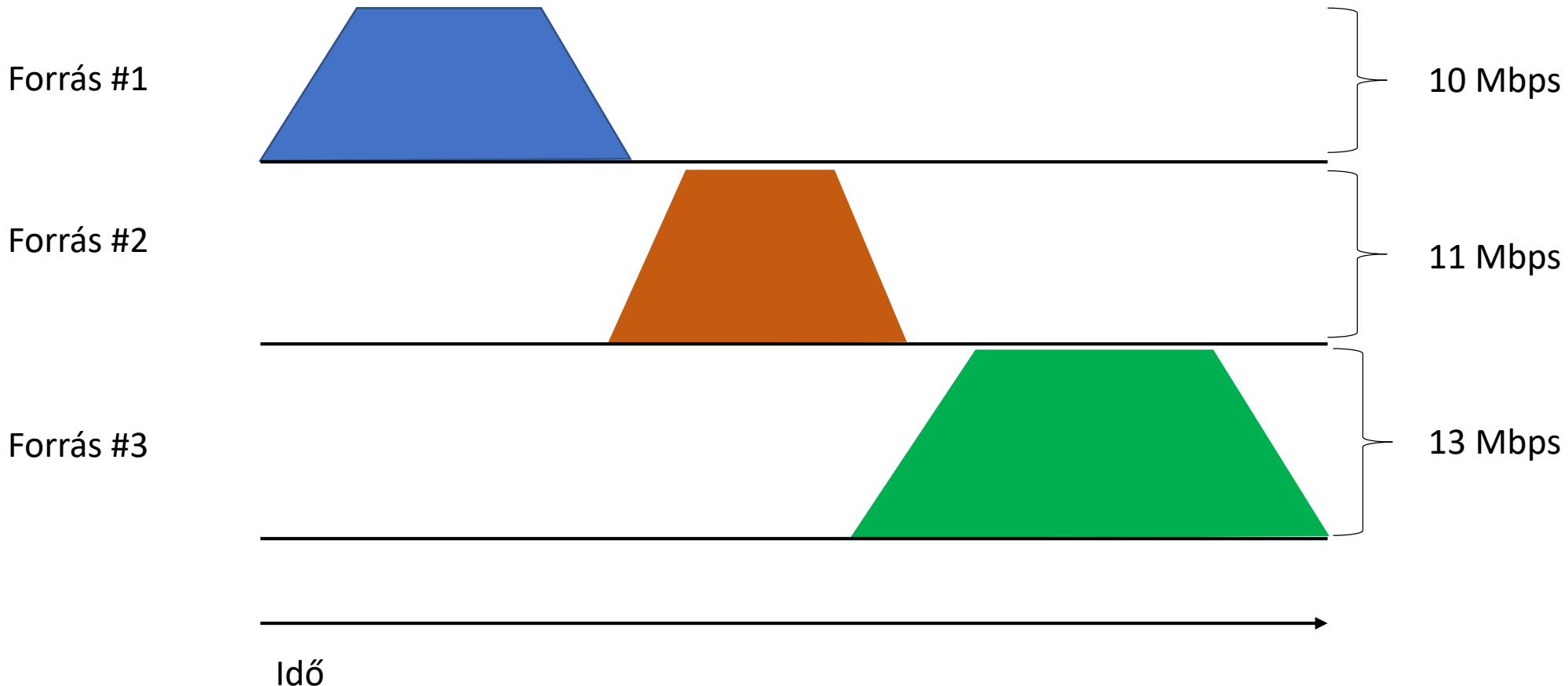
- Tegyük fel, hogy minden egyik forrásnak 10 Mbps sávszélességre van szüksége

- Mit kapnak az alábbi módszerrel:
 - Előre foglalás
 - Igény szerinti

Tekintsük a lenti hálózati csúcsigényeket és folyam időtartamokat



Tekintsük a lenti hálózati csúcsigényeket és folyam időtartamokat



Mit kapnak a források **előre foglalás** és igény szerinti megosztási stratégiák esetén?

- | First-come first-served
- | Egyenlő elosztás (10 Mbps)

Csúcs és átlagos ráták kapcsolata

Minden folyam (flow) rendelkezik

- Csúcsrátával (peak rate): P
- Átlagos rátával: A

Előre foglalás esetén **P erőforrást kell** lefoglalni.

- A lefoglalt erőforrás átlagos kihasználtsági szintjét ekkor **A/P** adja meg.
- $P=100 \text{ Mbps}$, $A=10 \text{ Mbps}$, a kihasználtsági szint = 10%

Igény szerinti erőforrás-kezelés esetén általában nagyobb kihasználtsági szintet tudunk elérni.

- Függ a versengő források számától és a folyamok forgalmi löketeitől (burstiness)

Melyik a jobb?

Jó válasz nincs, a felhasználástól függ

Ha P/A kicsi, akkor általában az előre foglalásnak van értelme

- Hang forgalom esetén ez az arány 3 vagy kisebb

Ha P/A nagy, akkor az előre foglalás jelentősen **erőforrás-pazarló**

- Az adatkommunikáció esetén a forgalom löketekben érkezik (bursty - „börsztös”)
- A P/A arány jellemzően >100

Melyik a jobb?

Jó válasz nincs, a felhasználástól függ

Ha P/A kicsi, akkor általában az előre foglalásnak van értelme

- Hang forgalom esetén ez az arány 3 vagy kisebb

Ha P/A nagy, akkor az előre foglalás jelentősen **erőforrás-pazarló**

- Az adatkommunikáció esetén a forgalom löketekben érkezik (bursty - „börsztös”)
- A P/A arány jellemzően >100

Ez az oka, hogy ...

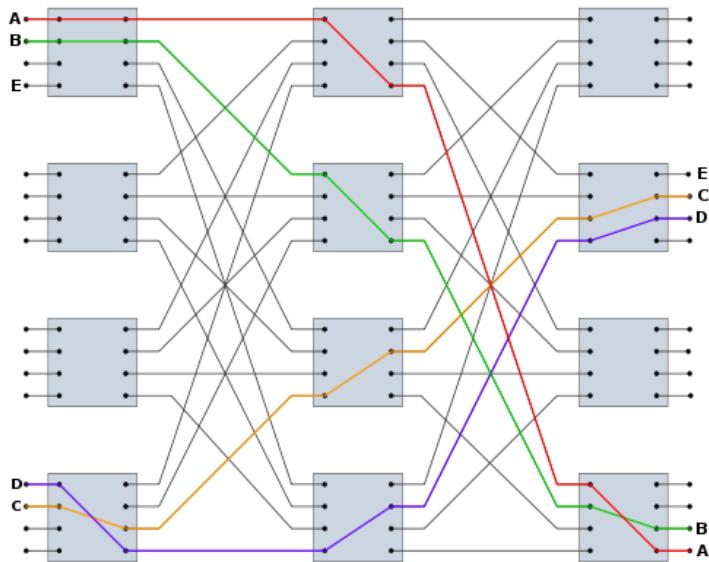
- A telefon hálózat előre foglalást használ,
- ...míg az Internet igény szerinti erőforrás kezelést alkalmaz.

Megvalósítások

Előre foglalással

Áramkörkapcsolt hálózat

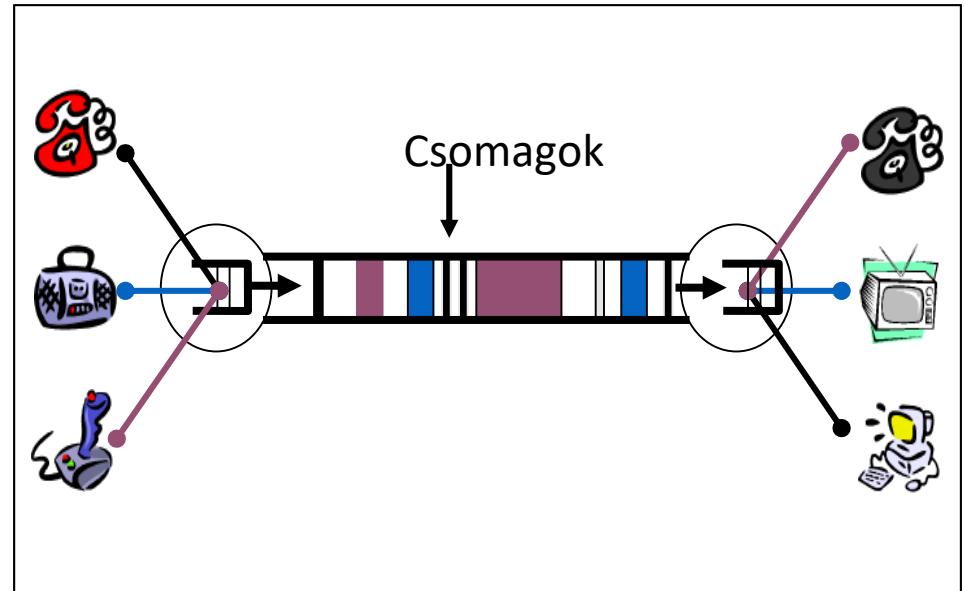
Pl. vezetékes telefon



Igény szerinti

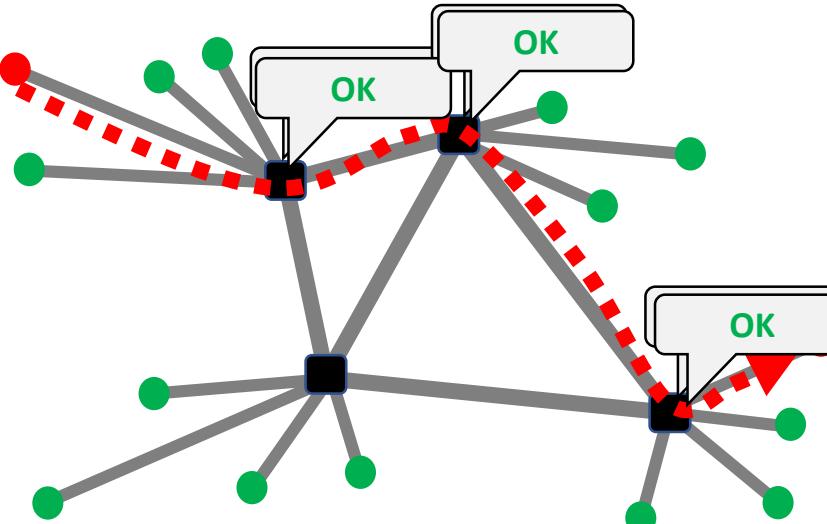
Csomagkapcsolt hálózat

Pl. Internet



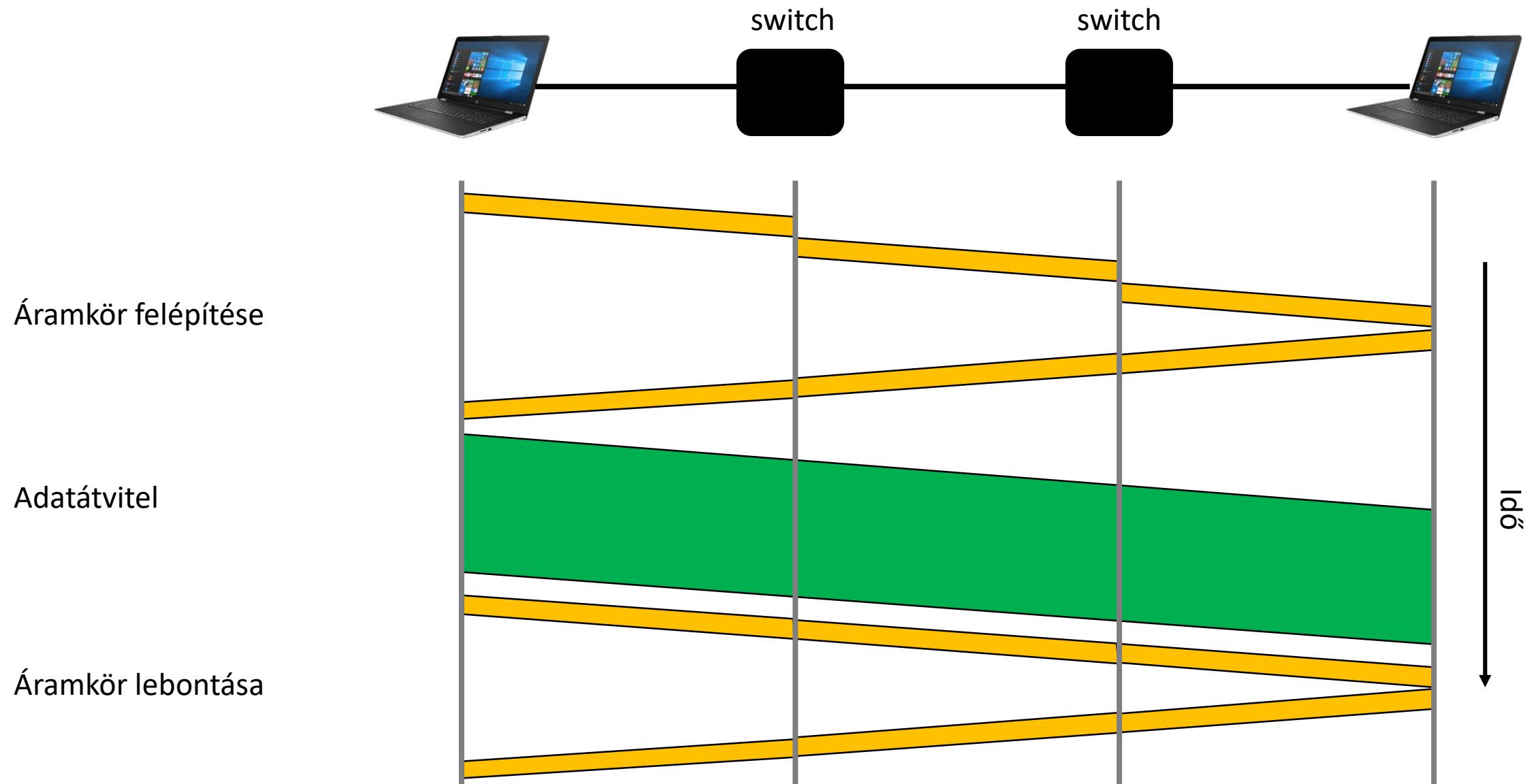
Az áramkörkapcsolt hálózat alapja a Resource Reservation Protocol

Erőforrás-foglaló protokoll



1. A forrás foglalási kérést küld 10 Mbps igényről a célállomásnak
2. A switchek kialakítják az „áramkört”
3. A forrás megkezdi az adatküldést
4. A forrás áramkör-lebontó üzenetet küld a cél felé (teardown)

Adatátvitel áramkörkapcsolt hálózaton



Löketszerű forgalom - Rossz teljesítmény

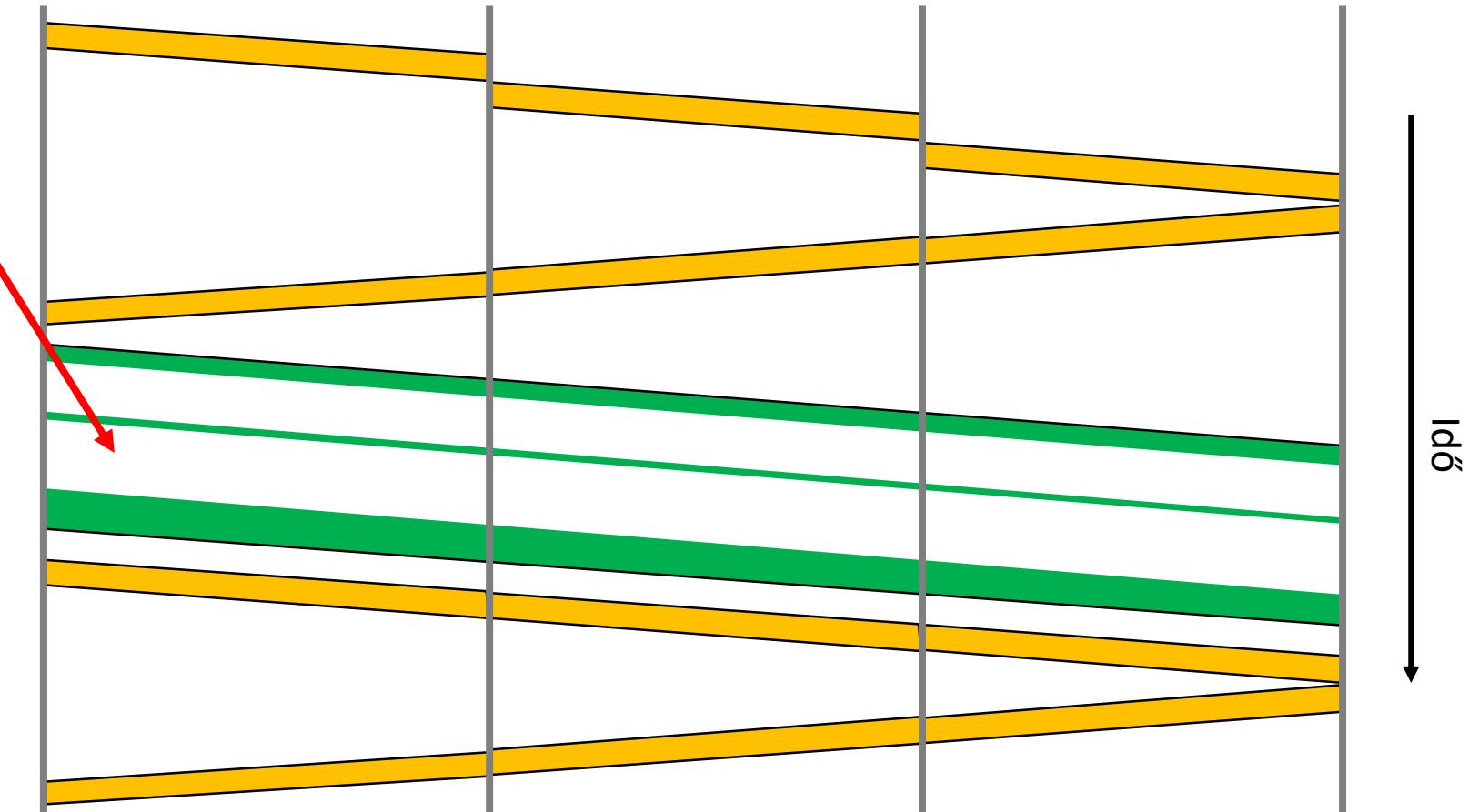
A löketszerű forgalom miatt az áramkör az idő nagy részében kihasználatlan.



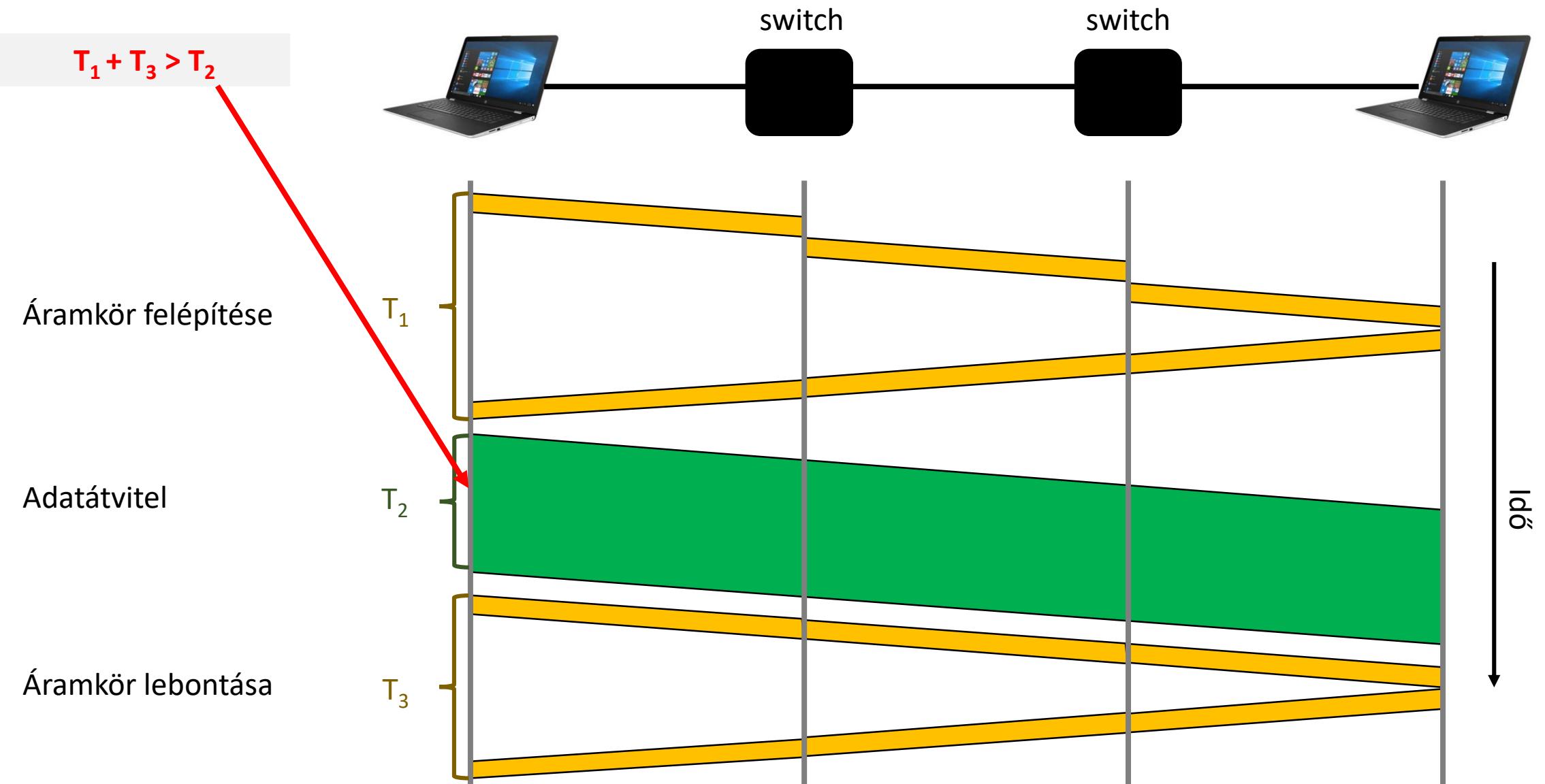
Áramkör felépítése

Adatátvitel

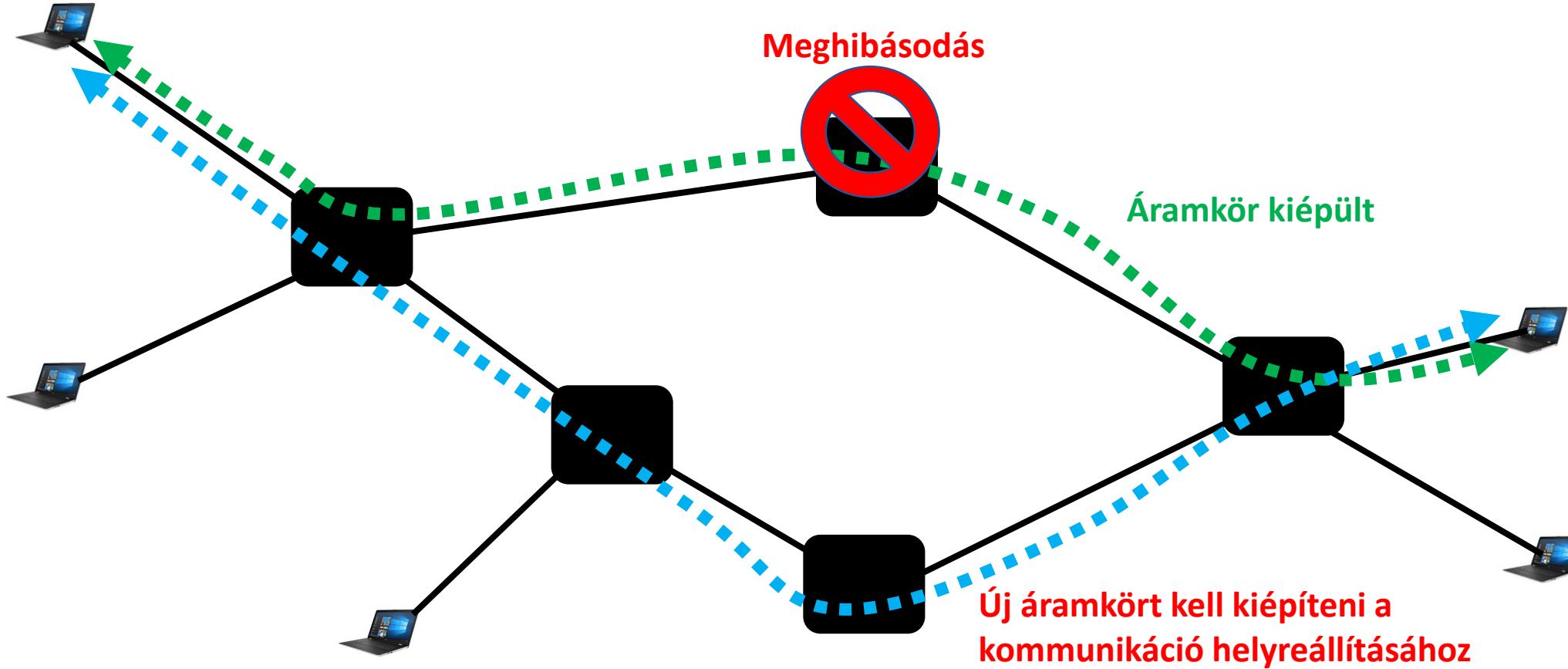
Áramkör lebontása



Rövid üzenetváltás – Rossz teljesítmény



További probléma a **meghibásodott** switch kikerülése (reroute)



Érvek/Ellenérvek

Előnyök

Kiszámítható teljesítmény

Egyszerű és gyors kapcsolás
Miután kiépült az áramkör

Hátrányok

Alacsony hatékonyság

Löketszerű forgalom
Rövid folyamok

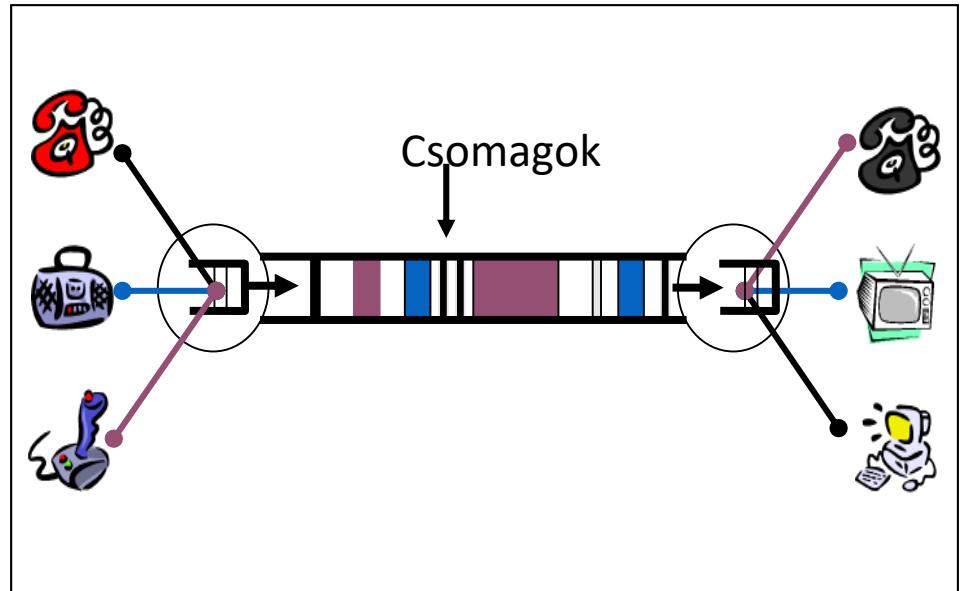
Bonyolult áramkör felépítés/lebontás
Megnövekedett késleltetés

Hiba esetén új áramkör szükséges

Megvalósítások

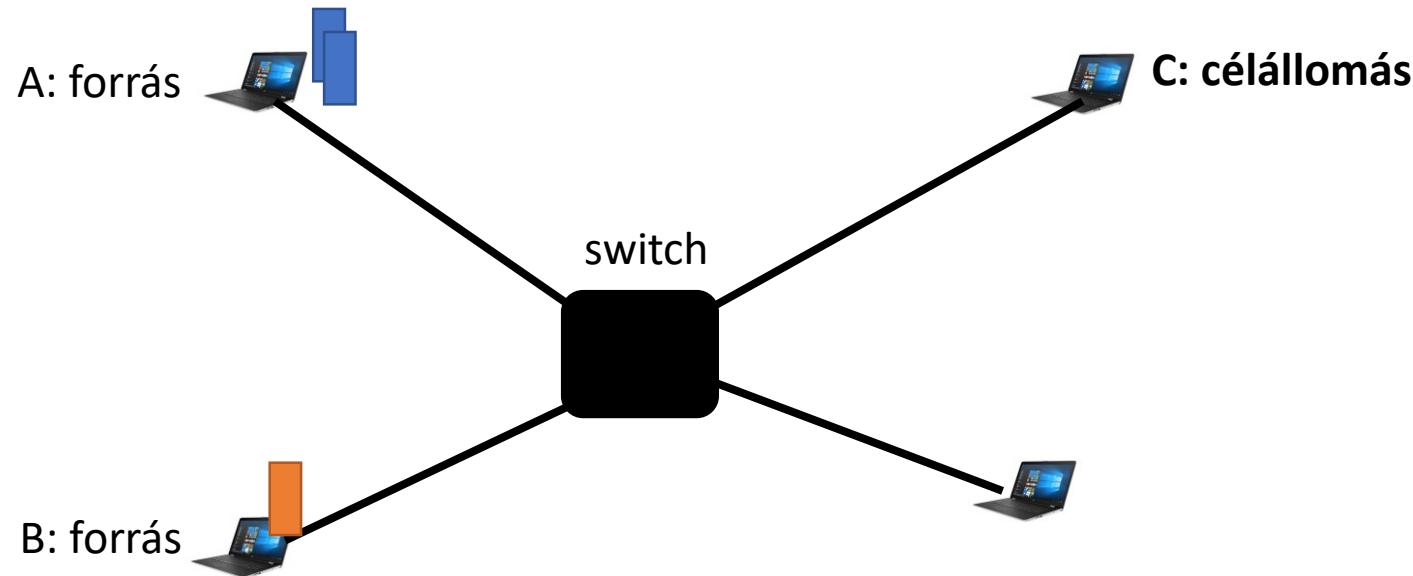
Igény szerinti

Csomagkapcsolt hálózat
Pl. Internet



Csomagkapcsolt hálózatok

Az adatátvitel egyedi csomagokban történik.



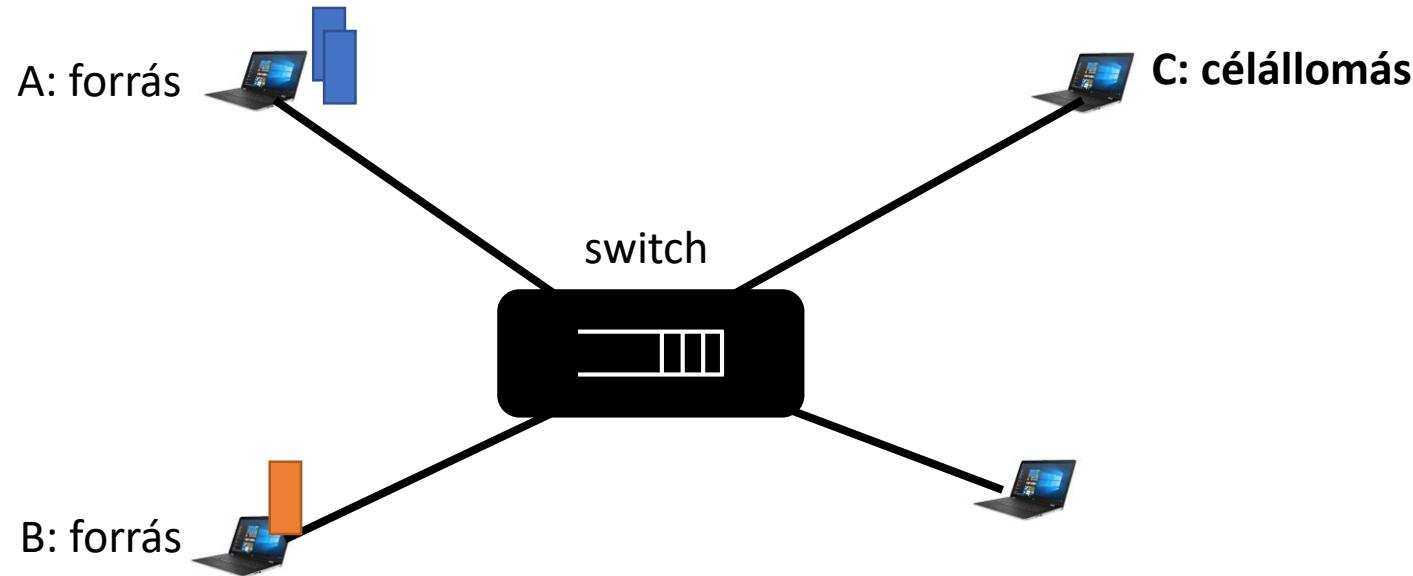
Minden csomag tartamazza a cél címét/azonosítóját (most C).

Nincs globális koordináció, azaz a csomagok zavarhatják egymást.
(Id. egyszerre érkeznek be a switchhez)

Pufferelés szükséges a löketek kezeléséhez.

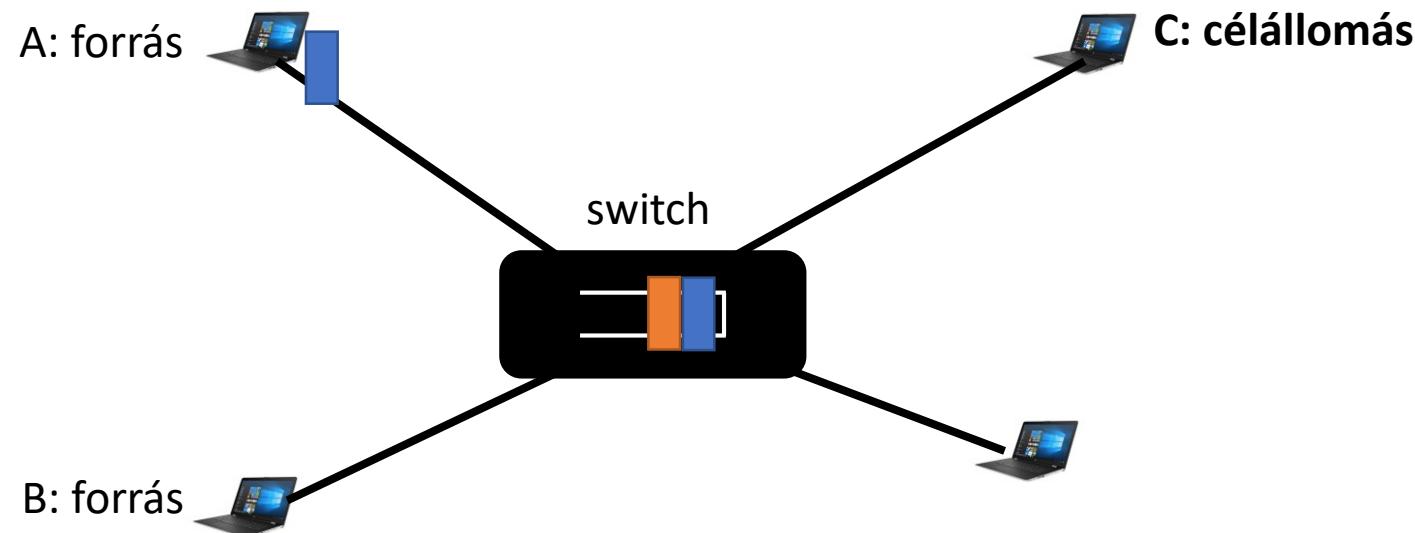
Csomagkapcsolt hálózatok

Pufferelés az átmeneti túlterhelések kezeléséhez

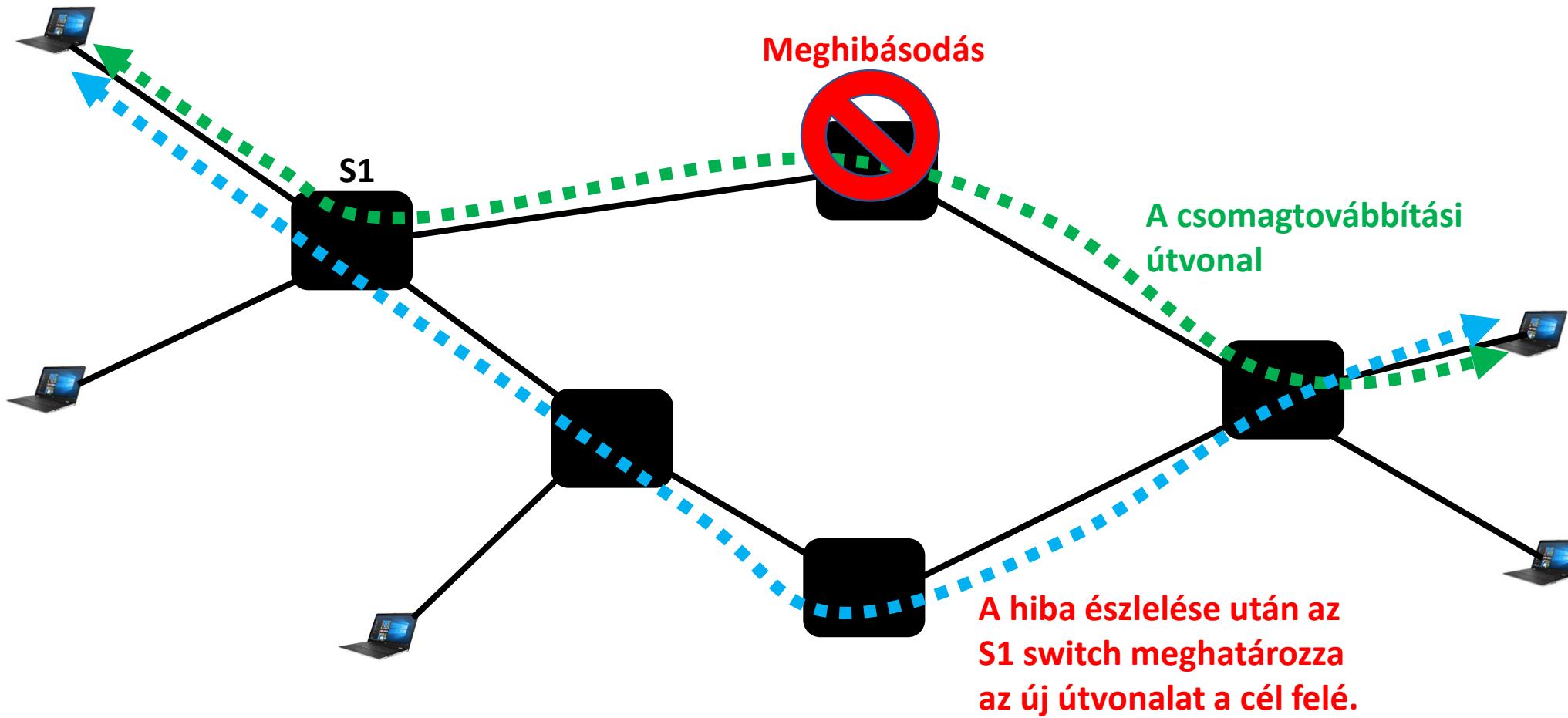


Csomagkapcsolt hálózatok

Pufferelés az átmeneti túlterhelések kezeléséhez



Hiba tolerancia



Érvek ellenérvek

Előnyök

Hatékony erőforrásgazdálkodás

Egyszerű megvalósítás

Hibatolerancia

Hátrányok

Kiszámíthatatlan teljesítmény

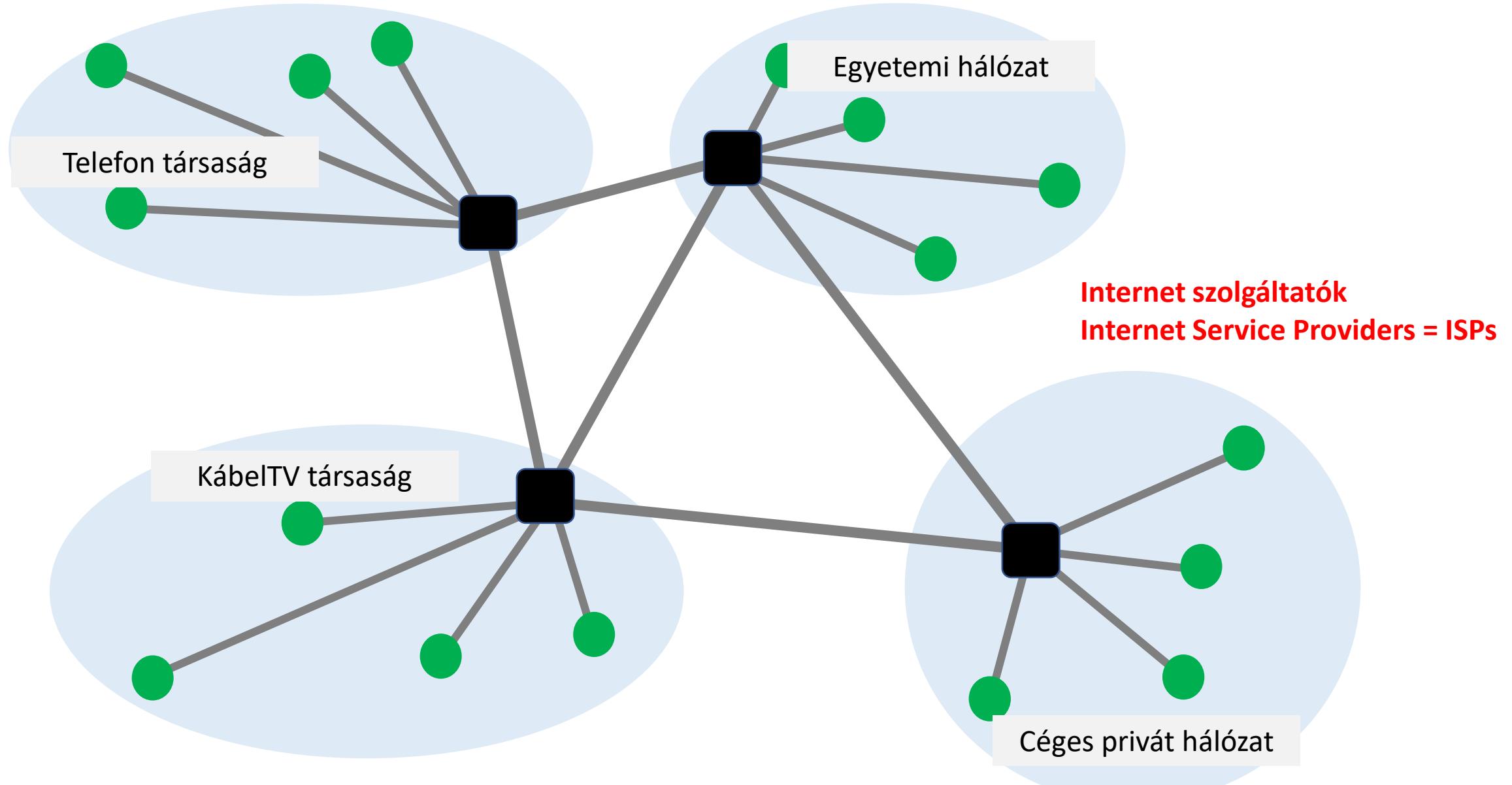
Szükséges puffer-kezelés és torlódás-vezérlés

Az Internet csomagkapcsolt

Rugalmasság és hatékonyság

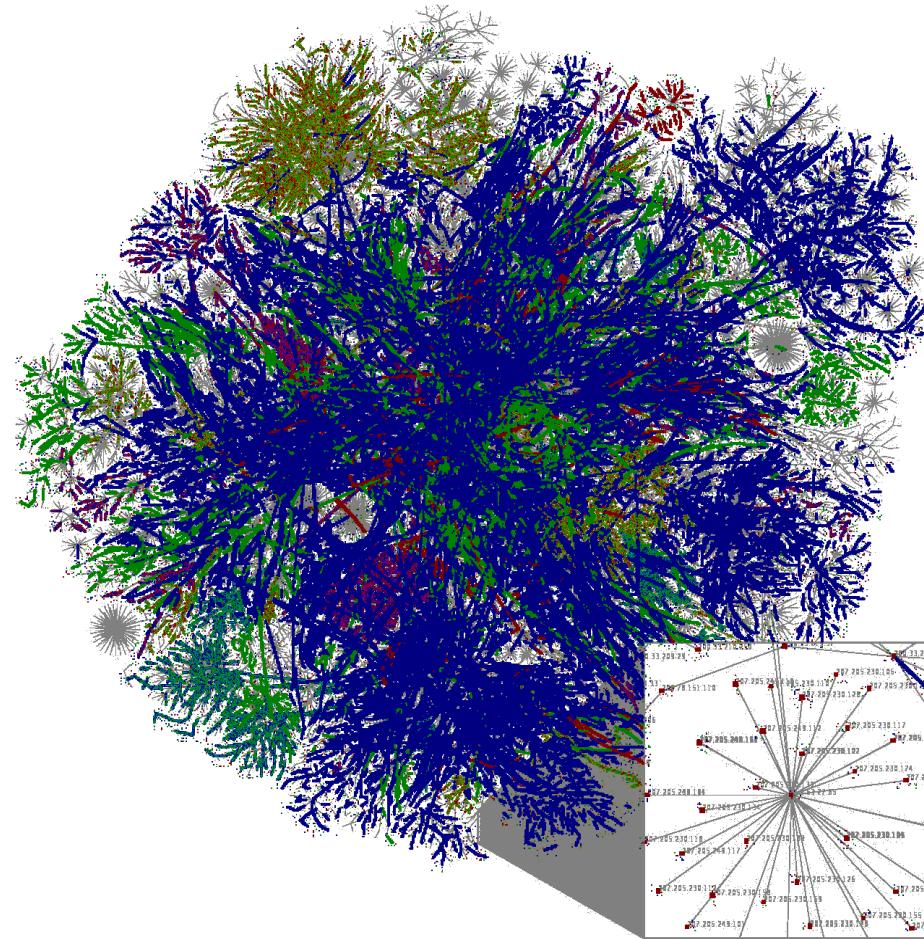
Áttekintés

Hogyan szervezzük a hálózatot?

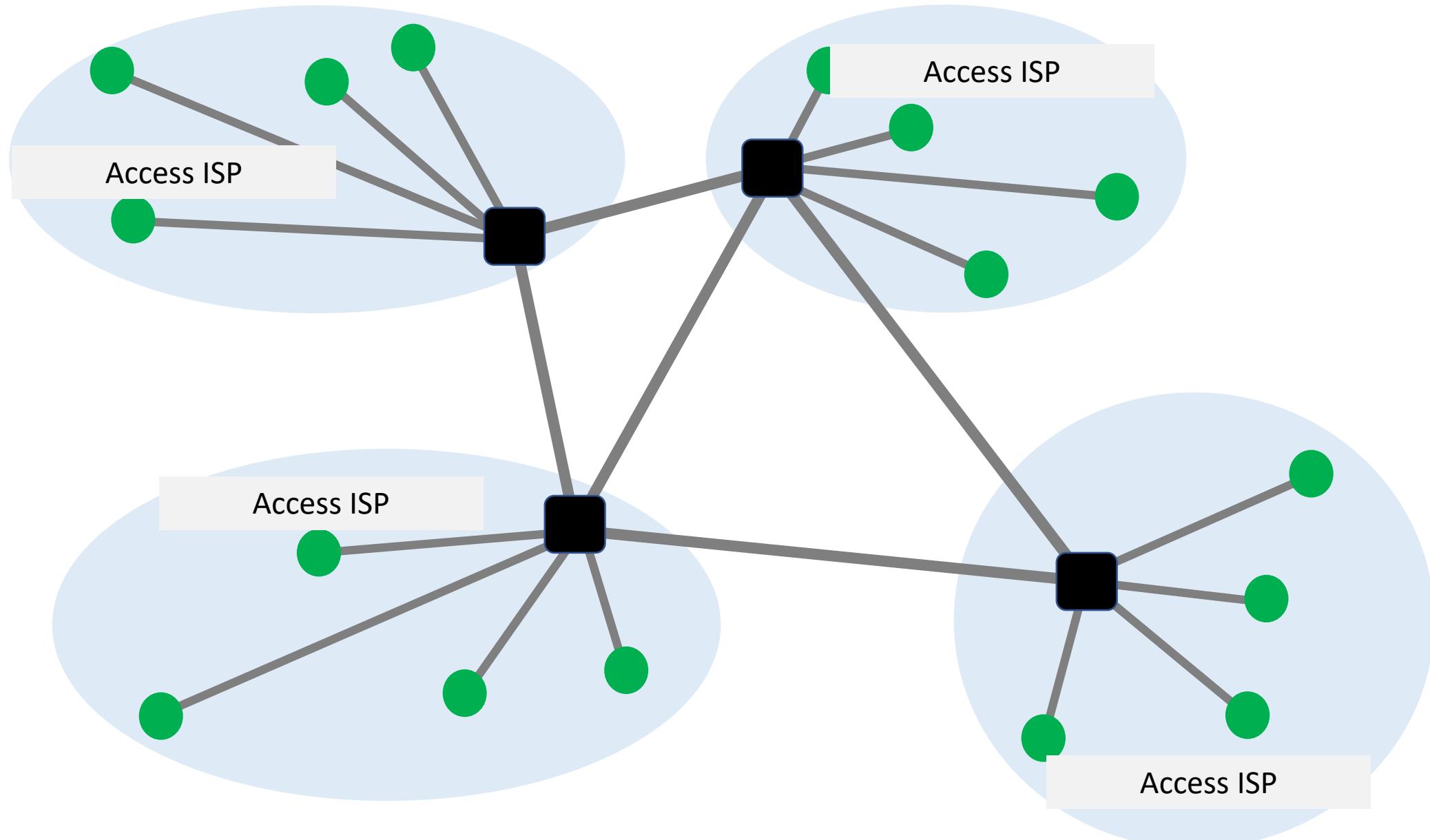


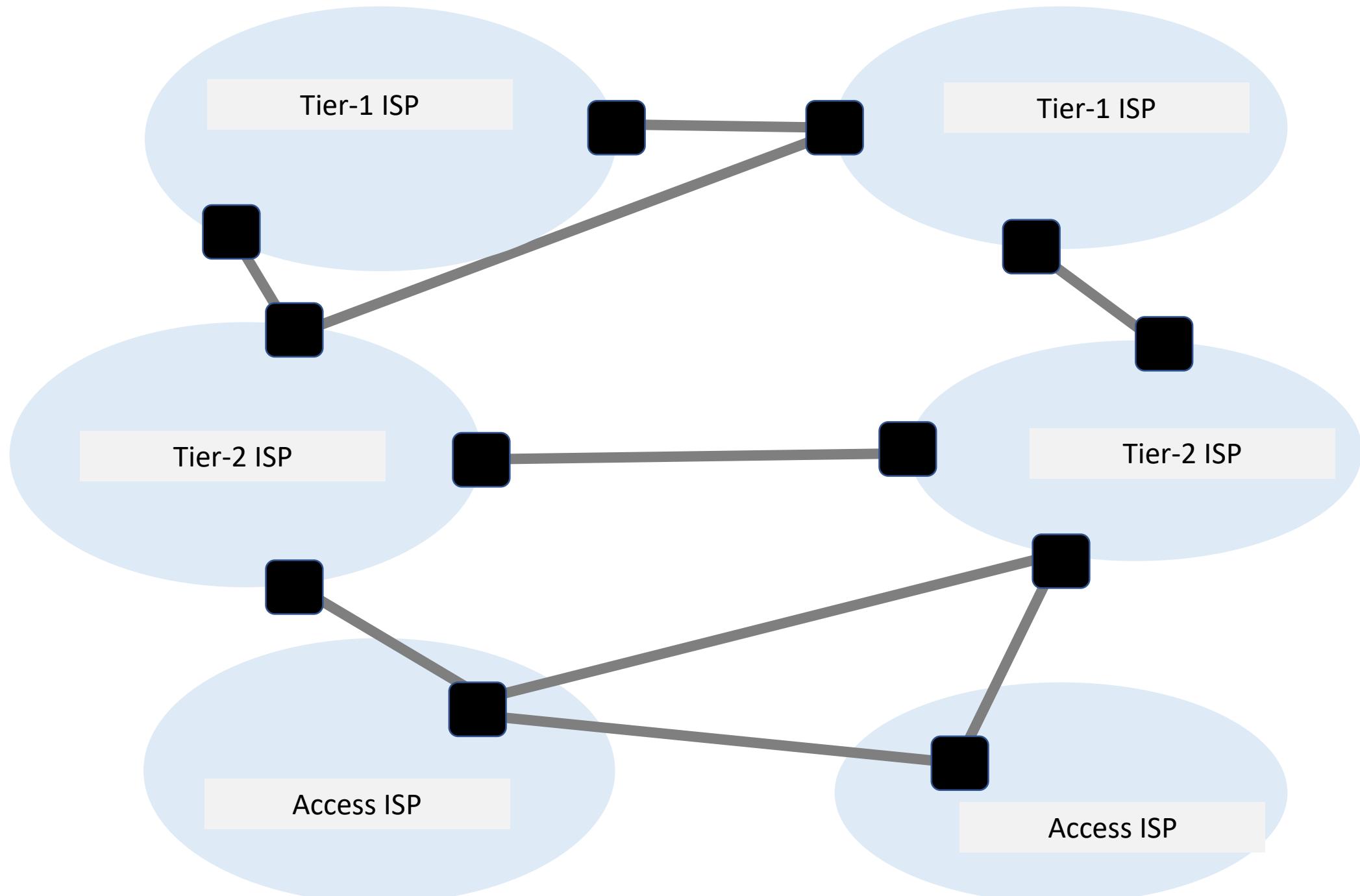
Mi az internet?

- Hálózatok hálózata
- A világra kiterjedő nyitott WAN
- Jellemzői
 - rendszerfüggetlenség;
 - nincs központi felügyelet;
 - építőelemei a LAN-ok;
 - globális;
 - olyan szolgáltatásokat nyújt, mint a **World Wide Web**, e-mail vagy fájlátvitel.



ISP – Internet szolgáltató





Az Internet hierarchikus struktúrája

szolgáltató-vásárló (provider-customer) viszonyok

Tier-1

nemzetközi

Nincs szolgáltatója

Tier-2

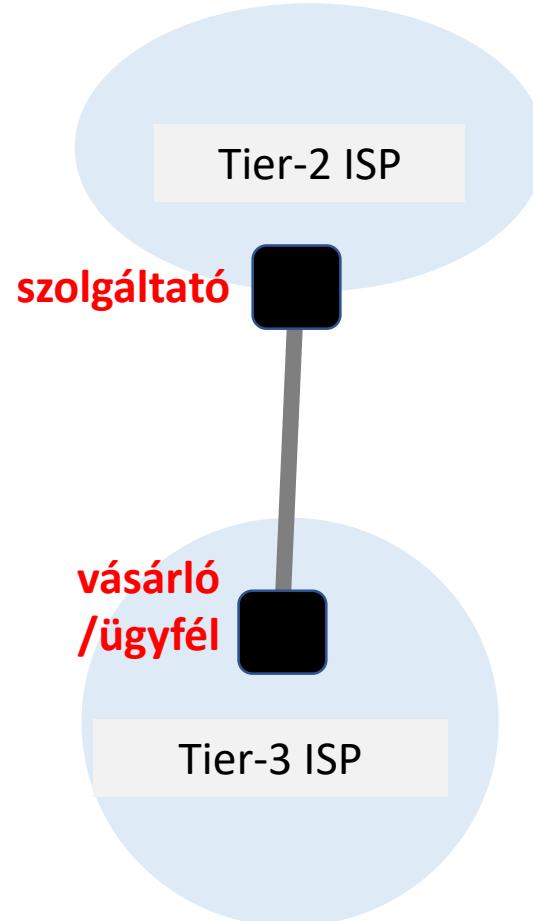
nemzeti

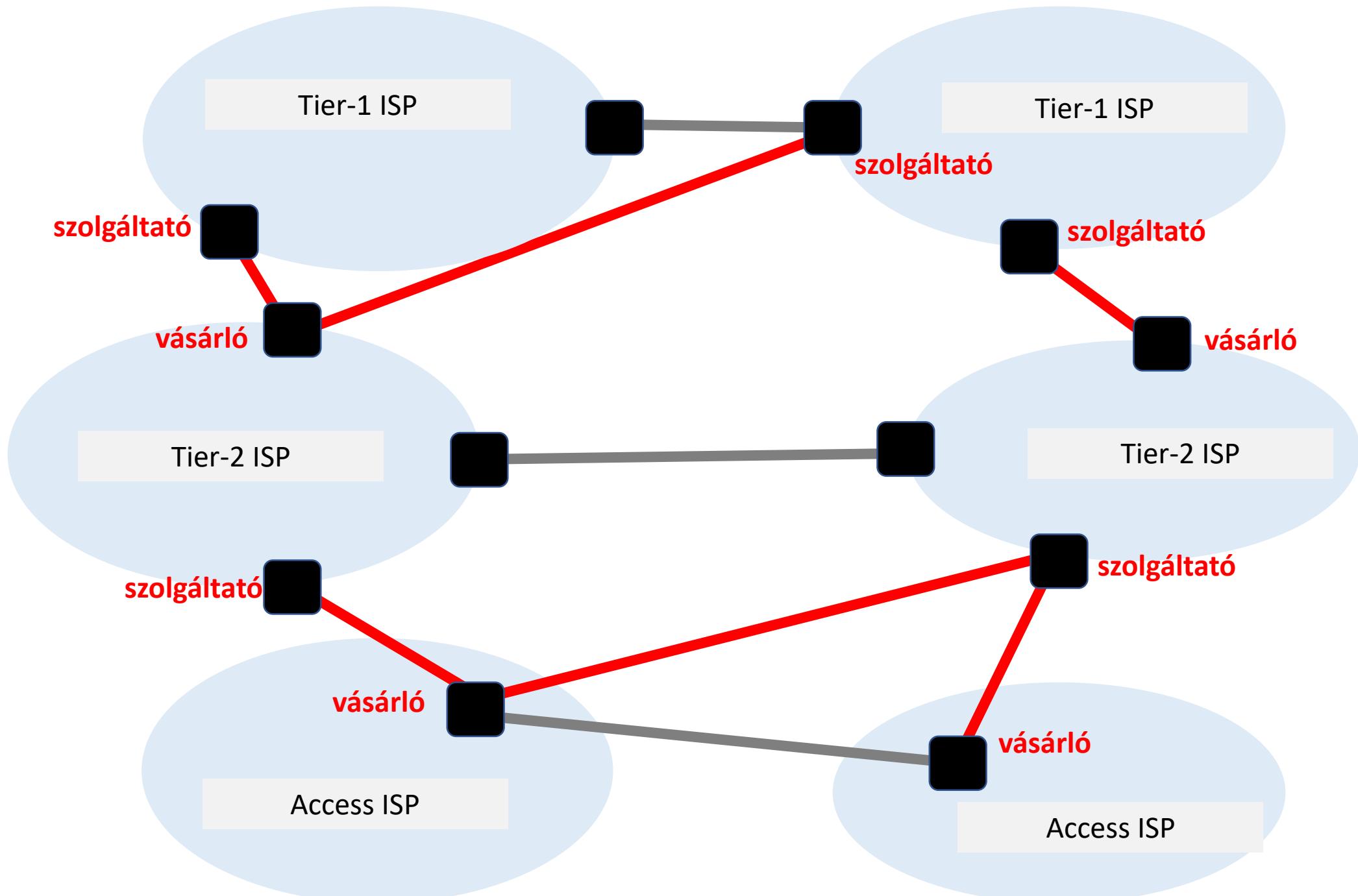
Tier-3 szolgáltatóknak nyújt átjárást
Legalább egy szolgáltatója van

Tier-3

helyi

Nem nyújt átjárást más szolgáltatóknak
Legalább egy szolgáltatója van





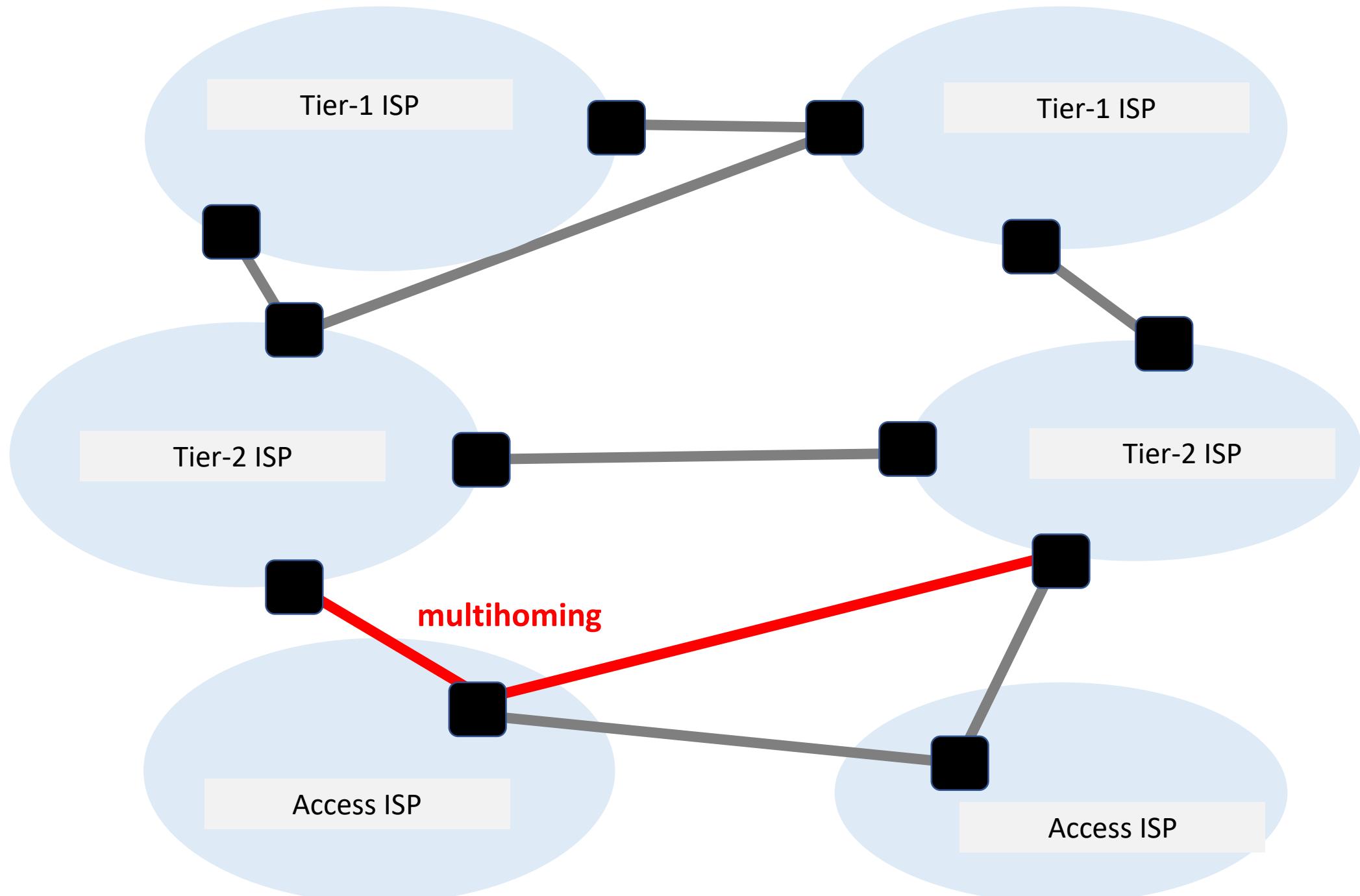
Hálózatok eloszlása a Tier-ekben

~50.000 hálózat összesen

pár tucat

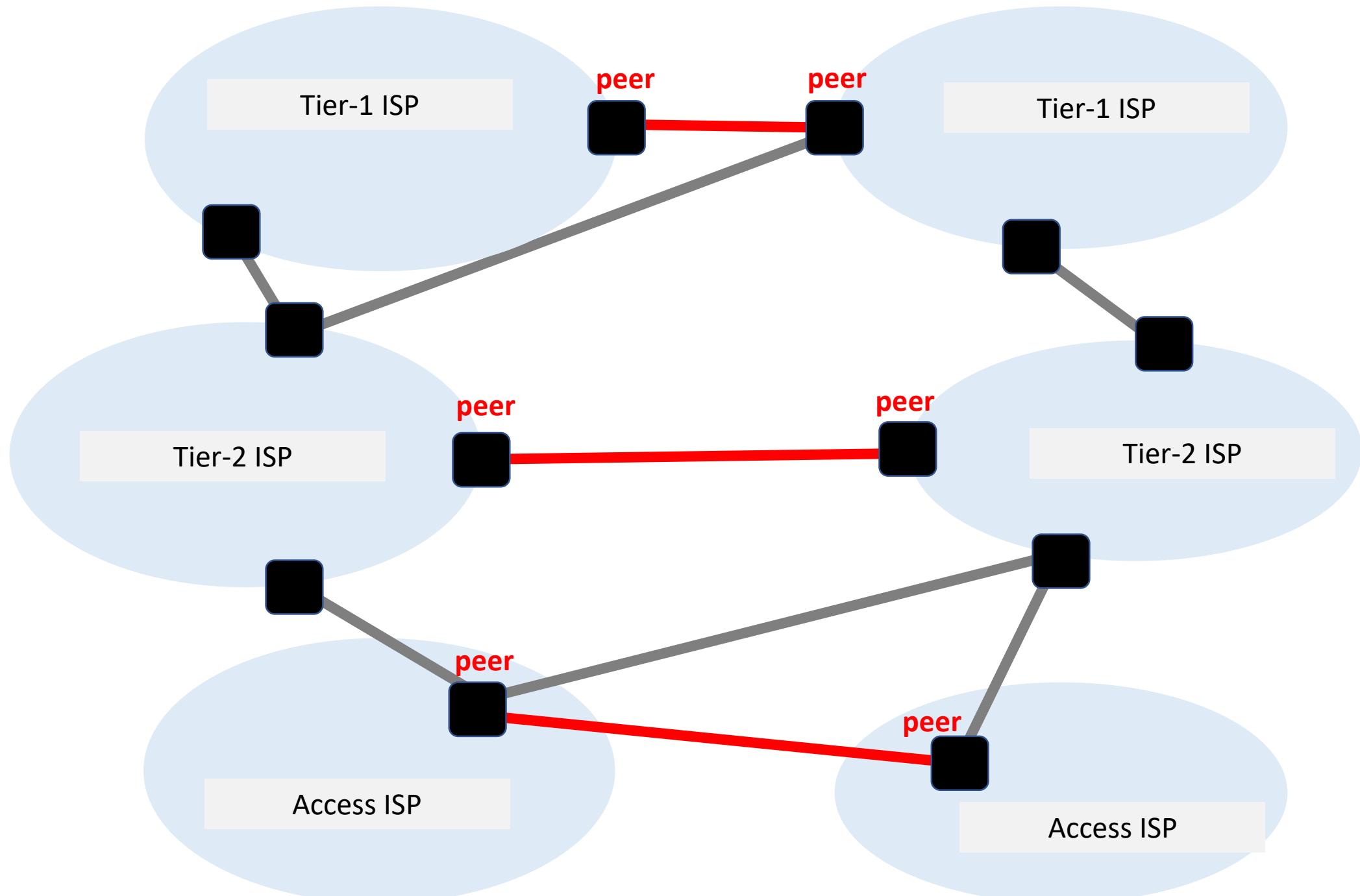
több ezer

85-90%



Némely hálózat között közvetlen kapcsolat is létezik
– csökkenti a szolgáltatónak fizetendő számlát

Ezt hívjuk „peering”-nek – ez egyfajta kölcsönös kapcsolat...



A szomszédos hálózatok egyesével való összekapcsolása túl költséges lenne

Infrastruktúra költségek

Fizikai linkek kiépítése vagy bérlese

Sávszélesség költségek

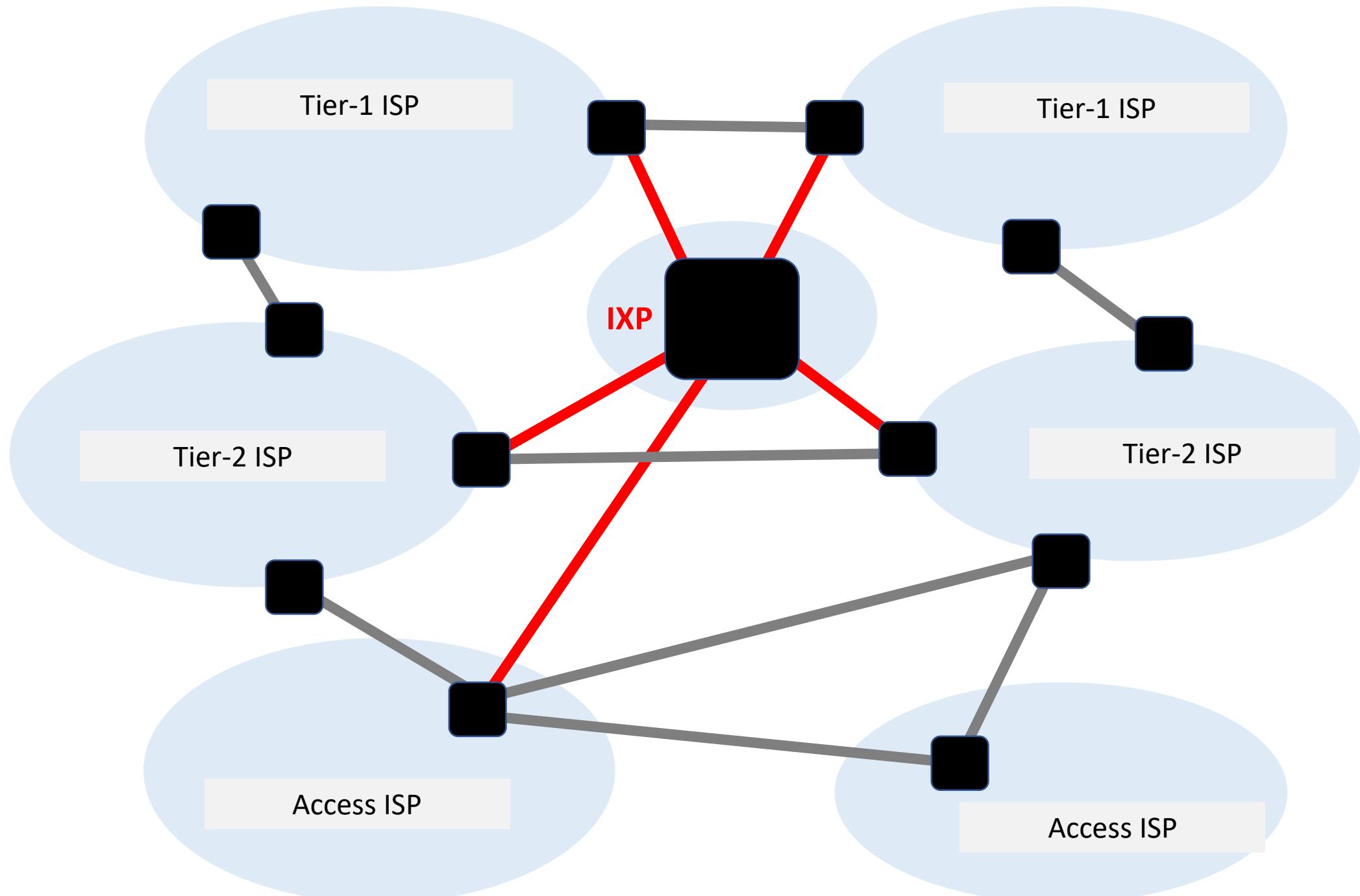
A linkek nem feltétlenül lesznek teljesen kihasználva

Humán költségek

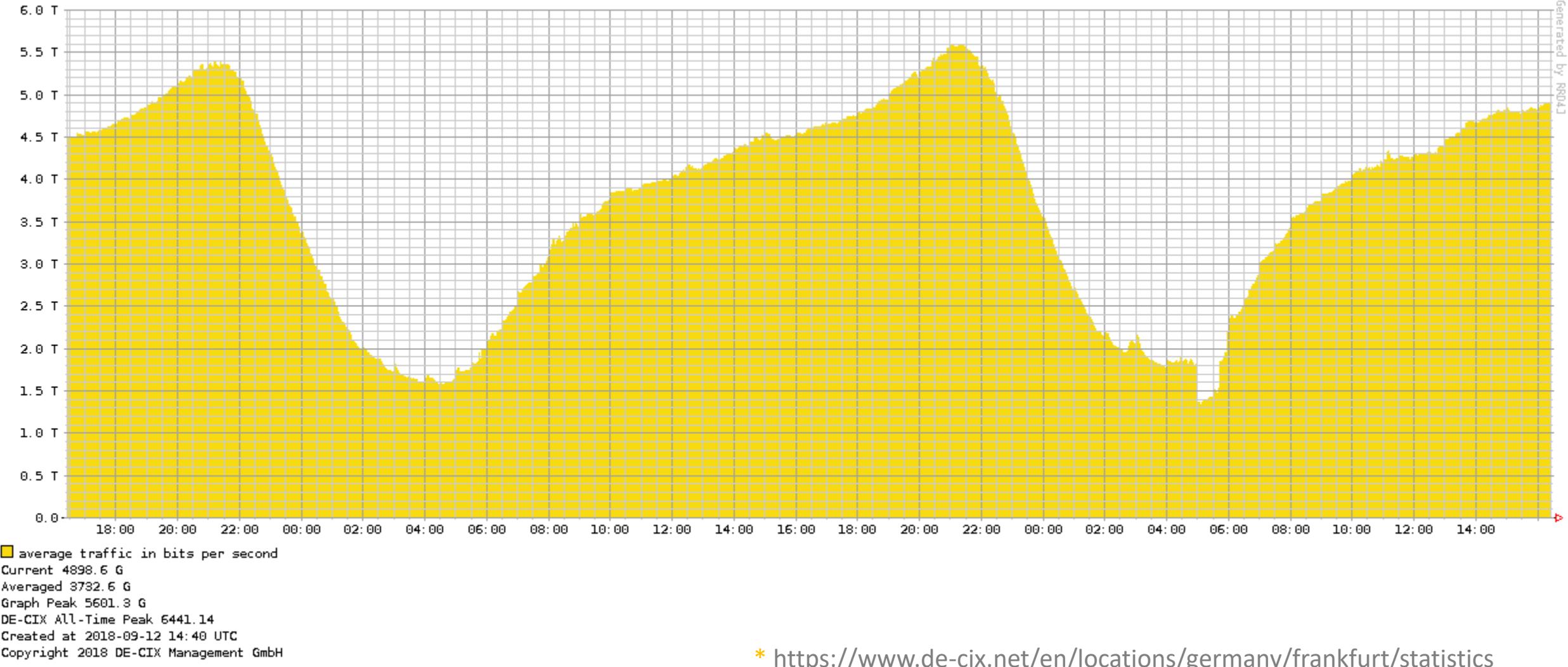
Minden kapcsolatot egyedi módon kell kezelní

A problémát az úgynévezett Internet eXchage Pontok (IXP) oldják meg

**Az IXP-k lehetővé teszik több hálózat összekapcsolását
egy fizikai (földrajzi/topológiai) helyen.**



Egy IXP két napja – DE-CIX Frankfurt



Az Internet rövid története

Az egész az 50-es években kezdődött...

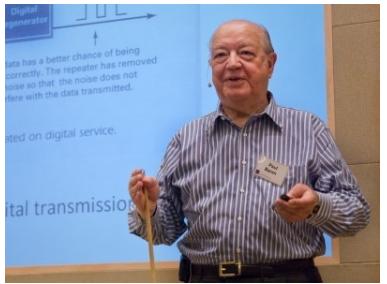
Telefonhálózat – a kommunikációs hálózat
teljesen áramkörkapcsolt



Elkezdik másra is használni a hálózatokat
hadászat, számítógépek, stb.

azonban az áramkörkapcsolt megoldás ezeknek nem felelt meg...
nem elég hatékony és rugalmas löketszerű terhelések kezelésére

Három legfontosabb kérdés



Paul Baran RAND

Hogyan tervezünk sokkal **rugalmasabb** hálózatokat?
... csomagkapcsolt hálózatok feltalálása



Leonard Kleinrock
UCLA

Hogyan tervezünk sokkal **hatékonyabb** hálózatokat?
... csomagkapcsolt hálózatok feltalálása

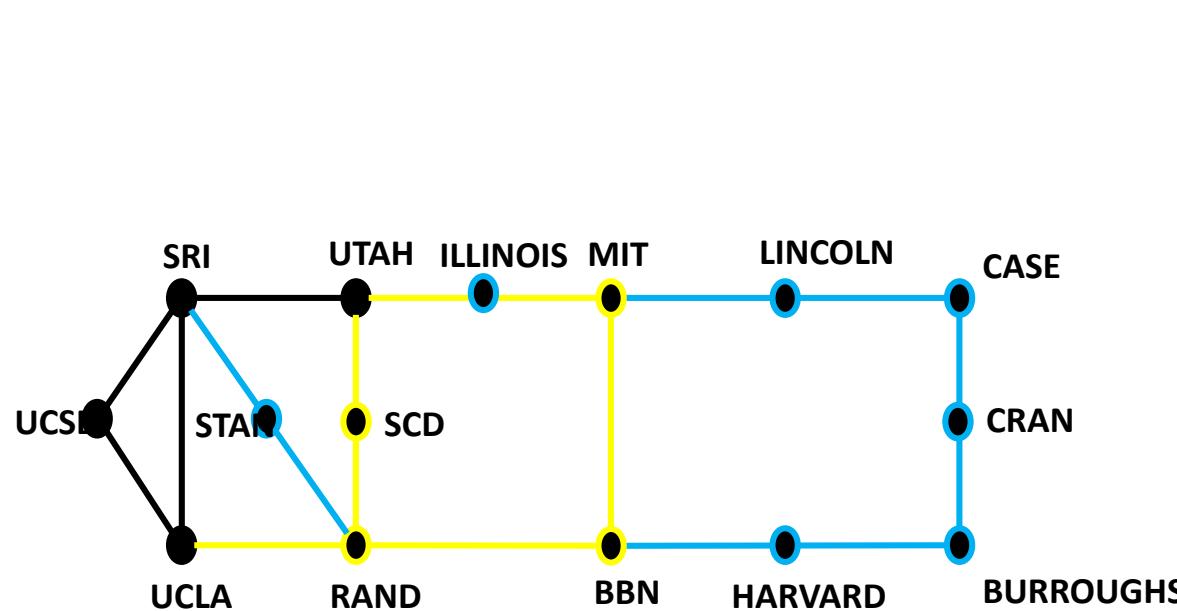


Vint Cerf & Bob Kahn
DARPA

Hogyan **kapcsoljuk össze** ezeket a hálózatokat?
... a ma ismert Internet feltalálása

A 60-as évek a csomagkapcsolt hálózatokról szólt...

Advanced Research Projects Agency NETwork (ARPANET)



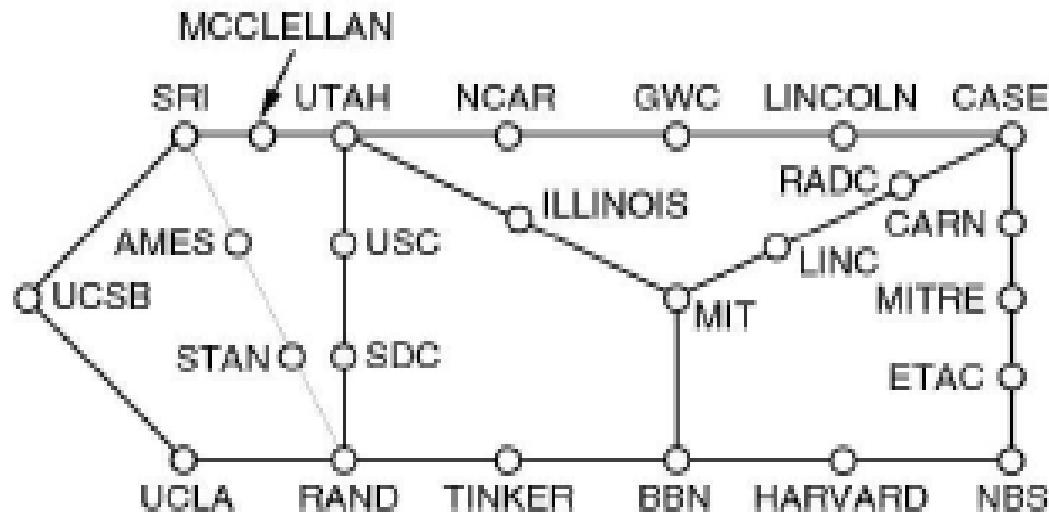
1969 december

1970 július

1971 március

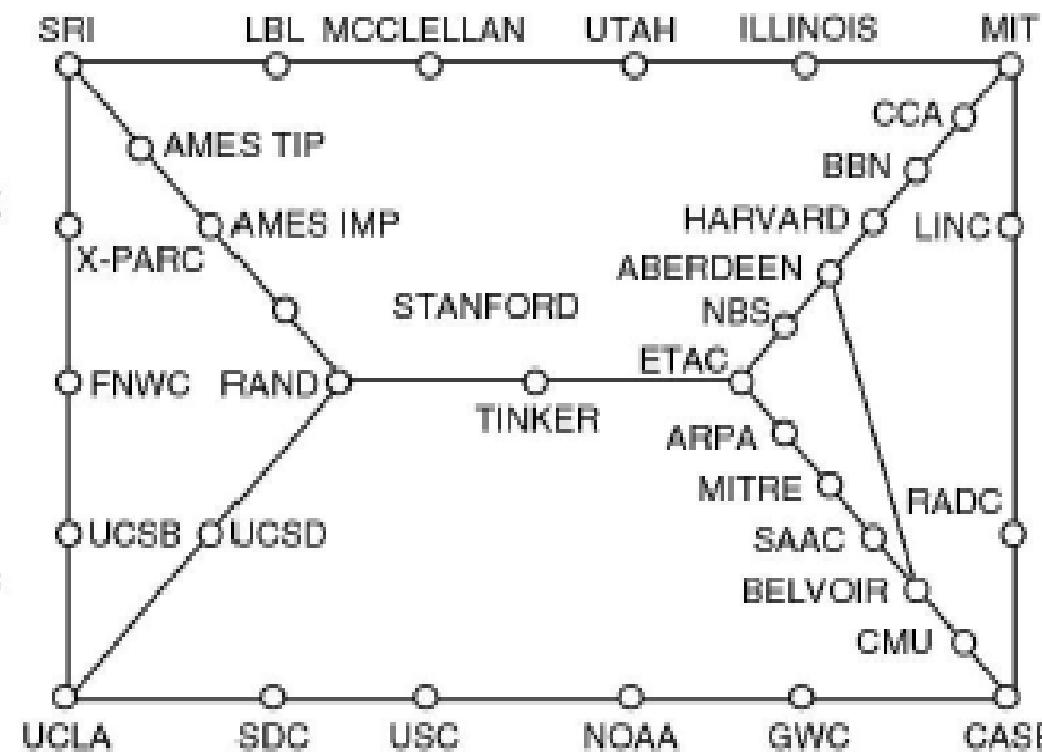
ARPANET

1972 április



ARPANET

1972 szeptember



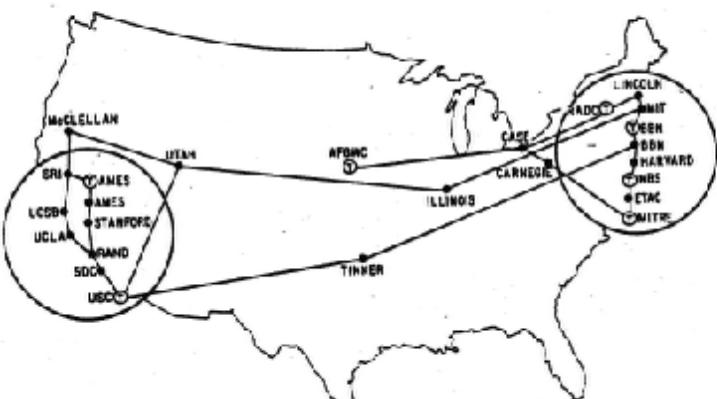
ARPANET



Dezember 1969



Juni 1970



März 1972



Juli 1977

Robert Kahn koncepciója – DARPA 1972

- **Minden (lokális) hálózat autonóm**
 - önállóan dolgozik
 - nem kell elkülönítve konfigurálni a WAN-hoz
- **Kommunikáció a „legjobb szándék” (angolul *best effort*) elv szerint**
 - ha egy csomag nem éri el a célt, akkor törlődik
 - az alkalmazás újraküldi ilyen esetekben
- **„Black box” megközelítés a kapcsolatokhoz**
 - a Black Box-okat később *Gateway*-eknek és *Router*-eknek keresztelték át
 - csomaginformációk nem kerülnek megőrzésre
 - nincs folyam-felügyelet
- **Nincs globális felügyelet**

Ezek az Internet alapelvei

Az Interneten átküldött első üzenet: „LO”

1969. október 29.

Leonard Kleinrock a UCLA-ről megpróbál
távolról belépni egy stanfordi számítógépre

UCLA We typed the L... Do you see it?

Yes! We see the L Stanford

We typed the O... Do you see it?

Yes! We see the O

We typed the G.

... és a rendszer összeomlott...

A 70-es évek már az Ethernet, TCP/IP és az email korszaka volt...

1971

Network Control Program (NCP)

A TCP/IP elődje

1972

Email és Telnet

1973

Ethernet

1974

TCP/IP

Vint Cerf és Bob Kahn cikke

80-as években minden a TCP/IP-ről szól...

- 1983 **NCP-ről TCP/IP-re**
Domain Name Service (DNS)
- 1985 **NSFNet (TCP/IP) az ARPANET utódja**
- 198x **Internet összeomlások a torlódások miatt**
- 1986 **Van Jacobson megmenti az Internetet**
torlódásvezérlés – congestion control



Van Jacobson

90-as évek – minden az Internetről és a webről szól...

1989

ARPANET vége

A WEB megszületése

Tim Berners Lee (CERN)



1993

Első kereső motor (Excite)

1995

NSFNet vége

1998

A Google megújítja a keresést

Folytatása következik...

Vége az első résznek.



Computer Networks

Sándor Laki

ELTE-Ericsson Communication Networks Laboratory

ELTE FI – Department Of Information Systems

lakis@elte.hu

<http://lakis.web.elte.hu>

Eötvös Loránd
University



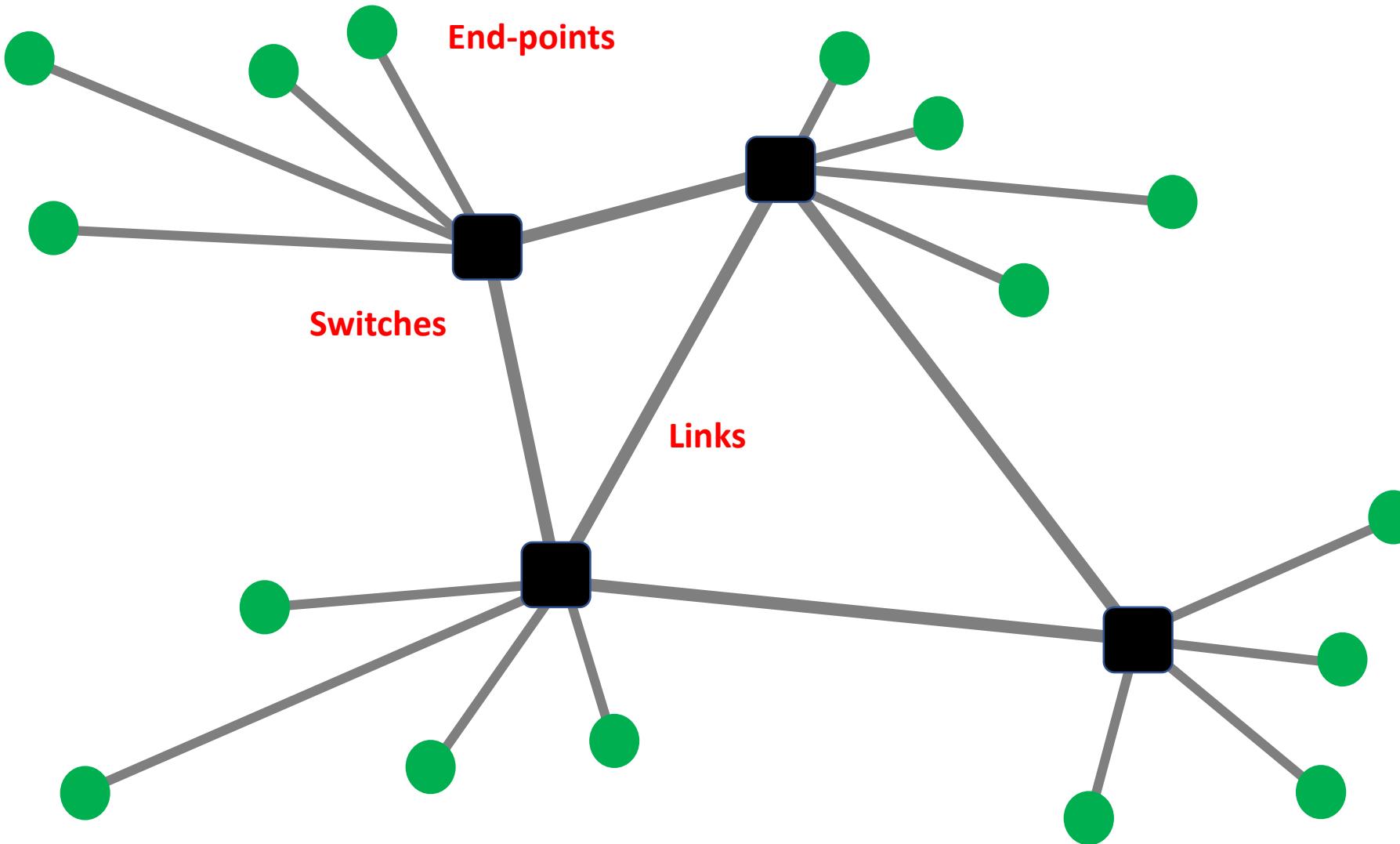
*Based on the slides of Laurent Vanbever.
Further inspiration: Scott Shenker & Jennifer Rexford & Phillipa Gill*

Last week on
Computer Networks

Overview

What is a network made of?

Three main components



Overview

How to share network resources?

Resource handling

Two different approaches for sharing

Reservation

Reserve the needed
bandwidth in advance

Flow-level multiplexing

On-demand

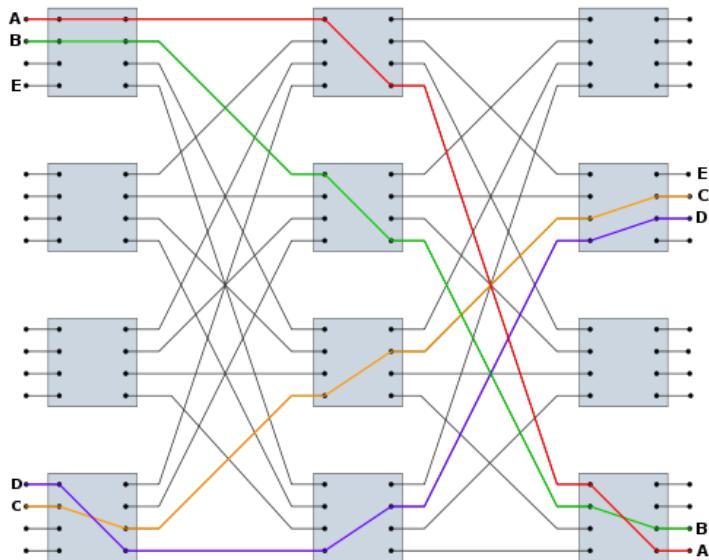
Send data when needed
Packet-level multiplexing

Implementation

Reservation

Circuit-switching

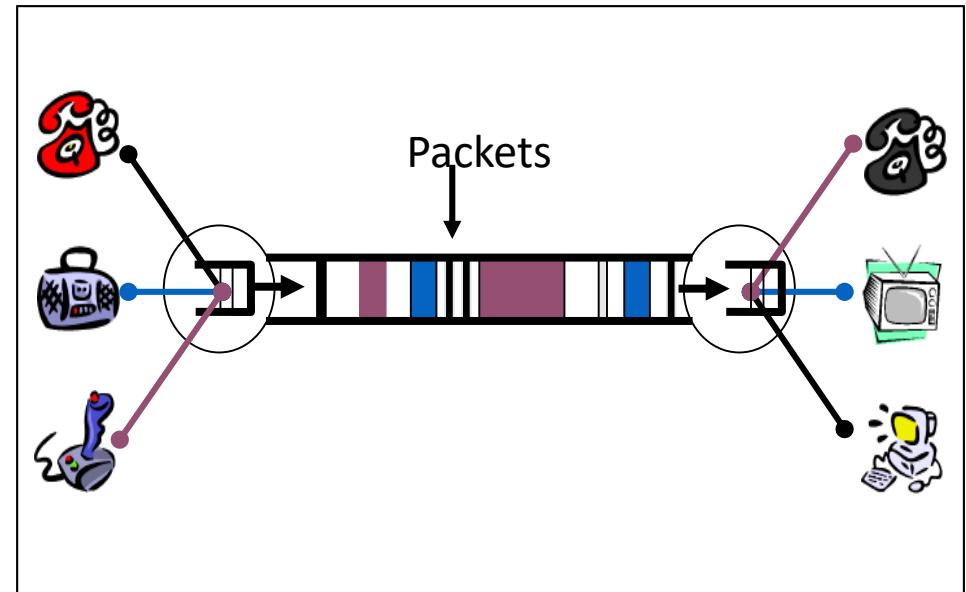
e.g. landline phone networks



On-demand

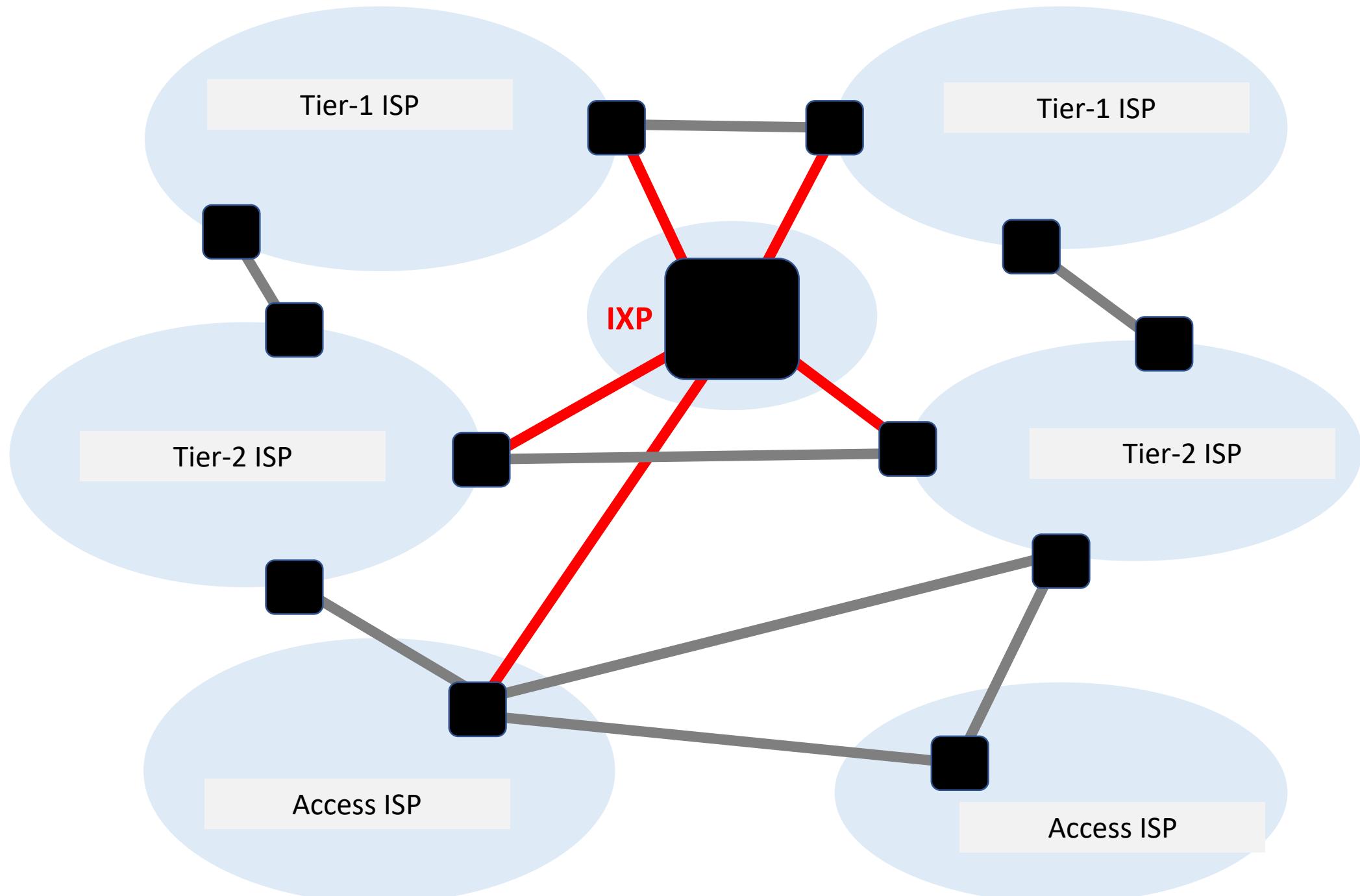
Packet-switching

e.g. Internet



Overview

How to organize the network?



This week

How does communication happen?

How do we characterize it?

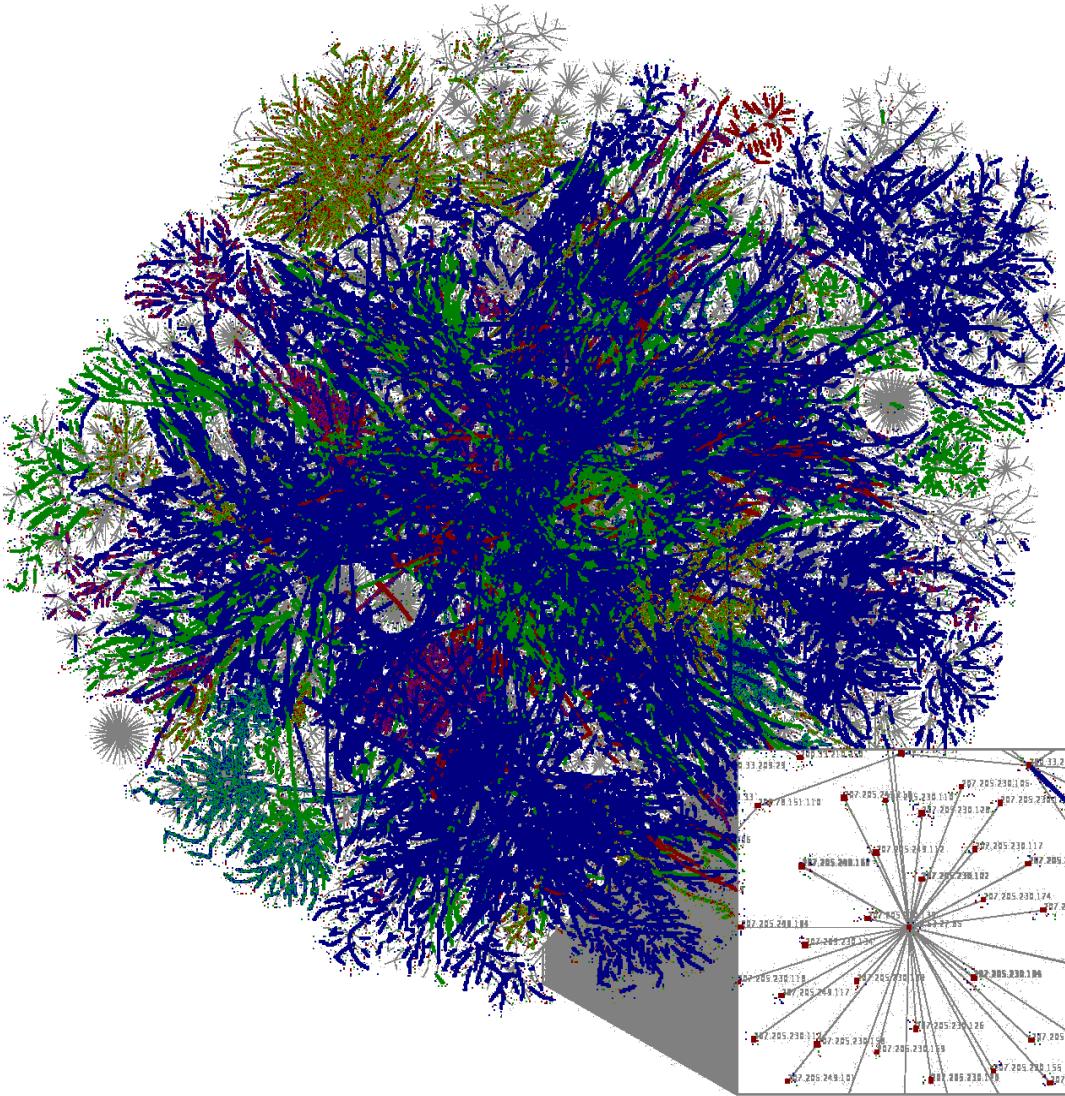
Briefly...

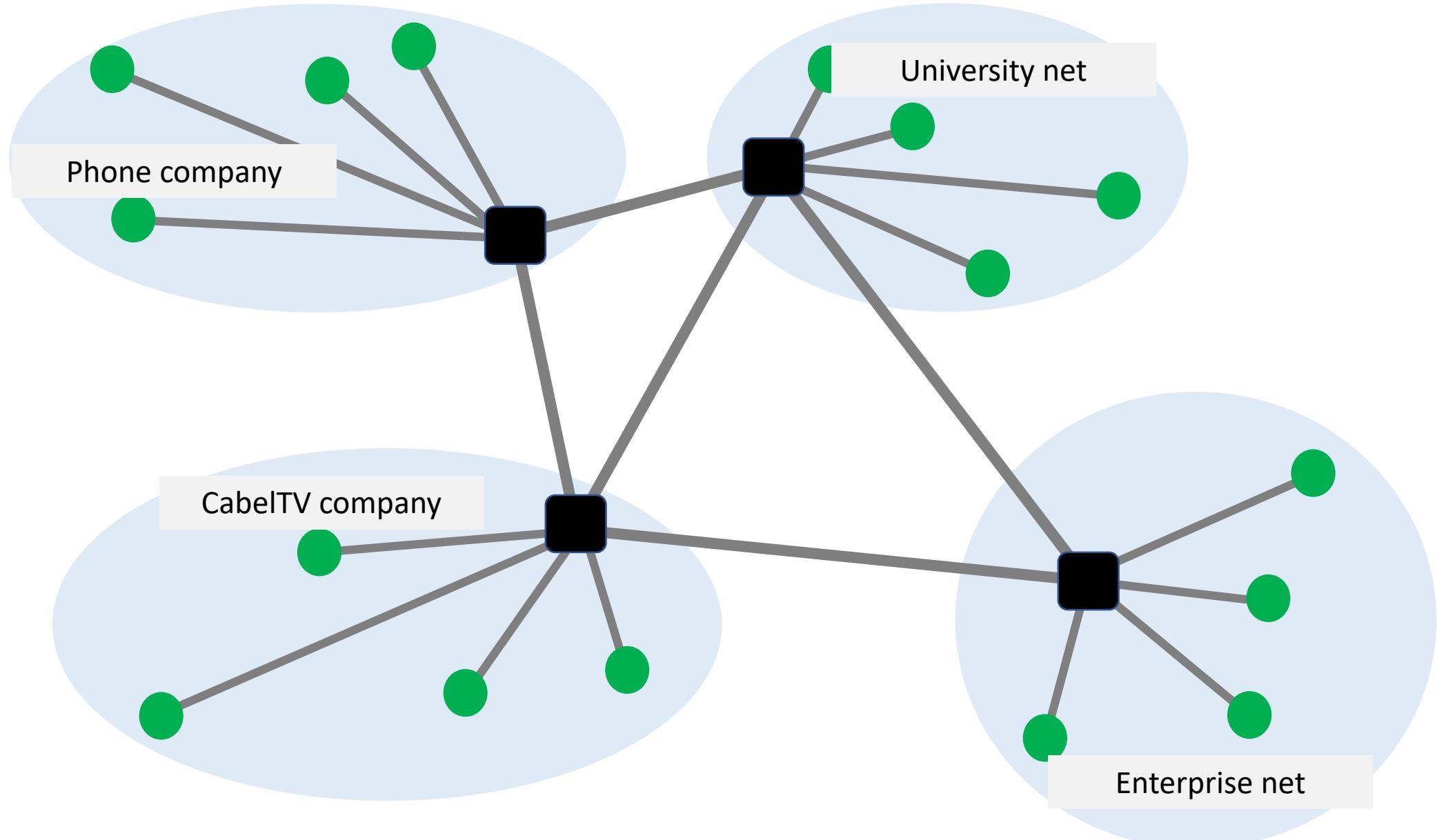
The Internet should allow

**processes on different hosts
to exchange data**

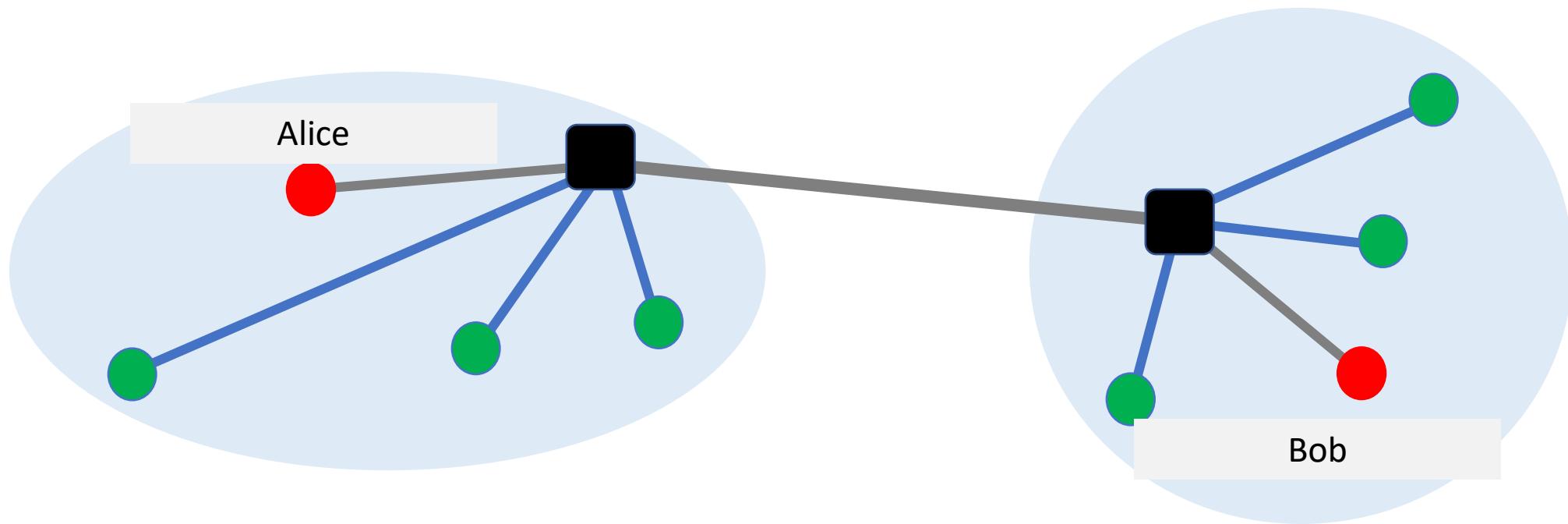
everything else is just commentary...

Ok, but how to do that in a complex system like the Internet?





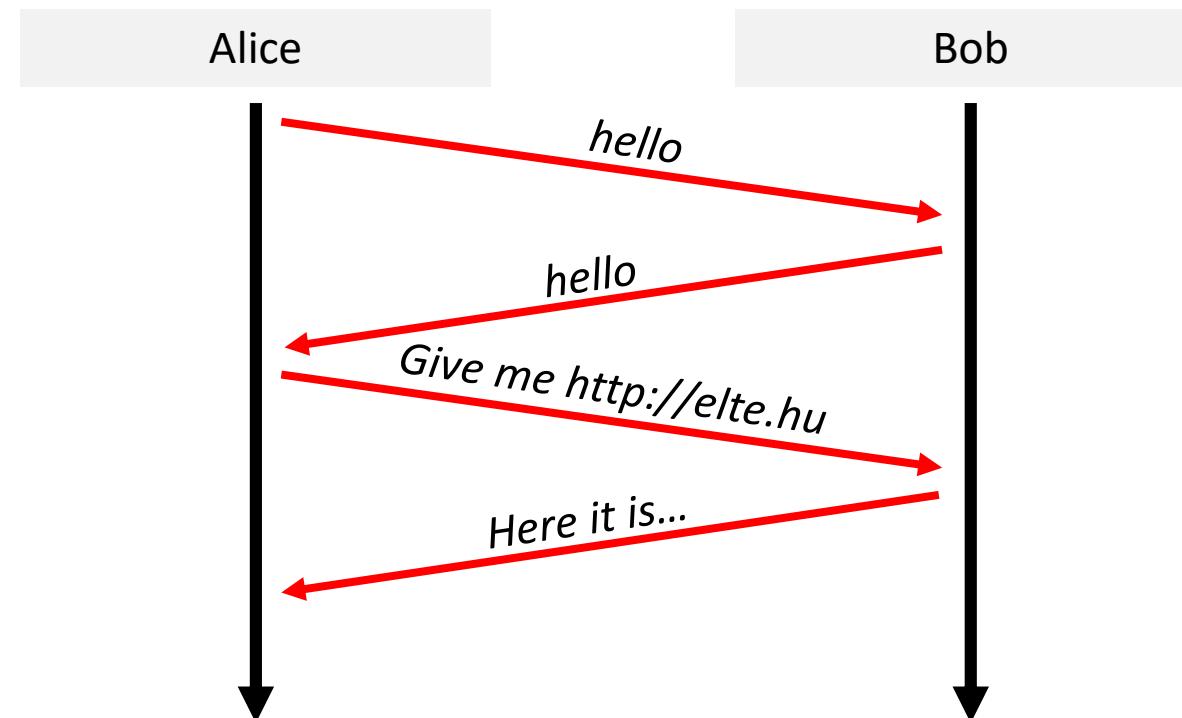
To exchange data, Alice and Bob use
a set of network protocols



A protocol is like a conversational convention

The protocol defines the order and rules the parties should follow

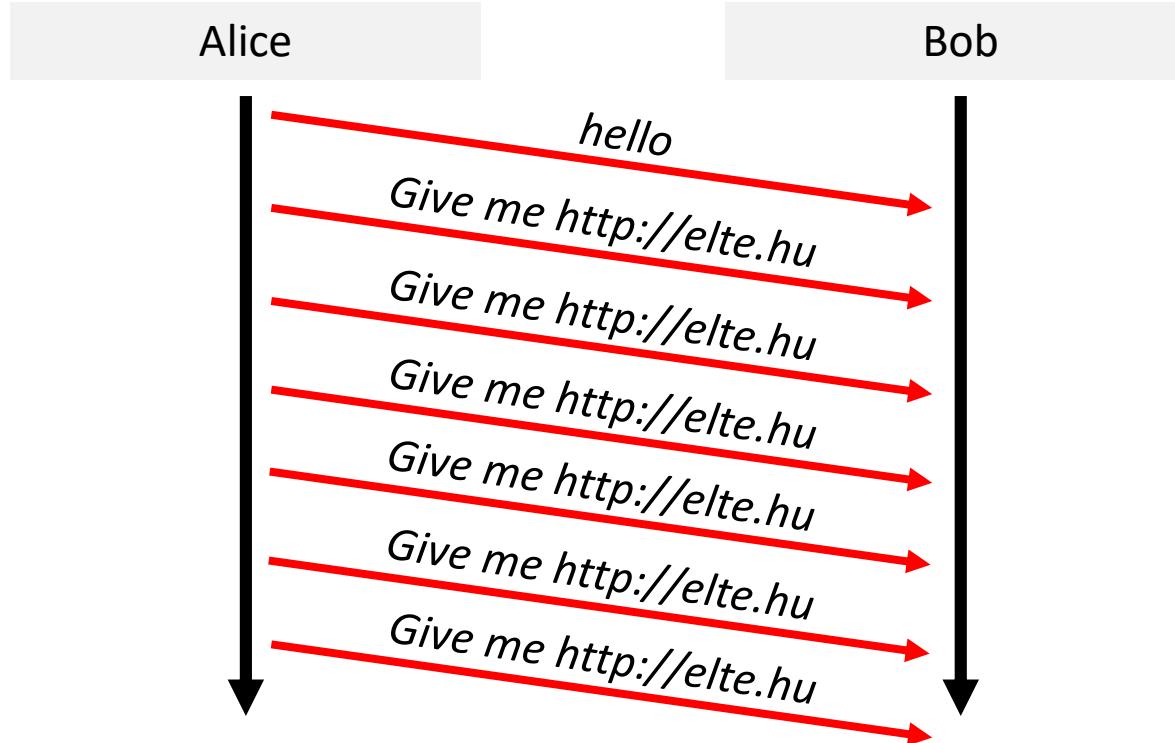
Who should talk next and how to respond...



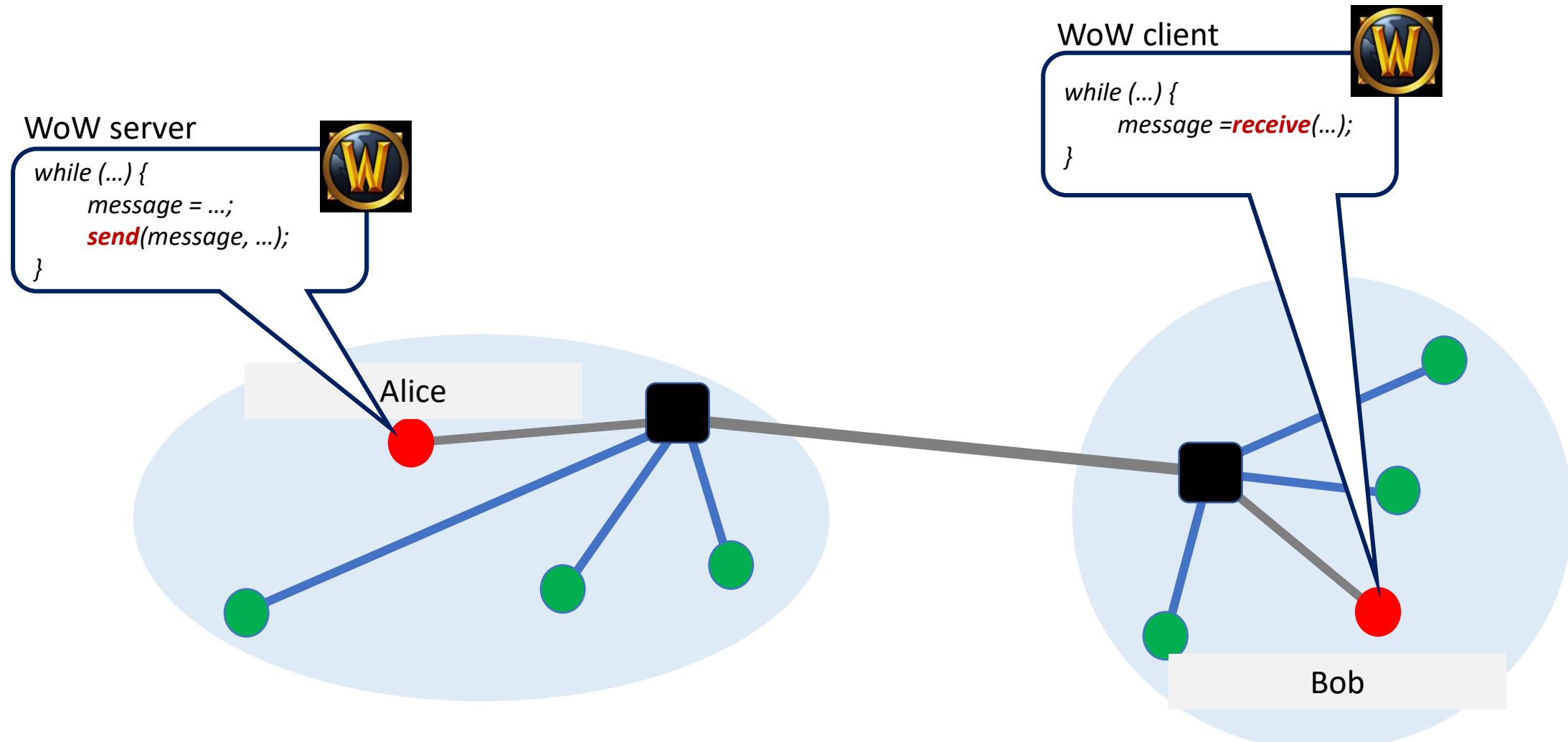
There are other kind of implementations...



Gimme, gimme, gimme
a web site after Midnight

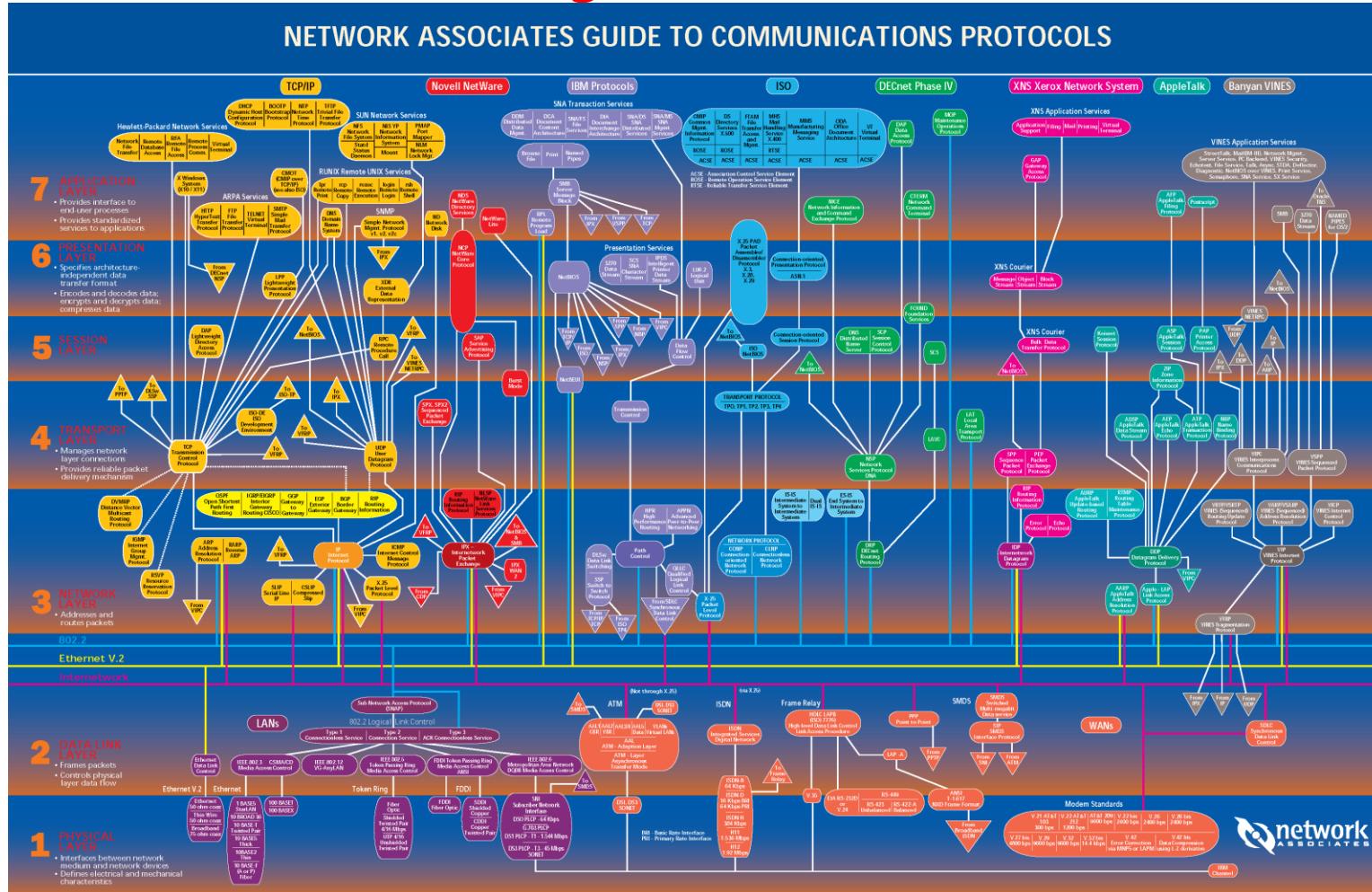


Each protocol is governed by a specific API



In practice, many existing protocols...

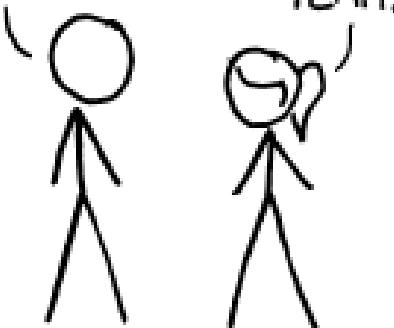
*How does the Internet **organize this**???*



HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)

SITUATION:
THERE ARE
14 COMPETING
STANDARDS.

14?! RIDICULOUS!
WE NEED TO DEVELOP
ONE UNIVERSAL STANDARD
THAT COVERS EVERYONE'S
USE CASES.



YEAH!

SOON:

SITUATION:
THERE ARE
15 COMPETING
STANDARDS.

Modularity is a key component of any good system

- | | |
|---|---|
| Problem | can't build large systems out of spaghetti code
<i>hard (if not, impossible) to understand, debug,
update</i> |
| | |
| need to bound the scope of changes
<i>evolve the system without rewriting it from scratch</i> | |
| | |
| Solution | Modularity is how we do it
<i>...and understand the system at a higher-level</i> |

A.M.
TURING
AWARD
2008



BARBARA LISKOV

Developed the Liskov substitution principle



„Modularity,
based on abstraction,
is **the way** things get done”

Barbara Liskov, MIT

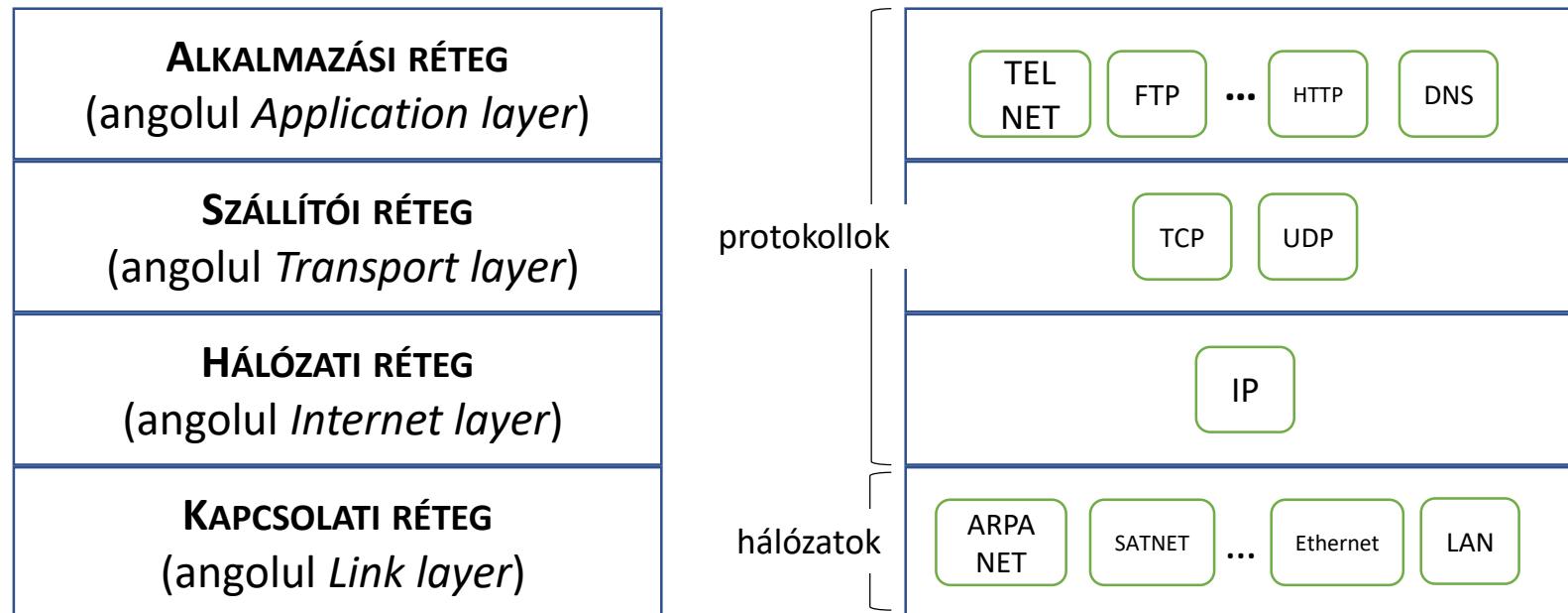
To provide structure to the design of network protocols,
network designers organize **protocols** in layers

**and the network hardware/software
that implement them**

Hálózatok modelljei

- Internet rétegmodelljei
 - TCP/IP modell: 4 réteget különböztet meg. 1982 márciusában az amerikai hadászati célú számítógépes hálózatok standardja lett. 1985-től népszerűsítették kereskedelmi felhasználásra. (*Interop*)
 - Hibrid TCP/IP modell: 5 réteget különböztet meg (*Tanenbaum, Stallings, Kurose, Forouzan*)
- Nyílt rendszerek hálózatának standard modellje
 - *Open System Interconnection Reference Model*: Röviden OSI referencia modell, amely egy 7-rétegű standard, koncepcionális modellt definiál kommunikációs hálózatok belső funkcionálásaihoz. (*ISO/IEC 7498-1*)

TCP/IP modell (RFC 1122)

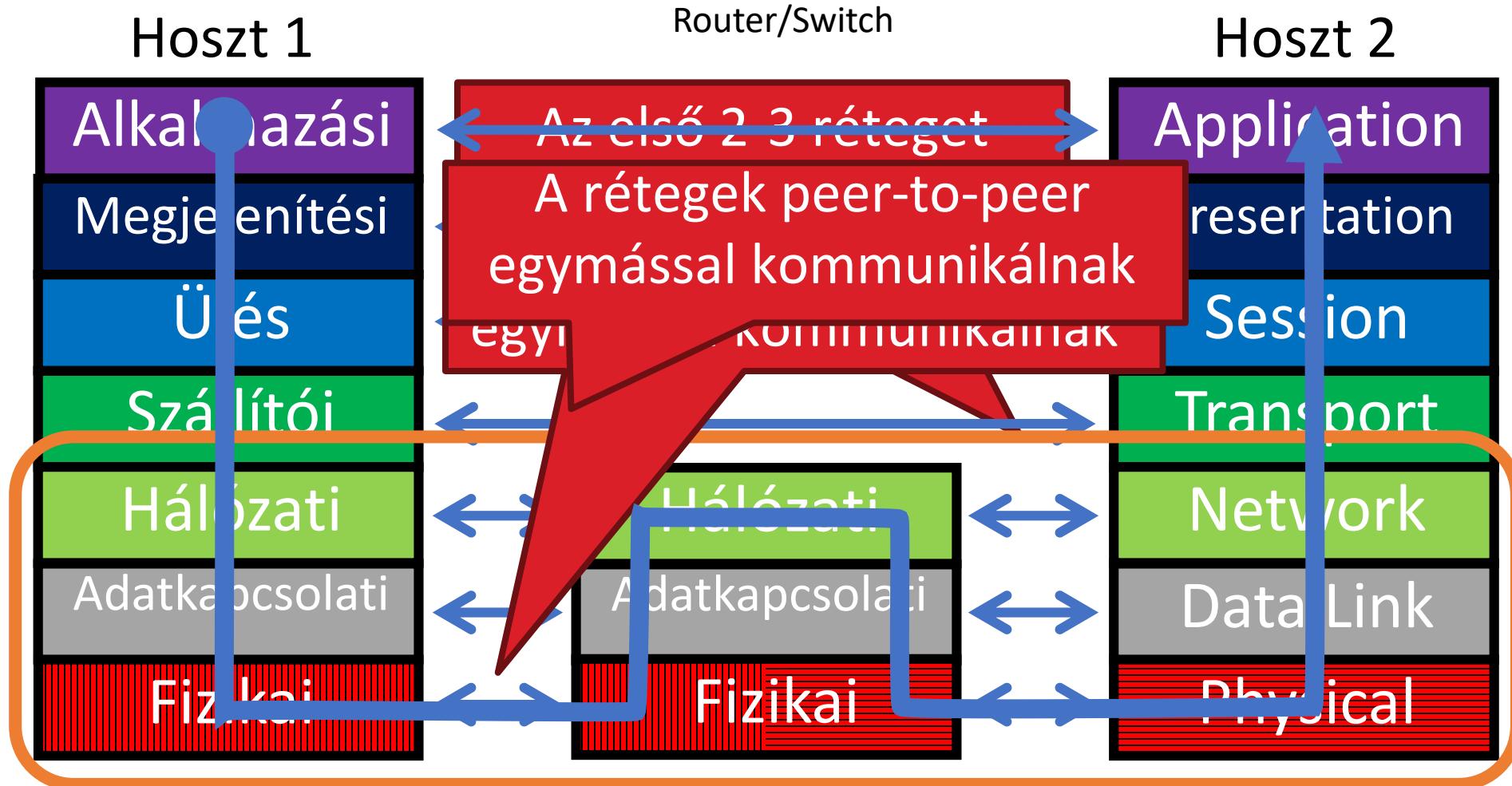


TCP/IP modell rétegei („bottom-up”)

- **Kapcsolati réteg / Host-to-network or Link layer**
 - nem specifikált
 - a LAN-tól függ
- **Internet réteg / Internet or Network layer**
 - speciális csomagformátum
 - útvonal meghatározás (routing)
 - csomag továbbítás (angolul *packet forwarding*)
- **Szállítói réteg / Transport layer**
 - **Transport Control Protocol**
 - megbízható, kétirányú bájt-folyam átviteli szolgáltatás
 - szegmentálás, folyamfelügyelet, multiplexálás
 - **User Datagram Protocol**
 - nem megbízható átviteli szolgáltatás
 - nincs folyamfelügyelet
- **Alkalmazási réteg / Application layer**
 - Szolgáltatások nyújtása: Telnet, FTP, SMTP, HTTP, NNTP, DNS, SSH, etc.

ISO OSI modell

OSI: Open Systems Interconnect Model



Rétegek jellemzése



- Szolgáltatás
 - Mit csinál az adott réteg?
- Interfész
 - Hogyan férhetünk hozzá a réteghez?
- Protokoll
 - Hogyan implementáljuk a réteget?

Fizikai réteg



- Szolgáltatás
 - Információt visz át két fizikailag összekötött eszköz között
 - definiálja az eszköz és a fizikai átviteli közeg kapcsolatát
- Interfész
 - Specifikálja egy bit átvitelét
- Protokoll
 - Egy bit kódolásának sémája
 - Feszültség szintek
 - Jelek időzítése
- Példák: koaxiális kábel, optikai kábel, rádió frekvenciás adó

Adatkapcsolati réteg



- Szolgáltatás
 - Adatok keretekre tördelése: határok a csomagok között
 - Közeghuzzáférés vezérlés (MAC)
 - Per-hop megbízhatóság és folyamvezérlés
- Interfész
 - Keret küldése két közös médiumra kötött eszköz között
- Protokoll
 - Fizikai címzés (pl. MAC address, IB address)
- Példák: Ethernet, Wifi, InfiniBand

Hálózati réteg



- Szolgáltatás
 - Csomagtovábbítás
 - Útvonalválasztás
 - Csomag fragmentálás kezelése
 - Csomag ütemezés
 - Puffer kezelés
- Interfész
 - Csomag küldése egy adott végpontnak
- Protokoll
 - Globálisan egyedi címeket definiálása
 - Routing táblák karbantartása
- Példák: Internet Protocol (IPv4), IPv6

Szállítói réteg



- Szolgáltatás
 - Multiplexálás/demultiplexálás
 - Torlódásvezérlés
 - Megbízható, sorrendhelyes továbbítás
- Interfész
 - Üzenet küldése egy célállomásnak
- Protokoll
 - Port szám
 - Megbízhatóság/Hiba javítás
 - Folyamfelügyelet
- Példa: UDP, TCP

Ülés v. Munkamenet réteg



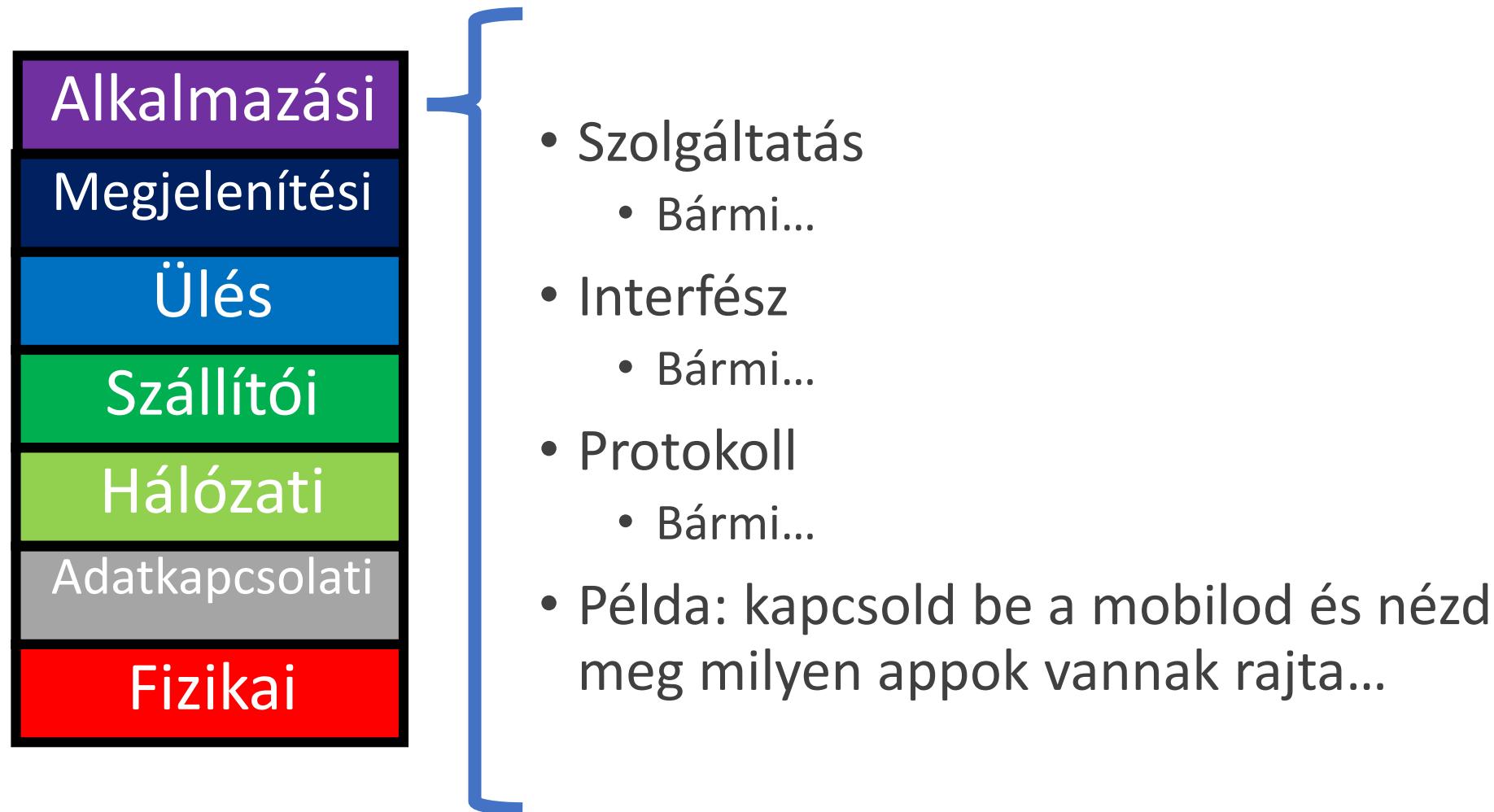
- Szolgáltatás
 - kapcsolat menedzsment: felépítés, fenntarás és bontás
 - munkamenet típusának meghatározása
 - szinkronizációs pont menedzsment (checkpoint)
- Interfész
 - Attól függ...
- Protokoll
 - Token menedzsment
 - Szinkronizációs checkpoints beszúrás
- Példa: nincs

Megjelenítési réteg



- Szolgáltatás
 - Adatkonverzió különböző reprezentációk között
 - Pl. big endian to little endian
 - Pl. Ascii to Unicode
- Interfész
 - Attól függ...
- Protokoll
 - Adatformátumokat definiál
 - Transzformációs szabályokat alkalmaz
- Példa: nincs

Alkalmazási réteg

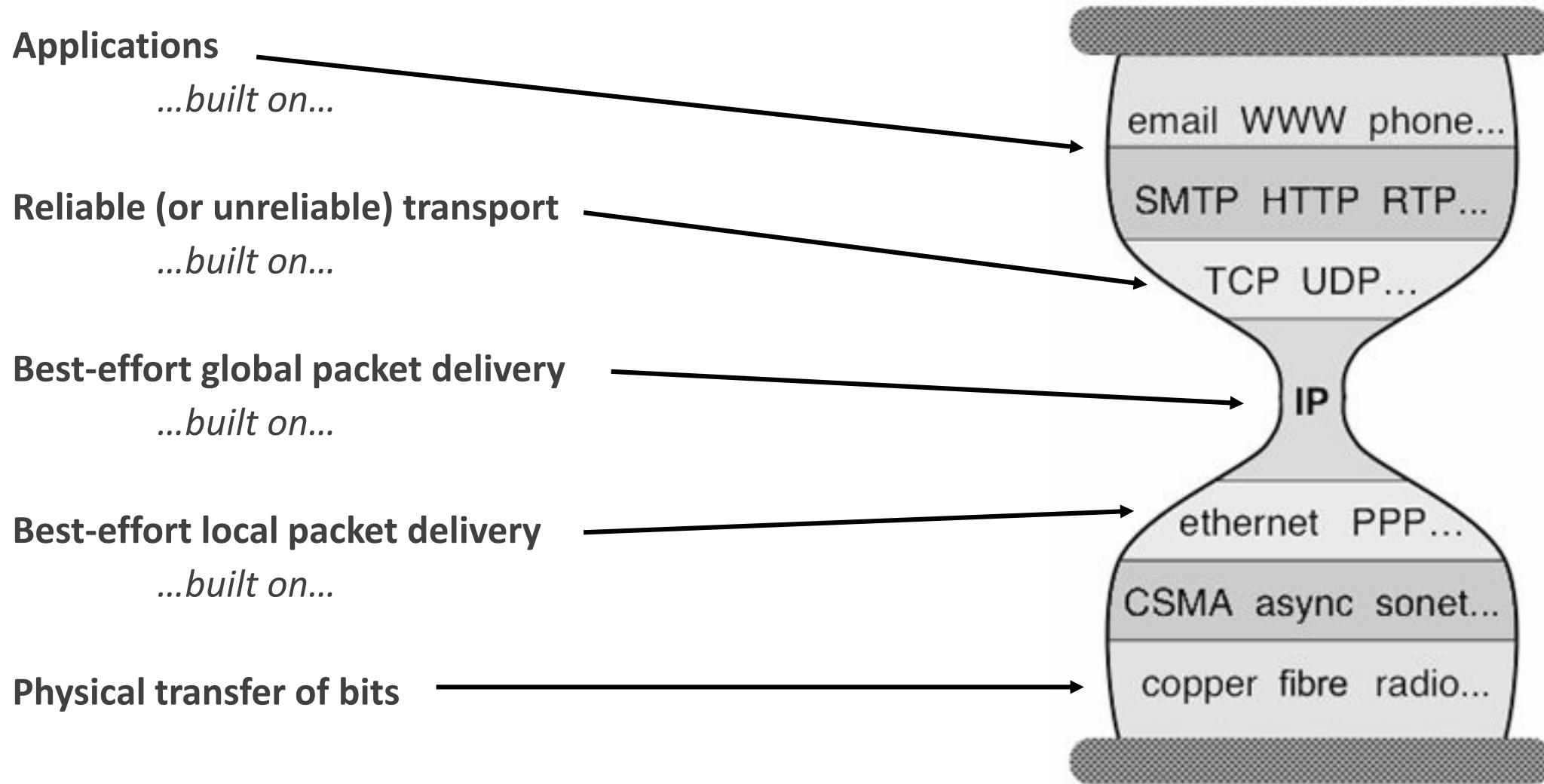


Hybrid model – 5 layers

Each layer provides a service to the layer above

	<u>layer</u>	<u>service provided</u>
L5	Application	<i>high level network access</i>
L4	Transport	<i>end-to-end delivery (reliable or not)</i>
L3	Network	<i>global best-effort delivery</i>
L2	Link	<i>local best-effort delivery</i>
L1	Physical	<i>physical transfer of bits</i>

Each layer provides a service to the layer above
by using the services of the layer directly below it



Each layer has a **unit of data**
(aka protocol data unit)

	<u>layer</u>	<u>role (PDU)</u>
L5	Application	exchanges messages between processes
L4	Transport	transports segments between end-systems
L3	Network	moves packets around the network
L2	Link	moves frames across a link
L1	Physical	moves bits across a physical medium

Each layer (except for L3) is implemented with different protocols

	<u>layer</u>	<u>protocols</u>
L5	Application	HTTP, SMTP, FTP, SIP, ...
L4	Transport	TCP, UDP, SCTP
L3	Network	IP
L2	Link	Ethernet, Wifi, ADSL, WiMAX, LTE, ...
L1	Physical	Twisted pair, fiber, coaxial cable, ...

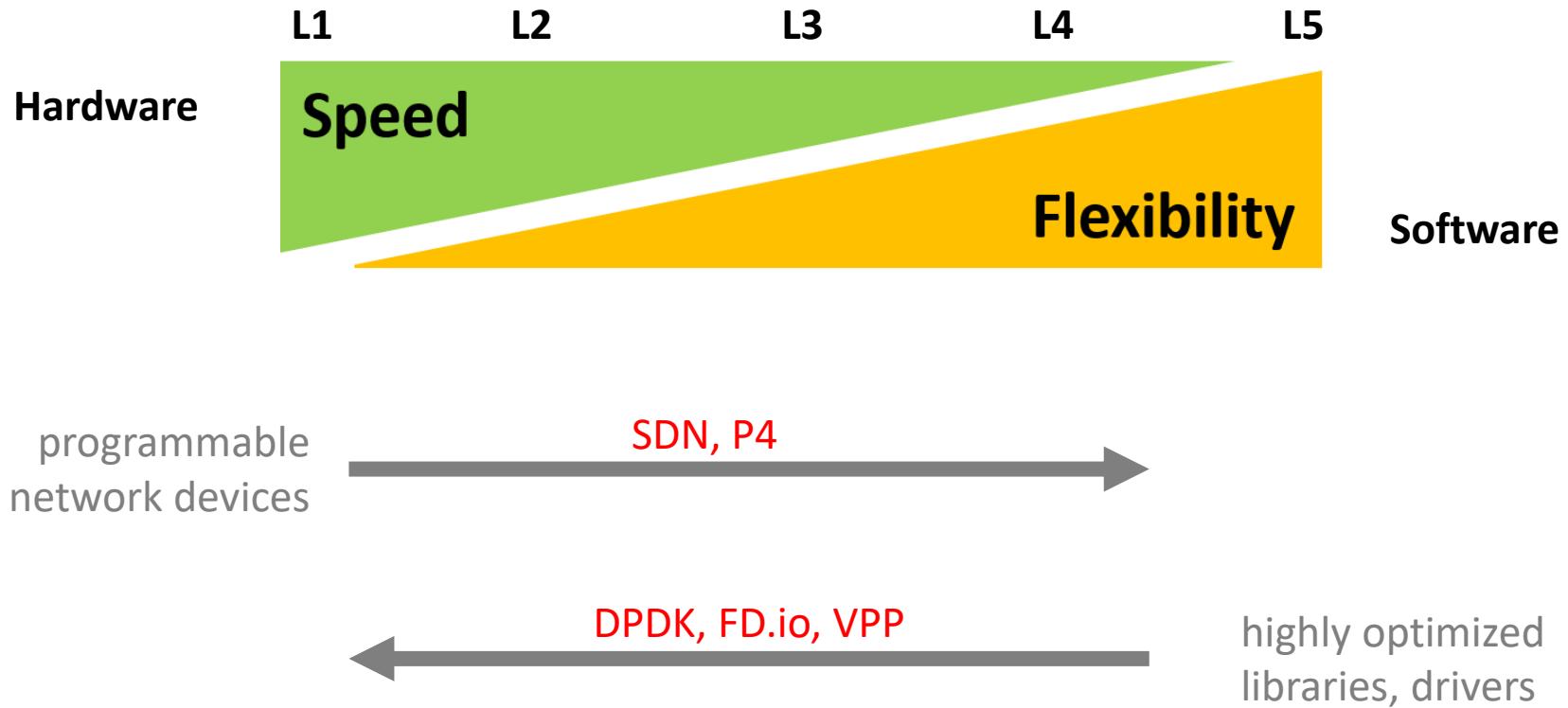
The Internet Protocol (IP) is the **glue**
acting as a unifying network layer

	<u>layer</u>	<u>protocols</u>
L5	Application	HTTP, SMTP, FTP, SIP, ...
L4	Transport	TCP, UDP, SCTP
L3	Network	IP
L2	Link	Ethernet, Wifi, ADSL, WiMAX, LTE, ...
L1	Physical	Twisted pair, fiber, coaxial cable, ...

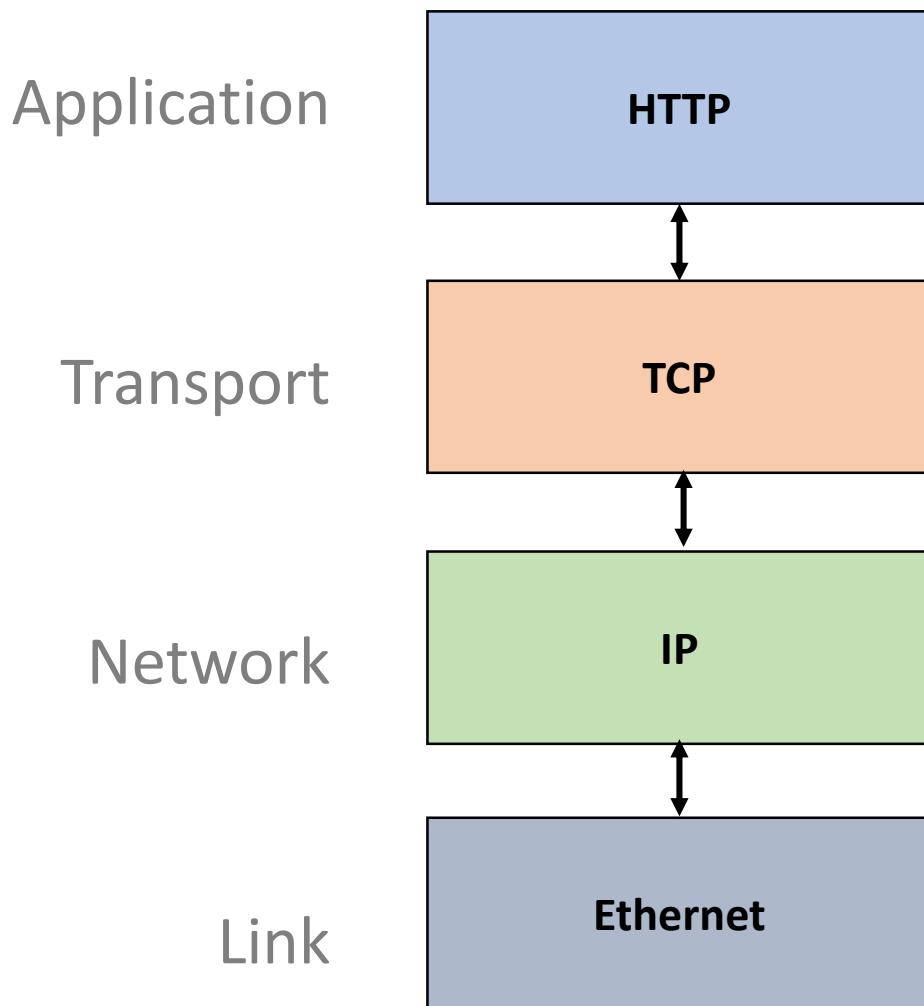
Each layer is implemented with different protocols and **technologies**

	<u>layer</u>	<u>technology</u>
L5	Application	Software
L4	Transport	Hardware
L3	Network	
L2	Link	
L1	Physical	

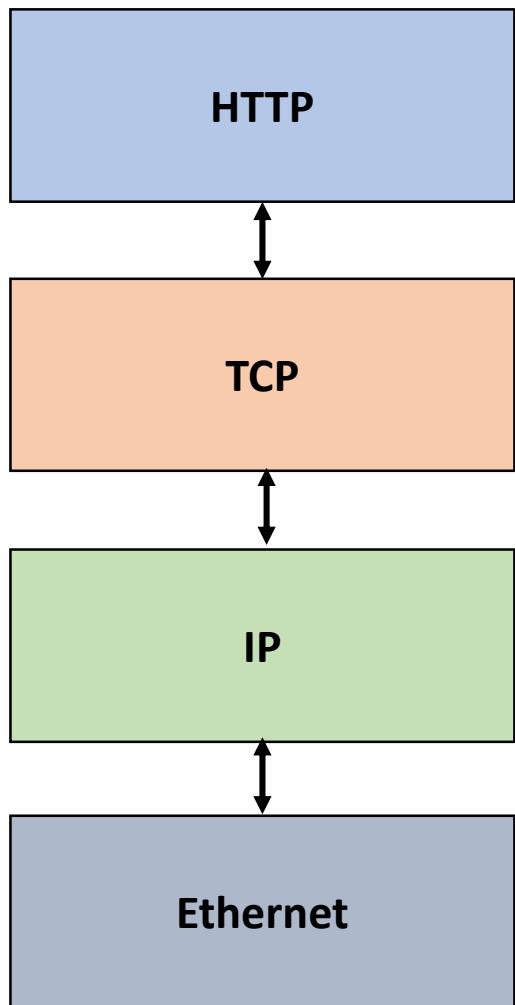
Software and hardware advancements



Each layer takes messages from the layer above,
and ***encapsulates*** with its own header and/or trailer



Application



Transport

Network

Link

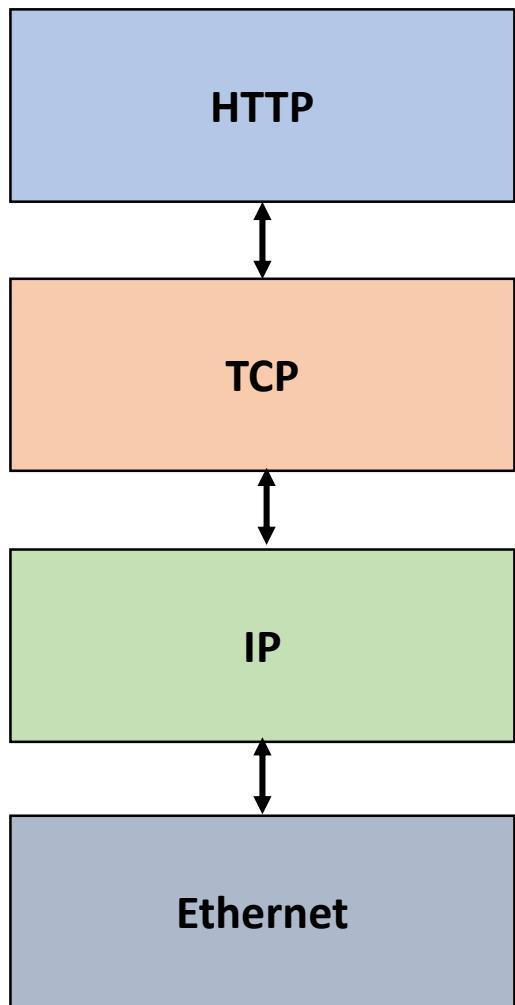
Header

HA

Message

GET google.com

Application



Transport

Network

Link

Header

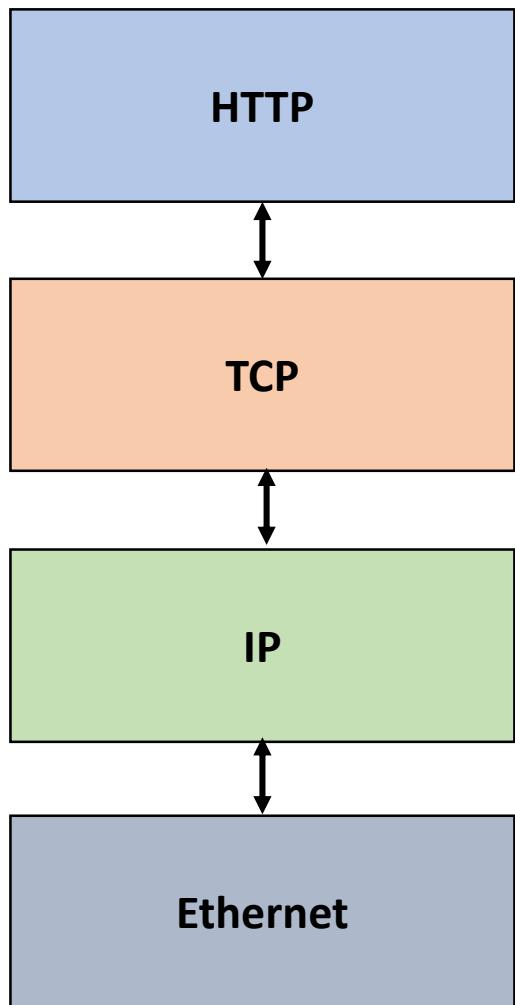
HA

Message

GET google.com



Application



Transport

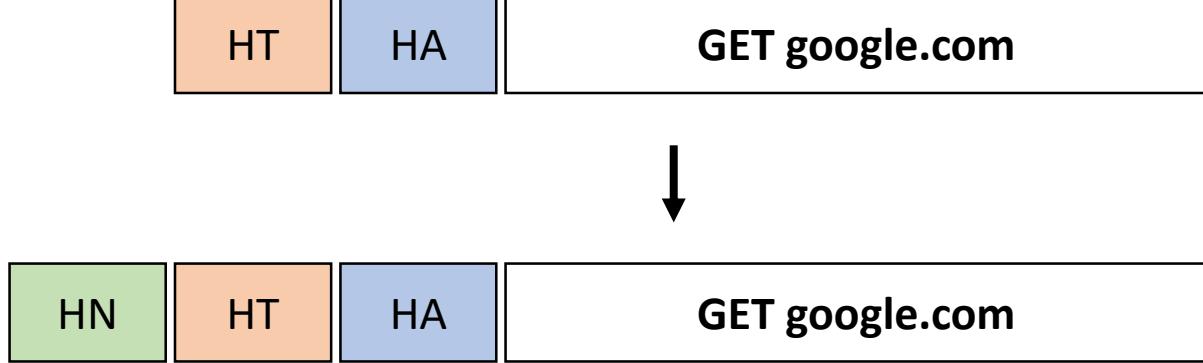
Header

HA

Message

GET google.com

Network



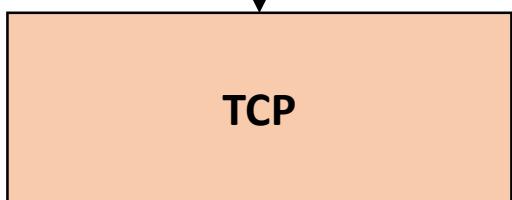
Link

Application



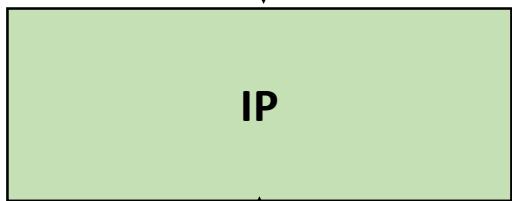
HTTP

Transport



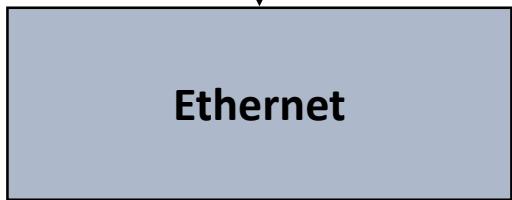
TCP

Network



IP

Link



Ethernet

Header

HA

Message

GET google.com



HT

HA

GET google.com



HN

HT

HA

GET google.com



HE

HN

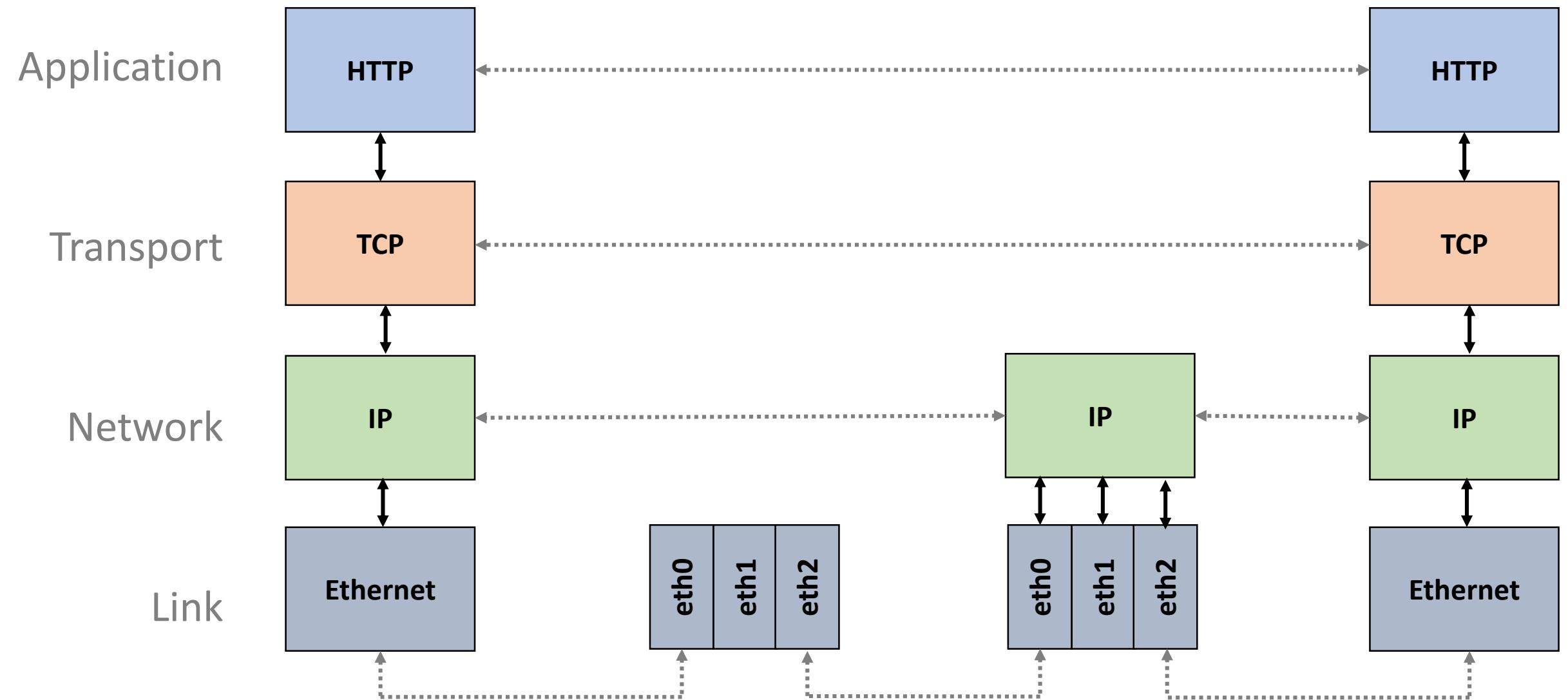
HT

HA

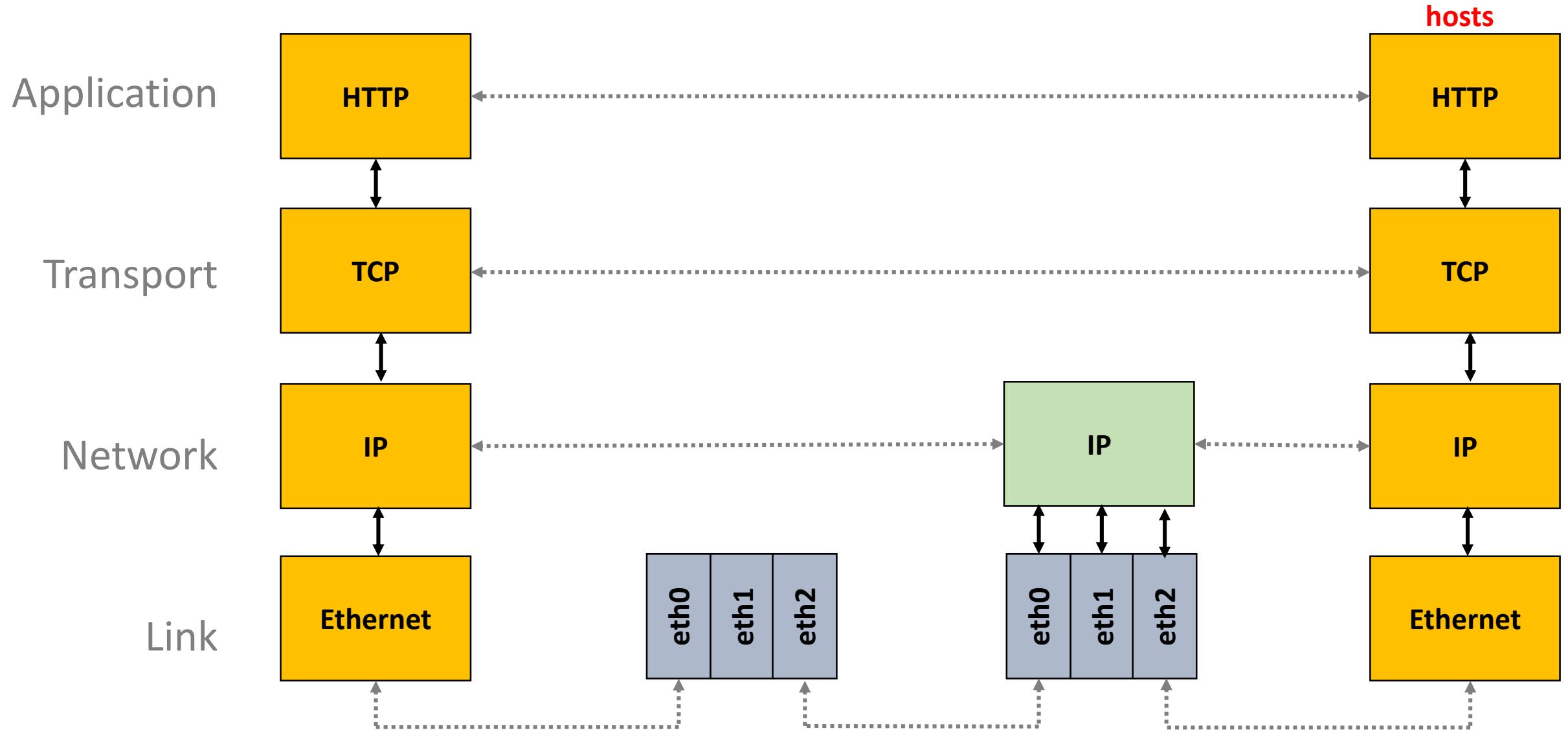
GET google.com

TE

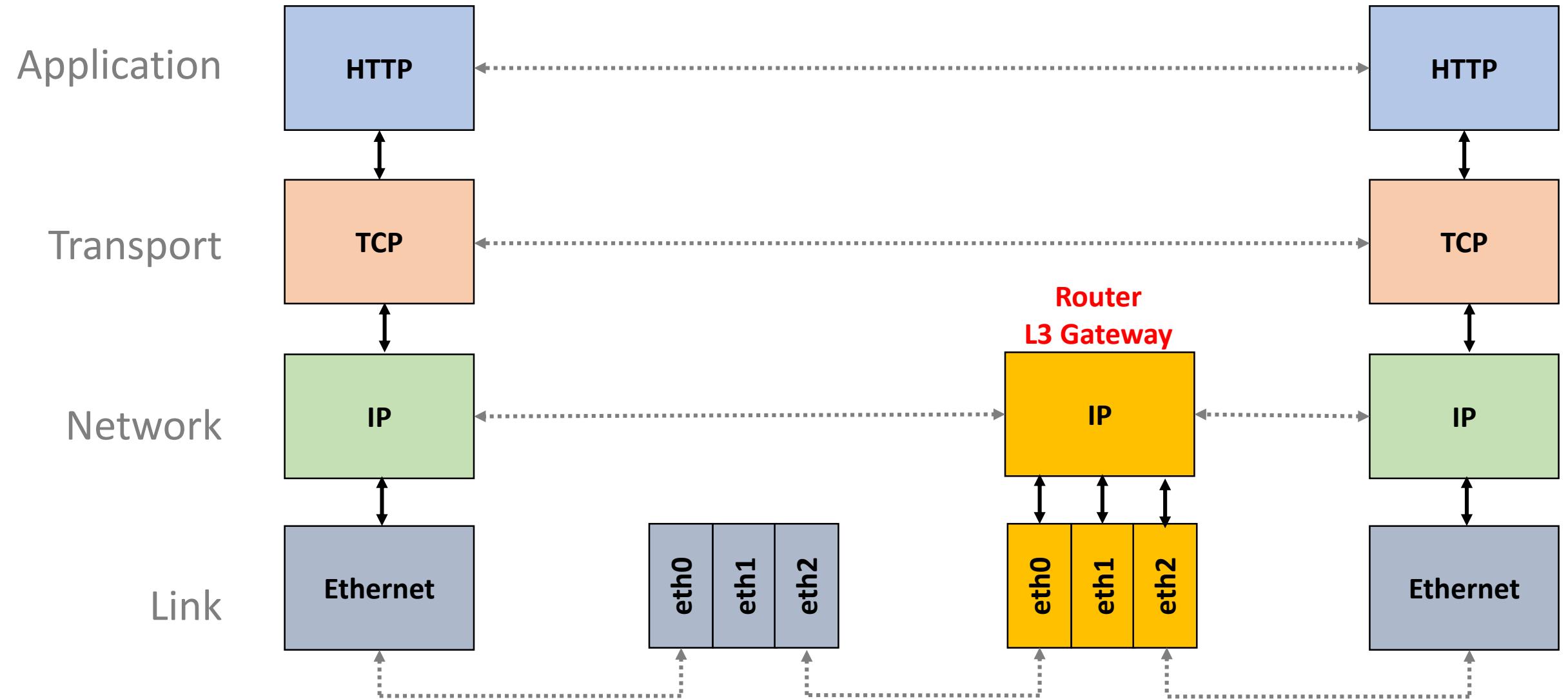
In practice, layers are **distributed**
on every network device



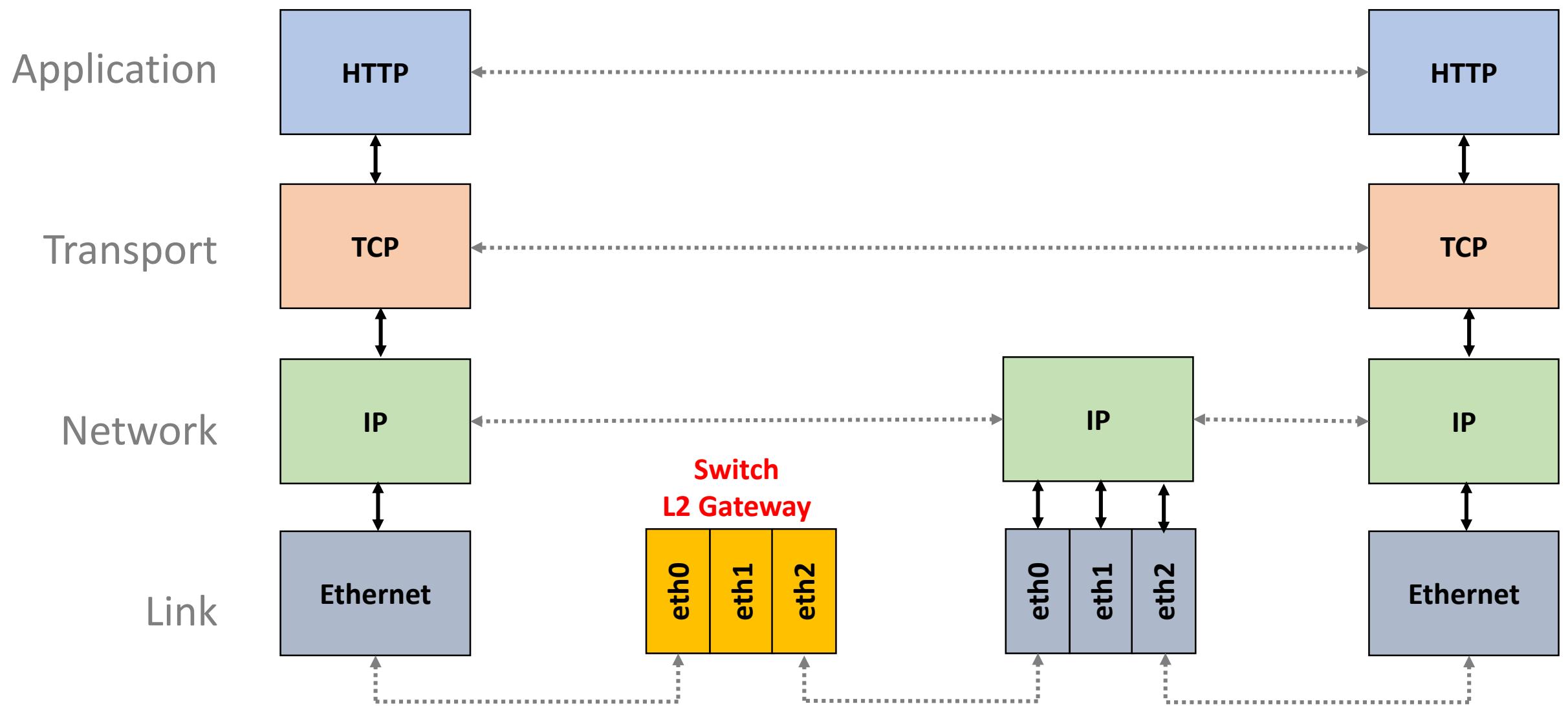
Since when bits arrive they must make it to the application, all the layers exist on a host



Routers act as **L3 gateway**
as such they implement L2 and L3



Switches act as **L2 gateway**
as such they only implement L2



Overview

How do we characterize the network?

A network *connection* is characterized by its **delay**, **loss rate** and **throughput**



How long does it take for a packet to reach the destination

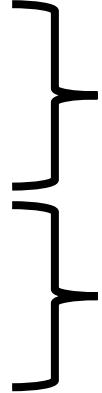
What fraction of packets sent to a destination are dropped?

At what rate is the destination receiving data from the source?

Delay



Each packet suffers from several types of delays
at *each node* along the path

- transmission delay**
 - + **propagation delay**
 - + **processing delay**
 - + **queueing delay**
- 
- due to **link properties**
- due to **traffic mix & switch internals**

= **total delay**

Each packet suffers from several types of delays
at *each node* along the path

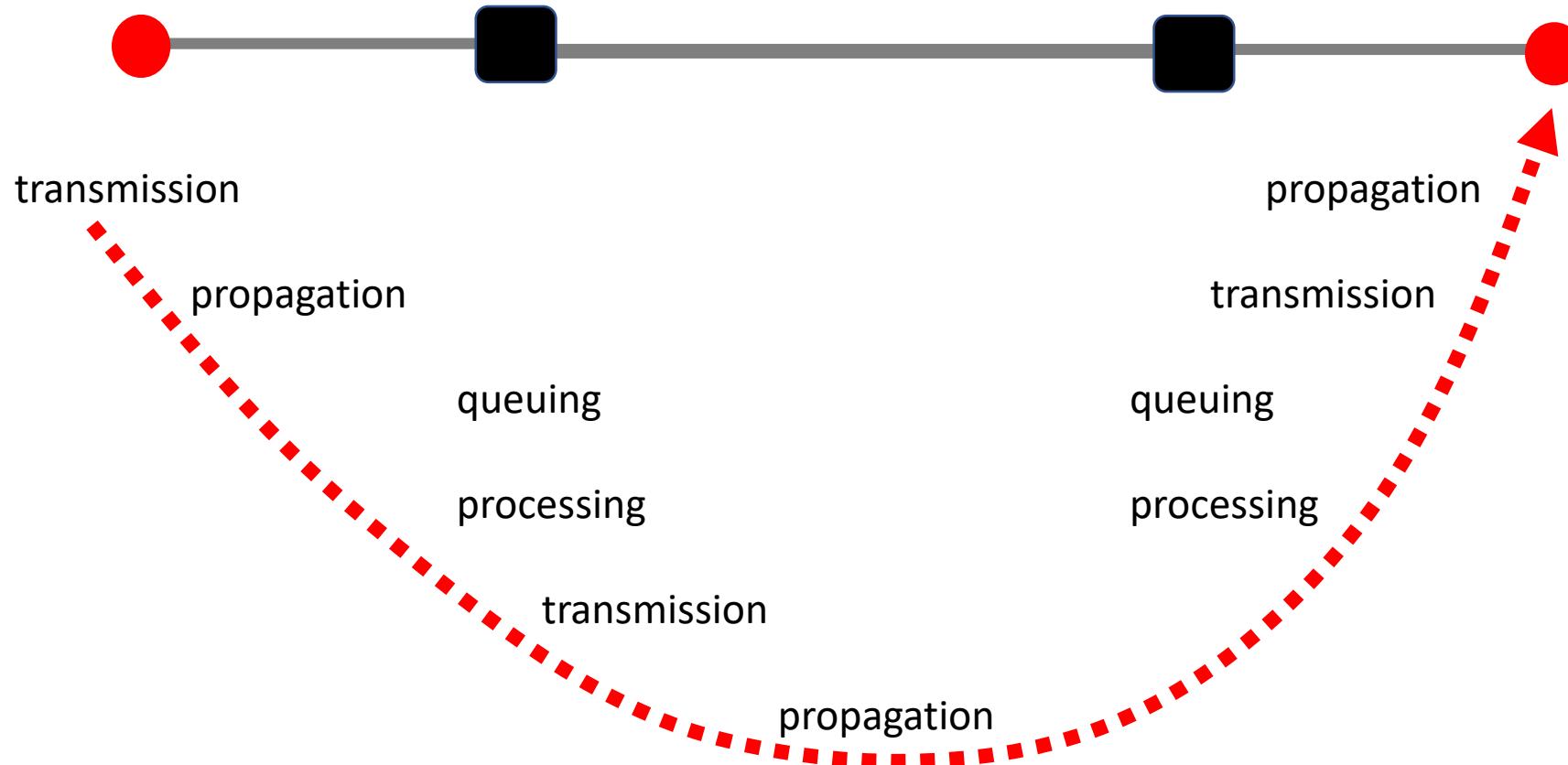
transmission delay

+ **propagation delay**

~~+ **processing delay**~~ **tend to be tiny**

+ **queueing delay**

= **total delay**



The transmission delay is the amount of time required to push all of the bits onto the link

$$\text{Transmission delay [sec]} = \frac{\text{packet size} [\#bits]}{\text{link bandwidth} [\#bits/sec]}$$

Example

$$= \frac{1000 \text{ bits}}{100 \text{ Mpbs}} = 10 \mu\text{sec}$$

The propagation delay is the amount of time required for a bit to travel to the end of the link

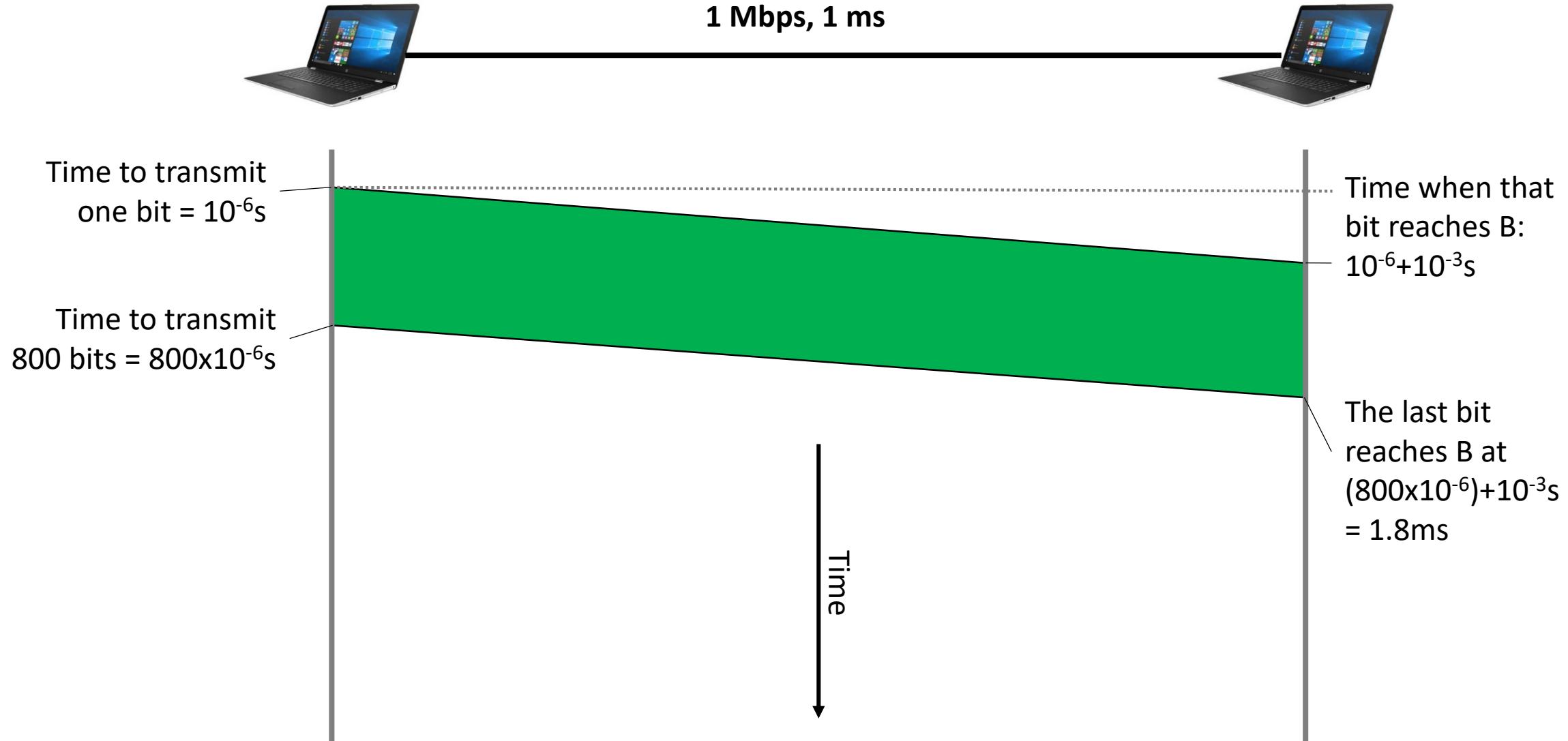
$$\text{Propagation delay} = \frac{\text{link length}}{\text{signal propagation speed}}$$

Example

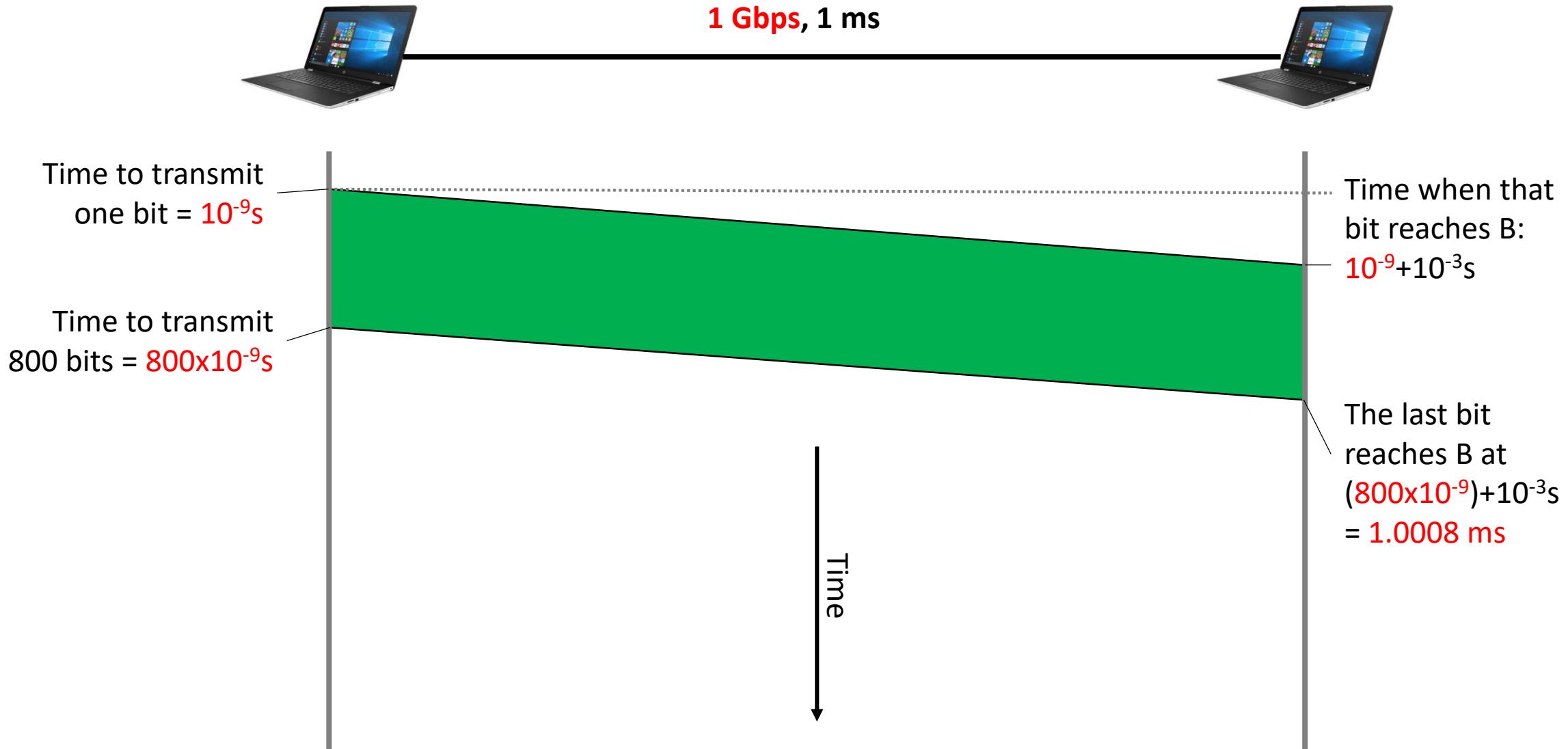
$$= \frac{30000 \text{ m}}{2 \times 10^8 \text{ m/sec}} = 150 \mu\text{sec}$$

(speed of light in fiber)

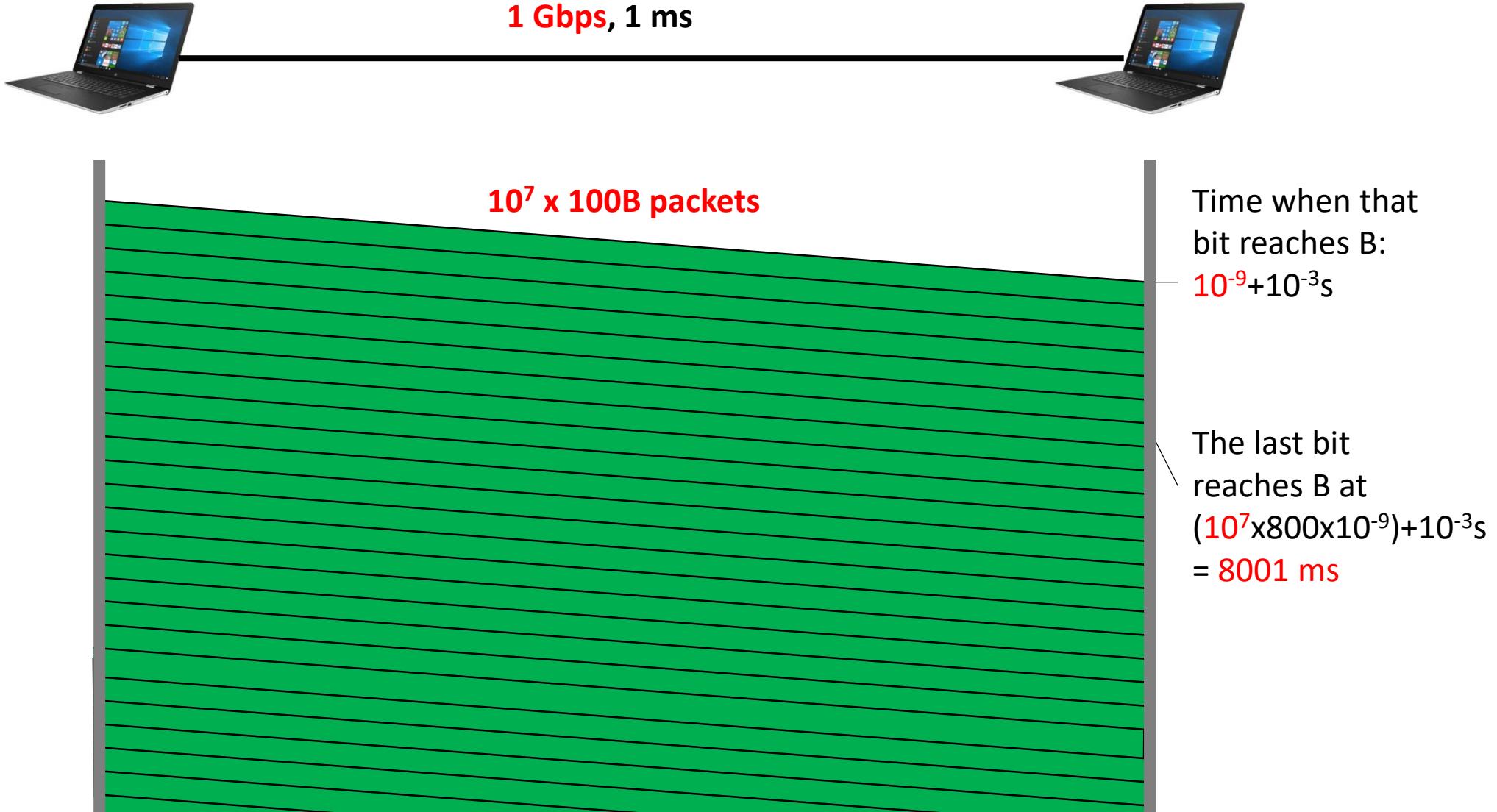
How long does it take to exchange 100 Bytes packet?



If we have a 1 Gbps link,
the total time decreases to **1.0008 ms**



If we now exchange a 1GB file
split in 100B packets



Different transmission characteristics imply different tradeoffs in terms of which delay dominates

$10^7 \times 100B$ pkts 1Gbps link ***transmission delay dominates***

1x100B pkt 1Gbps link ***propagation delay dominates***

1x100B pkt 1Mbps link ***both matter***

In the Internet, we cannot know in advance which one matter!

The queuing delay is the amount of time a packet **waits** (in a buffer) to be transmitted on a link

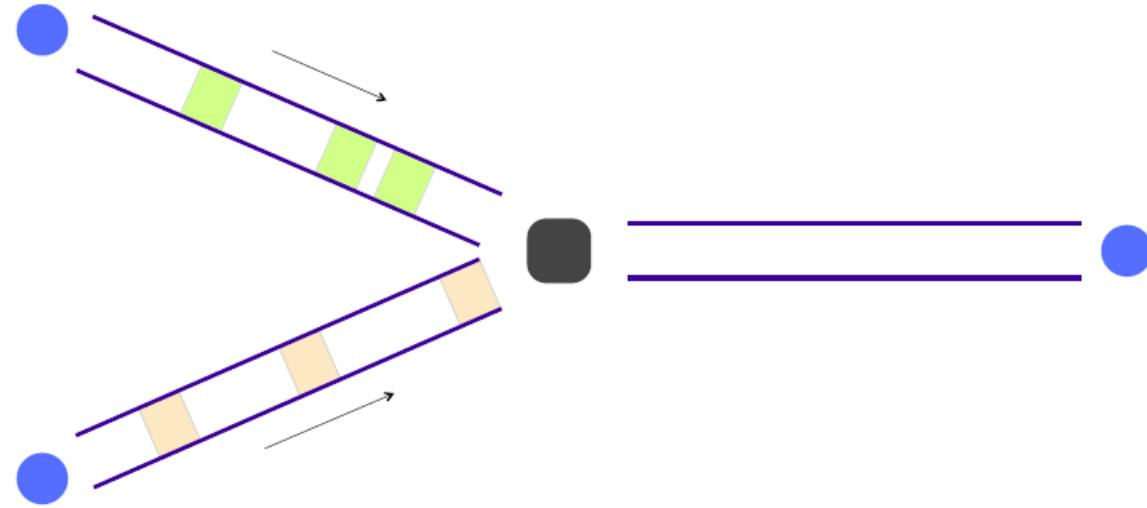
Queuing delay is the hardest to evaluate

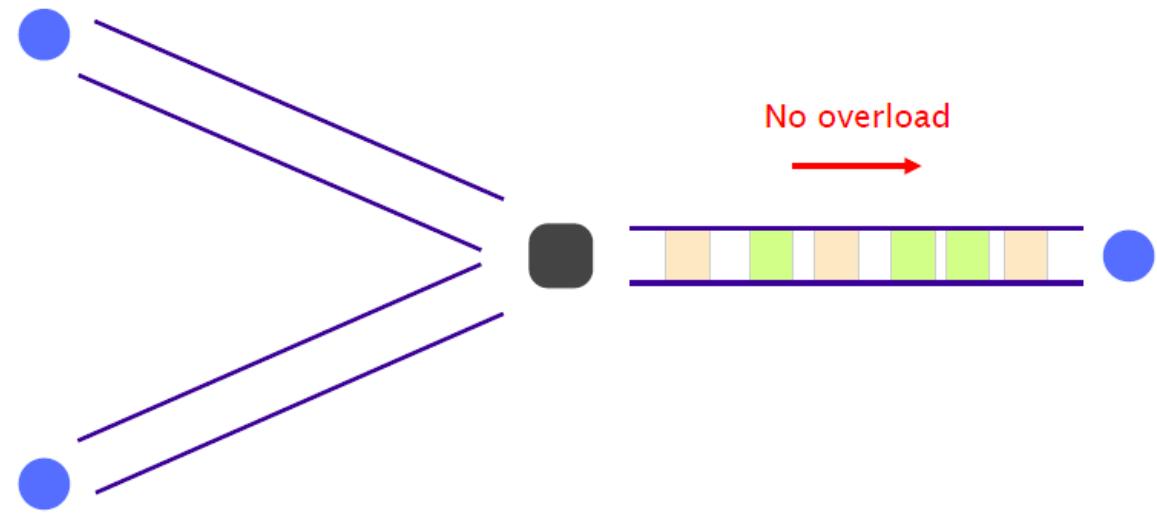
as it varies from packet to packet

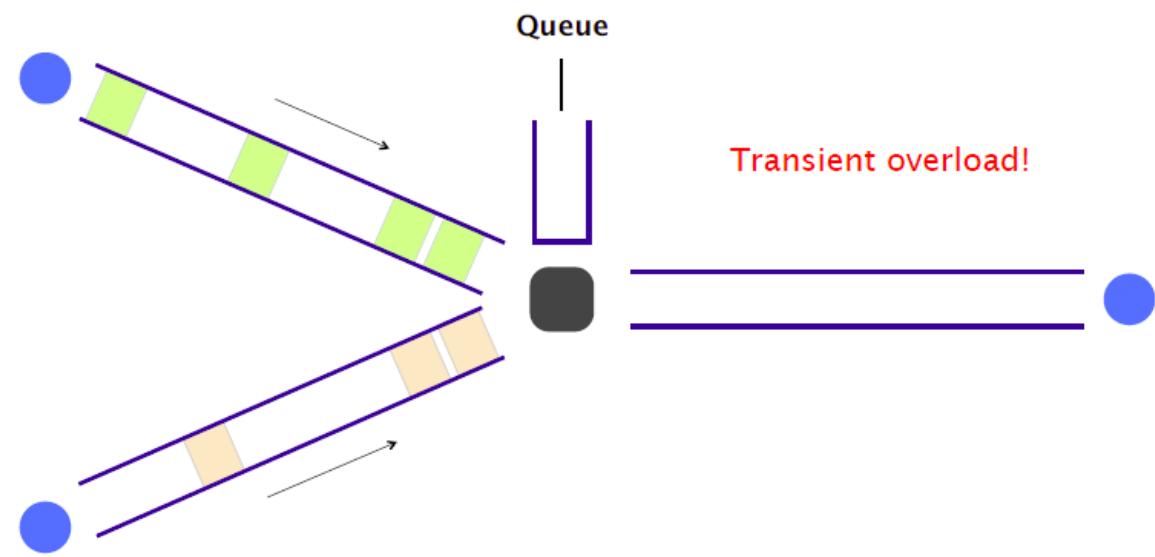
It is characterized with statistical measures

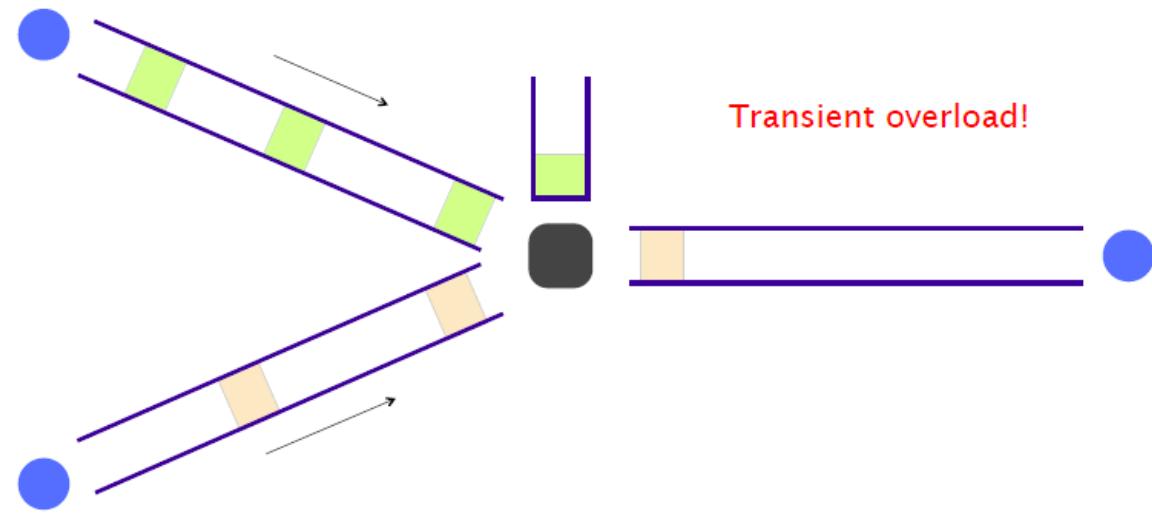
e.g., average delay & variance, probability of exceeding x

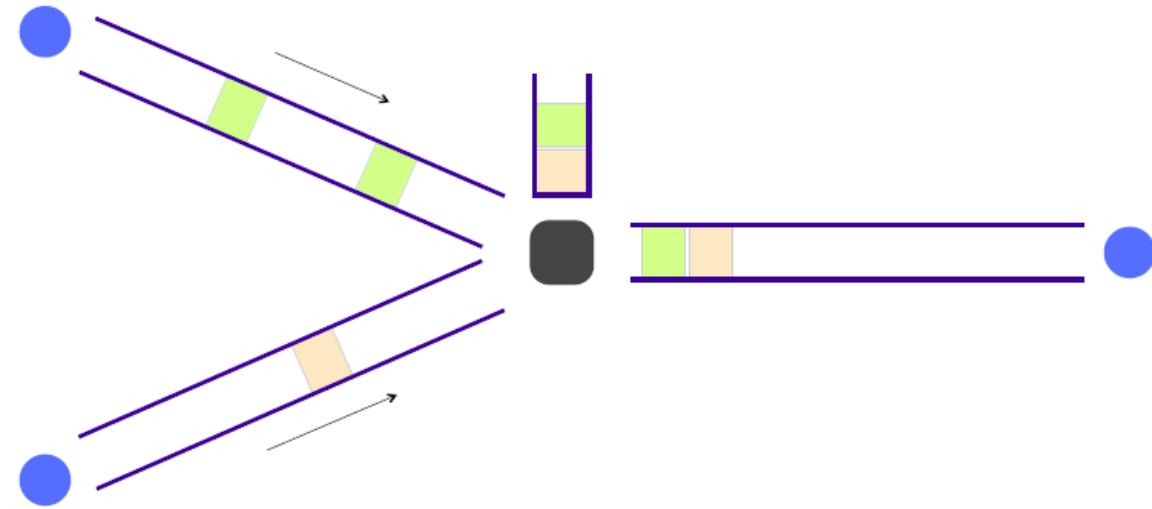
Queuing delay depends on the traffic pattern

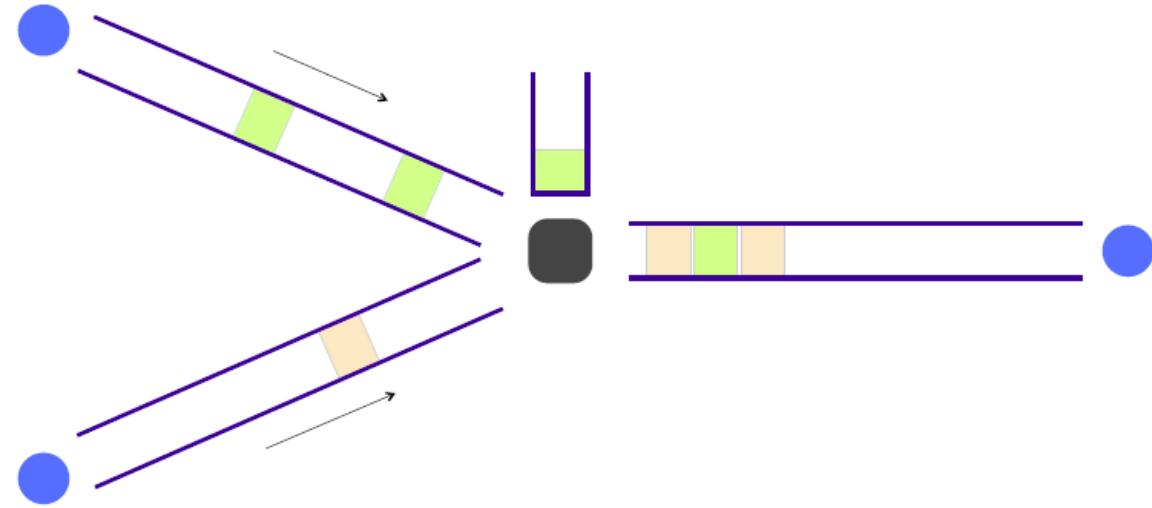


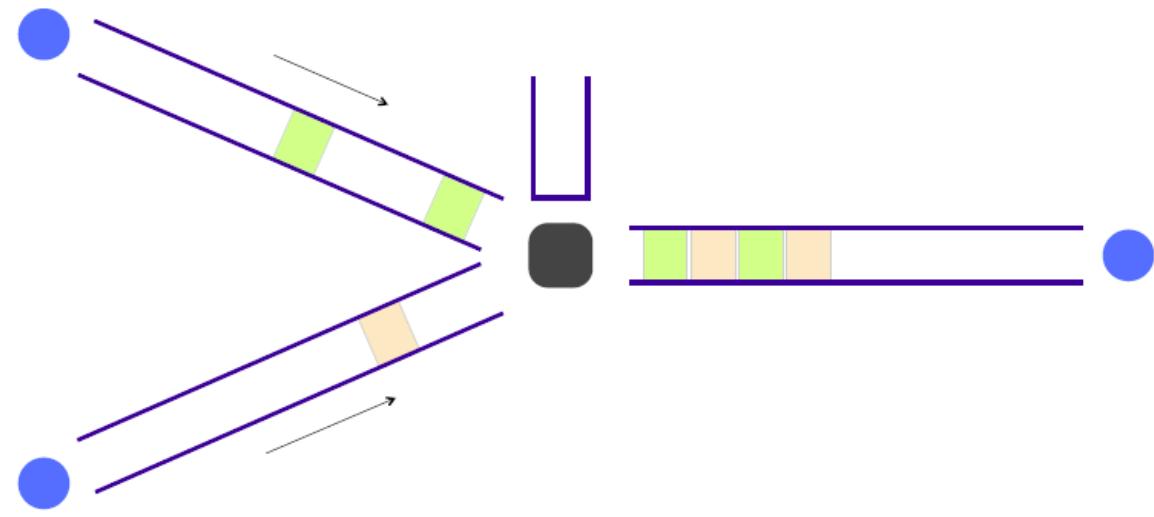




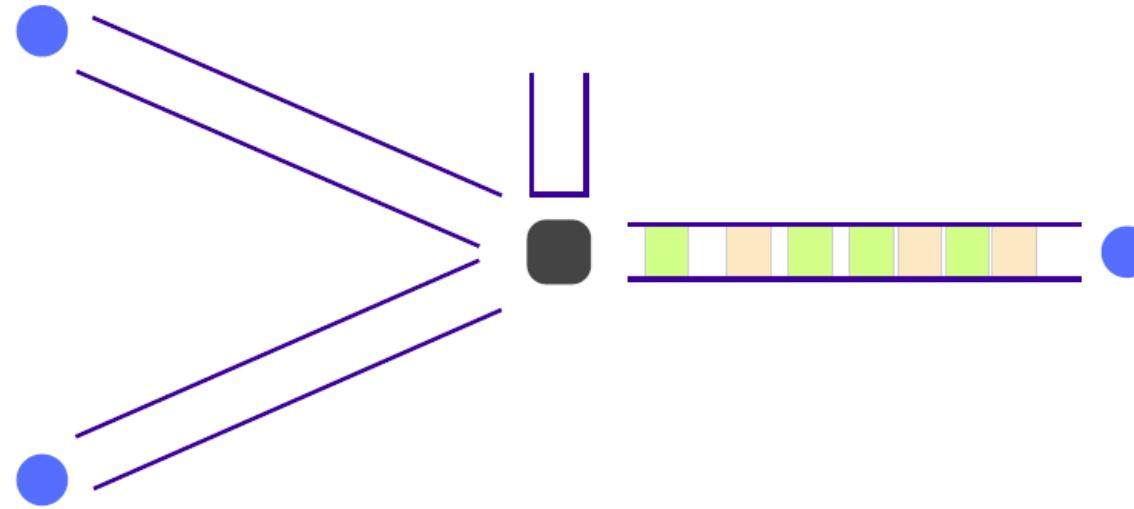








Queues absorb transient bursts,
but introduce queueing delays



The time a packet has to sit in a buffer before being processed depends on the traffic pattern

Queueing delay depends on:

arrival rate at the queue

transmission rate of the outgoing link

traffic **burstiness**

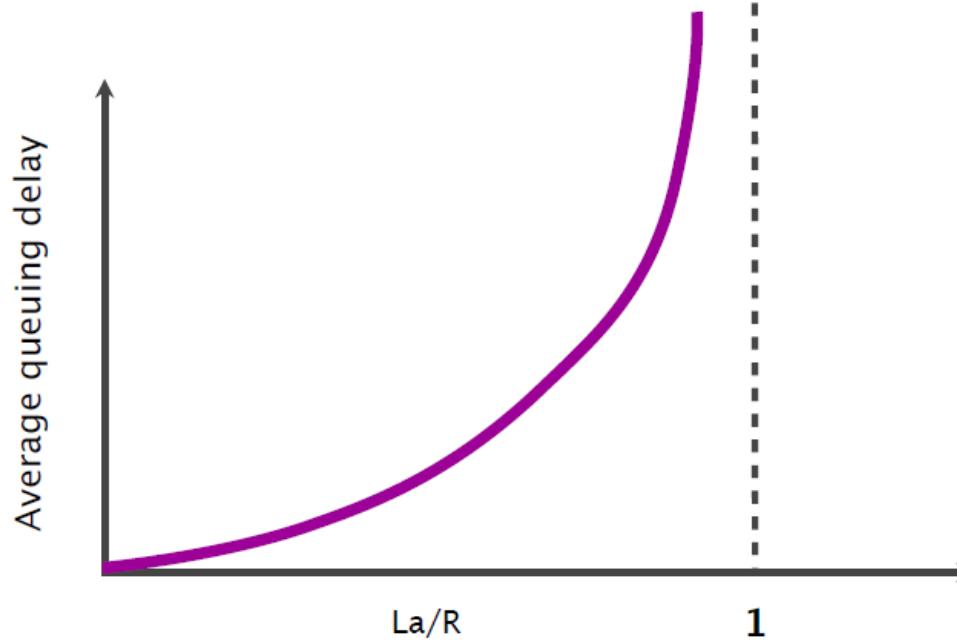
- average packet arrival rate a [packet/sec]
 - transmission rate of outgoing link R [bit/sec]
 - fixed packets length L [bit]
-
- average bits arrival rate La [bit/sec]
 - traffic intensity La/R

When the traffic intensity is >1 , the queue will increase without bound, and so does the queuing delay

Golden rule

**Design your queuing system,
so that it operates far from that point**

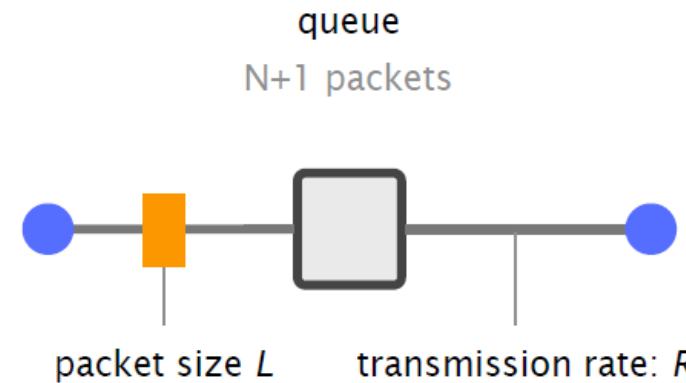
When the traffic intensity is ≤ 1 ,
queueing delay depends on the burst size



LOSS

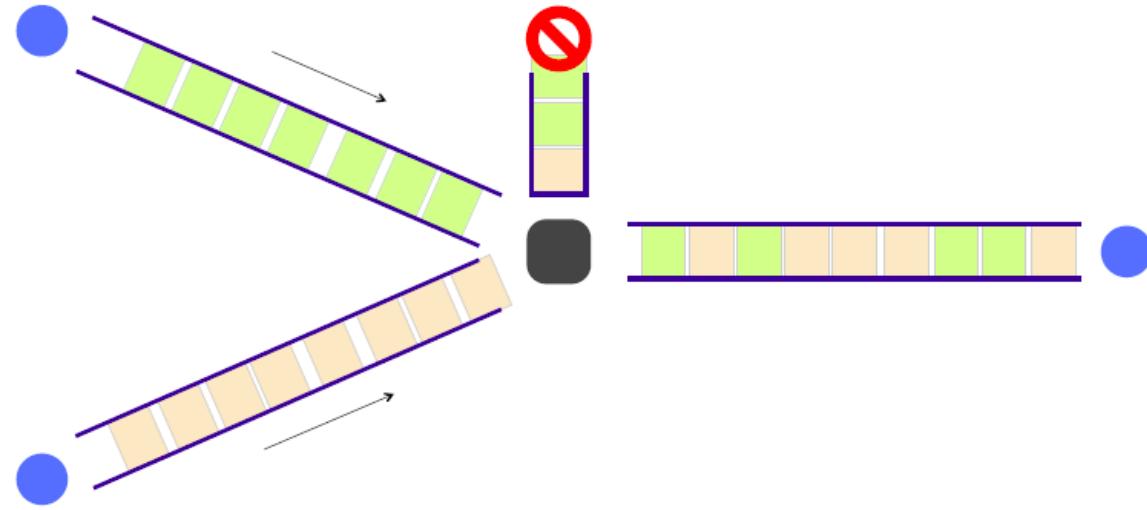


In practice, queues are not infinite.
There is an upper bound on queuing delay.



queuing delay upper bound: $N*L/R$

If the queue is persistently overloaded,
it will eventually drop packets (loss)



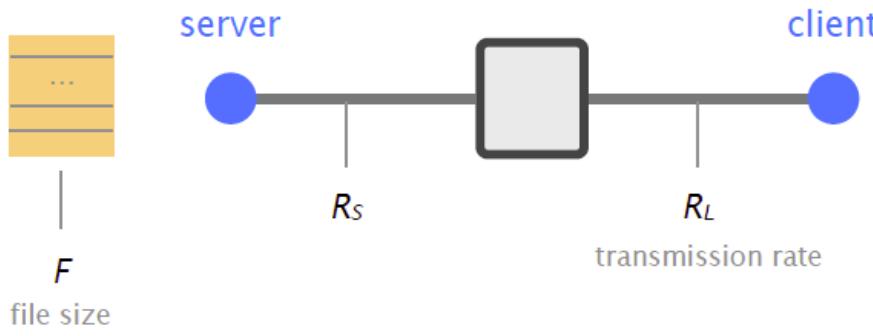
Throughput

throughput

The throughput is the instantaneous rate at which a host receives data

$$\text{Average throughput} = \frac{\text{data size} \quad [\#\text{bits}]}{\text{transfer time} \quad [\text{sec}]}$$

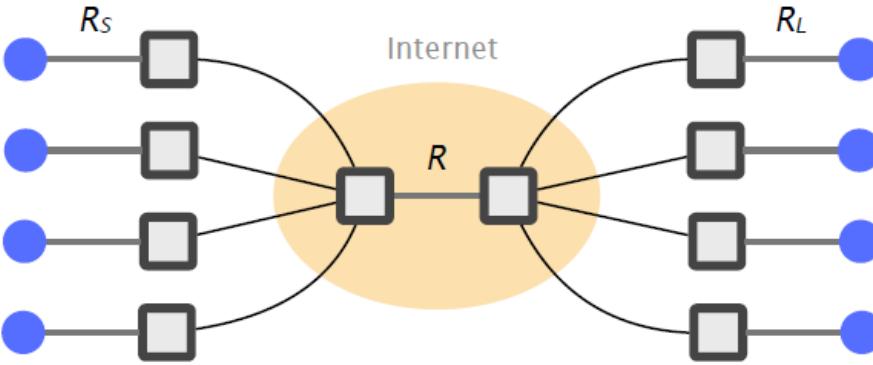
To compute throughput, one has to consider the bottleneck link



Average throughput

$\min(R_S, R_L)$
= transmission rate
of the bottleneck link

To compute throughput, one has to consider the bottleneck link... **and the intervening traffic**



if $4 * \min(R_S, R_L) > R$ the bottleneck is now in the core,
providing each download $R/4$ of throughput

As technology improves, throughput increase & delays are getting lower except for propagation
(speed of light)

Because of propagation delays,
Content Delivery Networks move content closer to you

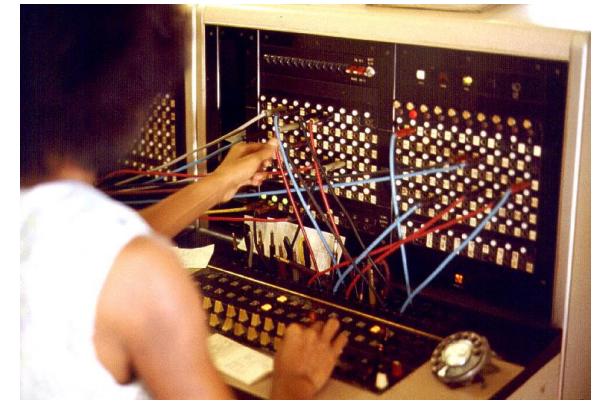


* <http://wwwnui.akamai.com/gnet/globe/index.html>

Brief history of Internet

The Internet history starts in the late 50's...

Phone networks = the communication network
fully circuit-switched



People wanted to use networks for other things
defense, computers, etc.

circuit-switching does not fit to these new requirements...
inefficient for bursty loads and not resilient

Three main questions



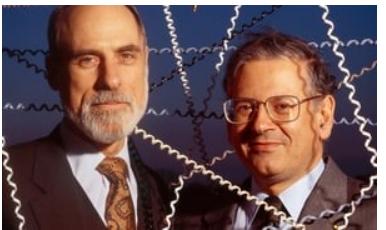
Paul Baran RAND

How can we design a more **resilient** network?
... led to the invention of packet switching



Leonard Kleinrock
UCLA

How can we design a more **efficient** network?
... also led to the invention of packet switching

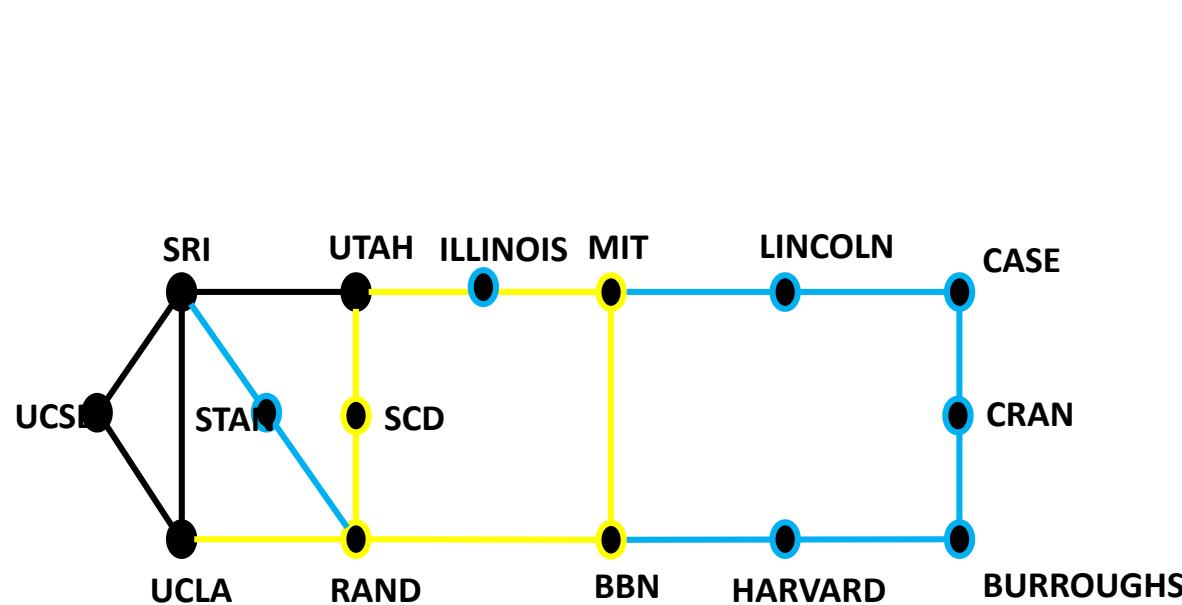


Vint Cerf & Bob Kahn
DARPA

How can we **connect** all these networks **together**?
... the invention of Internet as we know it

The 60's was all about packet switching...

Advanced Research Projects Agency NETwork (ARPANET)



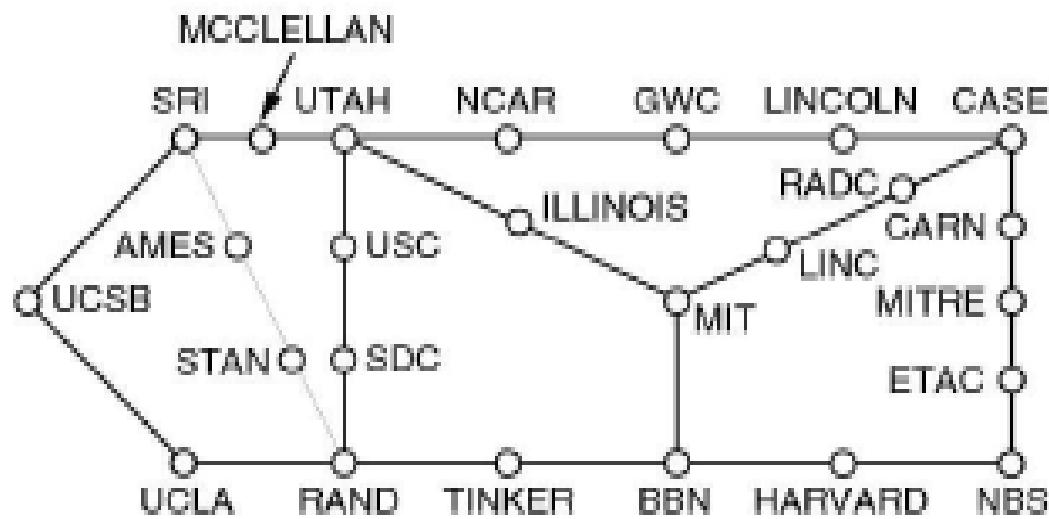
1969 december

1970 july

1971 march

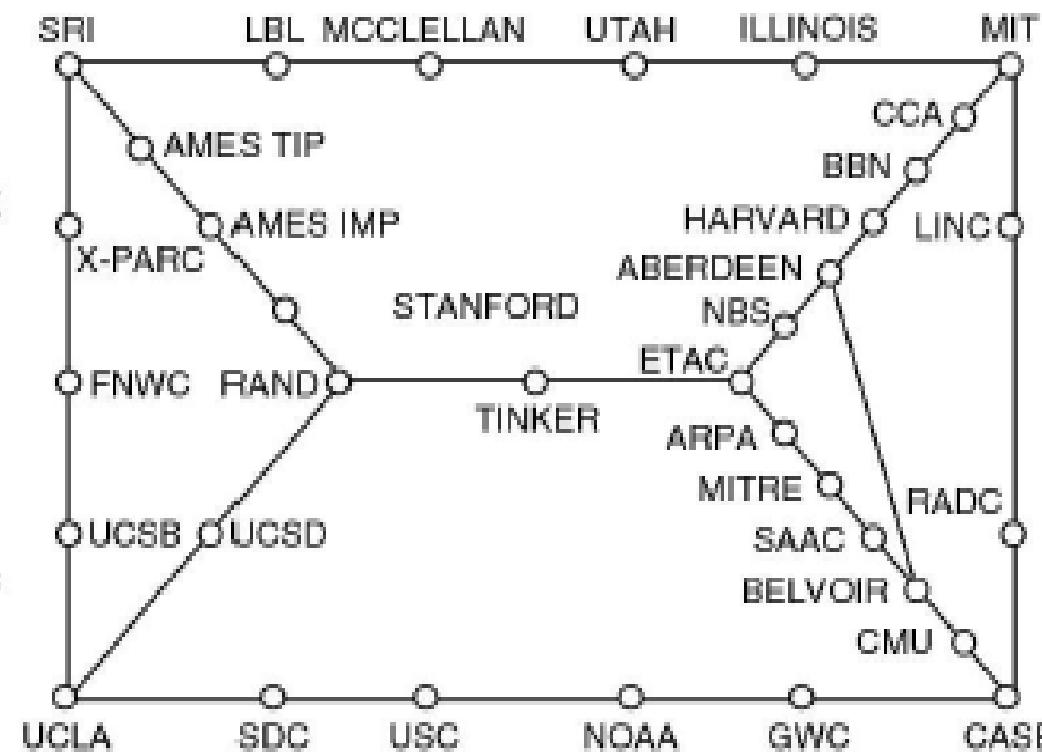
ARPANET

April 1972



ARPANET

September 1972



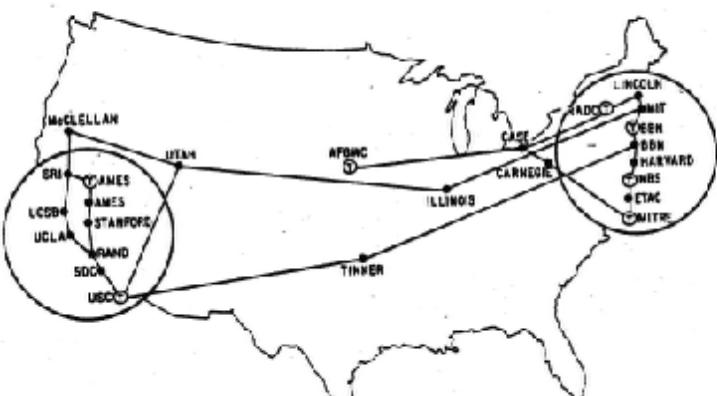
ARPANET



Dezember 1969



Juni 1970



März 1972



Juli 1977

The first message over the Internet : „LO”

29. Oct. 1969

Leonard Kleinrock from UCLA tries to log in
a Stanford computer

UCLA We typed the L... Do you see it?

Yes! We see the L Stanford

We typed the O... Do you see it?

Yes! We see the O

We typed the G.

... and the system crashed ...

The 70's about Ethernet, TCP/IP and email...

1971

Network Control Program (NCP)

Predecessor of TCP/IP

1972

Email and Telnet

1973

Ethernet

1974

TCP/IP

Paper of Vint Cerf and Bob Kahn

80's when TCP/IP went mainstream

1983

NCP to TCP/IP

Domain Name Service (DNS)

1985

NSFNet (TCP/IP)

198x

First Internet crashes caused by congestion

1986

Van Jacobson saves the Internet

congestion control



Van Jacobson

90's – the Internet going commercial...

1989

ARPANET closed

Birth of the WEB

Tim Berners Lee (CERN)



1993

First search engine (Excite)

1995

NSFNet closed

1998

Google reinvents searching

The new millennium bringing Web 2.0

1998

IPv6 standardization

2004

Facebook goes online

2006

Google buys YouTube

2007

Netflix strats streaming videos

2007

First iPhone with mobile Internet access

Fast Internet access everywhere, every device needs an Internet connection

2009

Mining of the Bitcoin genesis block



Fast mobile Internet access: 4G/LTE

IoT boom

Internet of Everything

2018

Only 26% of Alexa Top 1000 sites reachable over IPv6

Soon?

Encrypted transport protocols

For example QUIC

Ultra-fast mobile access – 5G

To be continued...

Számítógépes Hálózatok

3. Előadás: Fizikai réteg

Based on slides from **Zoltán Ács ELTE** and D. Choffnes Northeastern U., Philippa Gill from StonyBrook University , Revised Spring 2016 by S. Laki

Fizikai réteg

2



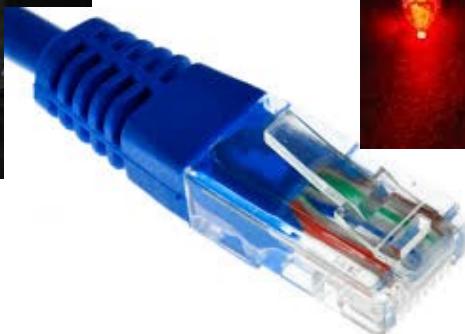
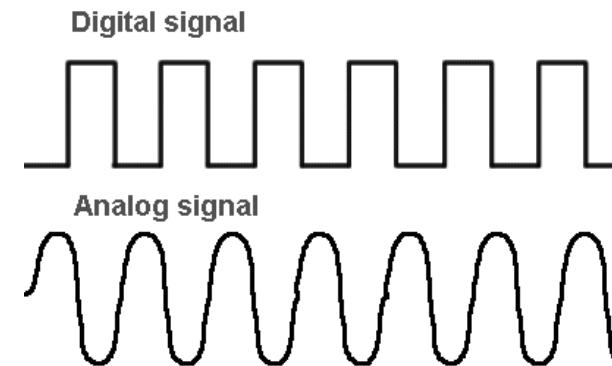
- Szolgáltatás
 - Információt visz át két fizikailag összekötött eszköz között
 - definiálja az eszköz és a fizikai átviteli közeg kapcsolatát
- Interfész
 - Specifikálja egy bit átvitelét
- Protokoll
 - Egy bit kódolásának sémája
 - Feszültség szintek
 - Jelek időzítése
- Példák: koaksiális kábel, optikai kábel, rádió frekvenciás adó

Alapfogalmak

Kihívások

4

- Digitális számítógépek
 - Nullák és egyesek
- Analóg világ
 - Amplitúdók és frekvenciák



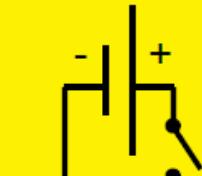
Egyszerű adatátvitel

- 1-es bit: feszültség vagy áramerősség
- 0-ás bit: nincs feszültség

Converting bits to voltage

Bit 1: The switch is turned on.

Bit 0: It is turned off.



Converting voltage to bits

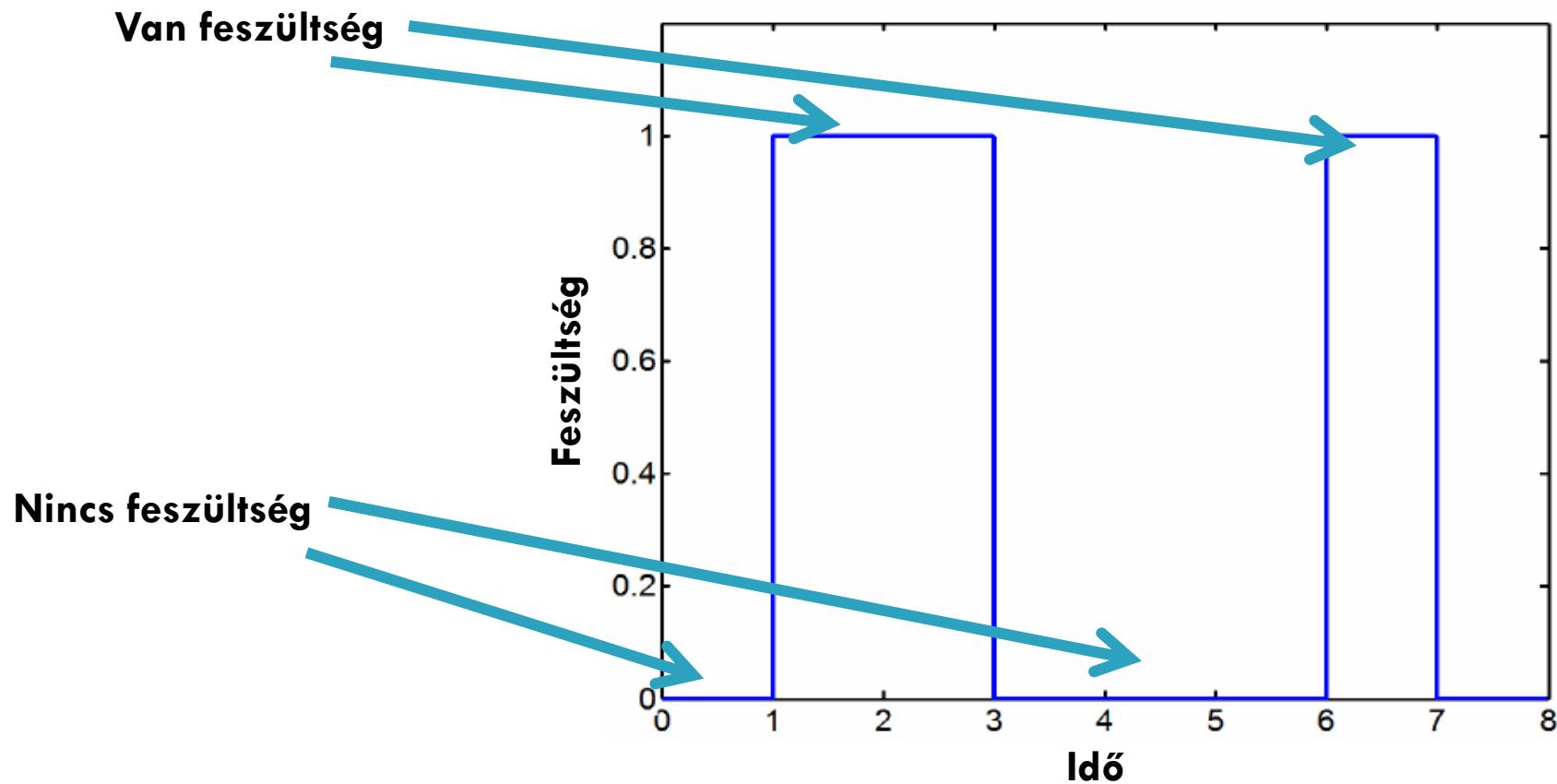
Voltage: Bit 1

No voltage: Bit 0



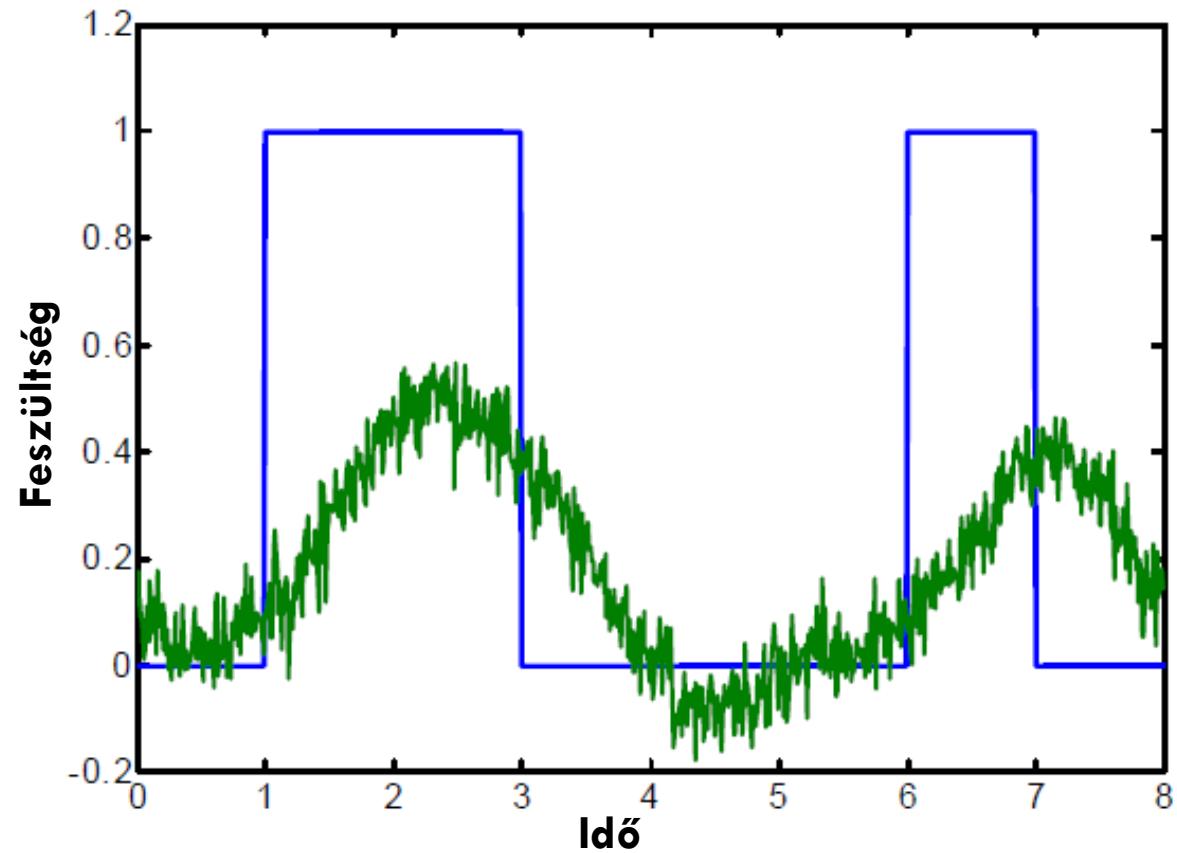
A „b” karakter átvitele

- ❑ Egynél több bit szükséges a „b” karakter átviteléhez
- ❑ A „b” ASCII kódja bináris formában: 01100010



A „b” karakter átvitele

Túl rossz vétel



Elméleti alapok – adatátvitel

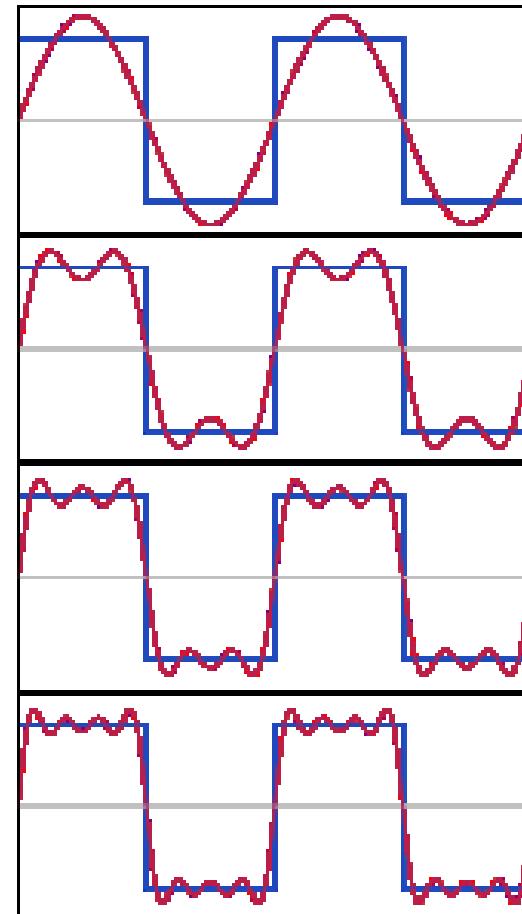
8

Adatátvitel vezeték esetén valamilyen fizikai jellemző változtatásával lehetséges (pl.: feszültség, áramerősség)

- a viselkedést $f(t)$ függvénytel jellemzhetjük
- Bármely T periódusidejű $g(t)$ periodikus függvény előáll a következő alakban:

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

ahol $f = \frac{1}{T}$ az alapfrekvencia, a_n és b_n pedig az n -edik harmonikus szinuszos illetve koszinuszos amplitúdók.



Elméleti alapok – adatátvitel

9

$$a_n = \frac{2}{T} \int_0^T g(t) \sin(2\pi n f t) dt$$

$$b_n = \frac{2}{T} \int_0^T g(t) \cos(2\pi n f t) dt$$

$$c = \frac{2}{T} \int_0^T g(t) dt$$

Elméleti alapok – adatátvitel

10

□ Példa

- Tegyük fel, hogy az ASCII „b” karaktert küldjük, amely 8 biten ábrázolható, azaz a bitminta 01100010 .
- A jel Fourier-sora az alábbi együtthatókat tartalmazza:

$$a_n = \frac{1}{\pi n} \left[\cos\left(\pi \frac{n}{4}\right) - \cos\left(3\pi \frac{n}{4}\right) + \cos\left(6\pi \frac{n}{4}\right) - \cos\left(7\pi \frac{n}{4}\right) \right]$$

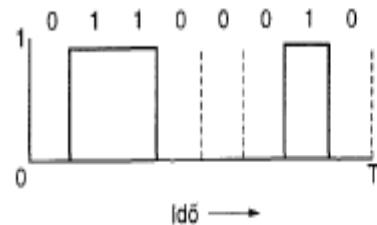
$$b_n = \frac{1}{\pi n} \left[\sin\left(3\pi \frac{n}{4}\right) - \sin\left(\pi \frac{n}{4}\right) + \sin\left(7\pi \frac{n}{4}\right) - \sin\left(6\pi \frac{n}{4}\right) \right]$$

$$c = \frac{3}{4}$$

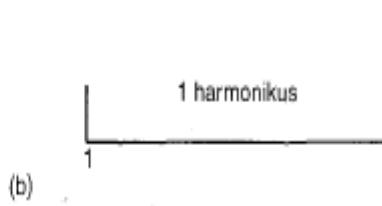
- A harmonikus amplitúdók négyzetösszege arányos a frekvencián továbbított energiával
- (energiaveszteség lehetséges)

Elméleti alapok – adatátvitel

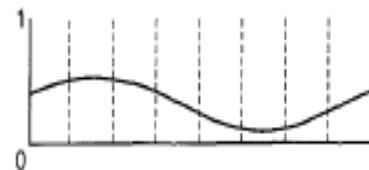
11



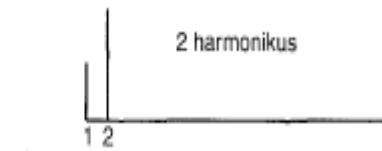
(a)



(b)

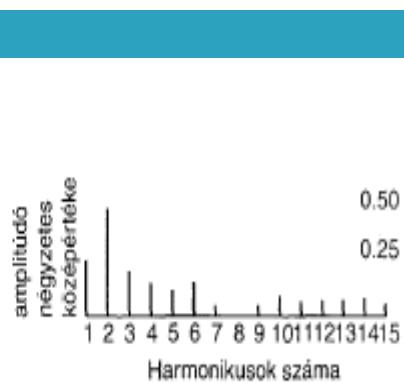


(c)



(d)

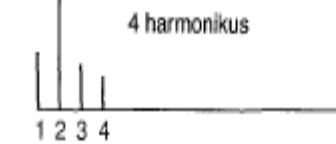
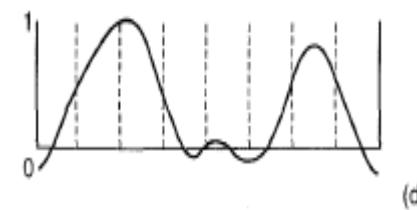
(Tanenbaum)



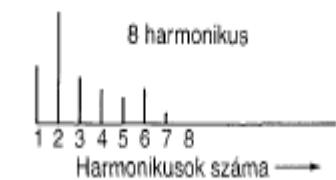
(e)

(e)

(Tanenbaum)



(d)



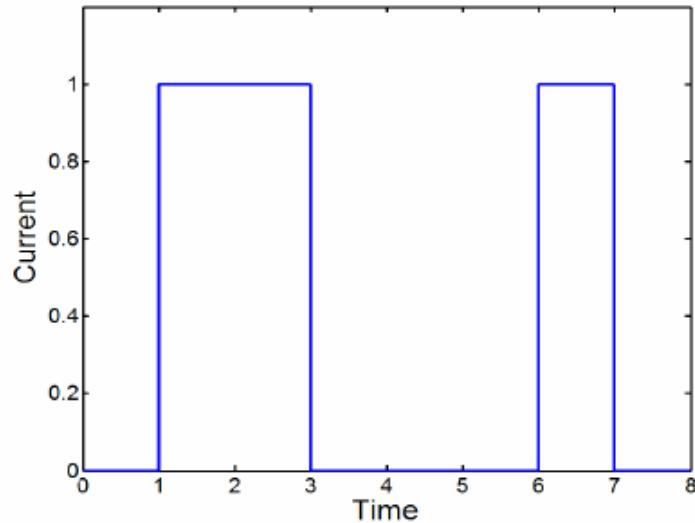
(e)

(Tanenbaum)

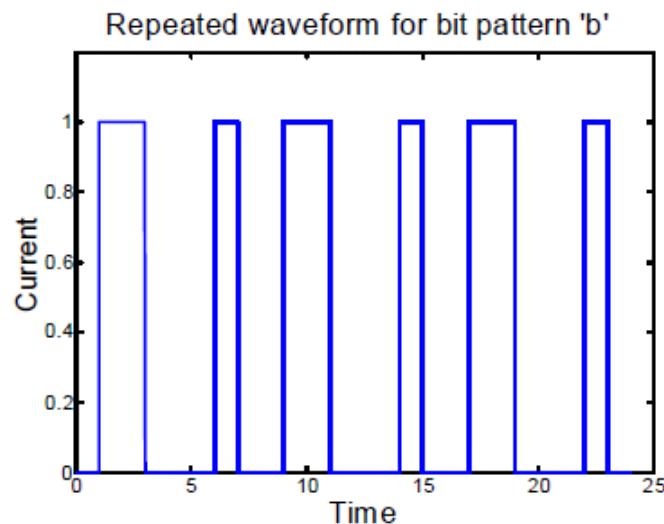
Fourier sor felhasználása

12

- A digitális szignál nem periodikus
 - Pl. „b” ASCII kódja 8 bit hosszú



- ...de elképzelhetjük, hogy végtelen sokszor ismétlődik, ami egy periodikus függvényt ad
 - Pl. „b” esetén a periódus 8 bit hosszú



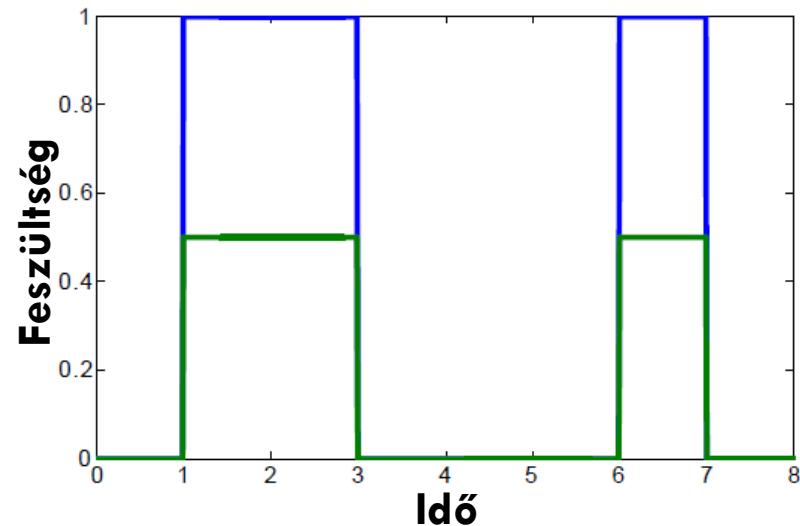
Elméleti alapok - Elnyelődés

13

- Elnyelődés (attenuation): α
 - Lényegében a küldési (P_0) és vételi (P_1) energiák hányadosa
 - Nagy elnyelődés esetén kevés energia éri el a fogadót
 - A jel helyreállítása lehetetlen
 - Mértékegysége decibel

$$\alpha [in \text{ } dB] = 10 \times \log_{10} \frac{P_0}{P_1} \text{ (decibel [dB])}$$

- Az elnyelőést befolyásoló tényezők
 - Átviteli közeg
 - Adó és vevő távolsága
 - ...



Elméleti alapok - Elnyelődés

14

□ Valódi közegben

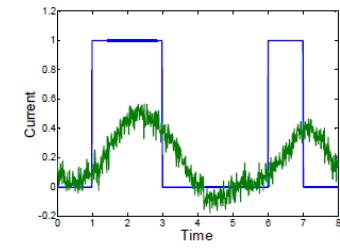
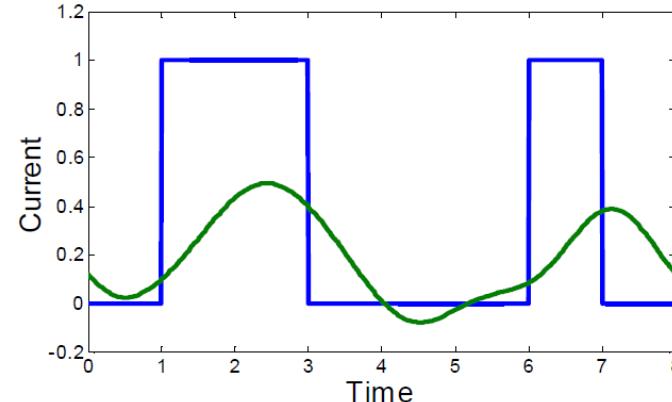
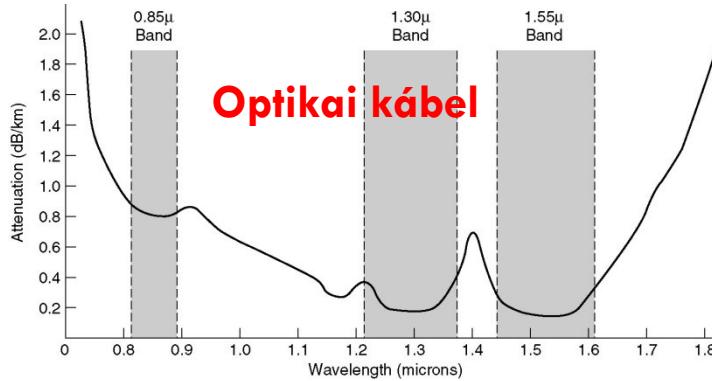
□ Frekvenciafüggő elnyelődés

□ Fáziseltolódás

- Különböző frekvenciáknak különböző a terjedési sebessége
- Frekvenciafüggő torzítás

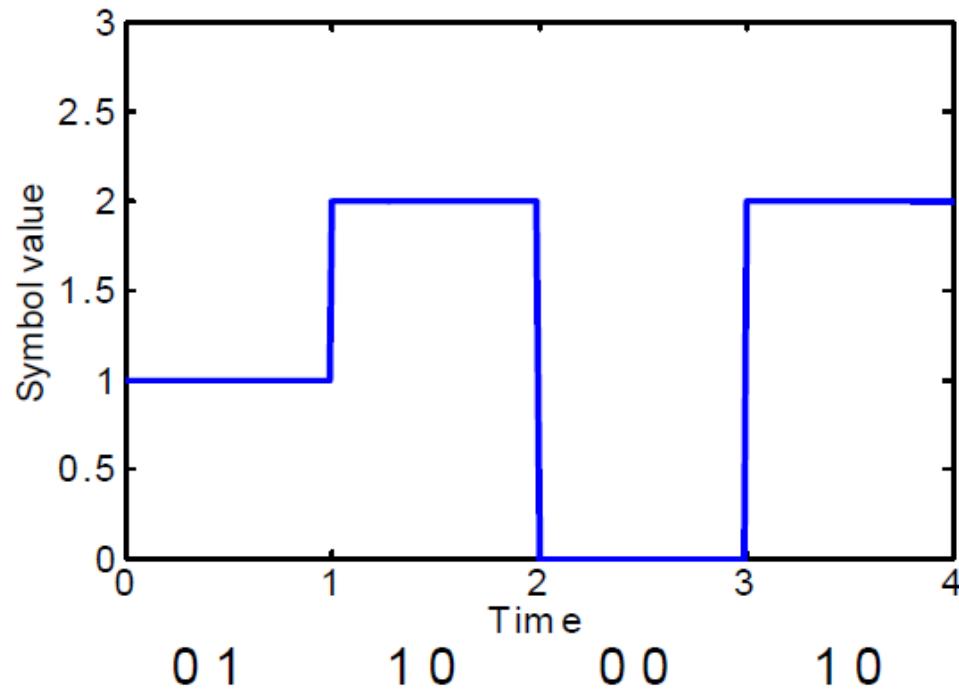
□ Zaj

- Hő, más rendszerek zavarása...



Szimbólumok és bitek

- Bitek helyett szimbólumok használata az átvitelhez
- Példa:
 - Vezessünk be 4 szimbólumot:
A(00),B(01),C(10),D(11)
 - Szimbólum ráta: (BAUD)
 - Elküldött szimbólumok száma másodpercenként
 - Adat ráta (bps):
 - Elküldött bitek száma másodpercenként

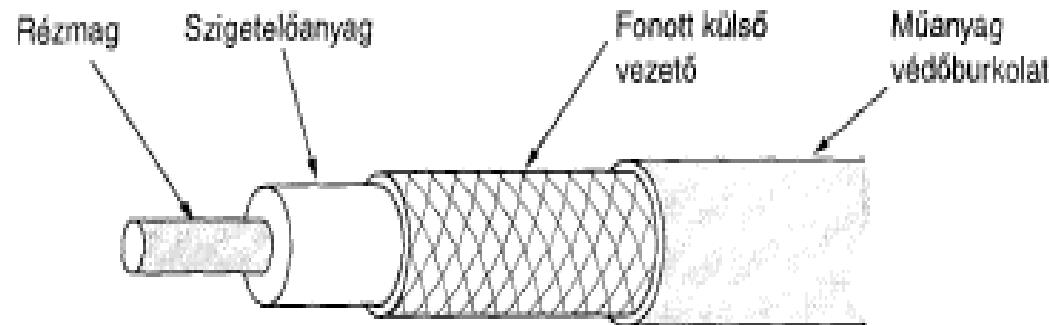


Példa:
Egy 600 Baudos modemmel, ami 16 szimbólumot különböztet meg 2400 bps adatráta érhető el.

Átviteli közegek – vezetékes 1 / 3

17

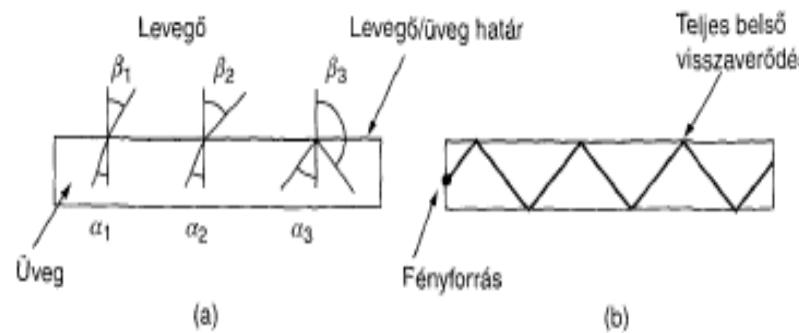
- **mágneses adathordozók** – sávszélesség jó, késleltetés nagy (nem on-line kapcsolat)
- **Sodort érpár** (angolul „twisted pair”) – főként távbeszélőrendszerekben használatos; dupla rézhuzal; analóg és digitális jelátvitel; UTP és STP
- **Koaxális kábel** – nagyobb sebesség és távolság érhető el, mint a sodorttal; analóg (75Ω) és digitális (50Ω) jelátvitel



Átviteli közegek – vezetékes 2/3

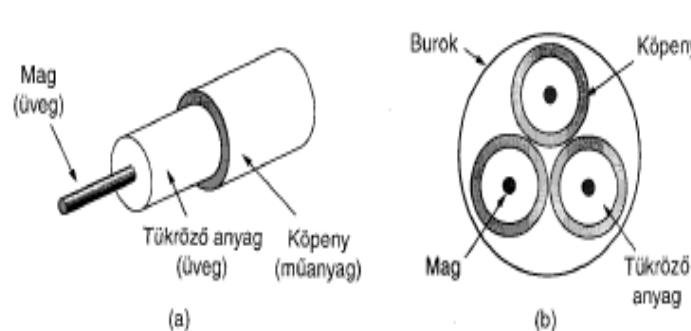
18

- **Fényvezető szálak** – részei: fényforrás, átviteli közeg és detektor; fényimpulzus 1-es bit, nincs fényimpulzus 0-s bit; sugaraknak más-más módusa van (határszög \leq beeső sugár szöge)



(Tanenbaum)

- **Fénykábelek felépítése:**



Átviteli közegek – vezetékes 3/3

19

- Fénykábelek összehetődése fényimpulzus típusa alapján

Jellemző	LED	Félvezető lézer
Adatátviteli sebesség	Alacsony	Magas
Módus	Többmóodusú	Több- vagy egymóodusú
Távolság	Kicsi	Nagy
Élettartam	Hosszú	Rövid
Hőmérsékletérzékenység	Kicsi	Jelentős
Ár	Olcso	Drága

Elméleti alapok – vezeték nélküli adatátvitel

20

- **Frekvencia:** elektromágneses hullám másodpercenkénti rezgésszáma.
 - Jelölés: f
 - Mértékegység: Hertz (Hz)
- **Hullámhossz:** két egymást követő hullámcsúcs (vagy hullámvölgy) közötti távolság
 - Jelölés: λ
- **Fénysebesség:** az elektromágneses hullámok terjedési sebessége vákuumban
 - Jelölés: c
 - Értéke: kb. $3 * 10^8 \frac{\text{m}}{\text{s}}$
 - Rézben és üvegszálban ez a sebesség nagyjából a 2/3-adára csökken
- Összefüggés a fenti mennyiségek között: $\lambda f = c$

Elméleti alapok – elektromágneses spektrum

21

Tartomány neve	Hullámhossz (centiméter)	Frekvencia (Hertz)
Rádió	>10	$< 3 * 10^9$
Mikrohullám	10 - 0.01	$3 * 10^9 - 3 * 10^{12}$
Infravörös	$0.01 - 7 \times 10^{-5}$	$3 \times 10^{12} - 4.3 \times 10^{14}$
Látható	$7 \times 10^{-5} - 4 \times 10^{-5}$	$4.3 * 10^{14} - 7.5 * 10^{14}$
Ultraibolya	$4 \times 10^{-5} - 10^{-7}$	$7.5 * 10^{14} - 3 * 10^{17}$
Röntgen sugarak	$10^{-7} - 10^{-9}$	$3 * 10^{17} - 3 * 10^{19}$
Gamma sugarak	$< 10^{-9}$	$> 3 * 10^{19}$

Elméleti alapok – elektromágneses spektrum

22

Például:
órakelek

Például:
tengerés
zeti
mobil,
rádió

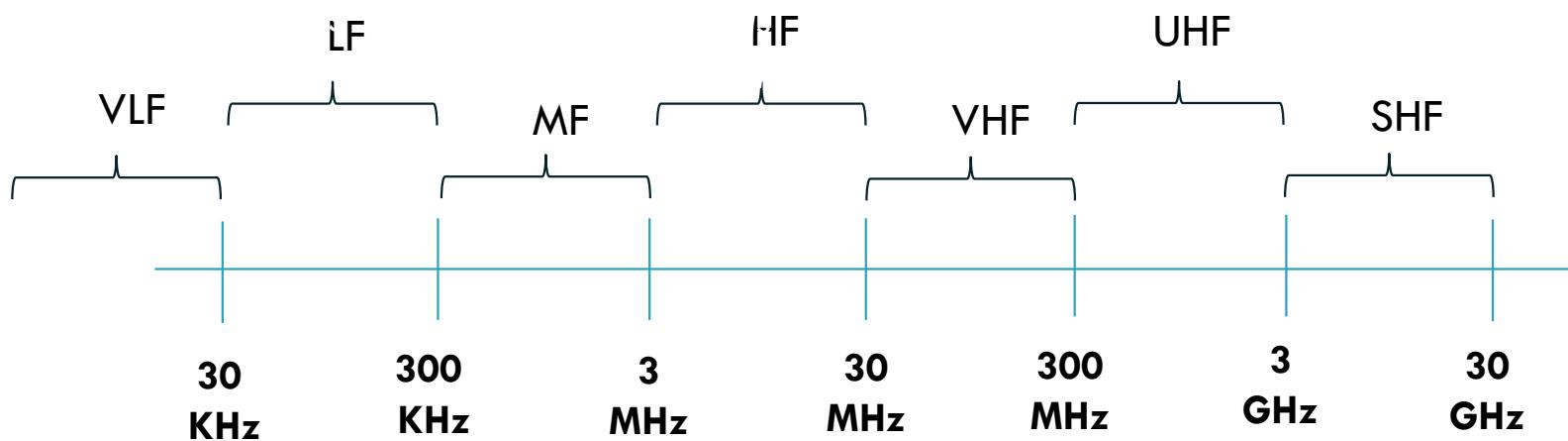
Például:
szárazfö
ldi
mobil,
rádió

Például:
légforga
lmi
mobil,
rádió

Például:
televízió,
rádió
navigáci
ó

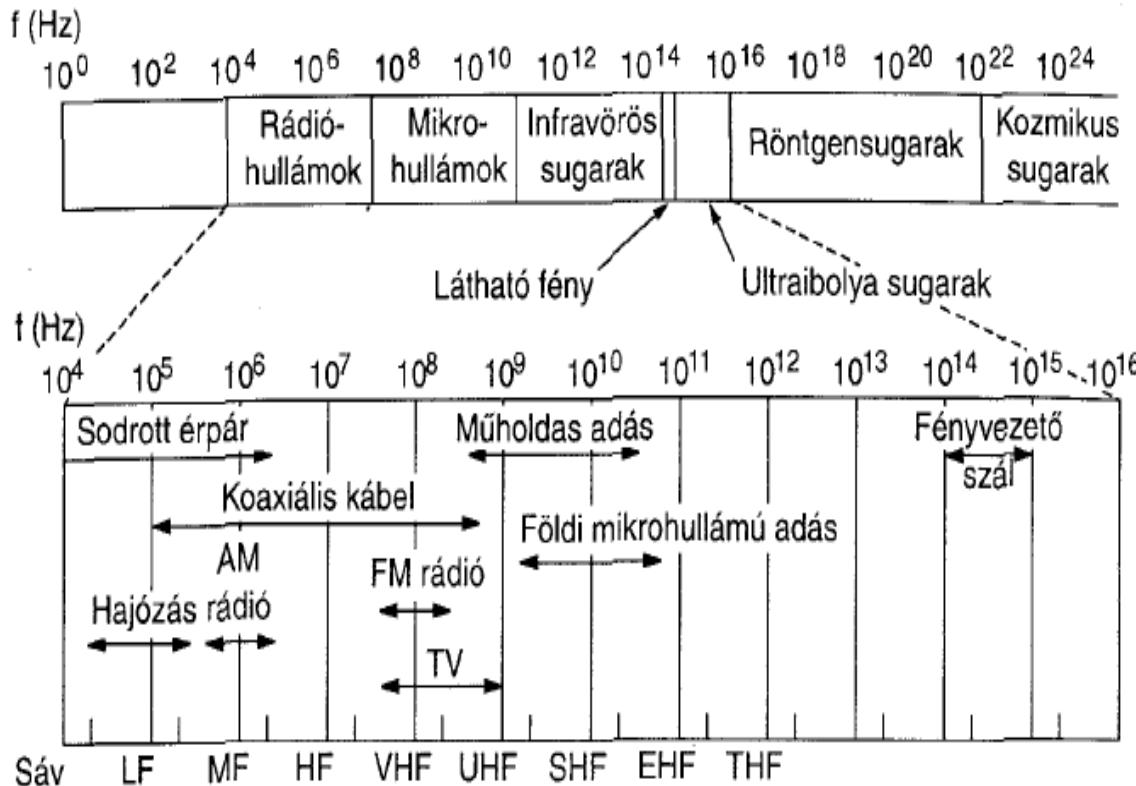
Például:
televíziós
adatszór
ás,
navigáci

Például:
szatellit
kommuni
káció



Elméleti alapok – elektromágneses spektrum

23

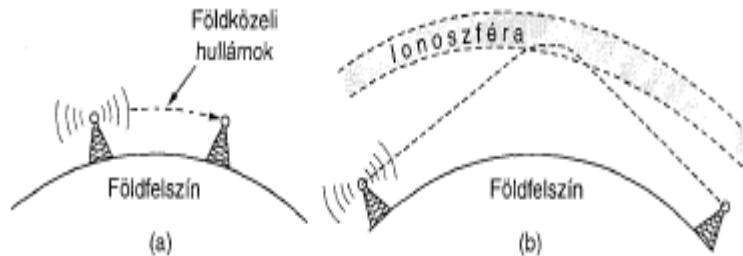


[Forrás: Tanenbaum]

Átviteli közegek – vezeték nélküli

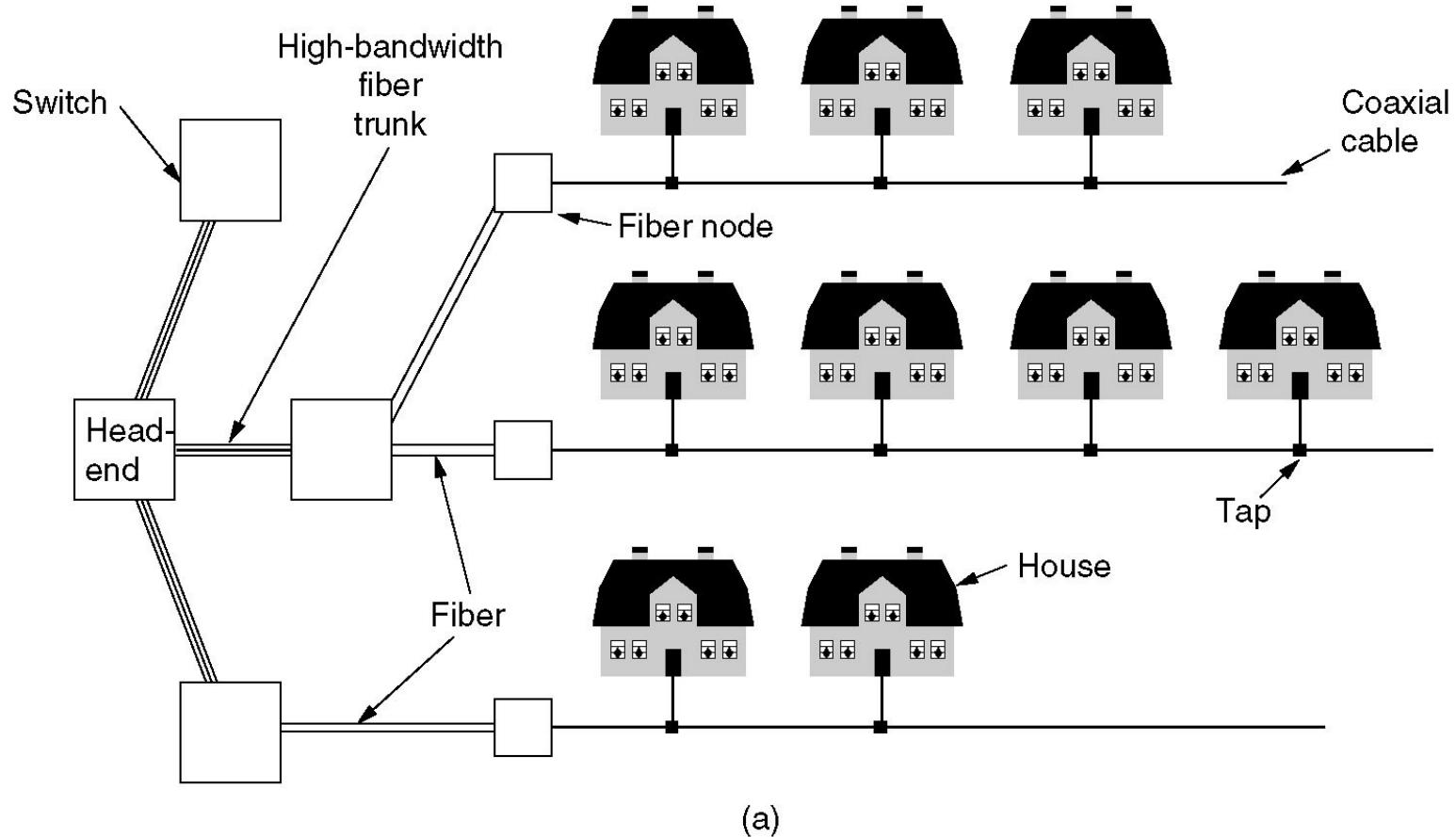
24

- **Rádiófrekvenciás átvitel** – egyszerűen előállíthatóak; nagy távolság; kültéri és beltéri alkalmazhatóság; frekvenciafüggő terjedési jellemzők

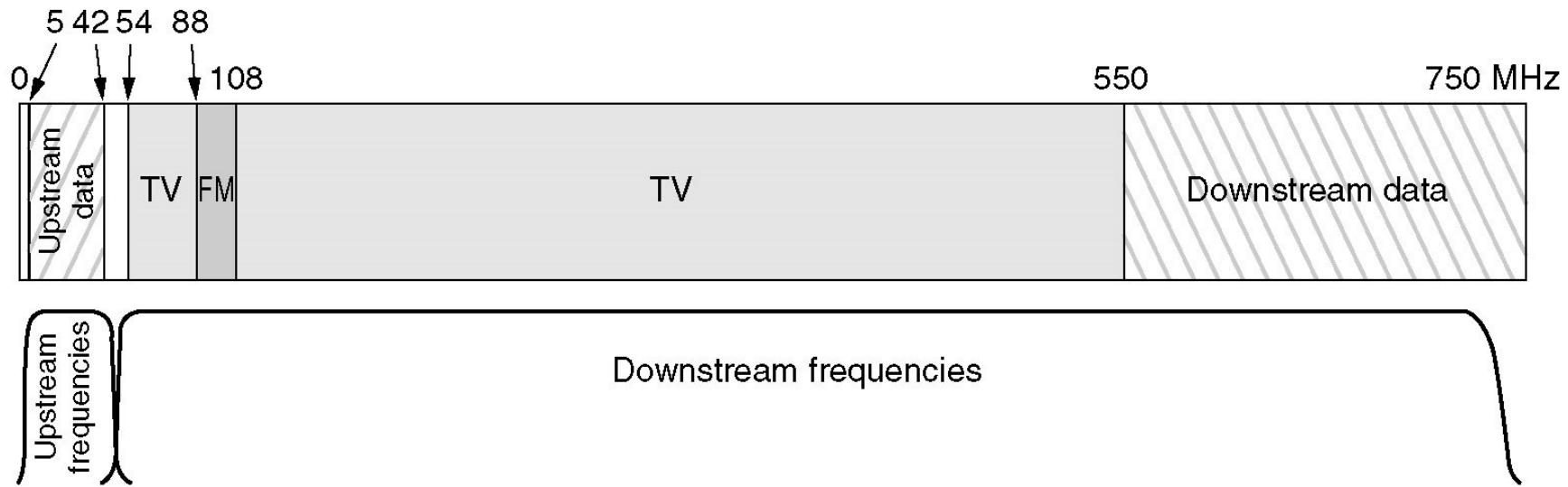


- **Mikrohullámú átvitel** – egyenes vonal mentén terjed; elhalkulás problémája; nem drága
- **Infravörös és milliméteres hullámú átvitel** – kistávolságú átvitel esetén; szilárd tárgyakon nem hatol át
- **Látható fényhullámú átvitel** – lézerforrás + fényérzékelő; nagy sávszélesség, olcsó, nem engedélyköteles; időjárás erősen befolyásolhatja;

Internet a kábel TV hálózaton



Internet a kábel TV hálózaton



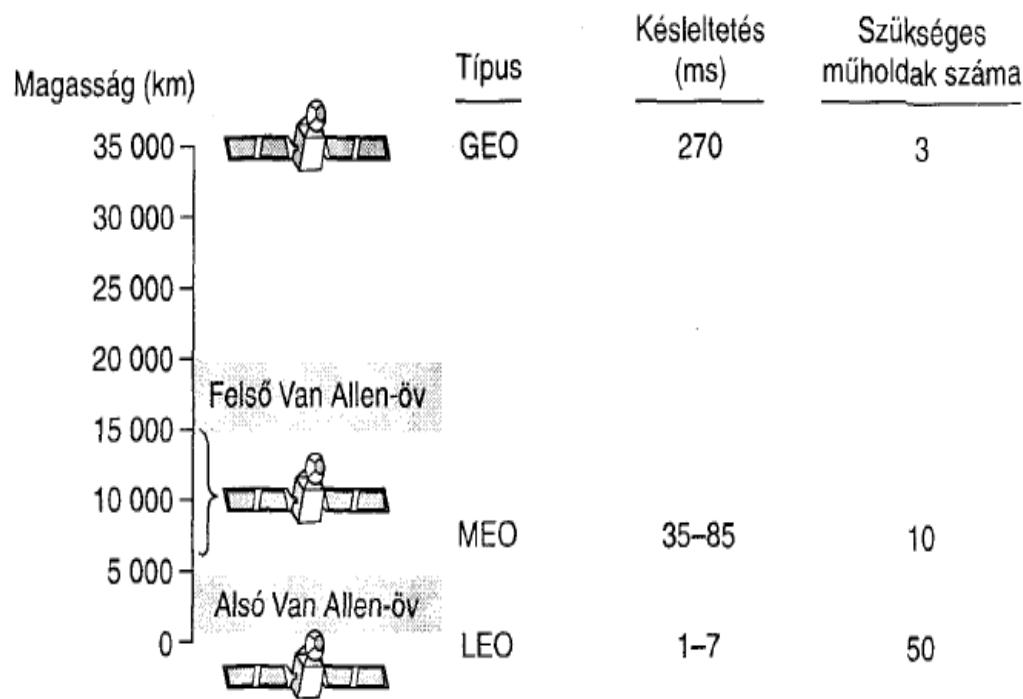
Frekvencia kiosztás egy tipikus kábel TV alapú
Internet elérés esetén

Átviteli közegek – kommunikáció műholdak

27

JELLEMZŐK

- Transzpondereket tartalmaz a spektrum részek figyelésére
- Jeleket felerősíti és továbbítja egy másik frekvencián
 - széles területen vagy
 - keskeny területen
- Magassággal nő a keringé idő is.



[Forrás: Tanenbaum]

Átviteli közegek – kommunikáció műholdak

28

FAJTÁI

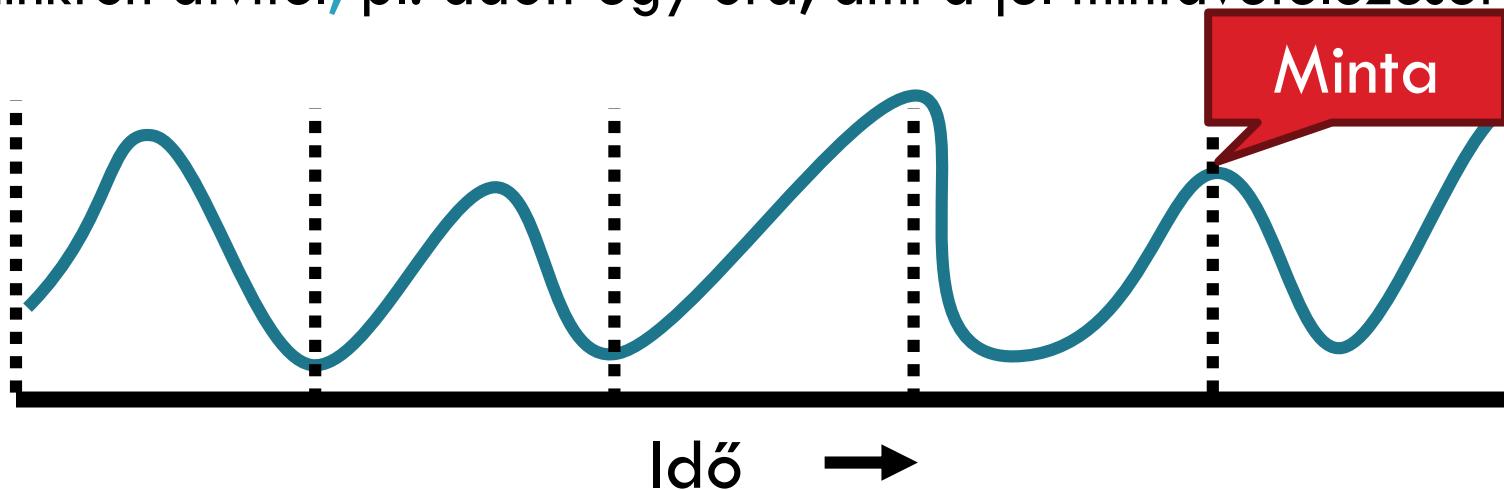
- **Geoszinkron műholdak** – 270 milliszekundum késleltetés, 3 műhold szükséges a föld lefedésére, 35800 kilométeres magasságban keringenek
- **Közepes röppályás műholdak** – 35-85 milliszekundum késleltetés, 10 műhold szükséges a föld lefedésére, a két Van Allen-öv közötti magasságban keringenek
- **Alacsony röppályás műholdak** – 1-7 milliszekundum késleltetés, 50 műhold szükséges a föld lefedésére, az alsó Van Allen-öv alatti tartományban keringenek

Adatátvitel

Kiinduló feltételek

30

- Két diszkrét jelünk van, ahol magas érték kódolja az 1-et és alacsony a 0-át.
- Szinkron átvitel, pl. adott egy óra, ami a jel mintavételezését vezéri



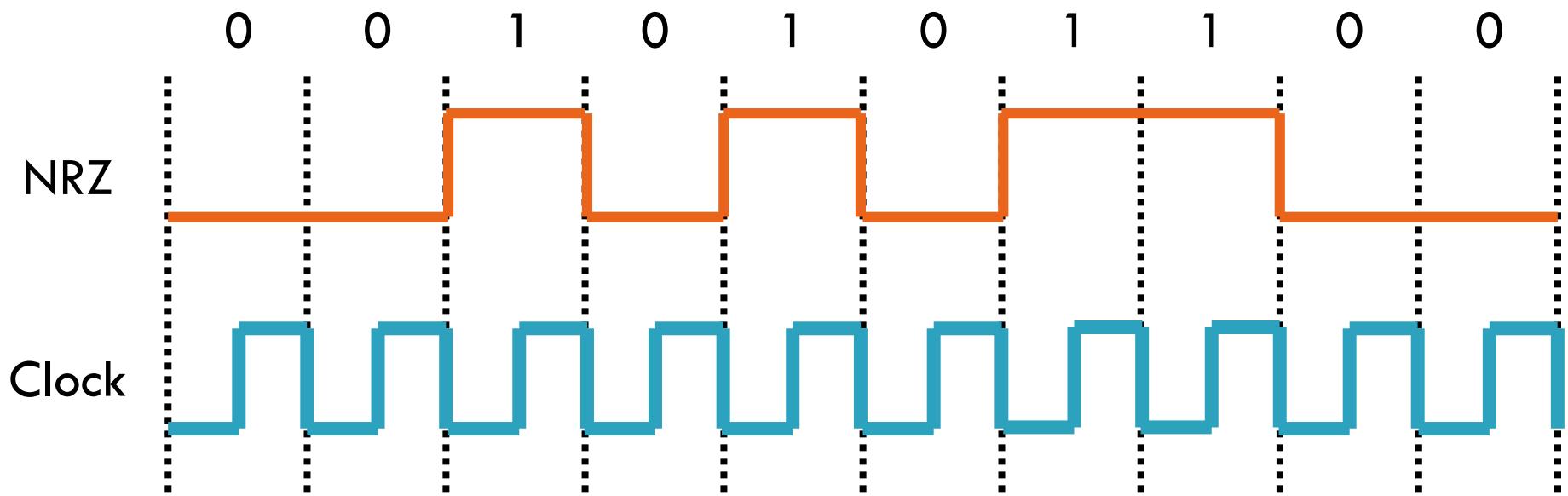
- A jel amplitúdója és az időbeli kiterjedése a fontos



Non-Return to Zero (NRZ) kódolás

31

- 1 → magas jel, 0 → alacsony jel



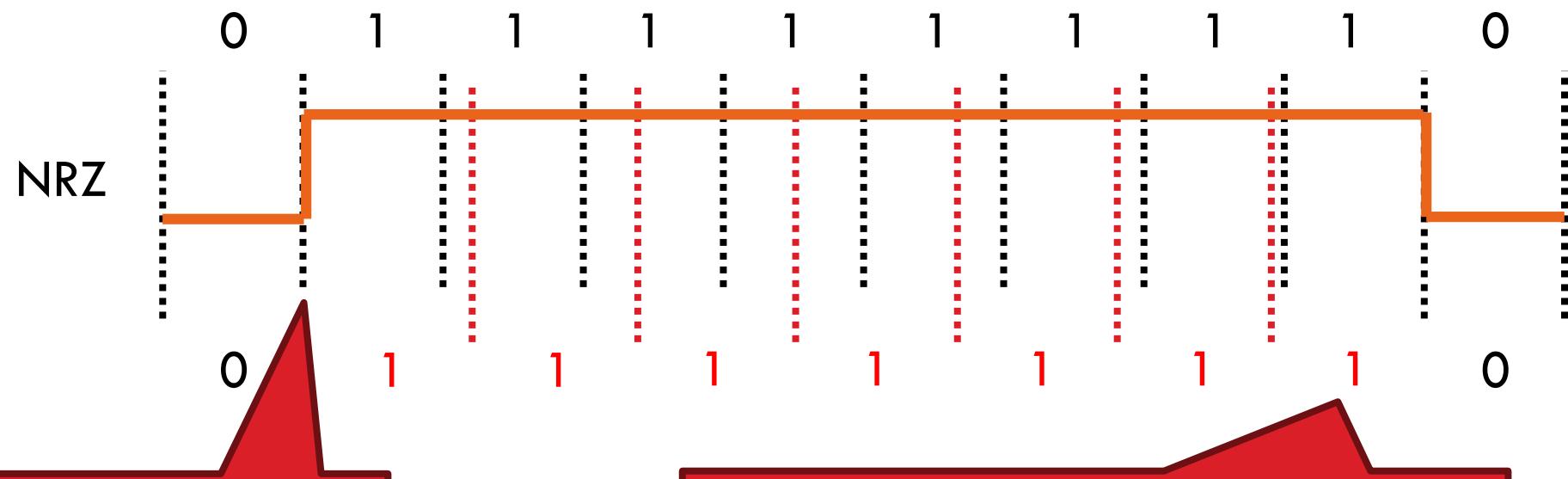
- Probléma: 0-ákból vagy 1-esekből álló hosszú sorozatok a szinkronizáció megszűnéséhez vezetnek
 - Hogyan különböztessünk meg sok nullát attól az állapottól, amikor nincs jel?
 - Hogyan hozzuk szinkronba az órákat egy hosszú egyeseket tartalmazó sorozat után?

Szinkronizáció megszűnése

(„deszinkronizáció”)

32

- Probléma: mikén állítsuk vissza az órát hosszú egyes vagy nullás sorozat után:



Az átmenetek
jelzik az óra
ütemét

A fogadó kihagy egy egyes
bitet az órák elcsúszása
miatt!!!

Szinkronizációs megoldás

33

- Felügyelet szükséges a szinkron működéshez
 1. Explicit órajel
 - párhuzamos átviteli csatornák használata,
 - szinkronizált adatok,
 - rövid átvitel esetén alkalmas.
 2. 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
 3. Szimbólum kódok
 - öönütemező jel – külön órajel szinkronizáció nélkül dekódolható jel,
 - a szignál tartalmazza a szinkronizáláshoz szükséges információt.

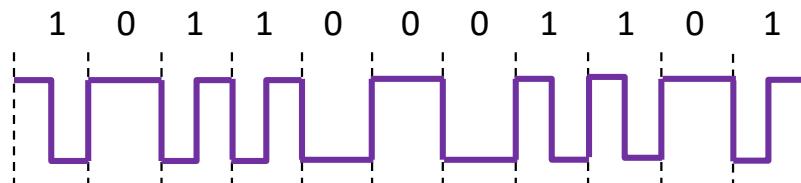
Digitális kódok 1/3

34

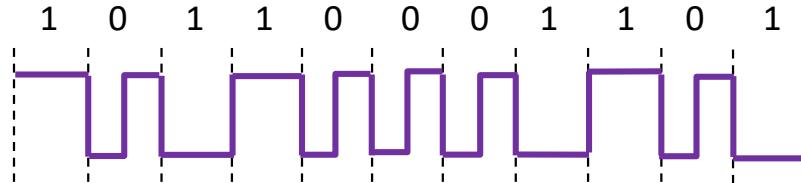
- A digitális kódok 3 lényeges momentumban térnek el:
 - i. Mi történik egy szignál intervallum elején?
 - ii. Mi történik egy szignál intervallum közepén?
 - iii. Mi történik egy szignál intervallum végén?

Néhány konkrét digitális kód

- *Biphase-Mark* (váltás, 1-es bit esetén váltás, semmi)



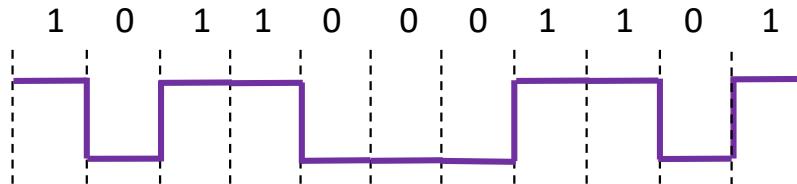
- *Biphase-Space* (váltás, 0-ás bit esetén váltás, semmi)



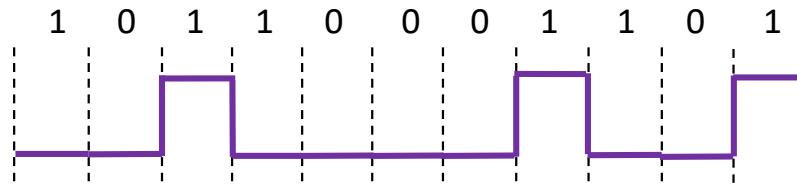
Digitális kódok 2/3

35

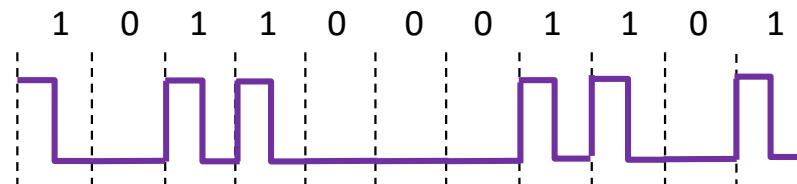
- NRZ-L (1-es bit magas jelszint/ 0-s bit alacsony jelszint, semmi, semmi)



- NRZ-M (1-es bit jelszint váltás/ 0-ás bit esetén nincs váltás, semmi, semmi)



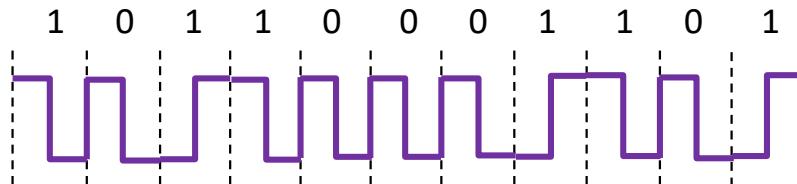
- RZ (1-es bit magas jelszint/ 0-s bit alacsony jelszint, 1-es bit esetén váltás, semmi)



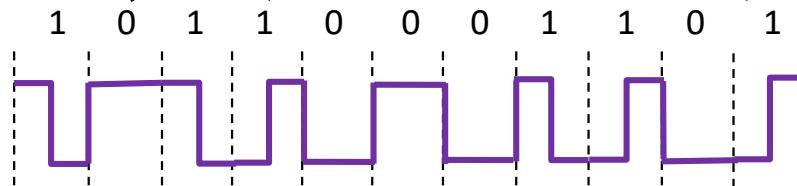
Digitális kódok 3/3

36

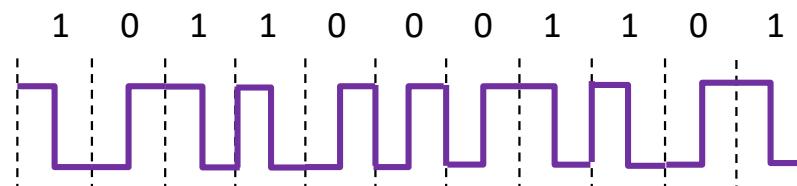
- Differential Manchester (0-s bit esetén váltás, váltás, semmi)



- Delay-Modulation (semmi, 1-es bit esetén váltás, 0-s bit következik váltás)



- Manchester (semmi, 1-es bit magasról alacsonyra/ 0-s alacsonyról magasra, semmi)

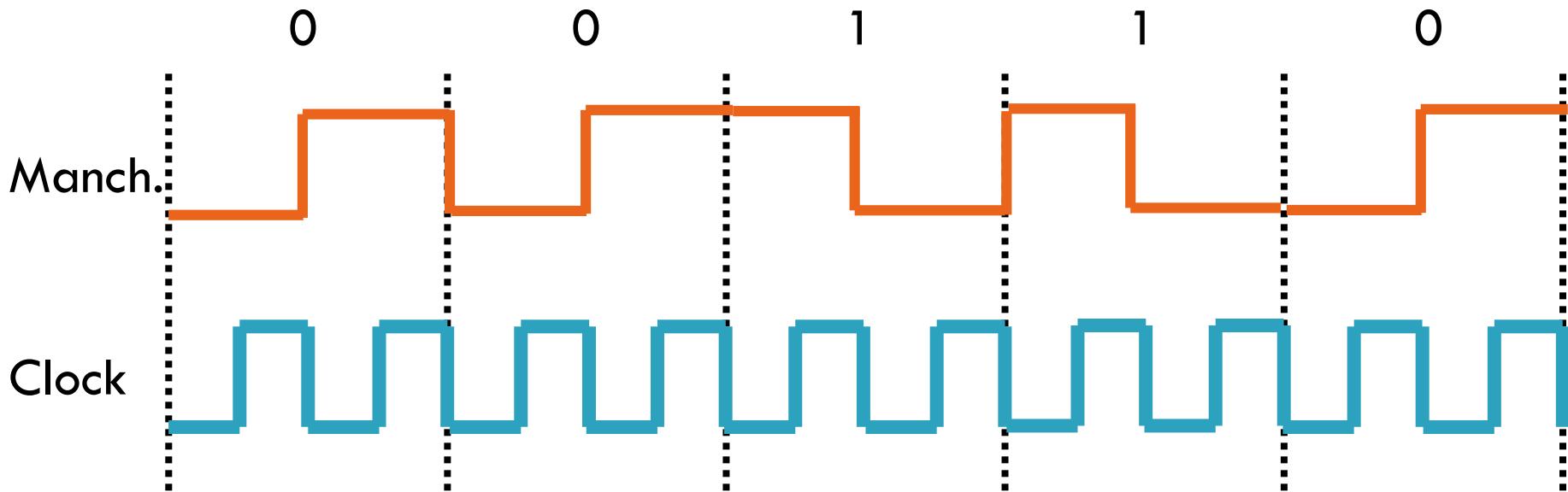


Ethernet példa:
10BASE-TX
100BASE-TX

Manchester (10 Mbps Ethernet 10BASE-TX)

38

- 1 → átmenet magasról alacsonyra, 0 → alacsonyról magasra

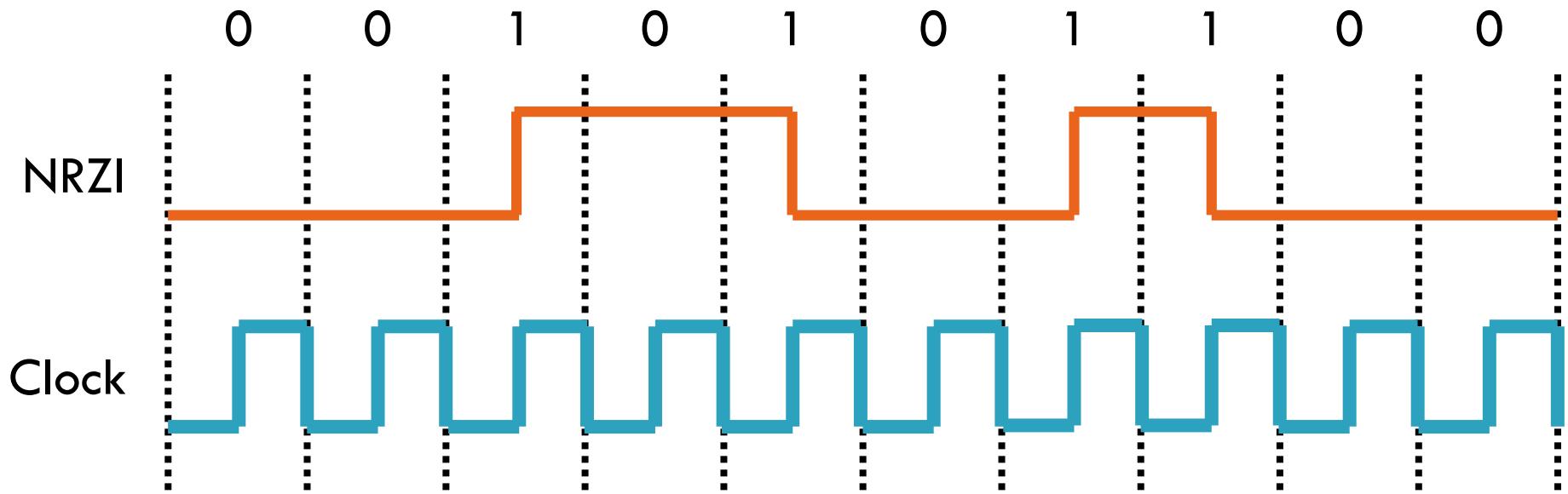


- Megoldás az órák elcsúszásának problémájára (minden bit átmenettel kódolt)
- Negatívum, hogy az átvitel felét használja ki (két óraidő ciklus per bit)

Non-Return to Zero Inverted (NRZI)

39

- 1 → átmenet, 0 → ugyanaz marad



- A csupa egyes sorozat problémáját megoldja ugyan, de a csupa nulla sorozatot ez sem kezeli...

4-bit/5-bit kódolás NRZI előtt (100 Mbps Ethernet - 100BASE-TX)

40

- Megfigyelés:
 - NRZI jól működik, amíg nincs csupa 0-ákból álló sorozat
- Ötlet - Kódoljunk minden 4 hosszú bitsorozatot 5-bitbe:
 - Nem lehet egynél több nulla a sorozat elején, és nem lehet kettőnél több a végén

4-bit	5-bit	4-bit	5-bit
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

- Hátrányok: 20%-ot veszítünk a hatékonyságból

4-bit/5-bit kódolás NRZI előtt (100 Mbps Ethernet - 100BASE-TX)

41

- Megfigyelés:
 - NRZI **8-bit/10-bit kódolás használata Gigabit Ethernet**
- Ötlet - Kódoljunk minden 4 hosszú bitsorozatot 5-bitbe:
 - Nem lehet egynél több nulla a sorozat elején, és nem lehet kettőnél több a végén

4-bit	5-bit	4-bit	5-bit
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

- Hátrányok: 20%-ot veszítünk a hatékonyságból

Jelátvitel

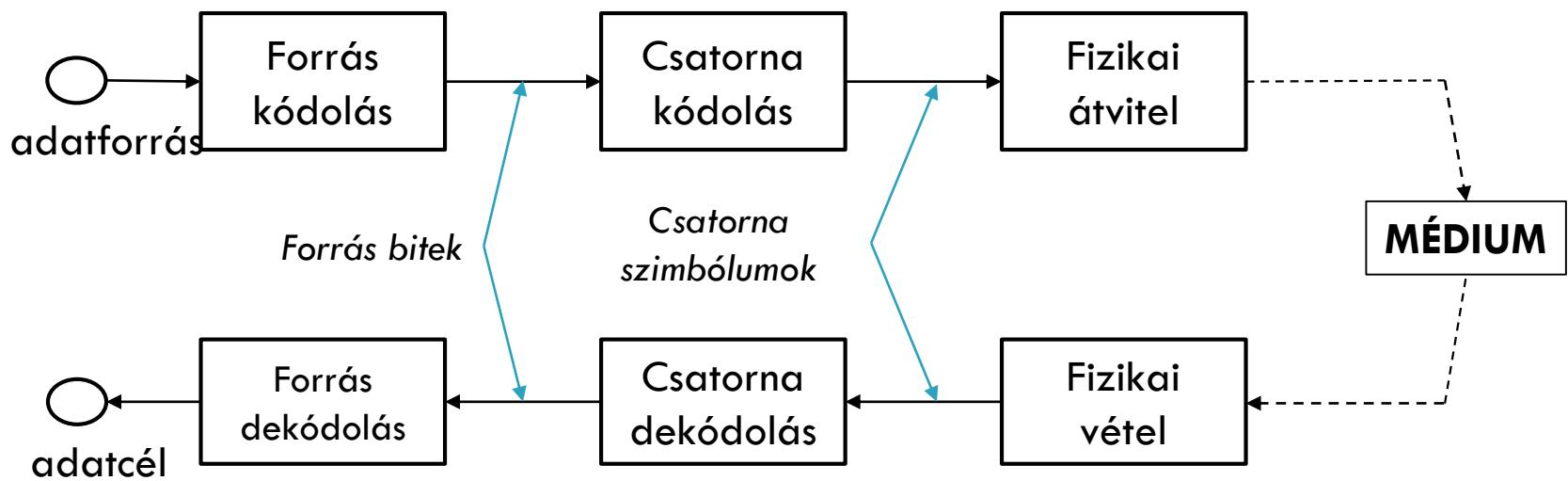
Alapsáv és széles-sáv

43

- Alapsáv vagy angolul *baseband*
 - a digitális jel direkt árammá vagy feszültséggé alakul;
 - a jel minden frekvencián átvitelre kerül;
 - átviteli korlátok.
- Szélessáv vagy angolul *broadband*
 - Egy széles frekvencia tartományban történik az átvitel;
 - a jel modulálására az alábbi lehetőségeket használhatjuk:
 - adatok vivőhullámra „ültetése” (*amplitúdó moduláció*);
 - vivőhullám megváltoztatása (*frekvencia vagy fázis moduláció*);
 - különböző vivőhullámok felhasználása egyidejűleg

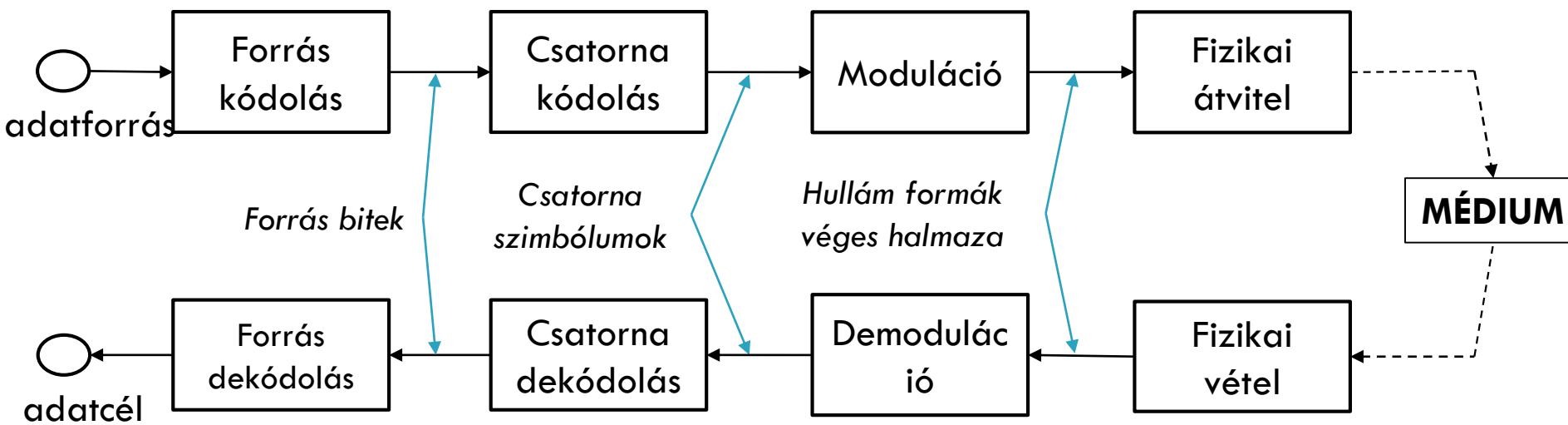
Digitális alapsávú átvitel struktúrája

44



Digitális szélessávú átvitel struktúrája

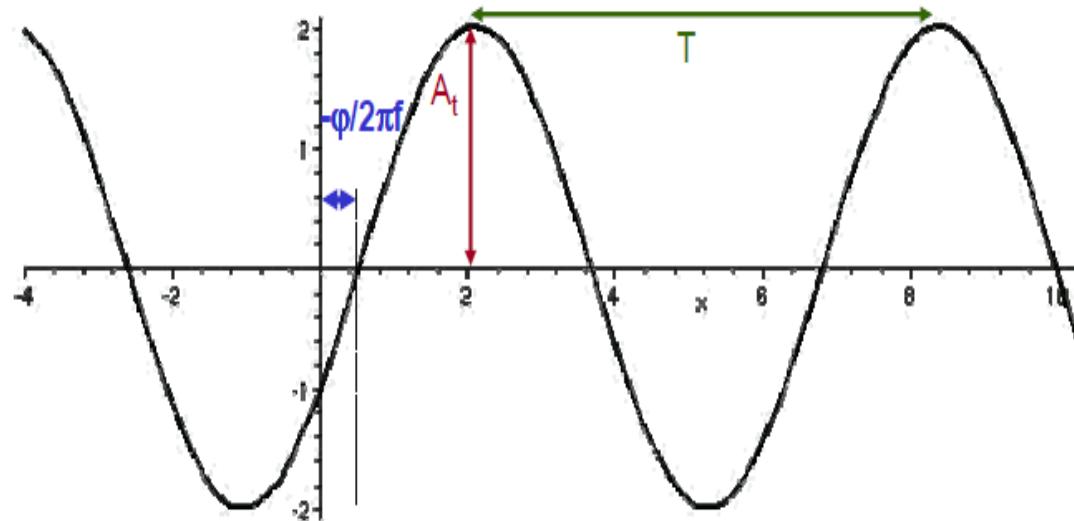
45



Amplitúdó ábrázolás

46

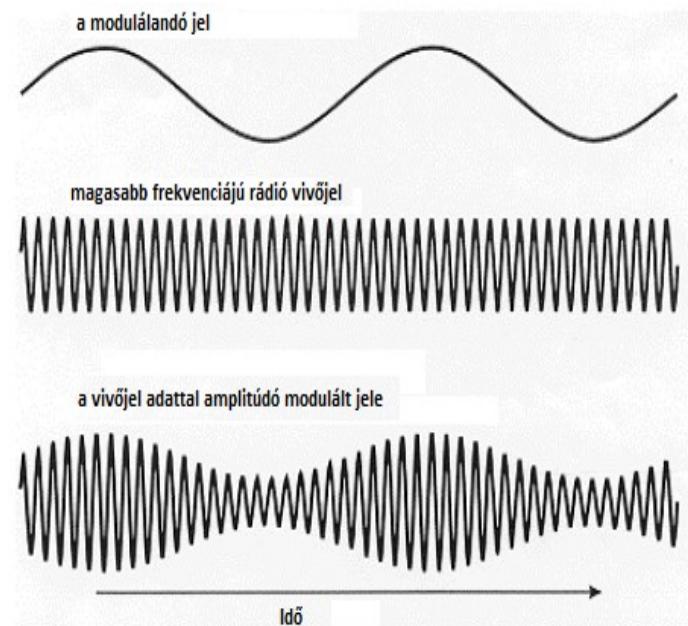
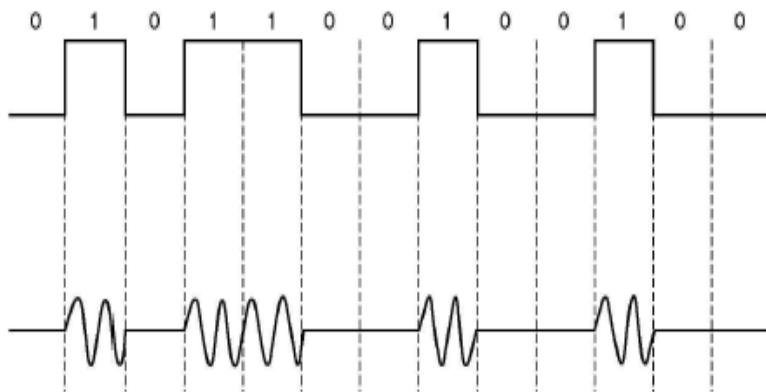
- Egy szinusz rezgés amplitúdó ábrázolása T periódus idejű függvényre $s(t) = A \sin(2\pi ft + \varphi)$, ahol A az amplitúdó, f a frekvencia és φ a fáziseltolás.



Amplitúdó moduláció

47

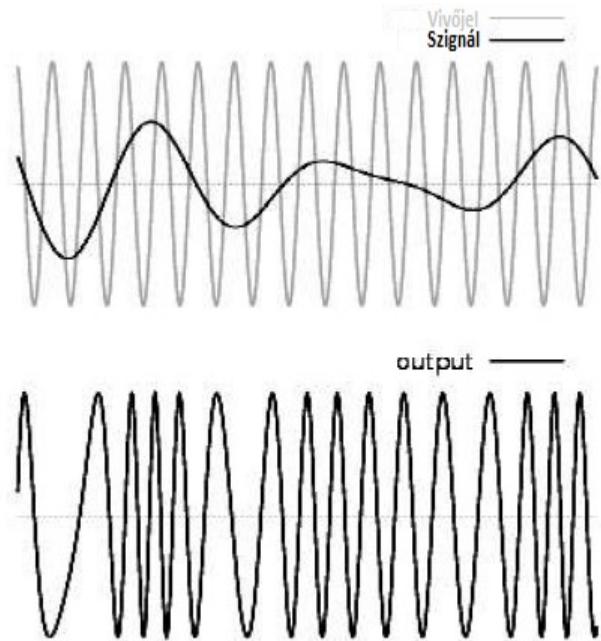
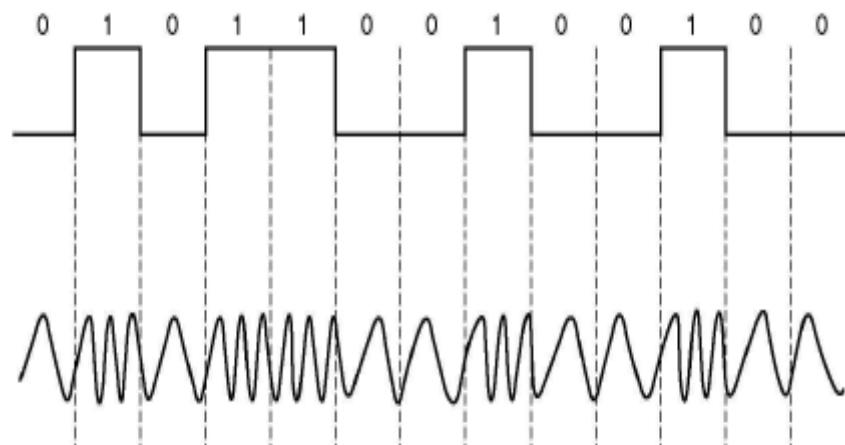
- Az $s(t)$ szignált a szinusz görbe amplitúdójaként kódoljuk, azaz:
$$f_A(t) = s(t) * \sin(2\pi ft + \varphi)$$
- analóg szignál: amplitúdó moduláció
- Digitális szignál: amplitúdó keying (szignál erőssége egy diszkrét halmaz értékeinek megfelelően változik)



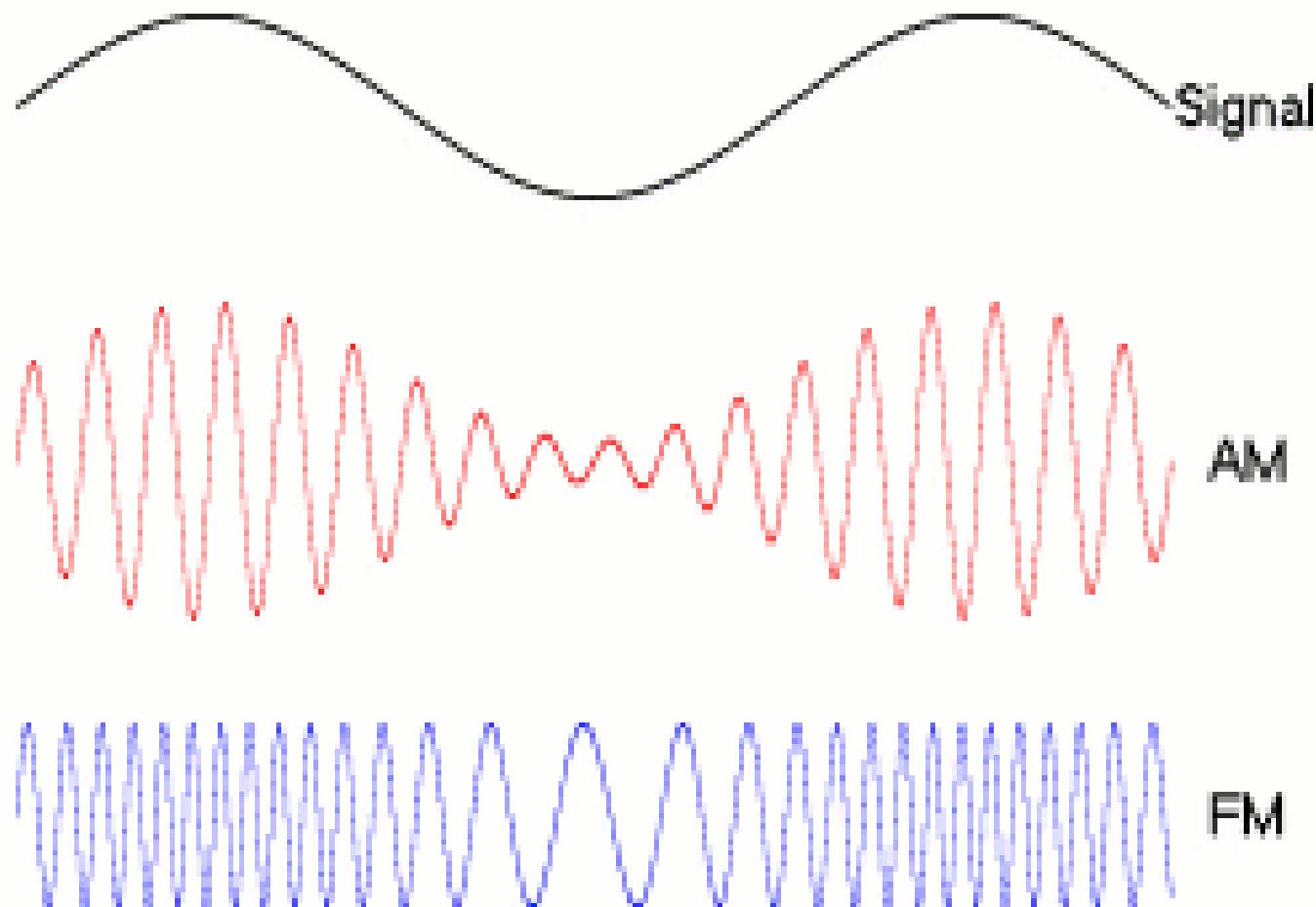
Frekvencia moduláció

48

- Az $s(t)$ szignált a szinusz görbe frekvenciájában kódoljuk, azaz:
$$f_F(t) = a * \sin(2\pi s(t)t + \varphi)$$
- analóg szignál: frekvencia moduláció
- Digitális szignál: frekvencia-eltolás keying (például egy diszkrét halmaz szimbólumaihoz különböző



Illusztráció - AM & FM analóg jel esetén



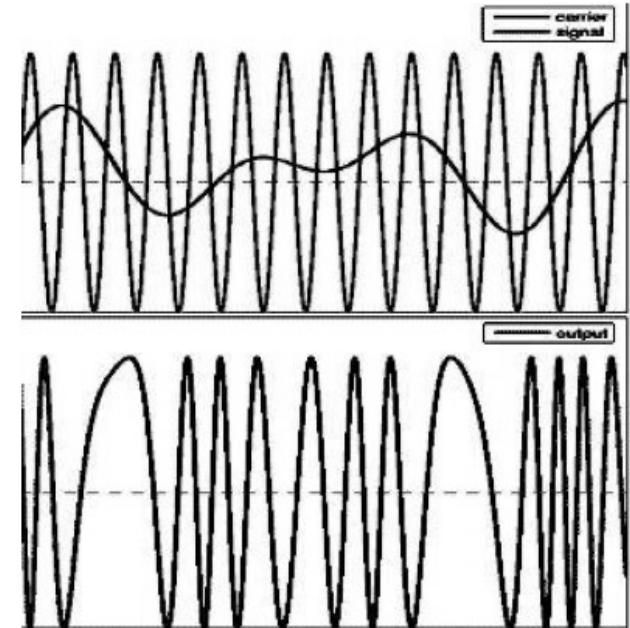
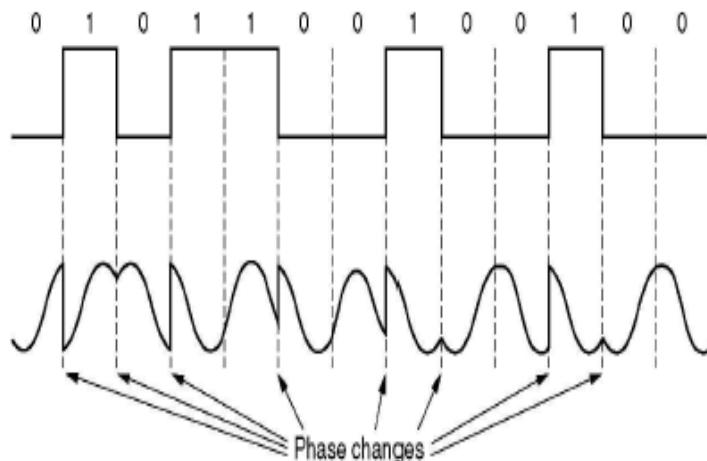
Fázis moduláció

50

- Az $s(t)$ szignált a szinusz görbe fázisában kódoljuk, azaz:

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

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

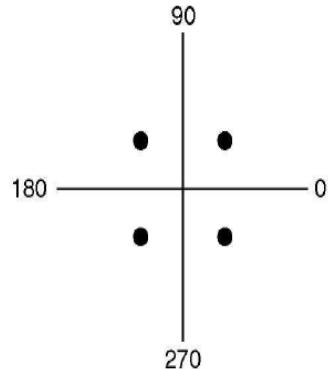


Több szimbólum használata

51

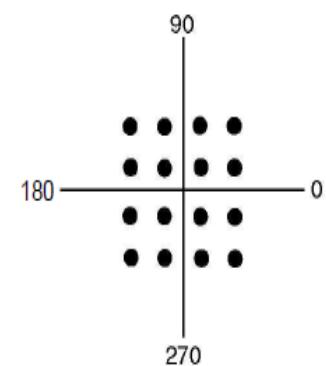
PSK különböző szimbólumokkal

- A fázis eltolások könnyen felismerhetőek a fogadó által
- Diszkrét halmaz kódolja a szimbólumokat
 - ▣ Például 4 szimbólum esetén: $\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}$
 - ▣ Ezzel kétszeres adatrátát kapunk a szimbólum rátához
 - ▣ Ezt nevezzük **Quadrature Phase Shift Keying**



Amplitúdó- és fázis-moduláció

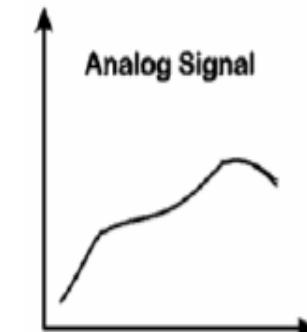
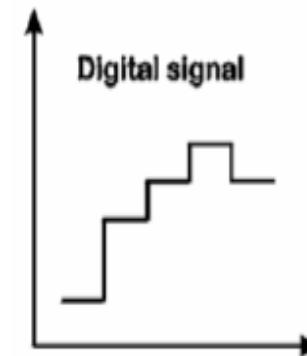
- Kombinálhatóak a módszerek
- Diszkrét halmaz kódolja a szimbólumokat
 - ▣ Például 16 különböző szimbólum (amplitúdó és fázis kombináció) használata
 - ▣ Ezzel négyeszeres adatrátát kapunk a szimbólum rátához
 - ▣ Ezt nevezzük **Quadrature Amplitude Modulation-16**



Digitális és analóg jelek összehasonlítása

52

- *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 előnyei*
 - Lehetőség van a vételpontosság helyreállítására illetve az eredeti jel helyreállítására
- *Analóg hátránya*
 - A fellépő hibák önmagukat erősíthetik



Csatorna hozzáférés módszerei (statikus)

Multiplexálás

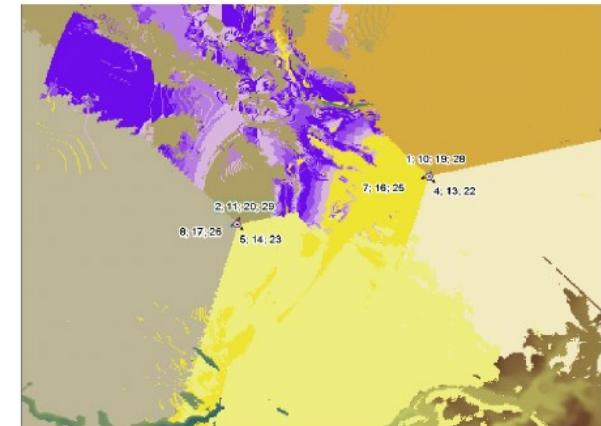
54

- Lehetővé teszi, hogy több jel egyidőben utazzon egy fizikai közegen
- Több jel átvitele érdekében a csatornát logikailag elkülönített kisebb csatornákra (alcsatornákra) bontjuk
- A küldő oldalon szükséges egy speciális eszköz (multiplexer), mely a jeleket a csatorna megfelelő alcsatornáira helyezi

Térbeli multiplexálás

55

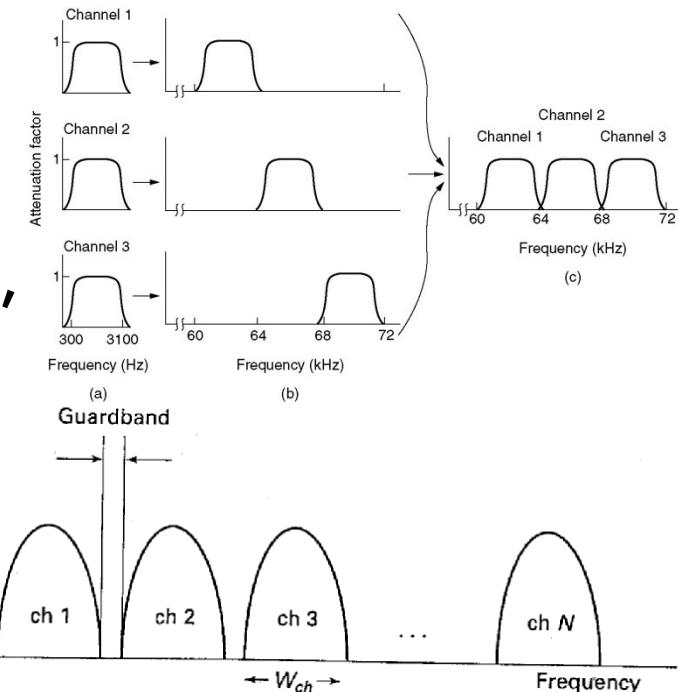
- Ez a legegyszerűbb multiplexálási módszer.
- Angolul **Space-Division Multiplexing**
- Vezetékes kommunikáció esetén minden egyes csatornához külön pont-pont vezeték tartozik.
- Vezeték nélküli kommunikáció esetén minden egyes csatornához külön antenna rendelődik.



Frekvencia multiplexálás

56

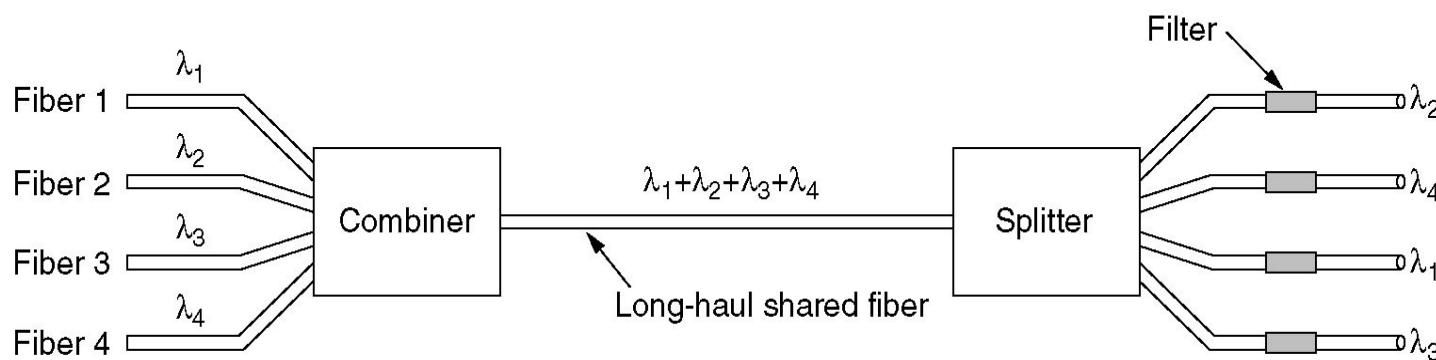
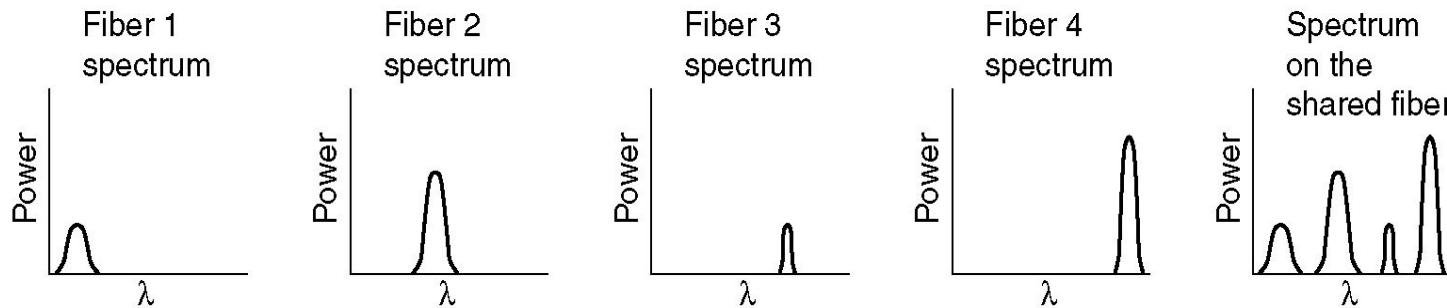
- Olyan módszertan, amelyben egy kommunikációs csatornán több szignál kombinációja adja az átvitelt.
- minden szignálhoz más frekvencia tartozik.
- Angolul **Frequency-Division Multiplexing**
- Tipikusan analóg vonalon használják.
- Többféle megvalósítása van:
 - XOR a szignálokon véletlen bitsorozattal,
 - pszeudo véletlen szám alapú választás



Hullámhossz multiplexálás

57

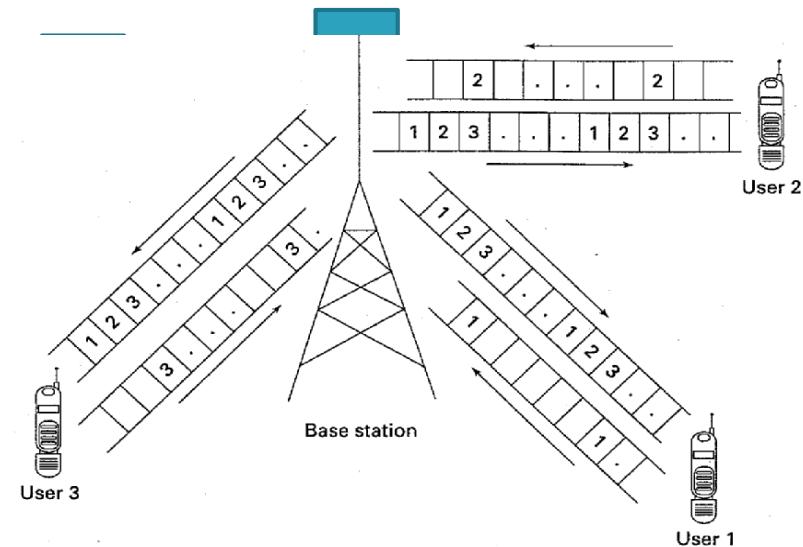
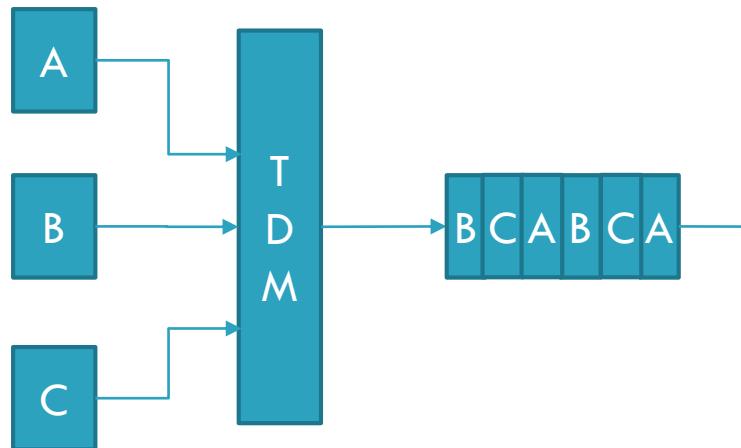
- Optikai kábelek nél alkalmazzák.
- Angolul **Wavelength-Division Multiplexing**



Időbeli multiplexálás

58

- Több párhuzamos adatfolyam átvitelét a jelsorozat rövid időintervallumokra szegmentálásával oldja meg.
- Diszkrét időszeletek használata. minden állomás saját időszeletet kap.
- Angolul **Time-Division Multiplexing**



Code Division Multiple Access 1/3

59

- a harmadik generációs mobiltelefon hálózatok alapját képezi (IS-95 szabvány)
- minden állomás egyfolytában sugározhat a rendelkezésre álló teljes frekvenciasávon
- Feltételezi, hogy a többszörös jelek lineárisan összeadódnak.
- **Kulcsa:** a hasznos jel kiszűrése

ALGORITMUS

- minden bitidőt m darab rövid intervallumra osztunk, ezek a töredékek (angolul *chip*)
- minden állomáshoz egy m bites kód tartozik, úgynevezett töredéksorozat (angolul *chip sequence*)
- Ha 1-es bitet akar továbbítani egy állomás, akkor elküldi a saját töredéksorozatát.
- Ha 0-es bitet akar továbbítani egy állomás, akkor elküldi a saját töredéksorozatának egyes komplementét.

Code Division Multiple Access 2/3

60

- m-szeres sávszélesség válik szükségessé, azaz szórt spektrumú kommunikációt valósít meg
- szemléltetésre bipoláris kódolást használunk:
 - bináris 0 esetén -1; bináris 1 esetén +1
 - az állomásokhoz rendelt töredék sorozatok **páronként ortogonálisak**

Code Division Multiple Access 3/3

61

- szinkron esetben a *Walsh* mátrix oszlopai vagy sorai egyszerű módon meghatároznak egy kölcsönösen ortogonális töredék sorozat halmazt

$$H(2^1) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, H(2^2) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix},$$

$$\forall k \in \mathbb{N} \wedge k \geq 2: H(2^k) = \begin{bmatrix} H(2^{k-1}) & H(2^{k-1}) \\ H(2^{k-1}) & -H(2^{k-1}) \end{bmatrix}$$

Code Division Multiple Access példa

62

A állomás

Chip kódja legyen $(1, -1)$.
Átvitelre szánt adat legyen
 1011

1. Egyedi szignál előállítása az $(1, 0, 1, 1)$ vektorra:
 $((1, -1), (-1, 1), (1, -1), (1, -1))$
2. Szignál modulálása a csatornára.

B állomás

Chip kódja legyen $(1, 1)$.
Átvitelre szánt adat legyen
 0011

1. Egyedi szignál előállítása az $(0, 0, 1, 1)$ vektorra:
 $((-1, -1), (-1, -1), (1, 1), (1, 1))$
2. Szignál modulálása a csatornára.

$$((1 + (-1), (-1) + (-1)), ((-1) + (-1), 1 + (-1)), (1 + 1, (-1) + 1), (1 + 1, (-1) + 1)) = \\ (0, -2, -2, 0, 2, 0, 2, 0)$$

Code Division Multiple Access példa

63

$$((1+(-1),(-1)+(-1)),((-1)+(-1),1+(-1)),(1+1,(-1)+1),(1+1,(-1)+1)) = \\ ((0,-2),(-2,0),(2,0),(2,0))$$

Vevő 1

Ismeri B chip kódját: (1,1).

1. Visszakódolás az ismert kóddal:

$$((0,-2)*(1,1),(-2,0)*(1,1),(2,0)*(1,1),(2,0)*(1,1))$$

2. Kapott (-2,-2,2,2) eredmény

értelemezése:

(-,-,+,+), azaz 0011 volt az üzenet B-től.

Vevő 2

Ismeri A chip kódját: (1,-1).

1. Visszakódolás az ismert kóddal:

$$((0,-2)*(1,-1),(-2,0)*(1,-1),(2,0)*(1,-1),\\ (2,0)*(1,-1))$$

2. Kapott (2,-2,2,2) eredmény

értelemezése:

(+,-,+,+), azaz 1011 volt az üzenet A-tól.

Médium többszörös használata

Összefoglalás

64

- Tér-multiplexálás avagy *SDM* (párhuzamos adatátviteli csatornák)
 - celluláris hálózatok
- Frekvencia-multiplexálás avagy *FDM*(a frekvencia tartomány felosztása és küldőhöz rendelése)
 - „**D**irect **S**equence **SS” (XOR a szignálokon véletlen bitsorozattal)**
 - „**F**requency **H**opping **SS” (pszeudo véletlen szám alapú választás)**
- Idő-multiplexálás avagy *TDM* (a médium használat időszeletekre osztása és küldőhöz rendelése)
 - diszkrét idő szeletek (*slot*)
 - koordináció vagy merev felosztás kell hozzá
- Hullámhossz-multiplexálás avagy *WDM* (optikai frekvencia-multiplexálás)
- Kód multiplexálás avagy *CDM* (mobil kommunikációban használatos)

Köszönöm a figyelmet!

Számítógépes Hálózatok

4. Előadás: Adatkapcsolati réteg

Adatkapcsolati réteg

2



- Szolgáltatás
 - Adatok keretekre tördelése: határok a csomagok között
 - Közeghozzáférés vezérlés (MAC)
 - Per-hop megbízhatóság és folyamvezérlés
- Interfész
 - Keret küldése két közös médiumra kötött eszköz között
- Protokoll
 - Fizikai címzés (pl. MAC address, IB address)
- Példák: Ethernet, Wifi, InfiniBand

Adatkapcsolati réteg

3



□ Funkciók:

- Adat blokkok (**keretek/frames**) küldése eszközök között
- A fizikai közeghez való hozzáférés szabályozása

□ Legfőbb kihívások:

- Hogyan **keretezzük** az adatokat?
- Hogyan ismerjük fel a **hibát**?
- Hogyan vezéreljük a **közeghozzáférést (MAC)**?
- Hogyan oldjuk fel vagy előzzük meg az **ütközési** helyzeteket?

Keret képzés / Keretezés / Framing

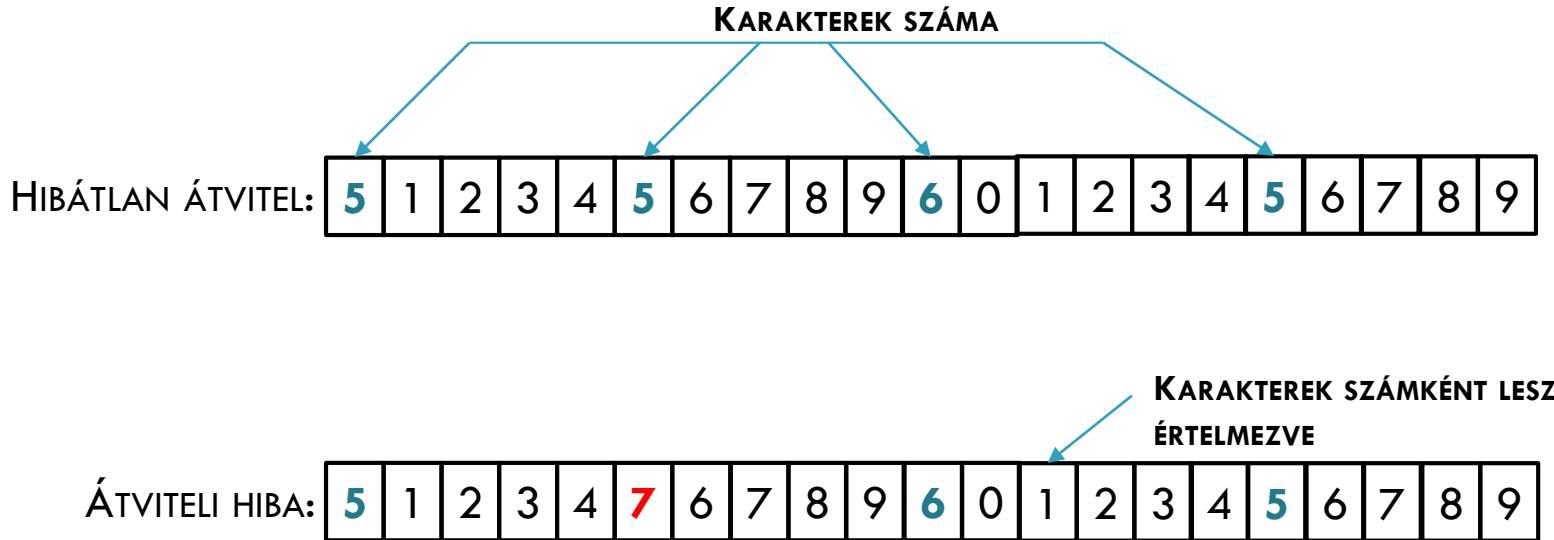
Keret képzés/Keretezés/Framing

5

- A bitek kódolását a fizikai réteg határozza meg
- A következő lépés az adatblokkok „kódolása”
 - Csomag-kapcsolt hálózatok
 - minden csomag útvonal (routing) információt is tartalmaz
 - az adathatárokat ismernünk kell a fejlécek olvasásához
 - a fizikai réteg nem garantál hibamentességet, az adatkapcsolati réteg feladata a hibajelzés illetve a szükség szerint javítás
 - Megoldás: keretekre tördelése a bitfolyamnak, és ellenőrző összegek számítása
 - a keretezés nem egyszerű feladat, mivel megbízható időzítésre nem nagyon van lehetőség
- Keret képzés fajtái
 - Bájt alapú protokollok
 - Bit alapú protokollok
 - Óra alapú protokollok

Bájt alapú: Karakterszámlálás

- a keretben lévő karakterek számának megadása a keret fejlécében lévő mezőben
- a vevő adatkapcsolati rétege tudni fogja a keret végét
- *Probléma:* nagyon érzékeny a hibára a módszer



Bájt alapú: Bájt beszúrás (Byte Stuffing)

7



- Egy speciális **FLAG** bájt (jelölő bájt) jelzi az adat keret elejét és végét
 - Korábban két speciális bájtot használtak: egyet a keret elejéhez és egyet a végéhez
- Probléma: Mi van, ha a **FLAG** szerepel az adat bájtok között is?
 - Szűrjunk be egy speciális **ESC** (Escape) bájtot az „adat” **FLAG** előtt
 - Mi van ha **ESC** is szerepel az adatban?
 - Szűrjunk be egy újabb **ESC** bájtot előtérbe.
 - Hasonlóan a C stringeknél látottakhoz:
 - `printf("You must \"escape\" quotes in strings");`
 - `printf("You must \\\\'escape\\\\ forward slashes as well");`
- Pont-pont alapú protokollok használják: modem, DSL, cellular, ...

Bájt beszúrás példa

KERETEZENDŐ ADAT

H	E	L	L	O	[SPACE]	[ESC]
---	---	---	---	---	---------	-------



KERETEZETT ADAT

[FLAG]	H	E	L	L	O	[SPACE]	[ESC]	[ESC]	[FLAG]
--------	---	---	---	---	---	---------	-------	-------	--------

Bit alapú: Bit beszúrás (Bit stuffing)

9

0111110

Adat

0111110

- minden keret speciális bitmintával kezdődik és végződik (hasonlóan a bájt beszúráshoz)
 - A kezdő és záró bitsorozat ugyanaz
 - Például: 0111110 a High-level Data Link Protocol (HDLC) esetén
- A Küldő az adatban előforduló minden 11111 részsorozat előre 0 bitet szűr be
 - Ezt nevezzük bit beszúrásnak
- A Fogadó miután az 11111 részsorozattal találkozik a fogadott adatban:
 - 11110 → eltávolítja a 0-t (mivel ez a beszúrás eredménye volt)
 - 11111 → ekkor még egy bitet olvas
 - 111110 → keret vége
 - 1111111 → ez hiba, hisz ilyen nem állhat elő a küldő oldalon. Eldobjuk a keretet!
- Hátránya: legrosszabb esetben 20% teljesítmény csökkenés
- Mi történik ha a záró bitminta meghibásodik?

Példa bit beszúrásra

AZ ÁTVITELRE SZÁNT BITSOROZAT BITBESZÚRÁS ELŐTT → 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

AZ ÁTVITELRE SZÁNT BITSOROZAT BITBESZÚRÁS
UTÁN (FEJLÉC NÉLKÜL) → 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0

BESZURT BITEK

A VEVŐNÉL MEGJELENŐ ÜZENET A REDUNDÁNS BITEK
ELTÁVOLÍTÁSA UTÁN → 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Óra alapú keretezés: SONET

□ Synchronous Optical Network

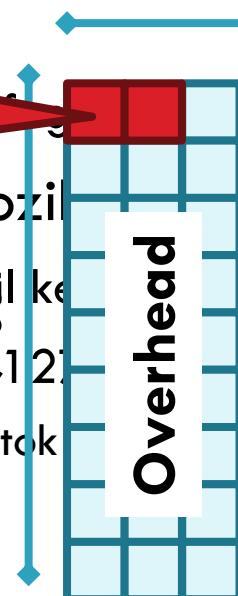
- Nagyon gyors optikai kábelben való átvitel
- STS- n , e.g. STS-1: 51.84 Mbps, STS-768: 36.7 Gbps

□ Az STS-1 keretei rögzített mérettel rendelkeznek

- $9 \times 90 = 810$ bájt \rightarrow 810 bájt fogadása után újabb keret-kezdő mintázat keresése

□ Speciális kezdő mintázat

- A bitek NRZ kódolással kerülnek a hálózatra
- Payload egy speciális 125 oszlopú
- A hosszú 0 és 1 sorozatok



Payload/szállított adat

Hiba felügyelet

Zaj kezelése

13

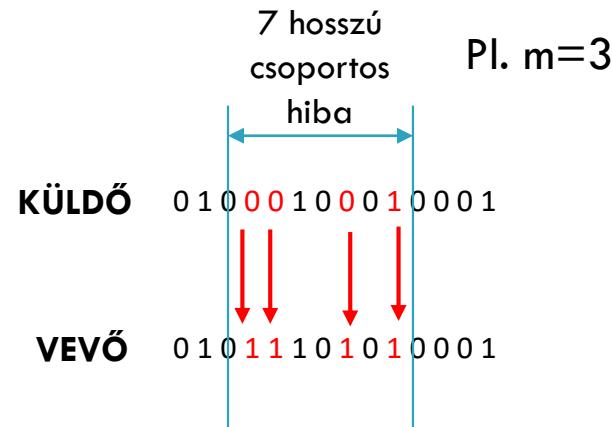
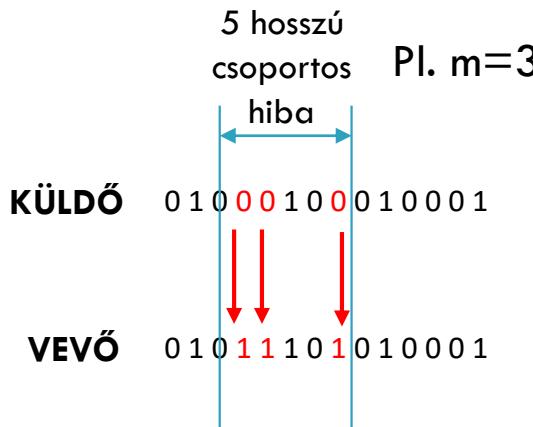
- A fizikai világ eredendően zajos
 - Interferencia az elektromos kábelek között
 - Áthallás a rádiós átvitelek között, mikrosütő, ...
 - Napviharok
- Hogyan detektáljuk a bithibákat az átvitelben?
- Hogyan állítsuk helyre a hibát?

Bithibák definíciók és példák

- egyszerű bithiba – az adategység 1 bitje nulláról egyre vagy egyről nulla változik. Például:

KÜLDŐ	0 1 1 0	0	0 1 0
	↓		
VEVŐ	0 1 1 0	1	0 1 0

- csoportos hiba (angolul *burst error*) – Az átviteli csatornán fogadott bitek egy olyan folytonos sorozata, amelynek az első és utolsó szimbóluma hibás, és nem létezik ezen két szimbólummal határolt részsorozatban olyan m hosszú részsorozat, amelyet helyesen fogadtunk volna a hiba burst-ön belül. A definícióban használt m paramétert védelmi övezetnek (*guard band*) nevezzük. ([Gilbert-Elliott modell](#))



Naiv hibadetektálás

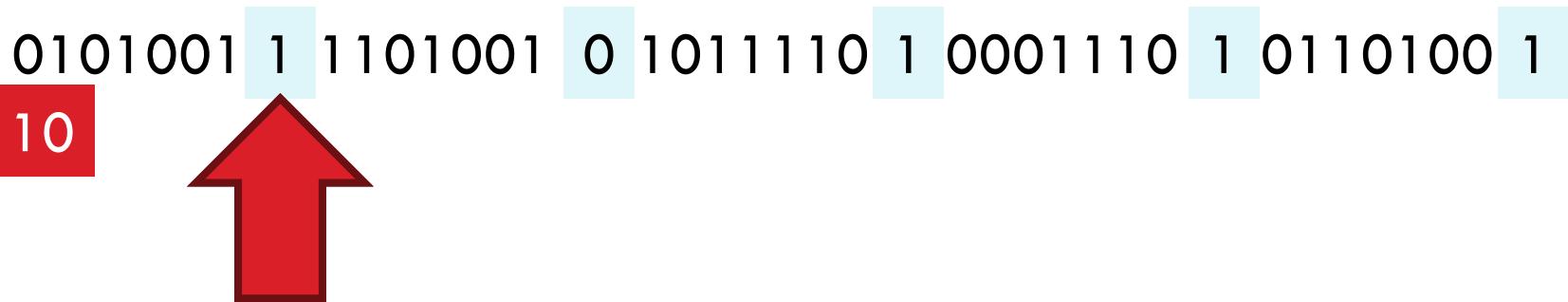
15

- Ötlet: küldjünk két kópiát minden egyes keretből
 - ▣ if (memcmp(frame1, frame2) != 0) { JAJ, HIBA TÖRTÉNT! }
- Miért rossz ötlet ez?
 - ▣ Túl magas ára van / a hatékonyság jelentősen lecsökken
 - ▣ Gyenge hibavédelemmel rendelkezik
 - Lényegében a duplán elküldött adat azt jelenti, hogy kétszer akkora esélye lesz a meghibásodásnak

Paritás Bit

16

- Ötlet: egy extra bitet adunk a bitsorozathoz úgy, hogy az egyesek száma végül **páros** legyen
 - ▣ Példa: 7-bites ASCII karakterek + 1 paritásbit



- 1-bit hiba detektálható
- 2-bit hiba nem detektálható
- Nem megbízható burstös hibák esetén

Hiba vezérlés

□ Stratégiák

□ Hiba javító kódok

- Előre hibajavítás
- Forward Error Correction (FEC)
- kevésbé megbízható csatornákon célszerűbb

□ Hiba detektálás és újraküldés

- Automatic Repeat Request (ARQ)
- megbízható csatornákon olcsóbb

Hiba vezérlés

□ Célok

□ Hiba detektálás

- javítással

- Forward error correction

- Javítás nélkül -> pl. eldobjuk a keretet

- Utólagos hibajavítás

- A hibás keret újraküldése

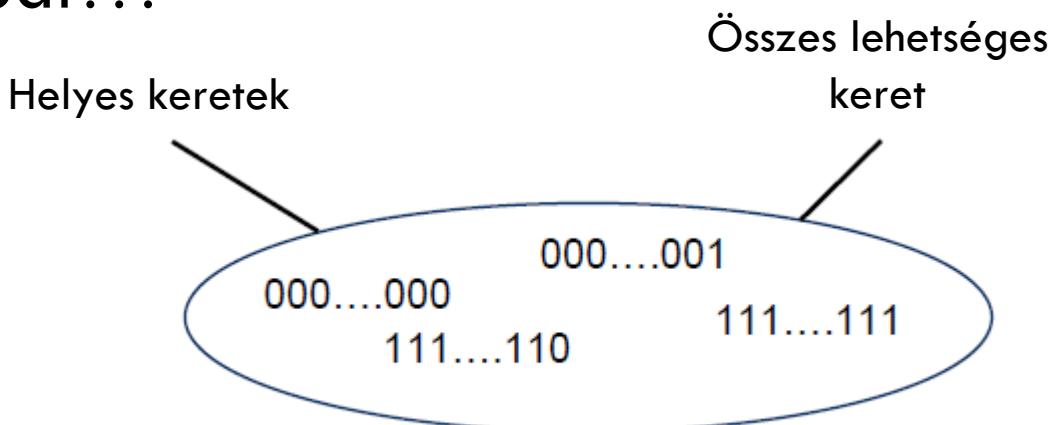
□ Hiba javítás

- Hiba detektálás nélkül

- Pl. hangátvitel

Redundancia

- Redundancia szükséges a hiba vezérléshez
- Redundancia nélkül
 - 2^m lehetséges üzenet írható le m biten
 - Mindegyik helyes (legal) üzenet és fontos adatot tartalmazhat
 - Ekkor minden hiba egy új helyes (legal) üzenetet eredményez
 - A hiba felismerése lehetetlen
- Hogyan ismerjük fel a hibát???



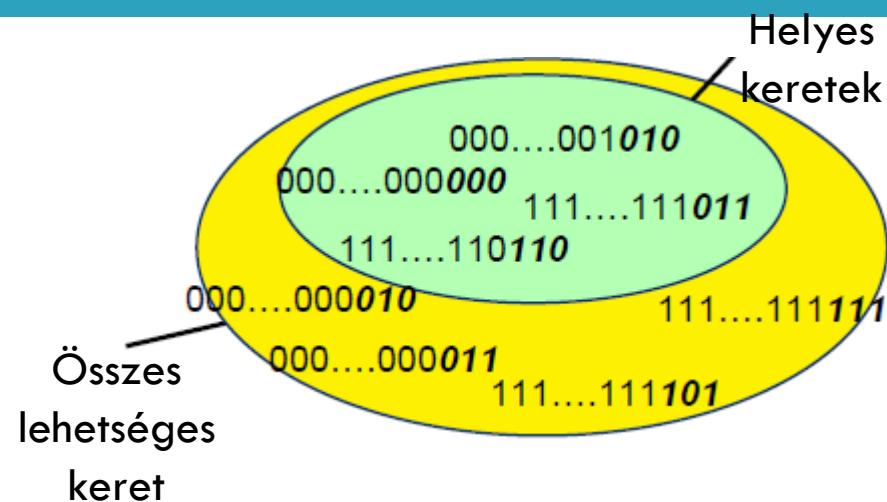
Redundancia

□ Egy keret felépítése:

- m adat bit (ez az üzenet)
- r redundáns/ellenőrző bit

■ Az üzenetből számolt,
új információt nem hordoz

- A teljes keret hossza: $n = m + r$

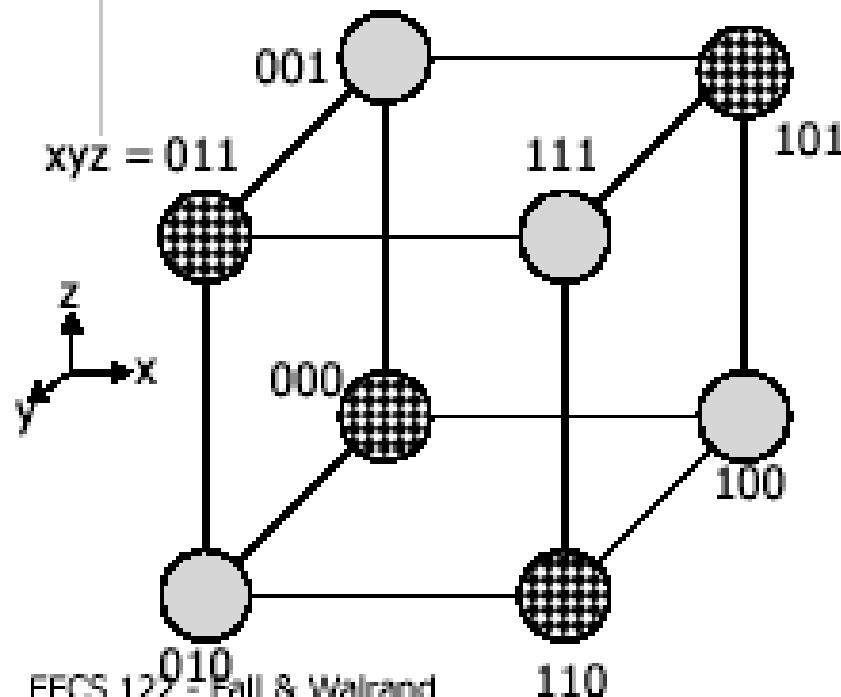


- Az így előálló n bites bitsorozatot n hosszú kódszónak nevezzük!

Error Control Codes

How Codes Work: Words and Codewords

- ◆ Code = subset of possible words: Codewords
- ◆ Example:
 - 3 bits => 8 words; codewords: subset



Words:
000, 001, 010, 011
100, 101, 110, 111

Code:
000, 011, 101, 110

Send only codewords

Elméleti alapok

- Tegyük fel, hogy a keret m bitet tartalmaz. (*üzenet bitek*)
- A redundáns bitek száma legyen r . (*ellenőrző bitek*)
- A küldendő keret tehát $n=m+r$ bit hosszú. (*kódszó*)

Hamming távolság

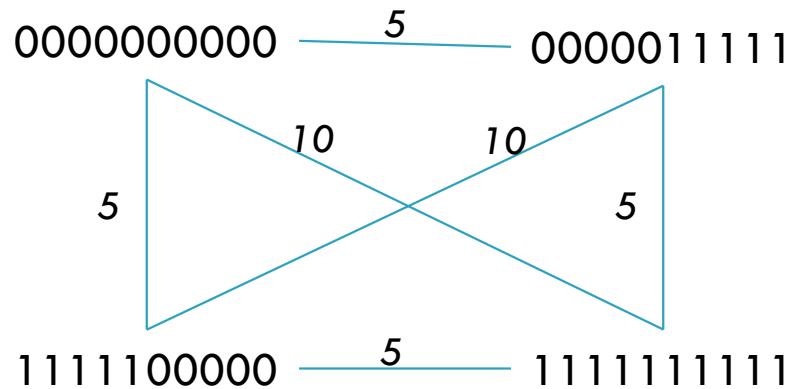
- Az olyan bitpozíciók számát, amelyeken a két kódszóban különböző bitek állnak, a két kódszó Hamming távolságának nevezzük.
 - Jelölés: $d(x,y)$
- Legyen S egyenlő hosszú bitszavak halmaza, ekkor S Hamming távolsága az alábbi:

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

- Jelölés: $d(S)$
- A Hamming távolság egy metrika.

Példa Hamming távolságra

- Legyen $S = \{0000000000, 0000011111, 1111100000, 1111111111\}$.
- Mi lesz a halmaz Hamming távolsága?
 - $d(S) = 5$



Hamming távolság használata

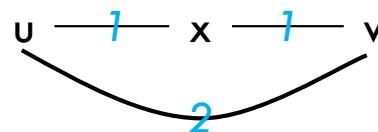
S halmaz legyen a megengedett azonos hosszú kódszavak halmaza.

$d(S)=1$ esetén

- nincs hibafelismerés
- megengedett kódszóból megengedett kódszó állhat elő 1 bit megváltoztatásával

$d(S)=2$ esetén

- ha az u kódszóhoz létezik olyan x megengedett kódszó, amelyre $d(u,x)=1$, akkor hiba történt.
- Feltéve, hogy az u és v megengedett kódszavak távolsága minimális, akkor a következő összefüggésnek teljesülnie kell: $2 = d(u,v) \leq d(u,x) + d(x,v)$.
- Azaz egy bithiba felismerhető, de nem javítható.



Hamming korlát bináris kódkönyvre 1/3

TÉTEL

Minden $C \subseteq \{0,1\}^n$ kód , ahol $d(C) = k$ ($\in \mathbb{N}_+$). Akkor teljesül az alábbi összefüggés:

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

BIZONYÍTÁS

1. Hány olyan bitszó létezik, amely egy tetszőleges $x \in C$ kódszótól pontosan $i \in \mathbb{N}_+$ távolságra helyezkedik el?
 - Pontosan $\binom{n}{i}$ lehetőség van.
2. Hány olyan bitszó létezik, amely egy tetszőleges $x \in C$ kódszótól legfeljebb $\left\lfloor \frac{k-1}{2} \right\rfloor$ távolságra helyezkedik el?
 - Pontosan $\sum_{i=0}^{\left\lfloor \frac{k-1}{2} \right\rfloor} \binom{n}{i}$ lehetőség van.

Hamming korlát bináris kódkönyvre 2/3

3. Lássuk be, hogy egy tetszőleges $x \in \{0,1\}^n$ bitszóhoz legfeljebb egy legális $u \in C$ kódszó létezhet, amelyre $d(x, u) \leq \frac{k-1}{2}$ teljesül.
- Indirekt tegyük fel, hogy létezhet két legális kódszó is a C kódkönyvben, jelölje őket u_1 és u_2 . Ekkor viszont az alábbi két feltétel együttesen teljesül:

$$d(x, u_1) \leq \frac{k-1}{2} \text{ és } d(x, u_2) \leq \frac{k-1}{2}$$

- Mi a két kódszó távolsága?

$$d(u_2, u_1) \leq d(u_2, y) + d(y, u_1) \leq \frac{k-1}{2} + \frac{k-1}{2} = k-1$$

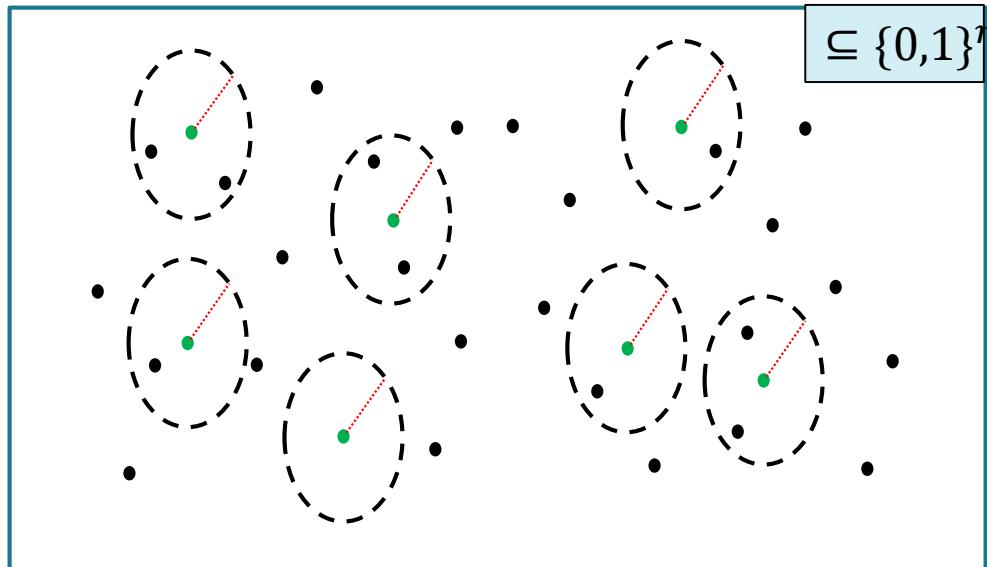
- Ez viszont ellentmond annak hogy a kódkönyv Hamming távolsága k , azaz az indirekt feltevésünk volt hibás. Vagyis tetszőleges bitszóhoz legfeljebb egy legális kódszó létezhet, amely a kódkönyv minimális távolságának felénél közelebb van a bitszóhoz.

Hamming korlát bináris kódkönyvre 3/3

4. A kódszavak $\frac{k-1}{2}$ sugarú környezeteiben található bitszavak egymással diszjunkt halmazainak uniója legfeljebb az n -hosszú bitszavak halmazát adhatja ki. Vagyis formálisan:

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

□



JELMAGYARÁZAT

- Kódszó
- Bitszó, amely nem kódszó

Hibafelismerés és javítás Hamming távolsággal

Hibafelismerés

- d bit hiba felismeréséhez a megengedett keretek halmazában legalább $d+1$ Hamming távolság szükséges.

Hibajavítás

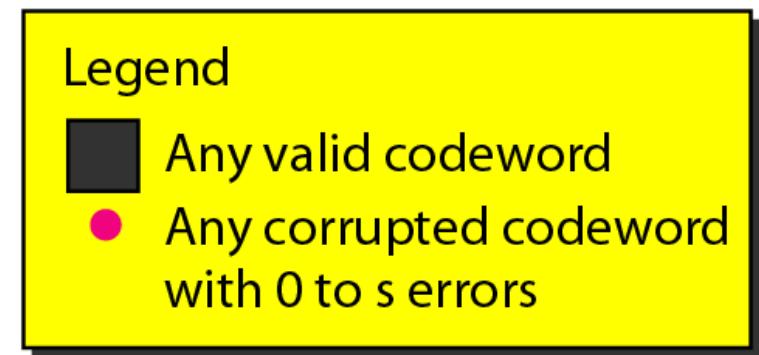
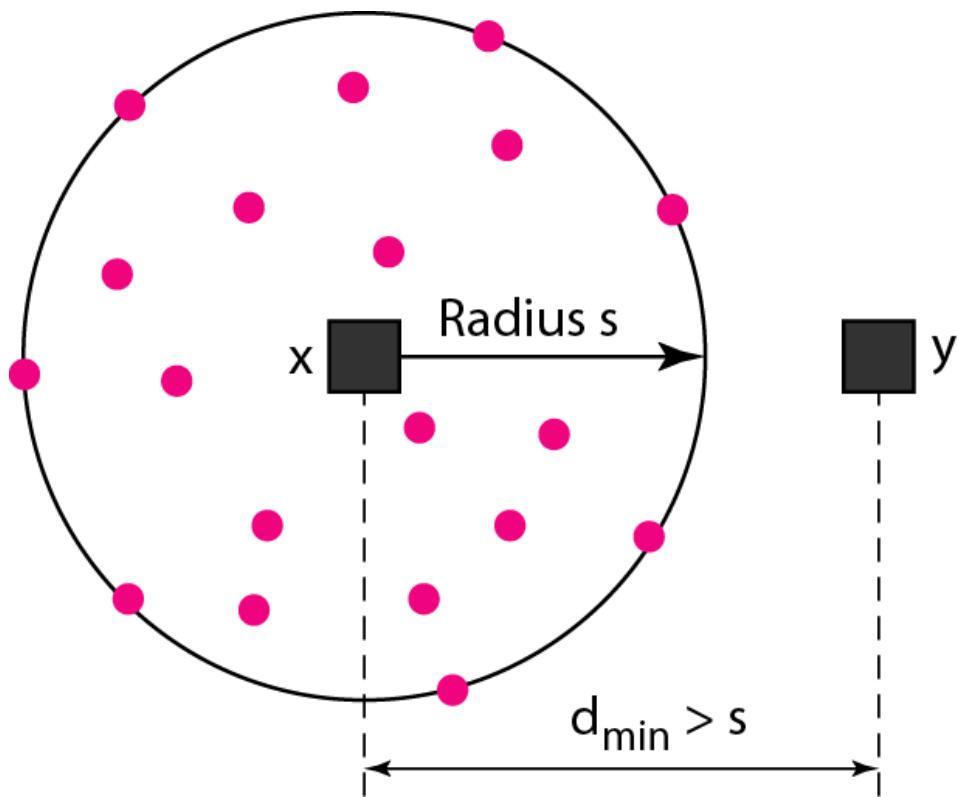
- d bit hiba javításához a megengedett keretek halmazában legalább $2d+1$ Hamming távolság szükséges

Definíciók

- Egy $S \subseteq \{0,1\}^n$ kód rátája $R_S = \frac{\log_2 |S|}{n}$. (a hatékonyságot karakterizálja)
- Egy $S \subseteq \{0,1\}^n$ kód távolsága $\delta_S = \frac{d(S)}{n}$. (a hibakezelési lehetőségeket karakterizálja)
- A jó kódoknak a rátája és a távolsága is nagy.

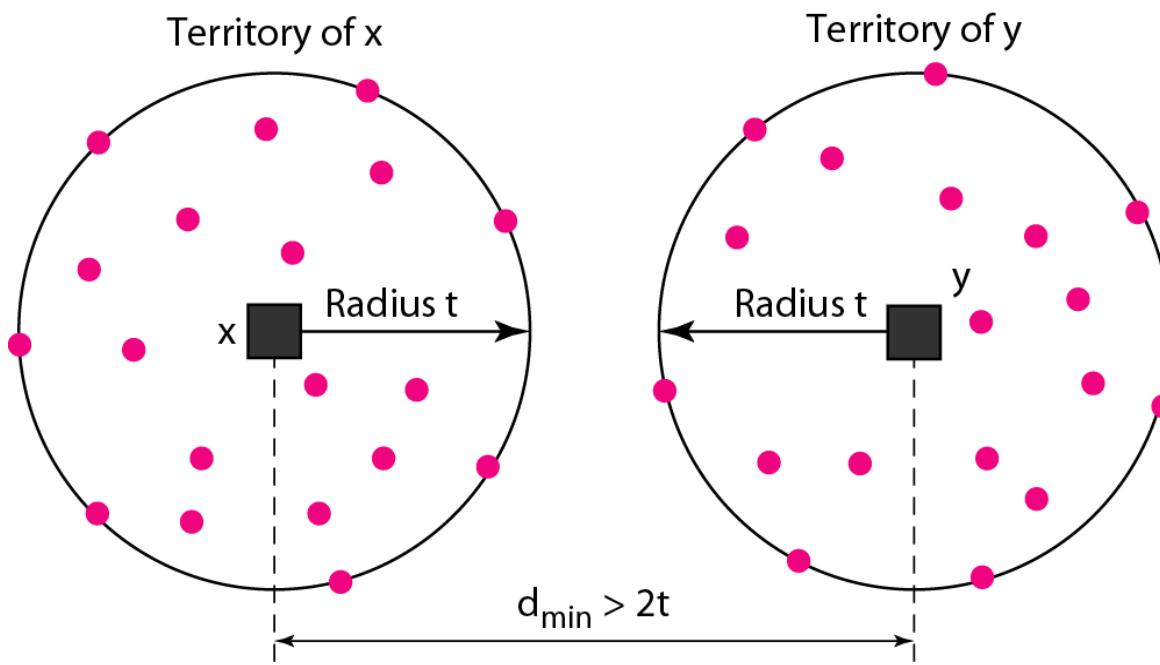
Hiba felismerés

d bithiba felismeréséhez legalább $d+1$ Hamming távolságú kód szükséges.



Hiba javítás

d bithiba javításához legalább $2d+1$ Hamming-távolságú kód szükséges.

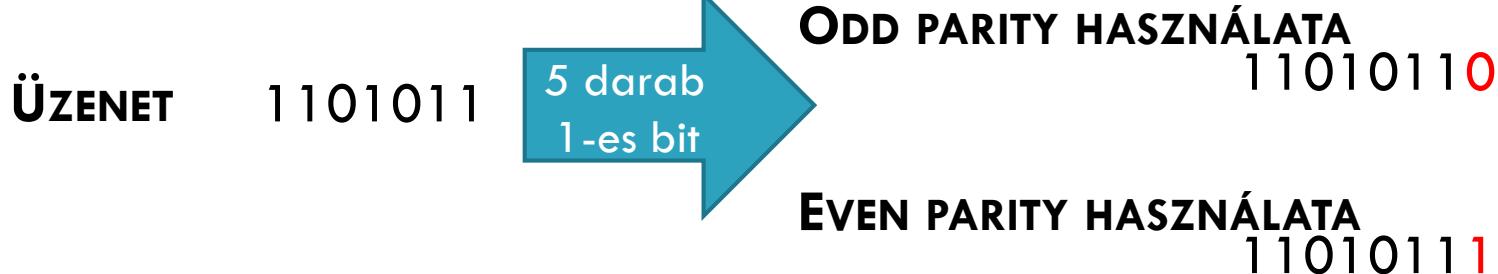


Legend

- Any valid codeword
- Any corrupted codeword with 1 to t errors

Újra a paritás bit használata 1/4

- a paritásbitet úgy választjuk meg, hogy a kódszóban levő 1-ek száma páros (vagy páratlan)
- **Odd parity** – ha az egyesek száma páratlan, akkor 0 befűzése; egyébként 1-es befűzése
- **Even parity** – ha az egyesek száma páros, akkor 0 befűzése; egyébként 1-es befűzése



Paritás bit használata 2/4

Egy paritást használó módszer (Hamming)

- a kódszó bitjeit számozzuk meg 1-gyel kezdődően;
- 2 egészhatvány sorszámú pozíciói lesznek az ellenőrző bitek, azaz 1,2,4,8,16,...;
- a maradék helyeket az üzenet bitjeivel töltjük fel;
- minden egyik ellenőrző bit a bitek valamelyen csoportjának a paritását állítja be párosra (vagy páratlanra)
- egy bit számos paritásszámítási csoportba tartozhat:
 - k pozíciót írjuk fel kettő hatványok összegeként, a felbontásban szereplő ellenőrző pozíciók ellenőrzik a k -adik pozíciót
 - Példa: $k=13$ -ra $k=1+4+8$, azaz az első, a negyedik illetve a nyolcadik ellenőrző bit fogja ellenőrizni

Paritás bit használata - példa 3/4

- Az ASCII kód 7 biten ábrázolja a karaktereket
- A példában *EVEN PARITY-t* használunk

ÜZENET BITEK KÓDSZÓBAN LÉVŐ POZÍCIÓNAK FELBONTÁSAI

- $3 = 1 + 2$
- $5 = 1 + 4$
- $6 = 2 + 4$
- $7 = 1 + 2 + 4$
- $9 = 1 + 8$
- $10 = 2 + 8$
- $11 = 1 + 2 + 8$

ASCII karakter	ASCII decimális	Üzenet forrás bitjei	Az előállt kódszavak
E	69	1000101	10100000101
L	76	1001100	10110011100
T	84	1010100	00110101100
E	69	1000101	10100000101
	32	0100000	10001100000
I	73	1001001	11110011001
K	75	1001011	00110010011

Paritás bit használata 4/4

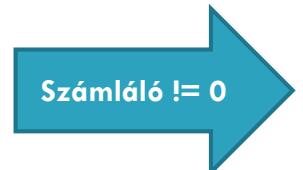
- a vevő az üzenet megérkezésekor 0-ára állítja a számlálóját, ezt követően megvizsgálja a paritás biteket, ha a k -adik paritás nem jó, akkor a számlálóhoz ad k -t
- Ha a számláló 0 lesz, akkor érvényes kódszónak tekinti a vevő a kapott üzenetet; ha a számláló nem nulla, akkor a hibás bit sorszámát tartalmazza, azaz ha például az első, a második és nyolcadik bit helytelen, akkor a megváltozott bit a tizenegyedik.

FOGADOTT E KARAKTER **10100100101**



SZÁMLÁLÓ = 2 + 4

FOGADOTT L KARAKTER **11110011100**



SZÁMLÁLÓ = 2

Hibajelző kódok

Hibajelző kódok

Polinom-kód, avagy ciklikus redundancia (CRC kód)

- Tekintsük a bitsorozatokat \mathbb{Z}_2 feletti polinomok reprezentációinak.

Polinom ábrázolása \mathbb{Z}_2 felett

$$p(x) = \sum_{i=0}^n a_i x^i = a_n x^n + \cdots + a_1 x^1 + a_0 x^0, \text{ ahol } a_i \in \{0,1\}$$

- A számítás mod 2 történik. (összeadás, kivonás, szorzás, osztás)
- reprezentálható az együtthatók $n+1$ -es vektorával, azaz (a_n, \dots, a_1, a_0)
- Például az ASCII „b” karakter kódja 01100010, aminek megfelelő polinom hatod fokú polinom
 $p(x) = 1 * x^6 + 1 * x^5 + 0 * x^4 + 0 * x^3 + 0 * x^2 + 1 * x^1 + 0 * x^0$
- Az összeadás és a kivonás gyakorlati szempontból a logikai KIZÁRÓ VAGY műveettel azonosak.

$$\begin{array}{r} 11110000 \\ - 10100110 \\ \hline 01010110 \end{array}$$
$$\begin{array}{r} 10011011 \\ + 11001010 \\ \hline 01010001 \end{array}$$

Definiáljuk a $G(x)$ generátor polinomot (G foka r), amelyet a küldő és a vevő egyaránt ismer.

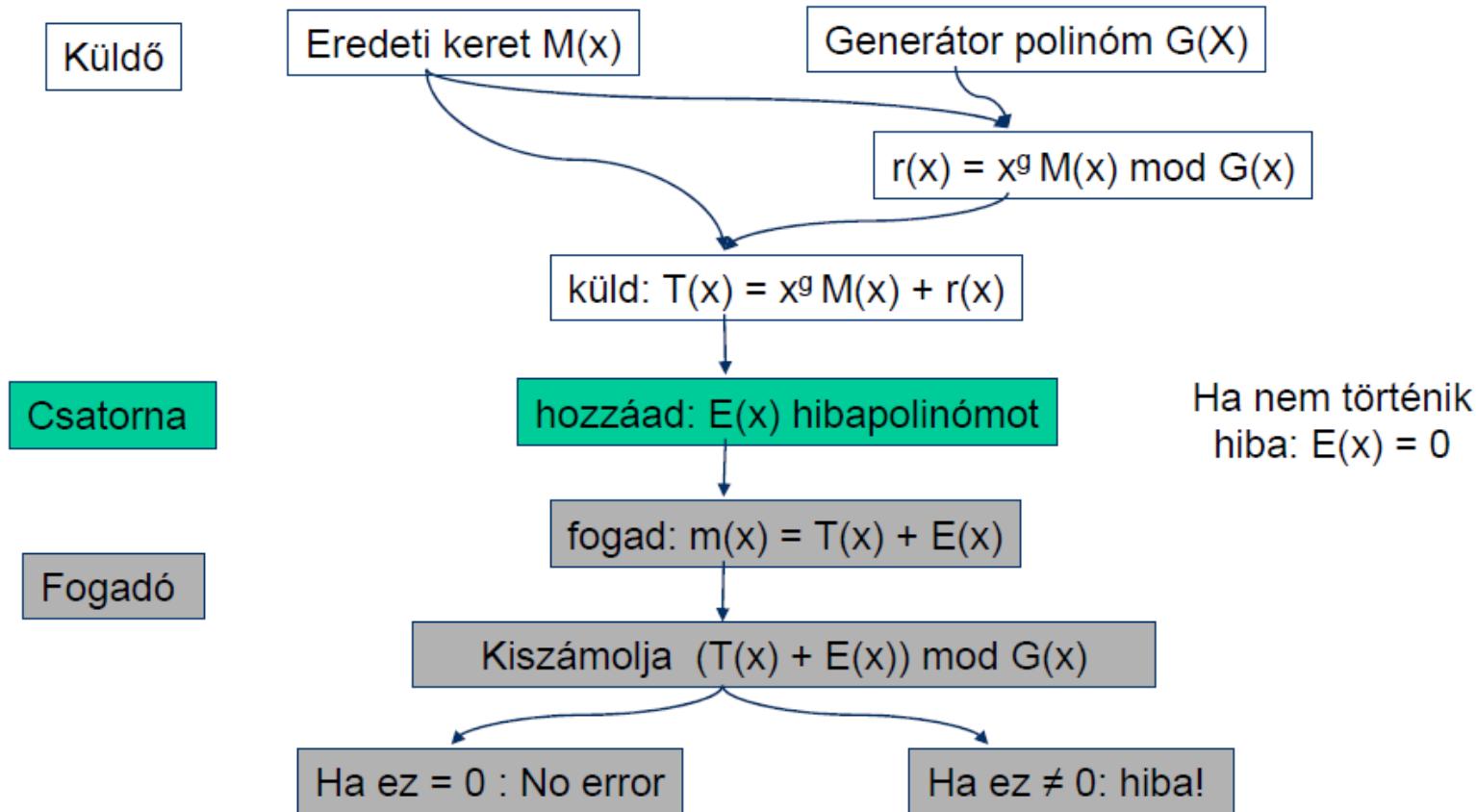
Algoritmus

1. Legyen $G(x)$ foka r . Fűzzünk r darab 0 bitet a keret alacsony helyi értékű végéhez, így az $m+r$ bitet fog tartalmazni és az $x^r M(x)$ polinomot fogja reprezentálni.
2. Osszuk el az $x^r M(x)$ tartozó bitsorozatot a $G(x)$ -hez tartozó bitsorozattal modulo 2.
3. Vonjuk ki a maradékot (mely minden r vagy kevesebb bitet tartalmaz) az $x^r M(x)$ -hez tartozó bitsorozatból moduló 2-es kivonással. Az eredmény az ellenőrző összeggel ellátott, továbbítandó keret. Jelölje a továbbítandó keretnek megfelelő a polinomot $T(x)$.
4. A vevő a $T(x) + E(x)$ polinomnak megfelelő sorozatot kapja, ahol $E(x)$ a hiba polinom. Ezt elosztja $G(x)$ generátor polinommal.
 - Ha az osztási maradék, amit $R(x)$ jelöl, nem nulla, akkor hiba történt.

CRC áttekintés

38

□ Forrás: Dr. Lukovszki Tamás fóliái

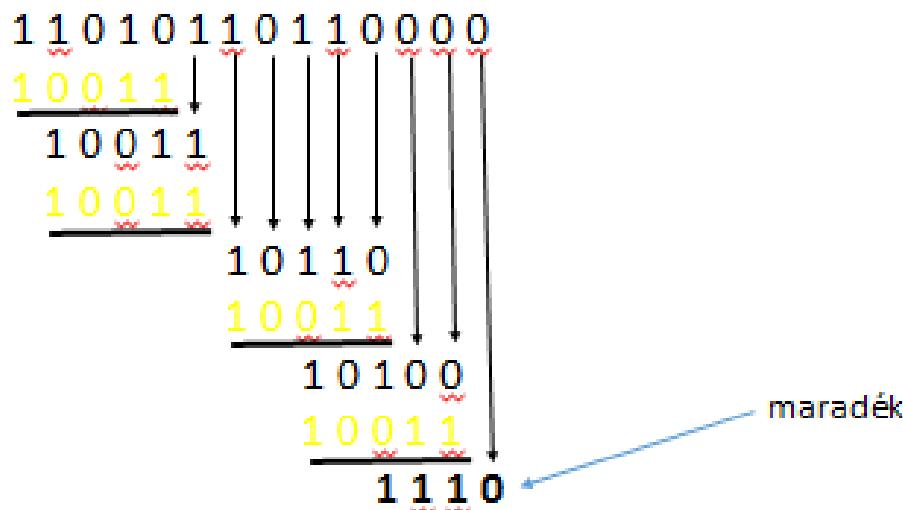


Példa CRC számításra

Keret: 1101011011

Generátor: 10011

A továbbítandó üzenet: 11010110111110



CRC áttekintés

- A $G(x)$ többszöröseinek megfelelő bithibákat nem ismerjük fel, azaz, ha $\exists j \in \mathbb{N}: E(x) = x^j G(x)$.
- $G(x)$ legmagasabb illetve legalacsonyabb fokú tagjának együtthatója minden 1.

Hiba események

- $E(x) = x^i$, azaz i a hibás bit sorszáma, mivel $G(x)$ kettő vagy több tagból áll, ezért minden egy bites hibát jelezni tud.
- $E(x) = x^i + x^j = x^j (x^{i-j} + 1)$ ($i > j$), azaz két izolált egy bites hiba esetén.
 - $G(x)$ ne legyen osztható x -szel;
 - $G(x)$ ne legyen osztható $(x^k + 1)$ -gyel semmilyen maximális kerethossznál kisebb k -ra. (Pl. $x^{15} + x^{14} + 1$)
- Ha $E(x)$ páratlan számú tagot tartalmaz, akkor nem lehet $x+1$ többszöröse. Azaz, ha $G(x)$ az $x+1$ többszöröse, akkor minden páratlan számú hiba felismerhető
- Egy r ellenőrző bittel ellátott polinom-kód minden legfeljebb r hosszúságú csoportos hibát jelezni tud

CRC a gyakorlatban

- IEEE 802 által használt polinom az
$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$$
- Néhány jó tulajdonságai a fenti polinomnak:
 1. minden legfeljebb 32 bites hibacsomót képes jelezni,
 2. minden páratlan számú bitet érintő hibacsomót tud jelezni.

Peterson és Brown (1961)

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

Számítógépes Hálózatok

5. Előadás: Adatkapcsolati réteg

Adatkapcsolati réteg

2



- Szolgáltatás
 - Adatok keretekre tördelése: határok a csomagok között
 - Közeghozzáférés vezérlés (MAC)
 - Per-hop megbízhatóság és folyamvezérlés
- Interfész
 - Keret küldése két közös médiumra kötött eszköz között
- Protokoll
 - Fizikai címzés (pl. MAC address, IP address)
- Példák: Ethernet, Wifi, InfiniBand

Adatkapcsolati réteg

3



□ Funkciók:

- Adat blokkok (**keretek/frames**) küldése eszközök között
- A fizikai közeghez való hozzáférés szabályozása

□ Legfőbb kihívások:

- Hogyan **keretezzük** az adatokat?
- Hogyan ismerjük fel a **hibát**?
- Hogyan vezéreljük a **közeghozzáférést (MAC)**?
- Hogyan oldjuk fel vagy előzzük meg az **ütközési** helyzeteket?

Forgalomszabályozás

Forgalomszabályozás

- gyors adó lassú vevő problémája (elárasztás)
- még hibamentes átvitel esetén se lesz képes a vevő kezelni a bejövő kereteket

Megoldási lehetőségek

1. visszacsatolás alapú forgalomszabályozás (avagy angolul *feedback-based flow control*)
 - engedélyezés
2. Sebesség alapú forgalomszabályozás (avagy angolul *rate-based flow control*)
 - protokollba integrált sebességkorlát
 - az adatkapcsolati réteg nem használja

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

Kommunikációs fajták

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

Korlátozás nélküli szimplex protokoll

a legegyszerűbb protokoll („utópia”)

A környezet

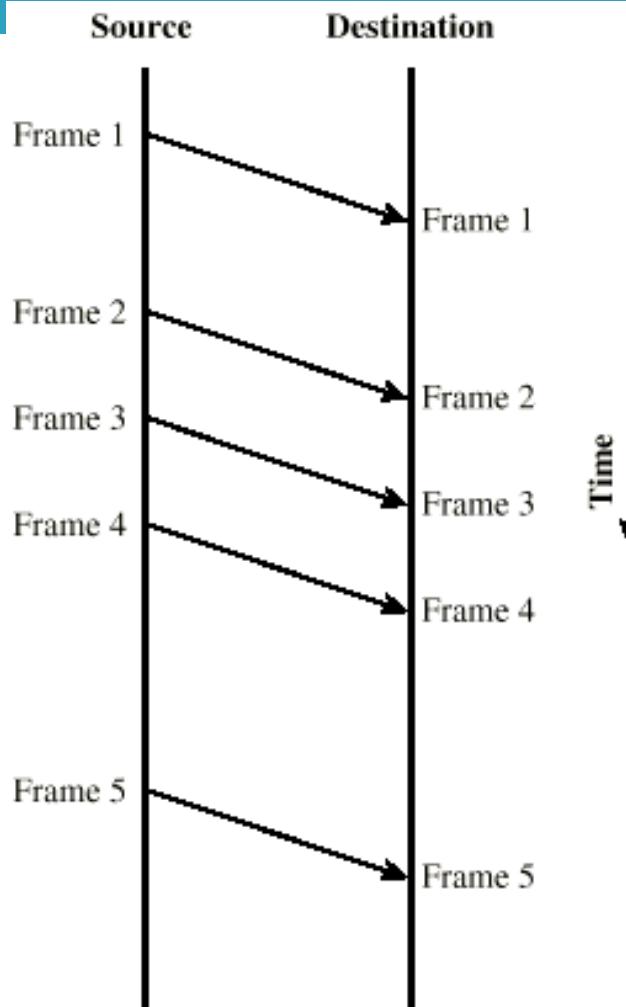
- mind az adó, mind a vevő hálózati rétegei mindenkorban készen állnak;
- a feldolgozási időktől eltekintünk;
- végtelen puffer-területet feltételezünk;
- Az adatkapcsolati rétegek közötti kommunikációs csatorna sosem rontja vagy veszíti el a kereteket;

A protokoll

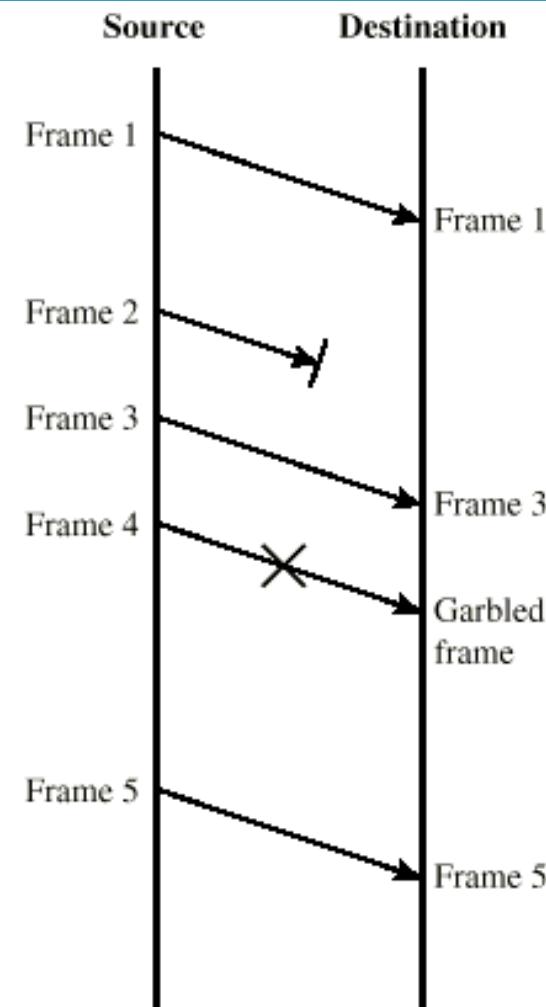
- résztvevők: *küldő* és *vevő*;
- nincs sem sorszámozás, sem nyugta;
- küldő végtelen ciklusban küldi kifele a kereteket folyamatosan;
- a vevő kezdetben várakozik az első keret megérkezésére, keret érkezésekor a hardver puffer tartalmát változóba teszi és az adatrészt továbbküldi a hálózati rétegnek

Átvitel hiba nélkül és hibával

8



(a) Error-free transmission



(b) Transmission with losses and errors

Szimplex megáll-és-vár protokoll (stop-and-wait protocol)

A környezet

- ❑ mind az adó, mind a vevő hálózati rétegei mindenkorban 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 vagy veszíti el a kereteket;

A protokoll

- ❑ résztvevők: küldő és vevő;
- ❑ küldő egyesével küldi kereteket és addig nem küld újat, még nem kap nyugtát a vevőtől;
- ❑ 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

Következmény: fél-duplex csatorna kell.

Szimplex protokoll zajos csatornához

A környezet

- mind az adó, mind a vevő hálózati rétegei mindenkorban 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

- résztvevők: *küldő* és *vevő*;
- 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,
 - ha nincs hiba, az adatrészt továbbküldi a hálózati rétegnek, végül nyugtázza a keretet;
 - Ha hiba van, akkor eldobja a keretet és nem nyugtáz.

Következmény: duplikátumok lehetnek.

Megáll-és-vár

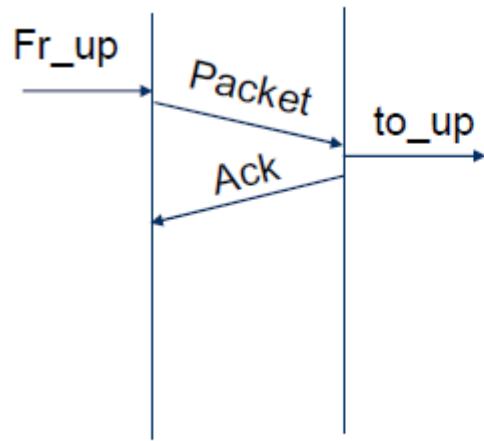
11

Egyszerű de nem hatékony
nagy távolságok és nagy
sebességű hálózat esetén.

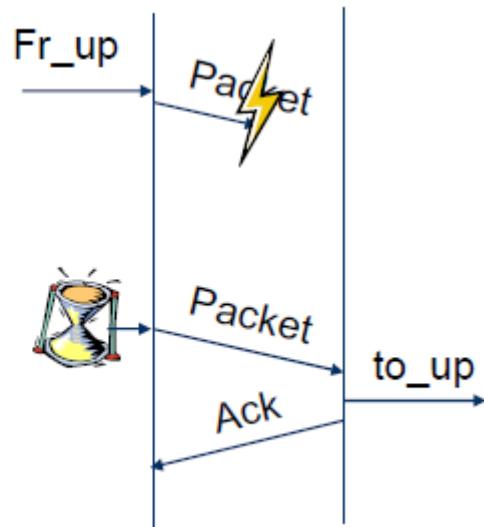
Küldhetnénk egymás után
folyamatosan???

Mi is a probléma?

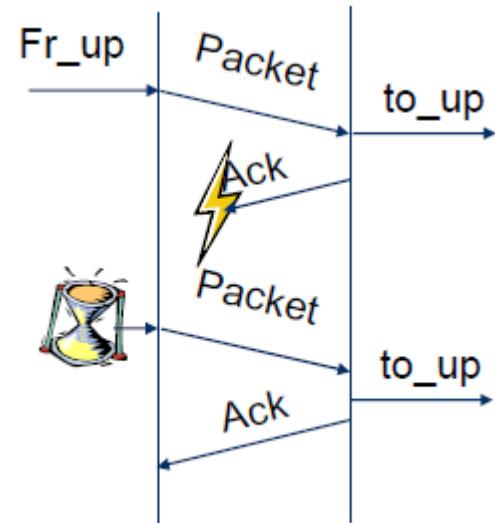
Általában



Csomagvesztés esetén

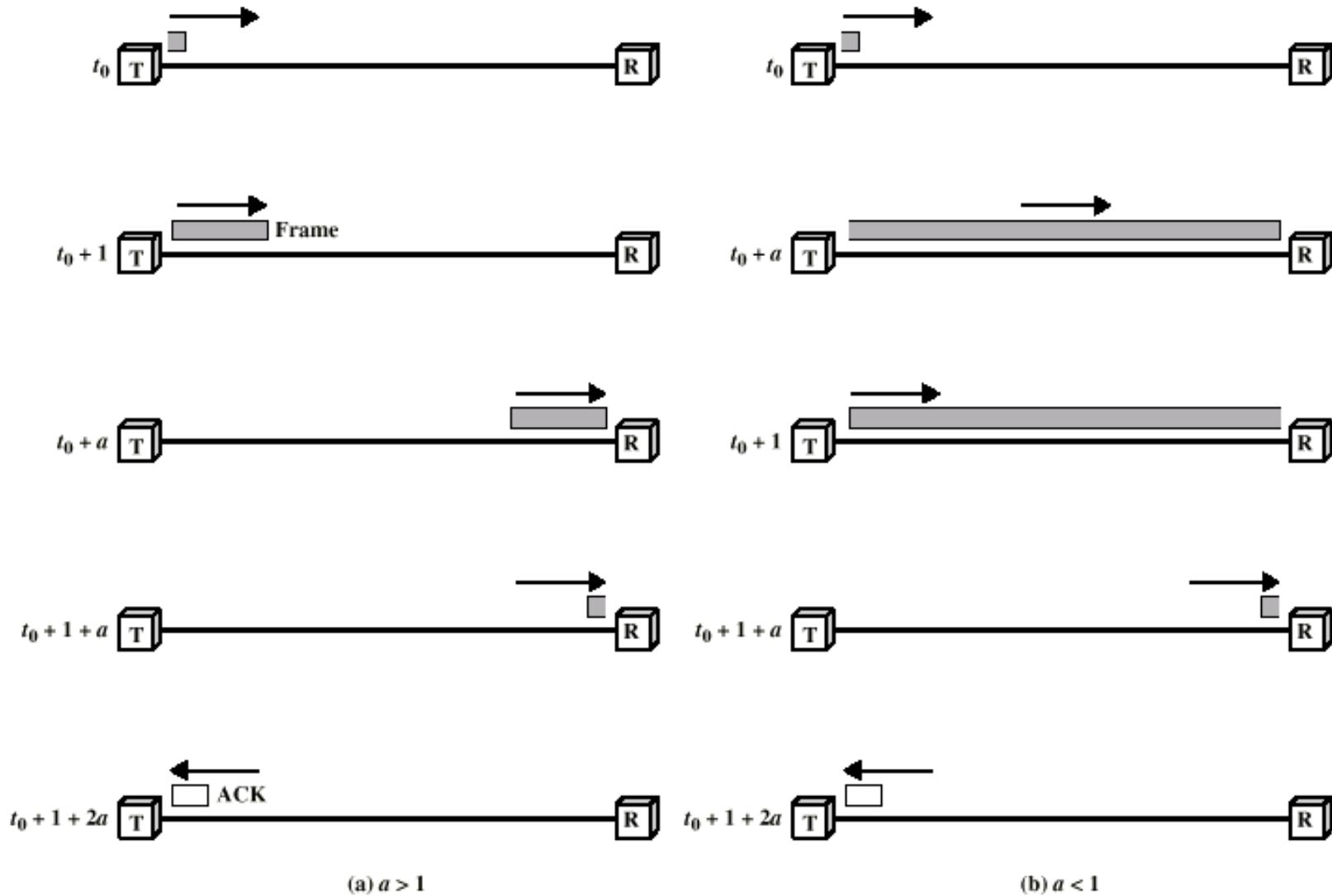


ACK vesztés esetén



Csatorna kihasználtság

13



Alternáló-bit protokoll (ABP)

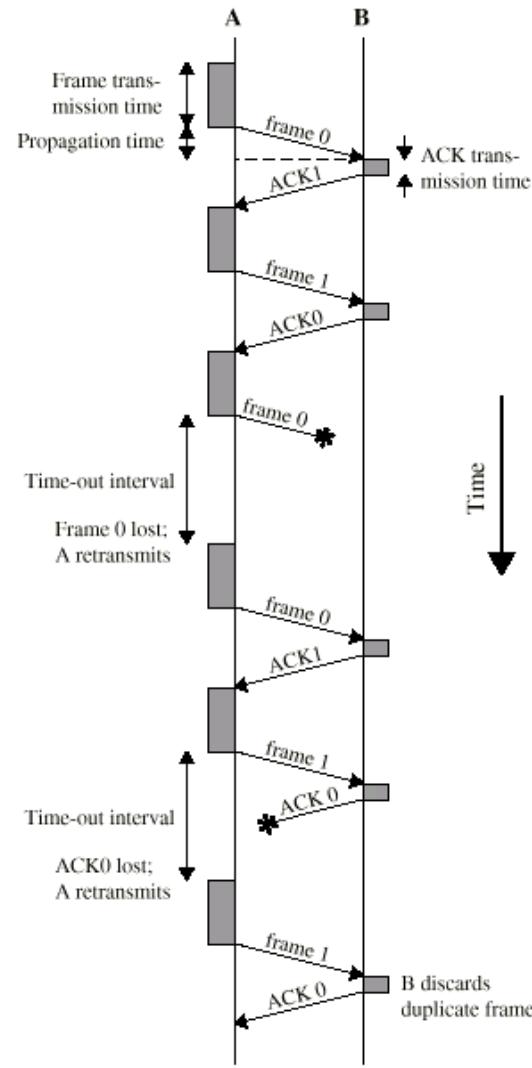
- **Megoldás:** sorszámok használata
- Mennyi sorszámra lesz szükség? {0,1} elegendő

A protokoll (ARQ) – Alternáló-bit protokoll

- résztvevők: küldő és vevő;
- küldő egyesével küldi a sorszámmal ellátott kereteket (kezdetben 0-s sorszámmal) és addig nem küld újat, még nem kap nyugtát a vevőtől egy megadott határidőn belül:
 - ha a nyugta megérkezik a határidőn belül, akkor lépteti a sorszámot mod 2 és küldi a következő sorszámmal ellátott keretet;
 - ha a határidő lejár, akkor ismételten elküldi az aktuális sorsszámmal ellátott keretet;
- a vevő kezdetben várakozik az első keret megérkezésére 0-s sorszámmal, keret érkezésekor a hardver puffer tartalmát változóba teszi, leellenőrzi a kontroll összeget és a sorszámot
 - ha nincs hiba, az adatrészt továbbküldi a hálózati rétegnek, végül nyugtázza a keretet és lépteti a sorszámat mod 2;
 - ha hiba van, akkor eldobja a keretet és nem nyugtáz.

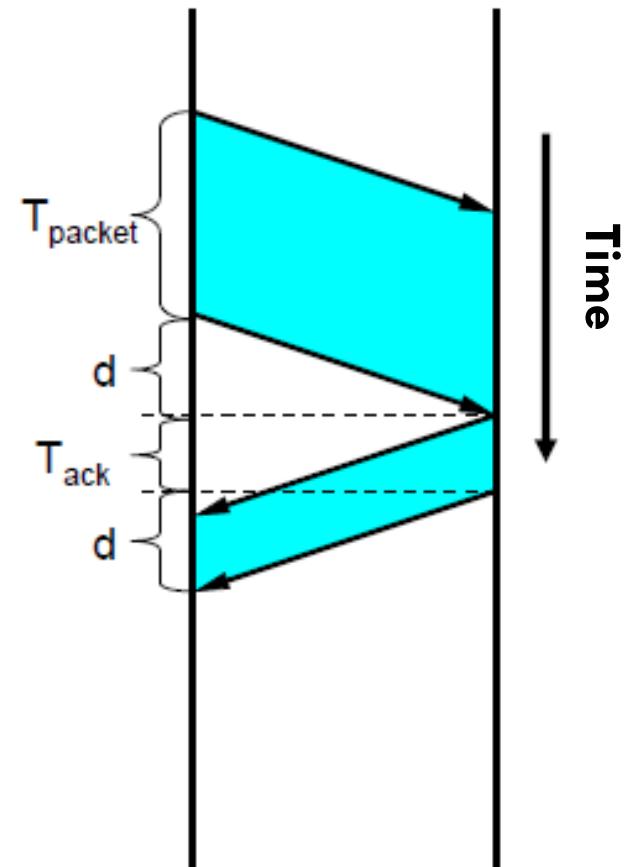
ABP

15



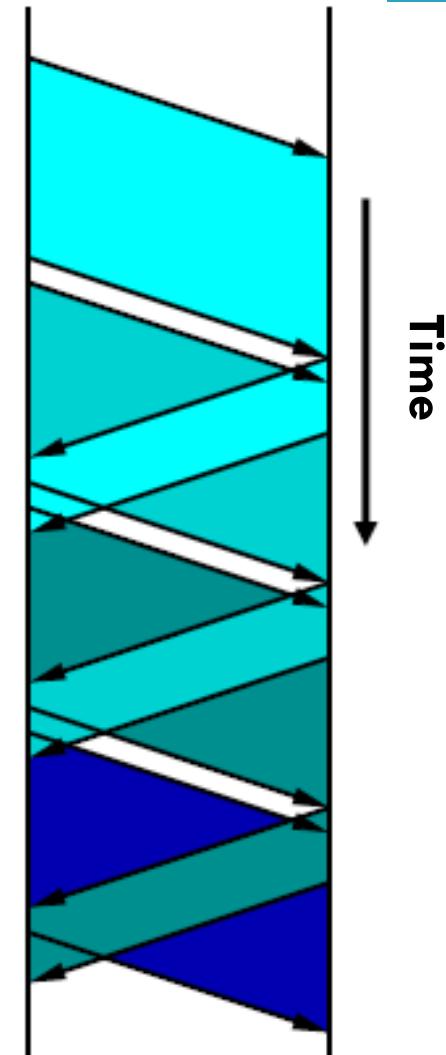
ABP – Csatorna kihasználtság

- Kihasználtság (η) a következő két elem aránya
 - A csomag elküldéséhez szükséges idő (T_{packet})
 - Az idő, ami a következő keret küldéséig eltelik
 - Az ábrán: $(T_{\text{packet}} + d + T_{\text{ack}} + d)$
- ABP esetén:
 - $\eta = T_{\text{packet}} / (T_{\text{packet}} + d + T_{\text{ack}} + d)$
- Nagy propagációs idő esetén az ABP nem hatékony



Hogyan javítsunk a hatékonyságon?

- A küldők egymás után küldik a kereteket
 - Több keretet is kiküldünk, nyugta megvárása nélkül.
 - Pipeline technika
- ABP kiterjesztése
 - Sorszámok bevezetésével



Csúszó-ablak protokollok 1/2

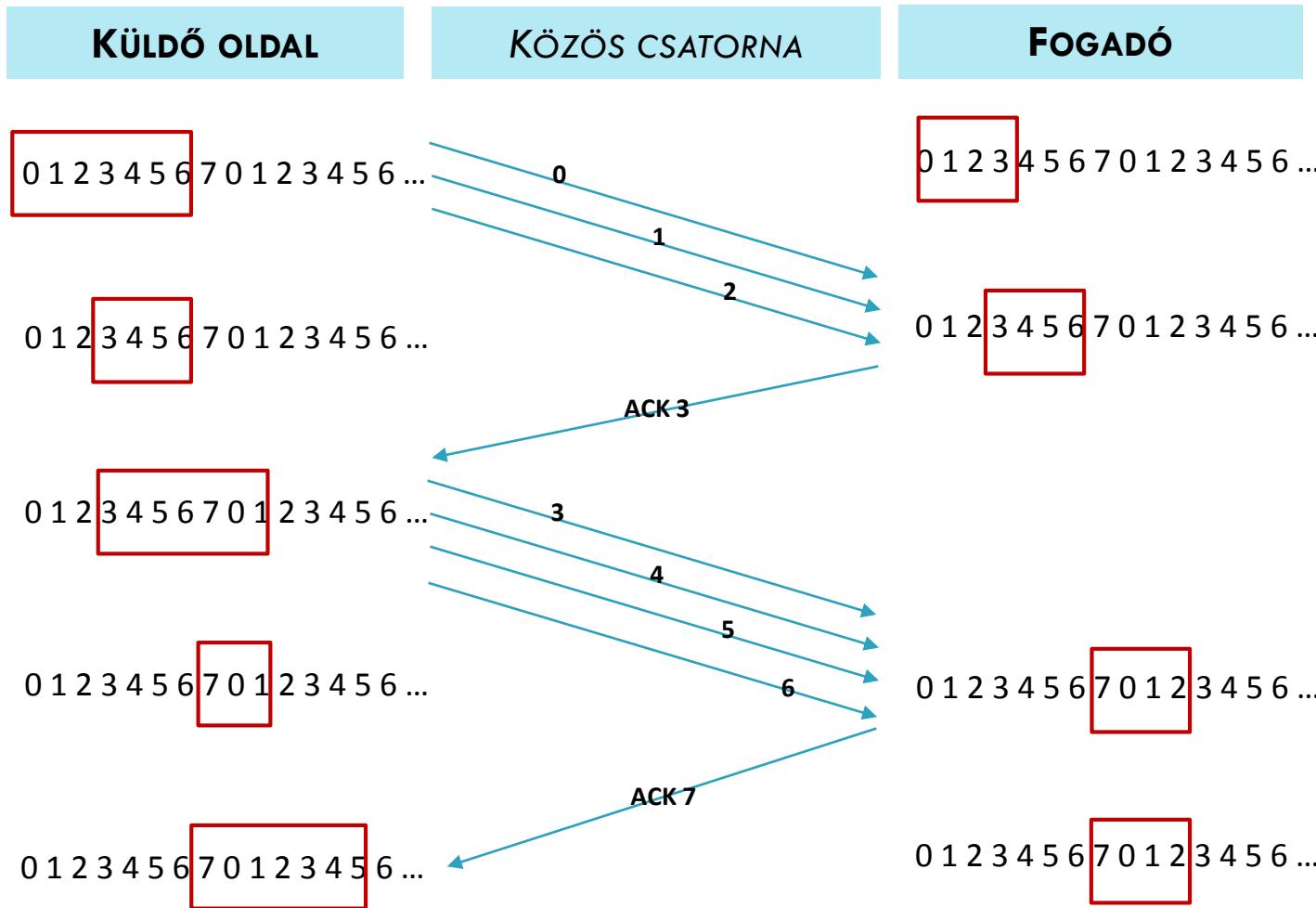
ALAPOK (ÁLTALÁNOS)

- 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, nyugtáztatlan keretet 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.

Példa 3-bites csúszó-ablak protokollra



Csúszó-ablak protokollok 2/2

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: $[0..2^K - 1]$.
- A adási ablak minden küldéssel szűkül, illetve nő egy nyugta érkezésével.

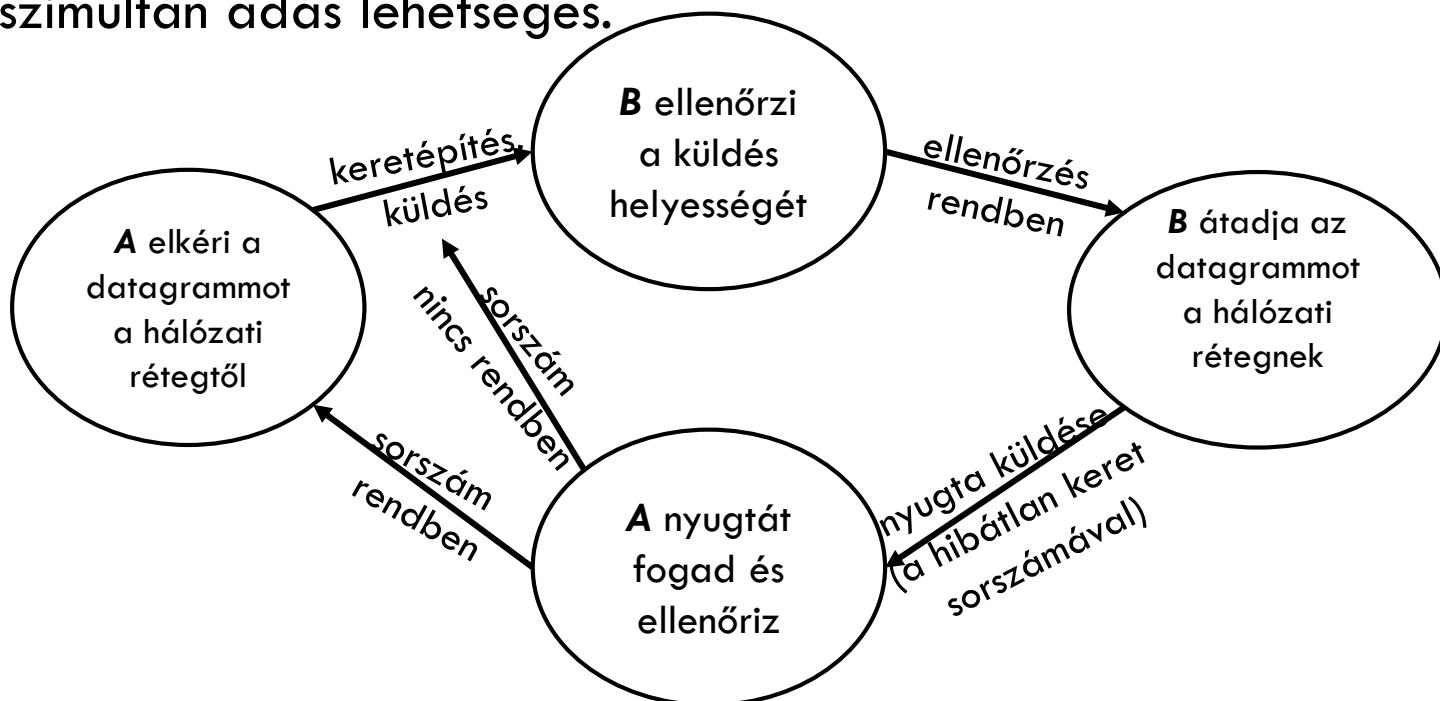
JELLEMZŐK (GYAKORLATI ALKALMAZÁS ESETÉN)

- 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);

Egybites csúszó-ablak protokoll állapotátmenetei

KÖRNYEZET

- A maximális ablak méret legyen 1.
- *Emlékeztetőül:* két irányú adatforgalom lehetséges, azaz szimultán adás lehetséges.



Pipelining

- Eddig feltételeztük, hogy a keret vevőhöz való megérkezéséhez és a nyugta visszaérkezéséhez együttesen szükséges idő elhanyagolható.
 - ▣ a nagy RTT a sávszélesség kihasználtságra hatással lehet
 - ▣ **Ötlet:** egyszerre több keret küldése
 - ▣ Ha az adatsebesség és az RTT szorzata nagy, akkor érdemes nagyméretű adási ablakot használni. (*pipelining*)
- Mi van ha egy hosszú folyam közepén történik egy keret hiba?
 1. „visszalépés N-nel”, avagy angolul *go-back-n*
 2. „szelektív ismétlés”, avagy angolul *selective-repeat*

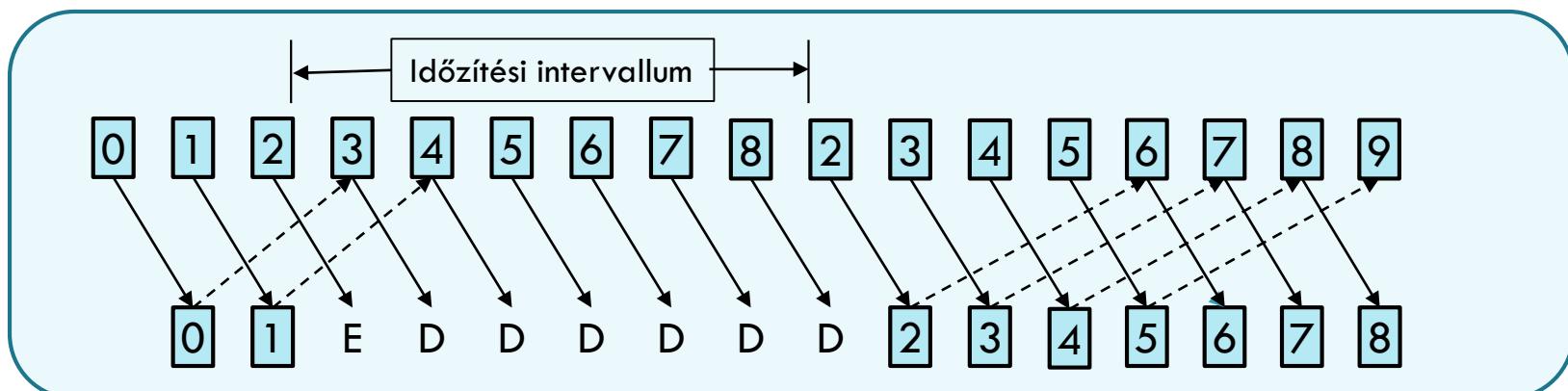
„visszalépés N-nel” stratégia

Stratégia lényege

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

Következmények

- Egy méretű vételi ablakot feltételezünk.
- Nagy sávszélességet pazarolhat el, ha nagy a hibaarány.



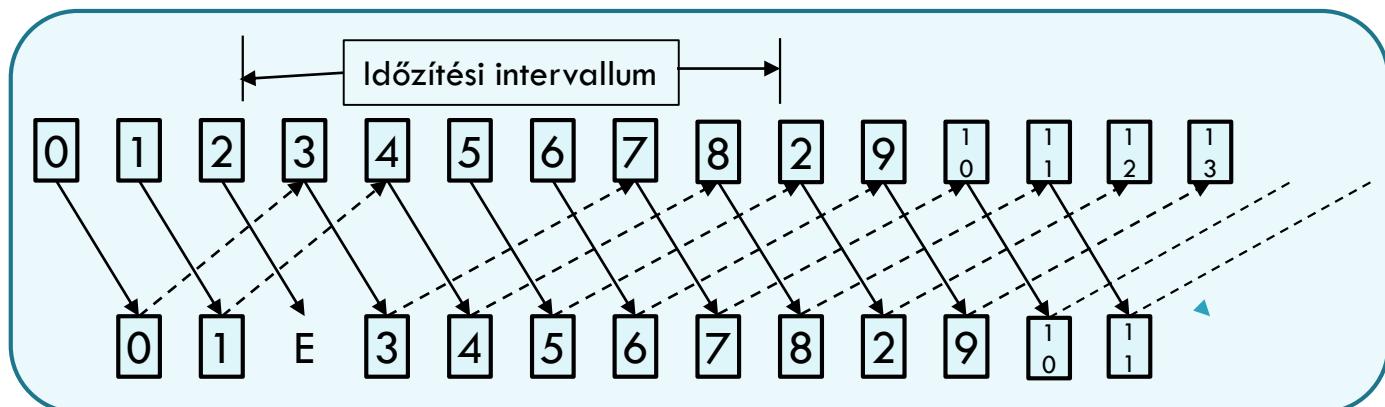
„szelektív ismétlés” stratégia

Stratégia lényege

- A hibás kereteket eldobja, de a jó kereteket a hibás után puffereli.
- Mikor az adónak lejár az időzítője, akkor a legrégebbi nyugtáztatlan 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.



Ethernet keret

Közeg hozzáférés vezérlése

Media Access Control (MAC)

Mi az a közeg hozzáférés ?

27

- Ethernet és a Wifi is többszörös hozzáférést biztosító technológiák
 - ▣ Az átviteli közegen több résztvevő osztozik
 - Adatszórás (broadcasting)
 - ▣ Az egyidejű átvitel **ütközést** okot
 - Lényegében meghiúsítja az átvitelt
- Követelmények a Media Access Control (MAC) protokolljaival szemben
 - ▣ Szabályok a közeg megosztására
 - ▣ Stratégiák az ütközések detektálásához, elkerüléséhez és feloldásához

MAC alréteg

28

- Eddigi tárgyalásaink során pont-pont összeköttetést feltételeztünk.
- Most az adatszóró csatornát (angolul *broadcast channel*) használó hálózatok tárgykörével foglalkozunk majd.
 - **Kulcskérdés:** Melyik állomás kapja a csatornahasználat jogát?
 - A csatorna kiosztás történhet:
 1. statikus módon (FDM, TDM)
 2. dinamikus módon
 - a) verseny vagy ütközés alapú protokollok (ALOHA, CSMA, CSMA/CD)
 - b) verseny-mentes protokollok (bittérkép-alapú protokollok, bináris visszaszámítás)
 - c) korlátozott verseny protokollok (adaptív fa protokollok)

Statikus csatornakiosztás

29

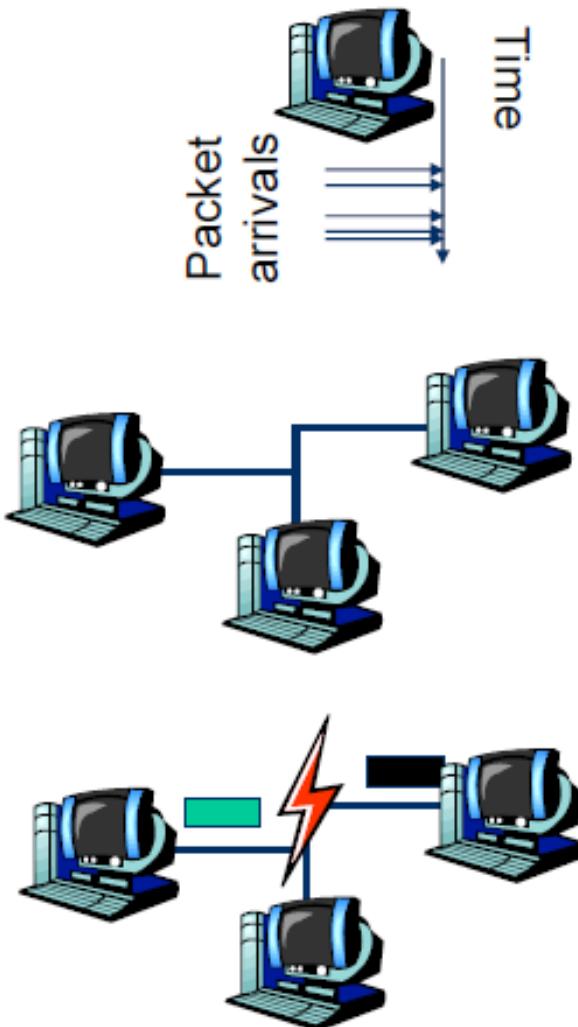
Frekvenciaosztásos nyálabolás

- N darab felhasználót feltételezünk, a sávszélet N egyenlő méretű sávra osztják, és minden egyes sávhoz hozzárendelnek egy felhasználót.
- Következésképpen az állomások nem fogják egymást zavarni.
- Előnyös a használata, ha fix számú felhasználó van és a felhasználók nagy forgalmi igényt támasztanak.
- Löketszerű forgalom esetén használata problémás.

Időosztásos nyálabolás

- N darab felhasználót feltételezünk, az időegységet N egyenlő méretű időrésre – úgynevezett *slot*-ra – osztják, és minden egyes réshez hozzárendelnek egy felhasználót.
- Löketszerű forgalom esetén használata nem hatékony.

Dinamikus csatornakiosztás



1. Állomás modell

- N terminál/állomás
- Annak a valószínűsége, hogy Δt idő alatt csomag érkezik $\lambda\Delta t$, ahol λ az érkezési folyam rátája.

2. Egyetlen csatorna feltételezés

- minden állomás egyenrangú.
- minden kommunikáció egyazon csatornán zajlik.
- minden állomás tud ezen küldeni és fogadni csomagot.

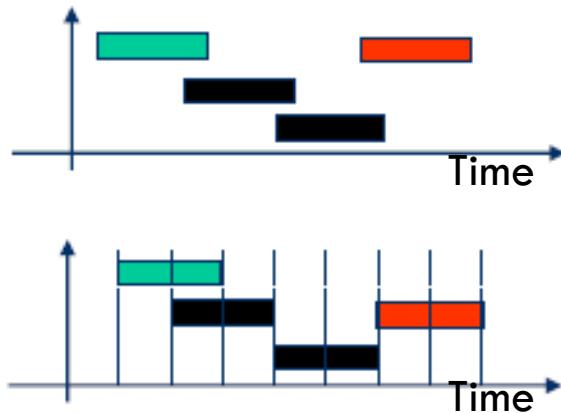
3. Ütközés feltételezés

- ha két keret egy időben kerül átvitelre, akkor átlapolódnak, és az eredményül kapott jel értelmezhetetlenné válik.
- ezt nevezzük ütközésnek.

4. Folytonos időmodell VS diszkrét időmodell

5. Vivőjel értékelés VS nincs vivőjel érzékelés

Dinamikus csatornakiosztás



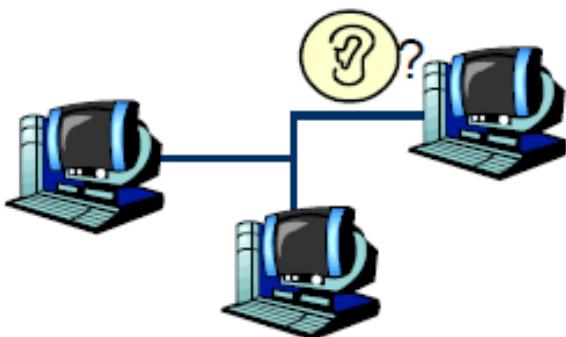
Használt időmodell

Kétféle időmodellt különböztetünk meg:

- a) **Folytonos** – Mindegyik állomás tetszőleges időpontban megkezdheti a küldésre kész keretnek sugárzását.
- b) **Diszkrét** – Az időt diszkrét részekre osztjuk. Keret továbbítás csak időrés elején lehetséges. Az időrész lehet üres, sikeres vagy ütközéses.

Vivőjel érzékelési képesség

Az egyes állomások vagy rendelkeznek ezzel a tulajdonsággal vagy nem.



- a) Ha **nincs**, akkor az állomások nem tudják megvizsgálni a közös csatorna állapotát, ezért egyszerűen elkezdenek küldeni, ha van rá lehetőségük.
- b) Ha **van**, akkor 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.

Megjegyzés: Ez egy egyszerűsített modell!

Hogyan mérjük a hatékonyságot?

□ Átvitel [Throughput] (S)

- A sikeresen átvitt csomagok/keretek száma egy időegység alatt

□ Késleltetés [Delay]

- Egy csomag átviteléhez szükséges idő

□ Fairség [Fairness]

- minden állomás egyenrangúként van kezelve

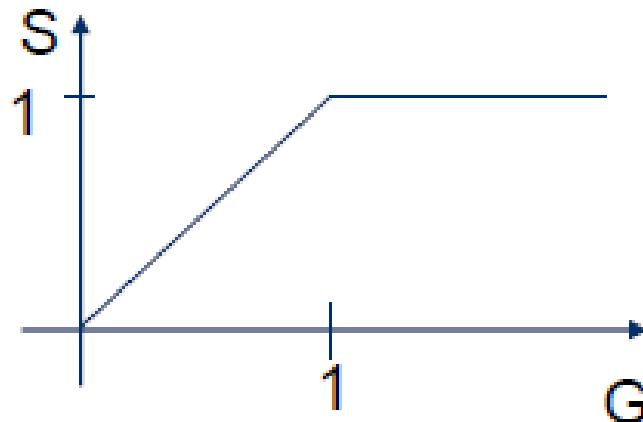
Átvitel és terhelés

□ Terhelés (G)

- A protokoll által kezelendő csomagok száma egy időegység alatt (beérkező kérések)
- $G > 1$: túlterhelés
- A csatorna egy kérést tud elvezetni

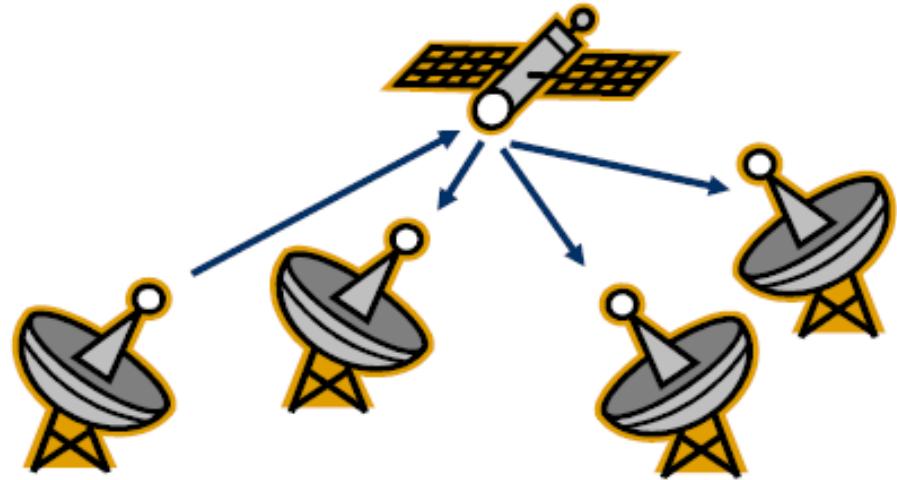
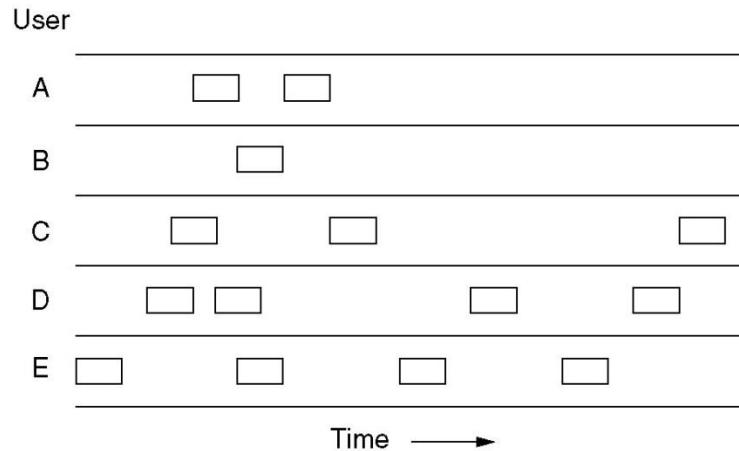
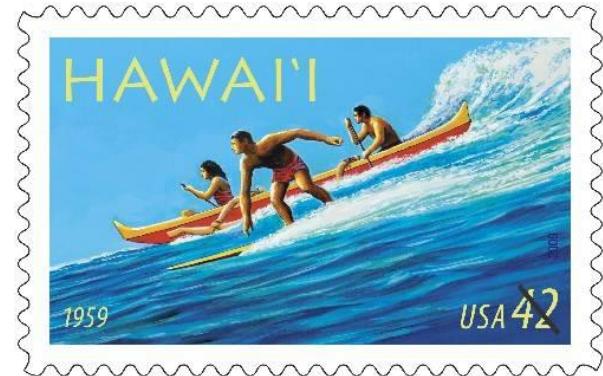
□ Ideális esetben

- Ha $G < 1$, $S = G$
- Ha $G \geq 1$, $S = 1$
- Ahol egy csomag kiküldése egy időegységet vesz igénybe.



(Tiszta) ALOHA

- Az algoritmust a 70-es években a Uni. of Hawaii fejlesztette
 - **Ha van elküldendő adat, akkor elküldi**
 - Alacsony költségű, nagyon egyszerű megoldás



ALOHA

35

□ Topológia: broadcast rádió több állomással

□ Protokoll:

- Az állomások azonnal küldenek
- A fogadók minden csomagot nyugtáznak
- Nincs nyugta = ütközés, véletlen ideig vár, majd újraküld

- Egyszerű, de radikális megoldás
- Korábbi megoldások, mind felosztották a csatornát
 - TDMA, FDMA, etc.
- Kévés küldő esetére készült

Teljesítmény elemzés -Poisson Folyam

- A „véletlen érkezések” egyik ünnepelt modellje a sorban-állás elméletben a Poisson folyam.
- A modell feltételezései:
 - Egy érkezés valószínűsége egy rövid Δt intervallum alatt arányos az intervallum hosszával és nem függ az intervallum kezdetétől (ezt nevezzük memória nélküli tulajdonságnak)
 - Annak a valószínűsége, hogy több érkezés történik egy rövid Δt intervallum alatt közelít a nullához.

Teljesítmény elemzés –Poisson eloszlás

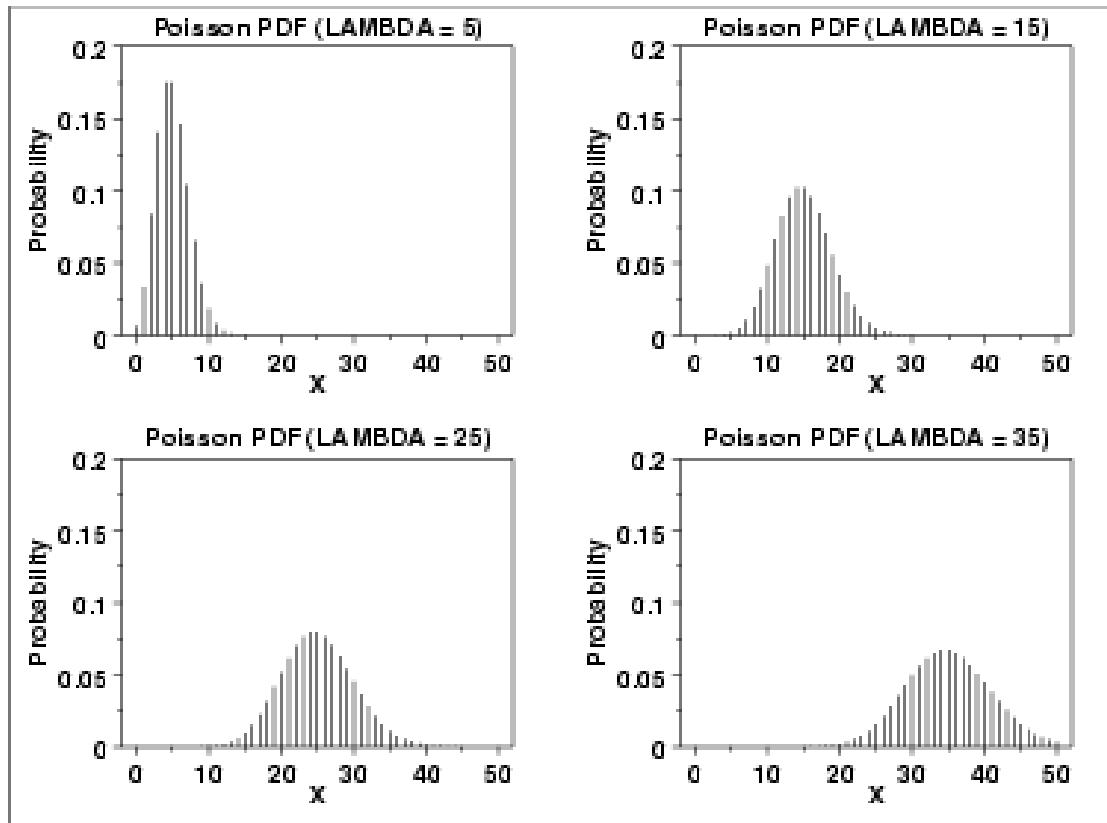
Annak a valószínűsége, hogy k érkezés történik egy t hosszú intervallum során:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

ahol λ az érkezési ráta. Azaz ez egy egy-paraméteres modell, ahol csak λ -át kell ismernünk.

Poisson Eloszlás példák

38



ALOHA vizsgálata

- Jelölés:
 - T_f = keret-idő (feldolgozási, átviteli és propagációs)
 - S : A sikeres keret átvitelek átlagos száma T_f idő alatt; (*throughput*)
 - G : T_f idő alatti összes átviteli kísérletek átlagos száma
 - D : Egy keret küldésre kész állapota és a sikeres átvitele között eltelt átlagos idő
- Feltételezések
 - minden keret konstans/azonos méretű
 - A csatorna zajmentes, hibák csak ütközések miatt történnek
 - A keretek nem kerülnek sorokba az egyedi állomásokon
 - Egy csatorna egy Poisson folyamként viselkedik

ALOHA vizsgálata

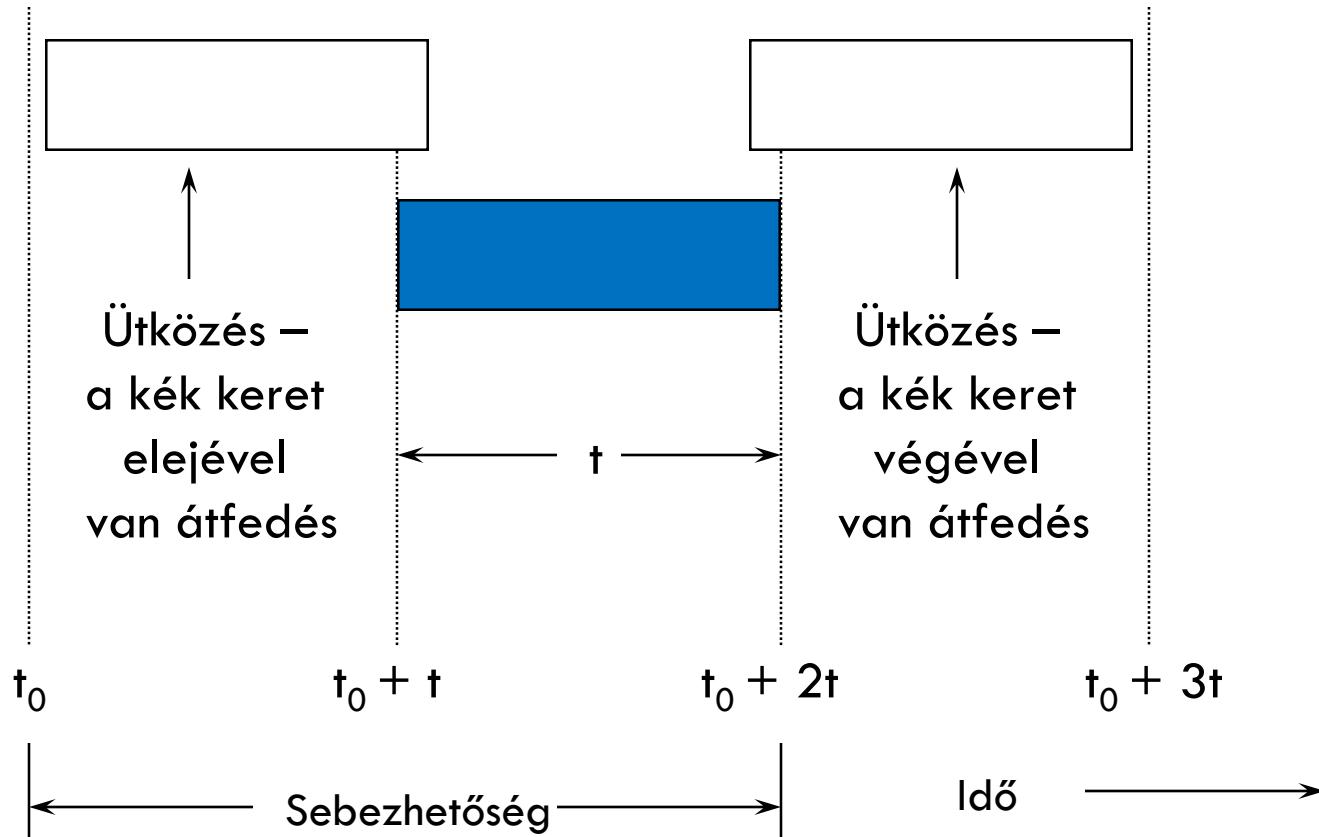
- Mivel S jelöli a „jó” átviteleket egy keret idő alatt és G jelöli az összes átviteli kísérletet egy keret idő alatt, így a következő összefüggést írhatjuk:

$$S = S(G) = G \times (\text{A „jó” átvitelek valószínűsége})$$

- A sebezhetőségi idő egy sikeres átviteléhez: $2T_f$
- Azaz a „jó” átvitel valószínűsége megegyezik annak a valószínűségével, hogy a sebezhetőségi idő alatt **nincs** beérkező keret.

ALOHA vizsgálata

41



Sebezhetőségi időintervallum a kékkel jelölt kerethez

ALOHA vizsgálata

Tudjuk, hogy:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

Azaz most $t = 2T_f$ és $k = 0$ (t legyen a seb. Idő, k=0, hogy ne érkezzen új keret a kék küldése során)

$$P_0(2T_f) = \frac{(\lambda \cdot 2T_f)^0 e^{-\lambda 2T_f}}{0!} = e^{-2G}$$

because $\lambda = \frac{G}{T_f}$. Thus, $S = G \cdot e^{-2G}$

ALOHA vizsgálata

43

- $S(G) = Ge^{-2G}$ függvényt G szerint deriválva és az eredményt nullának tekintve az egyenlet megoldásával megkapjuk a maximális sikeres átvitelhez tartozó G értéket:

$$G = 0.5,$$

melyre $S(G) = 1/2e = 0.18$. Azaz a maximális throughput csak 18%-a a teljes kapacitásnak!!!

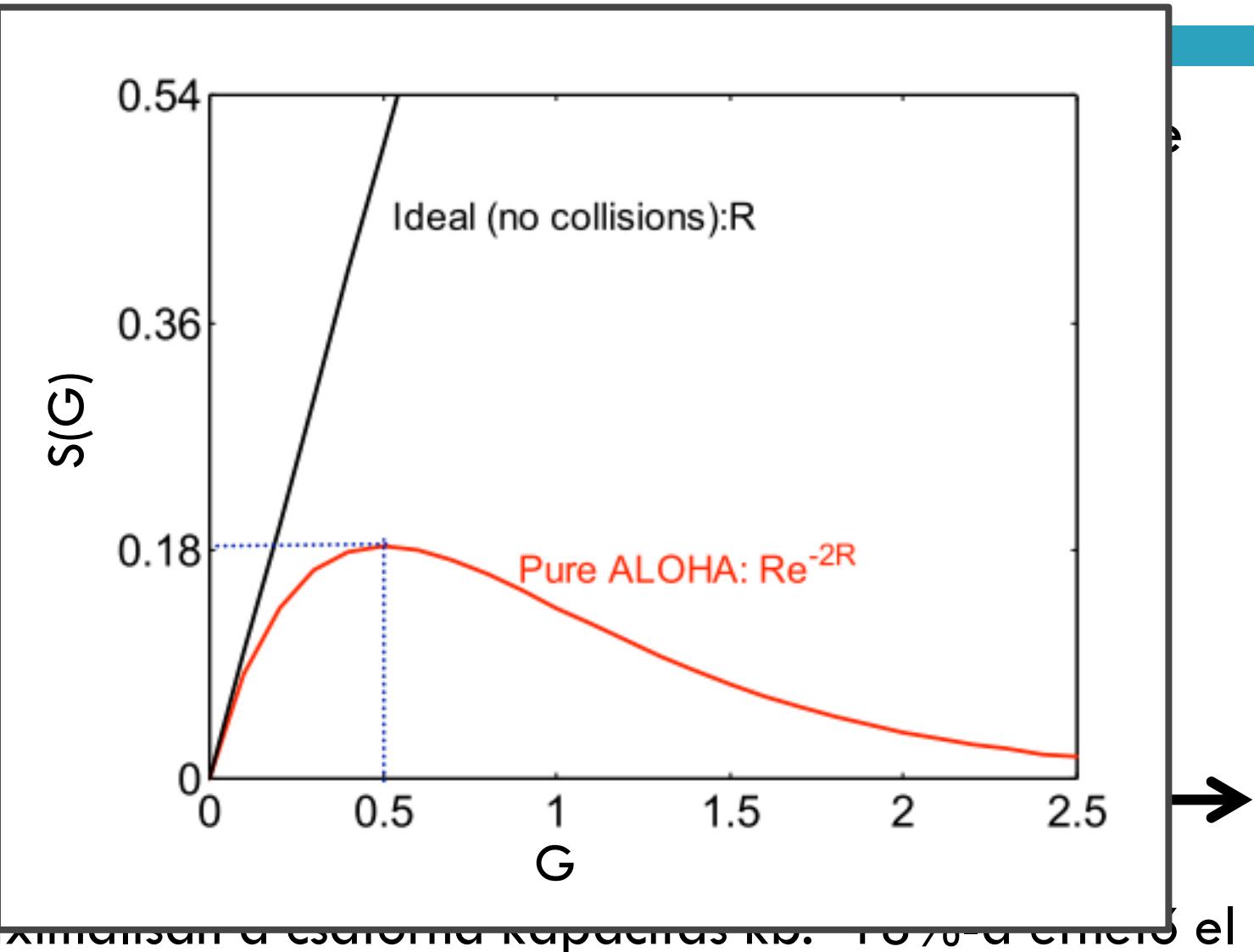
ALOHA vs TDMA

44

- A TD
- A v
- Az A
- Sol
- De

Senden
Senden

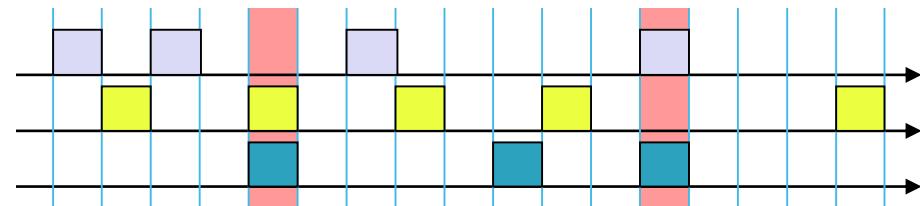
Maximierung der
Sekundärkapazität



Réselt ALOHA

45

- A csatornát azonos időrésekre bontjuk, melyek hossza pont egy keret átviteléhez szükséges idő.
- Átvitel csak az időrések határán lehetséges



- Algoritmus:
 - Amikor egy új A keret küldésre kész:
 - Az A keret kiküldésre kerül a (következő) időrés-határon

A réselt ALOHA vizsgálata

- A sebezhetőségi idő a felére csökken!!!
- Tudjuk, hogy:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

Ez esetben $t = T_f$ és továbbra is $k = 0$, amiből kapjuk, hogy:

$$P_0(T_f) = \frac{(\lambda \cdot T_f)^0 e^{-\lambda T_f}}{0!} = e^{-G}$$

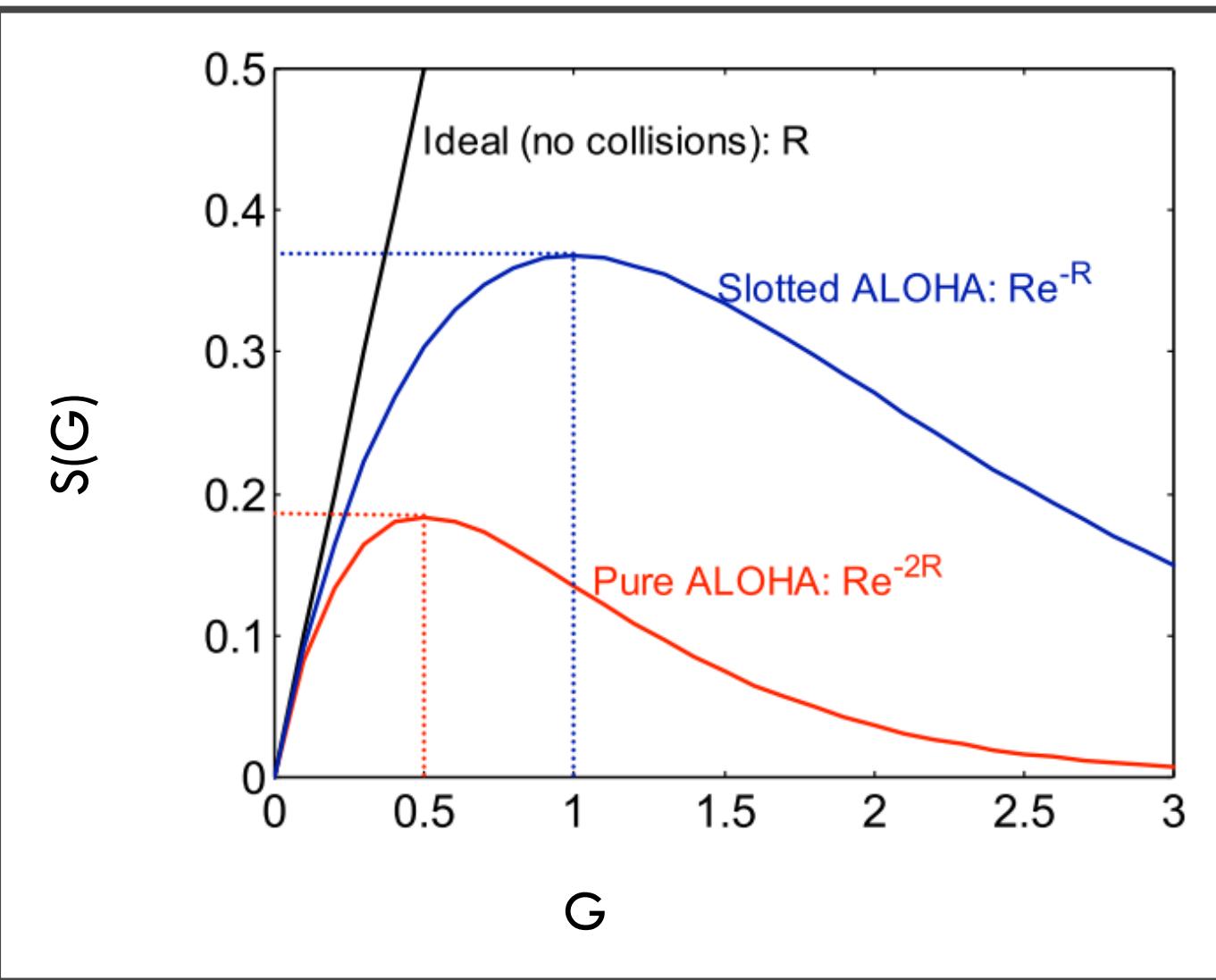
because $\lambda = \frac{G}{T_f}$. Thus, $S = G \cdot e^{-G}$

Réselt ALOHA

47

- Protokoll
- Ugatás
- Réselt ALOHA
- Azaz nem
- 37
- Az kezdeti

általán
órával



Köszönöm a figyelmet!

Számítógépes Hálózatok

6. Előadás: **Adatkapcsolati réteg**
 Hálózati réteg

Adatkapcsolati réteg

2



- Szolgáltatás
 - Adatok keretekre tördelése: határok a csomagok között
 - Közeghozzáférés vezérlés (MAC)
 - Per-hop megbízhatóság és folyamvezérlés
- Interfész
 - Keret küldése két közös médiumra kötött eszköz között
- Protokoll
 - Fizikai címzés (pl. MAC address, IP address)
- Példák: Ethernet, Wifi, InfiniBand

Adatkapcsolati réteg

3



□ Funkciók:

- ❑ Adat blokkok (**keretek/frames**) küldése eszközök között
- ❑ A fizikai közeghez való hozzáférés szabályozása

□ Legfőbb kihívások:

- ❑ Hogyan **keretezzük** az adatokat?
- ❑ Hogyan ismerjük fel a **hibát**?
- ❑ Hogyan vezéreljük a **közeghozzáférést (MAC)**?
- ❑ Hogyan oldjuk fel vagy előzzük meg az **ütközési** helyzeteket?

Közeg hozzáférés vezérlése

Media Access Control (MAC)

Mi az a közeg hozzáférés ?

5

- Ethernet és a Wifi is többszörös hozzáférést biztosító technológiák
 - Az átviteli közegen több résztvevő osztozik
 - Adatszórás (broadcasting)
 - Az egyidejű átvitel **ütközést** okot
 - Lényegében meghiúsítja az átvitelt
- Követelmények a Media Access Control (MAC) protokolljaival szemben
 - Szabályok a közeg megosztására
 - Stratégiák az ütközések detektálásához, elkerüléséhez és feloldásához

MAC alréteg

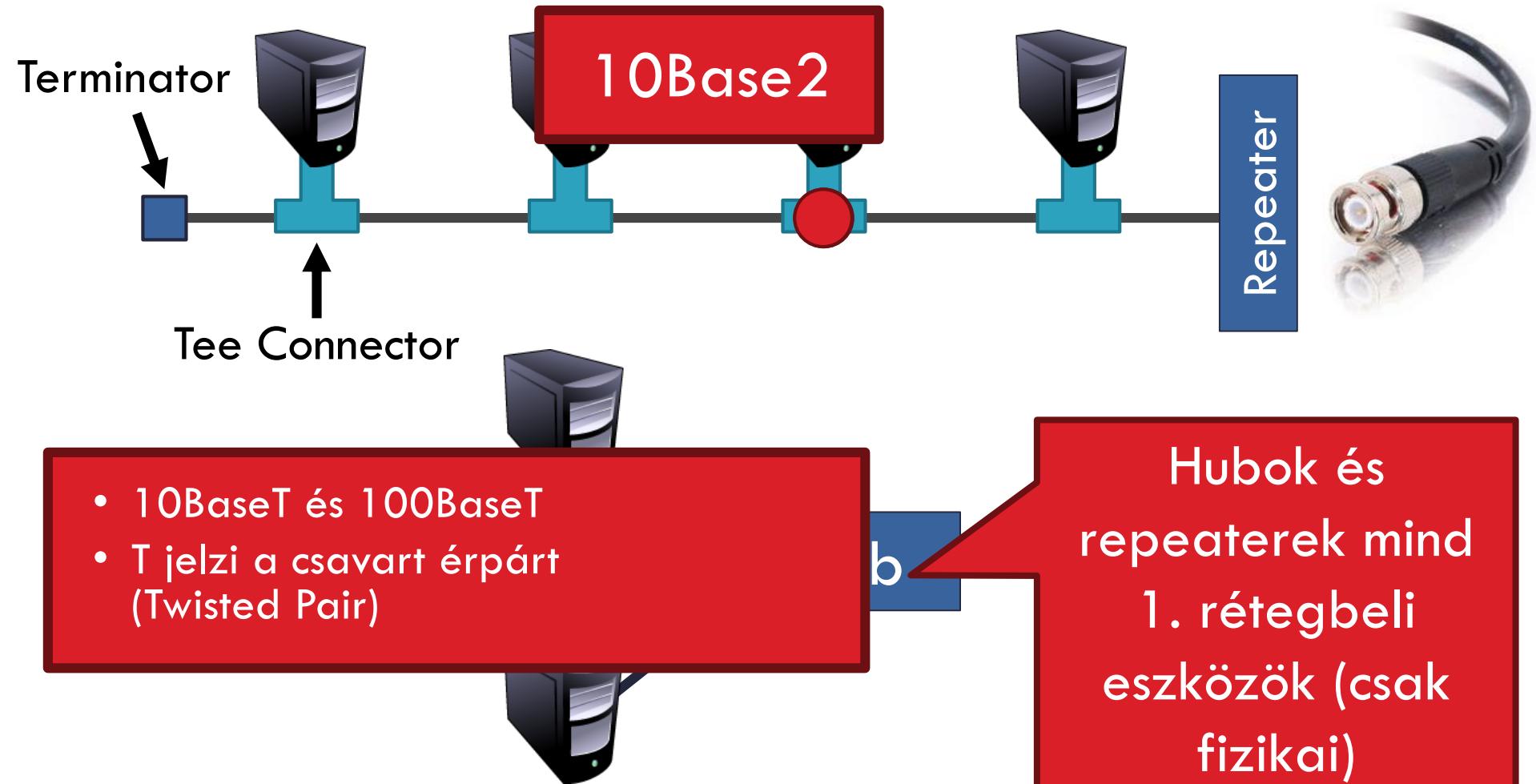
6

- Eddigi tárgyalásaink során pont-pont összeköttetést feltételeztünk.
- Most az adatszóró csatornát (angolul *broadcast channel*) használó hálózatok tárgykörével foglalkozunk majd.
 - **Kulcskérdés:** Melyik állomás kapja a csatornahasználat jogát?
 - A csatorna kiosztás történhet:
 1. statikus módon (FDM, TDM)
 2. dinamikus módon
 - a) verseny vagy ütközés alapú protokollok (ALOHA, CSMA, CSMA/CD)
 - b) verseny-mentes protokollok (bittérkép-alapú protokollok, bináris visszaszámítás)
 - c) korlátozott verseny protokollok (adaptív fa protokollok)

Adatszóró (Broadcast) Ethernet

7

- Eredetileg az Ethernet egy adatszóró technológia volt



Vivőjel érzékelés

Carrier Sense Multiple Access (CSMA)

- További feltételezés
 - minden állomás képes belehallgatni a csatornába és így el tudja dönteni, hogy azt más állomás használja-e átvitelre

1-perzisztens CSMA protokoll

9

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

Algoritmus

- 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.
- Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

Tulajdonságok

- A terjedési késleltetés nagymértékben befolyásolhatja a teljesítményét.
- Jobb teljesítményt mutat, mint az ALOHA protokollok.

Nem-perzisztens CSMA protokoll

10

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

Algoritmus

- 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.
- Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

Tulajdonságok

- Jobb teljesítményt mutat, mint az 1-perzisztens CSMA protokoll. (*intuitív*)

p-perzisztens CSMA protokoll

11

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

Algoritmus

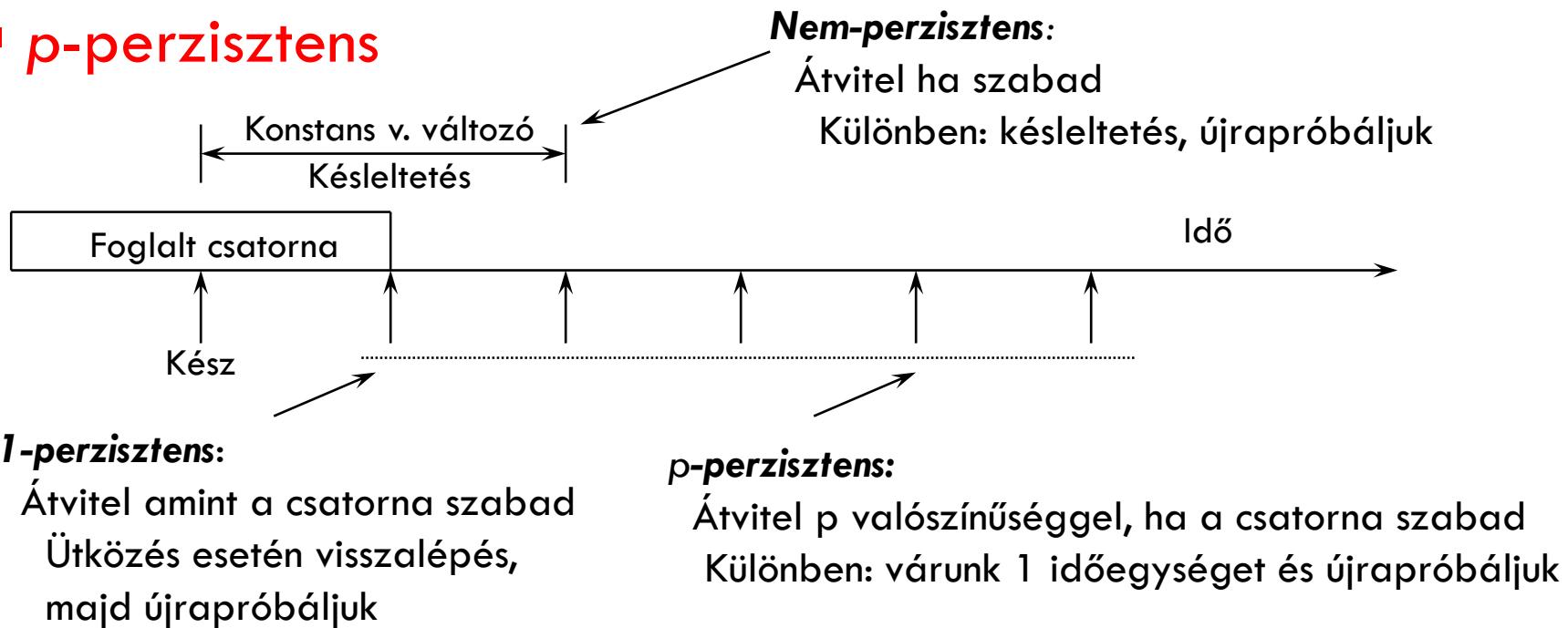
- Adás kész állapotban az állomás belehallgat a csatornába:
 - a) Ha foglalt, akkor vár a következő időrésig, majd megismétli az algoritmust.
 - b) Ha szabad, akkor p valószínűsséggel küld, illetve $1-p$ valószínűsséggel visszalép a szándékától a következő időrésig. Várakozás esetén a következő időrésben 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.
- Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

CSMA áttekintés

12

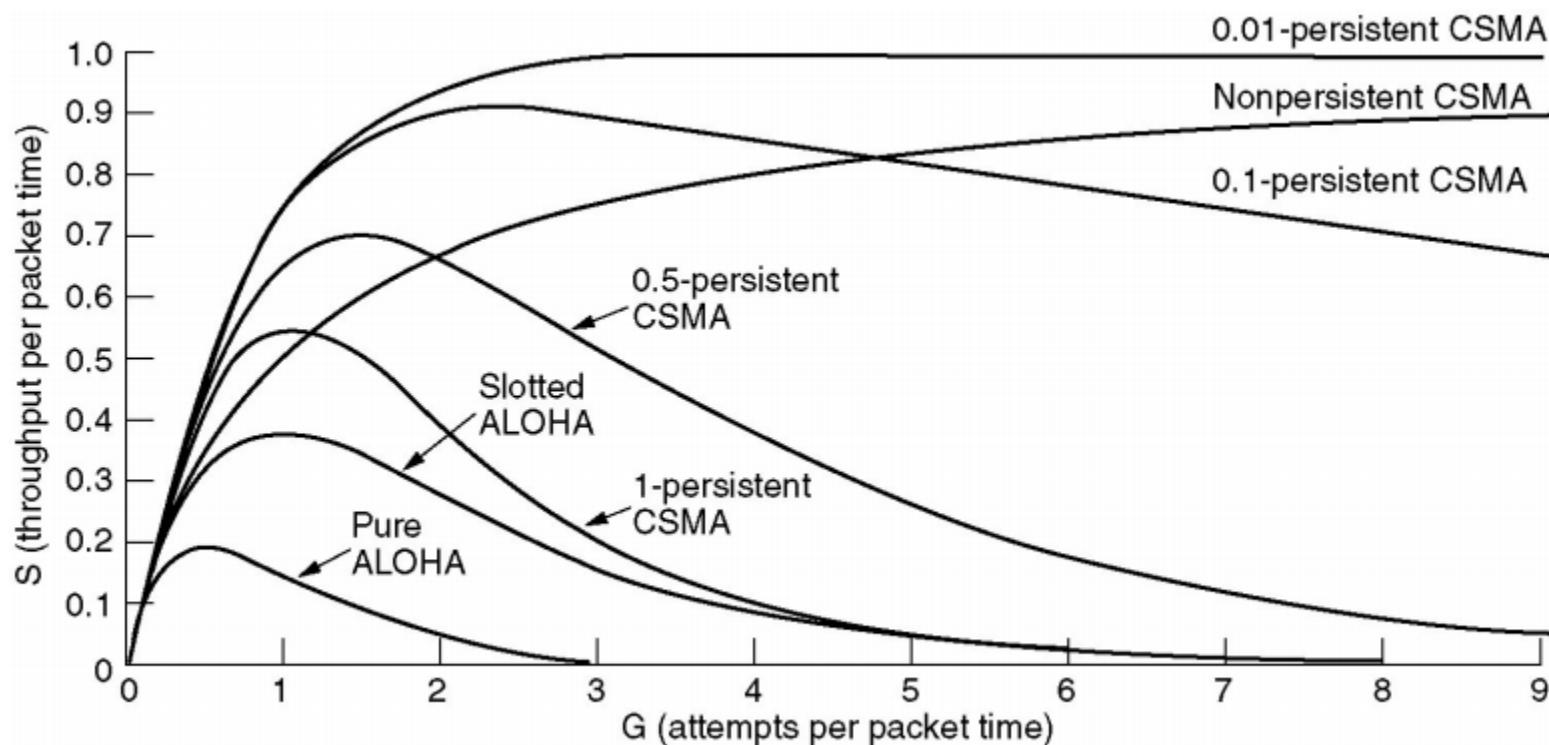
- Nem-perzisztens
- 1-perzisztens
- p -perzisztens

CSMA perzisztencia



CSMA és ALOHA protokollok összehasonlítása

13



CSMA/CD - CSMA ütközés detektálással (CD = Collision Detection)

- Ütközés érzékelés esetén meg lehessen szakítani az adást.
("Collision Detection")
 - minden állomás küldés közben megfigyeli a csatornát,
 - ha ütközést tapasztal, akkor megszakítja az adást, és véletlen ideig várakozik, majd újra elkezdi leadni a keretét.
- Mikor lehet egy állomás biztos abban, hogy megszerezte magának a csatornát?
 - Az ütközés detektálás minimális ideje az az idő, ami egy jelnek a két legtávolabbi állomás közötti átviteléhez szükséges.

CSMA/CD

- Egy állomás megszerezte a csatornát, ha minden más állomás érzékeli az átvitelét.
- Az ütközés detektálás működéséhez szükséges a keretek hosszára egy alsó korlátot adnunk
- Ethernet a CSMA/CD-t használja

CSMA/CD

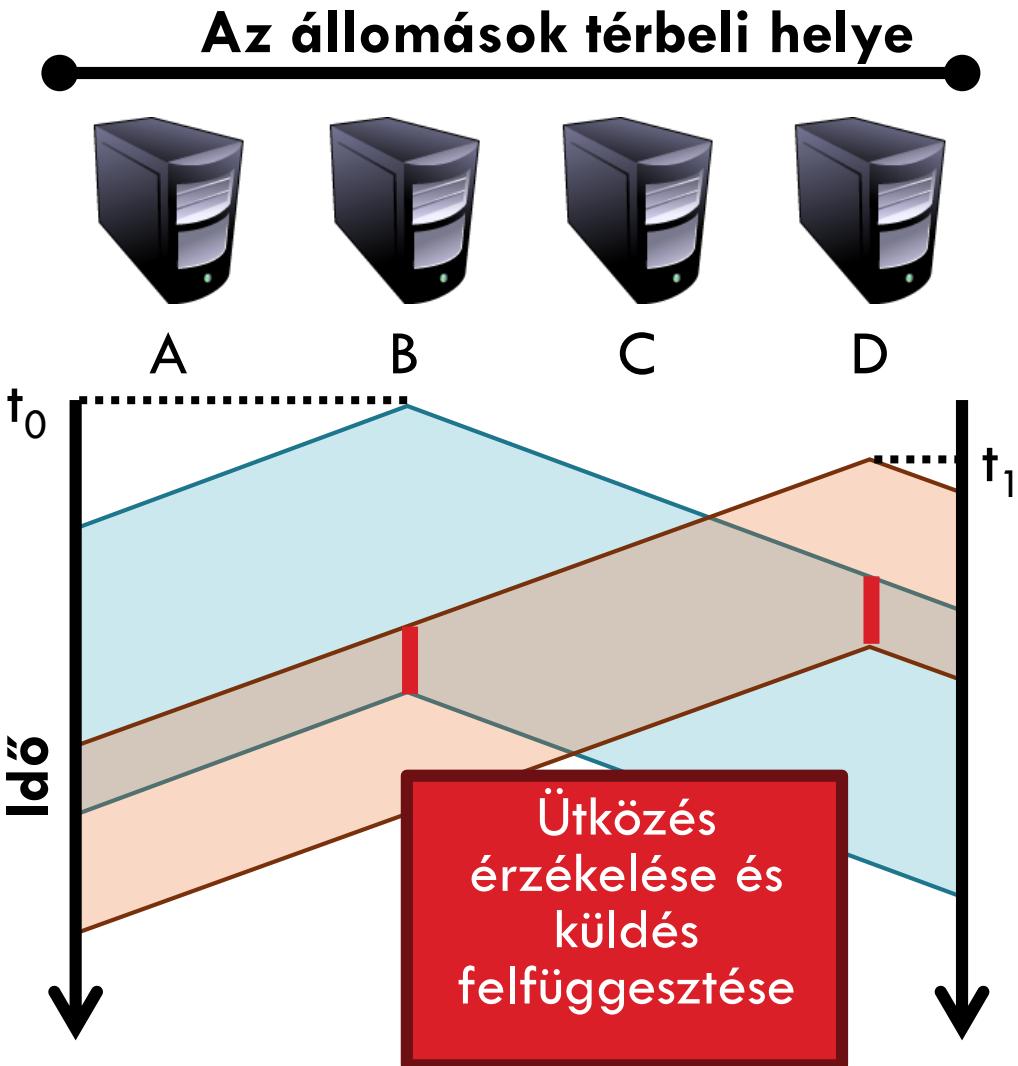
16

- Carrier sense multiple access with collision detection
- Alapvetés: a közeg lehetőséget ad a csatornába hallgatásra
- Algoritmus
 1. Használjuk valamely CSMA variánst
 2. A keret kiküldése után, figyeljük a közeget, hogy történik-e ütközés
 3. Ha nem volt ütközés, akkor a keretet leszállítottuk
 4. Ha ütközés történt, akkor azonnal megszakítjuk a küldést
 - Miért is folytatnánk hisz a keret már sérült...
 5. Alkalmazzuk az bináris exponenciális hátralék módszert az újraküldés során (binary exponential backoff)

CSMA/CD Ütközések

17

- Ütközések történhetnek
- Az ütközéseket gyorsan észleljük és felfüggesztjük az átvitelt
- Mi a szerepe a **távolságnak, propagációs időnek és a keret méretének?**



Binary Exponential Backoff – Bináris exponenciális hátralék

18

- Ütközés érzékelésekor a küldő egy ún. „jam” jelet küld
 - minden állomás tudomást szerezzen az ütközésről
- Binary exponential backoff működése:
 - Válasszunk egy $k \in [0, 2^n - 1]$ egyenletes eloszlás szerint, ahol $n =$ az ütközések száma
 - Várunk k időegységet (keretidőt) az újraküldésig
 - n felső határa 10, 16 sikertelen próbálkozás után pedig eldobjuk a keretet
- A hátralék idő versengési résekre van osztva

Binary Exponential Backoff

19

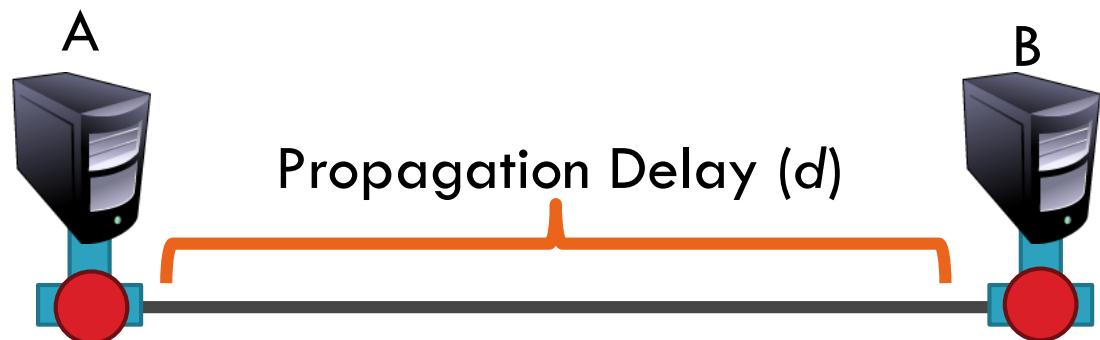
Tekintsünk két állomást, melyek üzenetei ütköztek

- Első ütközés után: válasszunk egyet a két időrés közül
 - A sikeres átvitel esélye az első ütközés után: 50%
 - Átlagos várakozási idő: 1,5 időrés
- Második ütközés után: válasszunk egyet a négy rés közül
 - Sikeres átvitel esélye ekkor: 75%
 - Átlagos várakozási idő: 2,5 rés
- Általában az m. ütközés után:
 - A sikeres átvitel esélye: $1 - 2^{-m}$
 - Average delay (in slots): $0,5 + 2^{(m-1)}$

Minimális keretméret

20

- Miért 64 bájt a minimális keretméret?
- Az állomásoknak elég időre van szüksége az ütközés detektálásához
- Mi a kapcsolat a keretméret és a kábelhossz között?
 1. t időpont: Az A állomás megkezdi az átvitelt
 2. $t + d$ időpont: A B állomás is megkezdi az átvitelt
 3. $t + 2*d$ időpont: A érzékeli az ütközést



Alapötlet: Az A állomásnak $2*d$ ideig kell küldenie!

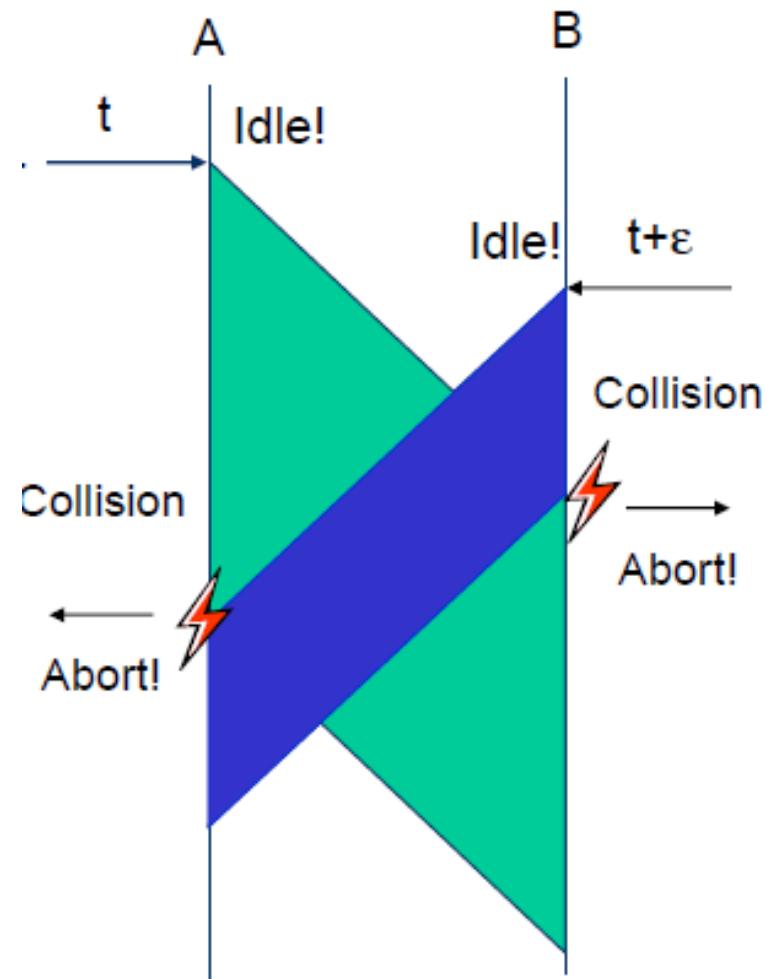
CSMA/CD

- CSMA/CD három állapota: versengés, átvitel és szabad.

- Ahhoz, hogy minden ütközést észleljünk szükséges:

$$T_f \geq 2T_{pg}$$

- ahol T_f egy keret elküldéséhez szükséges idő
- és T_{pg} a propagációs késés A és B állomások között



Minimális keretméret

22

- Az A küldésének 2*d ideig kell tartania

□ $\text{Min_keret} = \text{ráta (b/s)} * 2 * \text{d (s)}$

- 10 Mbps Ethernet

Azaz a gyorsabb szabványokkal...

Min_keret = r * 2 * távolság (m) / fényseb. (m/s)

- Azaz a kábel össz... a

Távolság = min_kelet * fényszabás / (2 * ráta)

$$(64B*8)*(2*10^8 \text{mps})/(2*10^7 \text{bps}) = 5120 \text{ m\u00e9ter}$$

Minimális keretméret

23

- Az A küldésének $2*d$ ideig kell tartania
 - Min_keret = ráta (b/s) * 2 * d (s)
 - ... de mi az a d ? propagációs késés, melyet a fénysebesség ismeretében ki tudunk számolni
 - Propagációs késés (d) = távolság (m) / fénysebesség (m/s)
 - Azaz:
 - Min_keret = ráta (b/s) * 2 * távolság (m) / fényseb. (m/s)
- Azaz a kábel összhossza
 - Távolság = min_keret * fénysebesség /(2 * ráta)

$$(64B*8)*(2*10^8 \text{mps})/(2*10^7 \text{bps}) = 5120 \text{ méter}$$

Kábelhossz példa

24

$$\text{min_keret} * \text{fénysebesség} / (2 * \text{ráta}) = \text{max_kábelhossz}$$

$$(64B * 8) * (2 * 10^8 \text{mps}) / (2 * 10 \text{Mbps}) = 5120 \text{ méter}$$

- Mi a maximális kábelhossz, ha a minimális keretméret 1024 bajtra változik?
 - 81,9 kilométer
- Mi a maximális kábelhossz, ha a ráta 1 Gbps-ra változik?
 - 51 méter
- Mi történik, ha mindenkető változik egyszerre?
 - 819 méter

Maximális keretméret

25

- Maximum Transmission Unit (MTU): 1500 bájt
- Pro:
 - ▣ Hosszú csomagokban levő biz hibák jelentős javítási költséget okozhatnak (pl. túl sok adatot kell újraküldeni)
- Kontra:
 - ▣ Több bájtot vesztegettünk el a fejlécekben
 - ▣ Összességében nagyobb csomag feldolgozási idő
- Adatközpontokban Jumbo keretek
 - ▣ 9000 bájtos keretek

Ütközésmentes protokollok

26

MOTIVÁCIÓ

- az ütközések hátrányosan hatnak a rendszer teljesítményére
 - hosszú kábel, rövid keret
- a CSMA/CD nem mindenhol alkalmazható

FELTÉTELEZÉSEK

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

Alapvető bittérkép protokoll

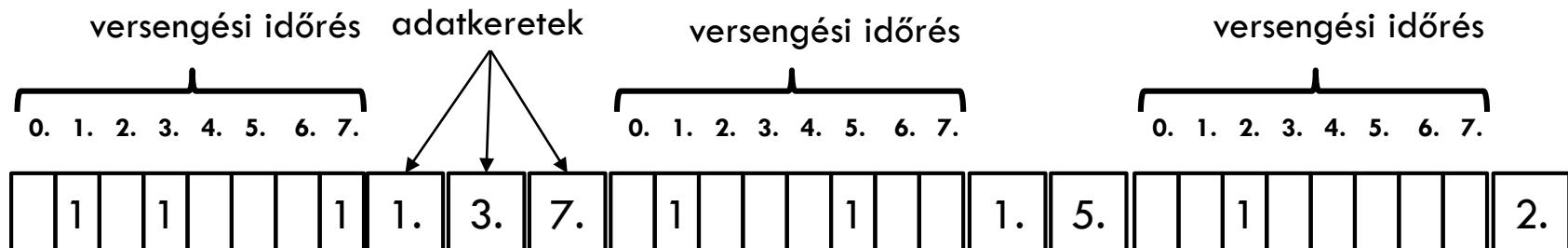
- Egy helyfoglalásos megoldás

27

- alapvető bittérkép eljárás

MŰKÖDÉS

- Az ütköztetési periódus N időrés
- Ha az i -edik állomás küldeni szeretne, akkor a i -edik versengési időrésben egy 1-es bit elküldésével jelezheti. (adatszórás)
- A versengési időszak végére minden állomás ismeri a küldőket. A küldés a sorszámok szerinti sorrendben történik meg.



Bináris visszaszámítás protokoll 1/2

28

- alapvető bittérkép eljárás hátrány, hogy az állomások számának növekedésével a versengési periódus hossza is nő

MŰKÖDÉS

- 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.

A HOSZT (0011)	0	-	-	-
B HOSZT (0110)	0	-	-	-
C HOSZT (1010)	1	0	1	0
D HOSZT (1011)	1	0	1	1

D kerete

Bináris visszaszámítás protokoll 2/2

29

- **Következmény:** a magasabb címmel rendelkező állomásoknak a prioritásuk is magasabb az alacsonyabb című állomásoknál

MOK ÉS WARD MÓDOSÍTÁSA

- ❑ Virtuális állomás címek használata.
 - ❑ minden sikeres átvitel után ciklikusan permutáljuk az állomások címét.

Korlátozott versenyes protokollok

30

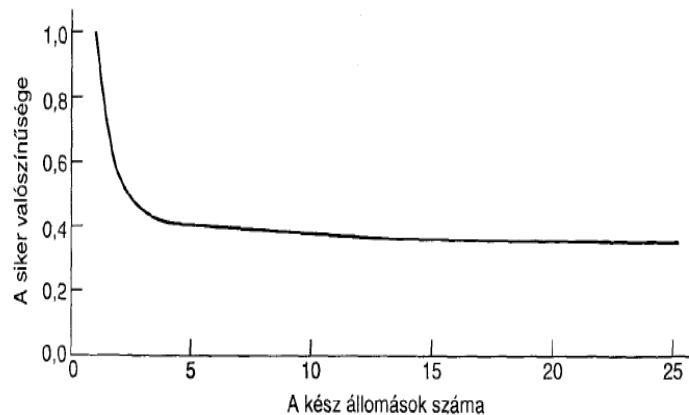
- **Cél:** Ötvözni a versenyhelyzetes és ütközésmentes protokollok jó tulajdonságait.
- **korlátozott versenyes protokoll** – Olyan protokoll, amely kis terhelés esetén versenyhelyzetes technikát használ a kis késleltetés érdekében, illetve nagy terhelés mellett ütközésmentes technikát alkalmaz a csatorna jó kihasználása érdekében.

SZIMMETRIKUS PROTOKOLLOK

- Adott résben k állomás verseng, minden állomás p valószínűsséggel adhat. A csatorna megszerzésének valószínűsége: $kp(1 - p)^{k-1}$.

$$P(\text{siker optimális } p \text{ mellett}) = \left(\frac{k-1}{k}\right)^{k-1}$$

- Azaz a csatorna megszerzésének esélyeit a versenyhelyzetek számának csökkentésével érhetjük el.

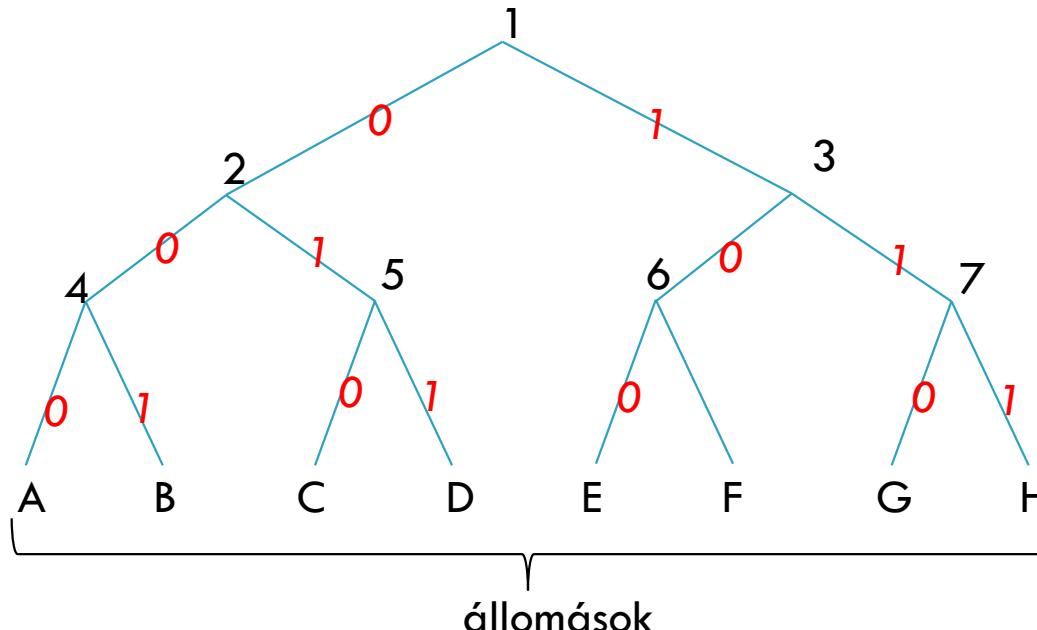


Adaptív fabejárási protokoll 1/2

31

Történeti háttér

- 1943 – Dorfman a katonák szifiliszes fertőzöttségét vizsgálta.
- 1979 – Capetanakis bináris fa reprezentáció az algoritmus számítógépes változatával.



Adaptív fabejárási protokoll 2/2

32

Működés

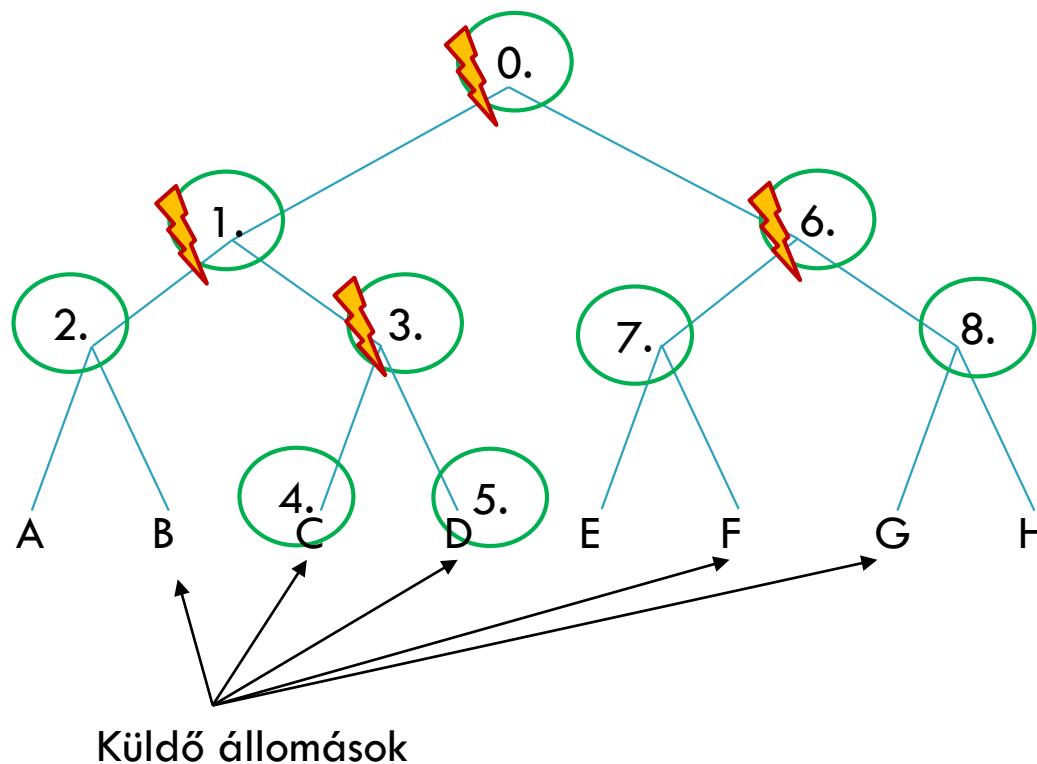
- 0-adik időrésben mindenki küldhet.
 - Ha ütközés történik, akkor megkezdődik a fa *mélységi* bejárása.
- A rések a fa egyes csomópontjaihoz vannak rendelve.
- Ütközéskor rekurzívan az adott csomópont bal illetve jobb gyerekcsomópontjánál folytatódik a keresés.
- Ha egy bitrész kihasználatlan marad, vagy pontosan egy állomás küld, akkor a szóban forgó csomópont keresése befejeződik.

Következmény

- Minél nagyobb a terhelés, annál mélyebben érdemes kezdeni a keresést.

Adaptív fabejárás példa

33



Az adatkapcsolati réteg „legtetején”...

34

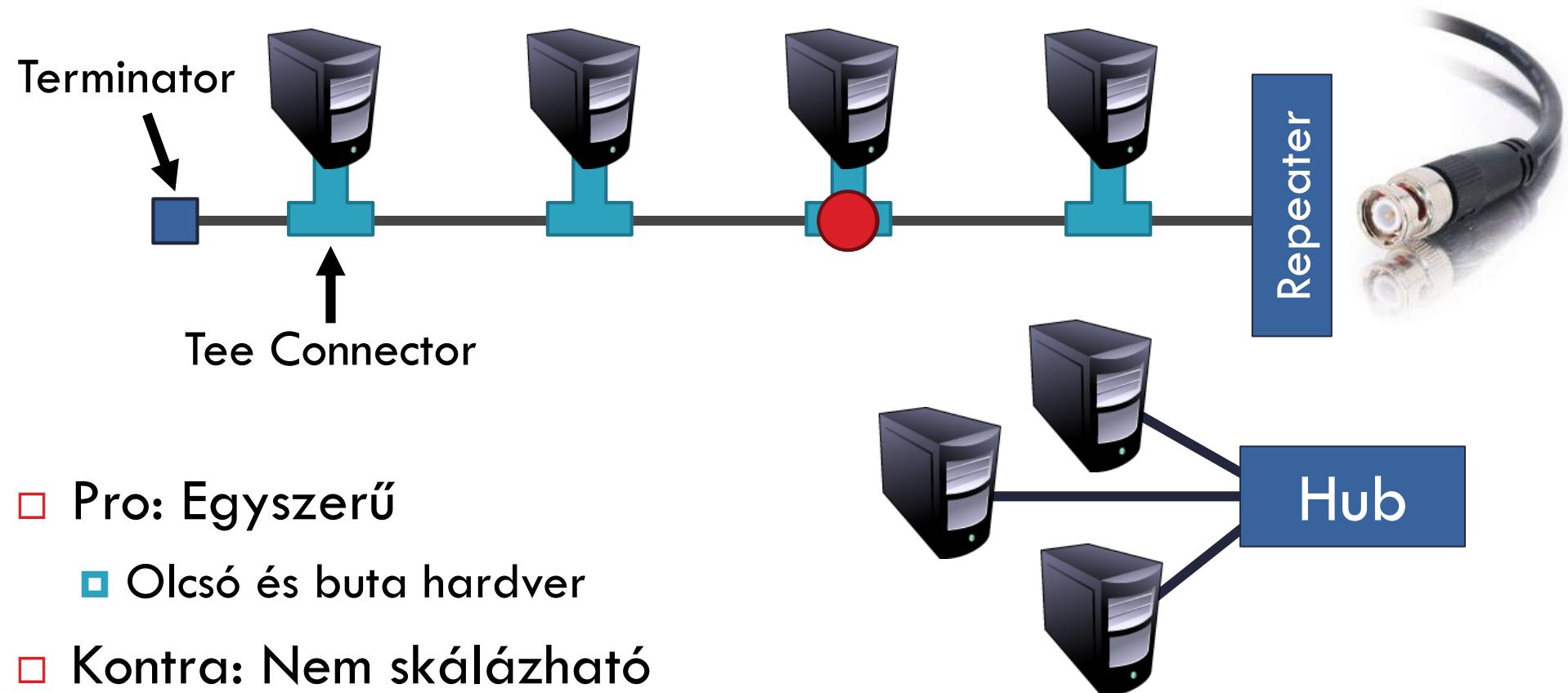


- Bridging, avagy hidak
 - Hogyan kapcsoljunk össze LAN-okat?
- Funkciók:
 - Keretek forgalomirányítása a LAN-ok között
- Kihívások:
 - Plug-and-play, önmagát konfiguráló
 - Esetleges hurkok feloldása

Visszatekintés

35

- Az Ethernet eredetileg adatszóró technológia volt



LAN-ok összekapcsolása

36

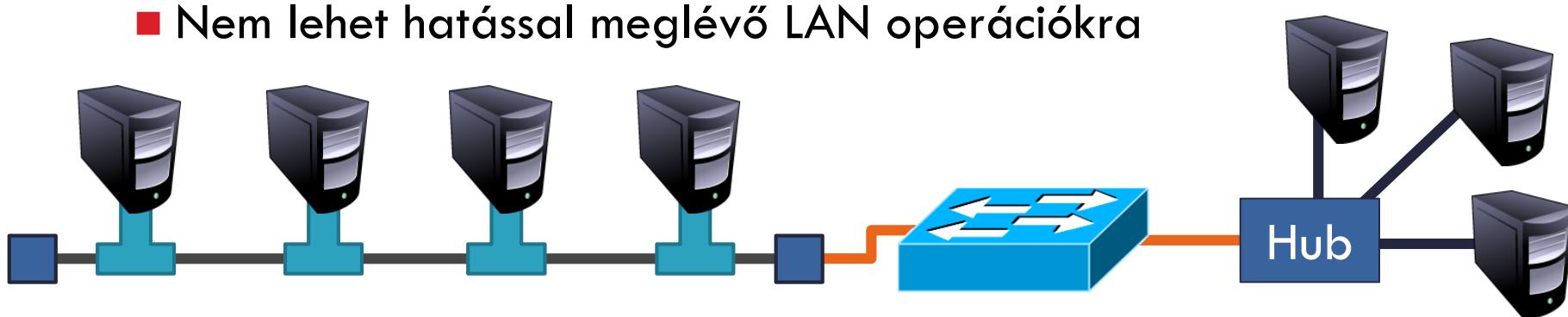


- A bridge-ek lekorlátozzák az ütközési tartományok méretét
 - Jelentősen növelik a skálázhatóságot
 - Kérdés: lehetne-e az egész Internet egy bridge-ekkel összekötött tartomány?
- Hátrány: a bridge-ek sokkal komplexebb eszközök a hub-oknál
 - Fizikai réteg VS Adatkapcsolati réteg
 - Memória pufferek, csomag feldolgozó hardver és routing (útválasztó) táblák szükségesek

Bridge-ek (magyarul: hidak)

37

- Az Ethernet switch eredeti formája
- Több IEEE 802 LAN-t kapcsol össze a 2. rétegben
- Célok
 - Ütközési tartományok számának csökkentése
 - Teljes átlátszóság
 - “Plug-and-play,” önmagát konfiguráló
 - Nem szükségesek hw és sw változtatások a hosztokon/hub-on
 - Nem lehet hatással meglévő LAN operációkra



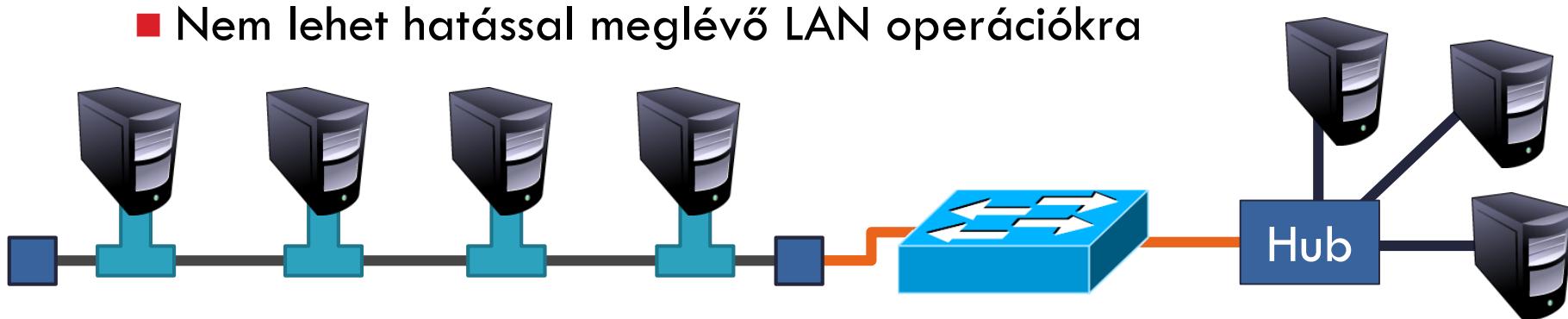
Bridge-ek (magyarul: hidak)

38

□ Az Ethernet switch eredeti formája

- Típus: IEEE 802.1 LAN
- 1. Keretek továbbítása
- 2. (MAC) címek tanulása
- 3. Feszítőfa (Spanning Tree) Algoritmus (a hurkok kezelésére)

- Nem szükségesek hw és sw változtatások a hosztokon/hub-on
- Nem lehet hatással meglévő LAN operációkra



Keret Továbbító Táblák

39

- Minden bridge karbantart egy **továbbító táblát (forwarding table)**

MAC Cím	Port	Kor
00:00:00:00:00:AA	1	1 perc
00:00:00:00:00:BB	2	7 perc
00:00:00:00:00:CC	3	2 mp
00:00:00:00:00:DD	1	3 perc



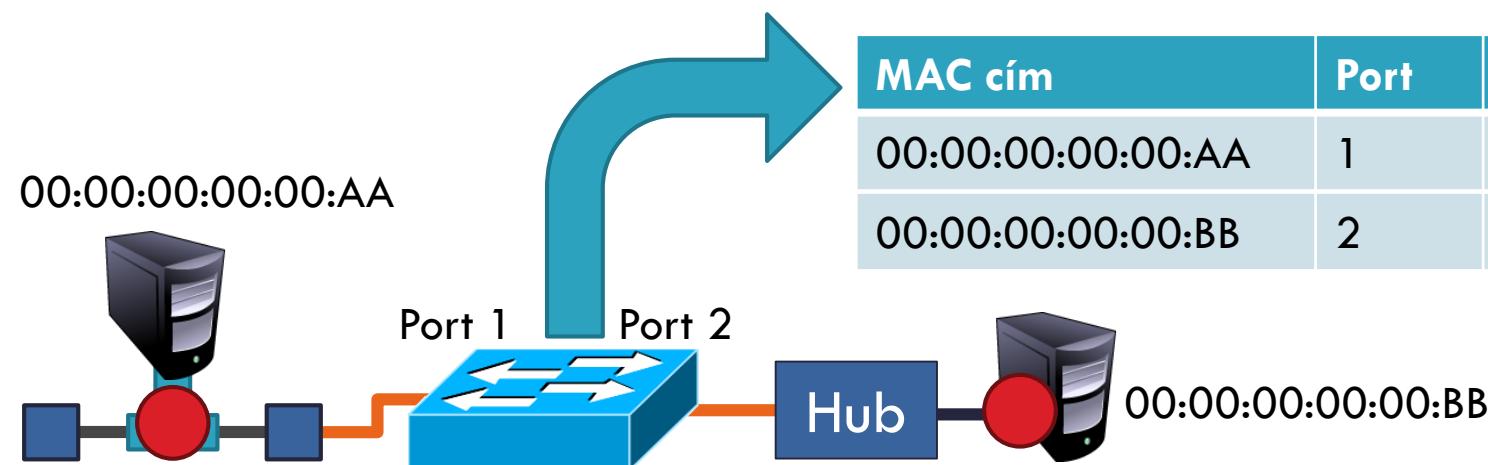
Címek tanulása

40

- Kézi beállítás is lehetséges, de...
 - Időigényes
 - Potenciális hiba forrás
 - Nem alkalmazkodik a változásokhoz (új hosztok léphetnek be és régiek hagyhatják el a hálózatot)
- Ehelyett: tanuljuk meg a címeket
 - Tekintsük a **forrás címeit** a különböző portokon kereteknek --- képezzünk ebből egy táblázatot

Töröljük a régi
bejegyzéseket

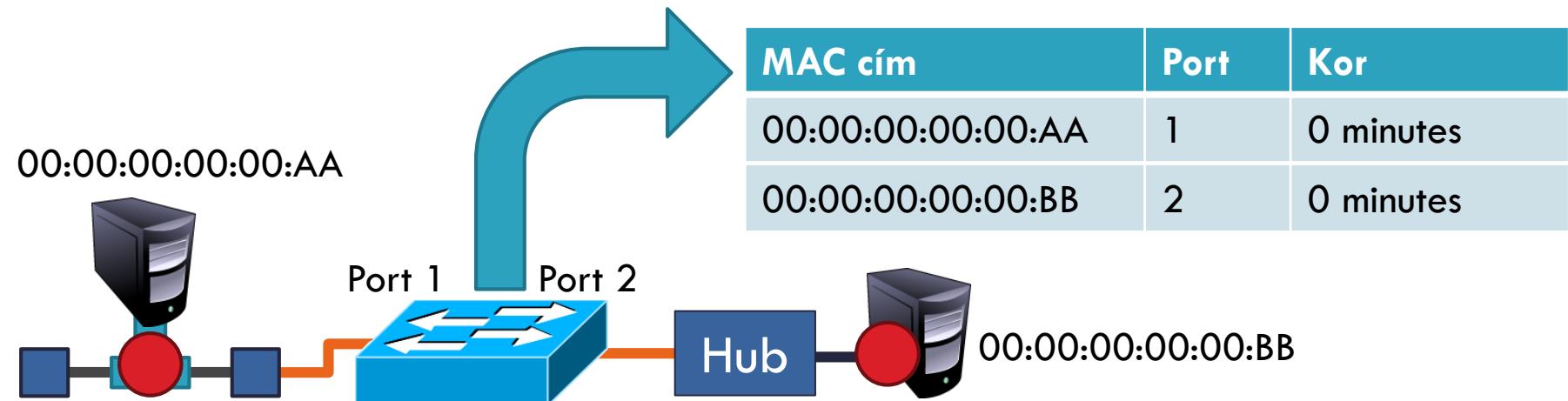
MAC cím	Port	Kor
00:00:00:00:00:AA	1	0 minutes
00:00:00:00:00:BB	2	0 minutes



Címek tanulása

41

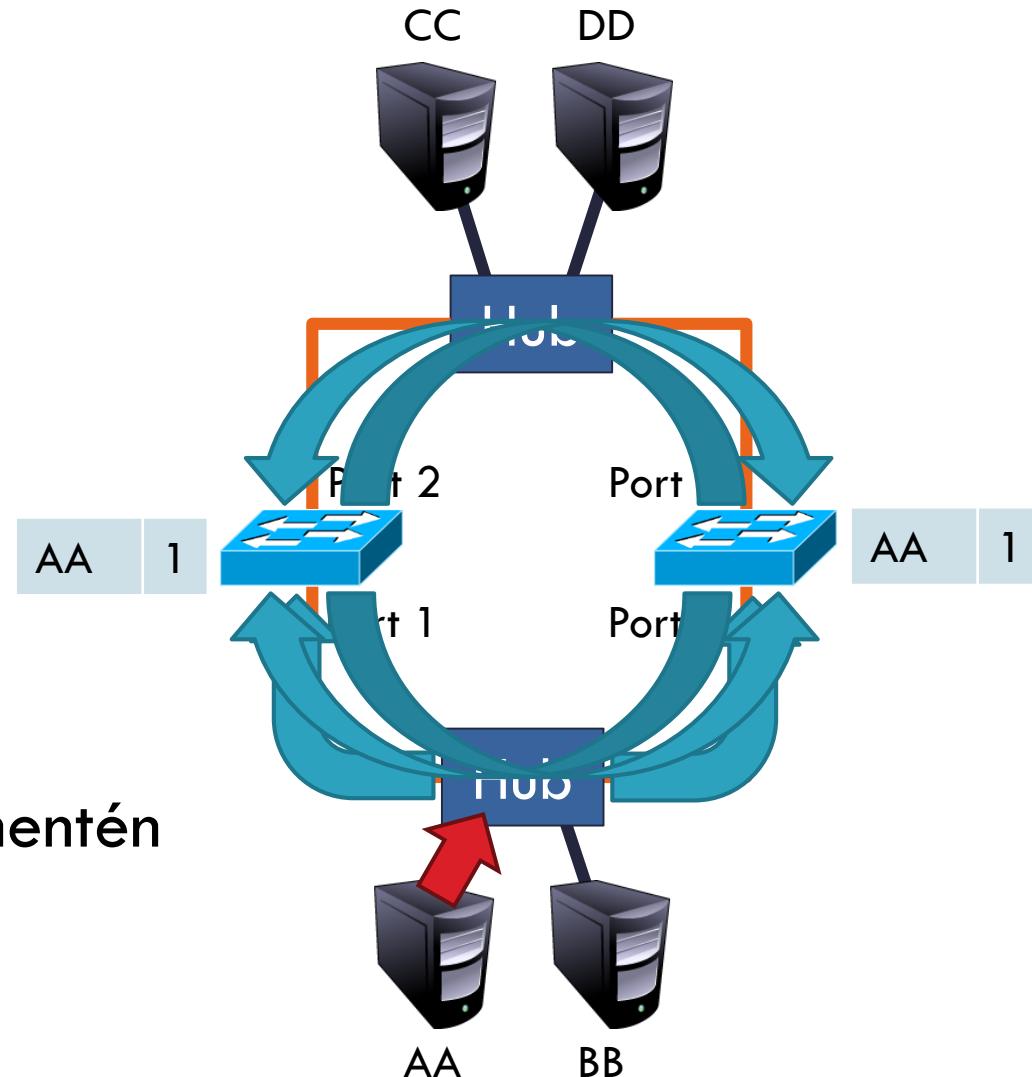
- Kézi beállítás is lehetséges, de...
 - Időigényes
 - Potenciális hiba forrás
 - Nem alkalmazkodik a változásokhoz (új hosztok léphetnek be és régiek hagyhatják el a hálózatot)
- Ehelyett: tanuljuk meg a címeket
 - Tekintsük a **forrás címeit** a különböző portokon beérkező kereteknek --- képezzünk ebből egy táblázatot



Hurkok problémája

42

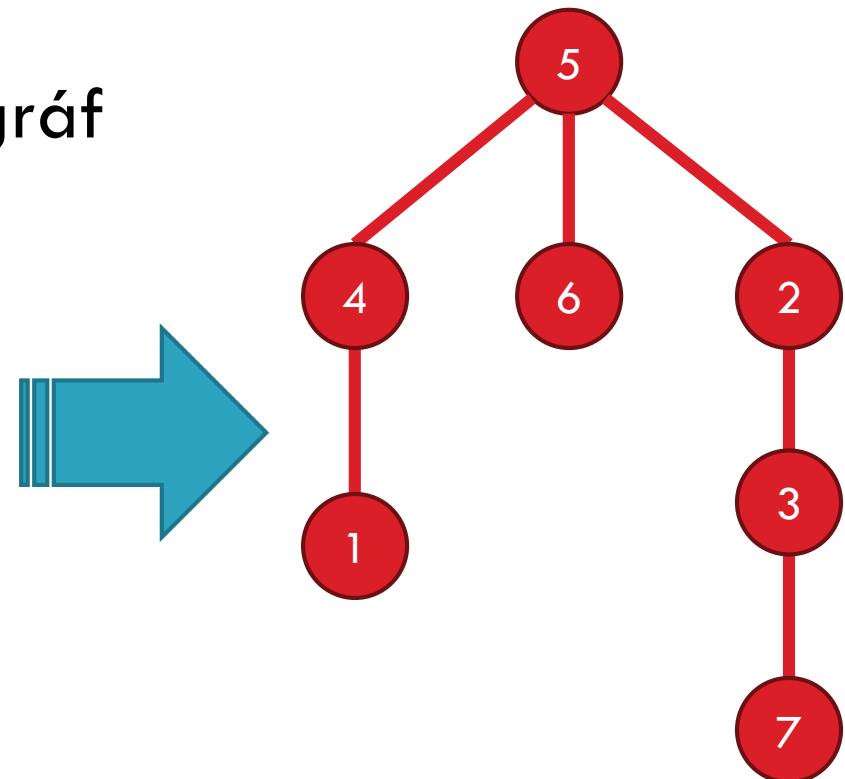
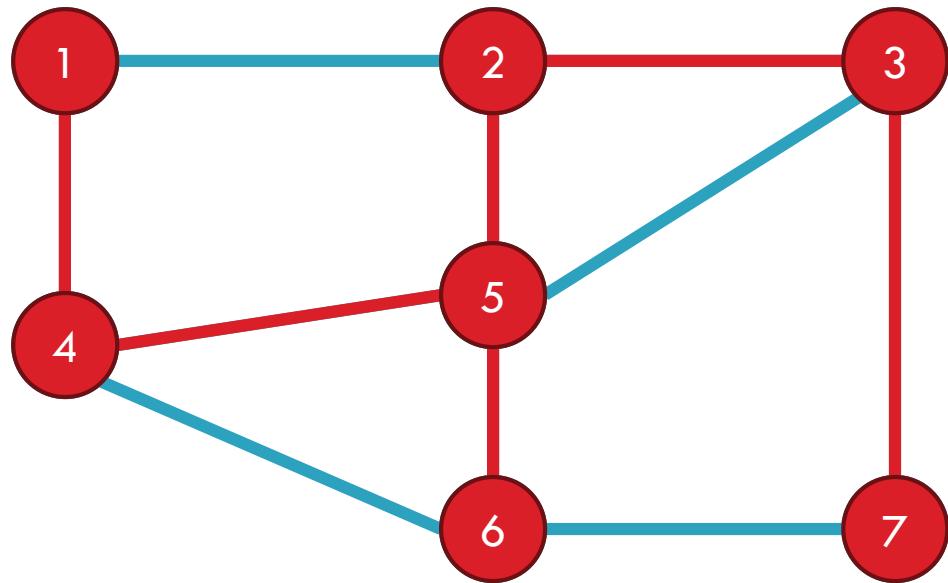
- <Src=AA, Dest=DD>
- Ez megy a végtelenséggig
 - Hogyan állítható meg?
- Távolítsuk el a hurkokat a topológiából
 - A kábelek kihúzása nélkül
- 802.1 (LAN) definiál egy algoritmust **feszítőfa** fépítéséhez és karbantartásához, mely mentén lehetséges a keretek továbbítása



Feszítőfa

43

- Egy gráf éleinek részhalmaza, melyre teljesül:
 - Lefed minden csomópontot
 - Nem tartalmaz köröket
- Továbbá a struktúra egy fa-gráf



A 802.1 feszítőfa algoritmusa

44

1. Az egyik bridge-et megválasztjuk a fa gyökerének
 2. minden bridge megkeresi a legrövidebb utat a gyökérhez
 3. Ezen utak unióját véve megkapjuk a feszítőfát
-
- A fa építése során a bridge-ek egymás között konfigurációs üzeneteket (Configuration Bridge Protocol Data Units [BPDU]) cserélnek
 - A gyökér elem megválasztásához
 - A legrövidebb utak meghatározásához
 - A gyökérhez legközelebbi szomszéd (next hop) állomás és a hozzá tartozó port azonosításához
 - A feszítőfához tartozó portok kiválasztása

Gyökér meghatározása

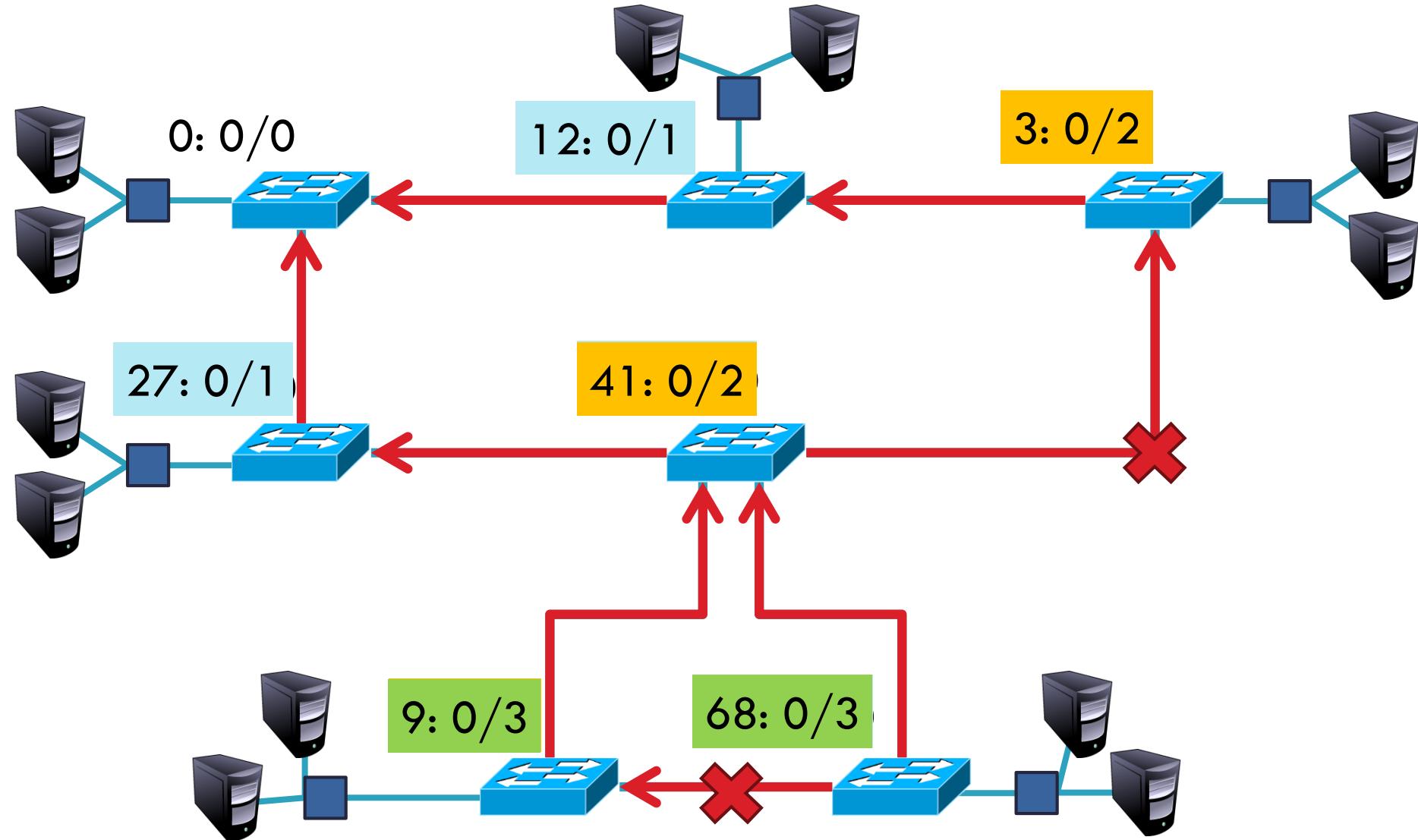
45

- Kezdetben minden állomás feltételezi magáról, hogy gyökér
- Bridge-ek minden irányba szétküldik a BPDU üzeneteiket:


The diagram shows a horizontal structure of three colored boxes representing fields in a BPDU message. From left to right: a blue box labeled 'Bridge ID', a red box labeled 'Gyökér ID' (Root ID), and an orange box labeled 'Út költség a gyökérhez' (Cost to root).
- A fogadott BPDU üzenet alapján, minden switch választ:
 - Egy új gyökér elemet (legkisebb ismert Gyökér ID alapján)
 - Egy új gyökér portot (melyik interfész megy a gyökér irányába)
 - Egy új kijelölt bridge-et (a következő állomás a gyökérhez vezető úton)

Feszítőfa építése

46



Bridge-ek vs. Switch-ek

Hidak vs. Kapcsolók

47

- A bridge-ek lehetővé teszik hogy növeljük a LAN-ok kapacitását
 - ▣ Csökkentik a sikeres átvitelhez szükséges elküldendő csomagok számát
 - ▣ Kezeli a hurkokat
- A switch-ek a bridge-ek speciális esetei
 - ▣ minden port egyetlen egy hoszthoz kapcsolódik
 - lehet egy kliens terminál
 - vagy akár egy másik switch
 - ▣ Full-duplex link-ek
 - ▣ Egyszerűsített hardver: nincs szükség CSMA/CD-re!
 - ▣ Különböző sebességű/rátájú portok is lehetségesek

Kapcsoljuk össze az Internetet

48

- Switch-ek képességei:
 - MAC cím alapú útvonalválasztás a hálózatban
 - Automatikusan megtanulja az utakat egy új állomáshoz
 - Feloldja a hurkokat
- Lehetne a teljes internet egy ily módon összekötött tartomány?

NEM

Korlátok

49

- Nem hatékony
 - Elárasztás ismeretlen állomások megtalálásához
- Gyenge teljesítmény
 - A feszítőfa nem foglalkozik a terhelés elosztással
 - Hot spots
- Nagyon gyenge skálázhatóság
 - minden switch-nek az Internet összes MAC címét ismerni kellene a továbbító táblájában!
- Az IP fogja ezt a problémát megoldani...

Hálózati réteg

50



- Szolgáltatás
 - Csomagtovábbítás
 - Útvonalválasztás
 - Csomag fragmentálás kezelése
 - Csomag ütemezés
 - Puffer kezelés
- Interfész
 - Csomag küldése egy adott végpontnak
- Protokoll
 - Globálisan egyedi címeket definiálása
 - Routing táblák karbantartása
- Példák: Internet Protocol (IPv4), IPv6

Forgalomirányító algoritmusok

51

DEFINÍCIÓ

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

- A folyamat két jól-elkülöníthető lépésre bontható fel:
 1. Forgalomirányító táblázatok feltöltése és karbantartása.
 2. Továbbítás.

ELVÁRÁSOK

helyesség, egyszerűség, robosztusság, stabilitás, **igazságosság**, optimalitás és hatékonyság

ALGORITMUS OSZTÁLYOK

1. Adaptív algoritmusok
 - A topológia és rendszerint a forgalom is befolyásolhatja a döntést
2. Nem-adaptív algoritmusok
 - offline meghatározás, betöltés a router-ekbe induláskor

Forgalomirányító algoritmusok

52

KÜLÖNBSÉGEK AZ EGYES ADAPTÍV ALGORITMUSOKBAN

1. Honnan kapják az információt?
 - szomszédok, helyileg, minden router-től
2. Mikor változtatják az útvonalakat?
 - meghatározott másodpercenként, terhelés változásra, topológia változásra
3. Milyen mértékeket használnak az optimalizáláshoz?
 - távolság, ugrások (hops) száma, becsült késleltetés

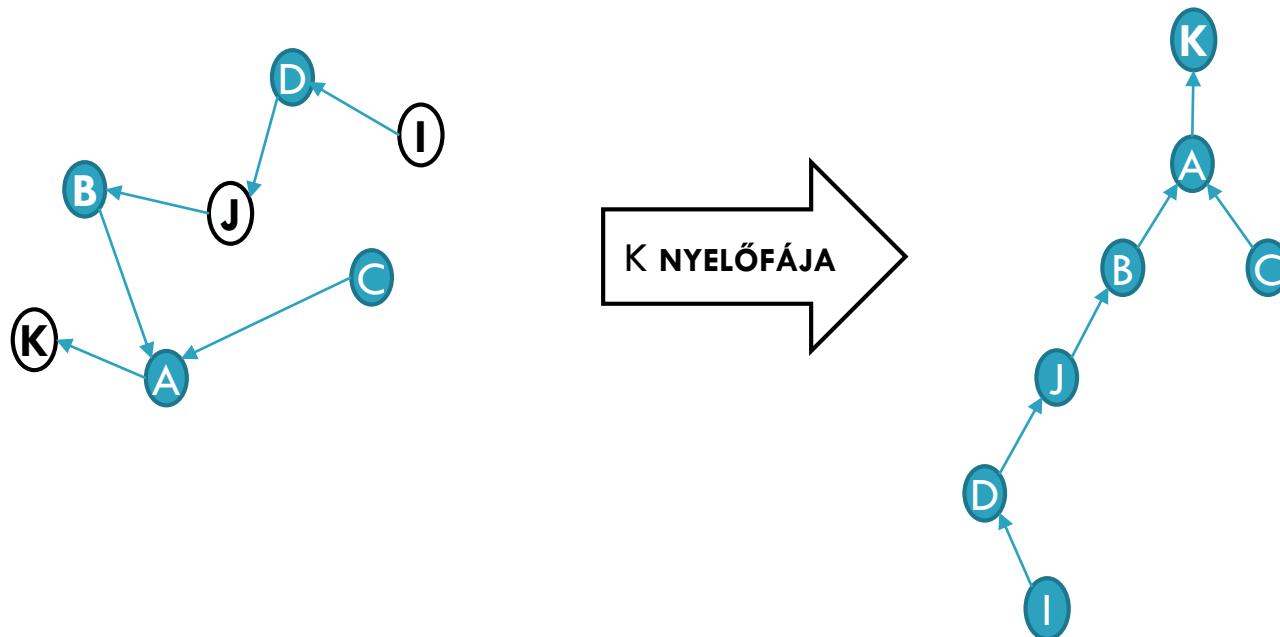
Optimalitási elv

53

Ha J router az I router-től K router felé vezető optimális útvonalon helyezkedik el, akkor a J -től a K -ig vezető útvonal ugyanerre esik.

□ Következmény

Az összes forrásból egy célba tartó optimális utak **egy olyan fát** alkotnak, melynek a gyökere a cél. Ezt nevezzük **nyelőfának**.



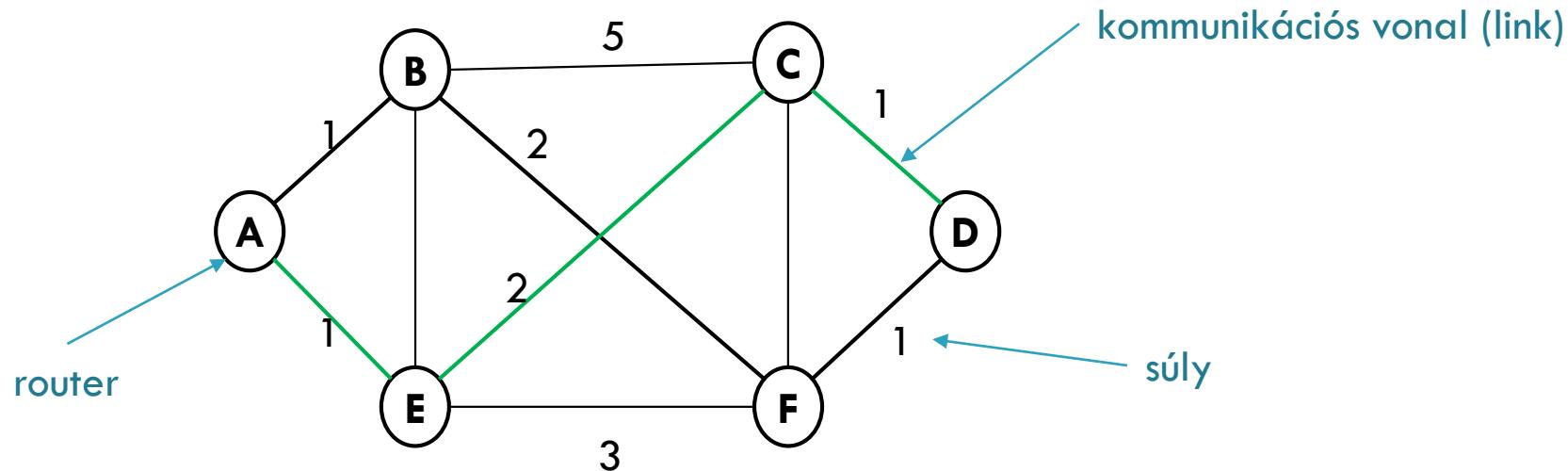
Legrövidebb út alapú forgalomirányítás

54

ALHÁLÓZAT REPREZENTÁCIÓJA

Az alhálózat tekinthető egy gráfnak, amelyben minden router egy csomópontnak és minden él egy kommunikációs vonalnak (link) felel meg. Az éleken értelmezünk egy $w: E \rightarrow \mathbb{R}_0^+$ nem-negatív súlyfüggvényt, amelyek a legrövidebb utak meghatározásánál használunk.

- $G=(V,E)$ gráf reprezentálja az alhálózatot
- P útvonal súlya: $w(P) = \sum_{e \in P} w(e)$



Távolságvektor alapú forgalomirányítás

55

- Dinamikus algoritmusoknak 2 csoportja van:
 - távolságvektor alapú illetve (distance vector routing)
 - kapcsolatállapot alapú (link-state routing)
- **Távolságvektor alapú:** minden router-nek egy táblázatot kell karbantartania, amelyben minden célhoz szerepel a legrövidebb ismert távolság, és annak a vonalnak az azonosítója, amelyiken a célhoz lehet eljutni. A táblázatokat a szomszédoktól származó információk alapján frissítik.
 - Elosztott Bellman-Ford forgalomirányítási algoritmusként is nevezik.
 - ARPANET eredeti forgalomirányító algoritmusa ez volt. RIP (Routing Information Protocol) néven is ezt használták.

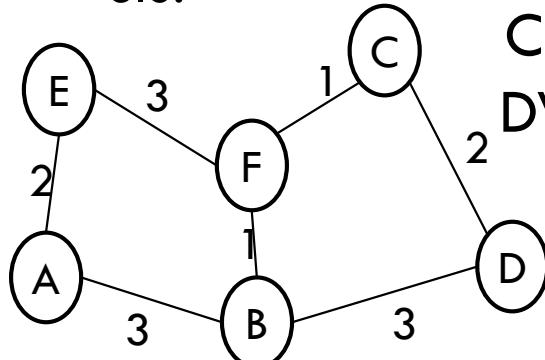
Távolságvektor alapú forgalomirányítás

Elosztott Bellman-Ford algoritmus

56

KÖRNYEZET ÉS MŰKÖDÉS

- minden csomópont csak a közvetlen szomszédjaival kommunikálhat.
- Aszinkron működés.
- minden állomásnak van saját távolság vektora. Ezt periodikusan elküldi a direkt szomszédoknak.
- A kapott távolság vektorok alapján minden csomópont új táblázatot állít elő.



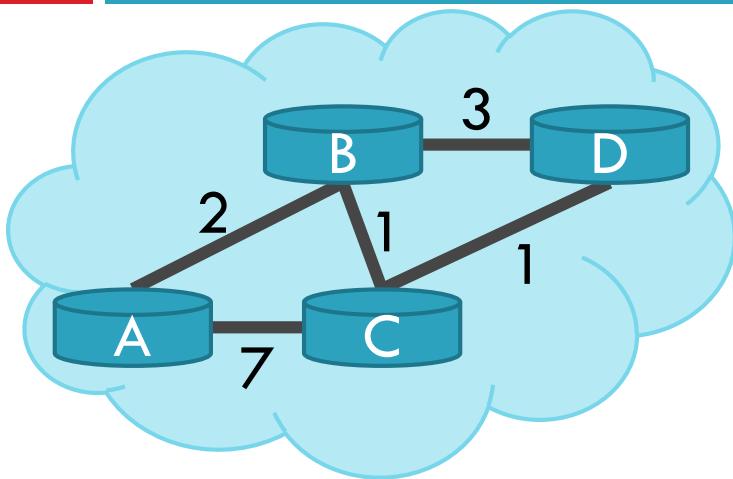
C állomás
DV táblája

Cél	Ktsg.
A	5
B	2
D	2
E	4
F	1

- Nincs bejegyzés C-hez
- Kezdetben csak a közvetlen szomszédokhoz van info
 - más célállomások költsége = ∞
- Végül kitöltött vektort kapunk

Distance Vector Initialization

57



Node A

Dest.	Cost	Next
B	2	B
C	7	C
D	∞	

Node B

Dest.	Cost	Next
A	2	A
C	1	C
D	3	D

Node C

Dest.	Cost	Next
A	7	A
B	1	B
D	1	D

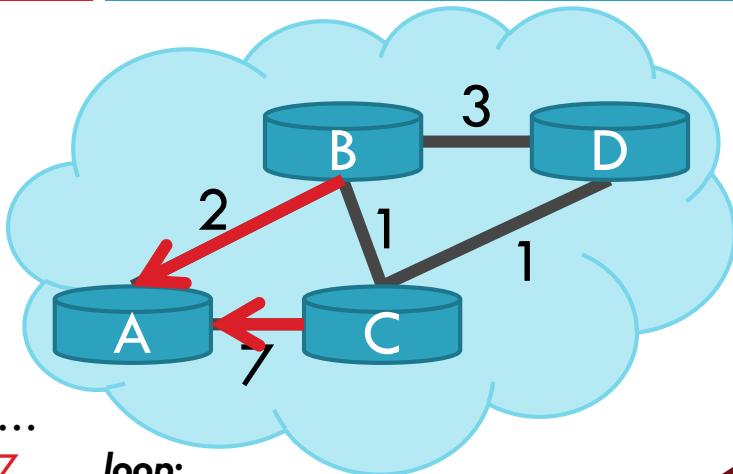
Node D

Dest.	Cost	Next
A	∞	
B	3	B
C	1	C

1. **Initialization:**
2. **for all neighbors V do**
3. **if V adjacent to A**
4. $D(A, V) = c(A, V);$
5. **else**
6. $D(A, V) = \infty;$
- ...

Distance Vector: 1st Iteration

58



```

...
7. loop:
...
12. else if (update D(V, Y) received)
13.     for all destinations Y
14.         if (destination Y is not yet updated)
15.             D(A, Y) = infinity
16.         else
17.             D(A, Y) = min(D(A, V) + D(V, Y));
18.             if (there is a new min. for dest. Y)
19.                 send D(A, Y) to all neighbors
20. forever

```

Node A			Node B		
Dest.	Cost	Next	Dest.	Cost	Next
B	2	B	A	2	A
C	3	B	C	1	C
D	5	B	D	2	C

The table represents the distance vectors for Node A and Node B. The first row shows the initial state where Node A has a cost of 2 to node B and Node B has a cost of 2 to node A. The second row shows Node A updating its cost to node C from 7 to 3, and Node B updating its cost to node C from infinity to 1. The third row shows Node A updating its cost to node D from infinity to 5, and Node B updating its cost to node D from infinity to 2. Red arrows indicate the update process: one arrow points from Node A's cost for C to Node B's cost for C, another points from Node A's cost for D to Node B's cost for D, and a third points from Node A's cost for D to Node B's Next column.

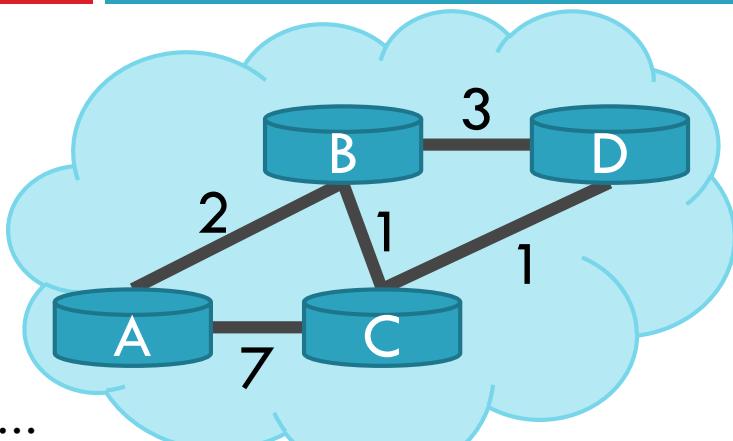
$D(A,C) = \min(D(A,C), D(A,B) + D(B,C))$

$D(A,D) = \min(D(A,D), D(A,B) + D(B,D))$

$= \min(8, 3 + 3) = 5$

Distance Vector: End of 3rd Iteration

59



Node A

Dest.	Cost	Next
B	2	B
C	3	B
D	4	B

Node B

Dest.	Cost	Next
A	2	A
C	1	C
D	2	C



- Nothing changes, algorithm terminates
- Until something changes...

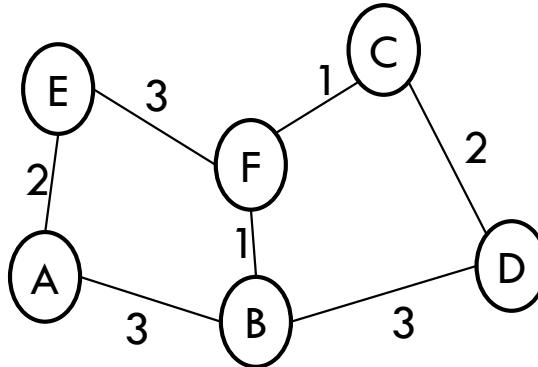
```
7. loop  
...  
12. else  
13. for  
14. if  
15.  
16. else  
17.     D(A, Y) =  
           min(D(A, Y),  
                  D(A, V) + D(V, Y));  
18. if (there is a new min. for dest. Y)  
    send D(A, Y) to all neighbors  
19.  
20. forever
```

Dest.	Cost	Next
A	3	B
B	1	B
D	1	D

Dest.	Cost	Next
A	4	C
B	2	C
C	1	C



Elosztott Bellman-Ford algoritmus – példa



Becsült késleltetés A-tól kezdetben

A	cost	N. Hop
B	3	B
C	∞	-
D	∞	-
E	2	E
F	∞	-

B vektora A-nak

A	3
B	0
C	∞
D	3
E	∞
F	3

E vektora A-nak

A	2
B	∞
C	∞
D	∞
E	0
F	3

Új becsült késleltetés A-tól

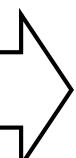
A	cost	N. Hop
B	3	B
C	∞	-
D	6	B
E	2	E
F	4	B

A vektor A-tól és E-nek

A	0
B	3
C	∞
D	6
E	2
F	4

Új becsült késleltetés A-tól

A	cost	N. Hop
B	3	B
C	5	B
D	6	B
E	2	E
F	4	B



```

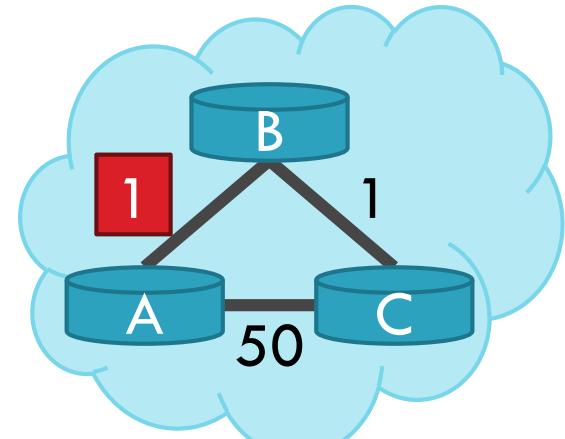
7. loop:
8.   wait (link cost update or update message)
9.   if ( $c(A, V)$  changes by  $d$ )
10.    for all destinations  $Y$  through  $V$  do
11.       $D(A, Y) = D(A, Y) + d$ 
12.   else if (update  $D(V, Y)$  received from  $V$ )
13.    for all destinations  $Y$  do
14.      if (destination  $Y$  through  $V$ )
15.         $D(A, Y) = D(A, V) + D(V, Y);$ 
16.      else
17.         $D(A, Y) = \min(D(A, Y), D(A, V) + D(V, Y));$ 

```

Link Cost
Algorithm

Good news travels fast

Algorithm
terminates



Node B

	D	C	N
A	4	A	
C	1	B	

	D	C	N
A	1	A	
C	1	B	

Node C

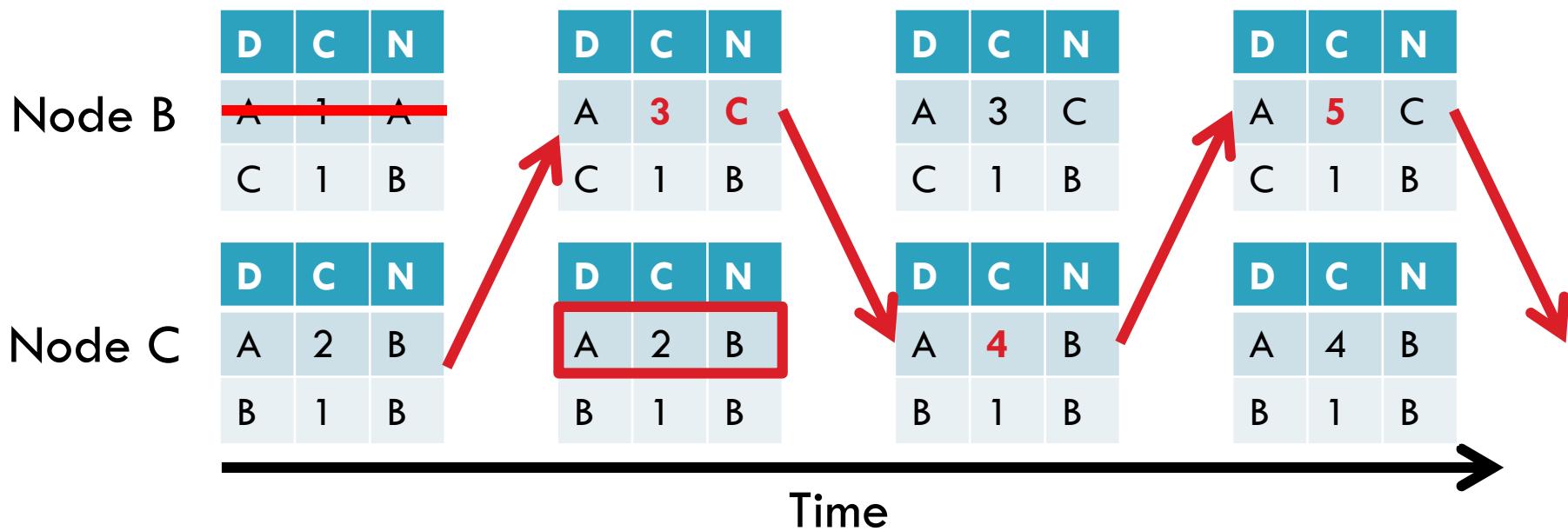
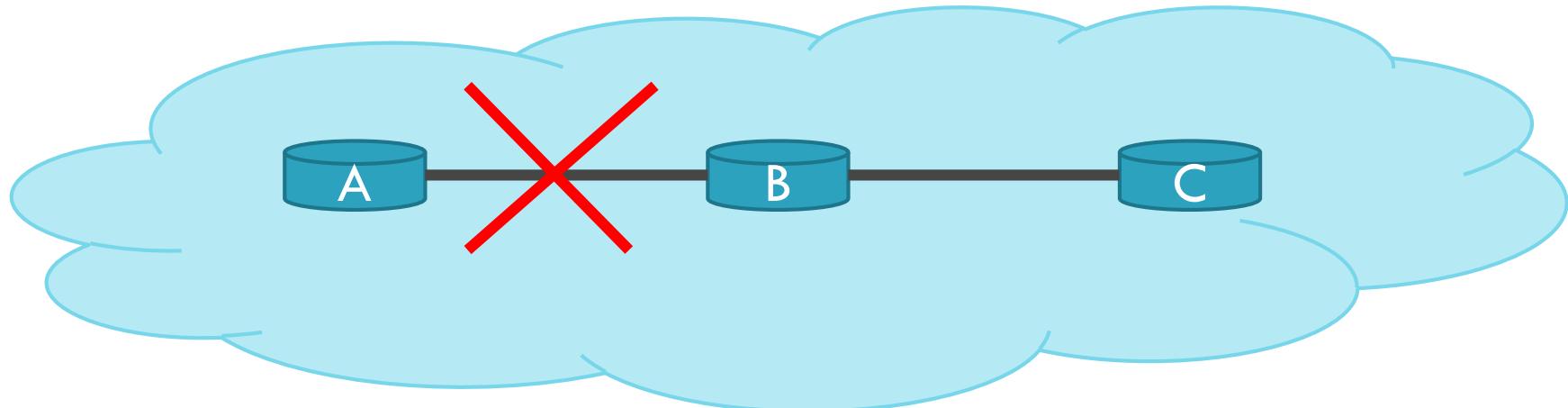
	D	C	N
A	5	B	
B	1	B	

	D	C	N
A	5	B	
B	1	B	

Time

Távolság vektor protokoll – Végtelenig számolás problémája (*count to infinity*)

62

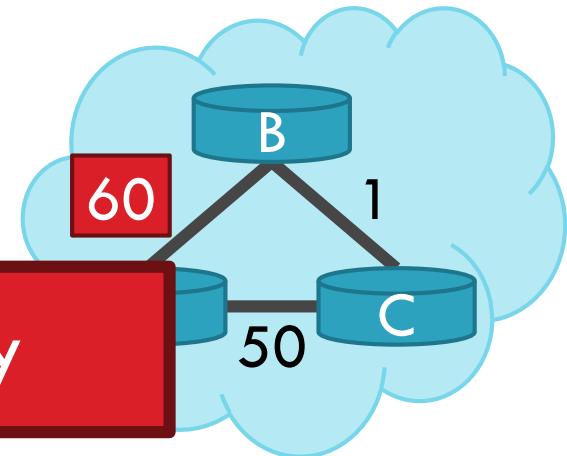


Példa - Count to Infinity Problem

63

- Node B knows $D(C, A) = 5$
- However, B does not know the path is C -> D -> A
- Thus, $D(B,A) = \infty$

Bad news travels slowly



Node B

D	C	N
A	4	A
C	1	B

D	C	N
A	6	C
C	1	B

D	C	N
A	6	C
C	1	B

D	C	N
A	8	C
C	1	B

Node C

D	C	N
A	5	B
B	1	B

D	C	N
A	5	B
B	1	B

D	C	N
A	7	B
B	1	B

D	C	N
A	7	B
B	1	B



Elosztott Bellman-Ford algoritmus – Végtelenig számolás problémája

64

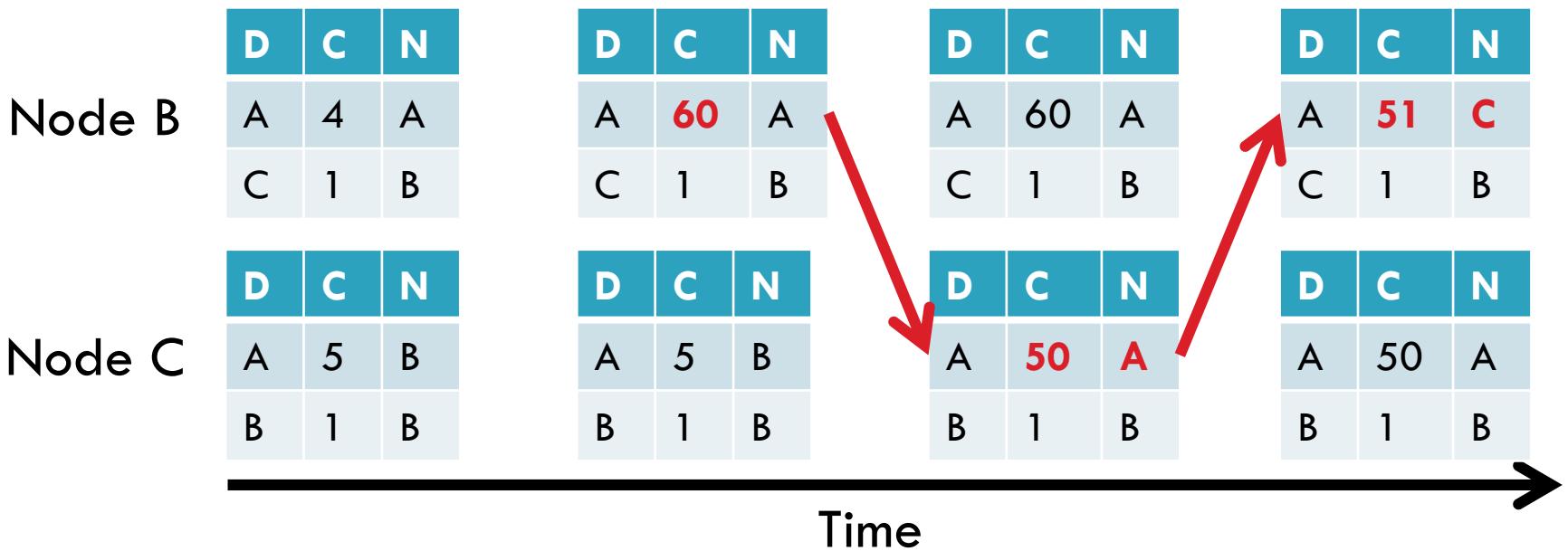
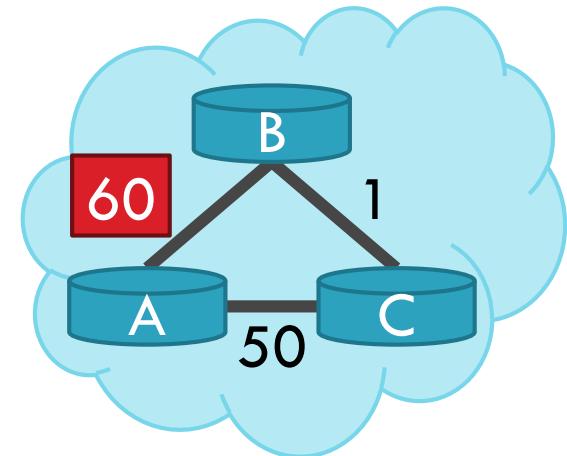
PROBLÉMA

- A „ jó hír” gyorsan terjed.
- A „rossz hír” lassan terjed.
- Azaz ciklusok keletkezhetnek.
- Lehetséges megoldás:
 - „**split horizon with poisoned reverse**”: negatív információt küld vissza arról a szomszédjának, amit tőle „tanult”. (RFC 1058)

Split horizon with Poisoned Reverse

65

- Ha C B-n keresztül irányítja a forgalmat A állomáshoz
 - ▣ C állomás B-nek $D(C, A) = \infty$ távolságot küld
 - ▣ Azaz B állomás nem fog C-n keresztül irányítani az A-ba menő forgalmat



Köszönöm a figyelmet!

Számítógépes Hálózatok

7. Előadás: Hálózati réteg

Based on slides from **Zoltán Ács ELTE** and D. Choffnes Northeastern U., Philippa Gill from StonyBrook University , Revised Spring 2016 by S. Laki

Hálózati réteg

2



- Szolgáltatás
 - ▣ Csomagtovábbítás
 - ▣ Útvonalválasztás
 - ▣ Csomag fragmentálás kezelése
 - ▣ Csomag ütemezés
 - ▣ Puffer kezelés
- Interfész
 - ▣ Csomag küldése egy adott végpontnak
- Protokoll
 - ▣ Globálisan egyedi címeket definiálása
 - ▣ Routing táblák karbantartása
- Példák: Internet Protocol (IPv4), IPv6

Forgalomirányító algoritmusok

3

DEFINÍCIÓ

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

- A folyamat két jól-elkülöníthető lépésre bontható fel:
 1. Forgalomirányító táblázatok feltöltése és karbantartása.
 2. Továbbítás.

ELVÁRÁSOK

helyesség, egyszerűség, robosztusság, stabilitás, **igazságosság**, optimalitás és hatékonyság

ALGORITMUS OSZTÁLYOK

1. Adaptív algoritmusok
 - A topológia és rendszerint a forgalom is befolyásolhatja a döntést
2. Nem-adaptív algoritmusok
 - offline meghatározás, betöltés a router-ekbe induláskor

Forgalomirányító algoritmusok

4

KÜLÖNBSÉGEK AZ EGYES ADAPTÍV ALGORITMUSOKBAN

1. Honnan kapják az információt?
 - szomszédok, helyileg, minden router-től
2. Mikor változtatják az útvonalakat?
 - meghatározott másodpercenként, terhelés változásra, topológia változásra
3. Milyen mértékeket használnak az optimalizáláshoz?
 - távolság, ugrások (hops) száma, becsült késleltetés

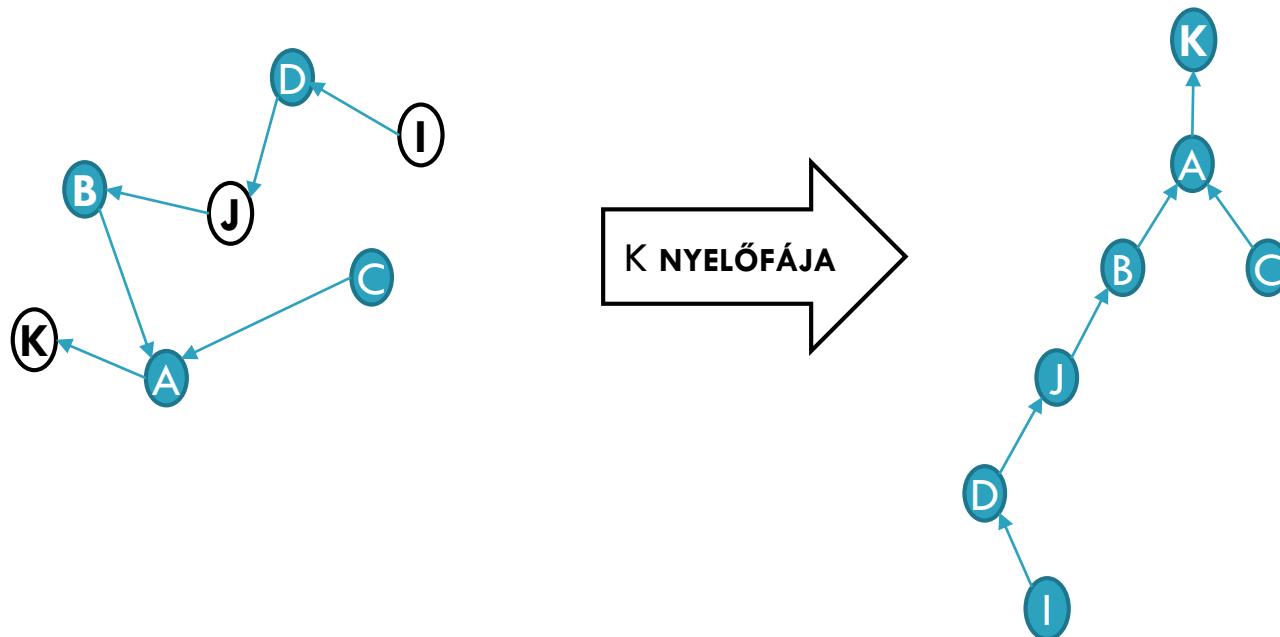
Optimalitási elv

5

Ha J router az I router-től K router felé vezető optimális útvonalon helyezkedik el, akkor a J -től a K -ig vezető útvonal ugyanerre esik.

□ Következmény

Az összes forrásból egy célba tartó optimális utak **egy olyan fát** alkotnak, melynek a gyökere a cél. Ezt nevezzük **nyelőfának**.



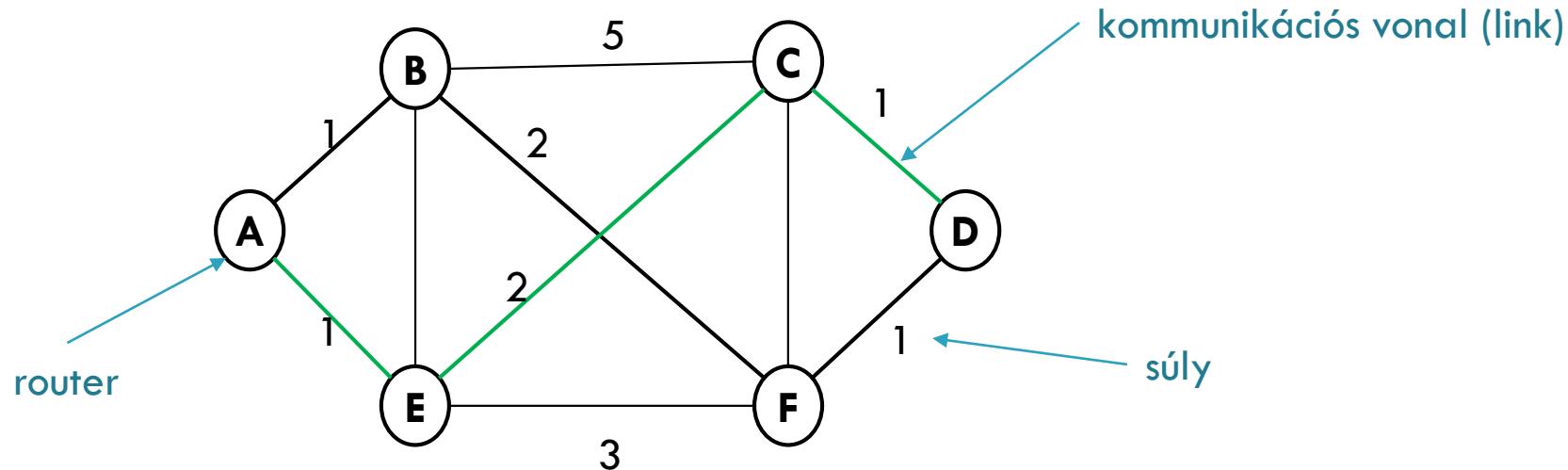
Legrövidebb út alapú forgalomirányítás

6

ALHÁLÓZAT REPREZENTÁCIÓJA

Az alhálózat tekinthető egy gráfnak, amelyben minden router egy csomópontnak és minden él egy kommunikációs vonalnak (link) felel meg. Az éleken értelmezünk egy $w: E \rightarrow \mathbb{R}_0^+$ nem-negatív súlyfüggvényt, amelyek a legrövidebb utak meghatározásánál használunk.

- $G=(V,E)$ gráf reprezentálja az alhálózatot
- P útvonal súlya: $w(P) = \sum_{e \in P} w(e)$



Távolságvektor alapú forgalomirányítás

7

- Dinamikus algoritmusoknak 2 csoportja van:
 - távolságvektor alapú illetve (distance vector routing)
 - kapcsolatállapot alapú (link-state routing)
- **Távolságvektor alapú:** minden router-nek egy táblázatot kell karbantartania, amelyben minden célhoz szerepel a legrövidebb ismert távolság, és annak a vonalnak az azonosítója, amelyiken a célhoz lehet eljutni. A táblázatokat a szomszédoktól származó információk alapján frissítik.
 - Elosztott Bellman-Ford forgalomirányítási algoritmusként is nevezik.
 - ARPANET eredeti forgalomirányító algoritmusa ez volt. RIP (Routing Information Protocol) néven is ezt használták.

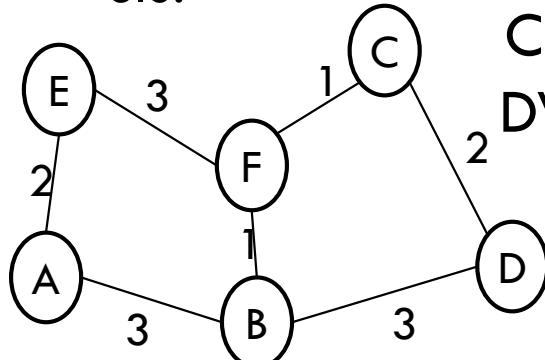
Távolságvektor alapú forgalomirányítás

Elosztott Bellman-Ford algoritmus

8

KÖRNYEZET ÉS MŰKÖDÉS

- minden csomópont csak a közvetlen szomszédjaival kommunikálhat.
- Aszinkron működés.
- minden állomásnak van saját távolság vektora. Ezt periodikusan elküldi a direkt szomszédoknak.
- A kapott távolság vektorok alapján minden csomópont új táblázatot állít elő.



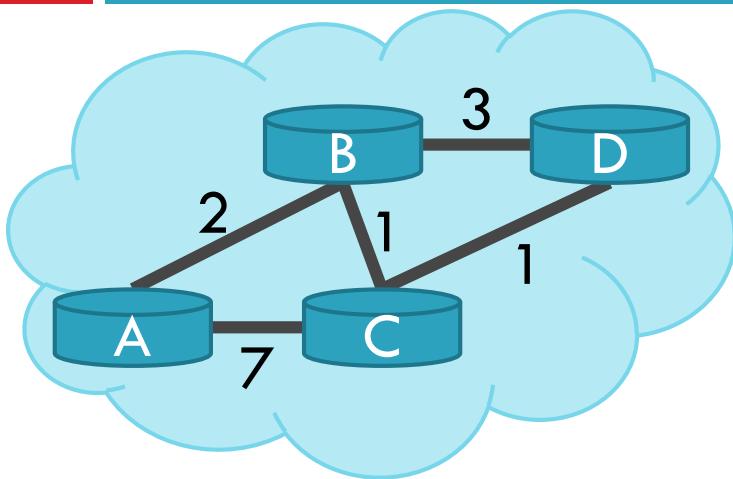
C állomás
DV táblája

Cél	Ktsg.
A	5
B	2
D	2
E	4
F	1

- Nincs bejegyzés C-hez
- Kezdetben csak a közvetlen szomszédokhoz van info
 - más célállomások költsége = ∞
- Végül kitöltött vektort kapunk

Distance Vector Initialization

9



Node A

Dest.	Cost	Next
B	2	B
C	7	C
D	∞	

Node B

Dest.	Cost	Next
A	2	A
C	1	C
D	3	D

1. **Initialization:**
2. **for all neighbors V do**
3. **if V adjacent to A**
4. $D(A, V) = c(A, V);$
5. **else**
6. $D(A, V) = \infty;$
- ...

Node C

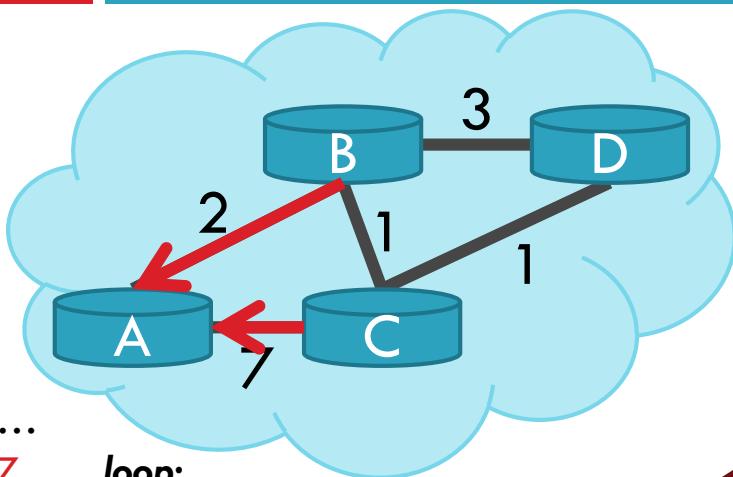
Dest.	Cost	Next
A	7	A
B	1	B
D	1	D

Node D

Dest.	Cost	Next
A	∞	
B	3	B
C	1	C

Distance Vector: 1st Iteration

10



```

...
7. loop:
...
12. else if (update D(V, Y) received)
13.     for all destinations Y
14.         if (destination Y is not yet updated)
15.             D(A, Y) = infinity
16.         else
17.             D(A, Y) = min(D(A, V) + D(V, Y));
18.             if (there is a new min. for dest. Y)
19.                 send D(A, Y) to all neighbors
20. forever

```

Dest.	Cost	Next
B	2	B
C	3	B
D	5	B

Dest.	Cost	Next
A	2	A
C	1	C
D	2	C

$$D(A,C) = D(A,B) + D(B,C)$$

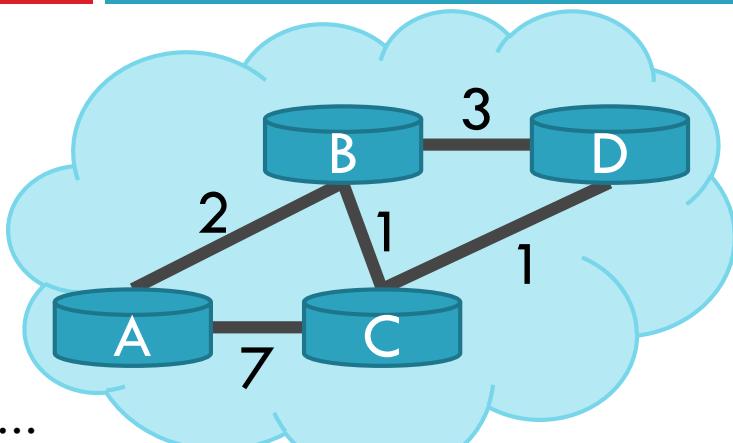
$$\begin{aligned} D(A,D) &= \min(D(A,D), D(A,B)+D(B,D)) \\ &= \min(8, 3 + 3) = 5 \end{aligned}$$

Dest.	Cost	Next
B	1	B
D	1	D

Dest.	Cost	Next
B	3	B
C	1	C

Distance Vector: End of 3rd Iteration

11



Node A

Dest.	Cost	Next
B	2	B
C	3	B
D	4	B

Node B

Dest.	Cost	Next
A	2	A
C	1	C
D	2	C



- Nothing changes, algorithm terminates
- Until something changes...

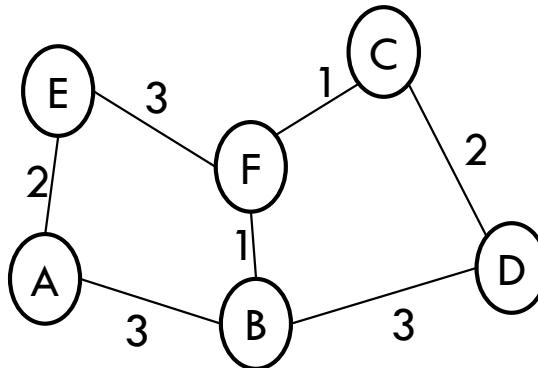
```
7. loop  
8.  
9. else  
10. for  
11. if  
12. else  
13. for  
14. if  
15.  
16. else  
17.     D(A, Y) =  
18.         min(D(A, Y),  
19.             D(A, V) + D(V, Y));  
20.     if (there is a new min. for dest. Y)  
21.         send D(A, Y) to all neighbors  
22. forever
```

Dest.	Cost	Next
A	3	B
B	1	B
D	1	D

Dest.	Cost	Next
A	4	C
B	2	C
C	1	C



Elosztott Bellman-Ford algoritmus – példa



Becsült késleltetés A-tól kezdetben

A	cost	N. Hop
B	3	B
C	∞	-
D	∞	-
E	2	E
F	∞	-

B vektora A-nak

A	3
B	0
C	∞
D	3
E	∞
F	3

E vektora A-nak

A	2
B	∞
C	∞
D	∞
E	0
F	3

Új becsült késleltetés A-tól

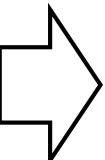
A	cost	N. Hop
B	3	B
C	∞	-
D	6	B
E	2	E
F	4	B

A vektorá B-nek és E-nek

A	0
B	3
C	∞
D	6
E	2
F	4

Új becsült késleltetés A-tól

A	cost	N. Hop
B	3	B
C	5	B
D	6	B
E	2	E
F	4	B



```

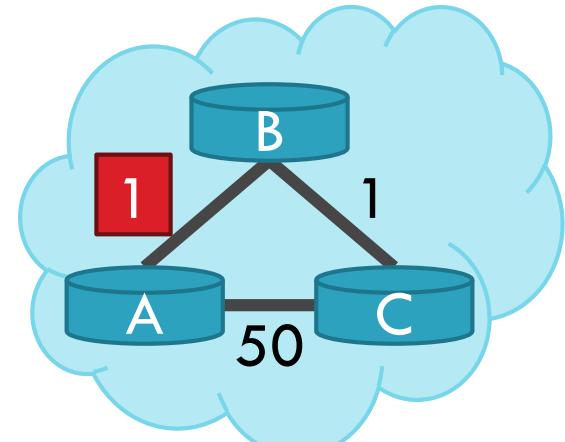
7. loop:
8.   wait (link cost update or update message)
9.   if ( $c(A, V)$  changes by  $d$ )
10.    for all destinations  $Y$  through  $V$  do
11.       $D(A, Y) = D(A, Y) + d$ 
12.   else if (update  $D(V, Y)$  received from  $V$ )
13.    for all destinations  $Y$  do
14.      if (destination  $Y$  through  $V$ )
15.         $D(A, Y) = D(A, V) + D(V, Y);$ 
16.      else
17.         $D(A, Y) = \min(D(A, Y), D(A, V) + D(V, Y));$ 

```

Link Cost
Algorithm

Good news travels fast

Algorithm
terminates



Node B

	D	C	N
A	4	A	
C	1	B	

	D	C	N
A	1	A	
C	1	B	

Node C

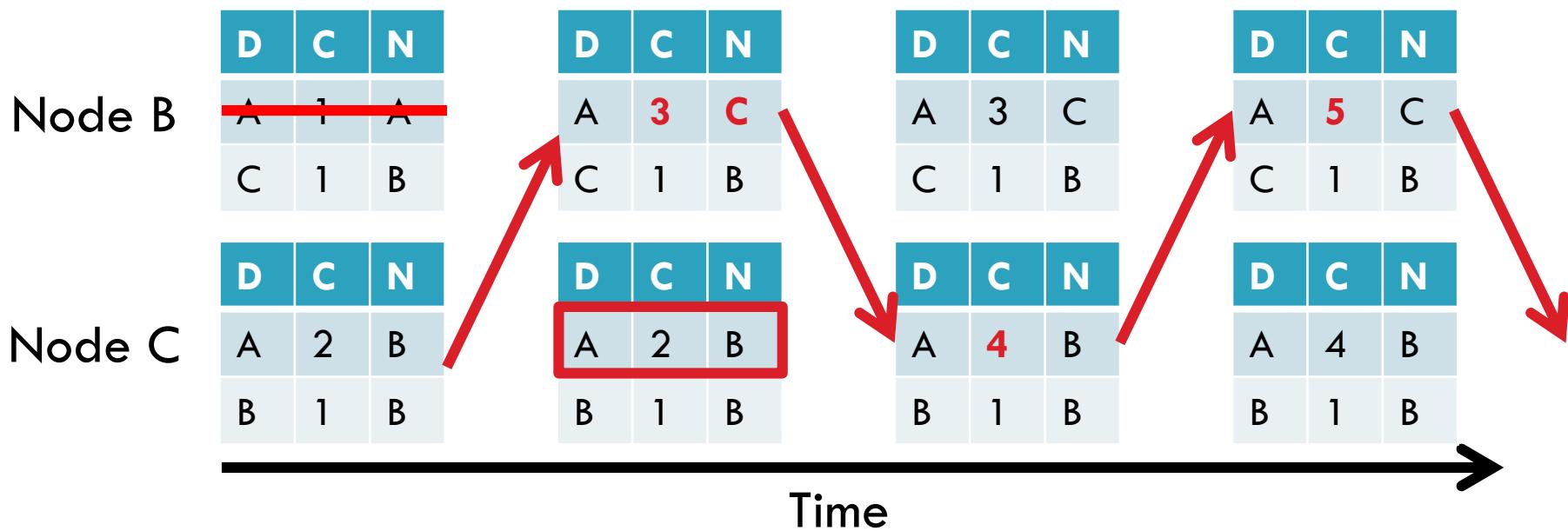
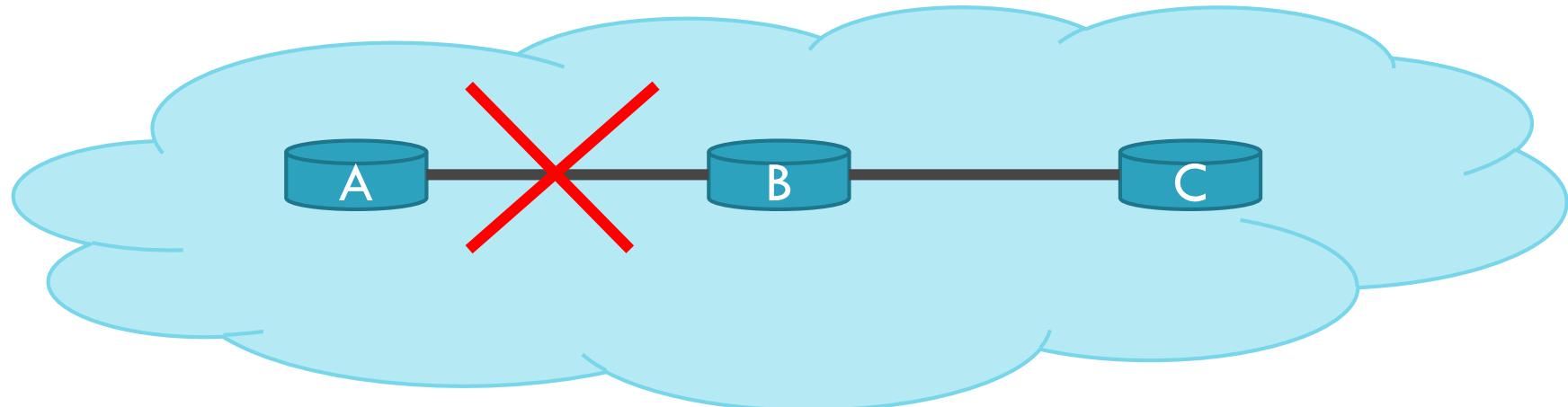
	D	C	N
A	5	B	
B	1	B	

	D	C	N
A	5	B	
B	1	B	

Time

Távolság vektor protokoll – Végtelenig számolás problémája (*count to infinity*)

14

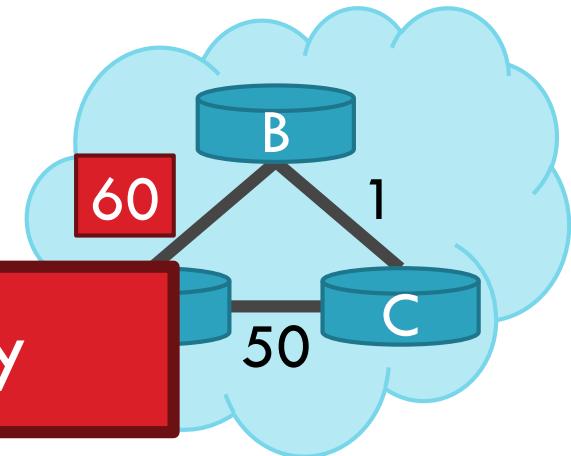


Példa - Count to Infinity Problem

15

- Node B knows $D(C, A) = 5$
- However, B does not know the path is C -> D -> A
- Thus, $D(B,A) = \infty$

Bad news travels slowly



Node B

D	C	N
A	4	A
C	1	B

D	C	N
A	6	C
C	1	B

D	C	N
A	6	C
C	1	B

D	C	N
A	8	C
C	1	B

Node C

D	C	N
A	5	B
B	1	B

D	C	N
A	5	B
B	1	B

D	C	N
A	7	B
B	1	B

D	C	N
A	7	B
B	1	B

Time

Elosztott Bellman-Ford algoritmus – Végtelenig számolás problémája

16

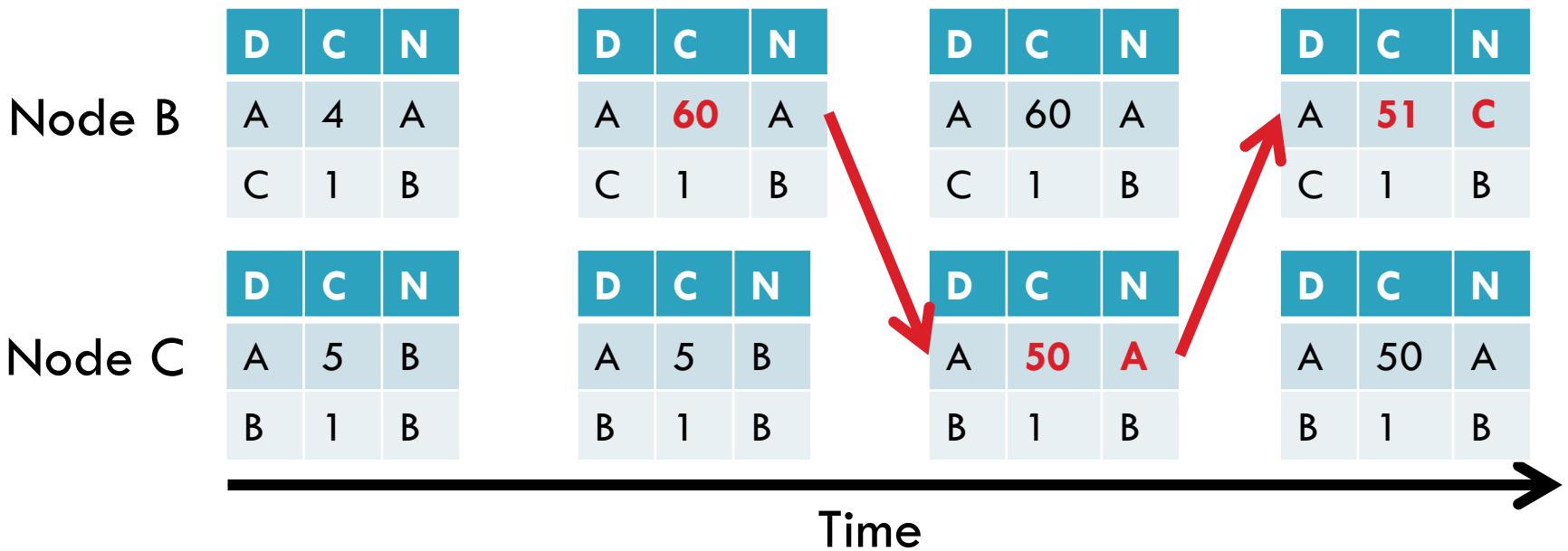
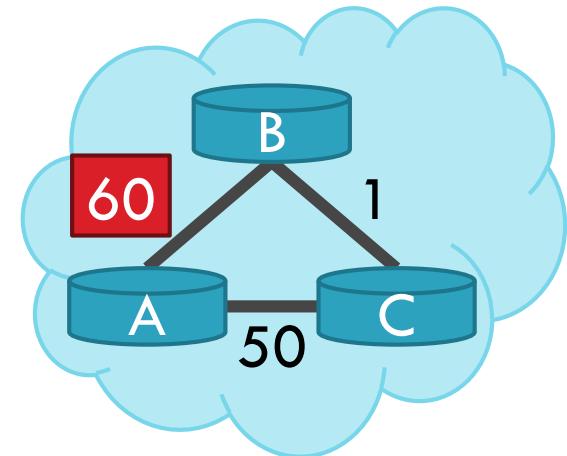
PROBLÉMA

- A „ jó hír” gyorsan terjed.
- A „rossz hír” lassan terjed.
- Azaz ciklusok keletkezhetnek.
- Lehetséges megoldás:
 - „**split horizon with poisoned reverse**”: negatív információt küld vissza arról a szomszédjának, amit tőle „tanult”. (RFC 1058)

Split horizon with Poisoned Reverse

17

- Ha C B-n keresztül irányítja a forgalmat A állomáshoz
 - ▣ C állomás B-nek $D(C, A) = \infty$ távolságot küld
 - ▣ Azaz B állomás nem fog C-n keresztül irányítani az A-ba menő forgalmat



Kapcsolatállapot alapú forgalomirányítás

Link-state routing

18

MOTIVÁCIÓ

1. Eltérő sávszélek figyelembevétele.
2. Távolság alapú algoritmusok lassan konvergáltak.

AZ ALAPÖTLET ÖT LÉPÉSBŐL TEVŐDIK ÖSSZE

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 router**-nek.
 - Dijkstra algoritmusát használják.
5. Kiszámítani a legrövidebb utat az összes többi router- hez.

Kapcsolatállapot alapú forgalomirányítás működése

19

1. A router beindulásakor az első feladat a szomszédok megismerése, ezért egy speciális HELLO csomag elküldésével éri el, amelyet minden kimenő vonalán kiküld. Elvárás, hogy a vonal másik végén lévő router válaszolt küldjön vissza, amelyben közli az azonosítóját (, ami globálisan egyedi!).
2. A késleltetés meghatározása, amelynek legközvetlenebb módja egy speciális ECHO csomag küldése, amelyet a másik oldalnak azonnal vissza kell küldenie. A körbeérési idő felével becsülhető a késleltetés. (Javítás lehet a többszöri kísérlet átlagából számított érték.)
3. Az adatok összegzése, és csomag előállítása a megismert információkról. A kapcsolatállapot tartalma: a feladó azonosítója, egy sorszám, egy korérték és a szomszédok lista. minden szomszédhoz megadják a felé tapasztalható késleltetést. Az előállítás történhet periodikusan vagy hiba esemény esetén. (Un. LSA – Link State Advertisement, azaz kapcsolatállapot hírdejtés)

Kapcsolatállapot alapú forgalomirányítás működése

20

4. A kapcsolat csomagok megbízható szétosztása. Erre használható az elárasztás módszere, viszont a csomagban van egy sorszám, amely minden küldésnél 1-gyel nő. A router-ek számon tartanak minden (forrás,sorszám) párt, amelyet látnak. Ha új érkezik, akkor azt küldik minden vonalon, kivéve azon, amin érkezett. A másod példányokat eldobják. A kisebb sorszámúakat elavultnak tekintik, és nem küldik tovább.

Probléma	Megoldás
Sorszámok egy idő után körbe érnek	32 bites sorszám használata
Router összeomlik	Kor bevezetése. A kor értéket másodpercenként csökkenti a router, ha a kor eléri a nullát, akkor el kell dobni.
A sorszám mező megsérül	

- ▣ **További finomítások:** tároló területre kerül először a csomag és nem a küldési sorba; nyugtázás

Kapcsolatállapot alapú forgalomirányítás működése

21

5. Új útvonalak számítása. Amint egy router a kapcsolatállapot csomagok egy teljes készletét összegyűjtötte, megszerkesztheti az alhálózat teljes gráfját, mivel minden kapcsolat képviselve van. Erre lefuttatható Dijkstra algoritmusa, eredményeképp pedig megkapjuk a forgalomirányító táblát.

JELLEMZŐK

- A router-ek és a router-ek szomszédinak átlagos számával arányos tárterület kell az algoritmus futtatásához. $O(kn)$, ahol k a szomszédok száma és n a router-ek száma. Azaz nagy hálózatok esetén a számítás költséges és memória igényes lesz.
- A hardver- és szoftver-problémák komoly gondot okozhatnak. A hálózat méretének növekedésével a hiba valószínűsége is nő.

Dijkstra algoritmus (1959)

22

- Statikus algoritmus
- **Cél:** két csomópont közötti legrövidebb út meghatározása.

INFORMÁLIS LEÍRÁS

- minden csomópontot felcímkézünk a forrás csomóponttól való legrövidebb ismert út mentén mért távolságával.
 - Kezdetben a 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.

Dijkstra algoritmus pszeudo-kód

23

Dijkstra(G,s,w)

Output: egy legrövidebb utak fája $T=(V,E')$ G-ben s gyökérrel

```
01  $E' := \emptyset;$ 
02  $ready[s] := true;$ 
03  $ready[v] := false; \forall v \in V \setminus \{s\};$ 
04  $d[s] := 0;$ 
05  $d[v] := \infty; \forall v \in V \setminus \{s\};$ 
06 priority_queue Q;
07 forall  $v \in Adj[s]$  do
08   pred[v] := s;
09   d[v] := w(s,v);
10  Q.Insert(v,d[v]);
11 od
12 while  $Q \neq \emptyset$  do
```

INICIALIZÁCIÓS FÁZIS

```
13   $v := Q.DeleteMin();$ 
14   $E' := E' \cup \{(pred[v],v)\};$ 
15  ready[v] := true;
16  forall  $u \in Adj[v]$  do
17    if  $u \in Q$  and  $d[v] + w(v,u) < d[u]$  then
18      pred[u] := v;
19      d[u] := d[v] + w(v,u);
20      Q.DecreasePriority(u,d[u]);  
JAVÍTÓ ÚT
21  else if  $u \notin Q$  and not ready[u] then
22    pred[u] := v;
23    d[u] := d[v] + w(v,u);
24    Q.Insert(u,d[u]);  
ÚJ ÚT
25  fi
26 od
27 od
```

ITERÁCIÓS LÉPÉSEK

OSPF vs. IS-IS

- Két eltérő implementáció a link-state routing stratégiának

OSPF

- Cégek és adatközpontok
- Több lehetőséget támogat
- IPv4 felett
 - LSA-k IPv4 feletti küldése
 - OSPFv3 szükséges az IPv6-hoz

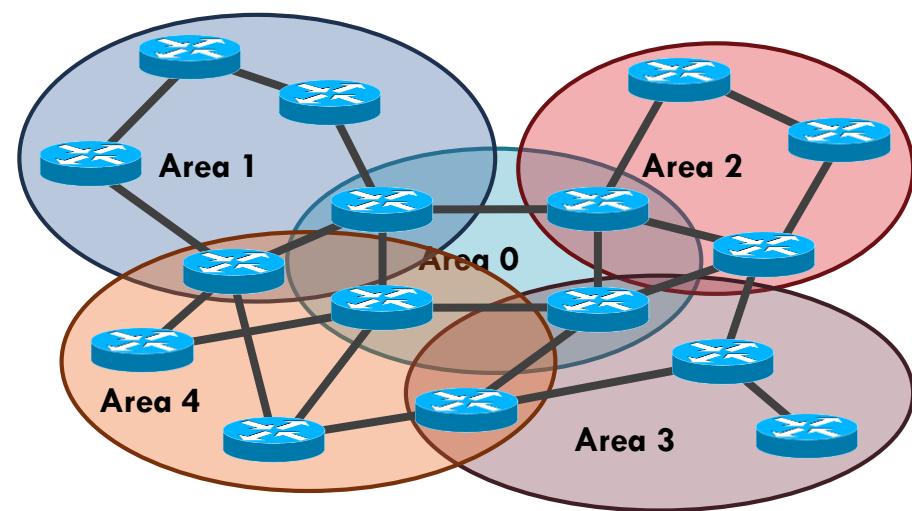
IS-IS

- Internet szolgáltatók által használt
- **Sokkal tömörebb**
 - Kisebb hálózati overhead
 - Több eszközt támogat
- Nem kötődik az IP-hez
 - Működik mind IPv4-gyel és IPv6-tal

Eltérő felépítés

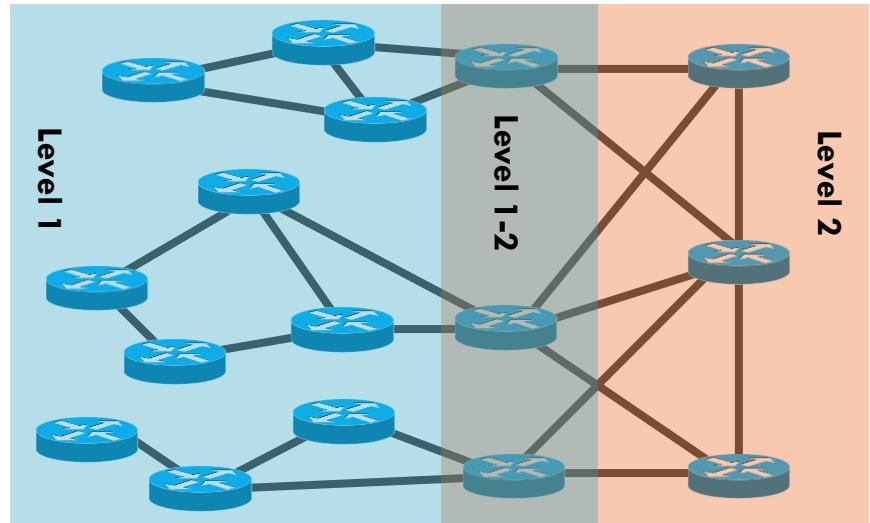
OSPF

- Átfedő területek köré szerveződik
- Area 0 a hálózat magja



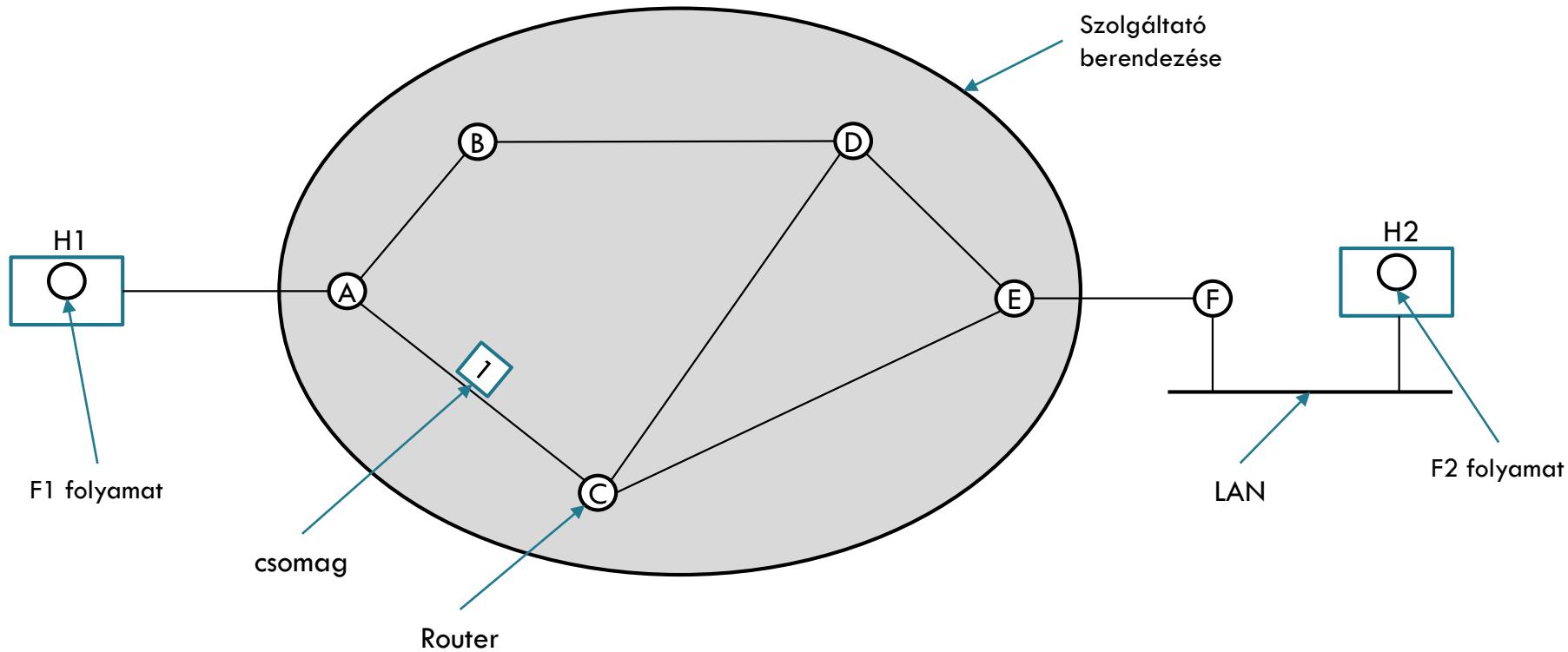
IS-IS

- 2-szintű hierarchia
- A 2. szint a gerinchálózat



Hálózati réteg protokolljai - Környezet

26



Szállítási réteg felé nyújtott szolgálatok

27

VEZÉRELVEK

1. A szolgálat legyen független az alhálózat kialakításától.
2. A szállítási réteg felé el kell takarni a jelenlevő alhálózatok számát, típusát és topológiáját.
3. A szállítási réteg számára rendelkezésre bocsátott hálózati címeknek egységes számozási rendszert kell alkotniuk, még LAN-ok és WAN-ok esetén is.

SZOLGÁLATOK KÉT FAJTÁJÁT KÜLÖNBÖZTETIK MEG

- Összeköttetés nélküli szolgálat (*Internet*)
 - datagram alhálózat
- Összeköttetés alapú szolgálat (ATM)
 - virtuális áramkör alhálózat



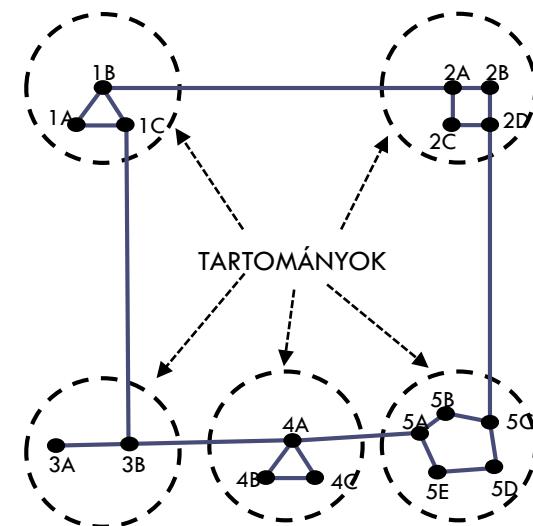
HÁLÓZATI RÉTEG – FORGALOMIRÁNYÍTÁS

Hierarchikus forgalomirányítás

29

MOTIVÁCIÓ

- A hálózat méretének növekedésével a router-ek forgalomirányító táblázatai is arányosan nőnek.
 - A memória, a CPU és a sávszélesség igény is megnövekszik a router-eknél.
- Ötlet: telefonhálózatokhoz hasonlóan hierarchikus forgalomirányítás alkalmazása.

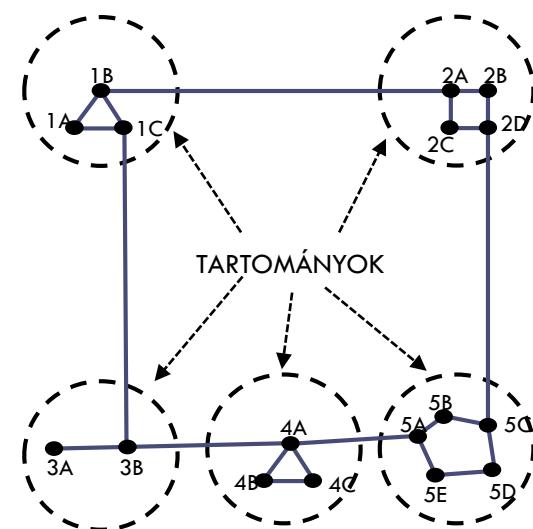


Hierarchikus forgalomirányítás

30

JELLEMZŐK

- A router-eket tartományokra osztjuk. A saját tartományát az összes router ismeri, de a többi belső szerkezetéről nincs tudomása.
- Nagy hálózatok esetén többszintű hierarchia lehet szükséges.
- N darab router-ből álló alhálózathoz az optimális szintek száma $\ln N$, amely router-enként $e * \ln N$ bejegyzést igényel. (Kamoun és Kleinrock, 1979)



Adatszóró forgalomirányítás

31

- **Adatszórás** (vagy angolul *broadcasting*) – egy csomag mindenhol történő egyidejű küldése.
- Több féle megvalósítás lehetséges:

1. Külön csomag küldése minden egyes rendeltetési helyre

- sávszélesség pazarlása, lista szükséges hozzá

2. Elárasztás.

- kétpontos kommunikációhoz nem megfelelő

Adatszóró forgalomirányítás

32

3. **Többcélú forgalomirányítás** (vagy angolul *multidestination routing*). Csomagban van egy lista a rendeltetési helyekről, amely alapján a router-ek eldöntik a vonalak használatát, minden egyik vonalhoz készít egy másolatot és belerakja a megfelelő célcím listát.
4. **A forrás router-hez tartozó nyelőfa használata.** A feszítőfa (vagy angolul *spanning tree*) az alhálózat részhalmaza, amelyben minden router benne van, de nem tartalmaz köröket. Ha minden router ismeri, hogy mely vonalai tartoznak a feszítőfához, akkor azokon továbbítja az adatszóró csomagot, kivéve azon a vonalon, amelyen érkezett.
 - nem mindig ismert a feszítőfa

Adatszóró forgalomirányítás 2/2

33

5. Visszairányú továbbítás (vagy angolul *reverse path forwarding*). Amikor egy adatszórásos csomag megérkezik egy routerhez, a router ellenőrzi, hogy azon a vonalon kapta-e meg, amelyen rendszerint ő szokott az adatszórás forrásához küldeni. Ha igen, akkor nagy esély van rá, hogy az adatszórásos csomag a legjobb utat követte a router-től, és ezért ez az első másolat, amely megérkezett a router-hez. Ha ez az eset, a router kimásolja minden vonalra, kivéve arra, amelyiken érkezett. Viszont, ha az adatszórásos csomag más vonalon érkezett, mint amit a forrás eléréséhez előnyben részesítünk, a csomagot eldobják, mint valószínű másodpéldányt.

Többes-küldéses forgalomirányítás

34

- **Többes-küldés** (vagy angolul *multicasting*) – egy csomag meghatározott csoporthoz történő egyidejű küldése.

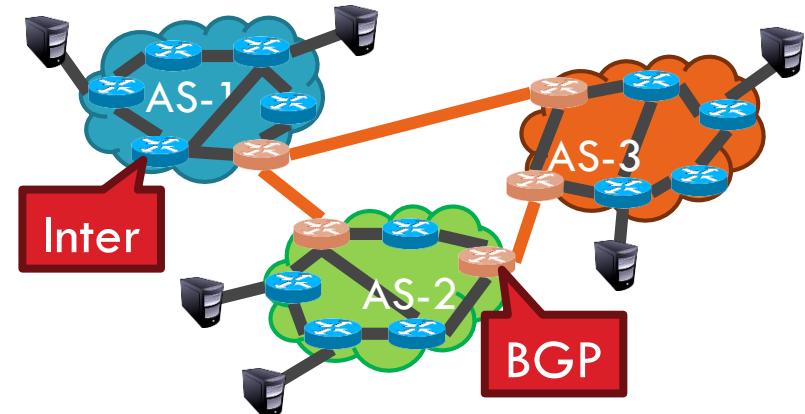
MULTICAST ROUTING

- Csoport kezelés is szükséges hozzá: létrehozás, megszüntetés, csatlakozási lehetőség és leválasztási lehetőség. (Ez nem a forgalomirányító algoritmus része!)
- minden router kiszámít egy az alhálózatban az összes többi routert lefedő feszítőfát.
- Többes-küldéses csomag esetén az első router levágja a feszítőfa azon ágait, amelyek nem csoporton belüli hoszthoz vezetnek. A csomagot csak a csonkolt feszítőfa mentén továbbítják.

Hierarchikus forgalomirányítás IP

35

- Hierarchikus (2 szintű)
 - AS-ek közötti:
 - EGP
 - Exterior Gateway Protocols
 - Tartományok közötti
 - AS-en belüli
 - IGP
 - Interior Gateway Protocols
 - Tartományon belüli
- AS – Autonom System – Autonóm Rendszer



Hálózati réteg az Interneten

36

- 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.

Hálózati réteg az Interneten

37

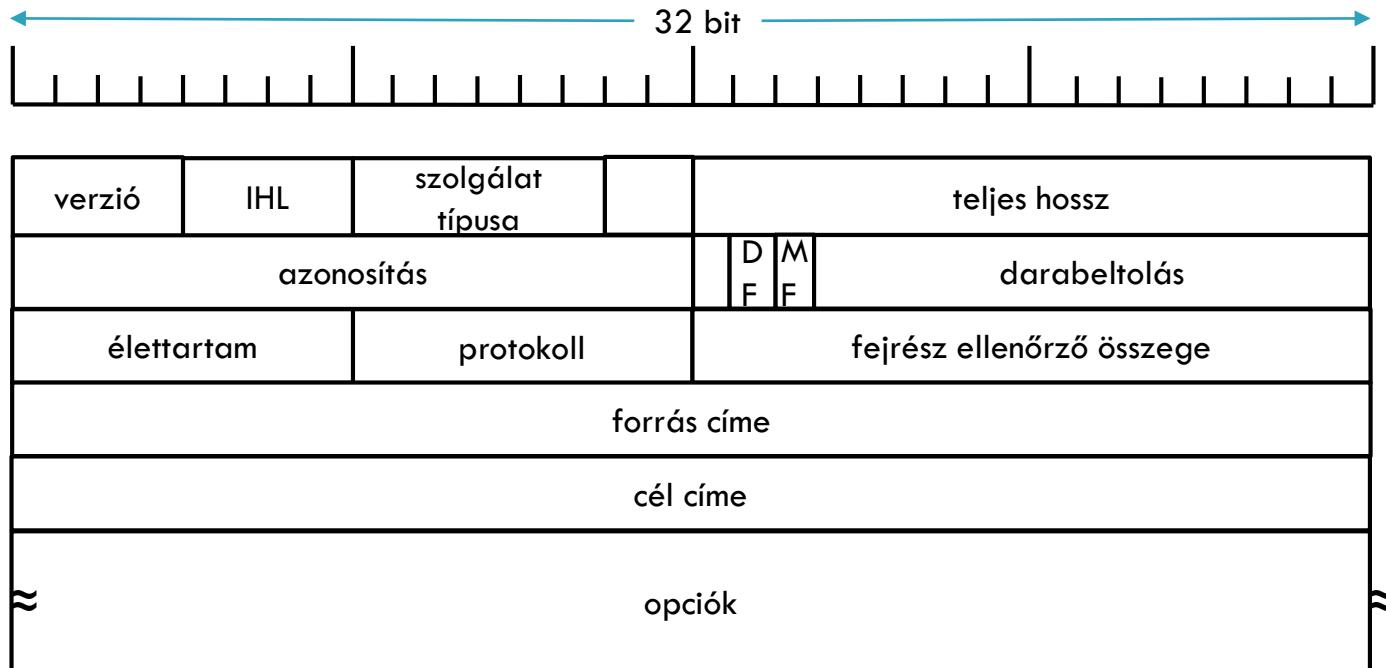
- Az Interneten a kommunikáció az alábbi módon működik:
 1. A szállítási réteg viszi az adatfolyamokat és datagramokra tördeli azokat.
 2. minden datagram átvitelre kerül az Interneten, esetleg menet közben kisebb egységekre darabolva.
 3. A célgép hálózati rétege összeállítja az eredeti datagramot,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



HÁLÓZATI RÉTEG – CÍMZÉS

Az IPv4 fejrésze

39



Az IP fejrésze

40

- **verzió:** IP melyik verzióját használja (jelenleg 4 és 6 közötti átmenet zajlik)
- **IHL:** a fejléc hosszát határozza meg 32-bites szavakban mérve, legkisebb értéke 5.
- **szolgálat típusa:** szolgálati osztályt jelöl (3-bites precedencia, 3 jelzőbit [D,T,R])
- **teljes hossz:** fejléc és adatrész együttes hossza bájtokban
- **azonosítás:** egy datagram minden darabja ugyanazt az azonosítás értékét hordozza.
- **DF:** „ne darabold” flag a router-eknek
- **MF:** „több darab” flag minden darabban be kell legyen állítva, kivéve az utolsót.
- **darabeltolás:** a darab helyét mutatja a datagramon belül. (elemi darab méret 8 bájt)

Az IP fejrésze

41

- **élettartam:** másodpercenként kellene csökkenteni a mező értékét, minden ugrásnál csökkentik eggyel az értékét
- **protokoll:** szállítási réteg protokolljának azonosítóját tartalmazza
- **ellenőrző összeg:** a router-eken 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 (később tárgyaljuk részletesen)
- **opciók:** következő verzió bővíthetősége miatt hagyták benne. Eredetileg 5 opció volt. (router-ek általában figyelmen kívül hagyják)

Címzés

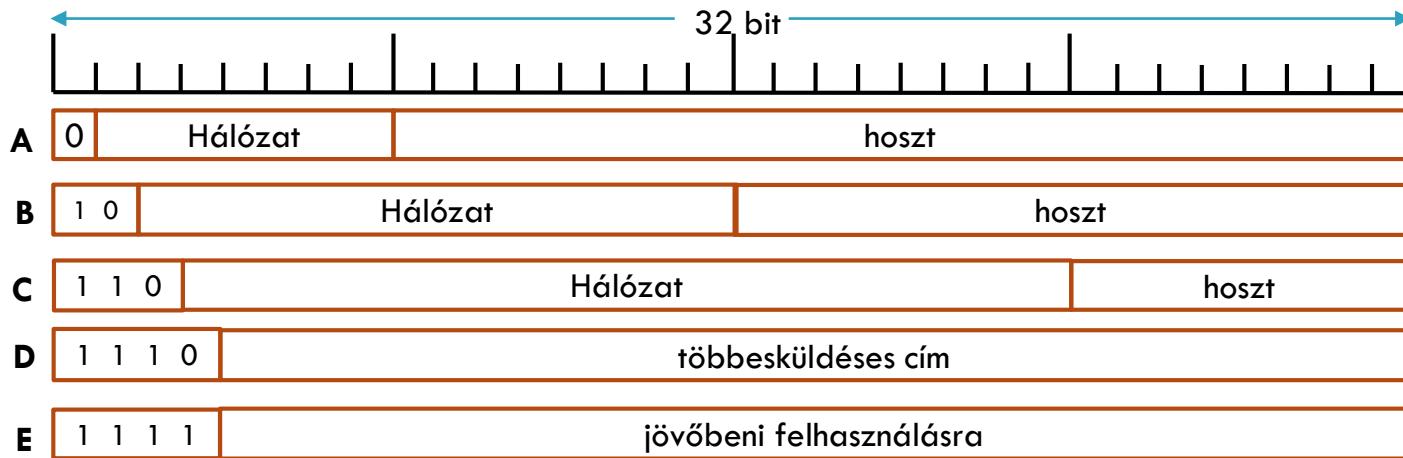
42



IP cím

43

- minden hoszt és minden router az Interneten rendelkezik egy IP-címmel, amely a hálózat számát és a hoszt számát kódolja. (egyedi kombináció)
- 4 bájton ábrázolják az IP-címet.
- több évtizeden keresztül 5 osztályos címzést használtak: A, B, C, D és E.



IP cím

44

- Az IP-t pontokkal elválasztott decimális rendszerben írják. Például: 192.168.0.1
 - Van pár speciális cím. Lásd az alábbiakban.

Ez egy hoszt.

0..0

hoszt

Ez egy hoszt ezen hálózaton.

1 1

Adatszórás a helyi hálózaton.

Hálózat

1..1

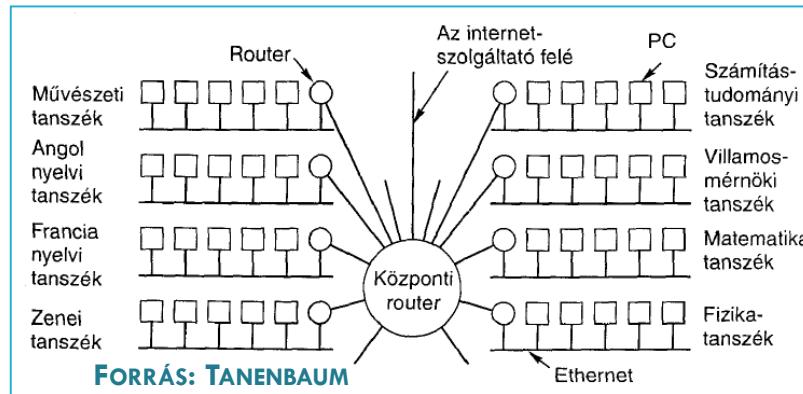
Adatszórás egy távoli hálózaton.

0 1 1 1 1 1 1

(bármí)

IP cím – alhálózatok

45



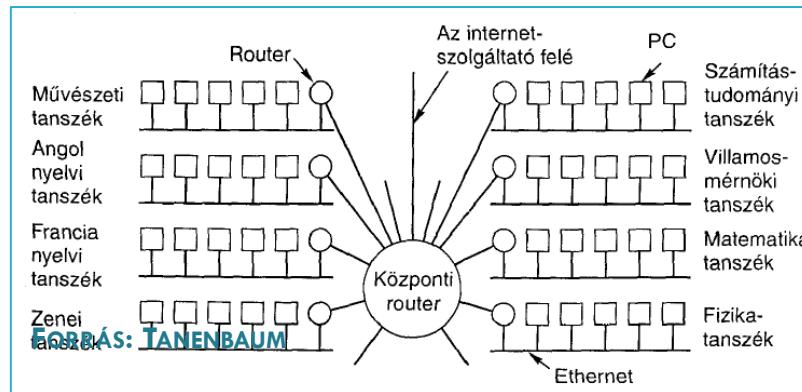
- Az azonos hálózatban lévő hosztok ugyanazzal a hálózatszámmal rendelkeznek.
- Egy hálózat belső felhasználás szempontjából több részre osztódhat, de a külvilág számára egyetlen hálózatként jelenik meg.
- **Alhálózat (avagy angolul subnet)**

IP cím – alhálózatok

46

AZONOSÍTÁS

- alhálózati maszk (avagy angolul *subnet mask*) ismerete kell a routernek
 - Két féle jelölés *IP-cím jellegű* vagy a *fix pozíciók száma*.
- A forgalomirányító táblázatba a router-eknél (*hálózat,0*) és (*saját hálózat, hoszt*) alakú bejegyzések.
- Ha nincs találat, akkor az alapértelmezett router felé továbbítják a csomagot.



IP cím – CIDR

47

- IP címek gyorsan fogytak. 1996-ban kötötték be a 100.000-edik hálózatot.
 - Az osztályok használata sok címet elpazarolt. (B osztályú címek népszerűsége)
- **Megoldás:** osztályok nélküli környezetek közötti forgalomirányítás (CIDR).
 - Például 2000 cím igénylése esetén 2048 méretű blokk kiadása.
- 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.
- Túl sok bejegyzés keletkezik.
 - Csoportos bejegyzések használata.

CIDR címzés példa

48

Mi történik, ha a router egy 135.46.57.14 IP cím felé tartó csomagot kap?

/22-ES CÍM ESETÉN

10001011 00101110 00111001 00001110
AND 11111111 11111111 11111100 00000000
10001011 00101110 00111000 00000000

Kimaszkolás eredménye

/23-ES CÍM ESETÉN

10001011 00101110 00111001 00001110
AND 11111111 11111111 11111110 00000000
10001011 00101110 00111000 00000000

- Vagyis 135.46.56.0/22-as vagy 135.46.56.0/23-as bejegyzést kell találni, azaz jelen esetben a 0.interface felé történik a továbbítás.

Cím/maszk	Következő ugrás
135.46.56.0/22	0.interface
135.46.60.0/23	1.interface
192.53.40.0/23	1.router
Alapértelmezett	2.router

CIDR bejegyzés aggregálás példa

49

- Lehet-e csoportosítani a következő bejegyzéseket, ha feltesszük, hogy a következő ugrás mindegyiknél az 1.router: 57.6.96.0/21, 57.6.104.0/21, 57.6.112.0/21, 57.6.120.0/21?

00111001 00000110 01100	000 00000000
00111001 00000110 01101	000 00000000
00111001 00000110 01110	000 00000000
00111001 00000110 01111	000 00000000

- Azaz az (57.6.96.0/19, 1.router) bejegyzés megfelelően csoportba fogja a 4 bejegyzést.

Forgalomirányítási tábla példa

50

Network Destination	Netmask	Gateway	Interface	Metric
0.0.0.0	0.0.0.0	192.168.0.1	192.168.0.100	10
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
192.168.0.0	255.255.255.0	192.168.0.100	192.168.0.100	10
192.168.0.100	255.255.255.255	127.0.0.1	127.0.0.1	10
192.168.0.255	255.255.255.255	192.168.0.100	192.168.0.100	10

- Gyors javítás az IP címek elfogyásának problémájára. (hálózati címfordítás)

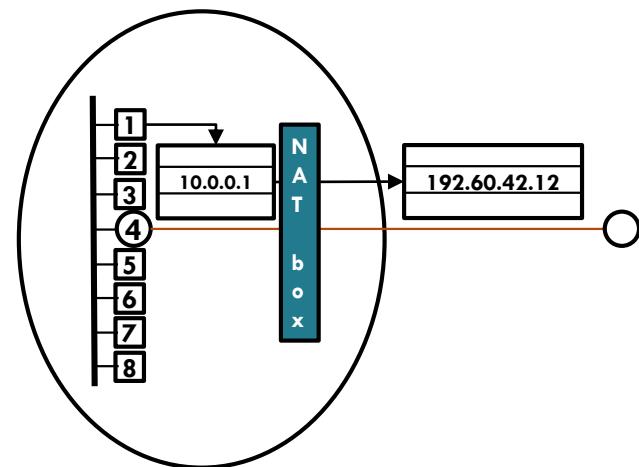
ALAPELVEK

- Az internet forgalomhoz minden cégnak egy vagy legalábbis kevés IP-címet adnak. A vállalaton belül minden számítógéphez egyedi IP-címet használnak a belső forgalomirányításra.
- A vállalaton kívüli csomagokban a címfordítást végzünk.
- 3 IP-címtartományt használunk:
 - 10.0.0.0/8, azaz 16 777 216 lehetséges hoszt;
 - 172.16.0.0/12, azaz 1 084 576 lehetséges hoszt;
 - 192.168.0.0/16, azaz 65 536 lehetséges hoszt;
- NAT box végzi a címfordítást

NAT

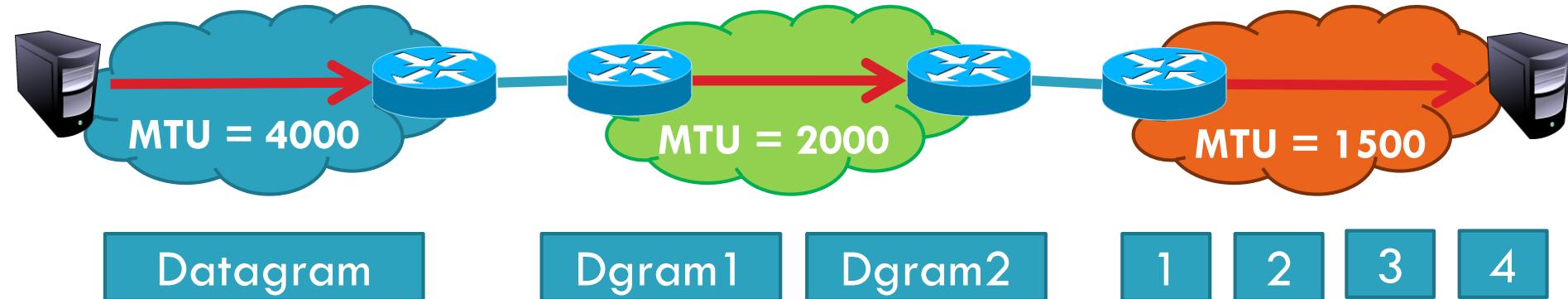
52

- Hogyan fogadja a választ?
 - A port mezők használata, ami mind a TCP, mind az UDP fejlécben van
 - Kimenő csomagnál egy mutatót tárolunk le, amit beírunk a forrás port mezőbe. 65536 bejegyzésből álló fordítási táblázatot kell a NAT box-nak kezelni.
 - A fordítási táblázatban benne van az eredeti IP és forrás port.
- **Ellenérvek:** sérti az IP architekturális modelljét, összeköttetés alapú hálózatot képez, rétegmodell alapelveit sérti, kötöttség a TCP és UDP fejléchez, szöveg törzsében is lehet az IP, szűkös port tartomány



IP Fragmentation – IP Fragmentáció (darabolás)

53

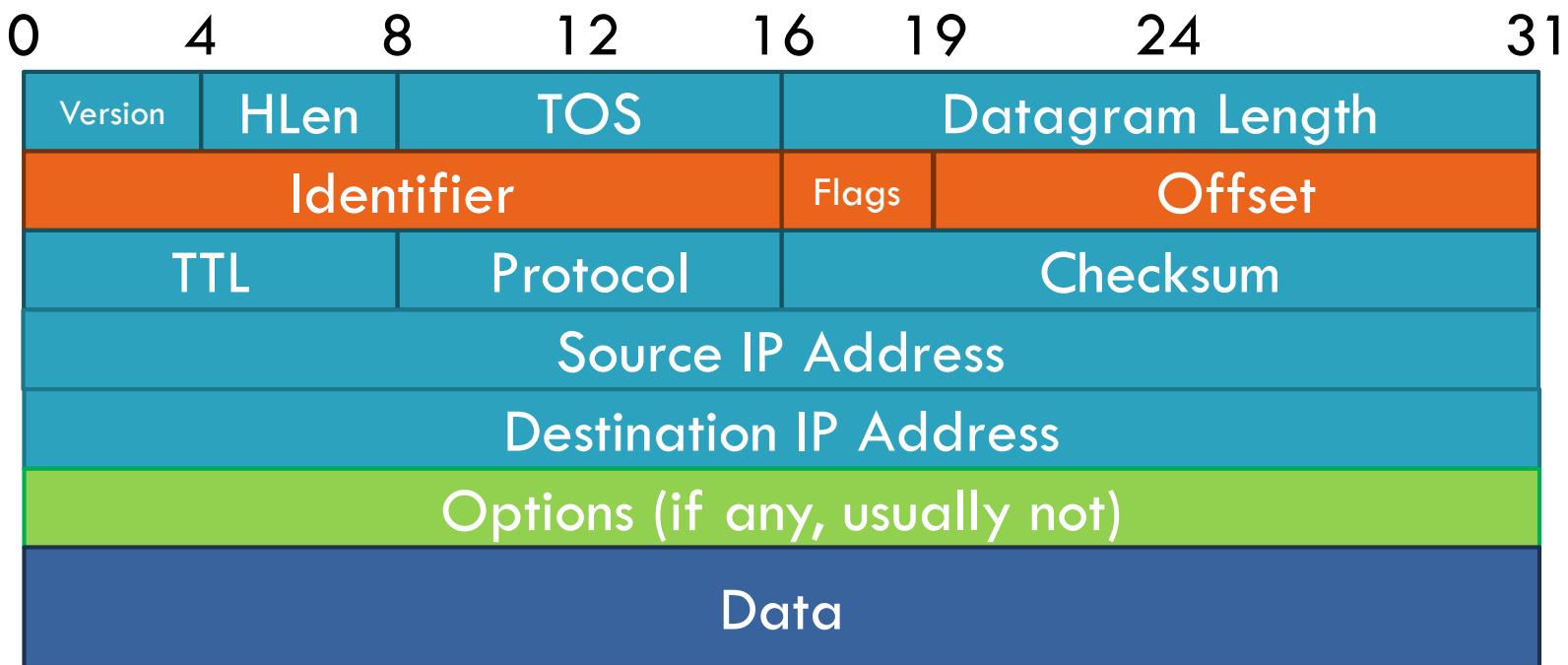


- Probléma: minden hálózatnak megvan a maga MTU-ja
 - MTU: Maximum Transmission Unit – lényegében a maximális használható csomag méret egy hálózatban
 - DARPA/Internet alapelv: hálózatok heterogének lehetnek
 - A minimális MTU nem ismert egy adott útvonalhoz
- IP esetén: fragmentáció
 - Vágjuk szét az IP csomagot, amikor az MTU csökken
 - Állítsuk helyre a darabokból a csomagot a fogadó állomásnál

IP fejléc: 2. szó

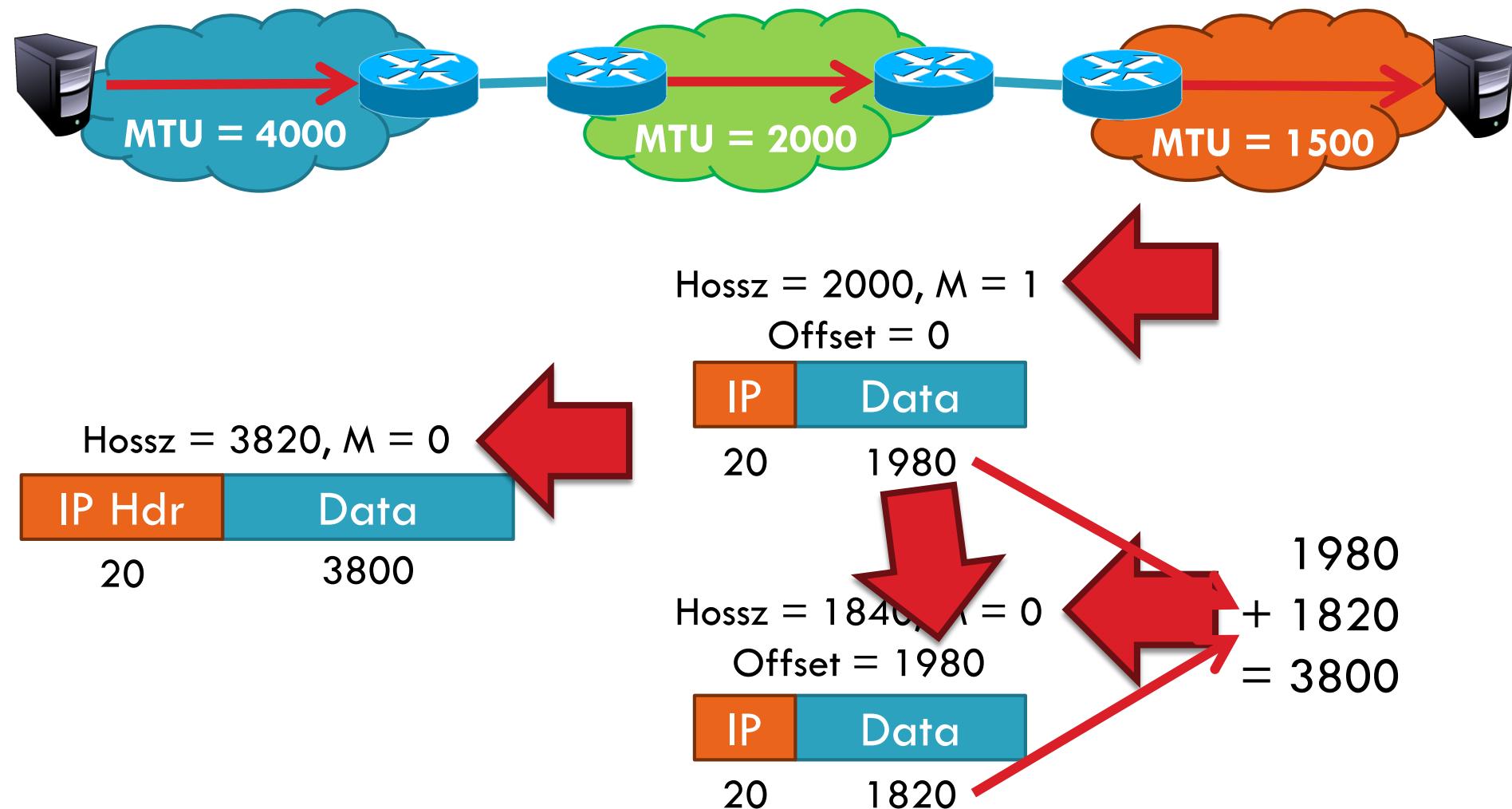
54

- Identifier (azonosító):
 - egyedi azonosító minden IP datagramhoz (csomaghoz)
- Flags (jelölő bitek):
 - M flag, értéke 0, ha ez az utolsó darab/fragment, különben 1
- Offset (eltolás):
 - A darab/fragment első bájtjának pozíciója



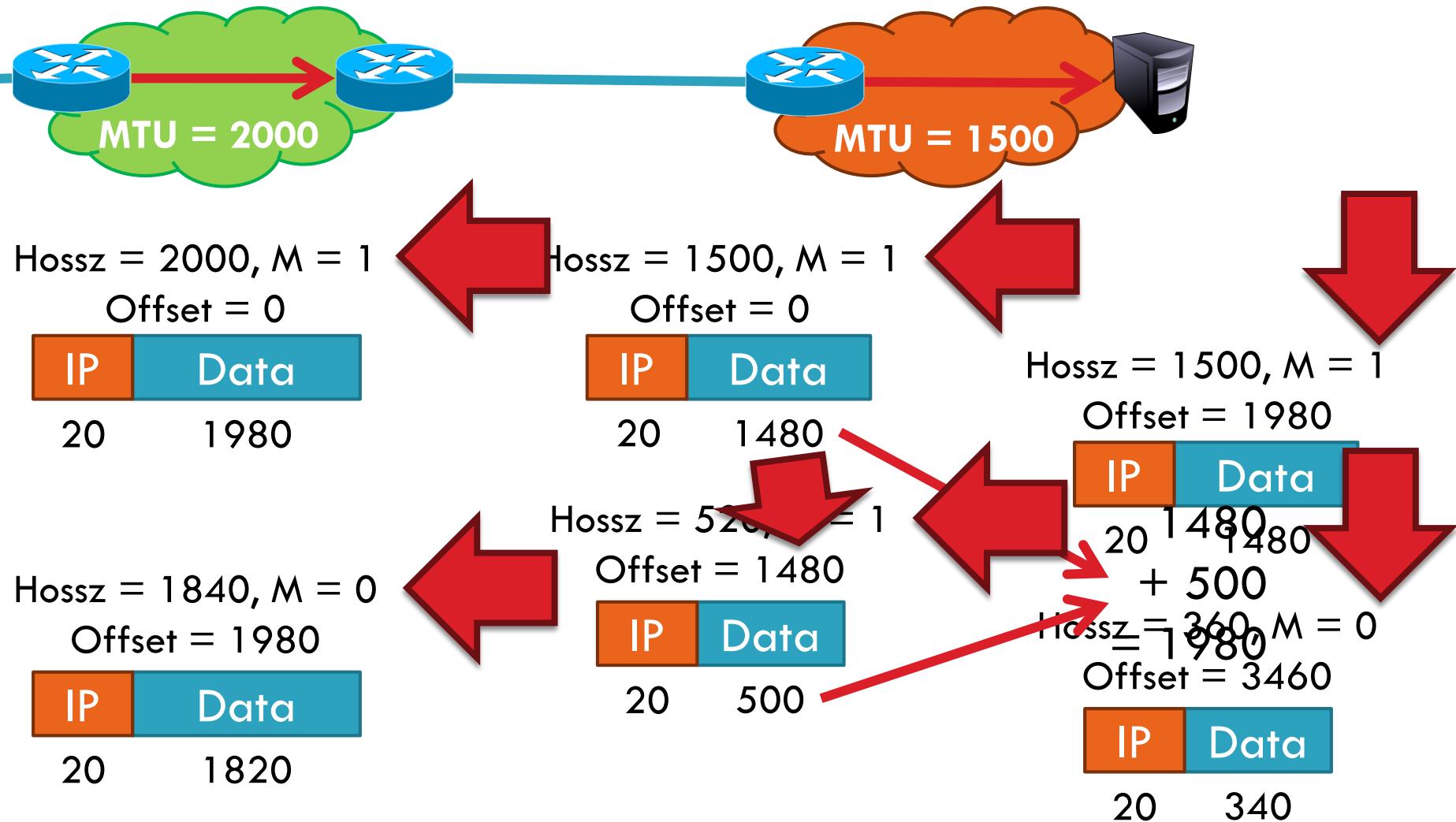
Példa

55



Példa

56



IP csomag helyreállítása

57

Hossz = 1500, M = 1, Offset = 0

IP	Data
20	1480

Hossz = 520, M = 1, Offset = 1480

IP	Data
20	500

Hossz = 1500, M = 1, Offset = 1980

IP	Data
20	1480

Hossz = 360, M = 0, Offset = 3460

IP	Data
20	340

- A végponton történik
- M = 0, akkor ebből a darabból tudjuk a teljes adatmennyiséget
 - Hossz – IPHDR_hossz + Offset
 - 360 – 20 + 3460 = 3800
- Kihívások:
 - Nem sorrendben beérkező darabok
 - Duplikátumok
 - Hiányzó darabok
- Memória kezelés szempontjából egy rémálom...

Fragmentáció

58

- Az Internet esetén
 - Elosztott és heterogén
 - minden hálózat maga választ MTU-t
 - Kapcsolat nélküli datagram/csomag alapú protokoll
 - minden darab tartalmazza a továbbításhoz szükséges összes információt
 - A darabok függetlenül kerülnek leszállításra, akár különböző útvonalon keresztül
 - Legjobb szándék elve szerint (best effort)
 - A router-ek és a fogadó is eldobhat darabokat
 - Nem követelmény a küldő értesítése a „hibáról”
 - A legtöbb feladat a végpontra hárul
 - Csomag helyreállítása a darabokból

Fregmantáció a valóságban

59

- A fragmentáció költséges
 - ▣ Memória és CPU költés a csomag visszaállításához
 - ▣ Ha lehetséges, el kell kerülni
- MTU felderítő protokoll
 - ▣ Csomagküldés a “don’t fragment” flag bittel
 - ▣ Folyamatosan csökkentjük a csomag méretét, amíg egy meg nem érkezik
 - ▣ Lehetséges “can’t fragment” hiba egy routertől, ami közvetlenül tartalmazza az adott hálózatban használt MTU-t
- Darabok kezelését végző router
 - ▣ Gyors, specializált hardver megoldás
 - ▣ Dedikált erőforrás a darabok kezeléséhez

Köszönöm a figyelmet!

Számítógépes Hálózatok

8. Előadás: Hálózati réteg 2

Based on slides from **Zoltán Ács ELTE** and D. Choffnes Northeastern U., Philippa Gill from StonyBrook University , Revised Spring 2016 by S. Laki

Hálózati réteg

2



- Szolgáltatás
 - ▣ Csomagtovábbítás
 - ▣ Útvonalválasztás
 - ▣ Csomag fragmentálás kezelése
 - ▣ Csomag ütemezés
 - ▣ Puffer kezelés
- Interfész
 - ▣ Csomag küldése egy adott végpontnak
- Protokoll
 - ▣ Globálisan egyedi címeket definiálása
 - ▣ Routing táblák karbantartása
- Példák: Internet Protocol (IPv4), IPv6

IP cím – CIDR

3

- IP címek gyorsan fogytak. 1996-ban kötötték be a 100.000-edik hálózatot.
 - Az osztályok használata sok címet elpazarolt. (B osztályú címek népszerűsége)
- **Megoldás:** osztályok nélküli környezetek közötti forgalomirányítás (CIDR).
 - Például 2000 cím igénylése esetén 2048 méretű blokk kiadása.
- 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.
- Túl sok bejegyzés keletkezik.
 - Csoportos bejegyzések használata.

CIDR címzés példa

4

Mi történik, ha a router egy 135.46.57.14 IP cím felé tartó csomagot kap?

/22-ES CÍM ESETÉN

10001011 00101110 00111001 00001110
AND 11111111 11111111 11111100 00000000
10001011 00101110 00111000 00000000

Kimaszkolás eredménye

/23-ES CÍM ESETÉN

10001011 00101110 00111001 00001110
AND 11111111 11111111 11111110 00000000
10001011 00101110 00111000 00000000

- Vagyis 135.46.56.0/22-as vagy 135.46.56.0/23-as bejegyzést kell találni, azaz jelen esetben a 0.interface felé történik a továbbítás.

Cím/maszk	Következő ugrás
135.46.56.0/22	0.interface
135.46.60.0/23	1.interface
192.53.40.0/23	1.router
Alapértelmezett	2.router

CIDR bejegyzés aggregálás példa

5

- Lehet-e csoportosítani a következő bejegyzéseket, ha feltesszük, hogy a következő ugrás mindegyiknél az 1.router: 57.6.96.0/21, 57.6.104.0/21, 57.6.112.0/21, 57.6.120.0/21?

00111001 00000110 01100	000 00000000
00111001 00000110 01101	000 00000000
00111001 00000110 01110	000 00000000
00111001 00000110 01111	000 00000000

- Azaz az (57.6.96.0/19, 1.router) bejegyzés megfelelően csoportba fogja a 4 bejegyzést.

Forgalomirányítási tábla példa

6

Network Destination	Netmask	Gateway	Interface	Metric
0.0.0.0	0.0.0.0	192.168.0.1	192.168.0.100	10
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
192.168.0.0	255.255.255.0	192.168.0.100	192.168.0.100	10
192.168.0.100	255.255.255.255	127.0.0.1	127.0.0.1	10
192.168.0.255	255.255.255.255	192.168.0.100	192.168.0.100	10

NAT

7

- Gyors javítás az IP címek elfogyásának problémájára. (hálózati címfordítás)

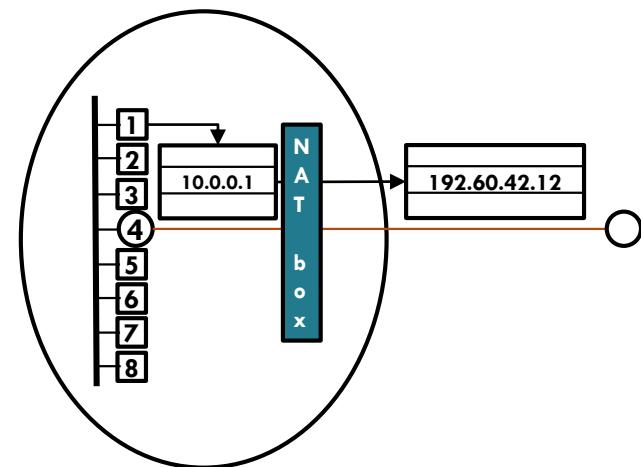
ALAPELVEK

- Az internet forgalomhoz minden cégnek egy vagy legalábbis kevés IP-címet adnak. A vállalaton belül minden számítógéphez egyedi IP-címet használnak a belső forgalomirányításra.
- A vállalaton kívüli csomagokban a címfordítást végzünk.
- 3 IP-címtartományt használunk:
 - 10.0.0.0/8, azaz 16 777 216 lehetséges hoszt;
 - 172.16.0.0/12, azaz 1 084 576 lehetséges hoszt;
 - 192.168.0.0/16, azaz 65 536 lehetséges hoszt;
- NAT box végzi a címfordítást

NAT

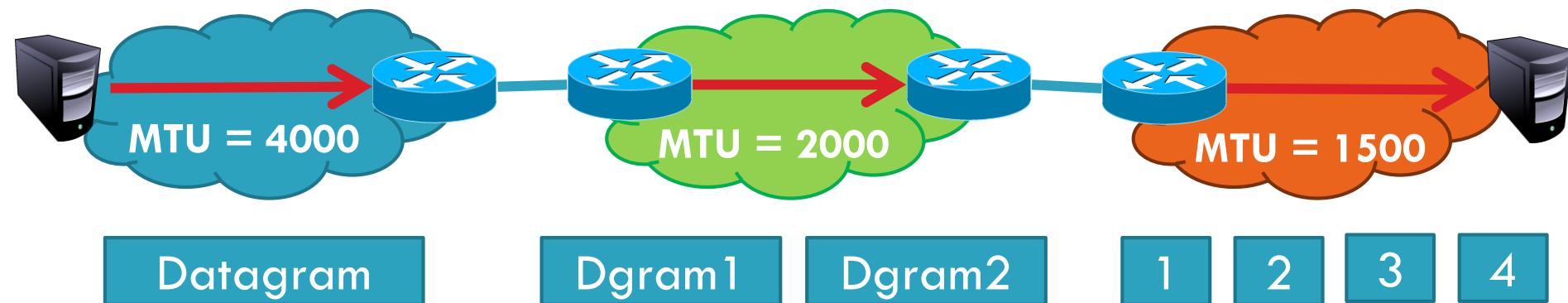
8

- Hogyan fogadja a választ?
 - A port mezők használata, ami mind a TCP, mind az UDP fejlécben van
 - Kimenő csomagnál egy mutatót tárolunk le, amit beírunk a forrás port mezőbe. 65536 bejegyzésből álló fordítási táblázatot kell a NAT box-nak kezelni.
 - A fordítási táblázatban benne van az eredeti IP és forrás port.
- **Ellenérvek:** sérti az IP architekturális modelljét, összeköttetés alapú hálózatot képez, rétegmodell alapelveit sérti, kötöttség a TCP és UDP fejléchez, szöveg törzsében is lehet az IP, szűkös port tartomány



IP Fragmentation – IP Fragmentáció (darabolás)

9

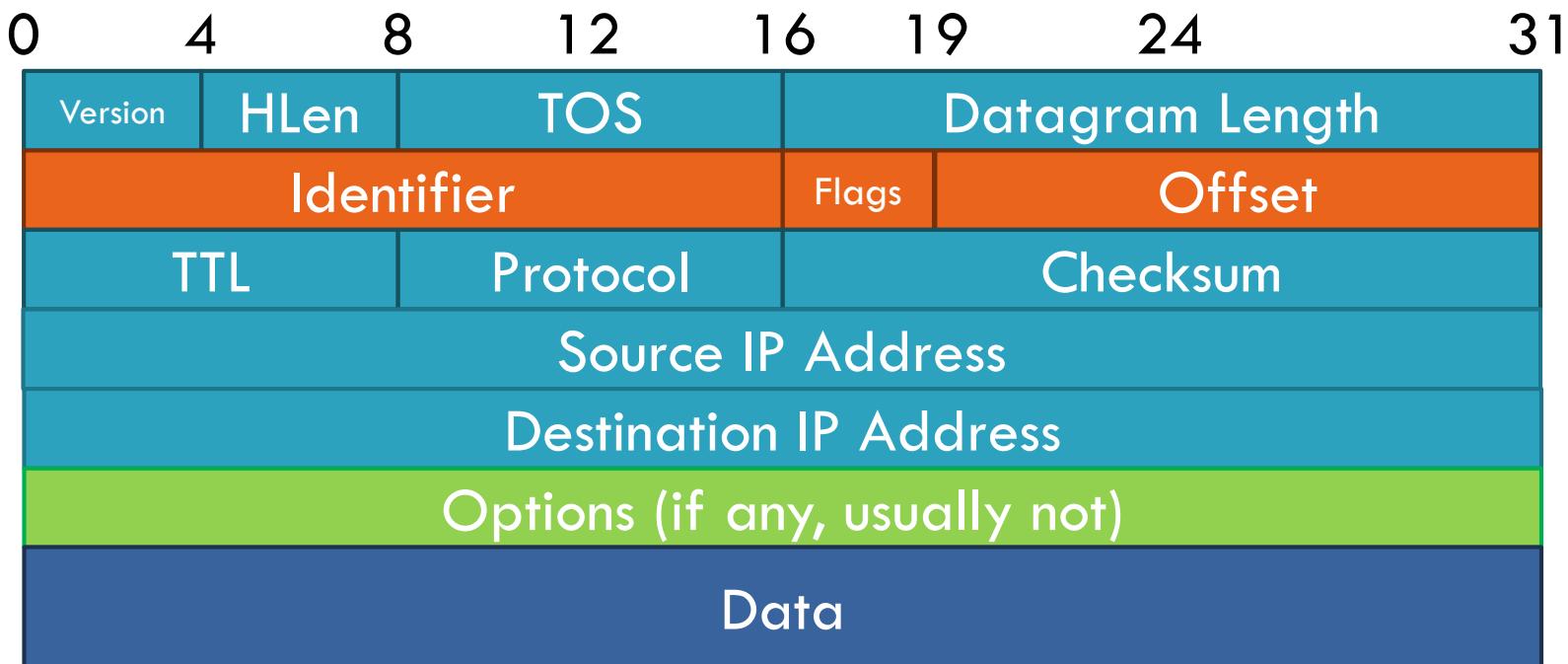


- Probléma: minden hálózatnak megvan a maga MTU-ja
 - MTU: Maximum Transmission Unit – lényegében a maximális használható csomag méret egy hálózatban
 - DARPA/Internet alapelv: hálózatok heterogének lehetnek
 - A minimális MTU nem ismert egy adott útvonalhoz
- IP esetén: fragmentáció
 - Vágjuk szét az IP csomagot, amikor az MTU csökken
 - Állítsuk helyre a darabokból a csomagot a fogadó állomásnál

IP fejléc: 2. szó

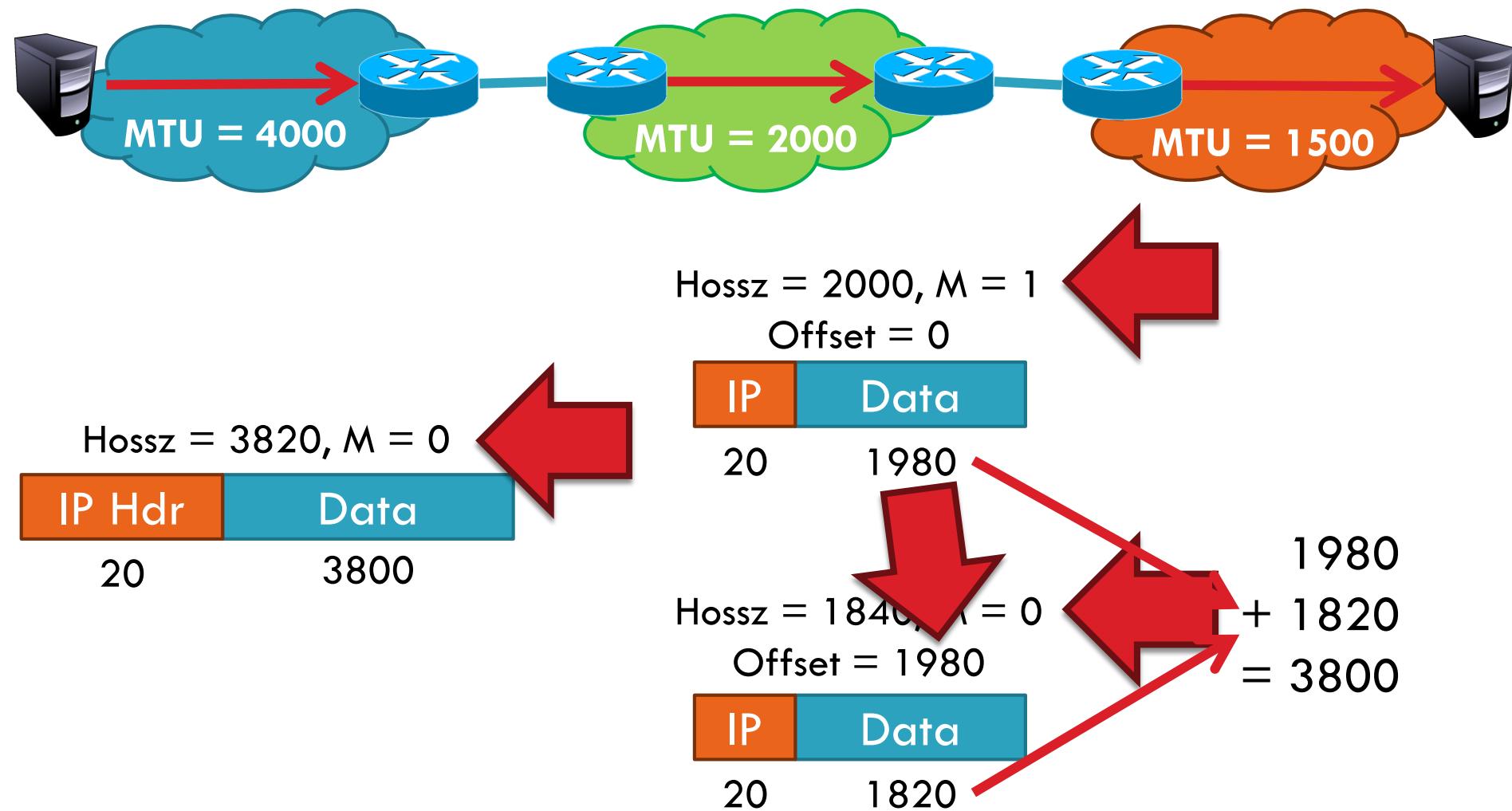
10

- Identifier (azonosító):
 - egyedi azonosító minden IP datagramhoz (csomaghoz)
- Flags (jelölő bitek):
 - M flag, értéke 0, ha ez az utolsó darab/fragment, különben 1
- Offset (eltolás):
 - A darab/fragment első bájtjának pozíciója



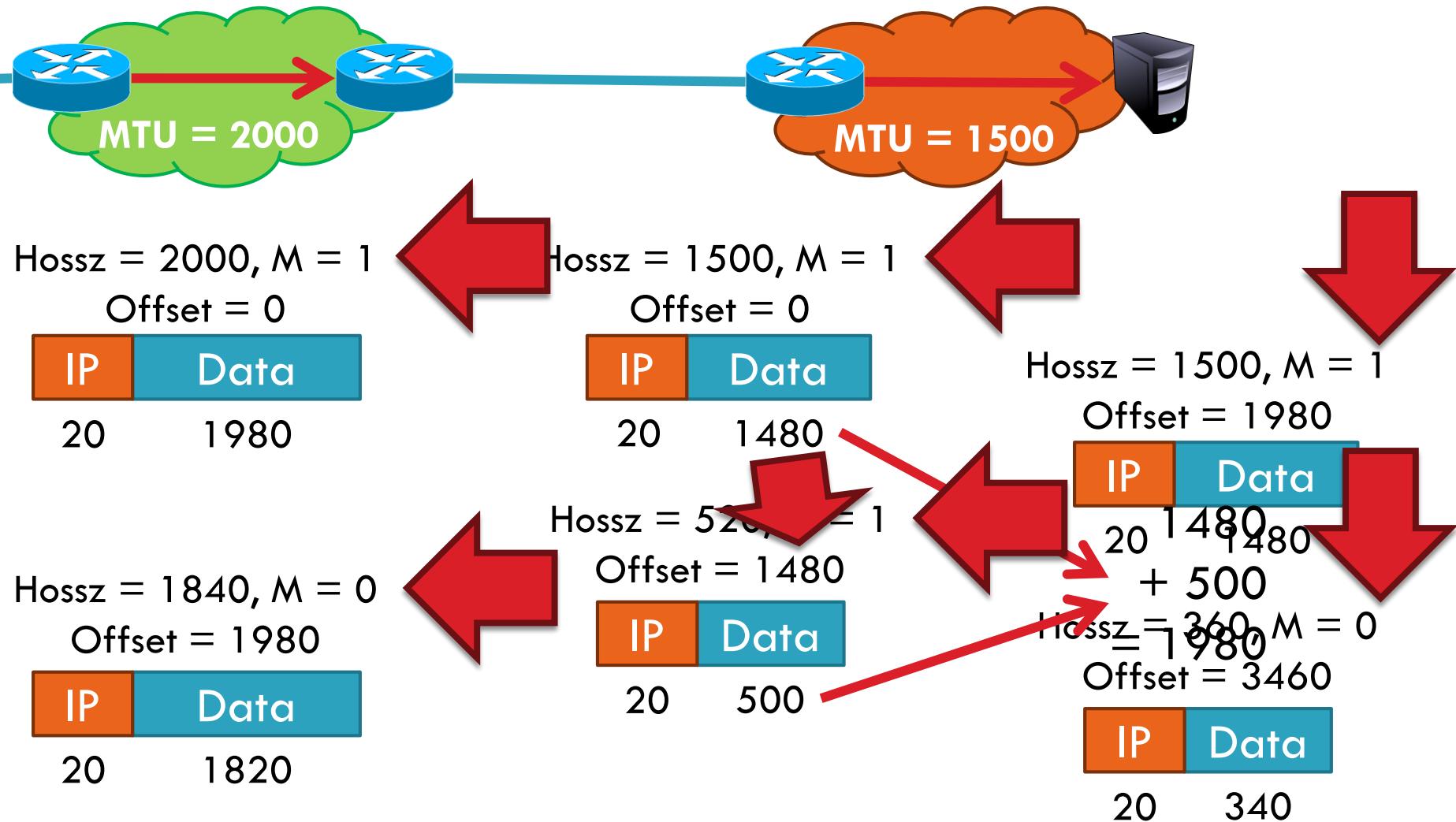
Példa

11



Példa

12



IP csomag helyreállítása

13

Hossz = 1500, M = 1, Offset = 0

IP	Data
20	1480

Hossz = 520, M = 1, Offset = 1480

IP	Data
20	500

Hossz = 1500, M = 1, Offset = 1980

IP	Data
20	1480

Hossz = 360, M = 0, Offset = 3460

IP	Data
20	340

- A végponton történik
- M = 0, akkor ebből a darabból tudjuk a teljes adatmennyiséget
 - Hossz – IPHDR_hossz + Offset
 - 360 – 20 + 3460 = 3800
- Kihívások:
 - Nem sorrendben beérkező darabok
 - Duplikátumok
 - Hiányzó darabok
- Memória kezelés szempontjából egy rémálom...

Fragmentáció

14

- Az Internet esetén
 - Elosztott és heterogén
 - minden hálózat maga választ MTU-t
 - Kapcsolat nélküli datagram/csomag alapú protokoll
 - minden darab tartalmazza a továbbításhoz szükséges összes információt
 - A darabok függetlenül kerülnek leszállításra, akár különböző útvonalon keresztül
 - Legjobb szándék elve szerint (best effort)
 - A router-ek és a fogadó is eldobhat darabokat
 - Nem követelmény a küldő értesítése a „hibáról”
 - A legtöbb feladat a végpontra hárul
 - Csomag helyreállítása a darabokból

Fregmantáció a valóságban

15

- A fragmentáció költséges
 - ▣ Memória és CPU költés a csomag visszaállításához
 - ▣ Ha lehetséges, el kell kerülni
- MTU felderítő protokoll
 - ▣ Csomagküldés a “don’t fragment” flag bittel
 - ▣ Folyamatosan csökkentjük a csomag méretét, amíg egy meg nem érkezik
 - ▣ Lehetséges “can’t fragment” hiba egy routertől, ami közvetlenül tartalmazza az adott hálózatban használt MTU-t
- Darabok kezelését végző router
 - ▣ Gyors, specializált hardver megoldás
 - ▣ Dedikált erőforrás a darabok kezeléséhez

IPv6

Fogyó IPv4 címek

17

- Probléma: az IPv4 címtartomány túl kicsi
 - $2^{32} = 4,294,967,296$ lehetséges cím
 - Ez kevesebb mint egy emberenként
- A világ egy részén már nincs kiosztható IP blokk
 - IANA az utolsó /8 blokkot 2011-ben osztotta ki

Régió	Regional Internet Registry (RIR)	Utolsó IP blokk kiosztása
Asia/Pacific	APNIC	April 19, 2011
Europe/Middle East	RIPE	September 14, 2012
North America	ARIN	13 Jan 2015 (Projected)
South America	LACNIC	13 Jan 2015 (Projected)
Africa	AFRINIC	17 Jan 2022(Projected)

IPv6

18

- IPv6, 1998(!)-ban mutatták be
 - 128 bites címek
 - $4.8 * 10^{28}$ cím/ember
- Cím formátum
 - 16 bites értékek 8 csoportba sorolva (‘:’-tal elválasztva)
 - minden csoport elején szereplő nulla sorozatok elhagyhatók
 - Csupa nulla csoportok elhagyhatók , ekkor ‘::’

2001:0db8:0000:0000:0000:ff00:0042:8329

2001:db8:0:0:0:ff00:42:8329

2001:db8::ff00:42:8329

IPv6

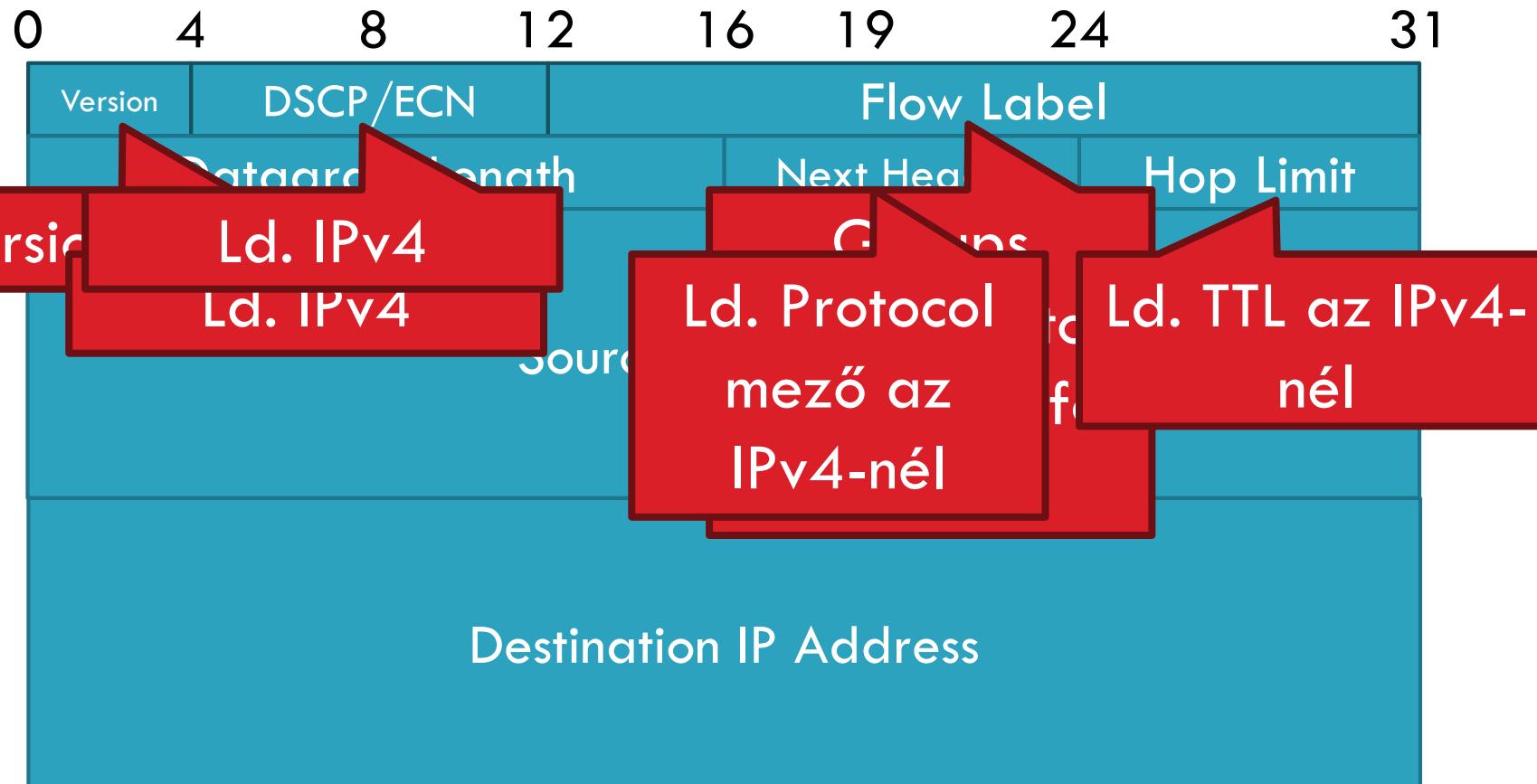
19

- Ki tudja a localhost IPv4 címét?
 - 127.0.0.1
- Mi ez az IPv6 esetén?
 - ::1

IPv6 Fejléc

20

- Az IPv4-nél látott kétszerese (320 bit vs. 160 bit)



Különbségek az IPv4-hez képest

21

- Számos mező hiányzik az IPv6 fejlécből
 - Fejléc hossza – beépült a Next Header mezőbe
 - Checksum – nem igazán használták már korábban se...
 - Identifier, Flags, Offset
 - IPv6 routerek nem támogatják a fragmentációt
 - Az állomások MTU felderítést alkalmaznak
- Az Internet felhasználás súlypontjainak megváltozása
 - Napjaink hálózatai sokkal homogénebbek, mint azt kezdetben gondolták
 - Azonban a routing költsége és bonyolultsága domináns

Teljesítmény növekmény

22

- Nincsenek ellenőrizendő kontrollösszegek (checksum)
- Nem szükséges a fragmentáció kezelése a routerekben
- Egyszerű routing tábla szerkezet
 - A cím térfogata nagy
 - Nincs szükség CIDR-re (de aggregáció szükséges)
 - A szabványos alhálózat méret 2^{64} cím
- Egyszerű auto-konfiguráció
 - Neighbor Discovery Protocol

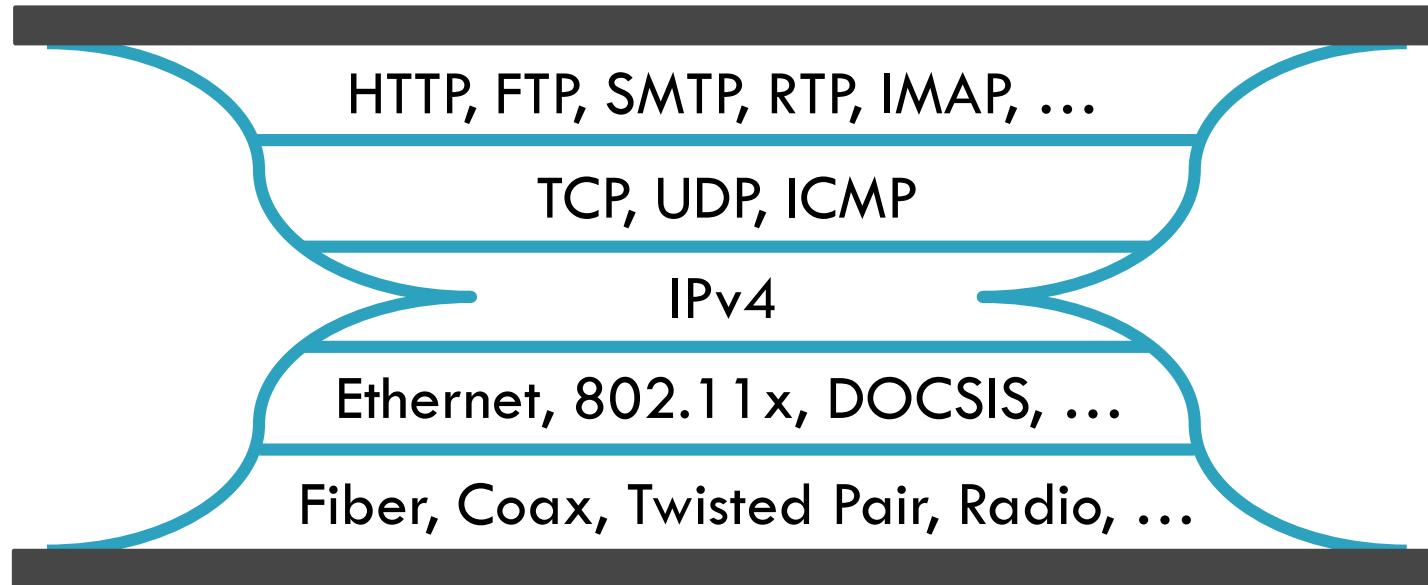
További IPv6 lehetőségek

23

- Forrás Routing
 - Az állomás meghatározhatja azt az útvonalat, amelyen a csomagjait továbbítani szeretné
- Mobil IP
 - Az állomások magukkal vihetik az IP címüket más hálózatokba
 - Forrás routing használata a csomagok irányításához
- Privacy kiterjesztések
 - Véletlenszerűen generált állomás azonosítók
 - Megnehezíti egy IP egy adott állomáshoz való kapcsolását
- Jumbograms
 - 4Gb-es datagramok küldése

Bevezetési nehézségek

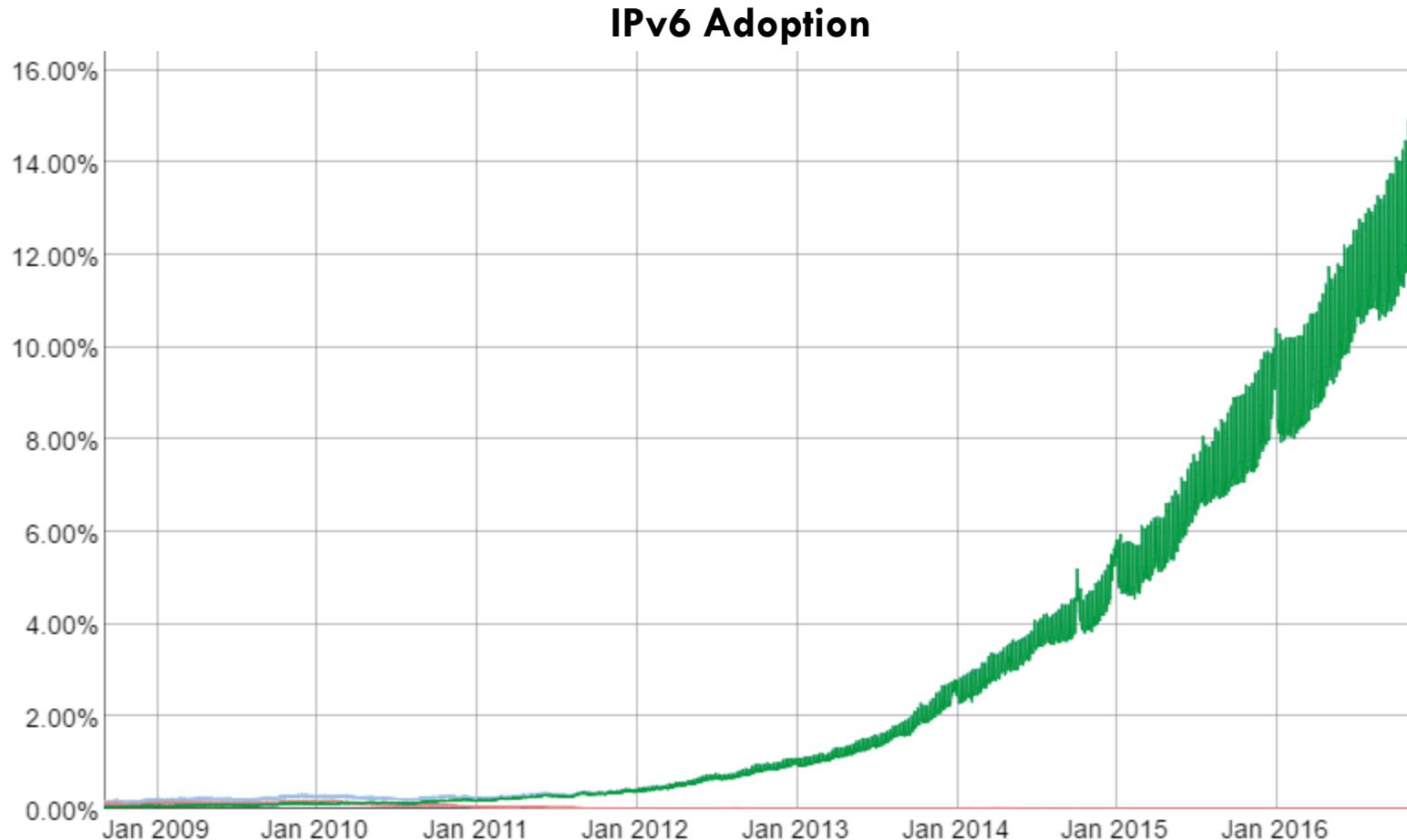
24



- IPv6 bevezetése a teljes Internet frissítését jelentné
 - minden router, minden hoszt
 - ICMPv6, DHCPv6, DNSv6
- 2013: 0.94%-a a Google forgalmának volt IPv6 feletti
- 2015: ez 2.5%

<https://www.google.com/intl/en/ipv6/statistics.html>

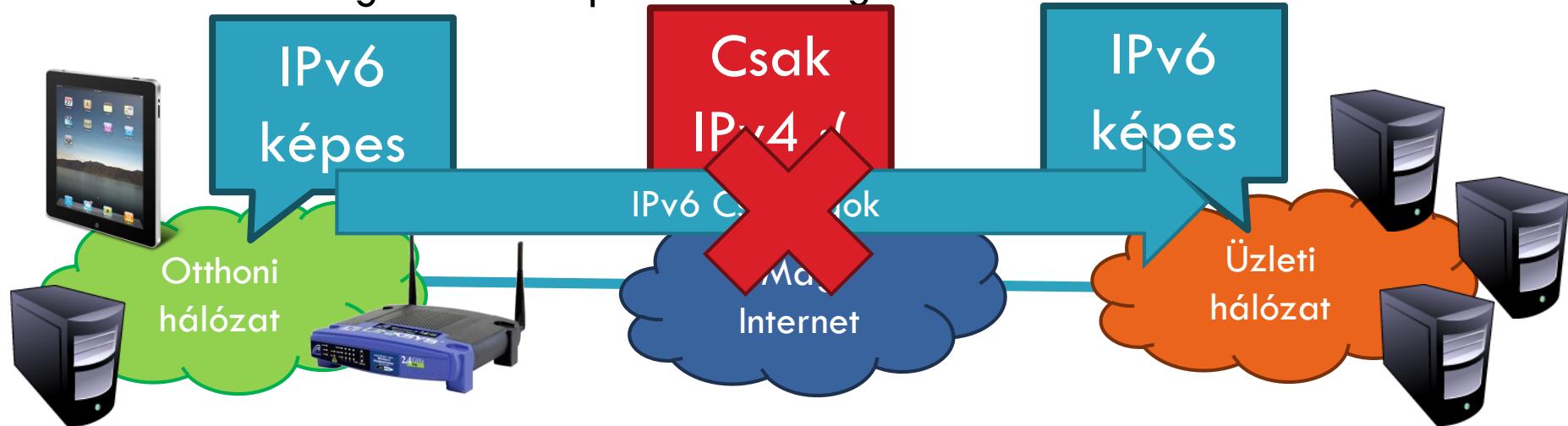
25



Átmenet IPv6-ra

26

- Hogyan történhet az átmenet IPv4-ről IPv6-ra?
 - Napjainkban a legtöbb végpont a hálózat széleken támogatja az IPv6-ot
 - Windows/OSX/iOS/Android minden tartalmaz IPv6 támogatást
 - Az itteni vezetéknélküli access point-ok is valószínűleg IPv6 képesek
 - Az Internet magja a probléma
 - IPv4 mag nem routolja az IPv6 forgalmat



Átmeneti megoldások

27

- Azaz hogyan routoljunk IPv6 forgalmaz IPv4 hálózat felett?
- Megoldás
 - Használunk **tunneleket** az IPv6 csomagok becsomagolására és IPv4 hálózaton való továbbítására
 - Számos különböző implementáció
 - 6to4
 - IPv6 Rapid Deployment (6rd)
 - Teredo
 - ...

Routing 2. felvonás

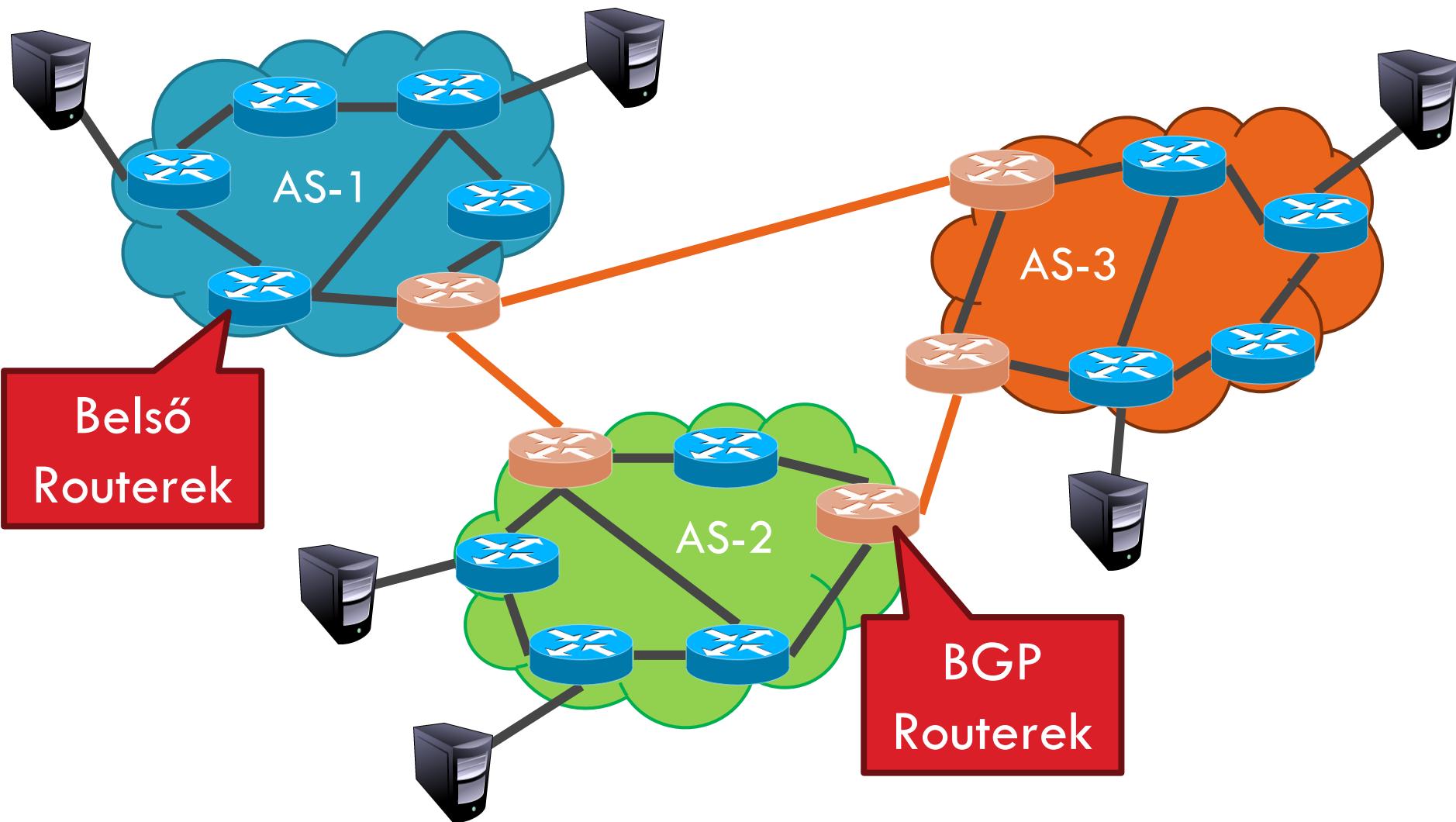
Újra: Internet forgalom irányítás

29

- Az Internet egy két szintű hierarchiába van szervezve
- Első szint – autonóm rendszerek (AS-ek)
 - AS – egy adminisztratív tartomány alatti hálózat
 - Pl.: ELTE, Comcast, AT&T, Verizon, Sprint, ...
- AS-en belül ún. **intra-domain routing** protokollokat használunk
 - Distance Vector, pl.: Routing Information Protocol (RIP)
 - Link State, pl.: Open Shortest Path First (OSPF)
- AS-ek között ún. **inter-domain routing** protokollokat
 - Border Gateway Routing (BGP)
 - Napjainkban: BGP-4

AS példa

30



Miért van szükség AS-ekre?

31

- A routing algoritmusok **nem elég hatékonyak** ahhoz, hogy a teljes Internet topológián működjene
- Különböző szervezetek **más-más politika** mentén akarnak forgalom irányítást (policy)
- Lehetőség, hogy a szervezetek **elrejtsék a belső hálózatuk szerkezetét**
- Lehetőség, hogy a szervezetek **eldöntsék**, hogy mely más szervezeteken keresztül forgalmazzanak
 - Egyszerűbb az útvonalak számítása
 - Nagyobb rugalmasság
 - Nagyobb autonómia/függetlenség

AS számok

32

- minden AS-t egy AS szám (ASN) azonosít
 - 16 bites érték (a legújabb protokollok már 32 bites azonosítókat is támogatnak)
 - 64512 – 65535 más célra foglalt
- Jelenleg kb. 40000 AS szám létezik
 - AT&T: 5074, 6341, 7018, ...
 - Sprint: 1239, 1240, 6211, 6242, ...
 - ELTE: 2012
 - Google 15169, 36561 (formerly YT), + others
 - Facebook 32934
 - Észak-amerikai AS-ek → <ftp://ftp.arin.net/info/asn.txt>

Inter-Domain Routing

33

- A globális összeköttetéshez szükséges!!!
 - Azaz minden AS-nek **ugyanazt** a protokollt kell használnia
 - Szemben az intra-domain routing-gal
- Milyen követelmények vannak?
 - Skálázódás
 - Rugalmas útvonal választás
 - Költség
 - Forgalom irányítás egy hiba kikerülésére
- Milyen protokollt válasszunk?
 - link state vagy distance vector?
 - Válasz: A BGP egy **path vector (útvonal vektor)** protokoll

Border Gateway Protocol

34

ÁLTALÁNOS

AS-ek közötti (*exterior gateway protocol*).

Eltérő célok vannak forgalomirányítási szempontból, mint az AS-eken belüli protokollnál.

Politikai szempontok szerepet játszhatnak a forgalomirányítási döntésben.

NÉHÁNY PÉLDA FORGALOMIRÁNYÍTÁSI KORLÁTOZÁSRA

- Ne legyen átmenő forgalom bizonyos AS-eken keresztül.
- Csak akkor haladjunk át Albánián, ha nincs más út a célhoz.
- Az IBM-nél kezdődő illetve végződő forgalom ne menjen át a Microsoft-on.
- A politikai jellegű szabályokat kézzel konfigurálják a BGP-routeren.
- A BGP router szempontjából a világ AS-ekből és a közöttük átmenő vonalakból áll.

DEFINÍCIÓ

- Két AS összekötött, ha van köztük a BGP-router-eiket összekötő él.

Border Gateway Protocol

35

HÁLÓZATOK CSOPORTOSÍTÁSA AZ ÁTMENŐ FORGALOM SZEMPONTJÁBÓL

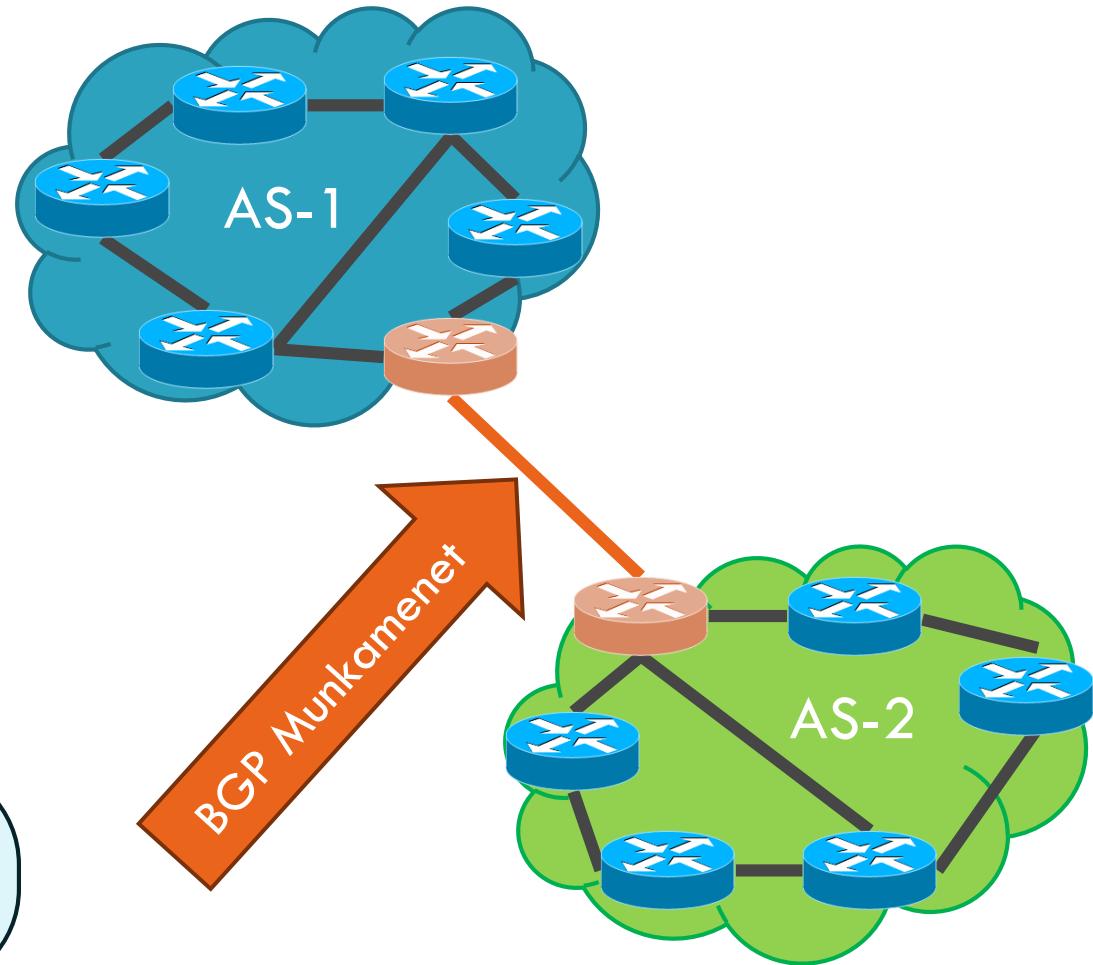
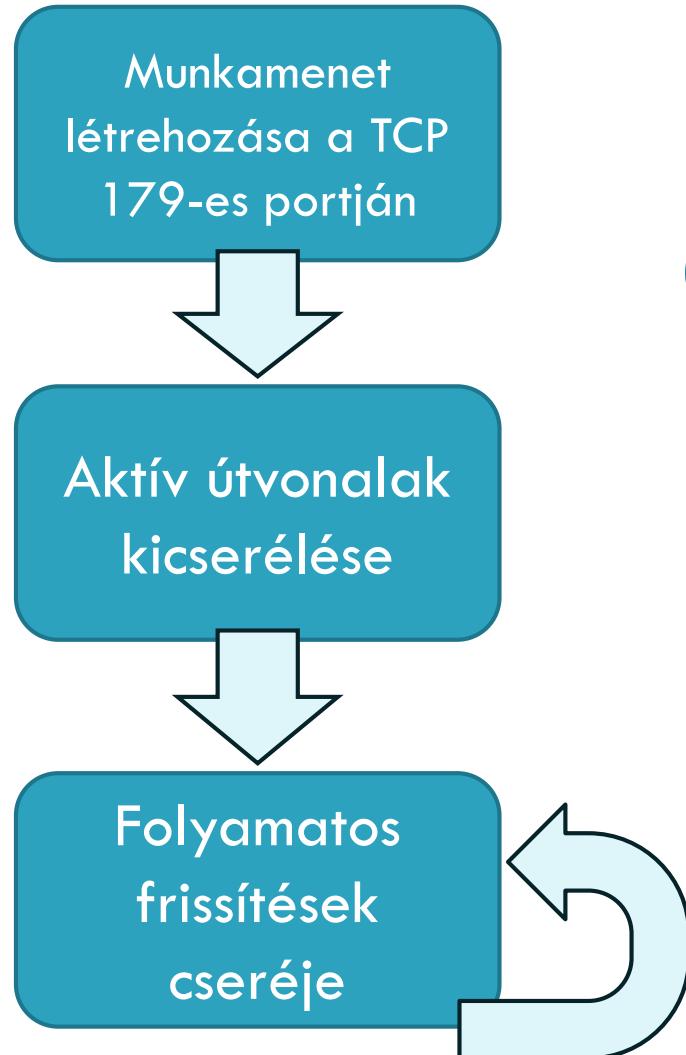
1. **Csonka hálózatok**, amelyeknek csak egyetlen összeköttetésük van a BGP gráffal.
2. **Többszörösen bekötött hálózatok**, amelyeket használhatna az átmenő forgalom, de ezek ezt megtagadják.
3. **Tranzit hálózatok**, amelyek némi megkötéssel, illetve általában fizetség ellenében, készek kezelní harmadik fél csomagjait.

JELLEMZŐK

- A BGP router-ek páronként TCP-összeköttetést létrehozva kommunikálnak egymással.
- A BGP alapvetően távolságvektor protokoll, viszont a router nyomon követi a használt útvonalat, és az útvonalat mondja meg a szomszédjainak.

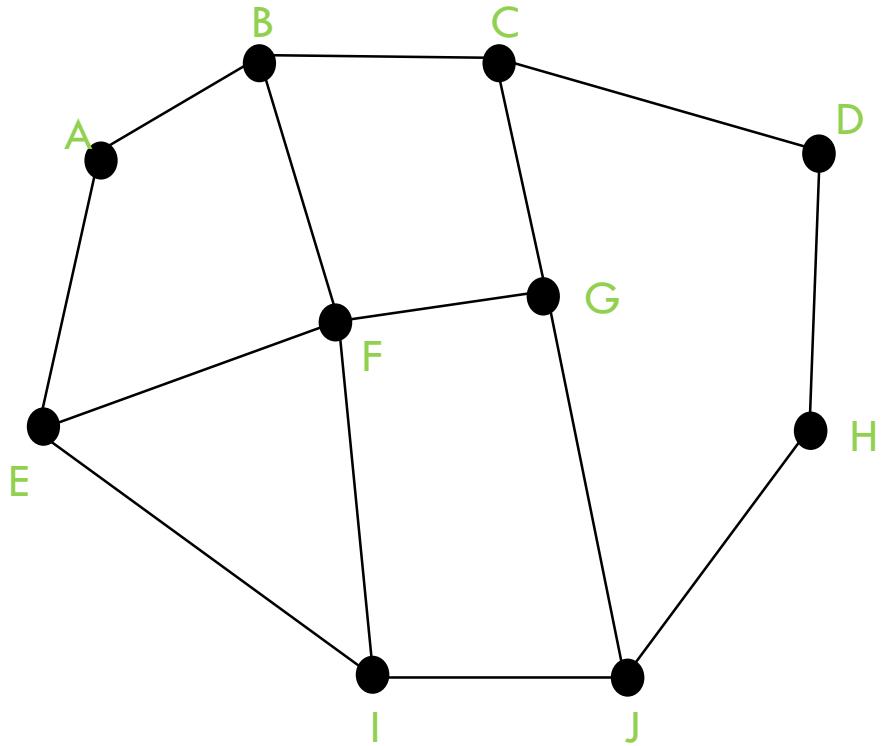
BGP egyszerűsített működése

36



Border Gateway Protocol

37



A *F* által a szomszédjaitól kapott *D*-re vonatkozó információ az alábbi:

B-től: „Én a *BCD*-t használom”

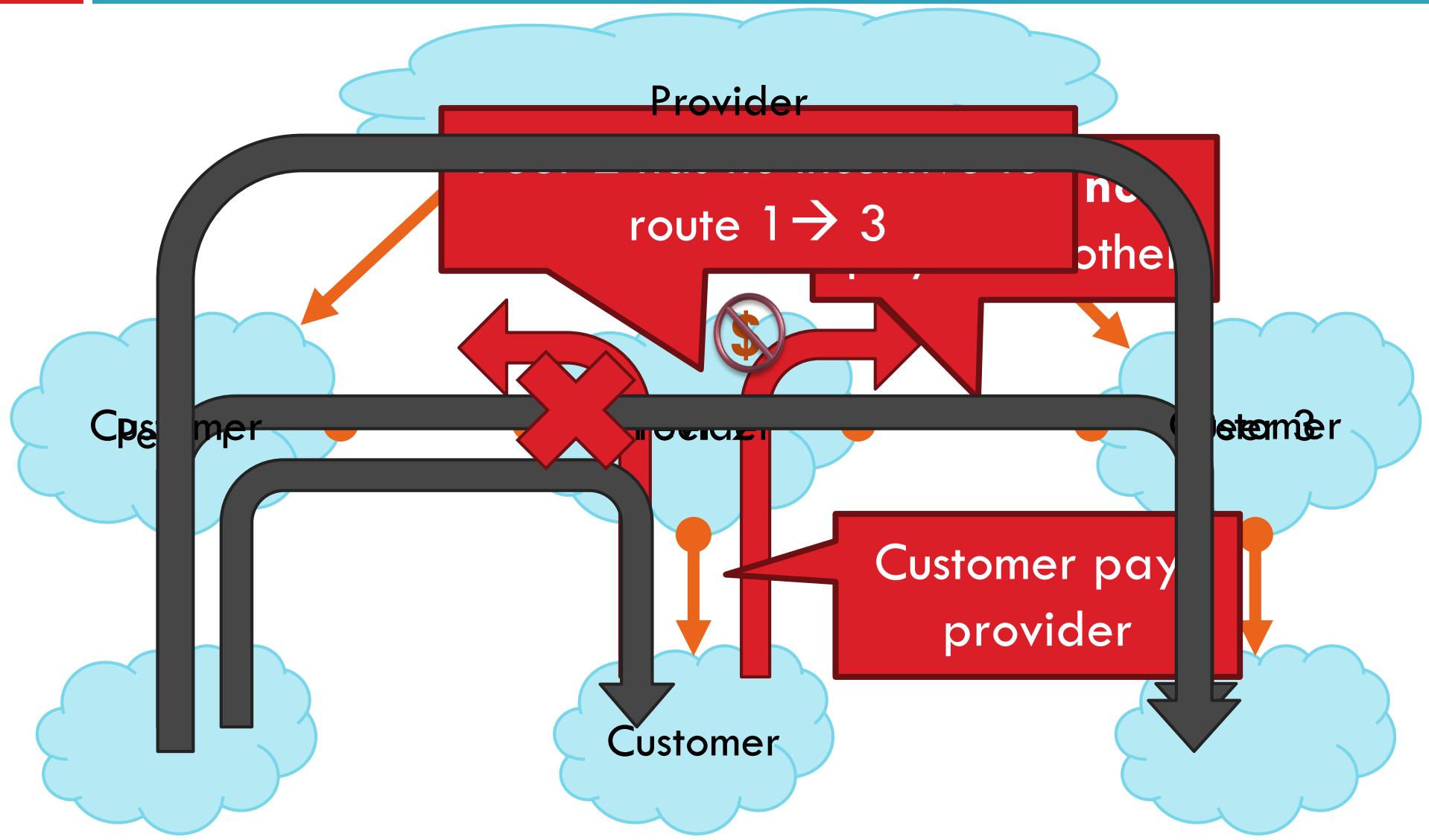
G-től: „Én a *GCD*-t használom”

I-től: „Én a *IFGCD*-t használom”

E-től: „Én a *EFGCD*-t használom”

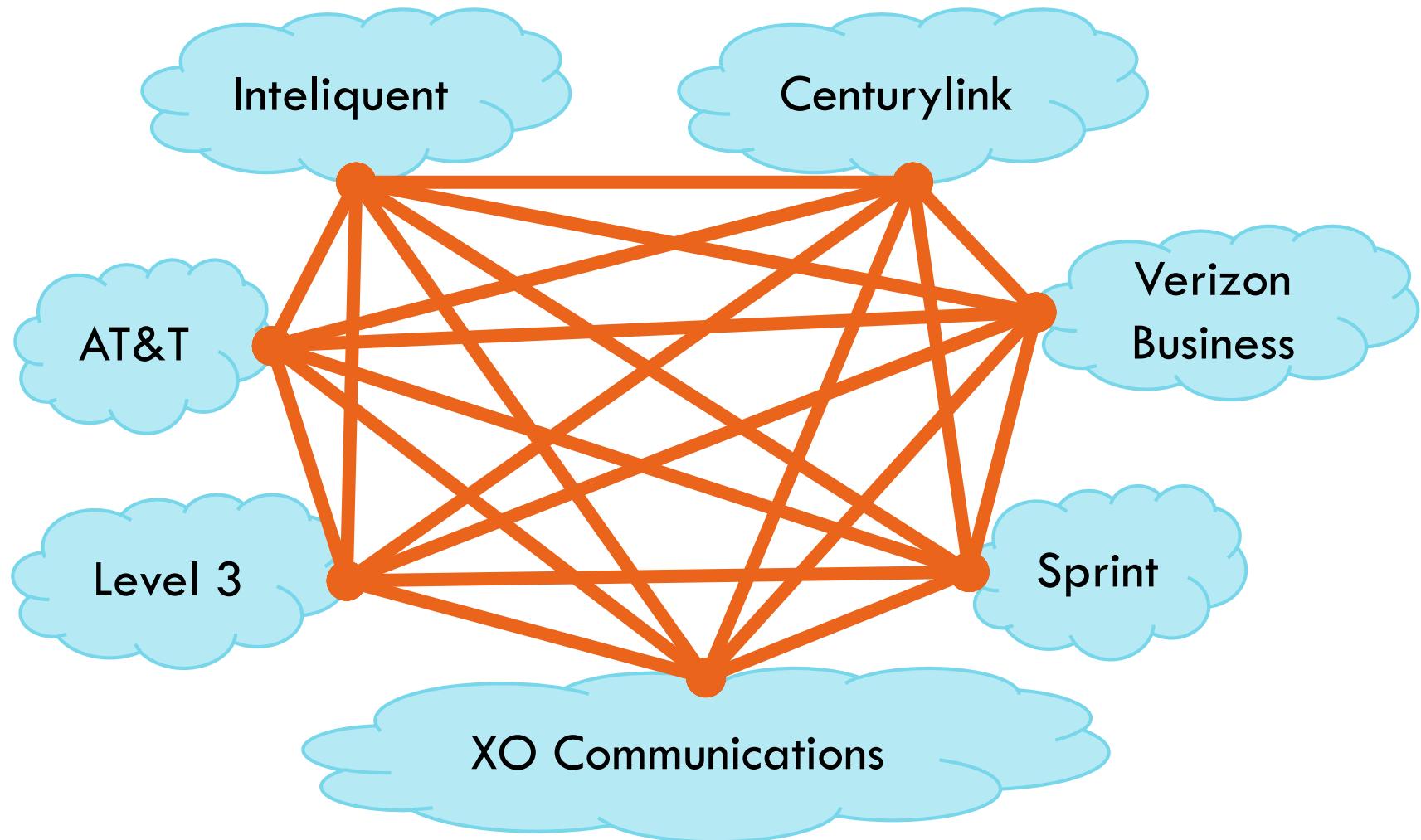
BGP kapcsolatok

38



Tier-1 ISP Peering

39



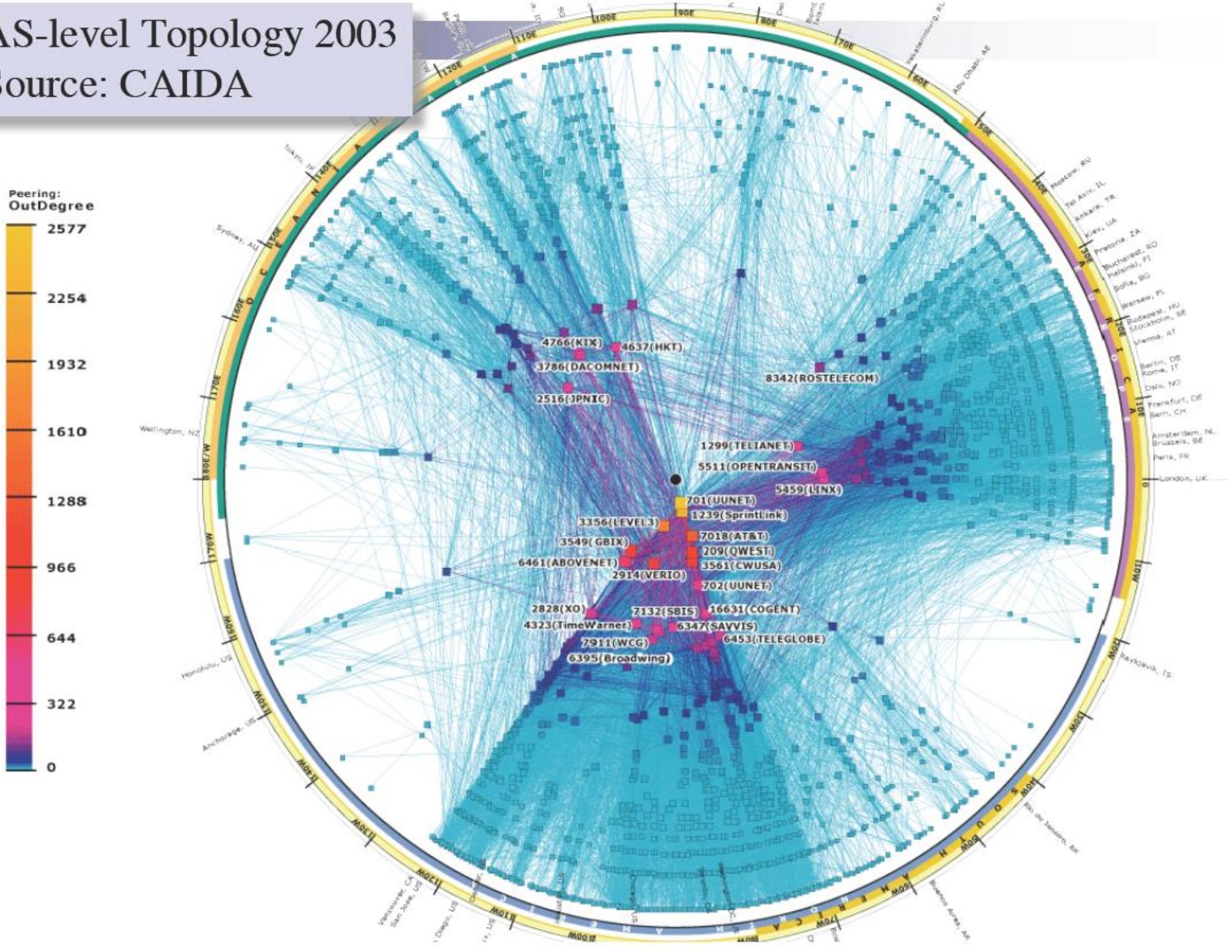
Tier-1 ISP Peering

40



AS-level Topology 2003

Source: CAIDA

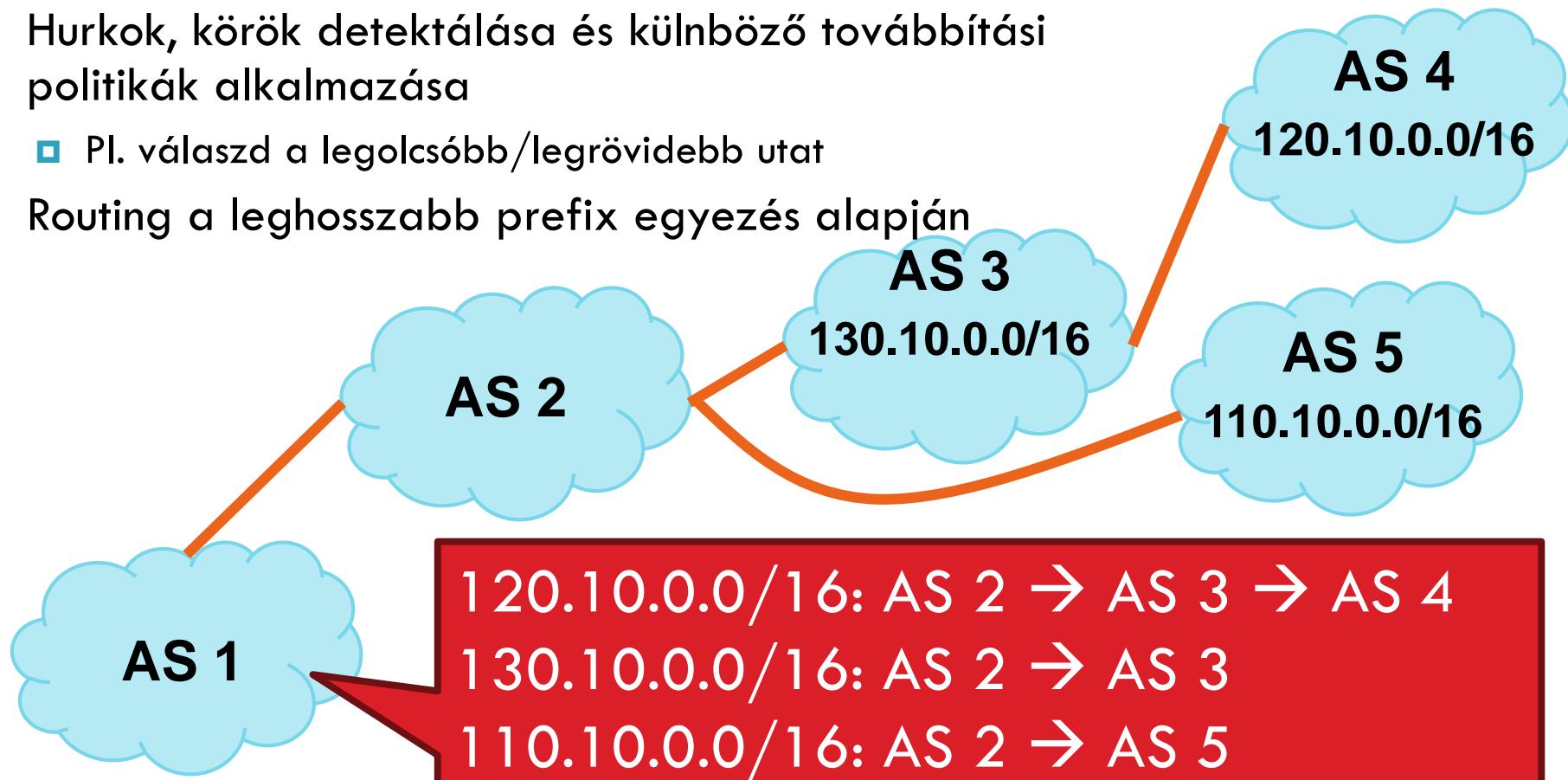


Útvonalvektor protokoll

Path Vector Protocol

42

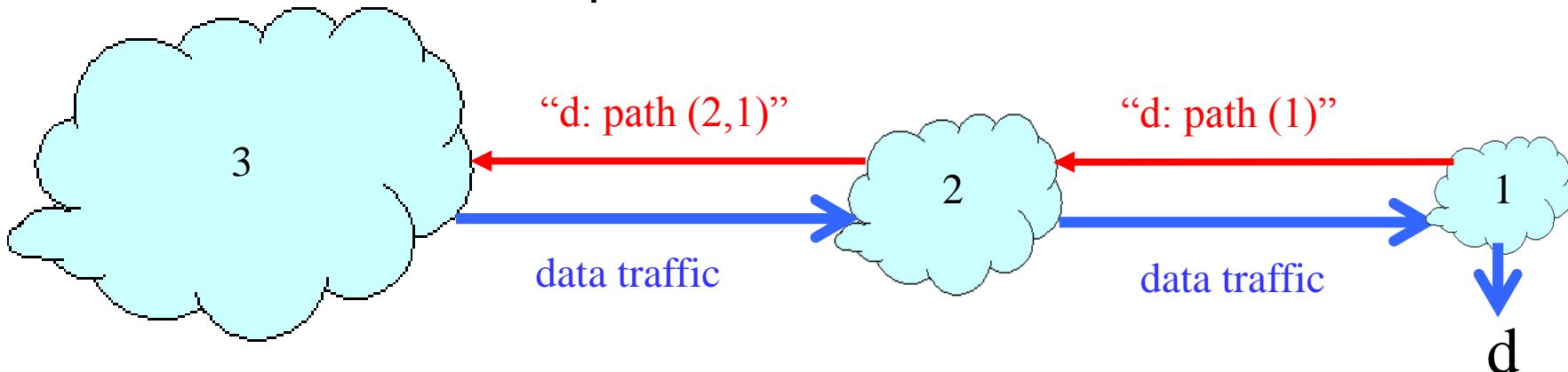
- AS-útvonal: AS-ek sorozata melyeken áthalad az útvonal
 - Hasonló a távolságvektorhoz, de további információt is tartalmaz
- Hurkok, körök detektálása és különböző továbbítási politikák alkalmazása
 - Pl. válaszd a legolcsóbb/legrövidebb utat
- Routing a leghosszabb prefix egyezés alapján



Útvonalvektor protokoll

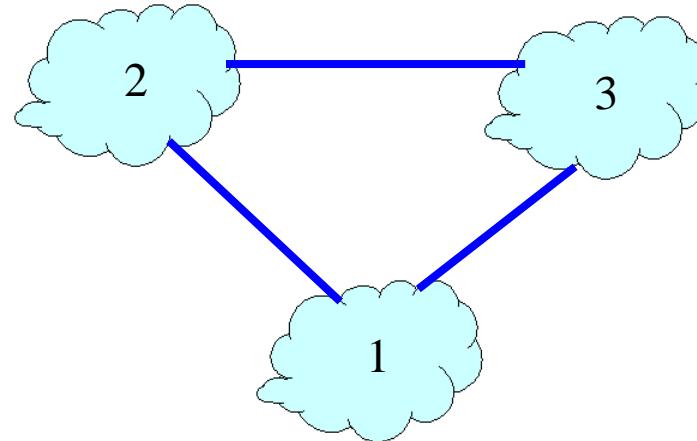
Path Vector Protocol

- A távolságvektor protokoll kiterjesztése
 - Rugalmas továbbítási politikák
 - Megoldja a végtelenig számolás problémáját
 - Útvonalvektor: Célállomás, következő ugrás (nh), AS útvonal
- Ötlet: a teljes útvonalat meghirdeti
 - Távolságvektor: távolság metrika küldése célállomásonként
 - Útvonalvektor: a teljes útvonal küldése célállomásonként



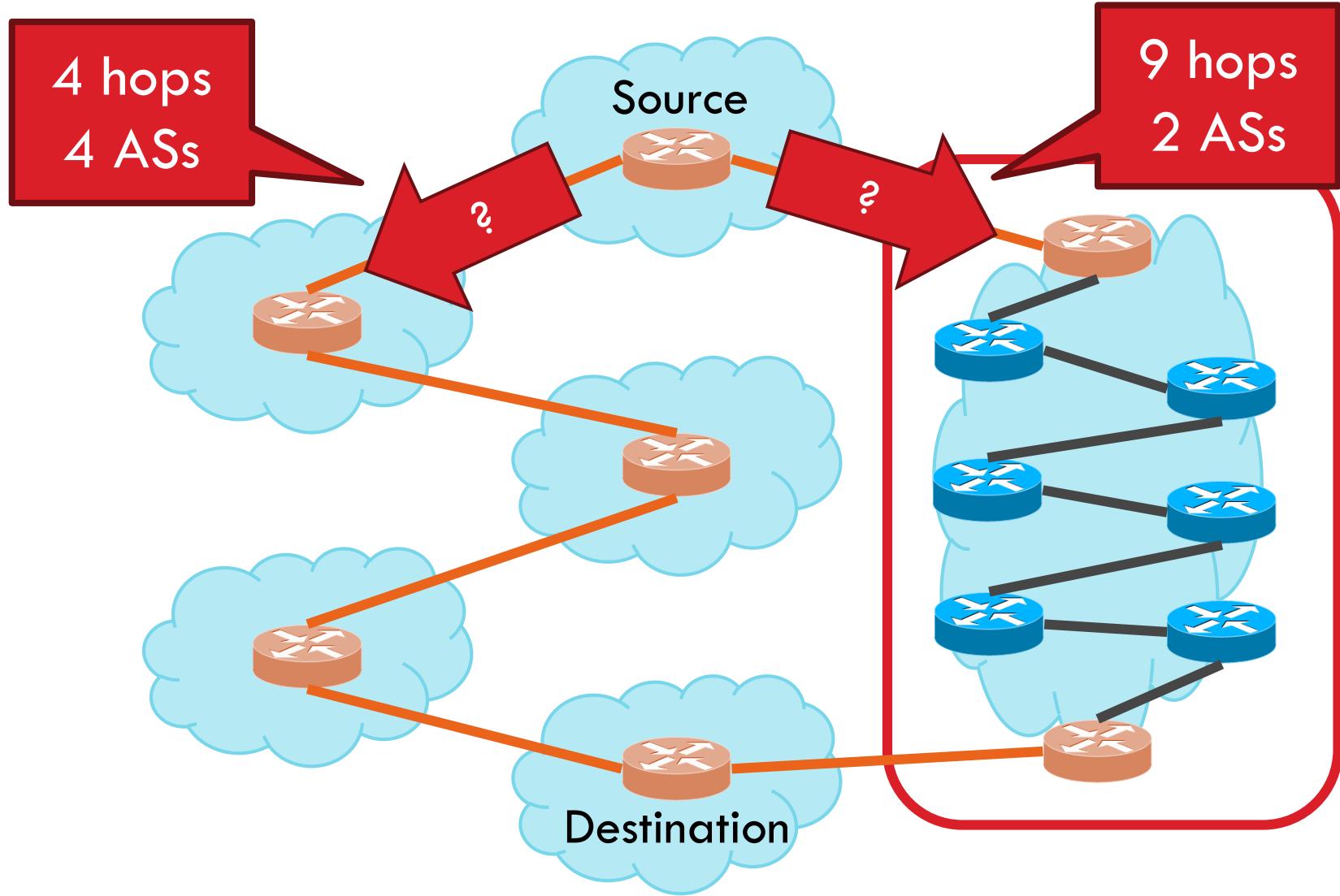
Rugalmas forgalomirányítás

- minden állomás hely/saját útválasztási politikát alkalmaz
 - Útvonal kiválasztás: Melyik útvonalat használjuk?
 - Útvonal export: Melyik útvonalat hirdessük meg?
- Példák
 - A 2. állomás által preferált útvonal: “2, 3, 1” (nem a “2, 1”)
 - Az 1. állomás nem hagyja, hogy a 3. állomás értesüljön az “1, 2” útvonalról



Shortest AS Path != Shortest Path

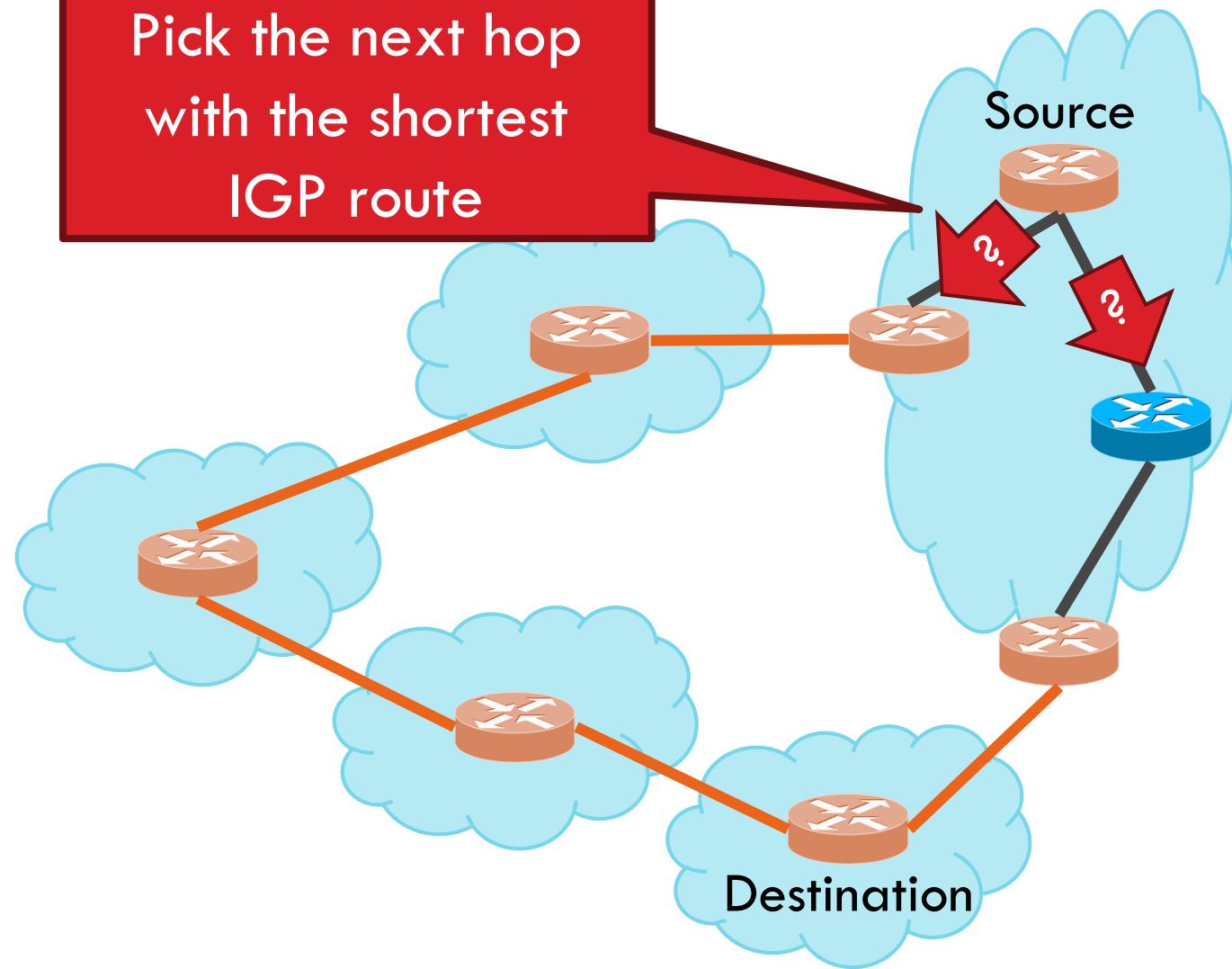
45



Hot Potato Routing

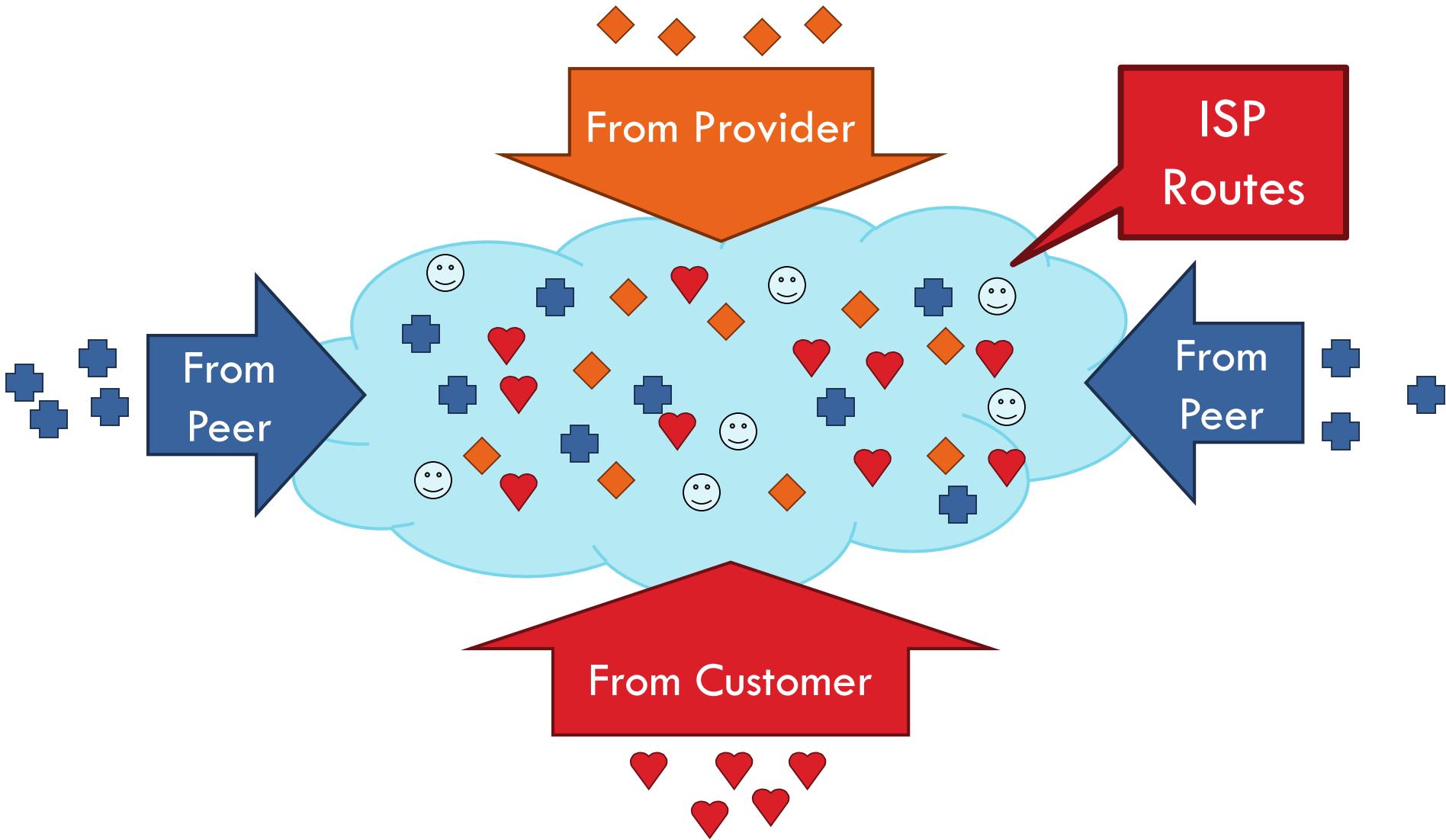
46

Pick the next hop
with the shortest
IGP route



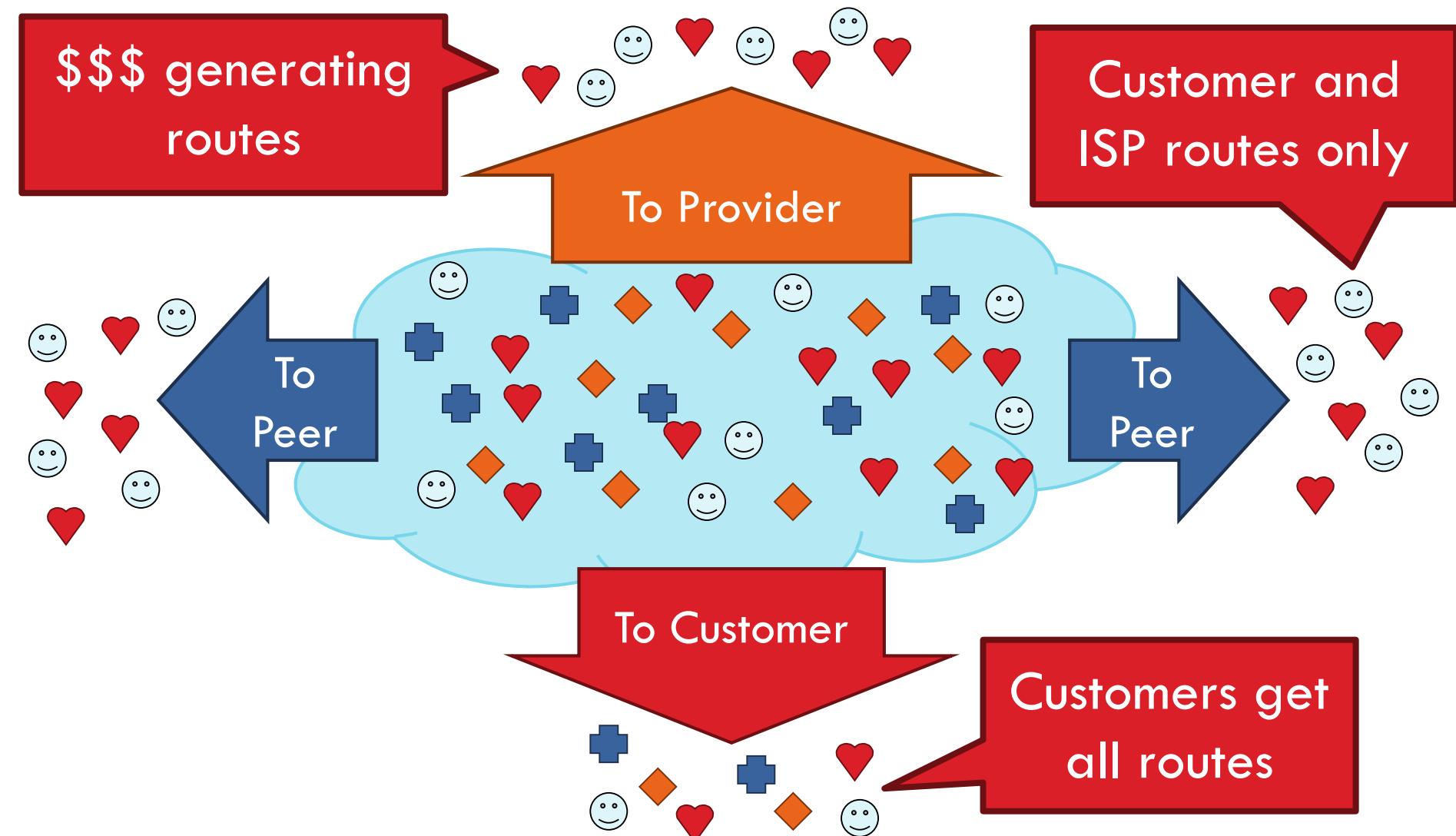
Importing Routes

47



Exporting Routes

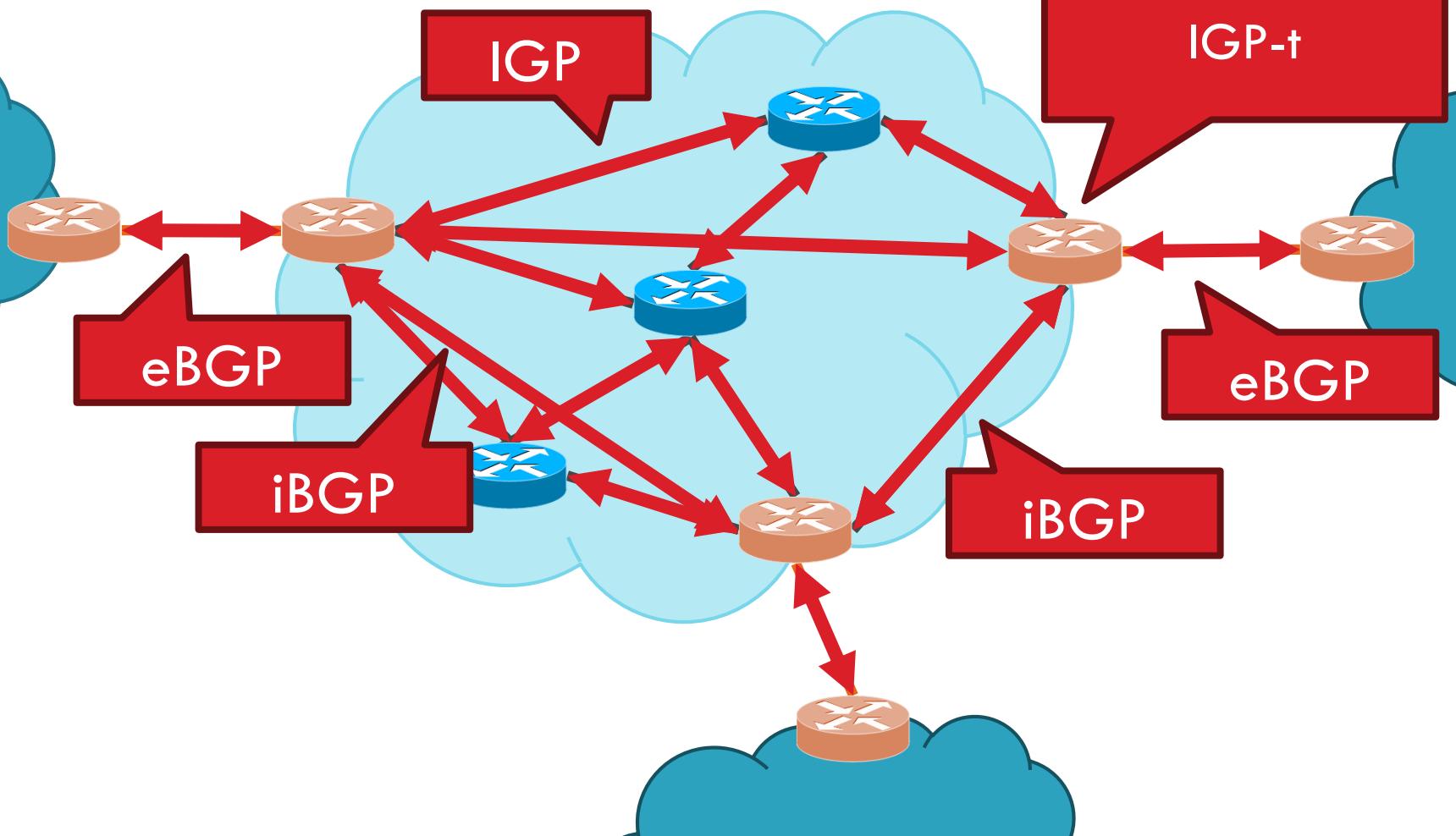
48



BGP

49

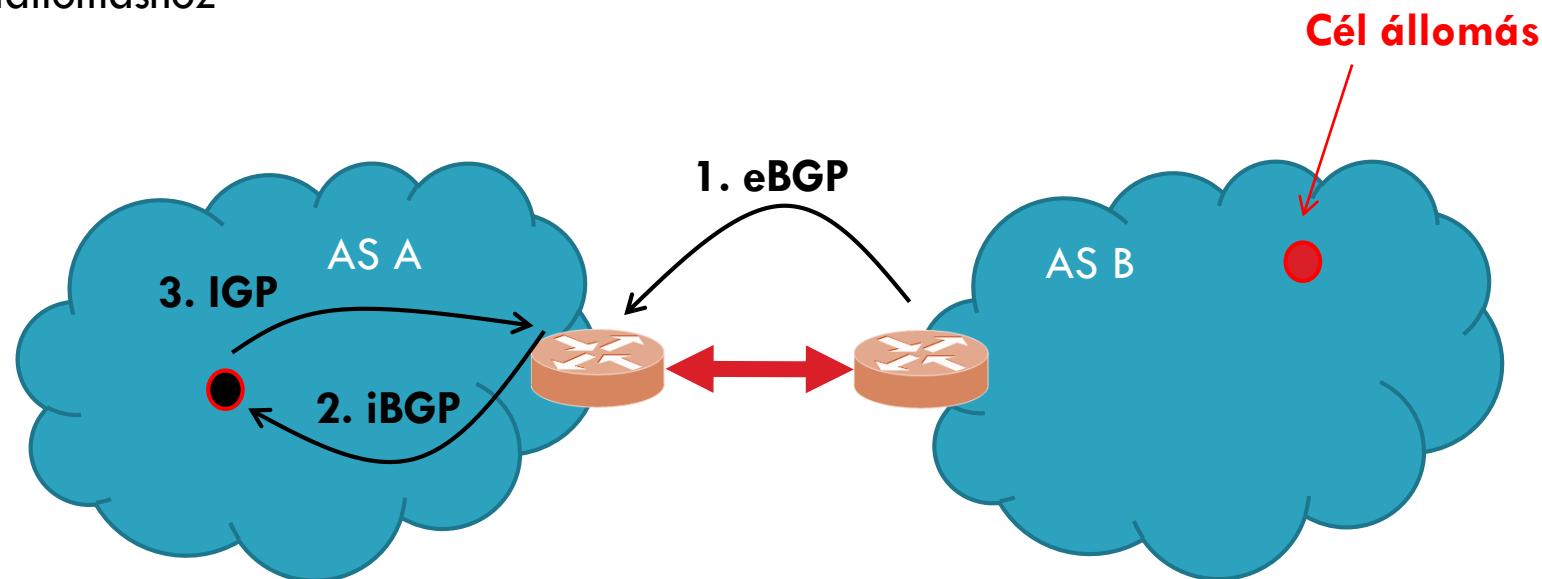
A határ-routerek
is beszélik az
IGP-t



IGB – iBGP – eBGP

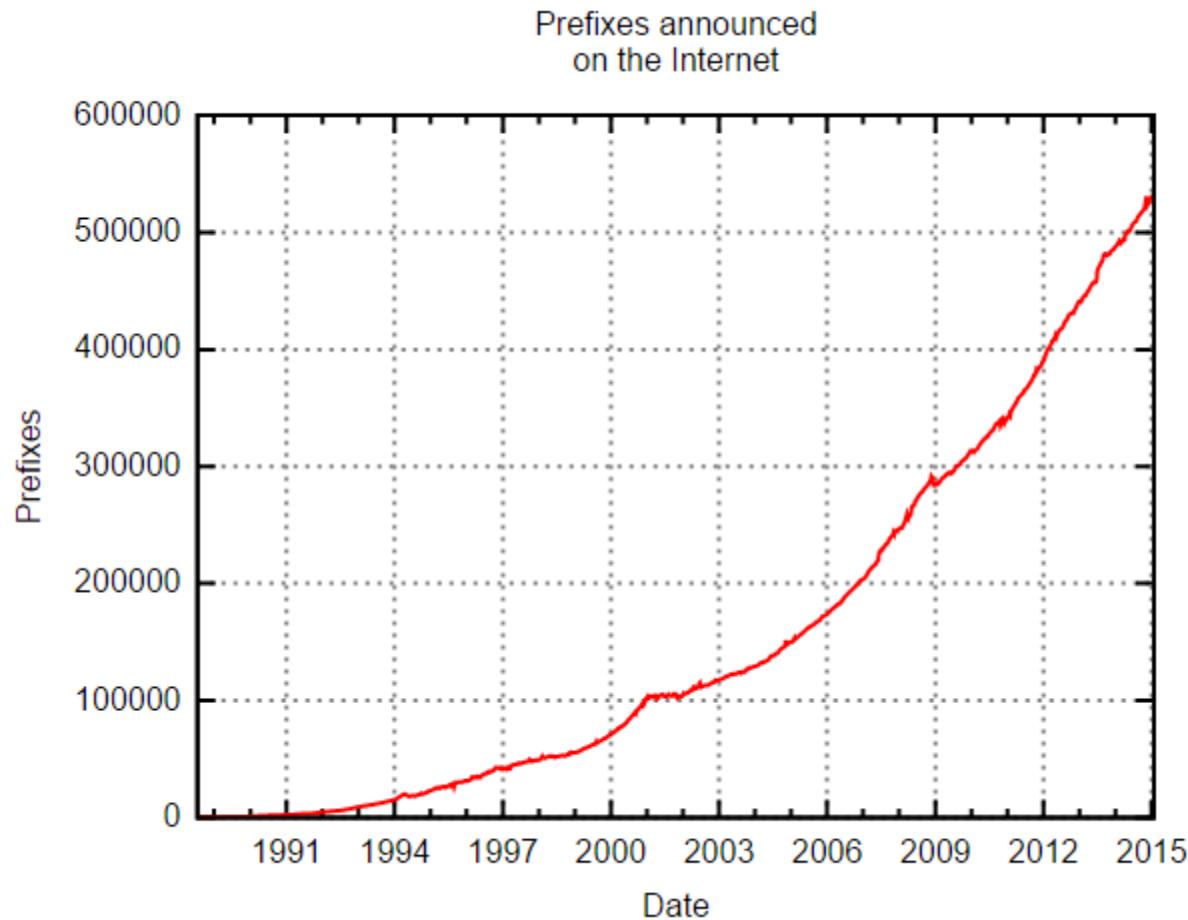
50

- eBGP: Routing információk cseréje autonóm rendszerek között
- IGP: útválasztás egy AS-en belül belső célállomáshoz
- iBGP: útválasztás egy AS-en belül egy külső célállomáshoz
- 1. eBGP – A megismeri az útvonal a célhoz, ehhez eBGP-t használunk
- 2. iBGP – A-ban levő router megtanulja a célhoz vezető utat az iBGP segítségével (a köv. ugrás a határ router)
- 3. IGP – IGP segítségével eljuttatja a csomagot az A határrouteréig



Forrás: wikipedia

51



További protokollok

Internet Control Message Protocol

53

FELADATA

- Váratlan események jelentése

HASZNÁLAT

- Többféle ICMP-üzenetet definiáltak:
 - Elérhetetlen cél;
 - Időtúllépés;
 - Paraméter probléma;
 - Forráslefojtás;
 - Visszhang kérés;
 - Visszhang válasz;
 - ...

Internet Control Message Protocol

54

- *Elérhetetlen* cél esetén a csomag kézbesítése sikertelen volt.
 - ▣ **Esemény lehetséges oka:** Egy nem darabolható csomag továbbításának útvonalán egy „kis csomagos hálózat” van.
- *Időtúllépés* esetén az IP csomag élettartam mezője elérte a 0-át.
 - ▣ **Esemény lehetséges oka:** Torlódás miatt hurok alakult ki vagy a számláló értéke túl alacsony volt.
- *Paraméter probléma* esetén a fejrészben érvénytelen mezőt észleltünk.
 - ▣ **Esemény lehetséges oka:** Egy az útvonalon szereplő router vagy a hoszt IP szoftverének hibáját jelezheti.

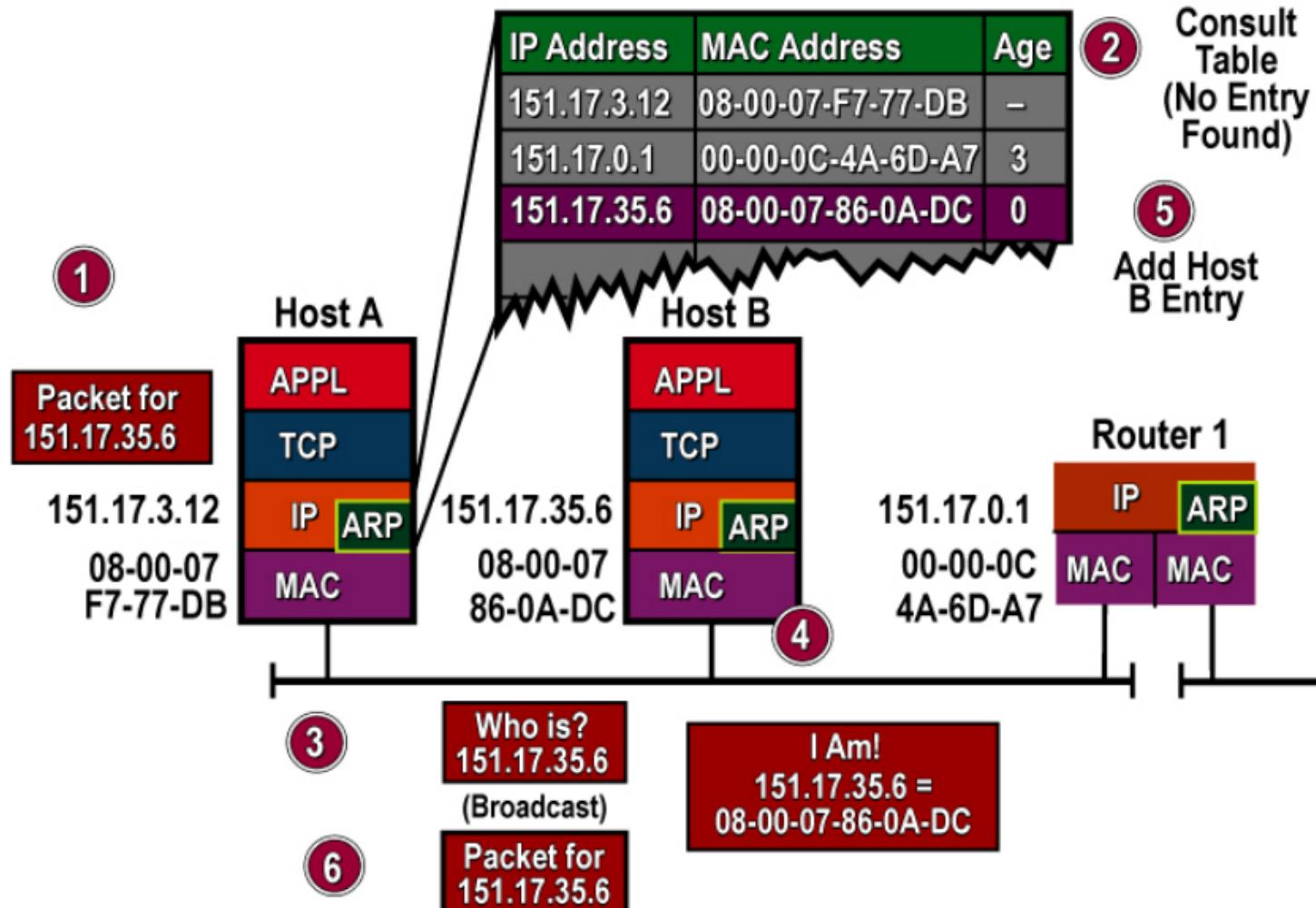
Internet Control Message Protocol

55

- Forráslefojtás esetén lefojtó csomagot küldünk.
 - ▣ **Esemény hatása:** A fogadó állomásnak a forgalmazását lassítania kellett.
- Visszhang kérés esetén egy hálózati állomás jelenlétét lehet ellenőrizni.
 - ▣ **Esemény hatása:** A fogadónak vissza kell küldeni egy visszhang választ.
- Átirányítás esetén a csomag rosszul irányítottságát jelzik.
 - **Esemény kiváltó oka:** Router észleli, hogy a csomag nem az optimális útvonal.

Address Resolution Protocol

56



Address Resolution Protocol

57

FELADATA

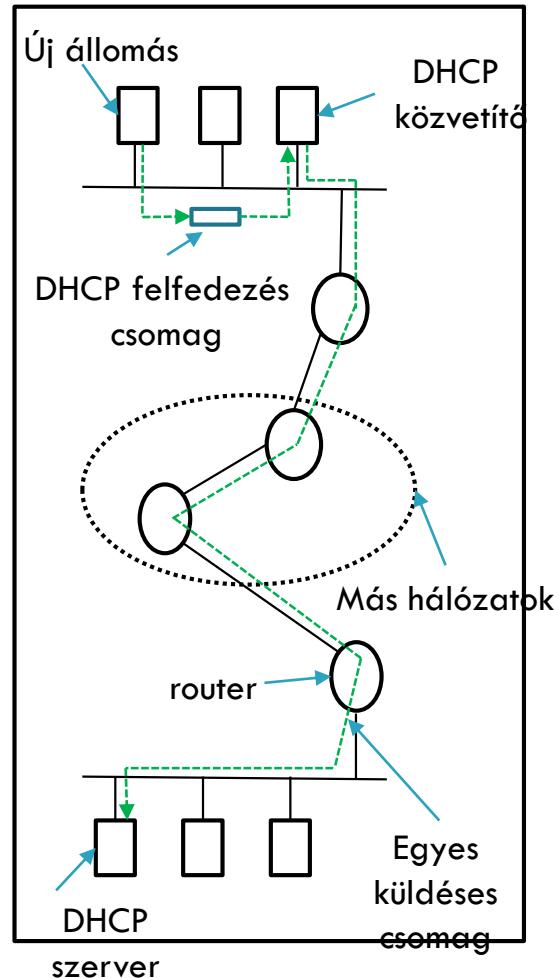
- Az IP cím megfeleltetése egy fizikai címnek.

HOZZÁRENDELÉS

- Adatszóró csomag kiküldése az *Ethernetre* „Ki-é a 192.60.34.12-es IP-cím?” kérdéssel az alhálózaton, és mindenegyes hoszt ellenőrzi, hogy övé-e a kérdéses IP-cím. Ha egyezik az IP a hoszt saját IP-jével, akkor a saját *Ethernet* címével válaszol. Erre szolgál az ARP.
- Opcionális javítási lehetőségek:
 - a fizikai cím IP hozzárendelések tárolása (*cache használata*);
 - Leképezések megváltoztathatósága (*időhatály bevezetése*);
- Mi történik távoli hálózaton lévő hoszt esetén?
 - A router is válaszoljon az ARP-re a hoszt alhálózatán. (*proxy ARP*)
 - Alapértelmezett Ethernet-cím használata az összes távoli forgalomhoz

Reverse Address Resolution Protocol

58



Reverse Address Resolution Protocol

FELADATA

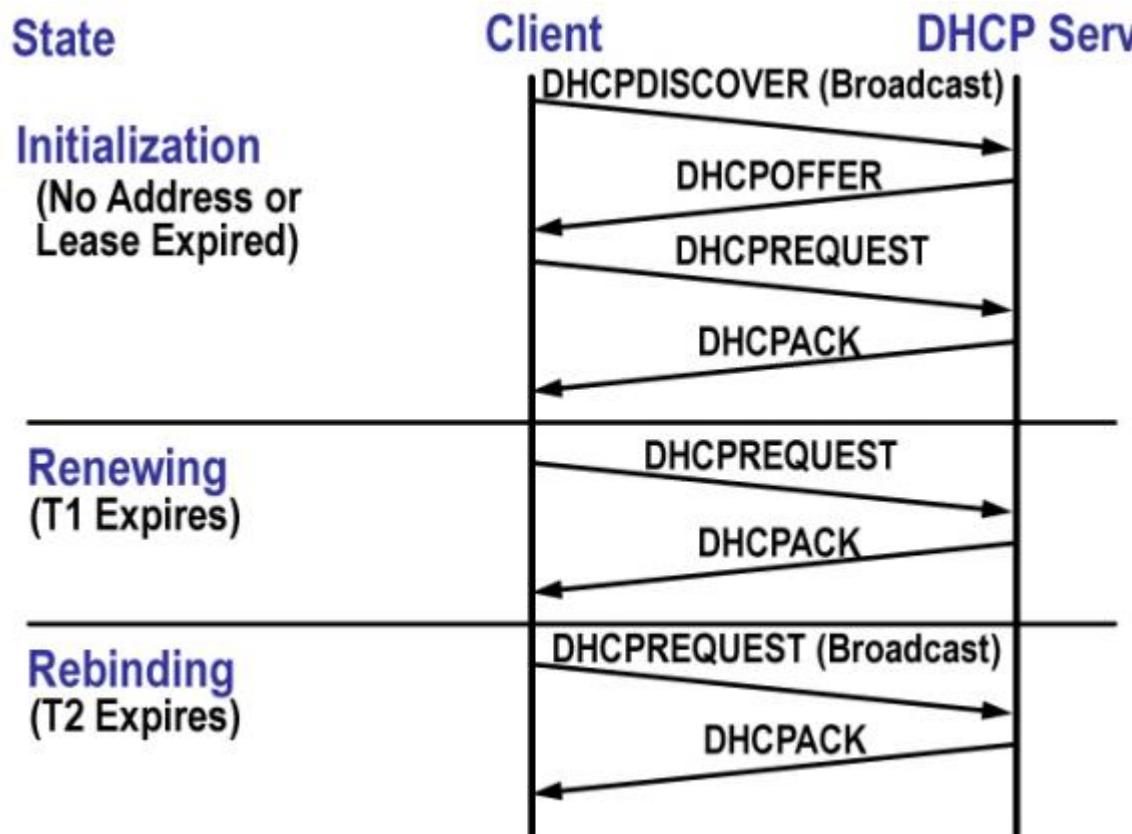
- A fizikai cím megfeleltetése egy IP címnek

HOZZÁRENDELÉS

- 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?” kérdéssel az alhálózaton. Az *RARP*-szerver pedig válaszol a megfelelő IP címmel, mikor meglátja a kérést
- Opcionális javítási lehetőségek:
 - *BOOTP* protokoll használata. UDP csomagok használata. Manuálisan kell a hozzárendelési táblázatot karbantartani. (statikus címkiosztás)
 - *DHCP* protokoll használata. Itt is külön kiszolgáló osztja ki a címeket a kérések alapján. A kiszolgáló és a kérő állomások nem kell hogy ugyanazon a *LAN*-on legyenek, ezért *LAN*-onként kell egy *DHCP relay agent*. (statikus és dinamikus címkiosztás)

DHCP: DYNAMIC HOST CONFIGURATION PROTOCOL

60



DHCP

61

- Lényegében ez már az **Alkalmazási réteg**
 - de logikailag ide tartozik
- Segítségével a hosztok automatikusan juthatnak hozzá a kommunikációjukhoz szükséges hálózati azonosítóhoz:
 - IP cím, hálózati maszk, alapértelmezett átjáró, stb.
- Eredetileg az RFC 1531 a BOOTP kiterjesztéseként definiálta. Újabb RFC-k: 1541, 2131 (aktuális)

DHCP lehetőségei

62

- IP címek osztása MAC cím alapján DHCP szerverrel
 - Szükség esetén (a DHCP szerveren előre beállított módon) egyes kliensek számára azok MAC címéhez fix IP cím rendelhető
- IP címek osztása dinamikusan
 - A DHCP szerveren beállított tartományból „érkezési sorrendben” kapják a kliensek az IP címeket
 - Elegendő annyi IP cím, ahány gép egyidejűleg működik
- Az IP címeiken kívül további szükséges hálózati paraméterek is kioszthatók
 - Hálózati maszk
 - Alapértelmezett átjáró
 - Névkiszolgáló
 - Domain név
 - Hálózati rendszerbetöltéshez szerver és fájlnév

DHCP – Címek bérlese

63

- A DHCP szerver a klienseknek az IP-címekeket bizonyos bérleti időtartamra (lease time) adja „bérbe”
 - Az időtartam hosszánál a szerver figyelembe veszi a kliens esetleges ilyen irányú kérését
 - Az időtartam hosszát a szerver beállításai korlátozzák
- A bérleti időtartam lejárta előtt a bérlet meghosszabbítható
- Az IP-cím explicit módon vissza is adható

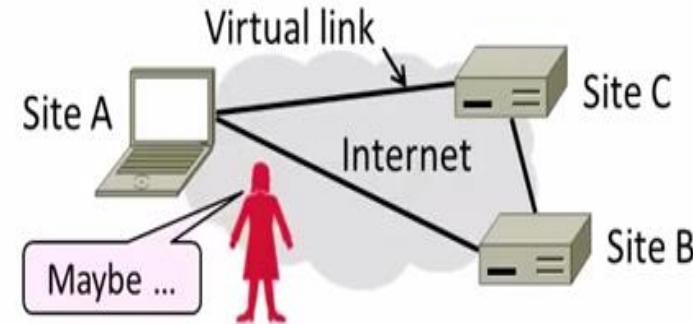
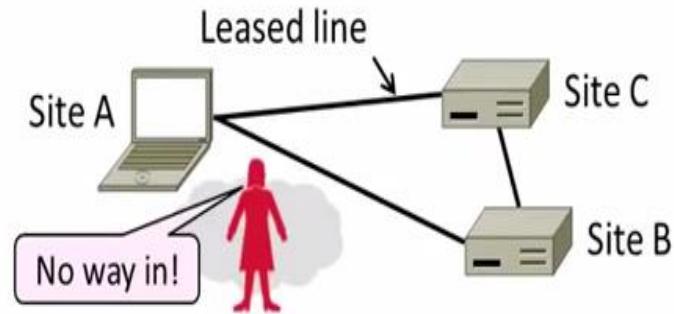
Virtuális magánhálózatok alapok

□ FŐ JELLEMZŐI

- Mint közeli hálózat fut az interneten keresztül.
- IPSEC-et használ az üzenetek titkosítására.
- Azaz informálisan megfogalmazva fizikailag távol lévő hosztok egy közös logikai egységet alkotnak.
 - Például távollévő telephelyek rendszerei.

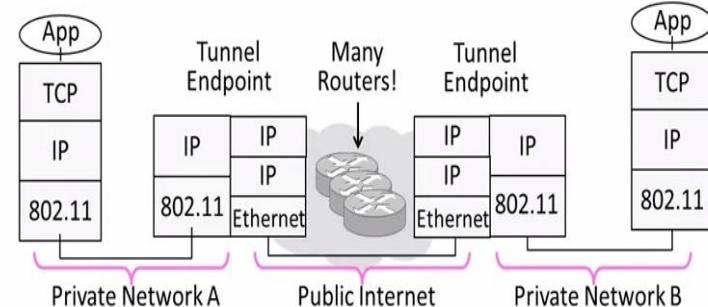
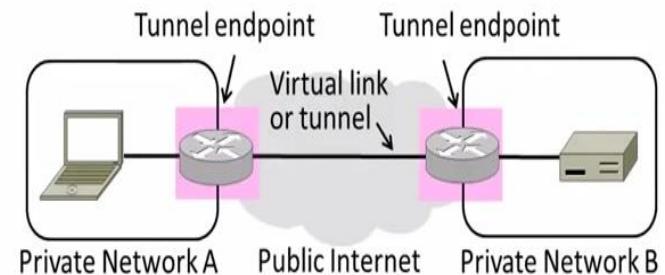
□ ALAPELV

- Bérelt vonalak helyett használjuk a publikusan hozzáférhető Internet-et.
- Így az Internettől **logikailag** elkülöníthető hálózatot kapunk. Ezek a virtuális magánhálózatok avagy VPN-ek.
- A célok közé kell felvenni a külső támadó kizárását.



Virtuális magánhálózatok alapok

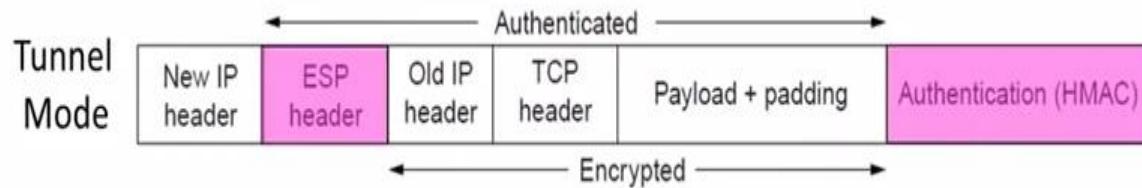
- A virtuális linkekkel alagutak képzésével valósítjuk meg.
- **ALAGÚTAK**
 - Egy magánhálózaton belül a hosztok egymásnak normál módon küldhetnek üzenetet.
 - Virtuális linken a végpontok beágyazzák a csomagokat.
 - IP az IP-be mechanizmus.
- Az alagutak képzése önmagában kevés a védelemhez. Mik a hiányosságok?
 - Bizalmasság, authentikáció
 - Egy támadó olvashat, küldhet üzeneteket.
 - Válasz: Kriptográfia használata.



Virtuális magánhálózatok alapok

□ IPSEC

- Hosszú távú célja az IP réteg biztonságossá tétele. (bizalmasság, autentikáció)
- Műveletei:
 - Hoszt párok kommunikációjához kulcsokat állít be.
 - A kommunikáció kapcsolatorientáltható tétele.
 - Fejlécek és láblécek hozzáadása az IP csomagok védelme érdekében.
- Több módot is támogat, amelyek közül az egyik az **alagút mód**.



Köszönöm a figyelmet!

Számítógépes Hálózatok

9. Előadás: ++Szállítói réteg

További protokollok

Internet Control Message Protocol

3

FELADATA

- Váratlan események jelentése

HASZNÁLAT

- Többféle ICMP-üzenetet definiáltak:
 - Elérhetetlen cél;
 - Időtúllépés;
 - Paraméter probléma;
 - Forráslefojtás;
 - Visszhang kérés;
 - Visszhang válasz;
 - ...

Internet Control Message Protocol

4

- *Elérhetetlen* cél esetén a csomag kézbesítése sikertelen volt.
 - ▣ **Esemény lehetséges oka:** Egy nem darabolható csomag továbbításának útvonalán egy „kis csomagos hálózat” van.
- *Időtúllépés* esetén az IP csomag élettartam mezője elérte a 0-át.
 - ▣ **Esemény lehetséges oka:** Torlódás miatt hurok alakult ki vagy a számláló értéke túl alacsony volt.
- *Paraméter probléma* esetén a fejrészben érvénytelen mezőt észleltünk.
 - ▣ **Esemény lehetséges oka:** Egy az útvonalon szereplő router vagy a hoszt IP szoftverének hibáját jelezheti.

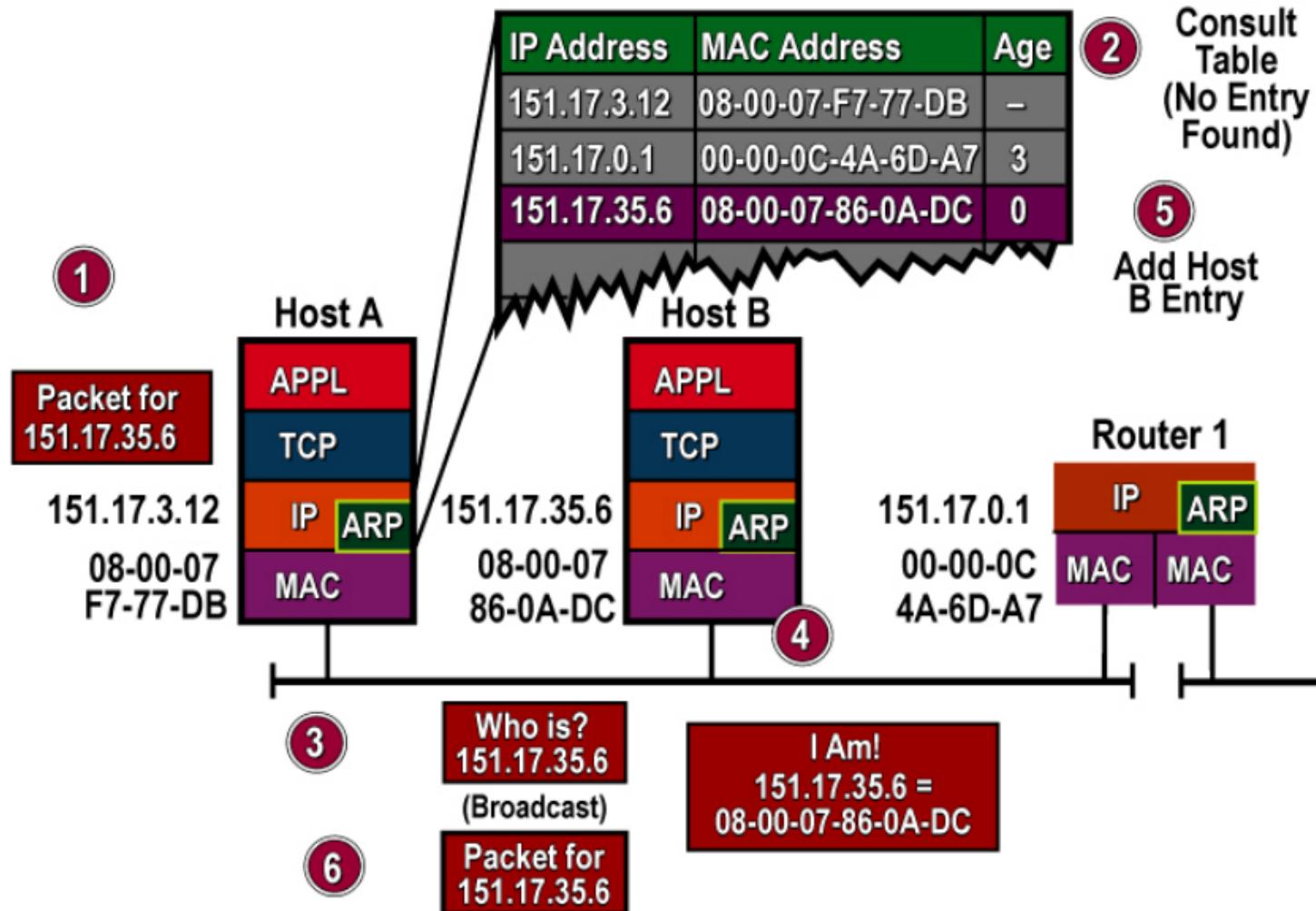
Internet Control Message Protocol

5

- Forráslefojtás esetén lefojtó csomagot küldünk.
 - ▣ **Esemény hatása:** A fogadó állomásnak a forgalmazását lassítania kellett.
- Visszhang kérés esetén egy hálózati állomás jelenlétét lehet ellenőrizni.
 - ▣ **Esemény hatása:** A fogadónak vissza kell küldeni egy visszhang választ.
- Átirányítás esetén a csomag rosszul irányítottságát jelzik.
 - **Esemény kiváltó oka:** Router észleli, hogy a csomag nem az optimális útvonal.

Address Resolution Protocol

6



Address Resolution Protocol

7

FELADATA

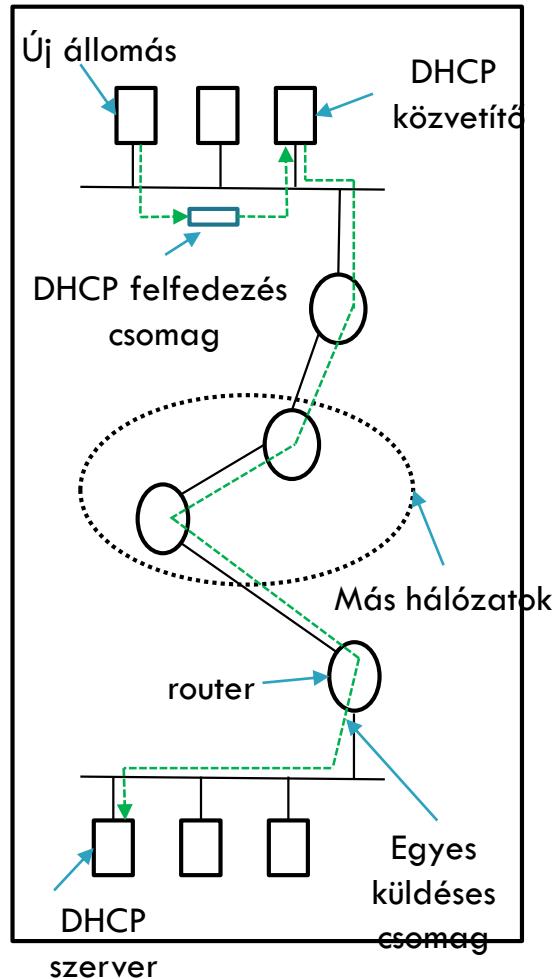
- Az IP cím megfeleltetése egy fizikai címnek.

HOZZÁRENDELÉS

- Adatszóró csomag kiküldése az Ethernetre „Ki-é a 192.60.34.12-es IP-cím?” kérdéssel az alhálózaton, és mindenegyes hoszt ellenőrzi, hogy övé-e a kérdéses IP-cím. Ha egyezik az IP a hoszt saját IP-jével, akkor a saját Ethernet címével válaszol. Erre szolgál az ARP.
- Opcionális javítási lehetőségek:
 - a fizikai cím IP hozzárendelések tárolása (cache használata);
 - Leképezések megváltoztathatósága (időhatály bevezetése);
- Mi történik távoli hálózaton lévő hoszt esetén?
 - A router is válaszoljon az ARP-re a hoszt alhálózatán. (proxy ARP)
 - Alapértelmezett Ethernet-cím használata az összes távoli forgalomhoz

Reverse Address Resolution Protocol

8



Reverse Address Resolution Protocol

FELADATA

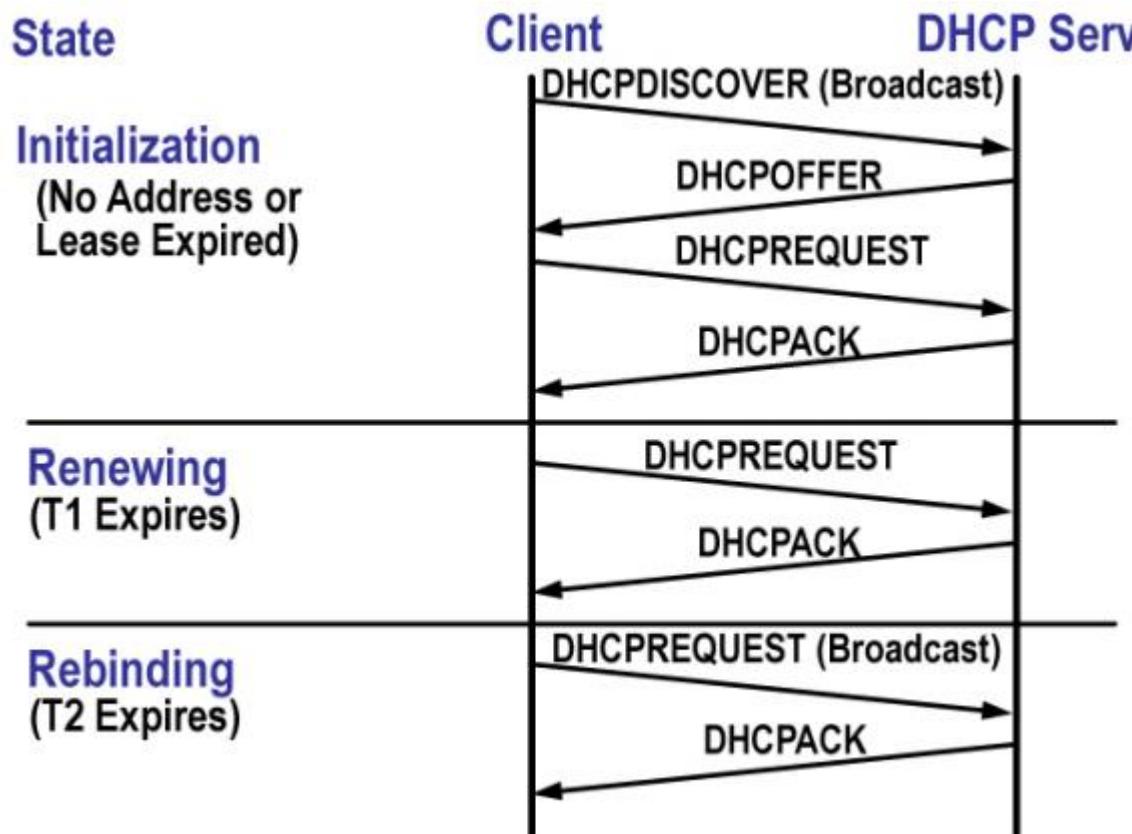
- A fizikai cím megfeleltetése egy IP címnek

HOZZÁRENDELÉS

- 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?” kérdéssel az alhálózaton. Az *RARP*-szerver pedig válaszol a megfelelő IP címmel, mikor meglátja a kérést
- Opcionális javítási lehetőségek:
 - *BOOTP* protokoll használata. UDP csomagok használata. Manuálisan kell a hozzárendelési táblázatot karbantartani. (statikus címkiosztás)
 - *DHCP* protokoll használata. Itt is külön kiszolgáló osztja ki a címeket a kérések alapján. A kiszolgáló és a kérő állomások nem kell hogy ugyanazon a *LAN*-on legyenek, ezért *LAN*-onként kell egy *DHCP relay agent*. (statikus és dinamikus címkiosztás)

DHCP: DYNAMIC HOST CONFIGURATION PROTOCOL

10



DHCP

11

- Lényegében ez már az **Alkalmazási réteg**
 - de logikailag ide tartozik
- Segítségével a hosztok automatikusan juthatnak hozzá a kommunikációjukhoz szükséges hálózati azonosítóhoz:
 - IP cím, hálózati maszk, alapértelmezett átjáró, stb.
- Eredetileg az RFC 1531 a BOOTP kiterjesztéseként definiálta. Újabb RFC-k: 1541, 2131 (aktuális)

DHCP lehetőségei

12

- IP címek osztása MAC cím alapján DHCP szerverrel
 - Szükség esetén (a DHCP szerveren előre beállított módon) egyes kliensek számára azok MAC címéhez fix IP cím rendelhető
- IP címek osztása dinamikusan
 - A DHCP szerveren beállított tartományból „érkezési sorrendben” kapják a kliensek az IP címeket
 - Elegendő annyi IP cím, ahány gép egyidejűleg működik
- Az IP címeiken kívül további szükséges hálózati paraméterek is kioszthatók
 - Hálózati maszk
 - Alapértelmezett átjáró
 - Névkiszolgáló
 - Domain név
 - Hálózati rendszerbetöltéshez szerver és fájlnév

DHCP – Címek bérlese

13

- A DHCP szerver a klienseknek az IP-címekeket bizonyos bérleti időtartamra (lease time) adja „bérbe”
 - Az időtartam hosszánál a szerver figyelembe veszi a kliens esetleges ilyen irányú kérését
 - Az időtartam hosszát a szerver beállításai korlátozzák
- A bérleti időtartam lejárta előtt a bérlet meghosszabbítható
- Az IP-cím explicit módon vissza is adható

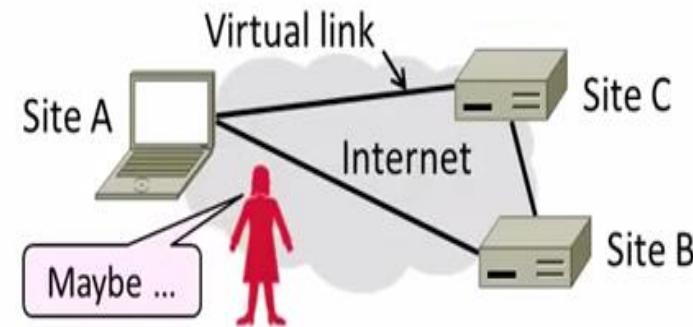
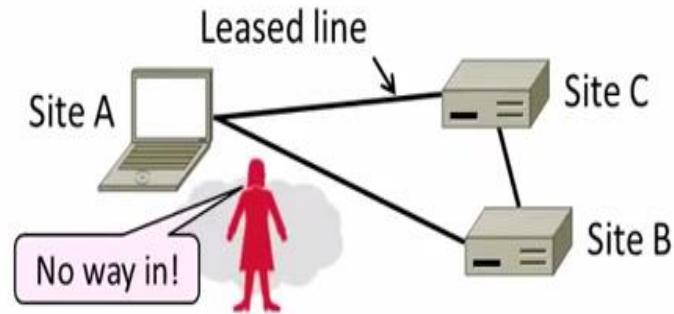
Virtuális magánhálózatok alapok

□ FŐ JELLEMZŐI

- Mint közeli hálózat fut az interneten keresztül.
- IPSEC-et használ az üzenetek titkosítására.
- Azaz informálisan megfogalmazva fizikailag távol lévő hosztok egy közös logikai egységet alkotnak.
 - Például távollévő telephelyek rendszerei.

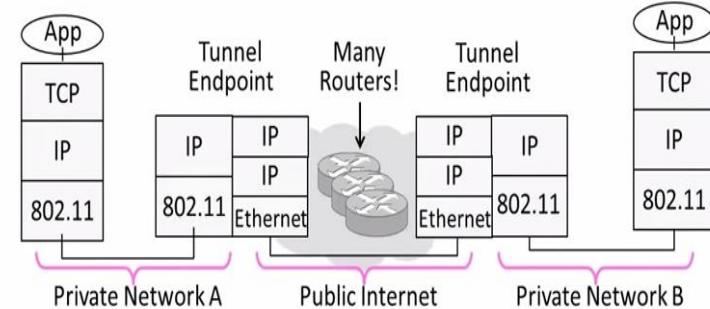
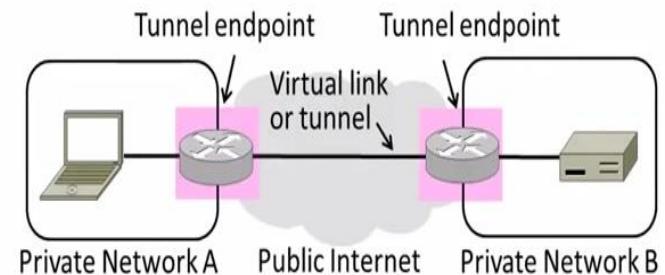
□ ALAPELV

- Bérelt vonalak helyett használjuk a publikusan hozzáférhető Internet-et.
- Így az Internettől **logikailag** elkülöníthető hálózatot kapunk. Ezek a virtuális magánhálózatok avagy VPN-ek.
- A célok közé kell felvenni a külső támadó kizárását.



Virtuális magánhálózatok alapok

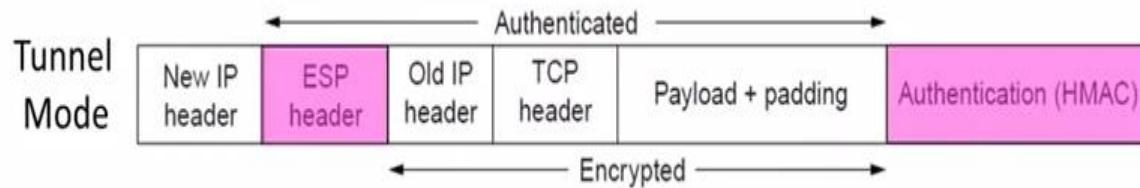
- A virtuális linkekkel alagutak képzésével valósítjuk meg.
- **ALAGÚTAK**
 - Egy magánhálózaton belül a hosztok egymásnak normál módon küldhetnek üzenetet.
 - Virtuális linken a végpontok beágyazzák a csomagokat.
 - IP az IP-be mechanizmus.
- Az alagutak képzése önmagában kevés a védelemhez. Mik a hiányosságok?
 - Bizalmasság, authentikáció
 - Egy támadó olvashat, küldhet üzeneteket.
 - Válasz: Kriptográfia használata.



Virtuális magánhálózatok alapok

□ IPSEC

- Hosszú távú célja az IP réteg biztonságossá tétele. (bizalmasság, autentikáció)
- Műveletei:
 - Hoszt párok kommunikációjához kulcsokat állít be.
 - A kommunikáció kapcsolatorientáltható tétele.
 - Fejlécek és láblécek hozzáadása az IP csomagok védelme érdekében.
- Több módot is támogat, amelyek közül az egyik az **alagút mód**.



Szállítói réteg

17



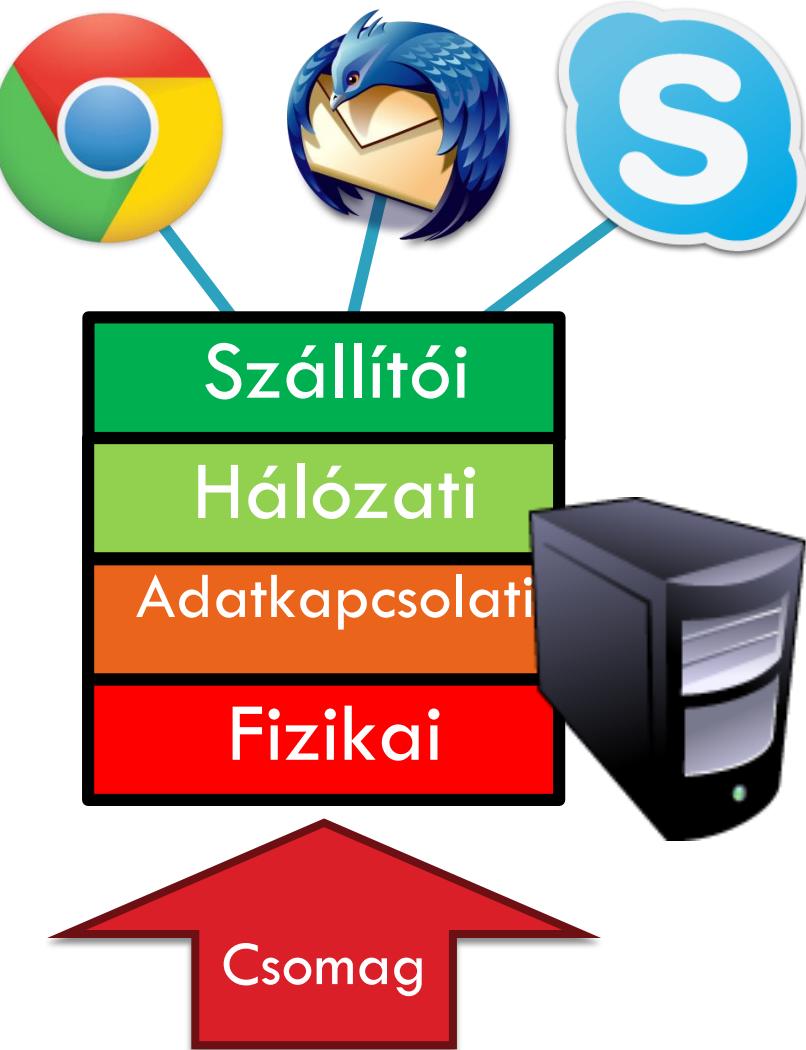
- **Feladat:**
 - ▣ Adatfolyamok demultiplexálása
- **További lehetséges feladatok:**
 - ▣ Hosszú élettartamú kapcsolatok
 - ▣ Megbízható, sorrendhelyes csomag leszállítás
 - ▣ Hiba detektálás
 - ▣ Folyam és torlódás vezérlés
- **Kihívások:**
 - ▣ Torlódások detektálása és kezelése
 - ▣ Fairség és csatorna kihasználás közötti egyensúly

- ❑ UDP
- ❑ TCP
- ❑ Torlódás vezérlés
- ❑ TCP evolúciója
- ❑ A TCP problémái

Multiplexálás

19

- Datagram hálózat
 - Nincs áramkör kapcsolás
 - Nincs kapcsolat
- A kliensek számos alkalmazást futtathatnak egyidőben
 - Kinek szállítsuk le a csomagot?
- IP fejléc “protokoll” mezője
 - 8 bit = 256 konkurens folyam
 - Ez nem elég...
- Demultiplexálás megoldása a szállítói réteg feladata



Forgalom demultiplexálása

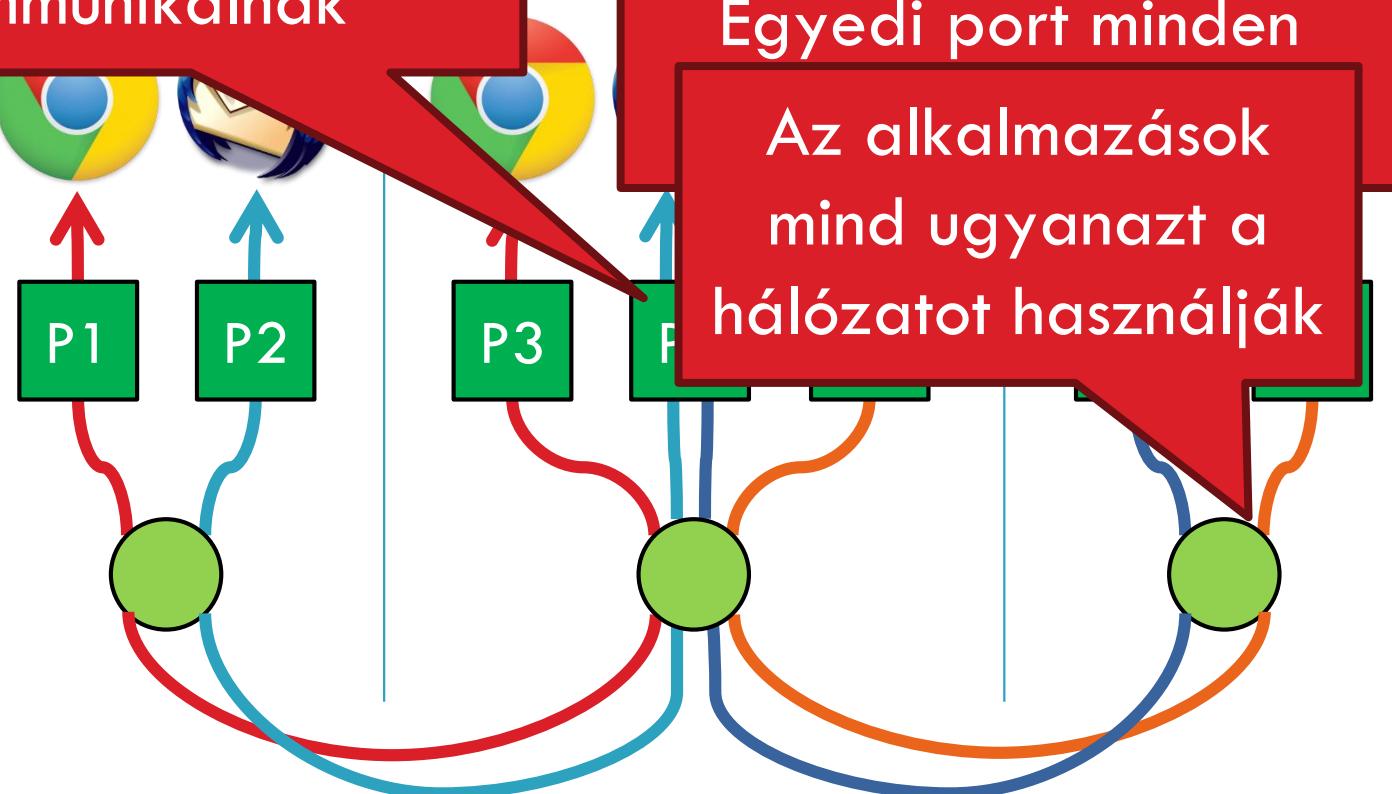
20

A szerver alkalmazások
számos klienssel
kommunikálnak

Alkalmazási

Szállítói

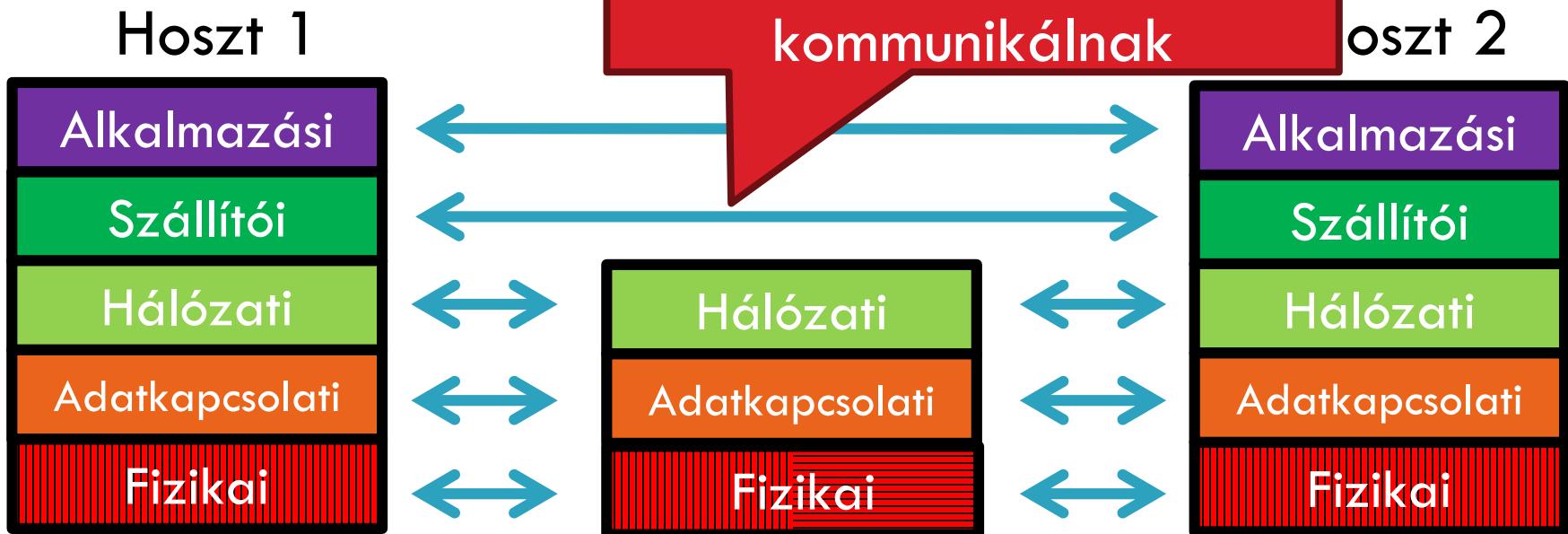
Hálózati



Végpontok azonosítása: $\langle \text{src_ip}, \text{src_port}, \text{dest_ip}, \text{dest_port}, \text{proto} \rangle$
ahol src_ip , dst_ip a forrás és cél IP cím,
 src_port , dest_port forrás és cél port, proto pedig UDP vagy TCP.

Réteg modellek

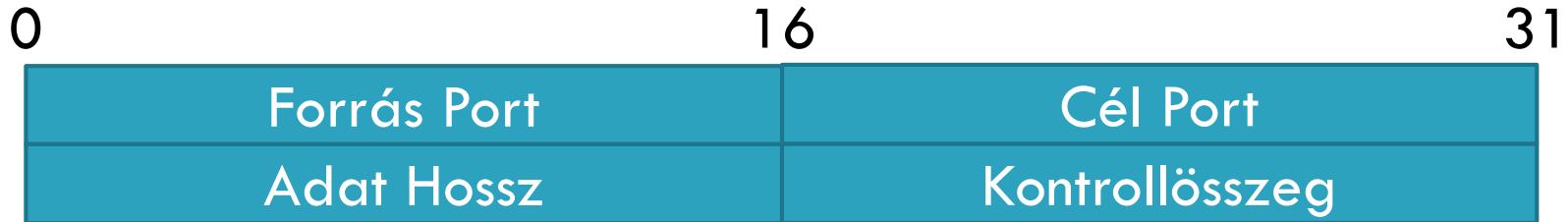
21



- A legalacsonyabb szintű végpont-végpont protokoll
 - A szállítói réteg fejlécei csak a forrás és cél végpontok olvassák
 - A routerek számára a szállítói réteg fejléce csak szállítandó adat (payload)

User Datagram Protocol (UDP)

22



- 8 bájtos UDP fejléc
- Egyszerű, kapcsolatnélküli átvitel
 - C socketek: SOCK_DGRAM
- Port számok teszik lehetővé a demultiplexálást
 - 16 bit = 65535 lehetséges port
 - 0 port nem engedélyezett
- Kontrollösszeg hiba detektáláshoz
 - Hibás csomagok felismerése
 - Nem detektálja az elveszett, duplikátum és helytelen sorrendben beérkező csomagokat (UDP esetén nincs ezekre garancia)

UDP felhasználások

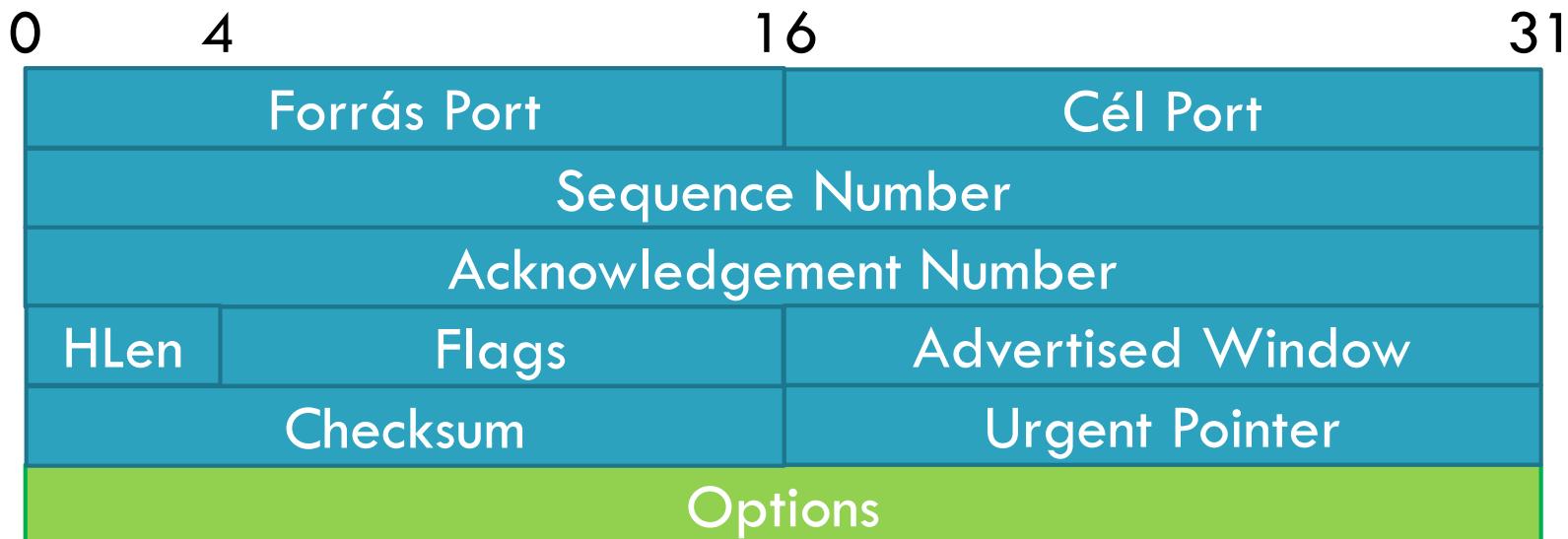
23

- A TCP után vezették be
 - Miért?
- Nem minden alkalmazásnak megfelelő a TCP
- UDP felett egyedi protokollok valósíthatók meg
 - Megbízhatóság? Helyes sorrend?
 - Folyam vezérlés? Torlódás vezérlés?
- Példák
 - RTMP, real-time média streamelés (pl. hang, video)
 - Facebook datacenter protocol

Transmission Control Protocol

24

- Megbízható, sorrend helyes, két irányú bájt folyamok
 - Port számok a demultiplexáláshoz
 - Kapcsolat alapú
 - Folyam vezérlés
 - Torlódás vezérlés, fair viselkedés
- 20 bájtos fejléc + options fejlécek



Kapcsolat felépítés

25

- Miért van szükség kapcsolat felépítésre?
 - Állapot kialakítása minden két végponton
 - Legfontosabb állapot: sorszámok/sequence numbers
 - Az elküldött bájtok számának nyilvántartása
 - Véletlenszerű kezdeti érték
- Fontos TCP flag-ek/jelölő bitek (1 bites)
 - SYN – szinkronizációs, kapcsolat felépítéshez
 - ACK – fogadott adat nyugtázása
 - FIN – vége, kapcsolat lezárásához

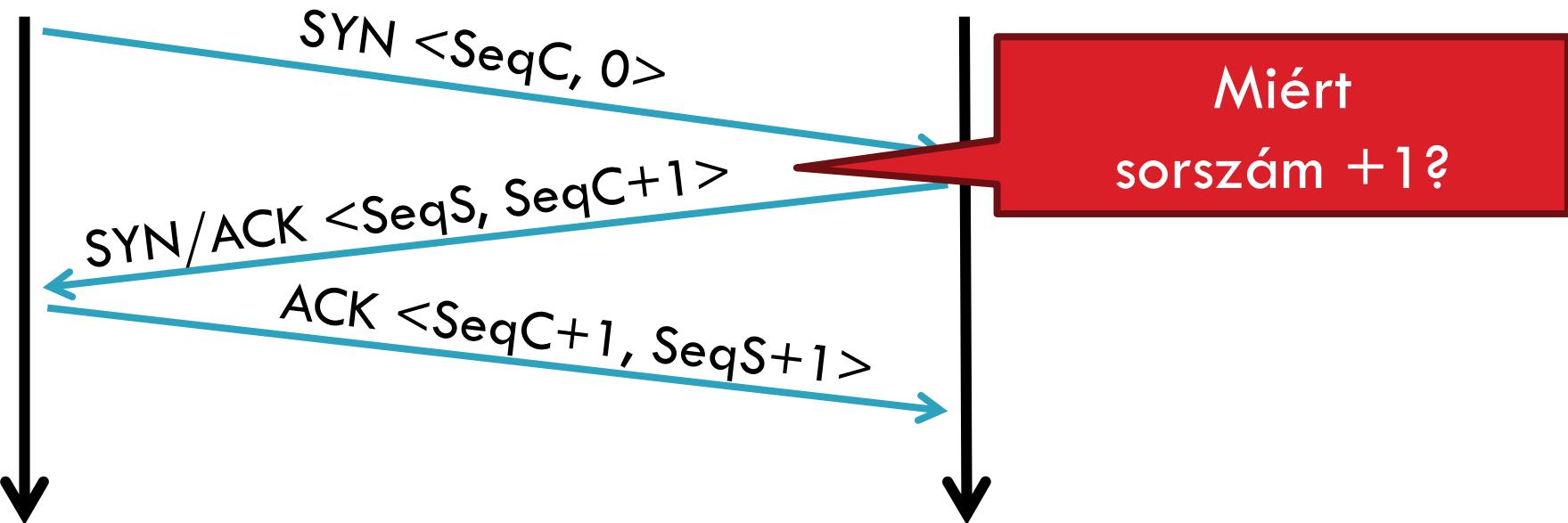
Three Way Handshake

Három-utas kézfogás

26

Kliens

Szerver



□ Mindkét oldalon:

- Másik fél értesítése a kezdő sorszámról
- A másik fél kezdő sorszámának nyugtázása

Kapcsolat felépítés problémája

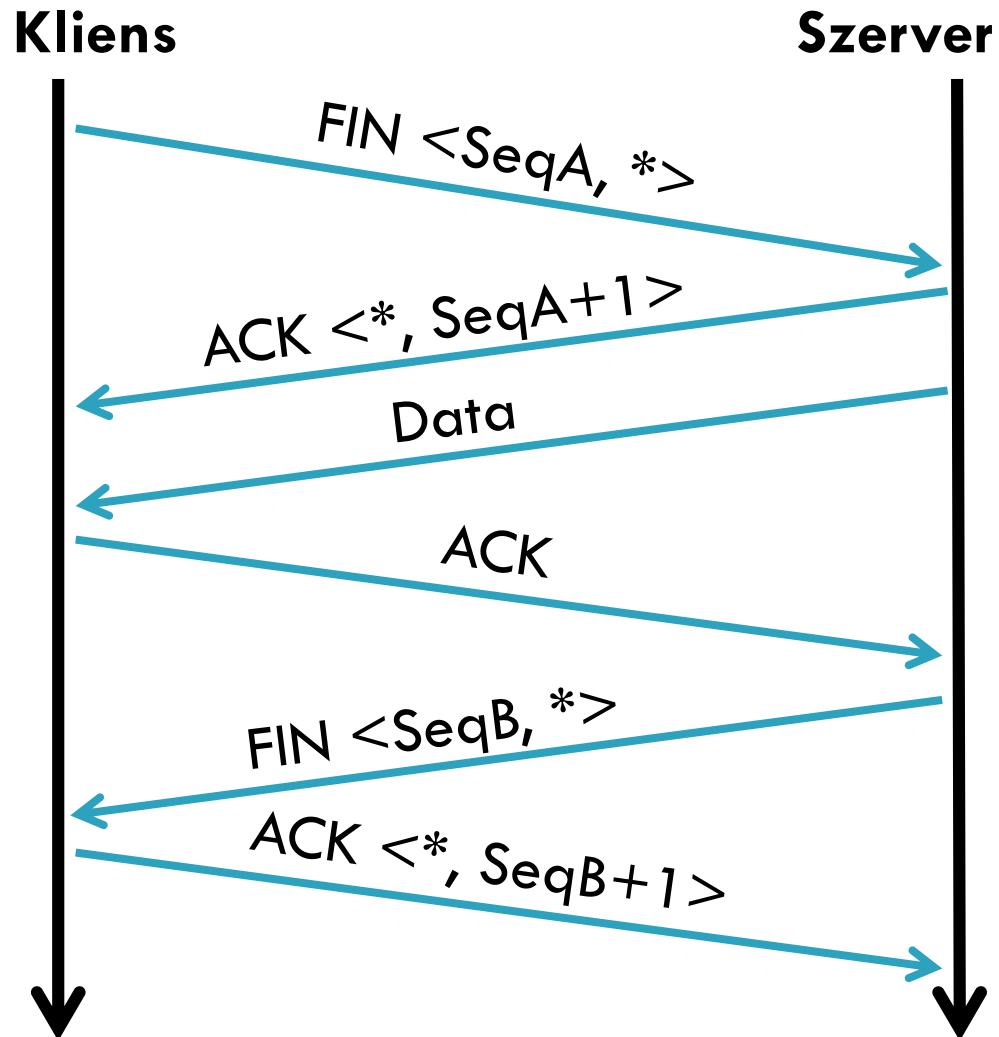
27

- Kapcsolódási zűrzavar
 - Azonos hoszt kapcsolatainak egyértelműsítése
 - Véletlenszerű sorszámmal - biztonság
- Forrás hamisítás
 - Kevin Mitnick
 - Jó random szám generátor kell hozzá!
- Kapcsolat állapotának kezelése
 - minden SYN állapotot foglal a szerveren
 - SYN flood = denial of service (DoS) támadás
 - Megoldás: SYN cookies

Kapcsolat lezárása

28

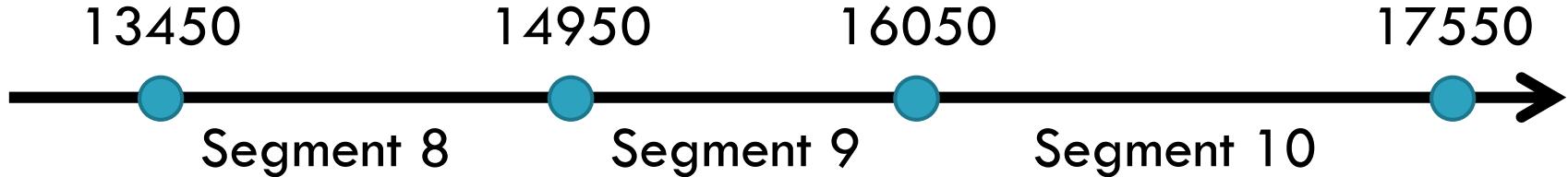
- Mindkét oldal kezdeményezheti a kapcsolat bontását
- A másik oldal még folytathatja a küldést
 - Félig nyitott kapcsolat
 - `shutdown()`
- Az utolsó FIN nyugtáza
 - Sorszám + 1
- Mi történik, ha a 2. FIN elveszik?



Sorszámok tere

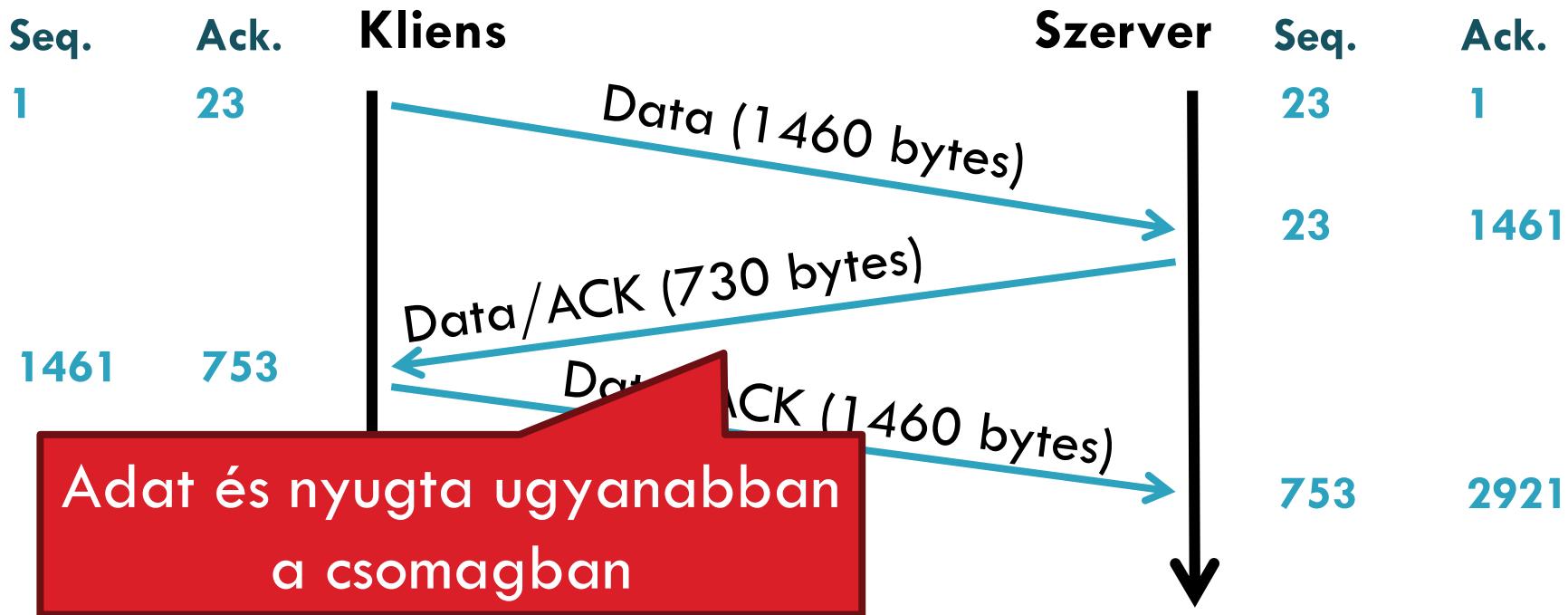
29

- A TCP egy absztrakt bájt folyamot valósít meg
 - A folyam minden bájtja számozott
 - 32-bites érték, körbefordul egy idő után
 - Kezdetben, véletlen érték a kapcsolat felépítésénél.
- A bájt folyamot szegmensekre bontjuk (TCP csomag)
 - A méretét behatárolja a Maximum Segment Size (MSS)
 - Úgy kell beállítani, hogy elkerüljük a fregmentációt
- minden szegmens egyedi sorszámmal rendelkezik



Kétirányú kapcsolat

30



- Mindkét fél küldhet és fogadhat adatot
- Különböző sorszámok a két irányba

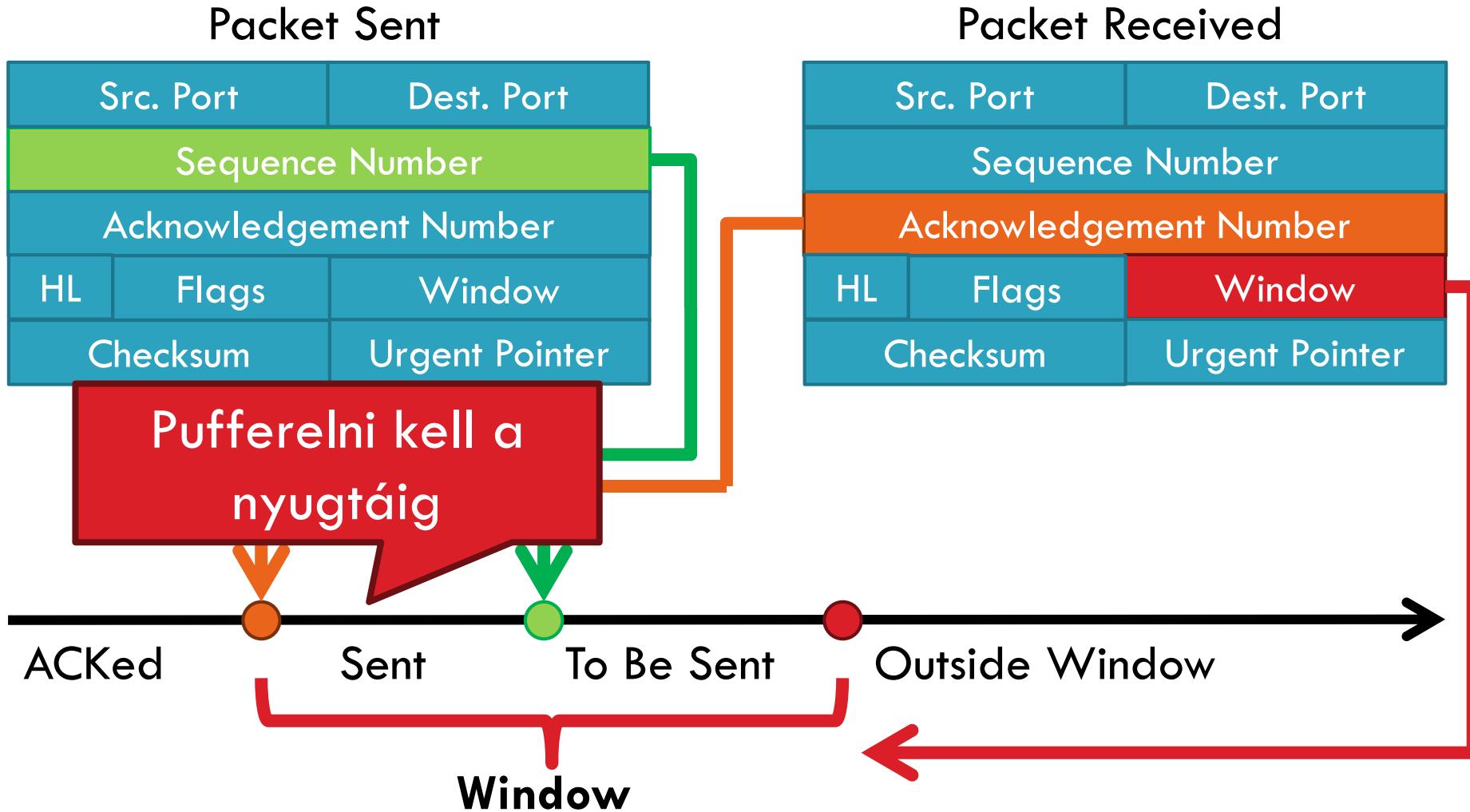
Folyam vezérlés

31

- Probléma: Hány csomagot tud a küldő átvinni?
 - Túl sok csomag túlterhelheti a fogadót
 - A fogadó oldali puffer-méret változhat a kapcsolat során
- Megoldás: csúszóablak
 - A fogadó elküldi a küldőnek a pufferének méretét
 - Ezt nevezzük meghirdetett ablaknak: **advertised window**
 - Egy n ablakmérethez, a küldő n bájtot küldhet el ACK fogadása nélkül
 - minden egyes ACK után, léptetjük a csúszóablakot
- Az ablak akár nulla is lehet!

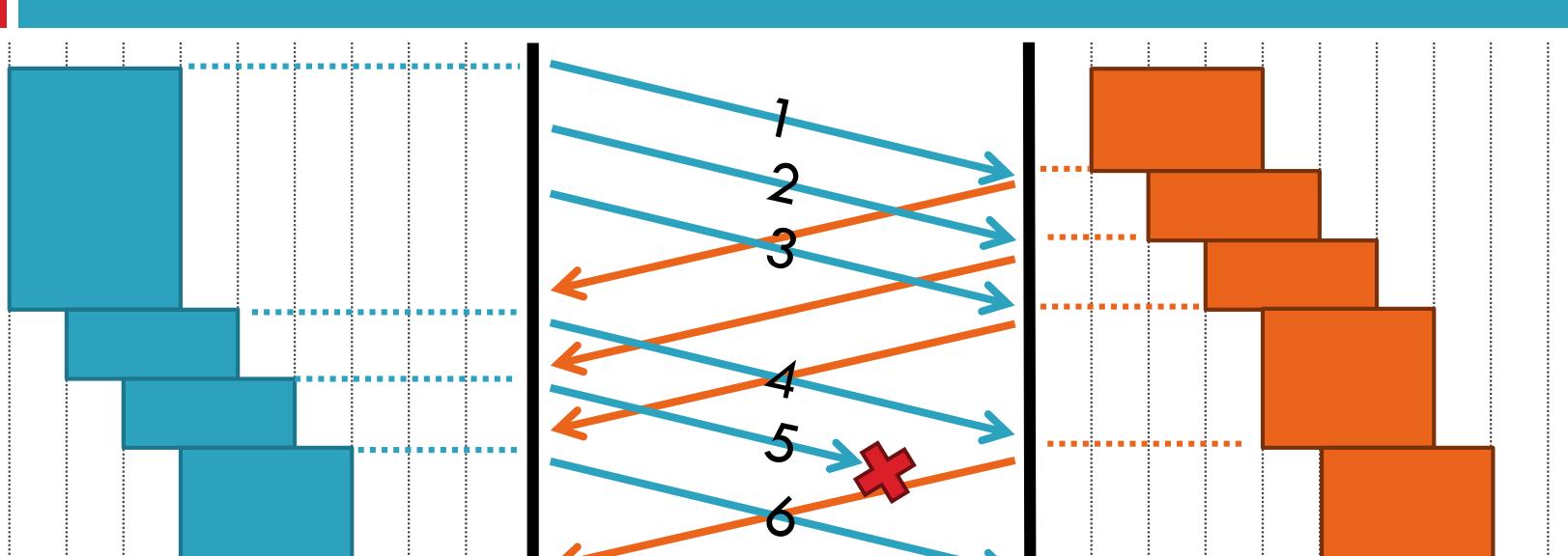
Folyam vezérlés - csúszóablak

32



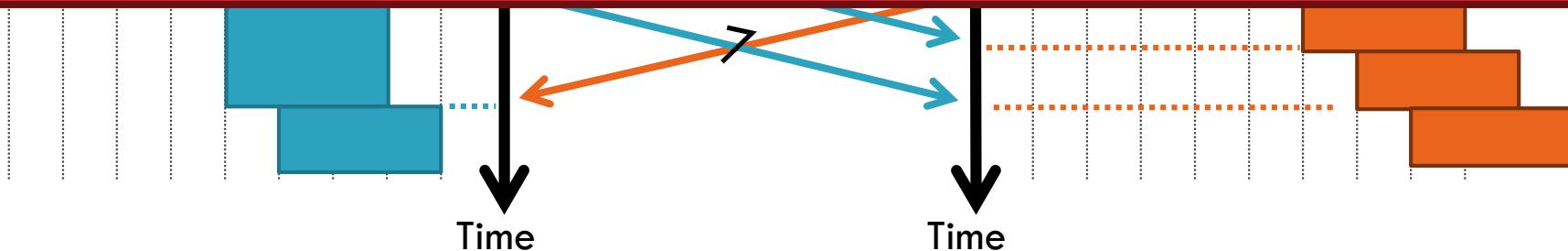
Csúszóablak példa

33



A TCP ACK ütemezett

- Rövid RTT → gyors ACK → az ablak gyorsan léptethető
- Hosszú RTT → lassú ACK → az ablak csak lassan „csúszik”



Megfigyelések

34

- Átvitel arányos $\sim w/RTT$
 - w : küldési ablakméret
 - RTT: körülfordulási idő
- A küldőnek pufferelni kell a nem nyugtázott csomagokat a lehetséges újraküldések miatt
- A fogadó elfogadhat nem sorrendben érkező csomagokat, de csak amíg az elfér a pufferben

Mit nyugtázhat a fogadó?

35

1. minden egyes csomagot
2. Használhat *kumulált nyugtát*, ahol egy n sorszámú nyugta minden $k < n$ sorszámú csomagot nyugtáz
3. Használhat *negatív nyugtát* (NACK), megjelölve, hogy mely csomag nem érkezett meg
4. Használhat *szelektív nyugtát* (SACK), jelezve, hogy mely csomagok érkeztek meg, akár nem megfelelő sorrendben
 - SACK egy TCP kiterjesztés
 - SACK TCP

Sorszámok

36

- 32 bites, unsigned
 - Miért ilyen nagy?
- A csúszó-ablakhoz szükséges...
 - $|\text{sorszámok tere}| > 2 * |\text{Küldő ablak mérete}|$
 - $2^{32} > 2 * 2^{16}$
- Elkóborolt csomagok kivédése
 - IP csomagok esetén a maximális élettartam (MSL) of 120 mp
 - Azaz egy csomag 2 percig bolyonghat egy hálózatban

Buta ablak szindróma

37

- Mi van, ha az ablak mérete nagyon kicsi?
 - Sok, apró csomag. A fejlécek dominálják az átvitelt.



- Lényegében olyan, mintha bájtonként küldenénk az üzenetet...
 1. `for (int x = 0; x < strlen(data); ++x)`
 2. `write(socket, data + x, 1);`

Nagle algoritmusa

38

1. Ha az ablak \geq MSS és az elérhető adat \geq MSS:
Küldjük el az adatot Egy teljes csomag küldése
 2. Különben ha van nem nyugtáztatott adat::
Várakoztassuk az adatot egy pufferben, amíg nyugtát nem kapunk
 3. Különben: küldjük az adatot Küldjünk egy nem teljes csomagot, ha nincs más
- Probléma: Nagle algoritmusa késlelteti az átvitelt
- Mi van, ha azonnal el kell küldeni egy csomagot?
1. int flag = 1;
 2. setsockopt(sock, IPPROTO_TCP, TCP_NODELAY, (char *) &flag, sizeof(int));

Hiba detektálás

39

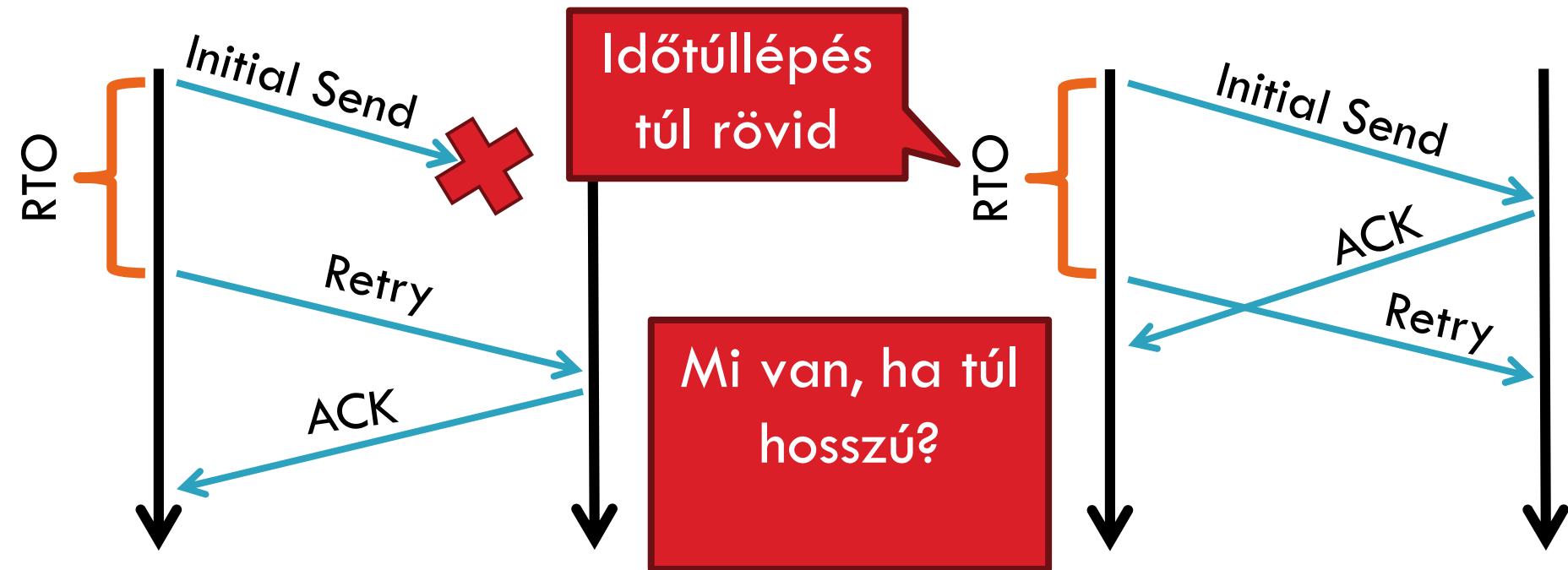
- A kontrollösszeg detektálja a hibás csomagokat
 - ▣ Az IP, TCP fejlécből és az adatból számoljuk
- A sorszámok segítenek a sorrendhelyes átvitelben
 - ▣ Duplikátumok eldobása
 - ▣ Helytelen sorrendben érkező csomagok sorba rendezése vagy eldobása
 - ▣ Hiányzó sorszámok elveszett csomagot jeleznek
- A küldő oldalon: elveszett csomagok detektálása
 - ▣ Időtúllépés (timeout) használata hiányzó nyugtákhoz
 - ▣ Szükséges az RTT becslése a időtúllépés beállításához
 - ▣ minden nem nyugtázott csomagot pufferelni kell a nyugtáig

Retransmission Time Outs (RTO)

Időtúllépés az újraküldéshez

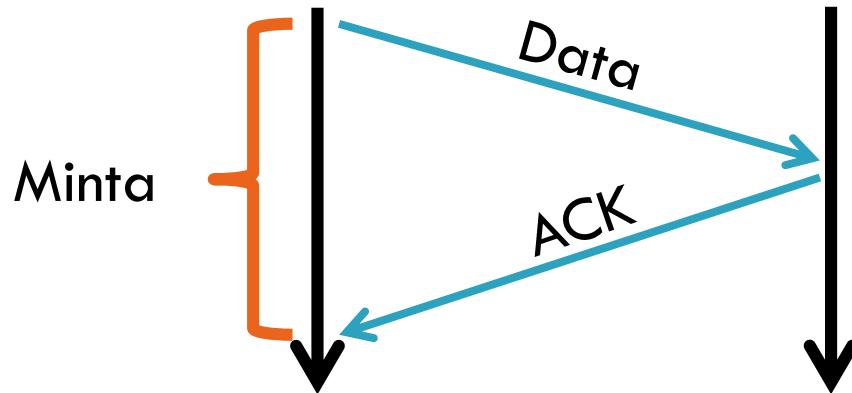
40

- Probléma: Időtúllépés RTT-hez kapcsolása



Round Trip Time becslés

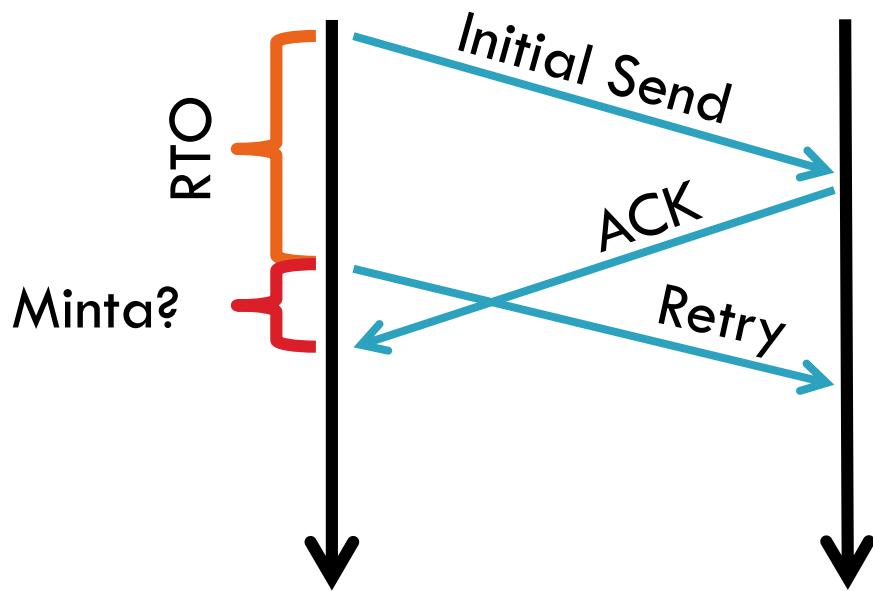
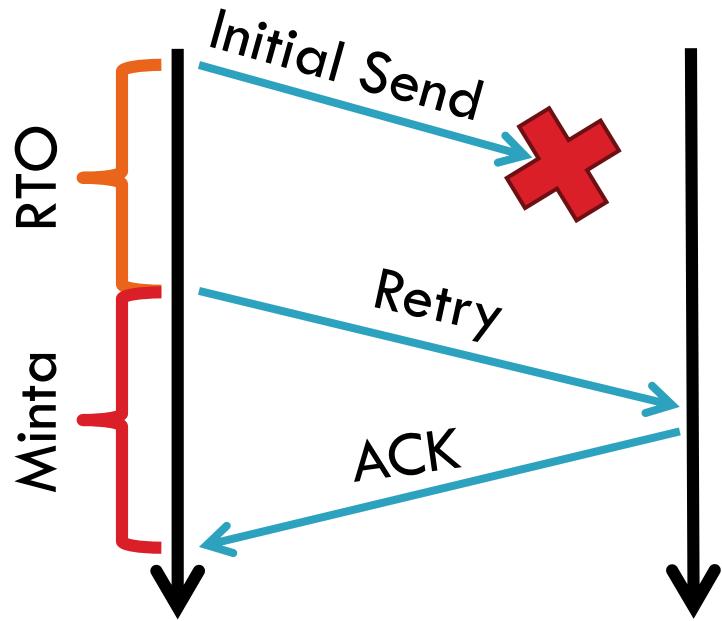
41



- Az eredeti TCP RTT becslője:
 - RTT becslése mozgó átlaggal
 - $\text{new_rtt} = \alpha (\text{old_rtt}) + (1 - \alpha)(\text{new_sample})$
 - Javasolt α : 0.8-0.9 (0.875 a legtöbb TCP esetén)
- $\text{RTO} = 2 * \text{new_rtt}$ (a TCP konzervatív becslése)

Az RTT minta félre is értelmezhető

42



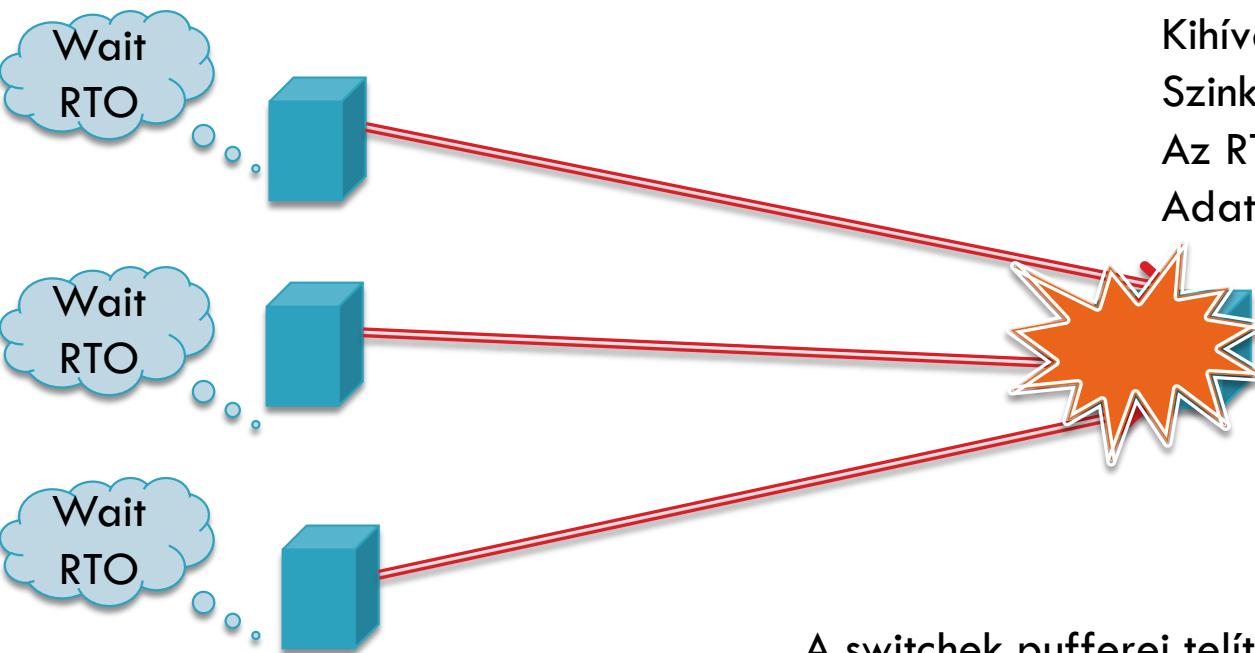
- ❑ Karn algoritmusá: dobjuk el azokat a mintákat, melyek egy csomag újraküldéséből származnak

RTO adatközpontokban???

43

- TCP Incast probléma – pl. Hadoop, Map Reduce, HDFS, GFS

Sok szimultán küldő egy fogadóhoz



Kihívás:

Szinkronizáció megtörése

Az RTO becslést WAN-ra terveztek

Adatközpontban sokkal kisebb RTT van

1-2ms vagy kevesebb

A switchek pufferei telítődnek és csomagok vesznek el!
Nyugta nem megy vissza 😞

Mi az a torlódás?

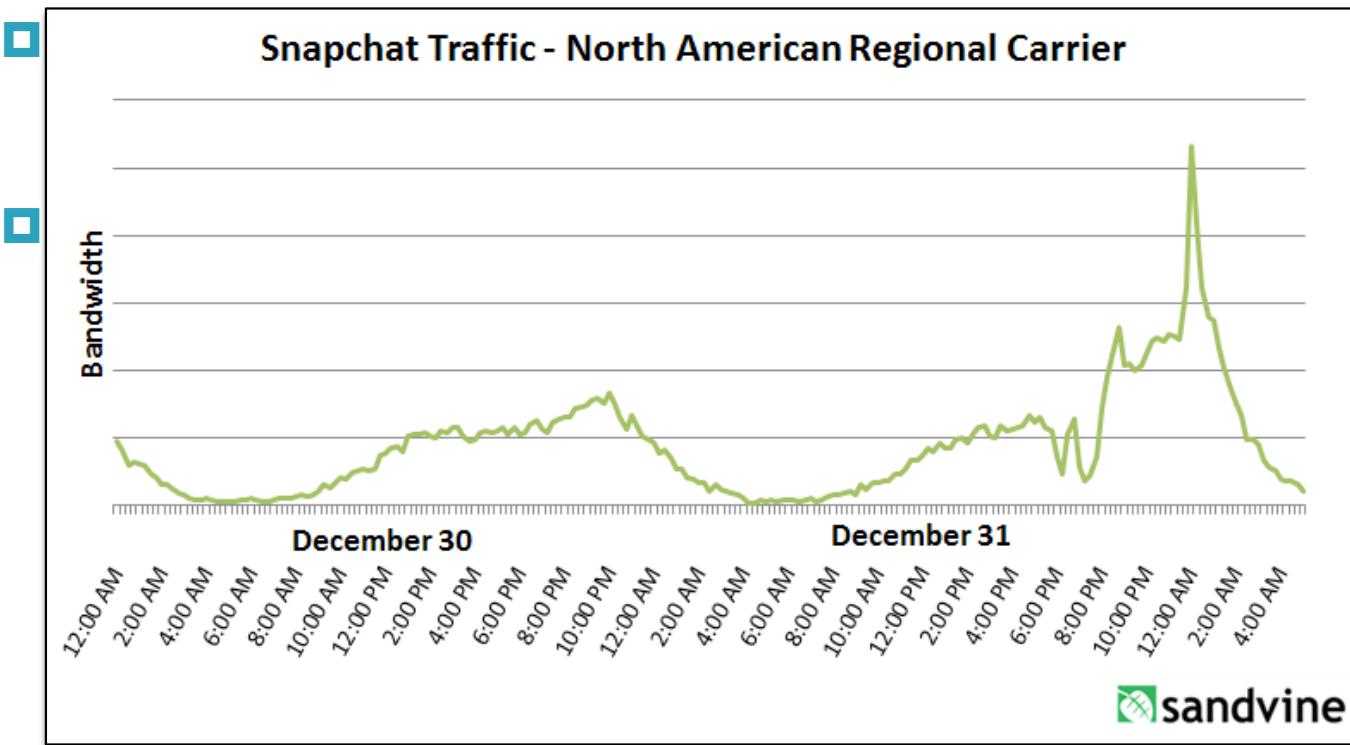
44

- A hálózat terhelése nagyobb, mint a kapacitása
 - ▣ A kapacitás nem egyenletes a hálózatban
 - Modem vs. Cellular vs. Cable vs. Fiber Optics
 - ▣ Számos folyam verseng a sávszélességért
 - otthoni kábel modem vs. corporate datacenter
 - ▣ A terhelés időben nem egyenletes
 - Vasárnap este 10:00 = BitTorrent Game of Thrones

Mi az a torlódás?

45

- A hálózat terhelése nagyobb, mint a kapacitása
 - A kapacitás nem egyenletes a hálózatban
 - Modem vs. Cellular vs. Cable vs. Fiber Optics



Miért rossz a torlódás?

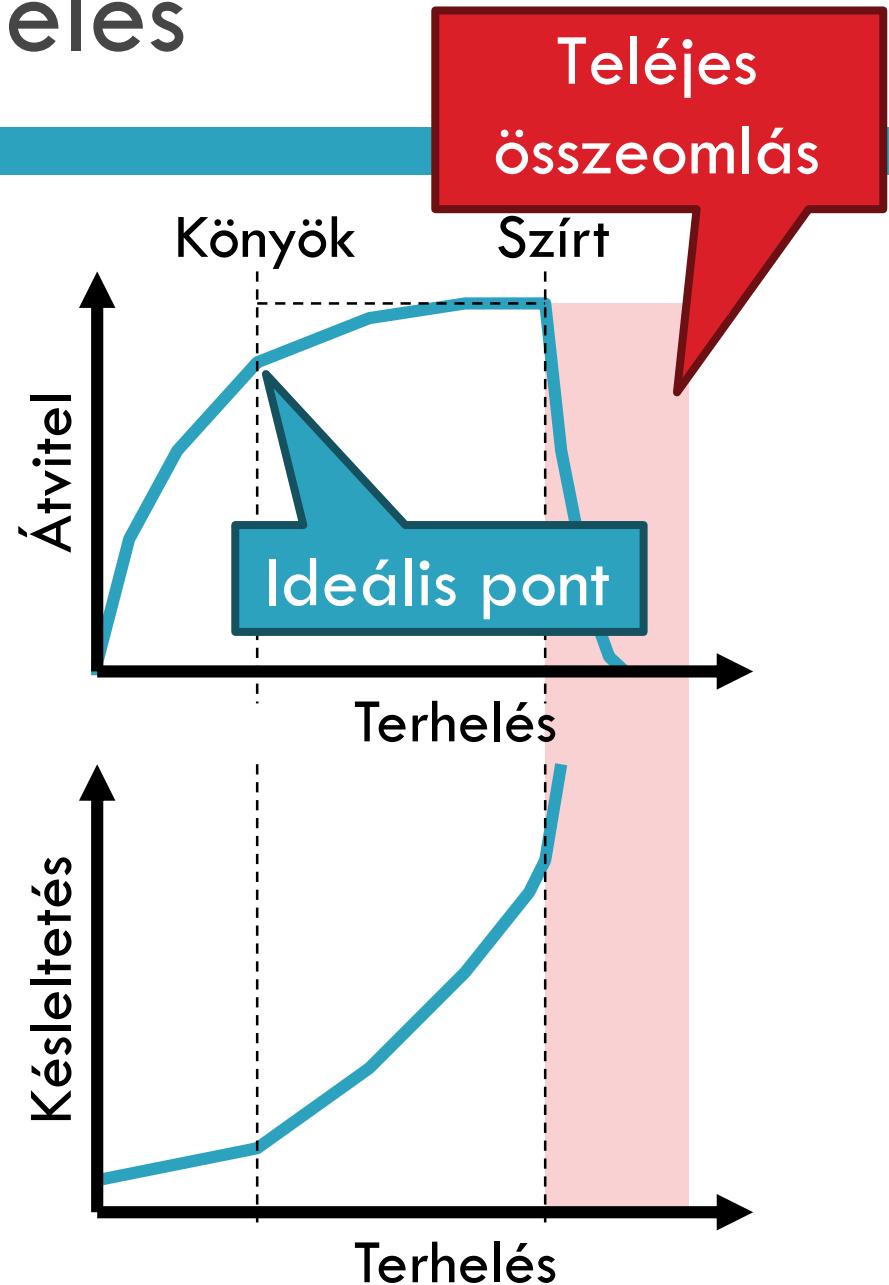
46

- Csomagvesztést eredményez
 - A routerek véges memóriával (puffer) rendelkeznek
 - Önhasonló Internet forgalom, nincs puffer, amiben ne okozna csomagvesztést
 - Ahogy a routerek pufferre elkezd telítődni, csomagokat kezd eldobni... (RED)
- Gyakorlati következmények
 - A routerek sorai telítődnek, megnövekedett késleltetés
 - Sávszélesség pazarlása az újraküldések miatt
 - Alacsony hálózati átvitel (goodput)

Megnövekedett terhelés

47

- Könyök („knee”)- a pont, ami után
 - Az átvitel szinte alig nő
 - Késleltetés viszont gyorsan emelkedik
- Egy egyszerű sorban ($M/M/1$)
 - Késleltetés = $1/(1 - \text{utilization})$
- Szírt („cliff”)- a pont, ami után
 - Átvitel lényegében leesik 0-ra
 - A késleltetés pedig $\rightarrow \infty$

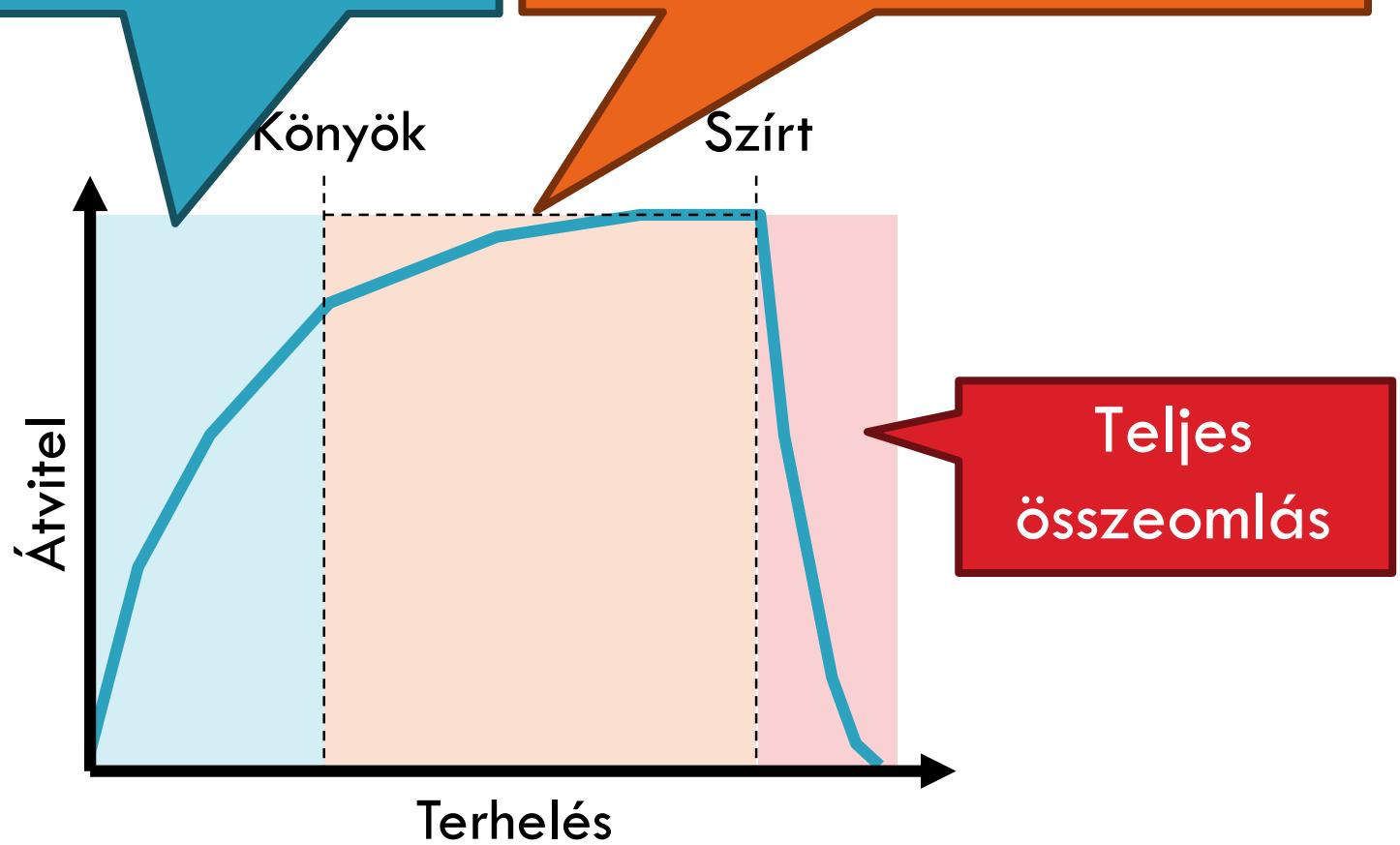


Torlódás vezérlés vs torlódás elkerülés

48

Torlódás elkerülés:
Maradj a könyök bal oldalán

Torlódás vezérlés
Maradj a szírt bal oldalán



Advertised Window

Meghirdetett ablak, újragondolva

49

- Megoldja-e a torlódás problémáját a TCP esetén a meghirdetett ablak használata?

NEM

- Ez az ablak csak a fogadót védi a túlterheléstől
- Egy kellően gyors fogadó kimaxolhatja ezt az ablakot
 - Mi van, ha a hálózat lassabb, mint a fogadó?
 - Mi van, ha vannak konkurens folyamok is?
- Következmények
 - Az ablak méret határozza meg a küldési rátát
 - Az ablaknak állíthatónak kell lennie, hogy elkerüljük a torlódás miatti teljes összeomlást...

Általános megoldások

50

- Ne csinálunk semmit, küldjük a csomagokat megkülönböztetés nélkül
 - Nagy csomagvesztés, jósolhatatlan teljesítmény
 - Teljes összeomláshoz vezethet
- Erőforrás foglalás
 - Folyamokhoz előre sávszélességet allokálunk
 - Csomagküldés előtt egy tárgyalási szakaszra is szükség van
 - Hálózati támogatás kell hozzá
- Dinamikus beállítás
 - Próbák használata a torlódási szint megbecséléshöz
 - Gyorsítás, ha torlódási szint alacsony
 - Lassítás, amint nő a torlódás
 - Nem rendezett dinamika, elosztott koordináció

TCP Torlódásvezérlés

51

- minden TCP kapcsolat rendelkezik egy ablakkal
 - A nem-nyugtázott csomagok számát vezérli
- Küldési ráta \sim window/RTT
- Ötlet: ablak méretének változtatása a küldési ráta vezérléséhez
- Vezessünk be egy **torlódási ablakot (congestion window)** a küldő oldalon
 - Torlódás vezérlés egy küldő oldali probléma
 - Jelölése: cwnd

Két fő komponens

52

1. Torlódás detektálás

- Eldobott csomag egy megbízható jel
 - Késleltetés alapú megoldások – nehéz és kockázatos
- Hogyan detektáljuk a csomag eldobását? Nyugtával
 - Időkorlát lejár ACK fogadása nélkül
 - Számos duplikált ACK jön be sorban (később lesz róla szó)

2. Ráta beállító algoritmus

- *cwnd* módosítása
- Sávszélesség próba
- Válasz lépés a torlódásra

Ráta vezérlés

53

- Tudjuk, hogy a TCP ACK ütemezett
 - Torlódás = késleltetés = hosszú várakozás a nyugták között
 - Nincs torlódás = alacsony késleltetés = gyors ACK
- Alapvető algoritmus
 - ACK fogadása esetén: növeljük a cwnd ablakot
 - Adat leszállítva, valószínűleg gyorsabban is küldhetünk
 - cwnd növekedése arányos az RTT-vel
 - Csomagvesztés esetén: csökkentsük a cwnd ablakot
 - Adat elveszett, torlódásnak kell lennie a hálózatban
- Kérdés: milyen függvényt használjuk a növeléshez és csökkentéshez? !!!!

Torlódás vezérlés megvalósítása

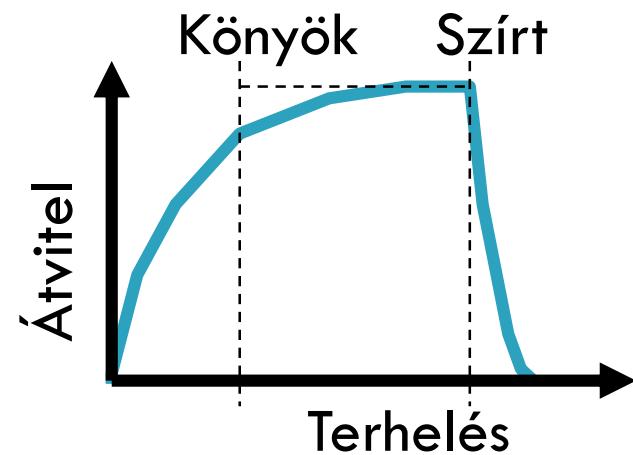
54

- Három változót kell nyilvántartani:
 - cwnd: torlódási ablak
 - adv_wnd: a fogadó meghirdetett ablaka
 - ssthresh: vágási érték (a cwnd frissítésére használjuk)
- Küldésnél használjuk: $wnd = \min(cwnd, adv_wnd)$
- A torlódás vezérlés két fázisa:
 1. Lassú indulás („Slow start”) ($cwnd < ssthresh$)
 - Az ún. bottleneck (legszűkebb) sávszélesség meghatározása a cél.
 2. Torlódás elkerülés ($cwnd \geq ssthresh$)
 - AIMD – Additive Increase Multiplicative Decrease

Lassú indulás - Slow Start

55

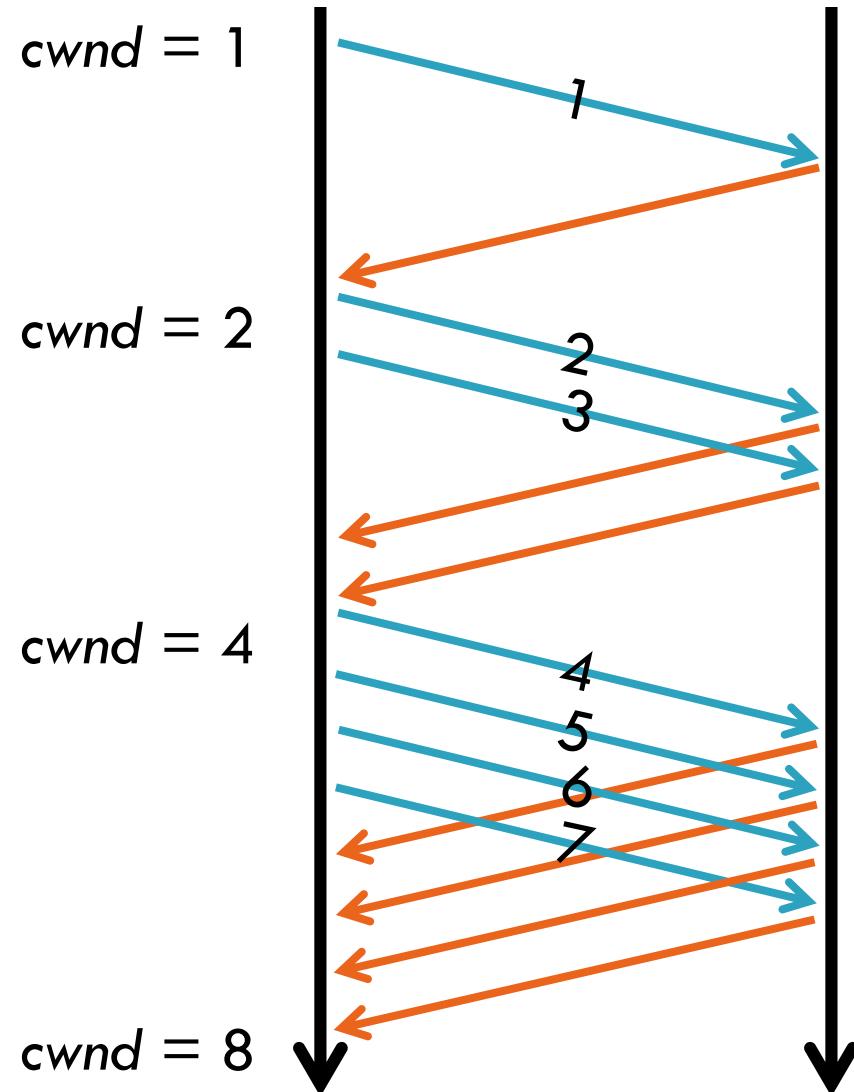
- Cél, hogy gyorsan elérjük a könyök pontot
- Egy kapcsolat kezdetén (vagy újraindításakor)
 - $cwnd = 1$
 - $ssthresh = adv_wnd$
 - minden nyugtázott szegmensre: $cwnd++$
- Egészen addig amíg
 - El nem érjük az $ssthresh$ értéket
 - Vagy csomagvesztés nem történik
- A Slow Start valójában nem lassú
 - $cwnd$ exponenciálisan nő



Slow Start példa

56

- $cwnd$ gyorsan nő
- Lelassul, amikor...
 - $cwnd \geq ssthresh$
 - Vagy csomagvesztés történik



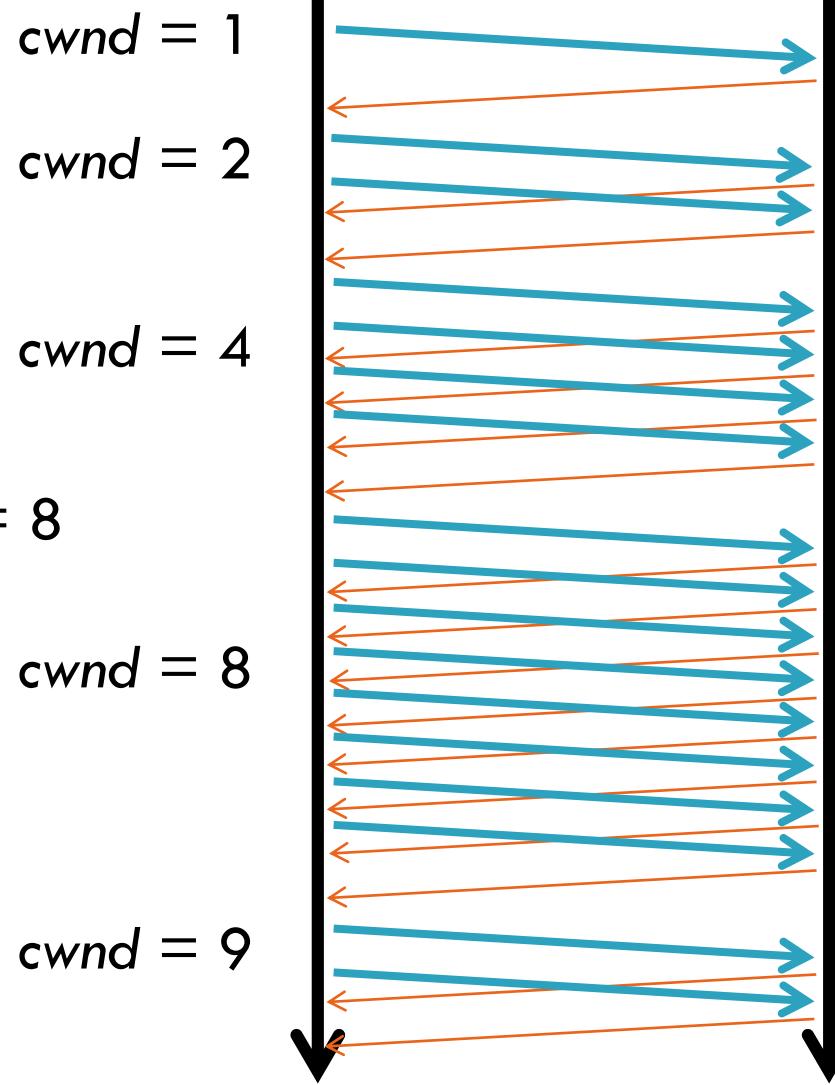
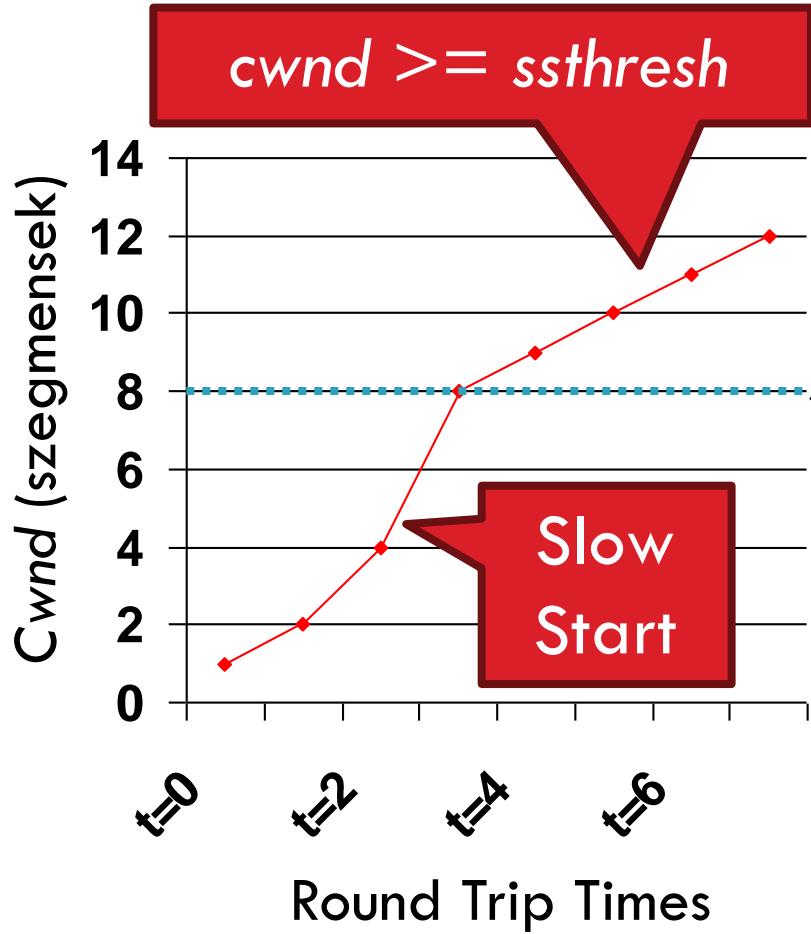
Torlódás elkerülés

57

- Additive Increase Multiplicative Decrease (AIMD) mód
- $ssthresh$ valójában egy alsóbecslés a könyök pontra
- **Ha $cwnd \geq ssthresh$ akkor**
 - Minden nyugtázott szegmens alkalmával növeljük a $cwnd$ értékét $(1/cwnd)$ -vel (azaz $cwnd += 1/cwnd$).
- Azaz a $cwnd$ eggyel nő, ha minden csomag nyugtázva lett.

Torlódás elkerülés példa

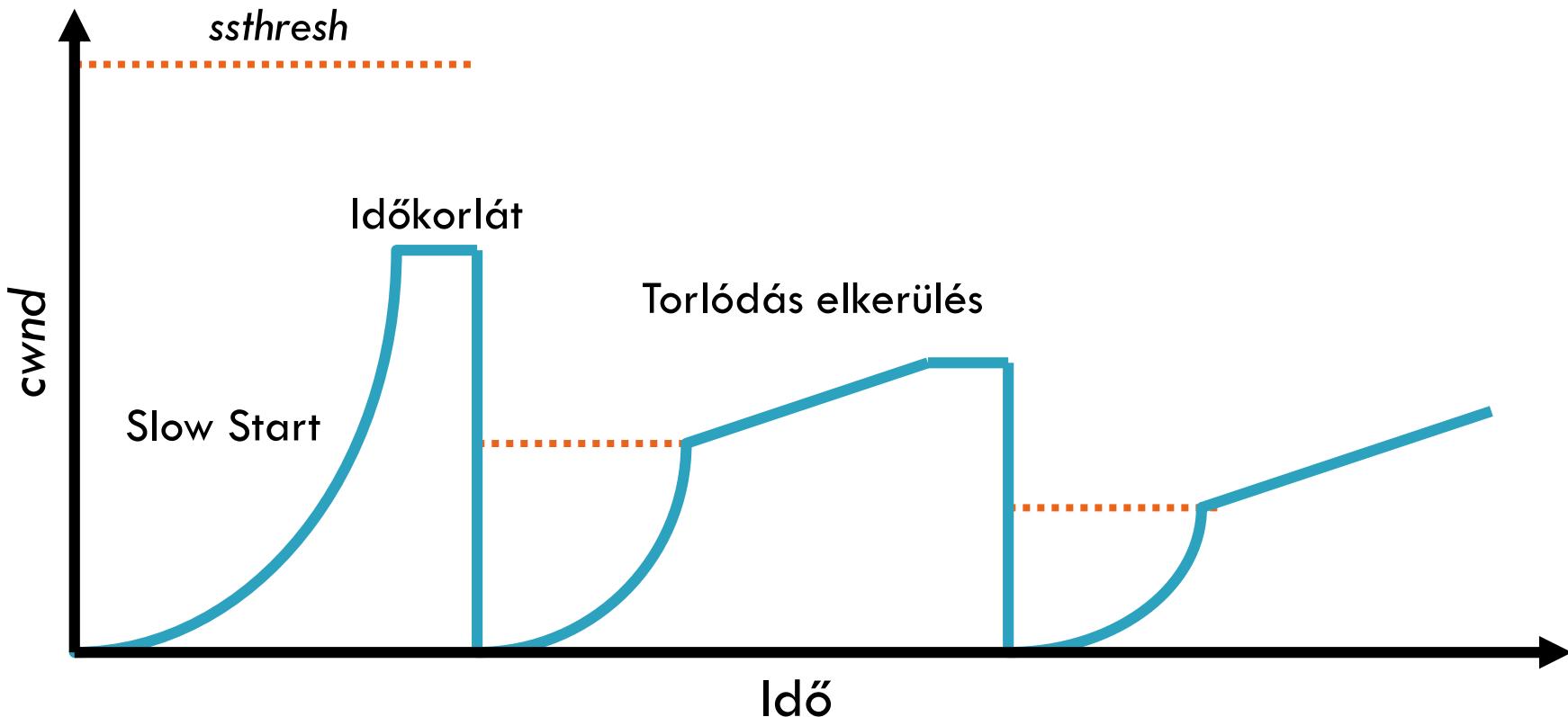
58



A teljes kép – TCP Tahoe

(az eredeti TCP)

59



Összefoglalás - TCP jellemzői

60

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájtfolyamokhoz.”

KAPCSOLATORIENTÁLT

- Két résztvevő, ahol egy résztvevőt egy IP-cím és egy port azonosít.
- A kapcsolat egyértelműen azonosított a résztvevő párral.
- Nincs se multi-, se broadcast üzenetküldés.
- A kapcsolatot fel kell építeni és le kell bontani.
- Egy kapcsolat a lezárásáig aktív.

Összefoglalás - TCP jellemzői

61

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájt-folyamokhoz.”

MEGBÍZHATÓSÁG

- minden csomag megérkezése nyugtázsra 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.

Összefoglalás - TCP jellemzői

62

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájt-folyamokhoz.”

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.

A TCP evolúciója

63

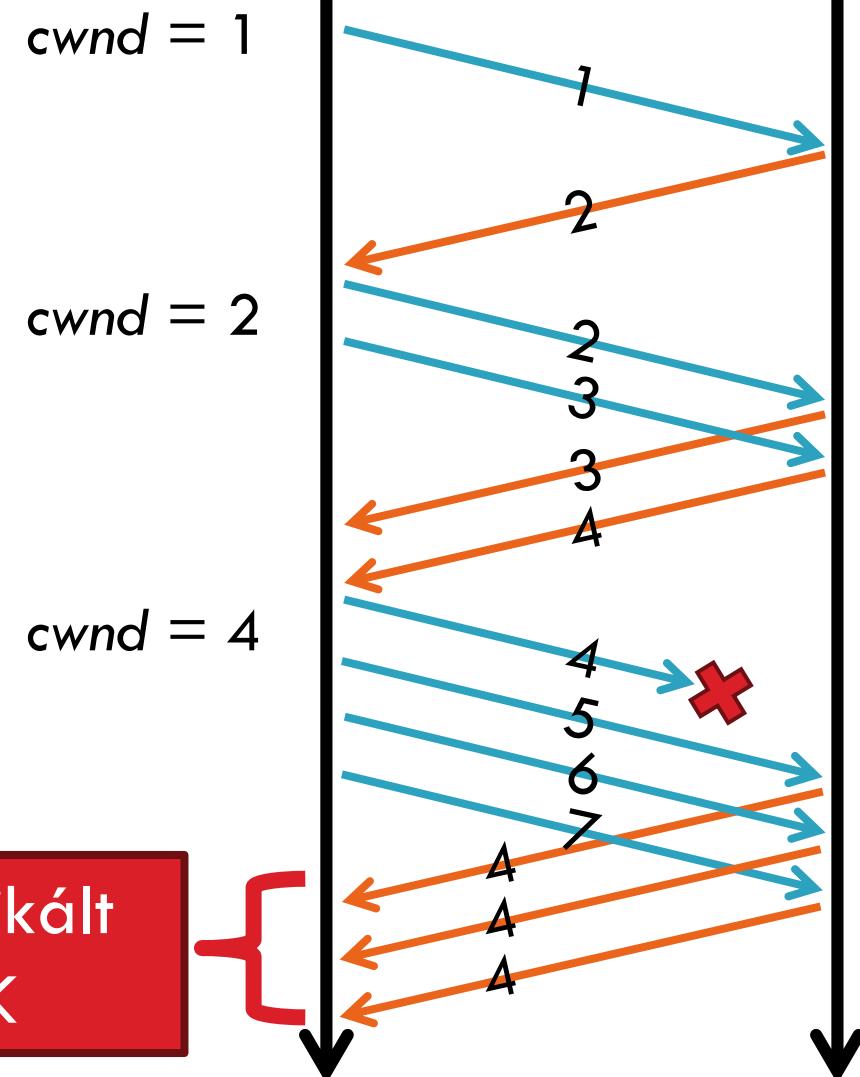
- Az eddigi megoldások a TCP Tahoe működéshez tartoztak
 - ▣ Eredeti TCP
- A TCP-t 1974-ben találták fel!
 - ▣ Napjainkba számos változata létezik
- Kezdeti népszerű változat: TCP Reno
 - ▣ Tahoe lehetőségei, plusz...
 - ▣ Gyors újraküldés (Fast retransmit)
 - 3 duplikált ACK? -> újraküldés (ne várunk az RTO-ra)
 - ▣ Gyors helyreállítás (Fast recovery)
 - Csomagvesztés esetén:
 - $\text{set cwnd} = \text{cwnd}/2$ ($\text{ssthresh} = \text{az új cwnd érték}$)

TCP Reno: Gyors újraküldés

64

- Probléma: Tahoe esetén ha egy csomag elveszik, akkor hosszú a várakozás az RTO-ig
- Reno: újraküldés 3 duplikált nyugta fogadása esetén
- Duplikált: ugyanaz a sorszám
 - Explicit jele a csomagvesztésnek

3 Duplikált
ACK



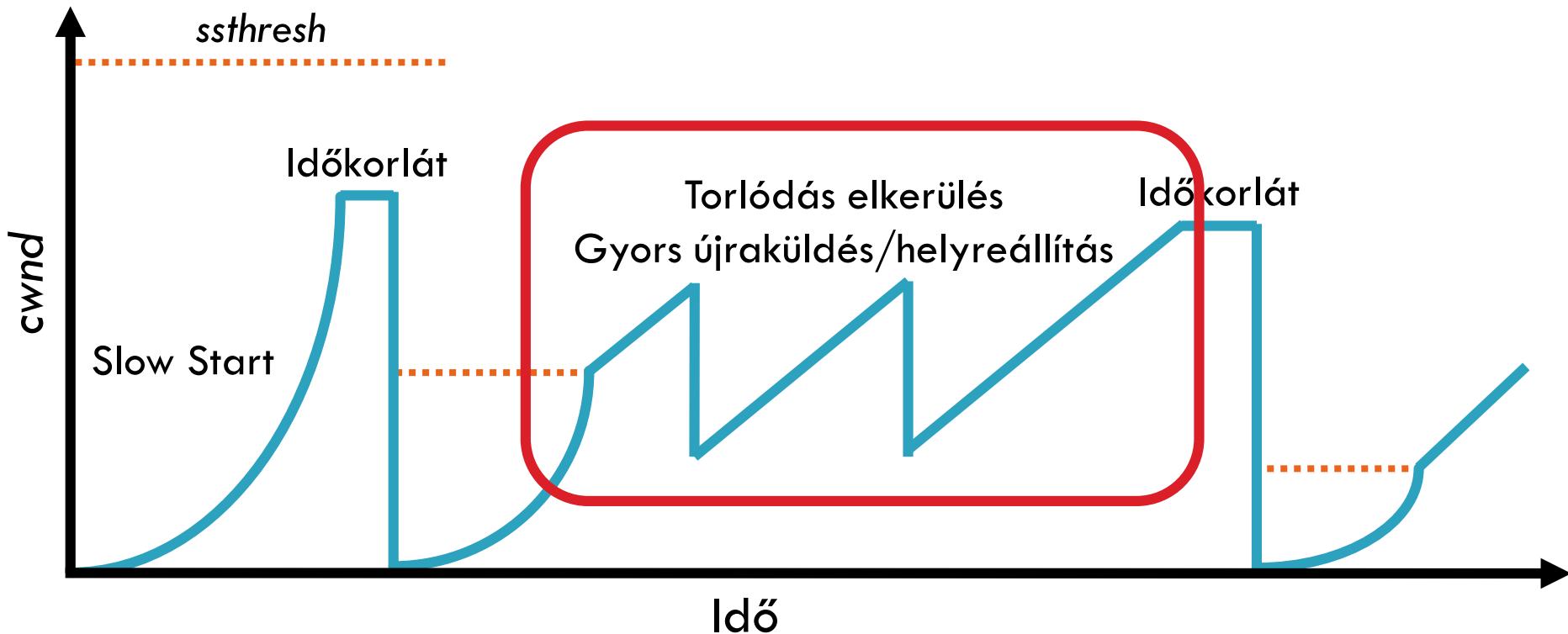
TCP Reno: Gyors helyreállítás

65

- Gyors újraküldés után módosítjuk a torlódási ablakot:
 - $cwnd := cwnd/2$ (valójában ez a *Multiplicative Decrease*)
 - $ssthresh :=$ az új $cwnd$
 - Azaz nem álltjuk vissza az eredeti 1-re a $cwnd$ -t!!!
 - Ezzel elkerüljük a felesleges slow start fázisokat!
 - Elkerüljük a költséges időkorlátokat
- Azonban ha az RTO lejár, továbbra is $cwnd = 1$
 - Visszatér a slow start fázishoz, hasonlóan a Tahoe-hoz
 - Olyan csomagokat jelez, melyeket egyáltalán nem szállítottunk le
 - A torlódás nagyon súlyos esetére figyelmeztet!!!

Példa: Gyors újraküldés/helyreállítás

66



- Stabil állapotban, a cwnd az optimális ablakméret körül oszcillál
- TCP minden csomagdobásokat kényszerít ki...

Számos TCP változat...

67

- Tahoe: az eredeti
 - ▣ Slow start és AIMD
 - ▣ Dinamikus RTO, RTT becsléssel
- Reno:
 - ▣ fast retransmit (3 dupACKs)
 - ▣ fast recovery ($cwnd = cwnd/2$ vesztés esetén)
- NewReno: javított gyors újraküldés
 - ▣ minden egyes duplikált ACK újraküldést vált ki
 - ▣ Probléma: >3 hibás sorrendben fogadott csomag is újraküldést okoz (hibásan!!!)...
- Vegas: késleltetés alapú torlódás elkerülés
- ...

TCP a valóságban

68

- Mi a legnépszerűbb variáns napjainkban?
 - Probléma: TCP rosszul teljesít nagy késleltetés-sávszélesség szorzattal rendelkező hálózatokban (a modern Internet ilyen)
 - Compound TCP (Windows)
 - Reno alapú
 - Két torlódási ablak: késleltetés alapú és vesztés alapú
 - Azaz egy összetett torlódás vezérlést alkalmaz
 - TCP CUBIC (Linux)
 - Fejlettebb BIC (Binary Increase Congestion Control) változat
 - Az ablakméretet egy harmadfokú egyenlet határozza meg
 - A legutolsó csomagvesztéstől eltelt T idővel paraméterezett

Nagy késleltetés-sávszélesség szorzat (Delay-bandwidth product)

69

- Probléma: A TCP nem teljesít jól ha
 - A hálózat kapacitása (sávszélessége) nagy
 - A késleltetés (RTT) nagy
 - Vagy ezek szorzata nagy
 - $b * d =$ maximális szállítás alatt levő adatmennyiségeg
 - Ezt nevezzük késleltetés-sávszélesség szorznak
- Miért teljesít ekkor gyengén a TCP?
 - A slow start és az additive increase csak lassan konvergál
 - A TCP ACK ütemezett (azaz csak minden ACK esetén történik esemény)
 - A nyugták beérkezési gyorsasága határozza meg, hogy milyen gyorsan tud reagálni
 - Nagy RTT → késleltetett nyugták → a TCP csak lassan reagál a megváltozott viszonyokra

Célok

70

- A TCP ablak gyorsabb növelése
 - A slow start és az additive increase túl lassú, ha nagy a sávszélesség
 - Sokkal gyorsabb konvergencia kell
- Fairség biztosítása más TCP változatokkal szemben
 - Az ablak növelése nem lehet túl agresszív
- Javított RTT fairség
 - A TCP Tahoe/Reno folyamok nem adnak fair erőforrás-megosztást nagyon eltérő RTT-k esetén
- Egyszerű implementáció

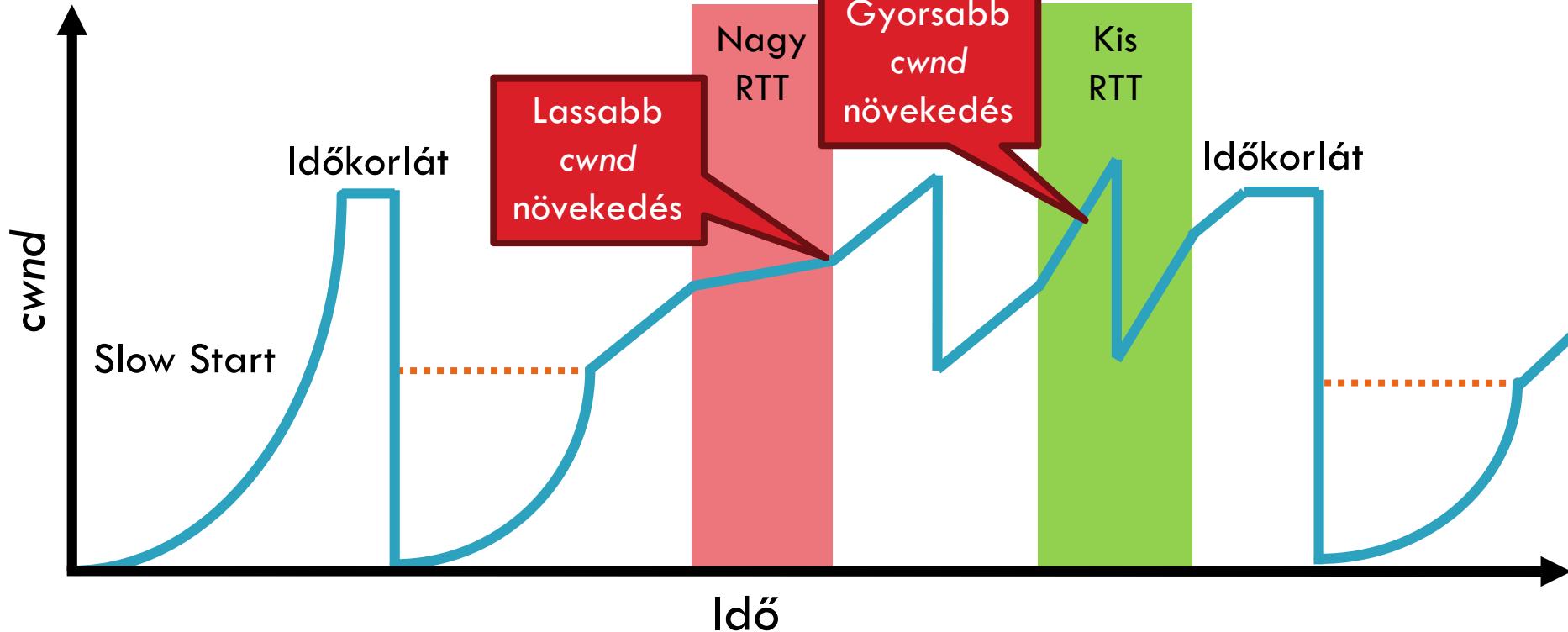
Compound TCP

71

- Alap TCP implementáció Windows rendszereken
- Ötlet: osszuk a *torlódási ablakot* két különálló ablakba
 - Hagyományos, vesztés alapú ablak
 - Új, késleltetés alapú ablak
- $wnd = \min(cwnd + dwnd, adv_wnd)$
 - cwnd-t az AIMD vezérli AIMD
 - *dwnd* a késleltetés alapú ablak
- A *dwnd* beállítása:
 - Ha nő az RTT, csökken a *dwnd* ($dwnd \geq 0$)
 - Ha csökken az RTT, nő a *dwnd*
 - A növekedés/csökkenés arányos a változás mértékével

Compound TCP példa

72



- Agresszívan reagál az RTT változására
- Előnyök: Gyors felfutás, sokkal fairebb viselkedés más folyamokkal szemben eltérő RTT esetén
- Hátrányok: folyamatos RTT becslés

TCP CUBIC

73

- Alap TCP implementáció Linux rendszereken
- Az AIMD helyettesítése egy „köbös” (CUBIC) függvényel

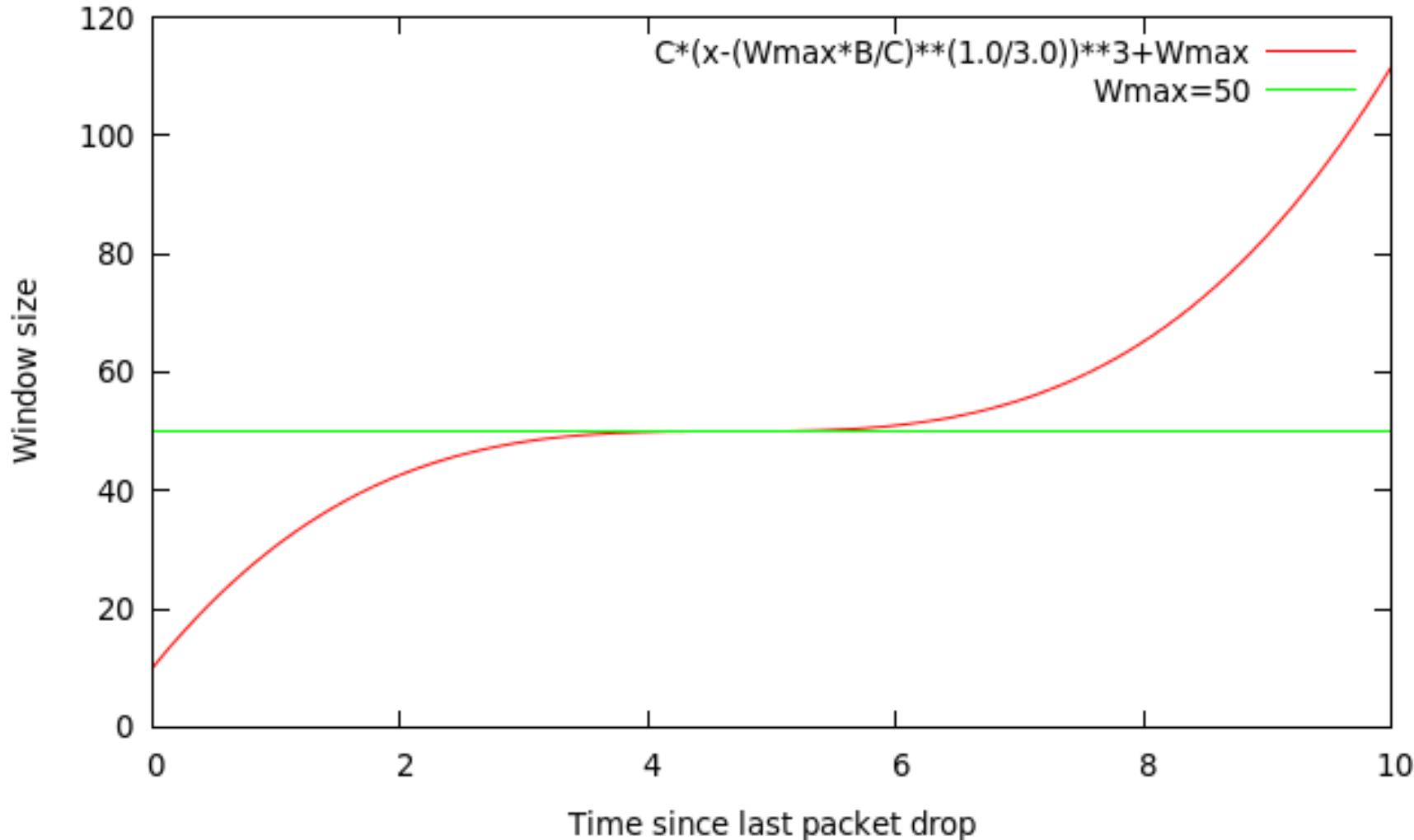
$$W_{cubic} = C(T - K)^3 + W_{max} \quad (1)$$

C is a scaling constant, and $K = \sqrt[3]{\frac{W_{max}\beta}{C}}$

- B → egy konstans a multiplicative increase fázishoz
- T → eltelt idő a legutóbbi csomagvesztés óta
- W_max → cwnd a legutolsó csomagvesztés idején

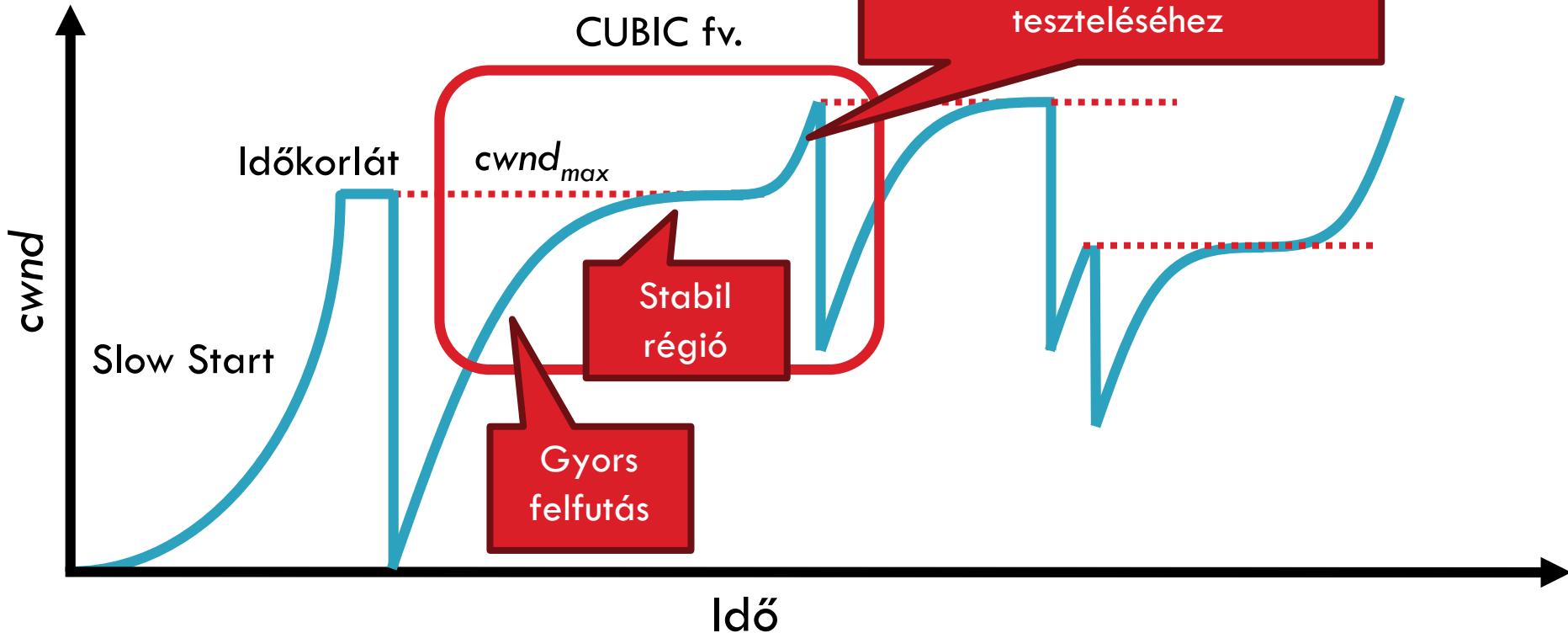
TCP CUBIC

74



TCP CUBIC példa

75



- Kevésbé pazarolja a sávszélességet a gyors felfutások miatt
- A stabil régió és a lassú gyorsítás segít a fairség biztosításában
 - A gyors felfutás sokkal agresszívabb, mint az additive increase
 - A Tahoe/Reno variánsokkal szembeni fairséghez a CUBIC-nak nem szabad ennyire agresszívnak lennie

Problémák a TCP-vel

76

- Az Internetes forgalom jelentős része TCP
- Azonban számos probléma okozója is egyben
 - Gyenge teljesítmény kis folyamok esetén
 - Gyenge teljesítmény wireless hálózatokban
 - DoS támadási felület

Kis folyamok (flows)

77

- Probléma: kis folyamok esetén torz viselkedés
 - 1 RTT szükséges a kapcsolat felépítésére (SYN, SYN/ACK)
 - pazarló
 - cwnd mindenkorban 1-gyel indul
 - Nincs lehetőség felgyorsulni a kevés adat miatt
- Az Internetes forgalom nagy része kis folyam
 - Többnyire HTTP átvitel, <100KB
 - A legtöbb TCP folyam el se hagyja a slow start fázist!!!
- Lehetséges megoldás (Google javaslat):
 - Kezdeti cwnd megnövelése 10-re
 - TCP Fast Open: kriptográfiai hashek használata a fogadó azonosítására, a három-utas kézfogás elhagyható helyette hash (cookie) küldése a syn csomagban

Wireless hálózatok

78

- Probléma: A Tahoe és Reno esetén
csomagvesztés = torlódás
 - WAN esetén ez helyes, ritka bit hibák
 - Azonban hamis vezeték nélküli hálózatokban, gyakori interferenciák
- TCP átvitel $\sim 1/\sqrt{\text{vesztési ráta}}$
 - Már néhány interferencia miatti csomagvesztés elég a teljesítmény drasztikus csökkenéséhez
- Lehetséges megoldások:
 - Réteg modell megsértése, adatkapcsolati információ a TCP-be
 - Késleltetés alapú torlódás vezérlés használata (pl. TCP Vegas)
 - Explicit torlódás jelzés - Explicit congestion notification (ECN)

Szolgáltatás megtagadása

Denial of Service (DoS)

79

- Probléma: a TCP kapcsolatok állapottal rendelkeznek
 - A SYN csomagok erőforrásokat foglalnak az szerveren
 - Az állapot legalább néhány percig fennmarad (RTO)
- SYN flood: elég sok SYN csomag küldése a szervernek ahhoz, hogy elfogyjon a memória és összeomoljon a kernel
- Megoldás: SYN cookie-k
 - Ötlet: ne tároljunk kezdeti állapotot a szerveren
 - Illesszük az állapotot a SYN/ACK csomagokba (a sorszám mezőbe (sequence number mező))
 - A kliensnek vissza kell tükrözni az állapotot...

Kitekintés

80

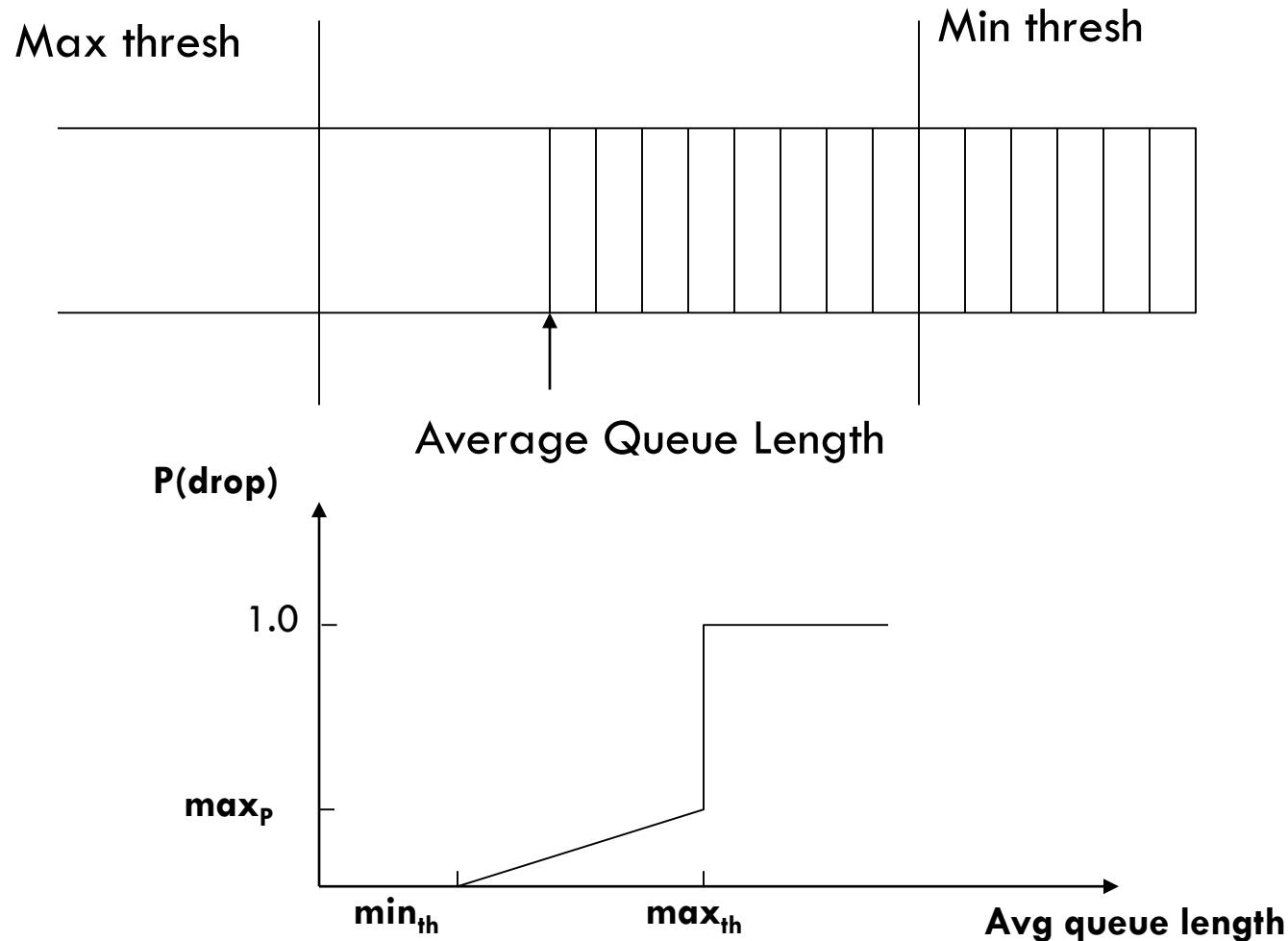
Typical Internet Queuing

- FIFO + drop-tail
 - ▣ Simplest choice
 - ▣ Used widely in the Internet
- FIFO (first-in-first-out)
 - ▣ Implies single class of traffic
- Drop-tail
 - ▣ Arriving packets get dropped when queue is full regardless of flow or importance
- Important distinction:
 - ▣ FIFO: scheduling discipline
 - ▣ Drop-tail: drop policy

RED Algorithm

- Maintain running average of queue length
- If $\text{avgq} < \text{min}_{\text{th}}$ do nothing
 - Low queuing, send packets through
- If $\text{avgq} > \text{max}_{\text{th}}$, drop packet
 - Protection from misbehaving sources
- Else mark packet in a manner proportional to queue length
 - Notify sources of incipient congestion
 - E.g. by ECN IP field or dropping packets with a given probability

RED Operation



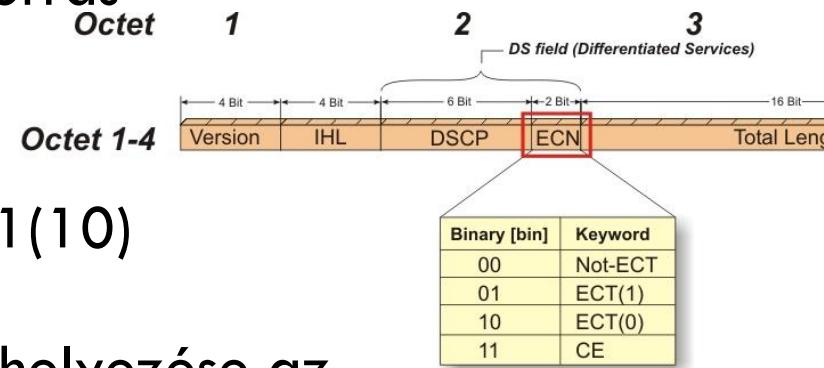
RED Algorithm

- Maintain running average of queue length
- For each packet arrival
 - Calculate average queue size (avg)
 - If $\text{min}_{\text{th}} \leq \text{avg} < \text{max}_{\text{th}}$
 - Calculate probability P_a
 - With probability P_a
 - Mark the arriving packet: drop or set-up ECN
 - Else if $\text{max}_{\text{th}} \leq \text{avg}$
 - Mark the arriving packet: drop, ECN

Csomag dobás vagy ECN jelölés

85

- Csomag dobás
 - Újraküldés szükséges
 - Egyszerűbb megvalósítás
 - Timeout lejárta után tud reagálni a forrás
- ECN jelölés
 - Végpont támogatás szükséges
 - Az IP csomag ECT-0 (01) vagy ECT-1(10) jelöléssel
 - Dobás helyett -> ECN CE (11) jel elhelyezése az IP fejlécben
 - A fogadó érzékeli a CE jelet, majd a visszamenő TCP nyugtába bebillent egy ECE flaget, mely jelzi a forrásnak a torlódást
 - Hagyományos TCP (CUBIC, RENO, stb.) források az ECE flaget csomagvesztésnek értelmezik, de újraküldés nem szükséges.



Data Center TCP: DCTCP

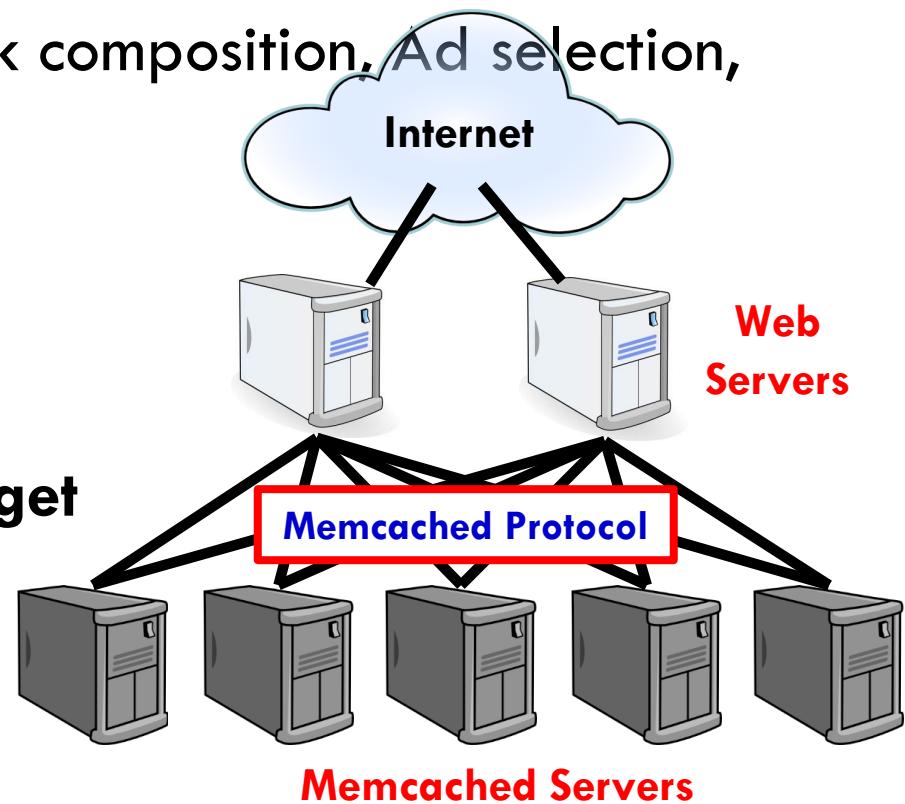
Generality of Partition/Aggregate

- The foundation for many large-scale web applications.
 - Web search, Social network composition, Ad selection, etc.

□ Example: Facebook

Partition/Aggregate ~ Multiget

- Aggregators: **Web Servers**
- Workers: **Memcached Servers**



Workloads

88

- Partition/Aggregate
(Query)



Delay-sensitive



- Short messages [50KB-1MB]
(Coordination, Control state)



Delay-sensitive



- Large flows [1MB-50MB]
(Data update)



Throughput-sensitive



Impairments

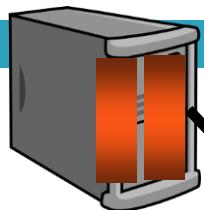
89

- Incast
- Queue Buildup
- Buffer Pressure

Incast

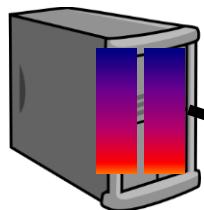
90

Worker 1

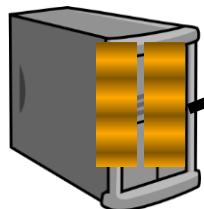


- Synchronized mice collide.
➤ Caused by Partition/Aggregate.

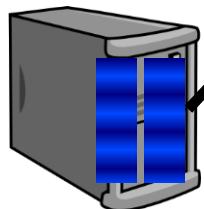
Worker 2



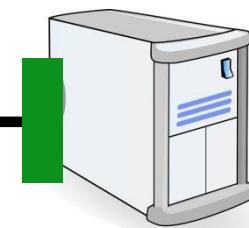
Worker 3



Worker 4



Aggregator

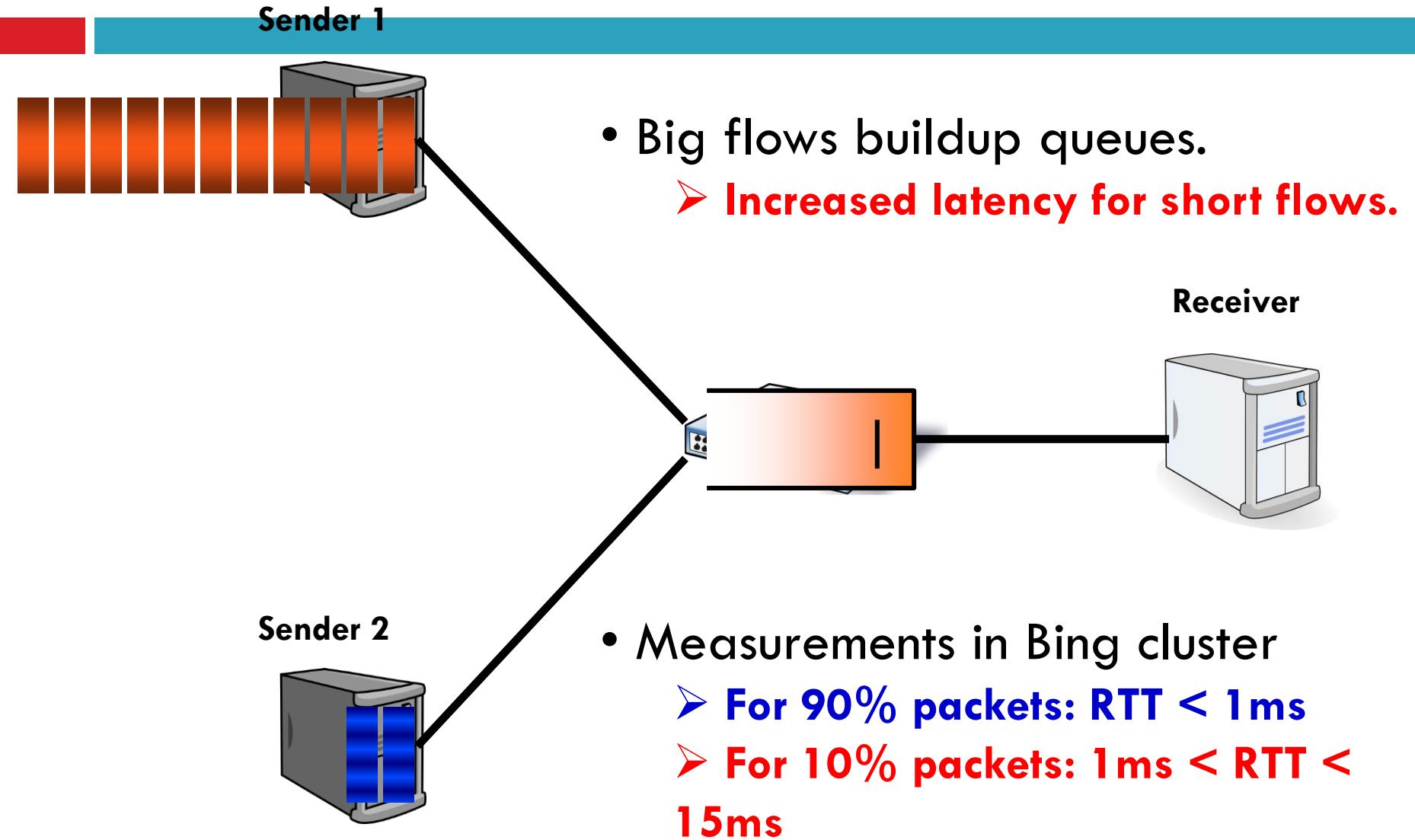


$RTO_{min} = 300 \text{ ms}$

TCP timeout



Queue Buildup



Data Center Transport Requirements

92

1. High Burst Tolerance

- Incast due to Partition/Aggregate is common.

2. Low Latency

- Short flows, queries

3. High Throughput

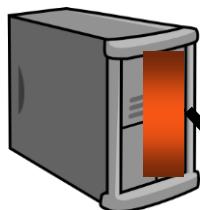
- Continuous data updates, large file transfers

The challenge is to achieve these three together.

DCTCP: The TCP/ECN Control Loop

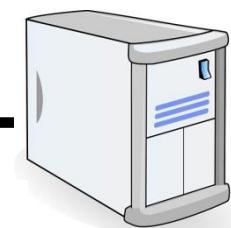
Sender 1

ECN = Explicit Congestion Notification

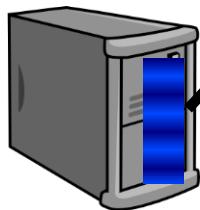


ECN Mark (1 bit)

Receiver



Sender 2



DCTCP: Two Key Ideas

18

1. React in proportion to the **extent** of congestion, not its **presence**.
 - ✓ Reduces **variance** in sending rates, lowering queuing requirements.

ECN Marks	TCP	DCTCP
1 0 1 1 1 0 1 1 1	Cut window by 50%	Cut window by 40%
0 0 0 0 0 0 0 0 1	Cut window by 50%	Cut window by 5%

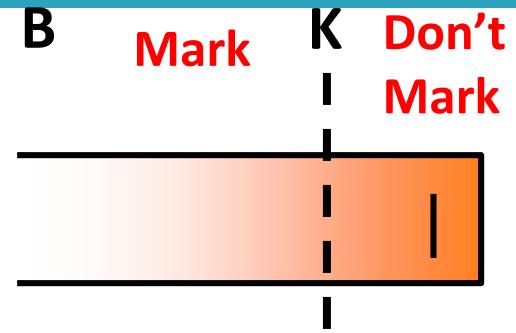
2. Mark based on **instantaneous** queue length.
 - ✓ Fast feedback to better deal with bursts.

Data Center TCP Algorithm

19

Switch side:

- Mark packets when **Queue Length > K**.



Sender side:

- Maintain running average of **fraction** of packets marked (α).

In each RTT:

$$F = \frac{\# \text{ of marked ACKs}}{\text{Total } \# \text{ of ACKs}}$$

$$\alpha \leftarrow (1 - g)\alpha + gF$$

- **Adaptive window decreases:** $Cwnd \leftarrow (1 - \frac{\alpha}{2})Cwnd$
- Note: decrease factor between 1 and 2.

Köszönöm a figyelmet!

Számítógépes Hálózatok

10. Előadás: Szállítói réteg++

Mi az a torlódás?

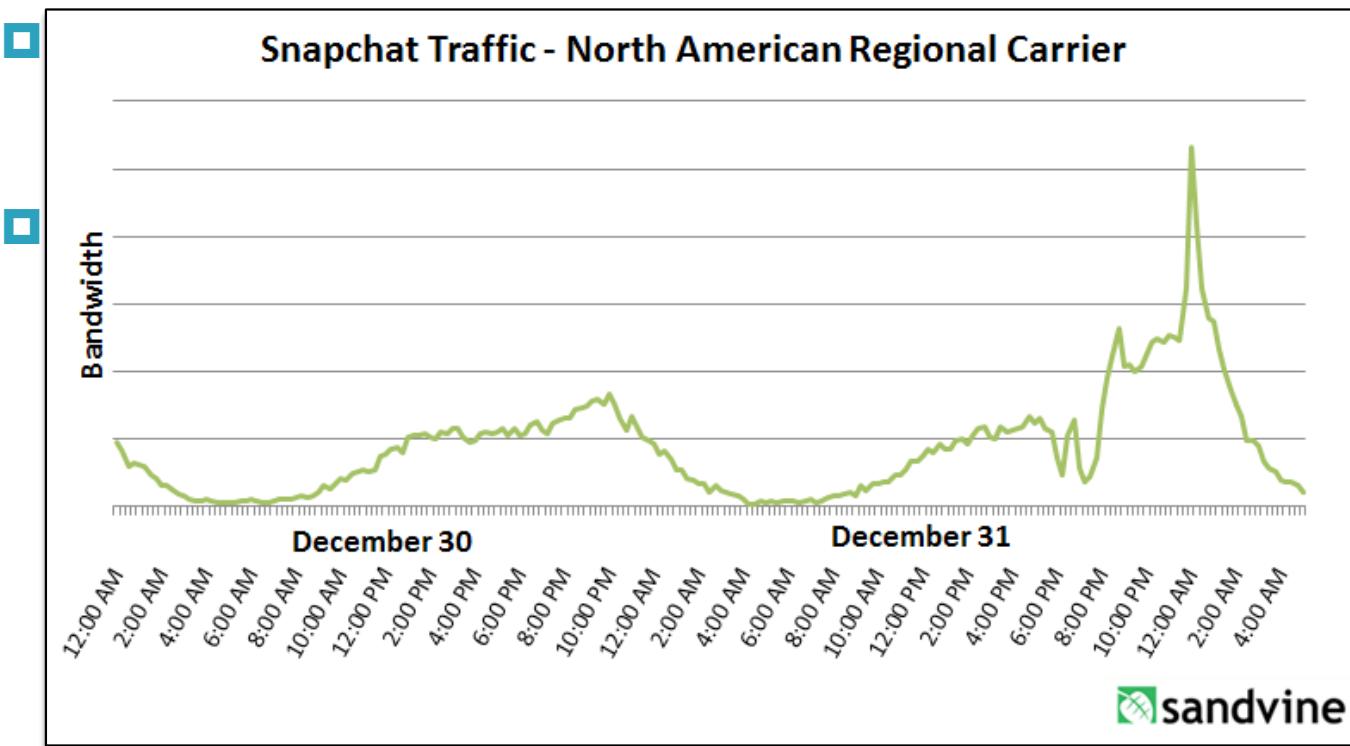
2

- A hálózat terhelése nagyobb, mint a kapacitása
 - ▣ A kapacitás nem egyenletes a hálózatban
 - Modem vs. Cellular vs. Cable vs. Fiber Optics
 - ▣ Számos folyam verseng a sávszélességért
 - otthoni kábel modem vs. corporate datacenter
 - ▣ A terhelés időben nem egyenletes
 - Vasárnap este 10:00 = BitTorrent Game of Thrones

Mi az a torlódás?

3

- A hálózat terhelése nagyobb, mint a kapacitása
 - A kapacitás nem egyenletes a hálózatban
 - Modem vs. Cellular vs. Cable vs. Fiber Optics



Miért rossz a torlódás?

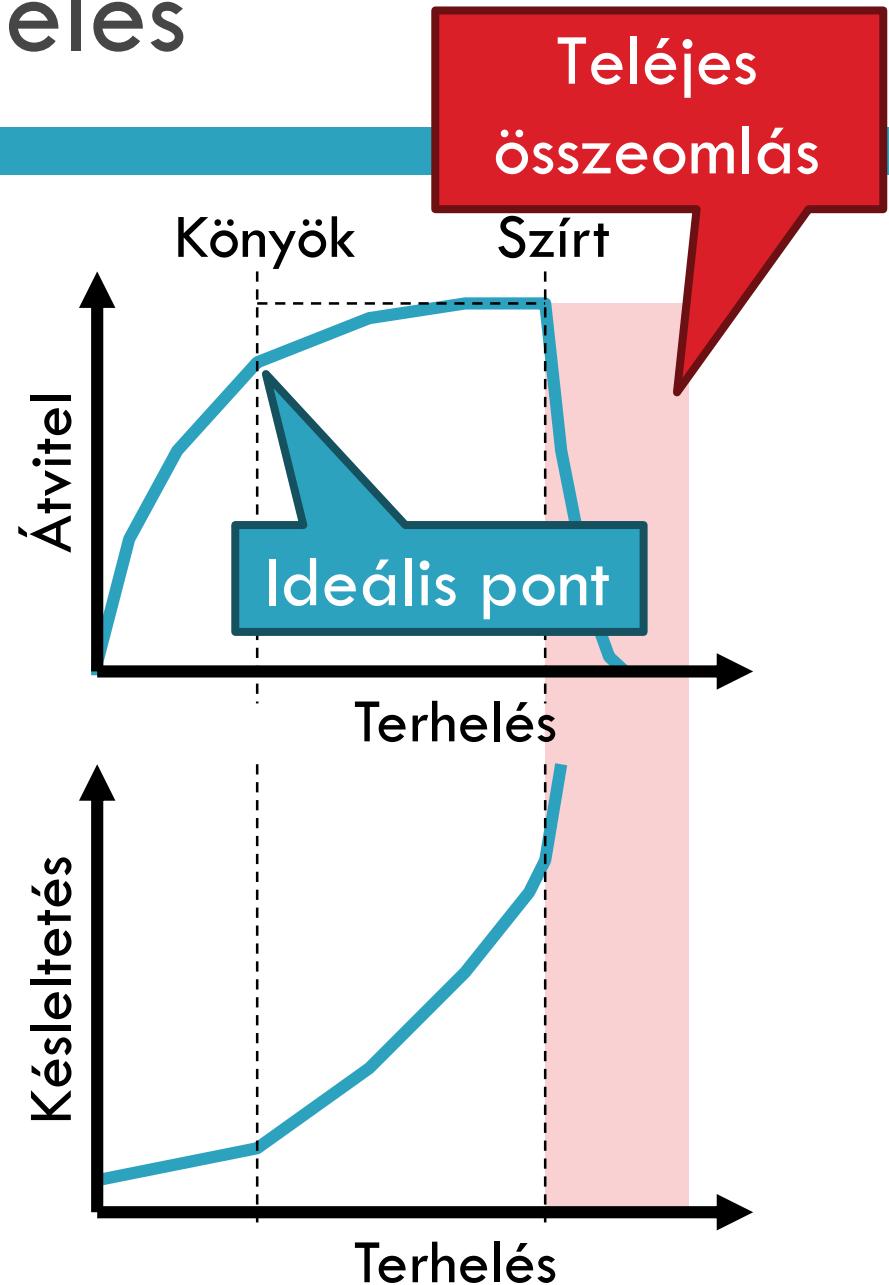
4

- Csomagvesztést eredményez
 - A routerek véges memóriával (puffer) rendelkeznek
 - Önhasonló Internet forgalom, nincs puffer, amiben ne okozna csomagvesztést
 - Ahogy a routerek pufferre elkezd telítődni, csomagokat kezd eldobni... (RED)
- Gyakorlati következmények
 - A routerek sorai telítődnek, megnövekedett késleltetés
 - Sávszélesség pazarlása az újraküldések miatt
 - Alacsony hálózati átvitel (goodput)

Megnövekedett terhelés

5

- Könyök („knee”)- a pont, ami után
 - Az átvitel szinte alig nő
 - Késleltetés viszont gyorsan emelkedik
- Egy egyszerű sorban ($M/M/1$)
 - Késleltetés = $1/(1 - \text{utilization})$
- Szírt („cliff”)- a pont, ami után
 - Átvitel lényegében leesik 0-ra
 - A késleltetés pedig $\rightarrow \infty$

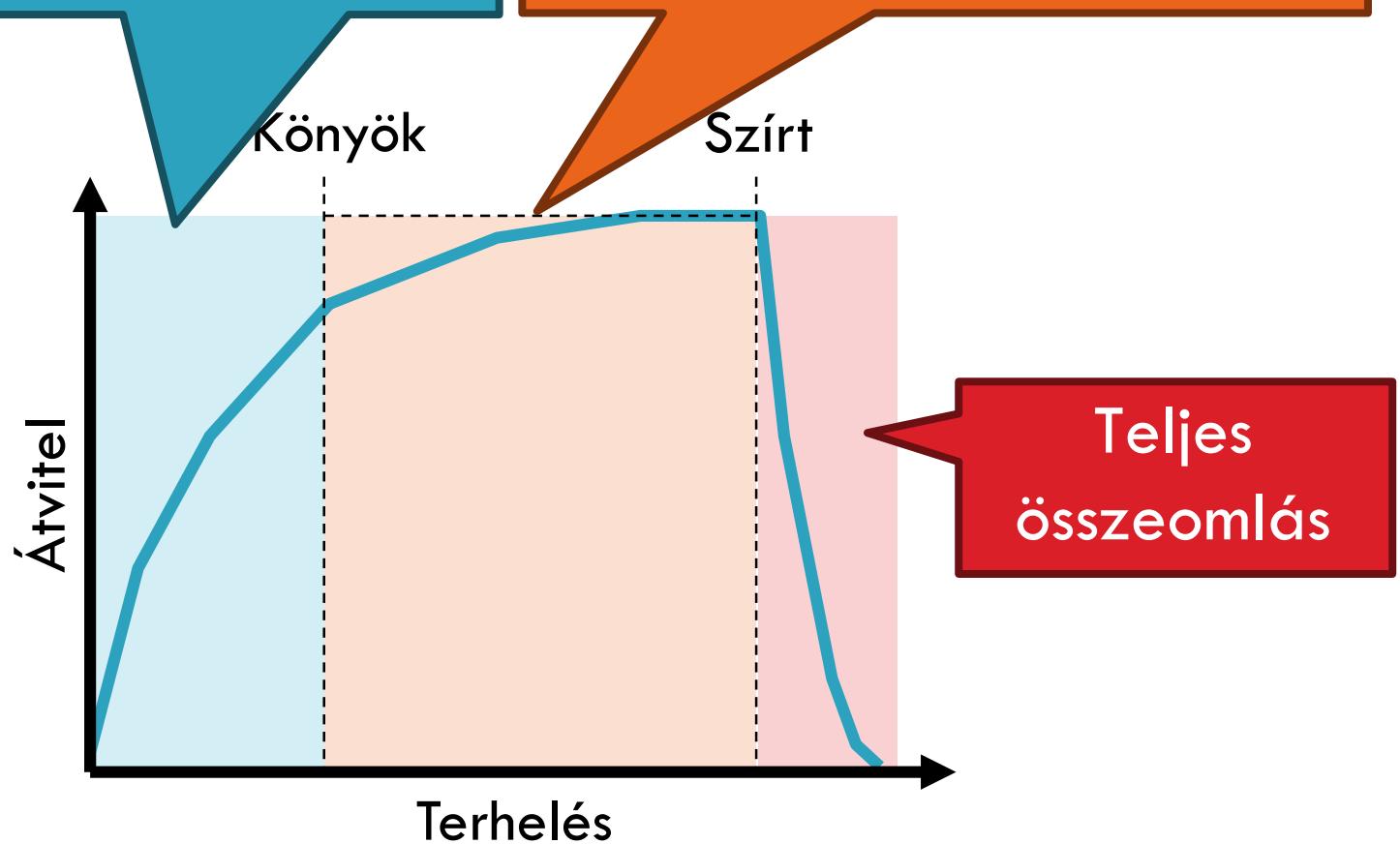


Torlódás vezérlés vs torlódás elkerülés

6

Torlódás elkerülés:
Maradj a könyök bal oldalán

Torlódás vezérlés
Maradj a szírt bal oldalán



Advertised Window

Meghirdetett ablak, újragondolva

7

- Megoldja-e a torlódás problémáját a TCP esetén a meghirdetett ablak használata?

NEM

- Ez az ablak csak a fogadót védi a túlterheléstől
- Egy kellően gyors fogadó kimaxolhatja ezt az ablakot
 - Mi van, ha a hálózat lassabb, mint a fogadó?
 - Mi van, ha vannak konkurens folyamok is?
- Következmények
 - Az ablak méret határozza meg a küldési rátát
 - Az ablaknak állíthatónak kell lennie, hogy elkerüljük a torlódás miatti teljes összeomlást...

Általános megoldások

8

- Ne csinálunk semmit, küldjük a csomagokat megkülönböztetés nélkül
 - Nagy csomagvesztés, jósolhatatlan teljesítmény
 - Teljes összeomláshoz vezethet
- Erőforrás foglalás
 - Folyamokhoz előre sávszélességet allokálunk
 - Csomagküldés előtt egy tárgyalási szakaszra is szükség van
 - Hálózati támogatás kell hozzá
- Dinamikus beállítás
 - Próbák használata a torlódási szint megbecséléshöz
 - Gyorsítás, ha torlódási szint alacsony
 - Lassítás, amint nő a torlódás
 - Nem rendezett dinamika, elosztott koordináció

TCP Torlódásvezérlés

9

- minden TCP kapcsolat rendelkezik egy ablakkal
 - A nem-nyugtázott csomagok számát vezérli
- Küldési ráta \sim window/RTT
- Ötlet: ablak méretének változtatása a küldési ráta vezérléséhez
- Vezessünk be egy **torlódási ablakot (congestion window)** a küldő oldalon
 - Torlódás vezérlés egy küldő oldali probléma
 - Jelölése: cwnd

Két fő komponens

10

1. Torlódás detektálás

- Eldobott csomag egy megbízható jel
 - Késleltetés alapú megoldások – nehéz és kockázatos
- Hogyan detektáljuk a csomag eldobását? Nyugtával
 - Időkorlát lejár ACK fogadása nélkül
 - Számos duplikált ACK jön be sorban (később lesz róla szó)

2. Ráta beállító algoritmus

- *cwnd* módosítása
- Sávszélesség próba
- Válasz lépés a torlódásra

Ráta vezérlés

11

- Tudjuk, hogy a TCP ACK ütemezett
 - Torlódás = késleltetés = hosszú várakozás a nyugták között
 - Nincs torlódás = alacsony késleltetés = gyors ACK
- Alapvető algoritmus
 - ACK fogadása esetén: növeljük a cwnd ablakot
 - Adat leszállítva, valószínűleg gyorsabban is küldhetünk
 - cwnd növekedése arányos az RTT-vel
 - Csomagvesztés esetén: csökkentsük a cwnd ablakot
 - Adat elveszett, torlódásnak kell lennie a hálózatban
- Kérdés: milyen függvényt használjuk a növeléshez és csökkentéshez? !!!!

Torlódás vezérlés megvalósítása

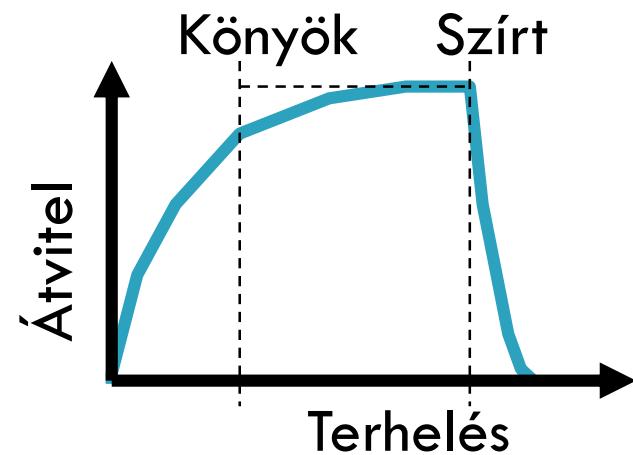
12

- Három változót kell nyilvántartani:
 - cwnd: torlódási ablak
 - adv_wnd: a fogadó meghirdetett ablaka
 - ssthresh: vágási érték (a cwnd frissítésére használjuk)
- Küldésnél használjuk: $wnd = \min(cwnd, adv_wnd)$
- A torlódás vezérlés két fázisa:
 1. Lassú indulás („Slow start”) ($cwnd < ssthresh$)
 - Az ún. bottleneck (legszűkebb) sávszélesség meghatározása a cél.
 2. Torlódás elkerülés ($cwnd \geq ssthresh$)
 - AIMD – Additive Increase Multiplicative Decrease

Lassú indulás - Slow Start

13

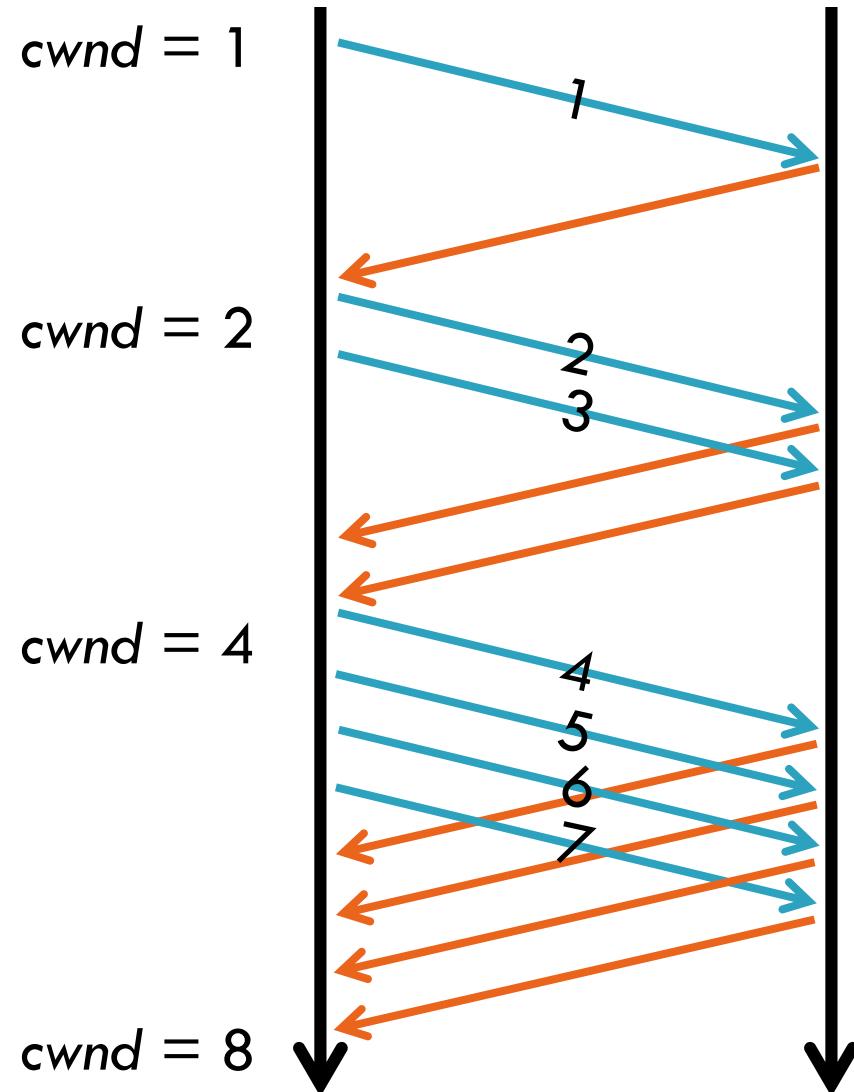
- Cél, hogy gyorsan elérjük a könyök pontot
- Egy kapcsolat kezdetén (vagy újraindításakor)
 - $cwnd = 1$
 - $ssthresh = adv_wnd$
 - minden nyugtázott szegmensre: $cwnd++$
- Egészen addig amíg
 - El nem érjük az $ssthresh$ értéket
 - Vagy csomagvesztés nem történik
- A Slow Start valójában nem lassú
 - $cwnd$ exponenciálisan nő



Slow Start példa

14

- $cwnd$ gyorsan nő
- Lelassul, amikor...
 - $cwnd \geq ssthresh$
 - Vagy csomagvesztés történik



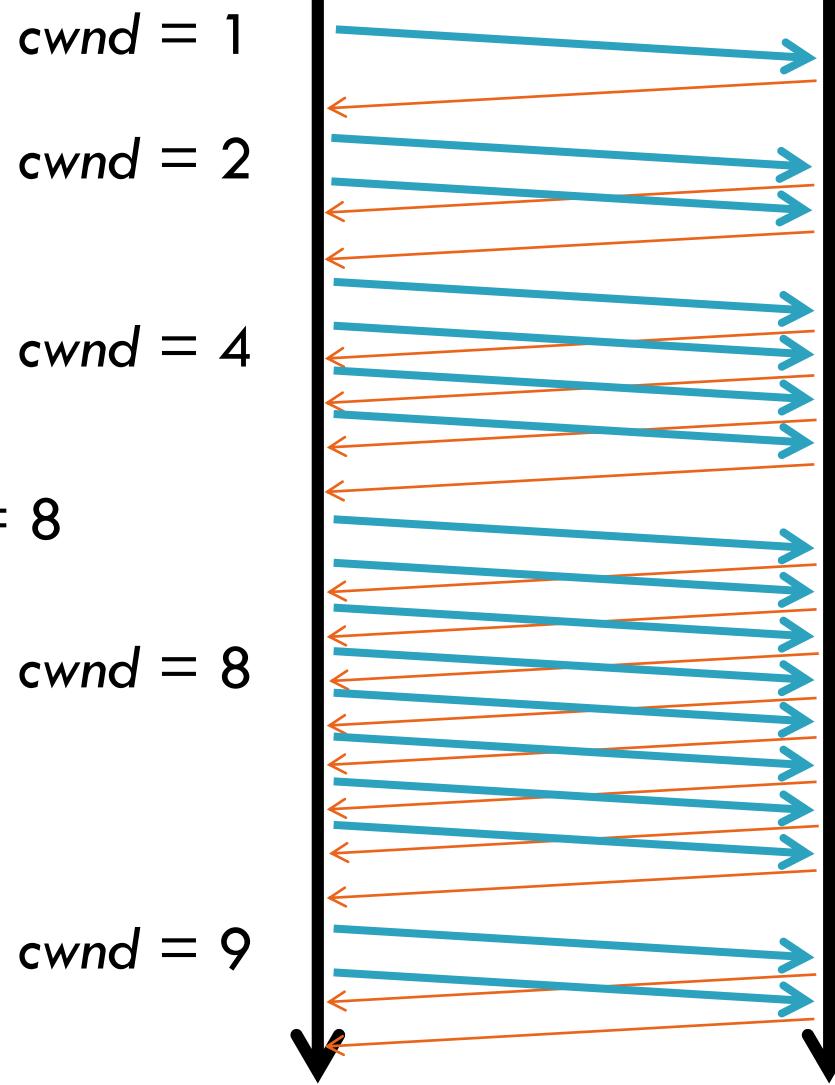
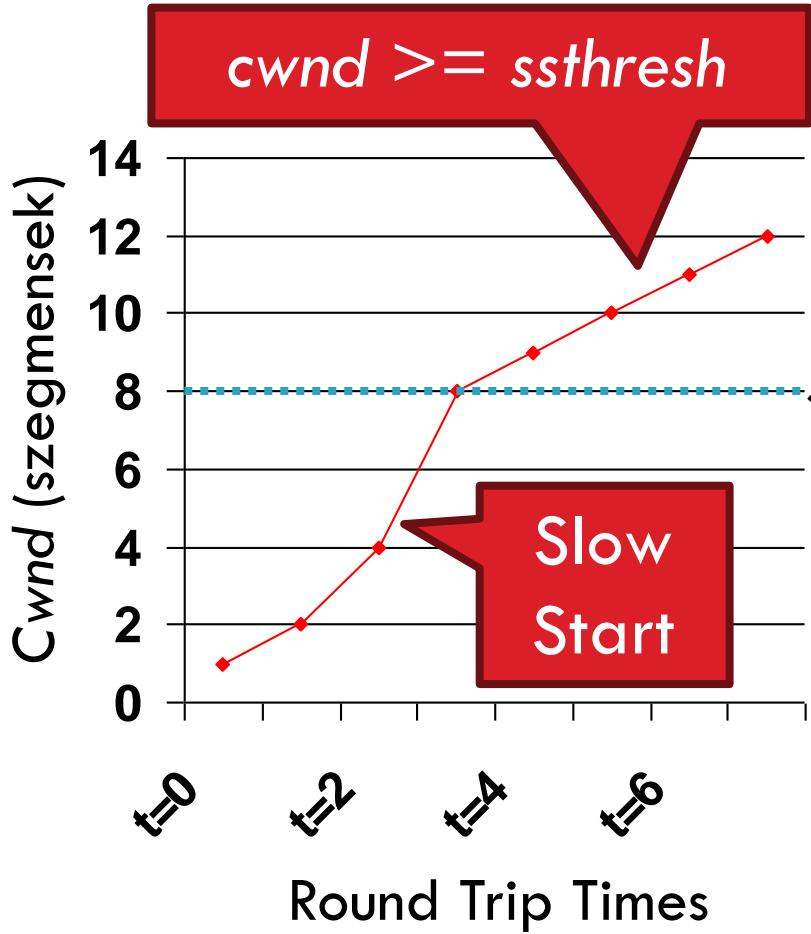
Torlódás elkerülés

15

- Additive Increase Multiplicative Decrease (AIMD) mód
- $ssthresh$ valójában egy alsóbecslés a könyök pontra
- **Ha $cwnd \geq ssthresh$ akkor**
 - Minden nyugtázott szegmens alkalmával növeljük a $cwnd$ értékét $(1/cwnd)$ -vel (azaz $cwnd += 1/cwnd$).
- Azaz a $cwnd$ eggyel nő, ha minden csomag nyugtázva lett.

Torlódás elkerülés példa

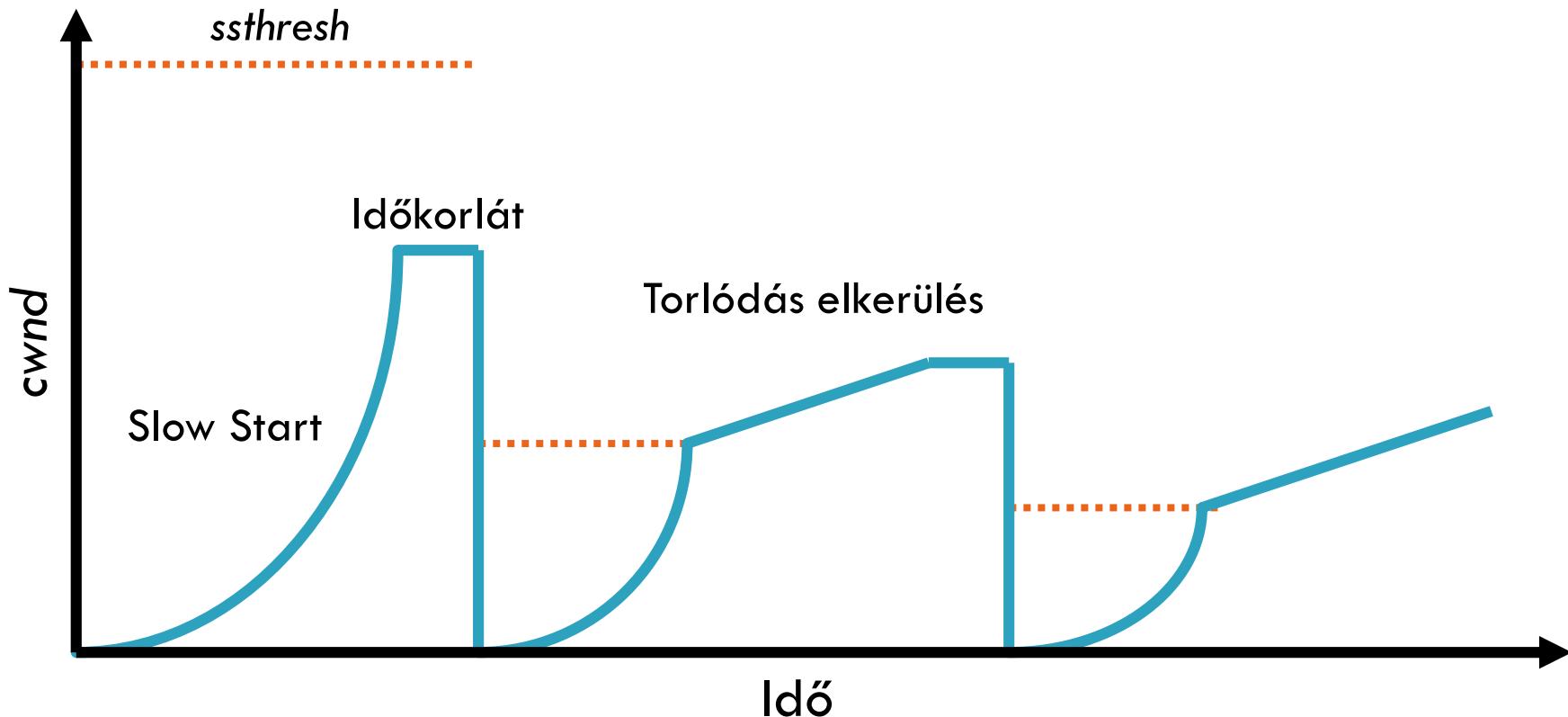
16



A teljes kép – TCP Tahoe

(az eredeti TCP)

17



Összefoglalás - TCP jellemzői

18

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájtfolyamokhoz.”

KAPCSOLATORIENTÁLT

- Két résztvevő, ahol egy résztvevőt egy IP-cím és egy port azonosít.
- A kapcsolat egyértelműen azonosított a résztvevő párral.
- Nincs se multi-, se broadcast üzenetküldés.
- A kapcsolatot fel kell építeni és le kell bontani.
- Egy kapcsolat a lezárásáig aktív.

Összefoglalás - TCP jellemzői

19

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájt-folyamokhoz.”

MEGBÍZHATÓSÁG

- minden csomag megérkezése nyugtázsra 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.

Összefoglalás - TCP jellemzői

20

„A TCP egy kapcsolatorientált megbízható szolgáltatás kétirányú bájt-folyamokhoz.”

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.

A TCP evolúciója

21

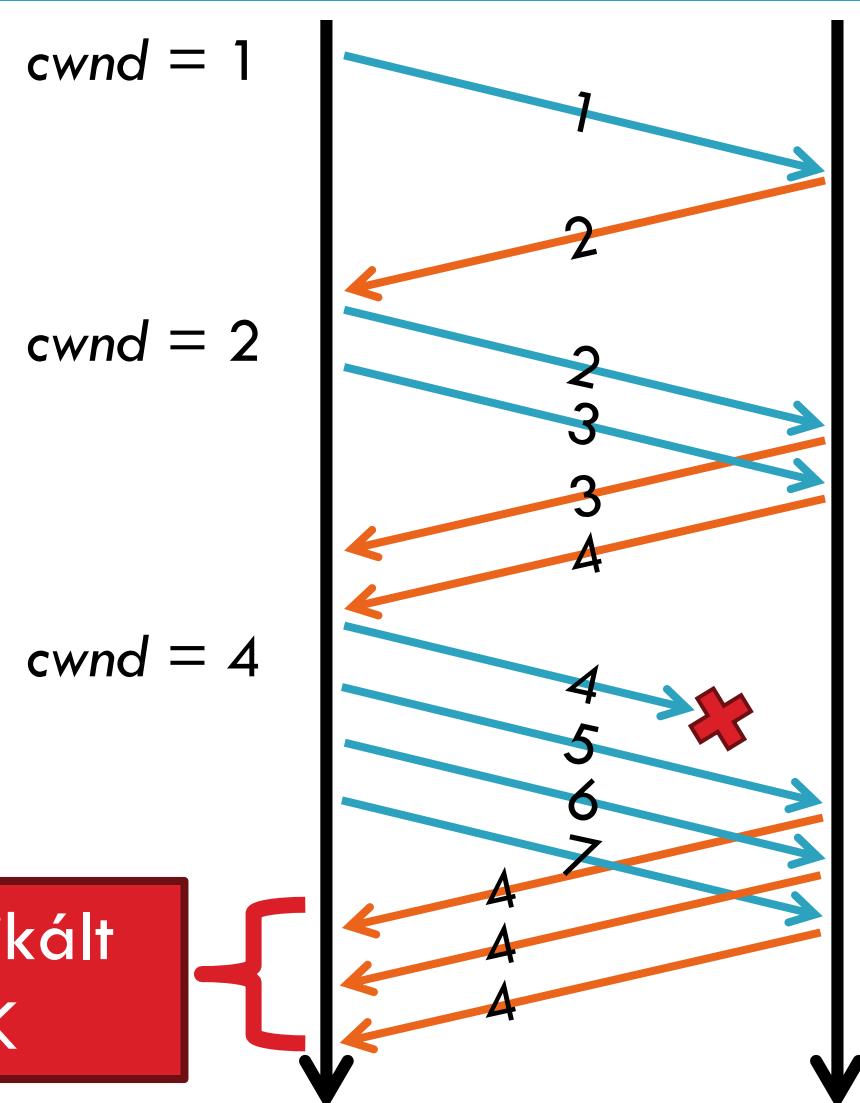
- Az eddigi megoldások a TCP Tahoe működéshez tartoztak
 - ▣ Eredeti TCP
- A TCP-t 1974-ben találták fel!
 - ▣ Napjainkba számos változata létezik
- Kezdeti népszerű változat: TCP Reno
 - ▣ Tahoe lehetőségei, plusz...
 - ▣ Gyors újraküldés (Fast retransmit)
 - 3 duplikált ACK? -> újraküldés (ne várunk az RTO-ra)
 - ▣ Gyors helyreállítás (Fast recovery)
 - Csomagvesztés esetén:
 - $\text{set cwnd} = \text{cwnd}/2$ ($\text{ssthresh} = \text{az új cwnd érték}$)

TCP Reno: Gyors újraküldés

22

- Probléma: Tahoe esetén ha egy csomag elveszik, akkor hosszú a várakozás az RTO-ig
- Reno: újraküldés 3 duplikált nyugta fogadása esetén
- Duplikált: ugyanaz a sorszám
 - Explicit jele a csomagvesztésnek

3 Duplikált
ACK



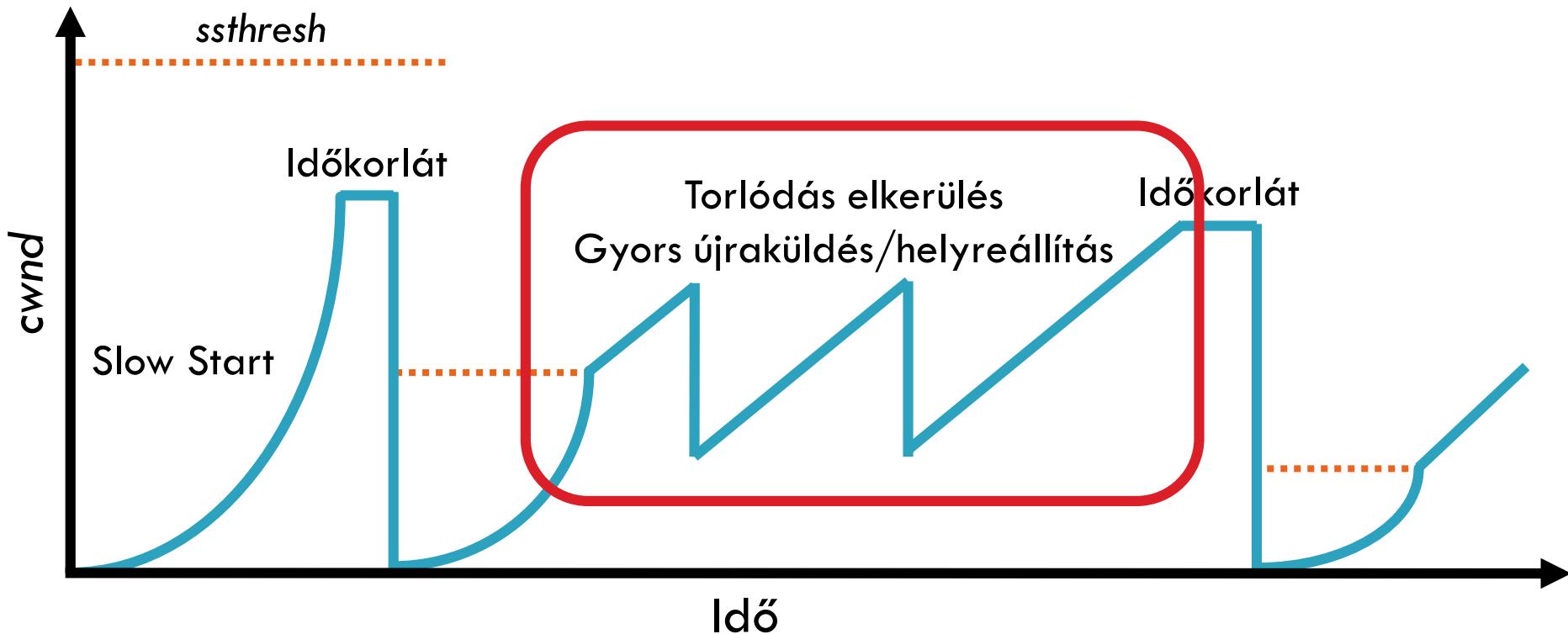
TCP Reno: Gyors helyreállítás

23

- Gyors újraküldés után módosítjuk a torlódási ablakot:
 - $cwnd := cwnd/2$ (valójában ez a *Multiplicative Decrease*)
 - $ssthresh :=$ az új $cwnd$
 - Azaz nem álltjuk vissza az eredeti 1-re a $cwnd$ -t!!!
 - Ezzel elkerüljük a felesleges slow start fázisokat!
 - Elkerüljük a költséges időkorlátokat
- Azonban ha az RTO lejár, továbbra is $cwnd = 1$
 - Visszatér a slow start fázishoz, hasonlóan a Tahoe-hoz
 - Olyan csomagokat jelez, melyeket egyáltalán nem szállítottunk le
 - A torlódás nagyon súlyos esetére figyelmeztet!!!

Példa: Gyors újraküldés/helyreállítás

24



- Stabil állapotban, a cwnd az optimális ablakméret körül oszcillál
- TCP minden csomagdobásokat kényszerít ki...

Számos TCP változat...

25

- Tahoe: az eredeti
 - ▣ Slow start és AIMD
 - ▣ Dinamikus RTO, RTT becsléssel
- Reno:
 - ▣ fast retransmit (3 dupACKs)
 - ▣ fast recovery ($cwnd = cwnd/2$ vesztés esetén)
- NewReno: javított gyors újraküldés
 - ▣ minden egyes duplikált ACK újraküldést vált ki
 - ▣ Probléma: >3 hibás sorrendben fogadott csomag is újraküldést okoz (hibásan!!!)...
- Vegas: késleltetés alapú torlódás elkerülés
- ...

TCP a valóságban

26

- Mi a legnépszerűbb variáns napjainkban?
 - Probléma: TCP rosszul teljesít nagy késleltetés-sávszélesség szorzattal rendelkező hálózatokban (a modern Internet ilyen)
 - Compound TCP (Windows)
 - Reno alapú
 - Két torlódási ablak: késleltetés alapú és vesztés alapú
 - Azaz egy összetett torlódás vezérlést alkalmaz
 - TCP CUBIC (Linux)
 - Fejlettebb BIC (Binary Increase Congestion Control) változat
 - Az ablakméretet egy harmadfokú egyenlet határozza meg
 - A legutolsó csomagvesztéstől eltelt T idővel paraméterezett

Nagy késleltetés-sávszélesség szorzat (Delay-bandwidth product)

27

- Probléma: A TCP nem teljesít jól ha
 - A hálózat kapacitása (sávszélessége) nagy
 - A késleltetés (RTT) nagy
 - Vagy ezek szorzata nagy
 - $b * d =$ maximális szállítás alatt levő adatmennyiségeg
 - Ezt nevezzük késleltetés-sávszélesség szorznak
- Miért teljesít ekkor gyengén a TCP?
 - A slow start és az additive increase csak lassan konvergál
 - A TCP ACK ütemezett (azaz csak minden ACK esetén történik esemény)
 - A nyugták beérkezési gyorsasága határozza meg, hogy milyen gyorsan tud reagálni
 - Nagy RTT → késleltetett nyugták → a TCP csak lassan reagál a megváltozott viszonyokra

Célok

28

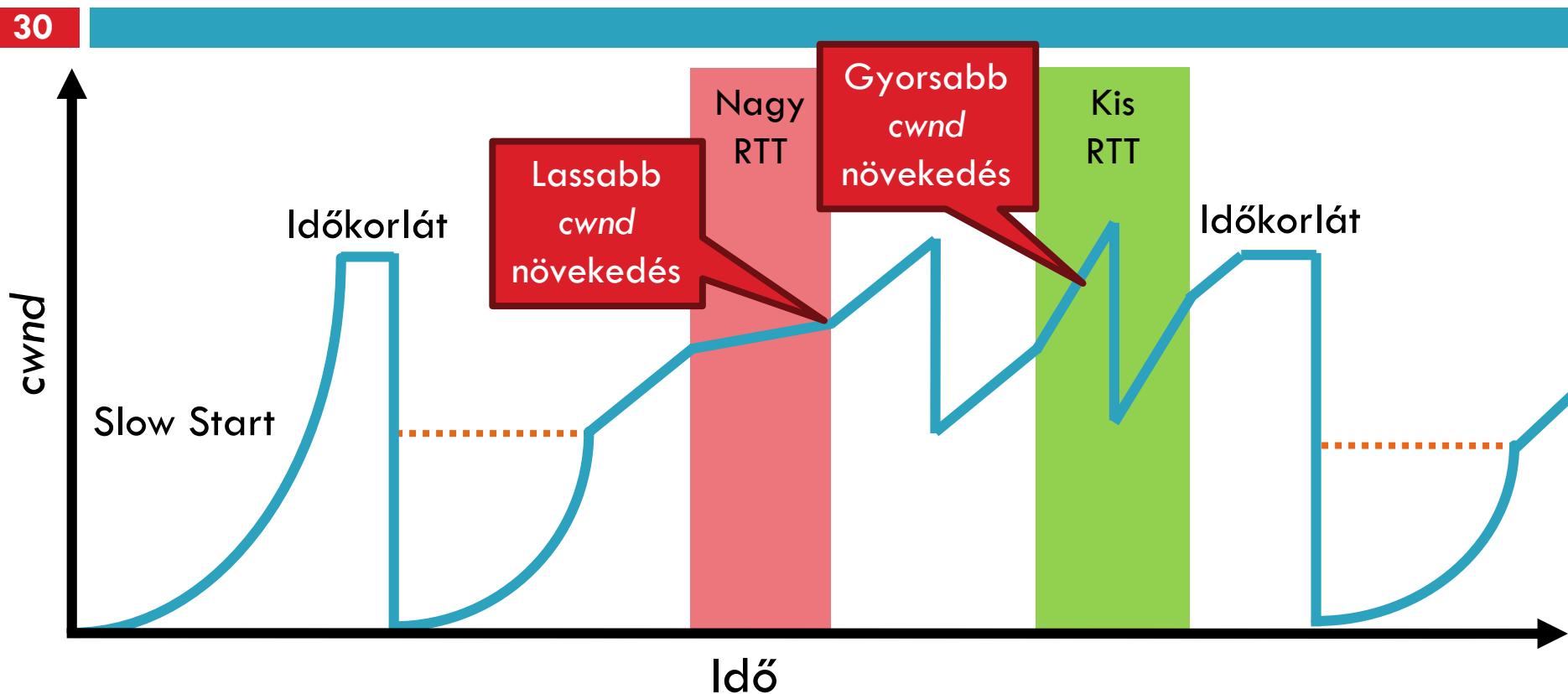
- A TCP ablak gyorsabb növelése
 - A slow start és az additive increase túl lassú, ha nagy a sávszélesség
 - Sokkal gyorsabb konvergencia kell
- Fairség biztosítása más TCP változatokkal szemben
 - Az ablak növelése nem lehet túl agresszív
- Javított RTT fairség
 - A TCP Tahoe/Reno folyamok nem adnak fair erőforrás-megosztást nagyon eltérő RTT-k esetén
- Egyszerű implementáció

Compound TCP

29

- Alap TCP implementáció Windows rendszereken
- Ötlet: osszuk a *torlódási ablakot* két különálló ablakba
 - Hagyományos, vesztés alapú ablak
 - Új, késleltetés alapú ablak
- $wnd = \min(cwnd + dwnd, adv_wnd)$
 - cwnd-t az AIMD vezérli AIMD
 - *dwnd* a késleltetés alapú ablak
- A *dwnd* beállítása:
 - Ha nő az RTT, csökken a *dwnd* ($dwnd \geq 0$)
 - Ha csökken az RTT, nő a *dwnd*
 - A növekedés/csökkenés arányos a változás mértékével

Compound TCP példa



- Agresszívan reagál az RTT változására
- Előnyök: Gyors felfutás, sokkal fairebb viselkedés más folyamokkal szemben eltérő RTT esetén
- Hátrányok: folyamatos RTT becslés

TCP CUBIC

31

- Alap TCP implementáció Linux rendszereken
- Az AIMD helyettesítése egy „köbös” (CUBIC) függvényel

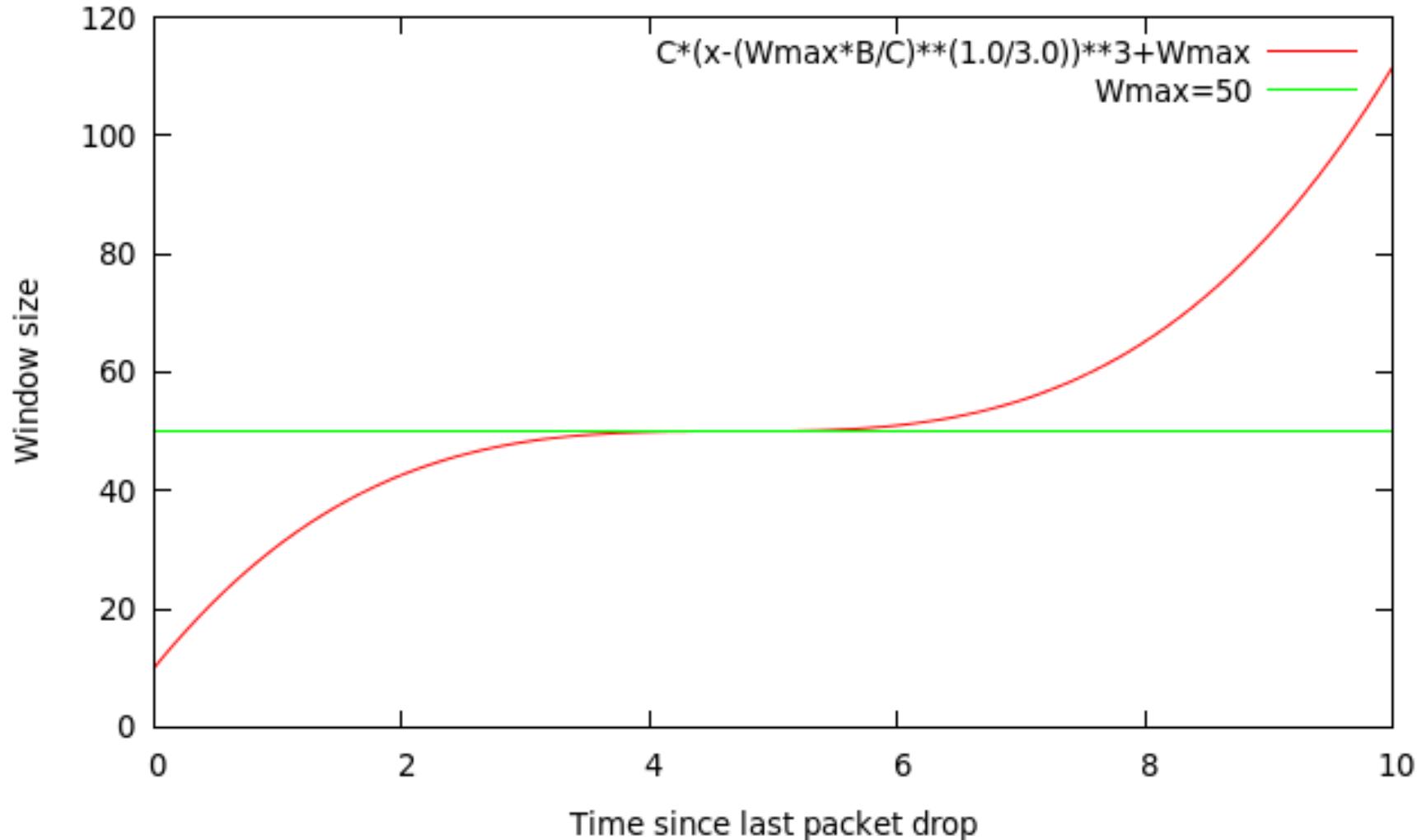
$$W_{cubic} = C(T - K)^3 + W_{max} \quad (1)$$

C is a scaling constant, and $K = \sqrt[3]{\frac{W_{max}\beta}{C}}$

- B → egy konstans a multiplicative increase fázishoz
- T → eltelt idő a legutóbbi csomagvesztés óta
- W_max → cwnd a legutolsó csomagvesztés idején

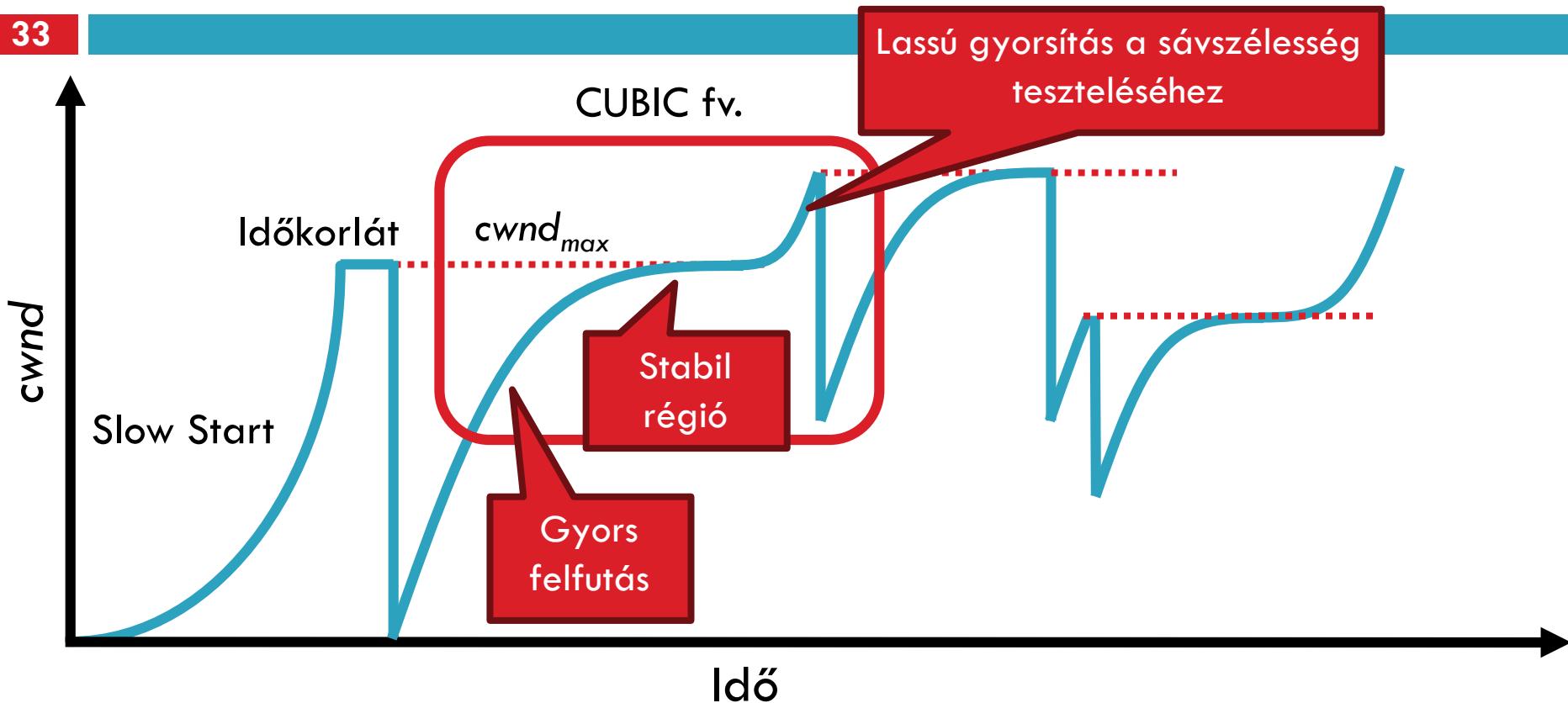
TCP CUBIC

32



TCP CUBIC példa

33



- Kevésbé pazarolja a sávszélességet a gyors felfutások miatt
- A stabil régió és a lassú gyorsítás segít a fairség biztosításában
 - A gyors felfutás sokkal agresszívabb, mint az additive increase
 - A Tahoe/Reno variánsokkal szembeni fairséghez a CUBIC-nak nem szabad ennyire agresszívnak lennie

Problémák a TCP-vel

34

- Az Internetes forgalom jelentős része TCP
- Azonban számos probléma okozója is egyben
 - Gyenge teljesítmény kis folyamok esetén
 - Gyenge teljesítmény wireless hálózatokban
 - DoS támadási felület

Kis folyamok (flows)

35

- Probléma: kis folyamok esetén torz viselkedés
 - 1 RTT szükséges a kapcsolat felépítésére (SYN, SYN/ACK)
 - pazarló
 - cwnd mindenkorban 1-gyel indul
 - Nincs lehetőség felgyorsulni a kevés adat miatt
- Az Internetes forgalom nagy része kis folyam
 - Többnyire HTTP átvitel, <100KB
 - A legtöbb TCP folyam el se hagyja a slow start fázist!!!
- Lehetséges megoldás (Google javaslat):
 - Kezdeti cwnd megnövelése 10-re
 - TCP Fast Open: kriptográfiai hashek használata a fogadó azonosítására, a három-utas kézfogás elhagyható helyette hash (cookie) küldése a syn csomagban

Wireless hálózatok

36

- Probléma: A Tahoe és Reno esetén
csomagvesztés = torlódás
 - WAN esetén ez helyes, ritka bit hibák
 - Azonban hamis vezeték nélküli hálózatokban, gyakori interferenciák
- TCP átvitel $\sim 1/\sqrt{\text{vesztési ráta}}$
 - Már néhány interferencia miatti csomagvesztés elég a teljesítmény drasztikus csökkenéséhez
- Lehetséges megoldások:
 - Réteg modell megsértése, adatkapcsolati információ a TCP-be
 - Késleltetés alapú torlódás vezérlés használata (pl. TCP Vegas)
 - Explicit torlódás jelzés - Explicit congestion notification (ECN)

Szolgáltatás megtagadása

Denial of Service (DoS)

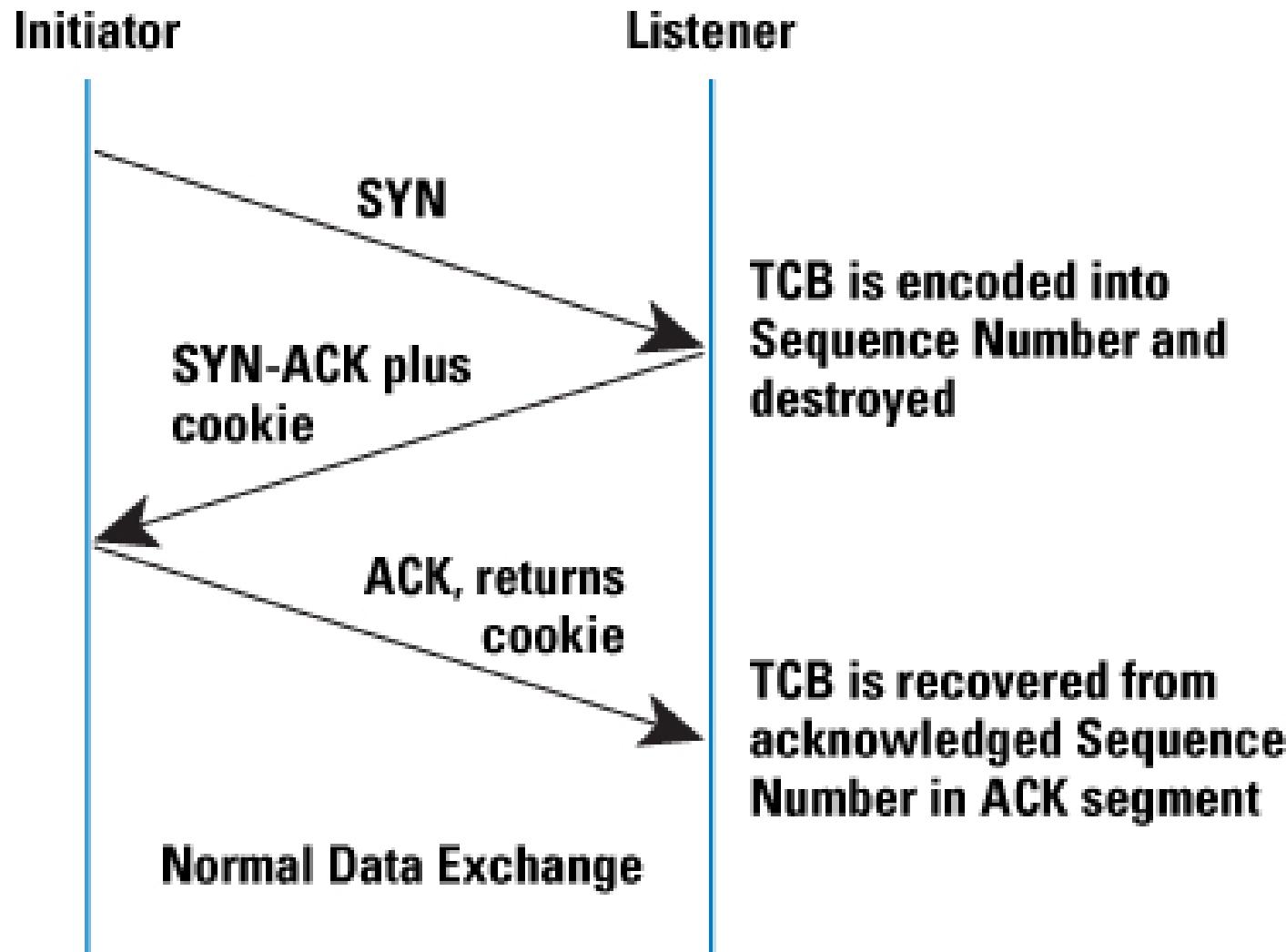
37

- Probléma: a TCP kapcsolatok állapottal rendelkeznek
 - A SYN csomagok erőforrásokat foglalnak az szerveren
 - Az állapot legalább néhány percig fennmarad (RTO)
- SYN flood: elég sok SYN csomag küldése a szervernek ahhoz, hogy elfogyjon a memória és összeomoljon a kernel
- Megoldás: SYN cookie-k
 - Ötlet: ne tároljunk kezdeti állapotot a szerveren
 - Illesszük az állapotot a SYN/ACK csomagokba (a sorszám mezőbe (sequence number mező))
 - A kliensnek vissza kell tükrözni az állapotot...

Szolgáltatás megtagadása

Denial of Service (DoS)

38



Kitekintés

39

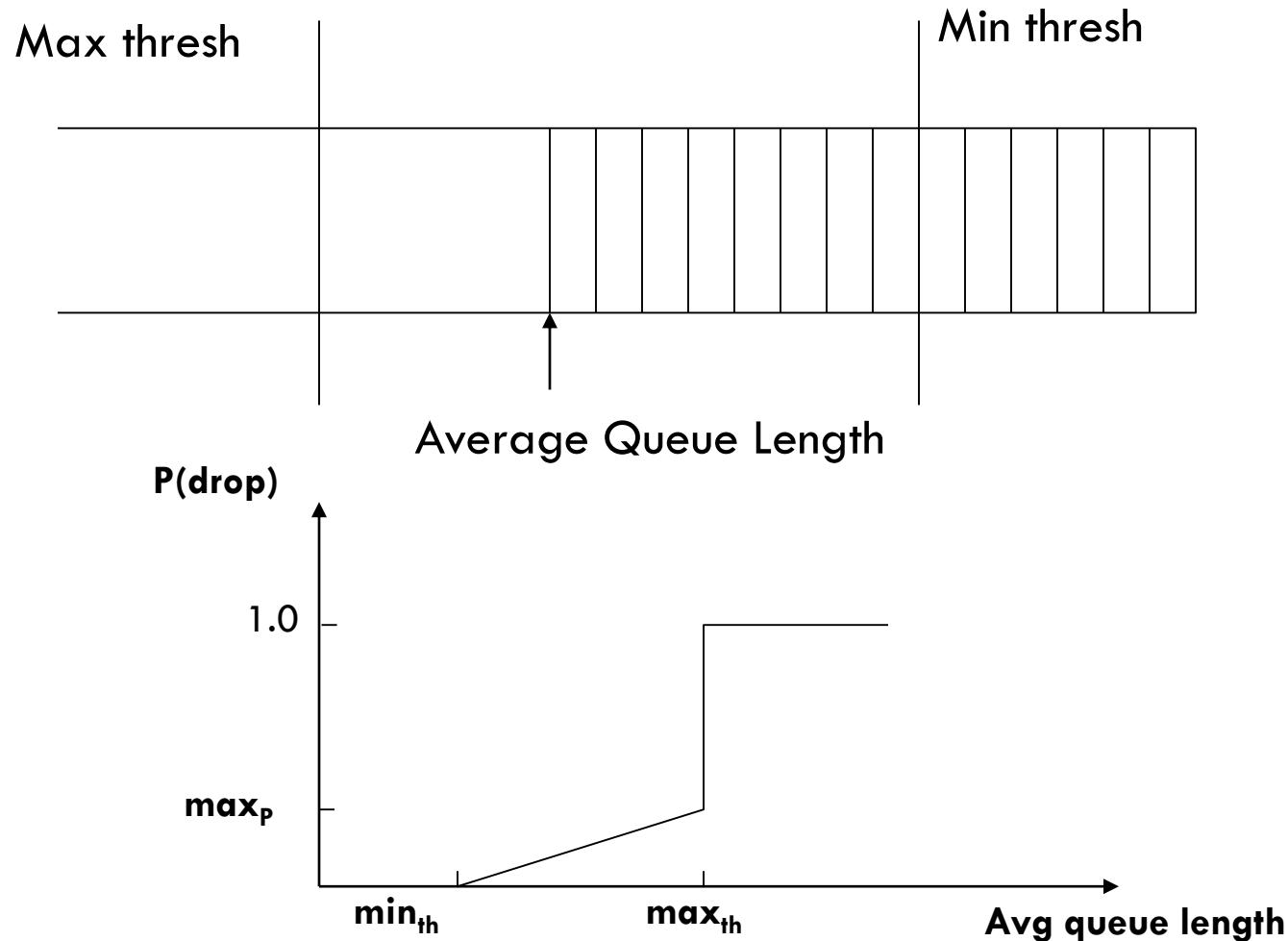
Typical Internet Queuing

- FIFO + drop-tail
 - ▣ Simplest choice
 - ▣ Used widely in the Internet
- FIFO (first-in-first-out)
 - ▣ Implies single class of traffic
- Drop-tail
 - ▣ Arriving packets get dropped when queue is full regardless of flow or importance
- Important distinction:
 - ▣ FIFO: scheduling discipline
 - ▣ Drop-tail: drop policy

RED Algorithm

- Maintain running average of queue length
- If $\text{avgq} < \text{min}_{\text{th}}$ do nothing
 - Low queuing, send packets through
- If $\text{avgq} > \text{max}_{\text{th}}$, drop packet
 - Protection from misbehaving sources
- Else mark packet in a manner proportional to queue length
 - Notify sources of incipient congestion
 - E.g. by ECN IP field or dropping packets with a given probability

RED Operation



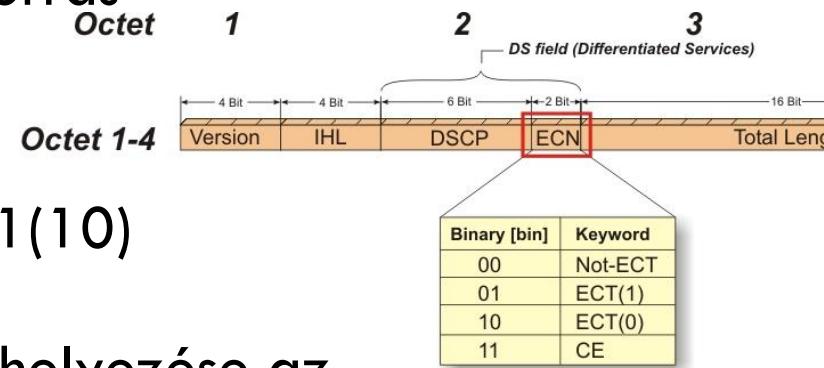
RED Algorithm

- Maintain running average of queue length
- For each packet arrival
 - Calculate average queue size (avg)
 - If $\text{min}_{\text{th}} \leq \text{avg} < \text{max}_{\text{th}}$
 - Calculate probability P_a
 - With probability P_a
 - Mark the arriving packet: drop or set-up ECN
 - Else if $\text{max}_{\text{th}} \leq \text{avg}$
 - Mark the arriving packet: drop, ECN

Csomag dobás vagy ECN jelölés

44

- Csomag dobás
 - Újraküldés szükséges
 - Egyszerűbb megvalósítás
 - Timeout lejárta után tud reagálni a forrás
- ECN jelölés
 - Végpont támogatás szükséges
 - Az IP csomag ECT-0 (01) vagy ECT-1(10) jelöléssel
 - Dobás helyett -> ECN CE (11) jel elhelyezése az IP fejlécben
 - A fogadó érzékeli a CE jelet, majd a visszamenő TCP nyugtába bebillent egy ECE flaget, mely jelzi a forrásnak a torlódást
 - Hagyományos TCP (CUBIC, RENO, stb.) források az ECE flaget csomagvesztésnek értelmezik, de újraküldés nem szükséges.



Data Center TCP: DCTCP

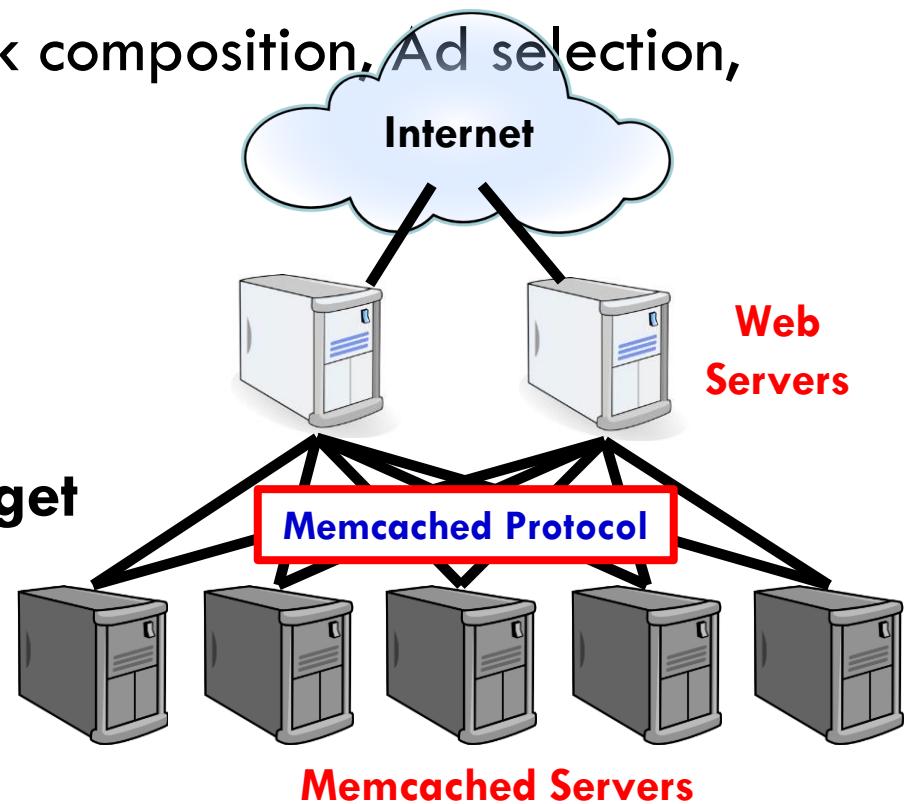
Generality of Partition/Aggregate

- The foundation for many large-scale web applications.
 - Web search, Social network composition, Ad selection, etc.

□ Example: Facebook

Partition/Aggregate ~ Multiget

- Aggregators: **Web Servers**
- Workers: **Memcached Servers**



Workloads

47

- Partition/Aggregate
(Query)



Delay-sensitive



- Short messages [50KB-1MB]
(Coordination, Control state)



Delay-sensitive



- Large flows [1MB-50MB]
(Data update)



Throughput-sensitive



Impairments

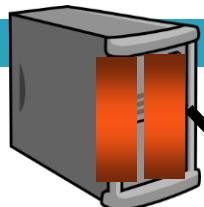
48

- Incast
- Queue Buildup
- Buffer Pressure

Incast

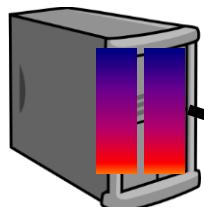
49

Worker 1

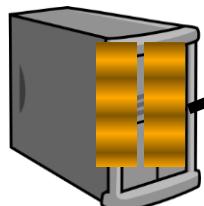


- Synchronized mice collide.
➤ Caused by Partition/Aggregate.

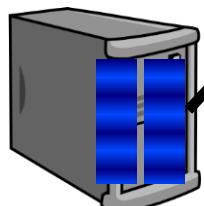
Worker 2



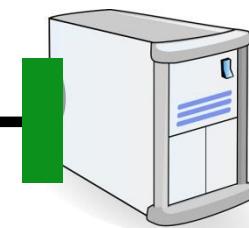
Worker 3



Worker 4



Aggregator

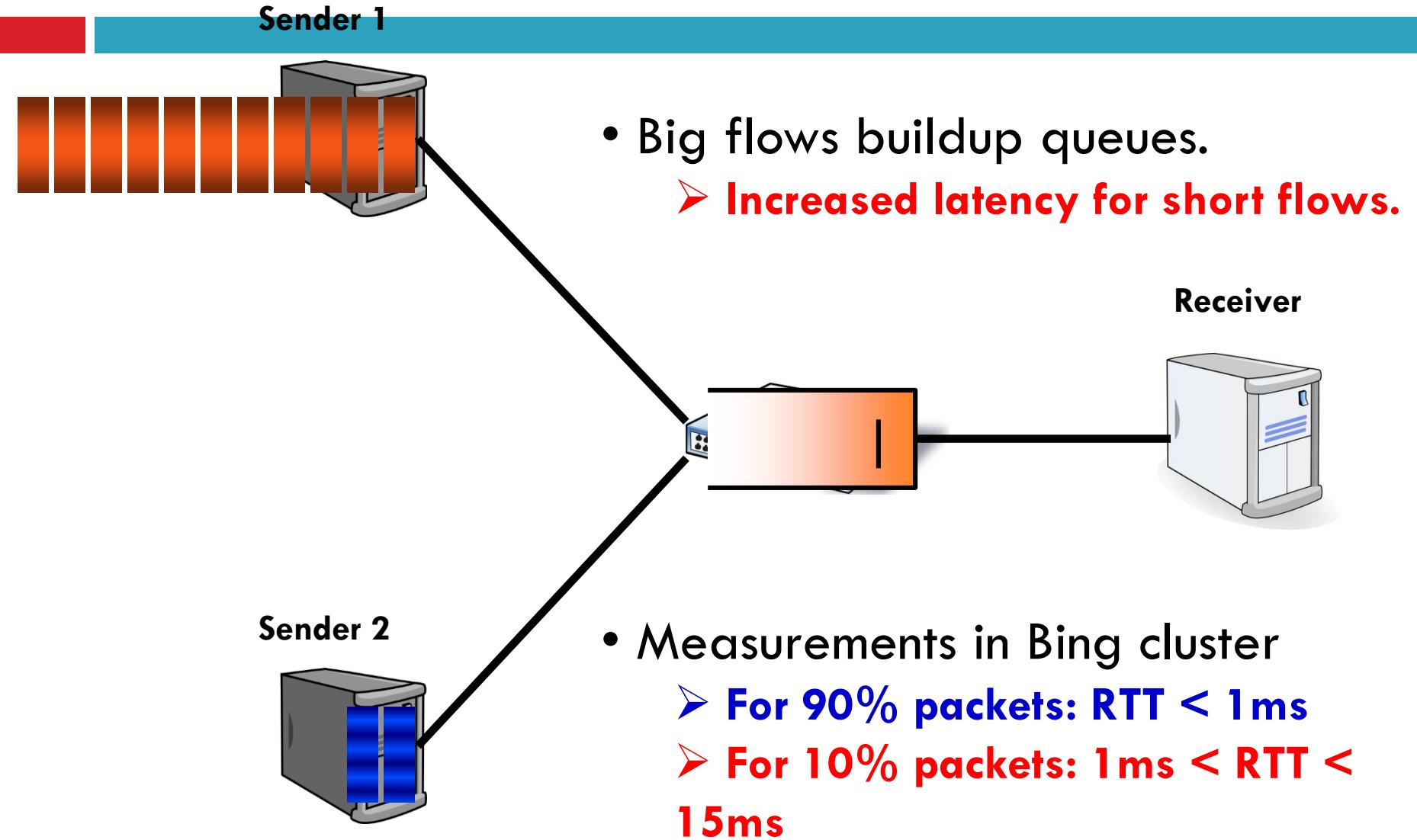


$RTO_{min} = 300 \text{ ms}$

TCP timeout



Queue Buildup



Data Center Transport Requirements

51

1. High Burst Tolerance

- Incast due to Partition/Aggregate is common.

2. Low Latency

- Short flows, queries

3. High Throughput

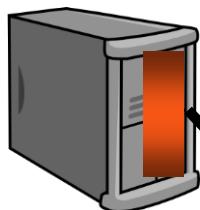
- Continuous data updates, large file transfers

The challenge is to achieve these three together.

DCTCP: The TCP/ECN Control Loop

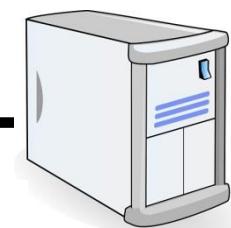
Sender 1

ECN = Explicit Congestion Notification

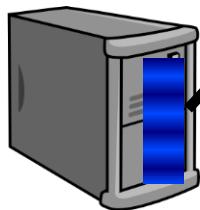


ECN Mark (1 bit)

Receiver



Sender 2



DCTCP: Two Key Ideas

18

1. React in proportion to the **extent** of congestion, not its **presence**.
 - ✓ Reduces **variance** in sending rates, lowering queuing requirements.

ECN Marks	TCP	DCTCP
1 0 1 1 1 0 1 1 1	Cut window by 50%	Cut window by 40%
0 0 0 0 0 0 0 0 1	Cut window by 50%	Cut window by 5%

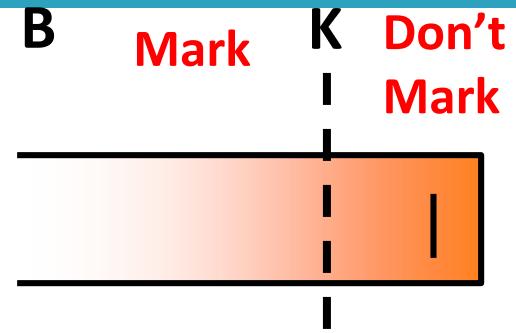
2. Mark based on **instantaneous** queue length.
 - ✓ Fast feedback to better deal with bursts.

Data Center TCP Algorithm

19

Switch side:

- Mark packets when **Queue Length > K**.



Sender side:

- Maintain running average of **fraction** of packets marked (α).

In each RTT:

$$F = \frac{\# \text{ of marked ACKs}}{\text{Total } \# \text{ of ACKs}}$$

$$\alpha \leftarrow (1 - g)\alpha + gF$$

- **Adaptive window decreases:** $Cwnd \leftarrow (1 - \frac{\alpha}{2})Cwnd$
- Note: decrease factor between 1 and 2.

DNS

„8. réteg” (A szénalapú csomópontok)

56

- Ha...
 - Fel szeretnél hívni valakit, akkor el kell kérned a telefonszámát
 - Nem hívhatod csak úgy “P I S T Á T”
 - Levelet küldenél valakinek, akkor szükséged van a címére
- Mi a helyzet az Internettel?
 - Ha el akarod érni a Google-t, szükséges annak IP címe
 - Tudja valaki a Google IP címét???
- A probléma bennünk van:
 - Az emberek nem képesek IP címek megjegyzésére
 - Ember számára értelmes nevek kellenek, melyek IP címekre képezhetők

Internetes nevek és címek

57

- Címek, pl. 129.10.117.100
 - Számítógépek által használt címkék a gépek azonosítására
 - A hálózat szerkezetét tükrözi
- Nevek, pl. www.northeastern.edu
 - Ember számára értelmes címkék a gépeknek
 - A szervezeti struktúrát tükrözi
- Hogyan képezzünk az egyikről a másikra?
 - Domain Name System (DNS)

Réges régen...

58

- A DNS előtt minden név-IP leképezés egy *hosts.txt*-ben volt
 - /etc/hosts - Linuxon
 - C:\Windows\System32\drivers\etc\hosts - Windowson
- Központosított, manuális rendszer
 - A változásokat emailben kellett beküldeni a SRI-nek
 - SRI=Stanford Research Institute
 - A gépek periodikus időközönként letöltötték (FTP) a *hosts.txt* fájlt
 - minden név megengedett volt – nem volt benne hierarchia („flat” (sík) felépítés)
 - alans_server_at_sbu_pwns_joo_lol_kthxbye

A DNS felé

59

- Végül a *hosts.txt* alapú rendszer szétesett
 - Nem skálázható, SRI nem bírt a terheléssel/igényekkel
 - Nehéz volt a nevek egyediségének biztosítása
 - PI. MIT
 - Massachusetts Institute of Technology?
 - Melbourne Institute of Technology?
 - Számos gép rendelkezett nem naprakész *hosts.txt*-vel
- Ez vezetett a DNS megszületéséhez...

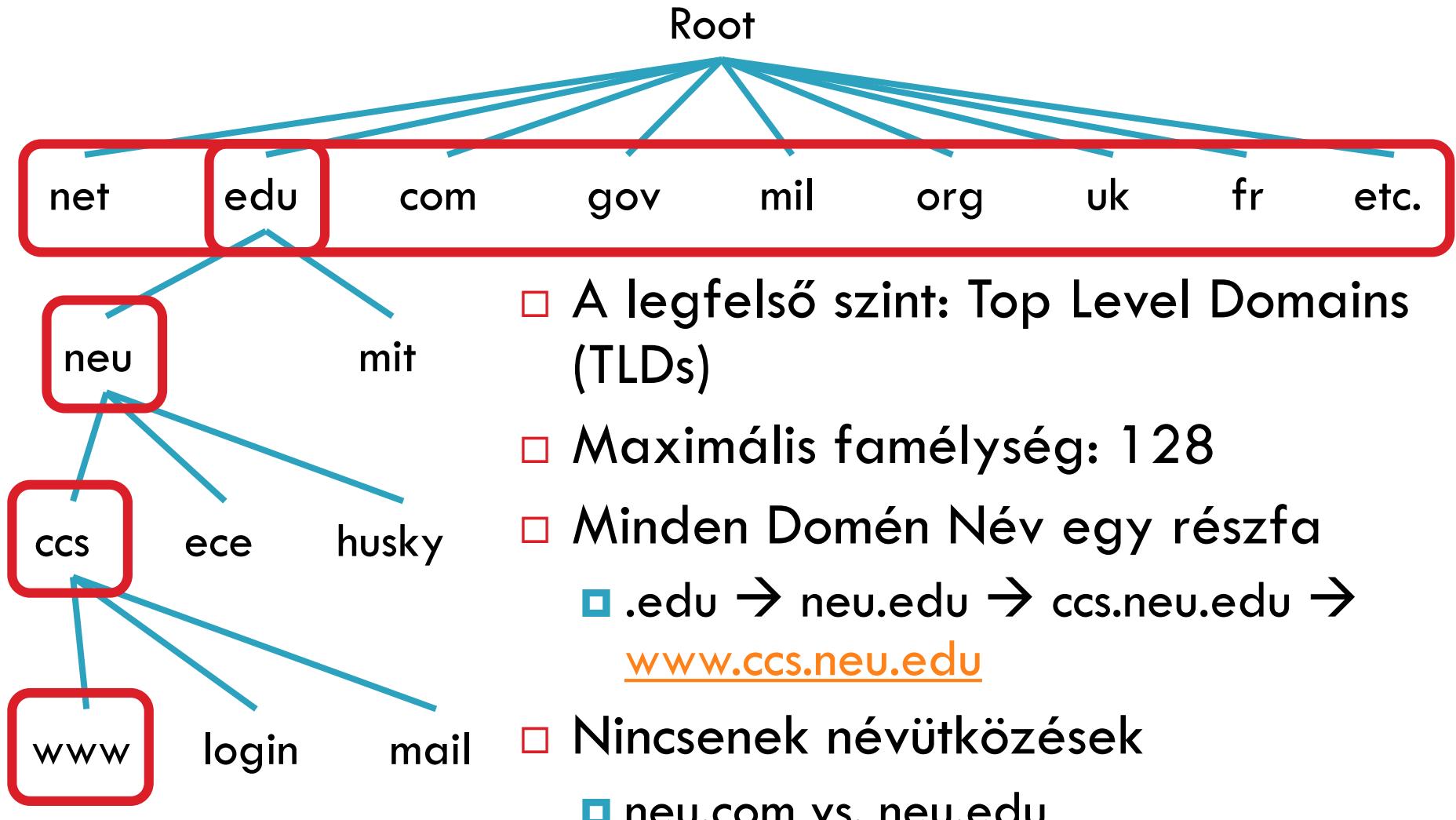
DNS általánosságban

60

- Domain Name System
- Elosztott adatbázis
 - Nem központosított
- Egyszerű kliens-szerver architektúra
 - UDP 53-as port, vannak TCP implementációk is
 - Rövid kérések – rövid válaszok; kérés-válasz típusú kommunikáció
- Hierarchikus névtér
 - Szemben a hosts.txt alapú flat megoldással
 - pl. .com → google.com → mail.google.com

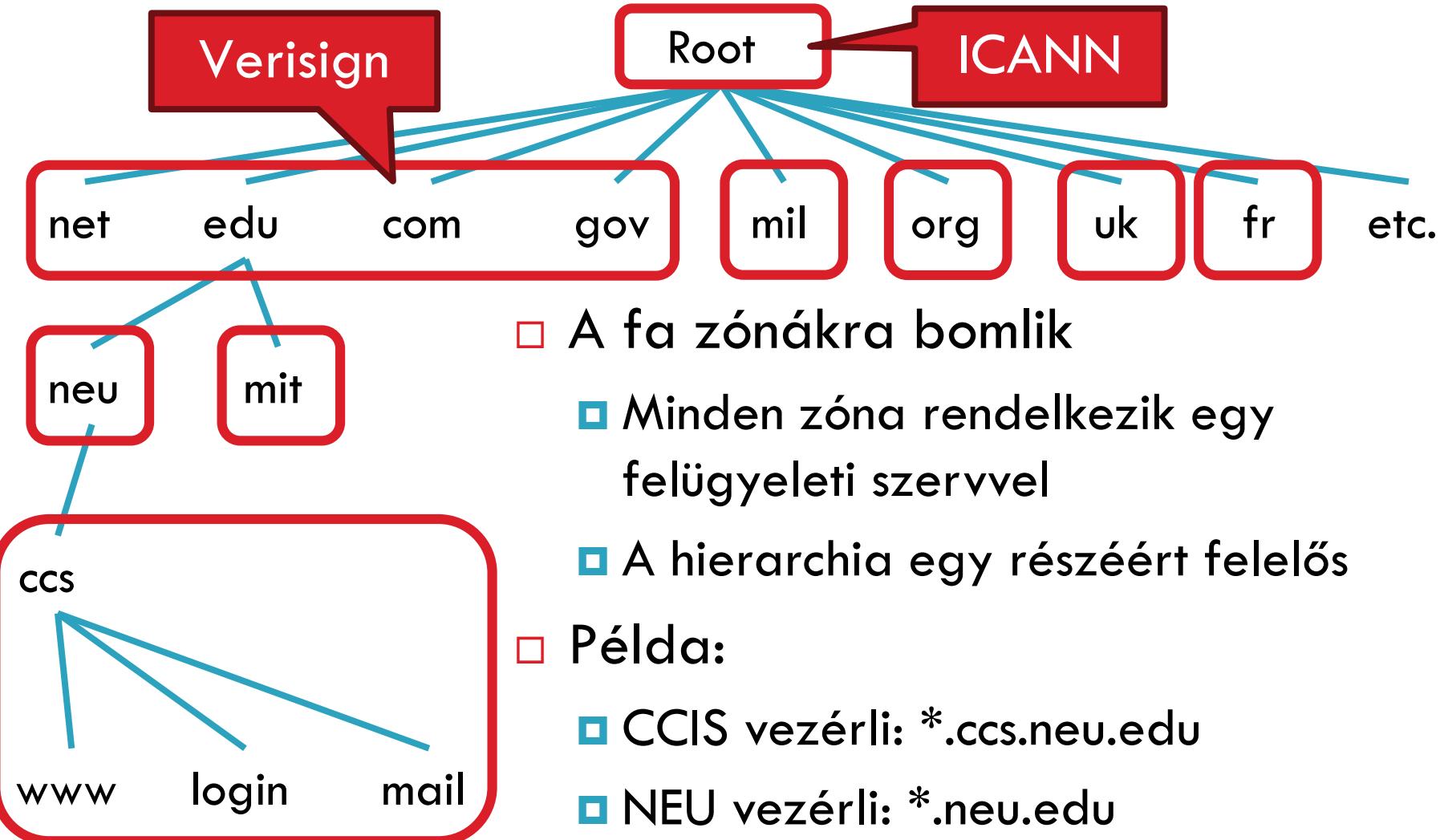
Név hierarchia

61



Hierarchikus adminisztráció

62



Szerver hierarchia

63

- Egy DNS szerver funkciói:
 - A hierarchia egy részét felügyeli
 - Nem szükséges minden DNS nevet tárolnia
 - A zónájához tartozó összes hoszt és domén rekordjainak tárolása
 - Másolatok lehetnek a robosztusság növelés végett
 - Ismeri a root szerverek címét
 - Ismeretlen nevek feloldása miatt kell
- A root szerverek minden TLD-t ismernek
 - Azaz innen indulva fel lehet tárni...

Top Level Domains

- Internet Corp. Assigned Names and Numbers (1998)
- 22+ általános TLDs létezik
 - klasszikusok: .com, .edu, .gov, .mil, .org, .net
 - később keletkeztek: .aero, .museum, .xxx
- ~250 TLDs a különböző ország kódoknak
 - Két betű (mint például .au, .hu), 2010-től plusz nemzetközi karakterek (például kínai)
 - Több elüzletisedett, például a .tv (Tuvalu)
 - Példa domén hack-ekre: instagr.am (Örményország), goo.gl (Grönland)

Root Name Servers

65

□ A Root Zone Fájlért felelős

- Listát vezet a TLD-kről és arról, hogy ki felügyeli őket.
- ~272KB a fájl mérete
- Pl. bejegyzése:

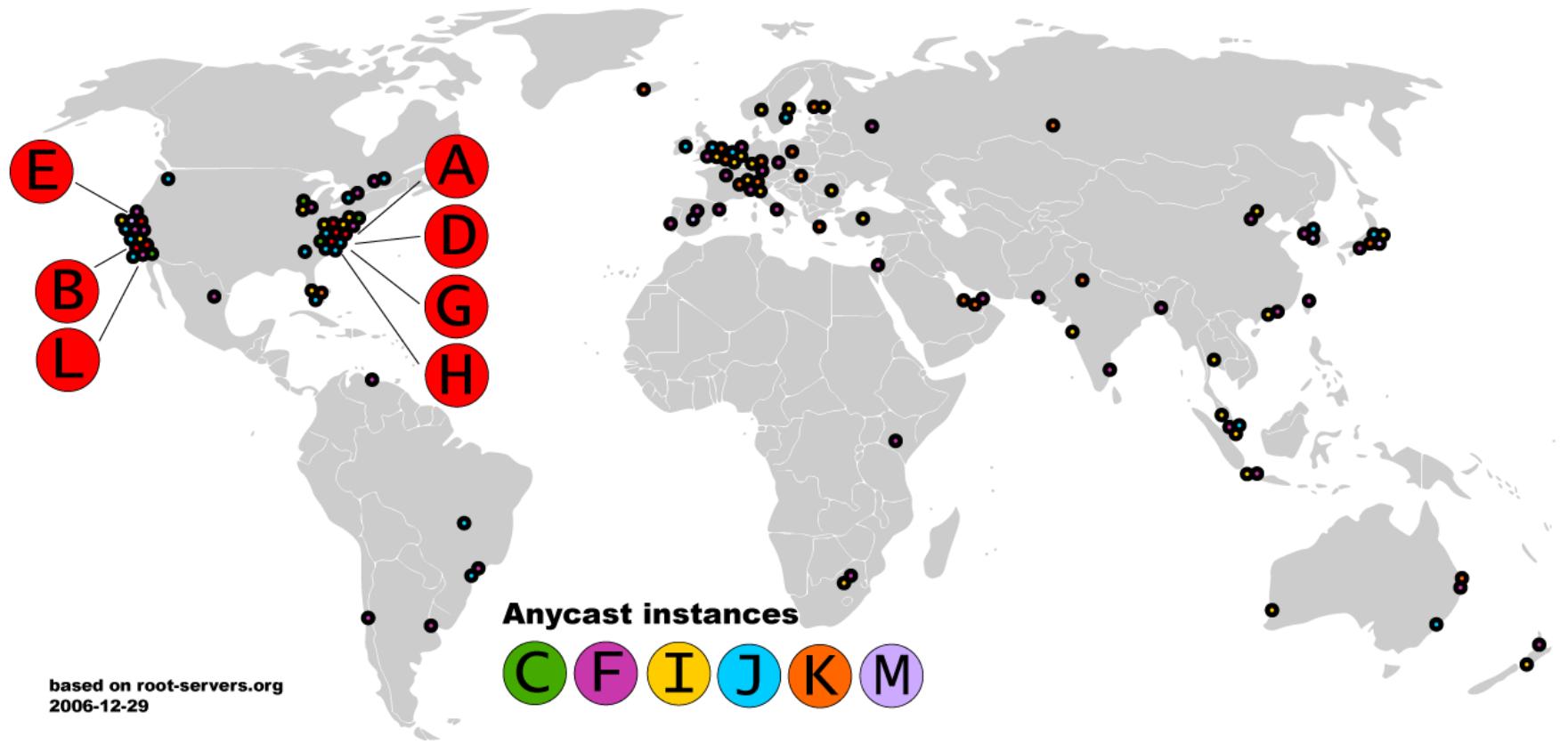
com.	172800	IN	NS	a.gtld-servers.net.
com.	172800	IN	NS	b.gtld-servers.net.
com.	172800	IN	NS	c.gtld-servers.net.

□ Az ICANN adminisztrálja

- 13 root szerver, címkék: A→M
 - Pl.: i.root-servers.net
 - 6 db ezek közül „anycastolt”, azaz globálisan számos replika létezik
- ## □ Ha név nem feloldható (lokálisan), akkor hozzájuk kell fordulni
- A gyakorlatban a legtöbb rendszer lokálisan tárolja ezt az információt (cache)

Map of the Roots

66



Lokális névszerverek

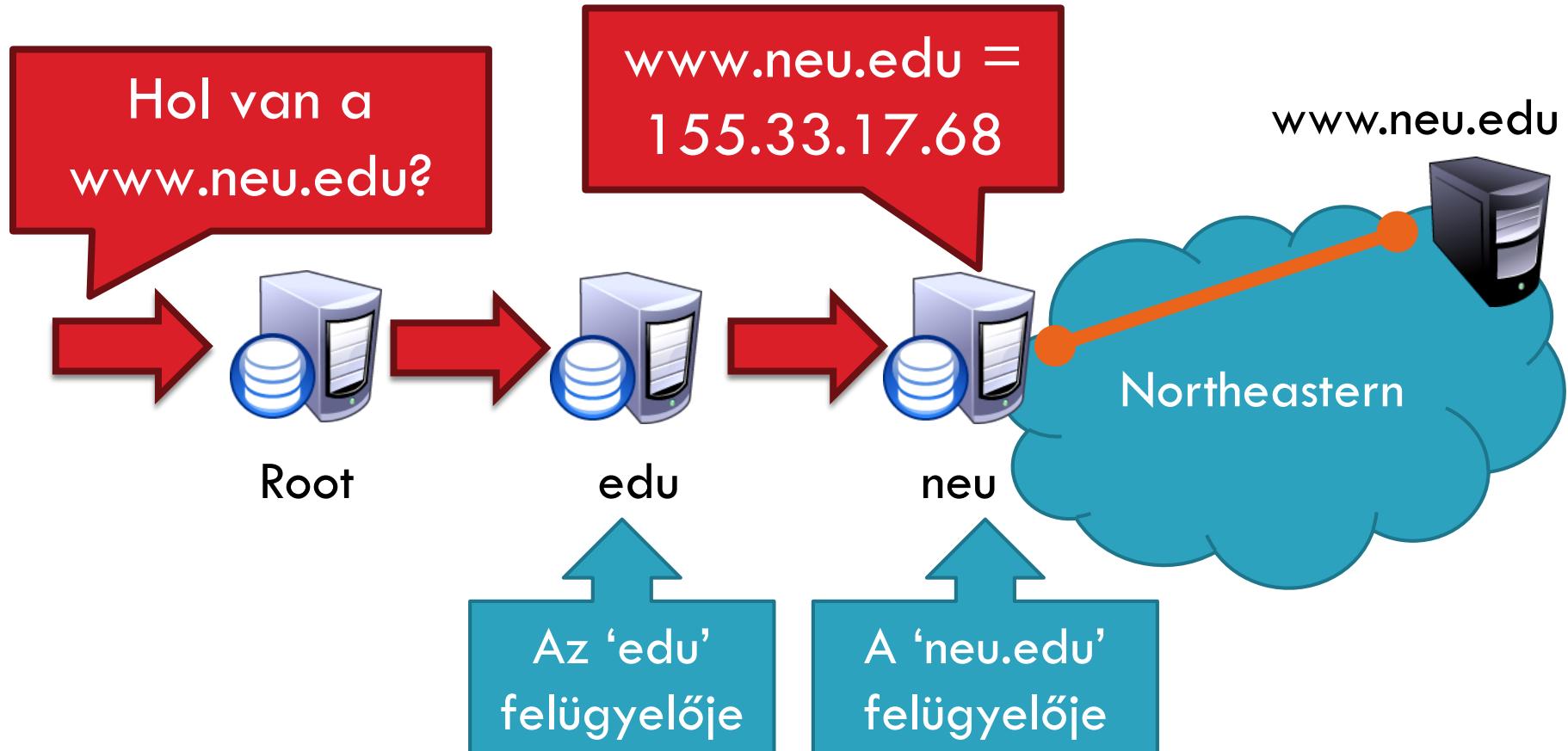
67



- minden ISP/cég rendelkezik egy lokális, default névszerverrel
- Gyakran a DHCP konfigurálja fel
- A DNS lekérdezések a lokális névszervernél kezdődnek
- Gyakran cache-be teszik a lekérdezés eredményét

Authoratív Névszerverek

68



- ❑ név → IP leképezéseket tárolja egy adott hoszthoz

Egyszerű doménnév feloldás

69

- minden hoszt ismer egy lokális DNS szervert
 - minden kérést ennek küld
- Ha a lokális DNS szerver tud válaszolni, akkor kész...
 1. A lokális szerver a felügyelő szerver az adott névhez
 2. A lokális szerver cache-ében van rekord a keresett névhez
- Különben menjünk végig a teljes hierarchián felülről lefelé egészen a keresett név felügyeleti szerveréig
 - minden lokális DNS szerver ismeri a root szervereket
 - Cache tartalma alapján bizonyos lépések átugrása, ha lehet
 - Pl. ha a root fájl tárolva van a cache-ben, akkor egyből ugorhatunk az „.edu” szerverére.

Lekérdezések

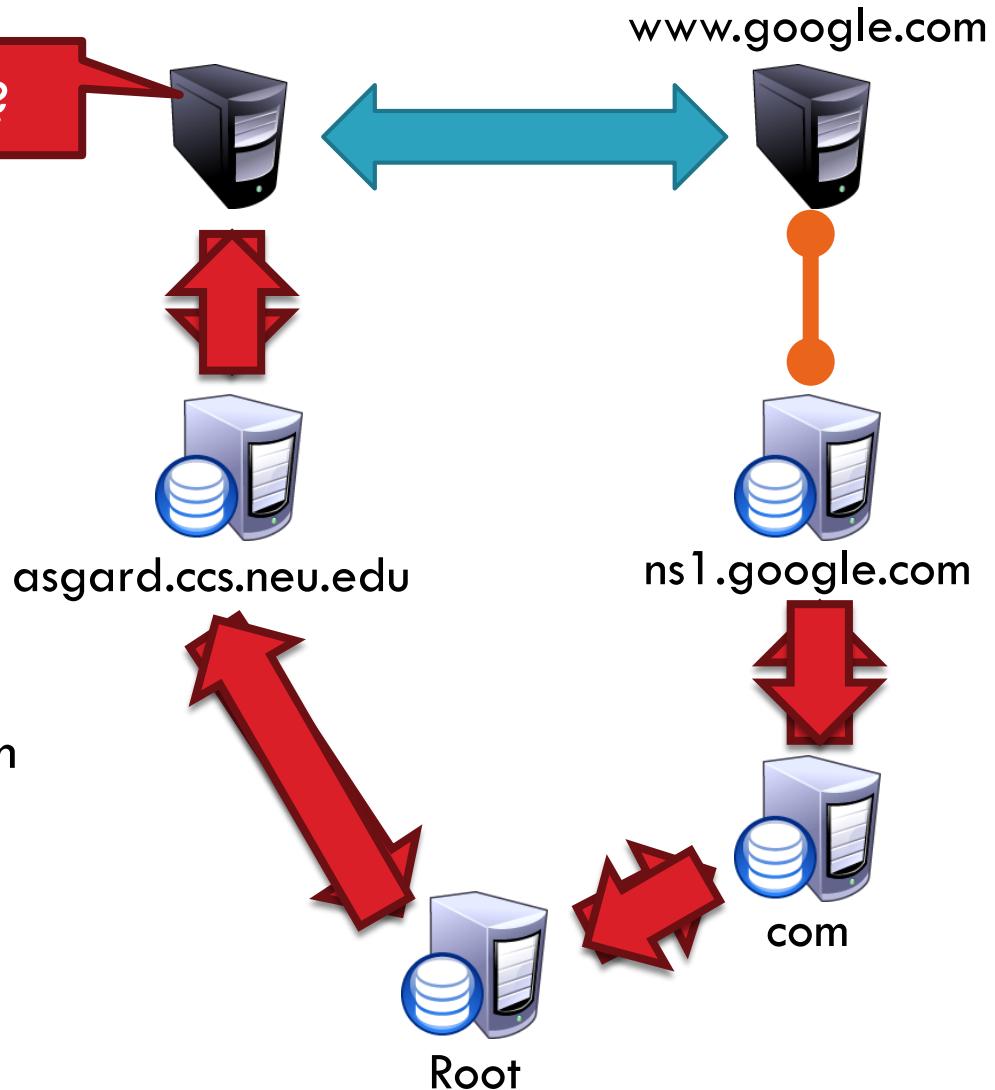
- A lekérdezésnek két fajtája van:
 - ▣ *Rekurzív lekérdezés* – Ha a névszerver végzi el a névfeloldást, és tér vissza a válasszal.
 - ▣ *Iteratív lekérdezés* – Ha a névszerver adja vissza a választ vagy legalább azt, hogy kitől kapható meg a következő válasz.
- Melyik a jobb?
 - ▣ *Rekurzív jellemzői*
 - Lehetővé teszi a szervernek a kliens terhelés kihelyezését a kezelhetőségért.
 - Lehetővé teszi a szervernek, hogy a kliensek egy csoportja felett végezzen cachelést, a jobb teljesítményért.
 - ▣ *Iteratív jellemzői*
 - Válasz után nem kell semmit tenni a kéréssel a névszervernek.
 - Könnyű magas terhelésű szervert építeni.

Rekurzív DNS lekérdezés

71

Hol van a www.google.com?

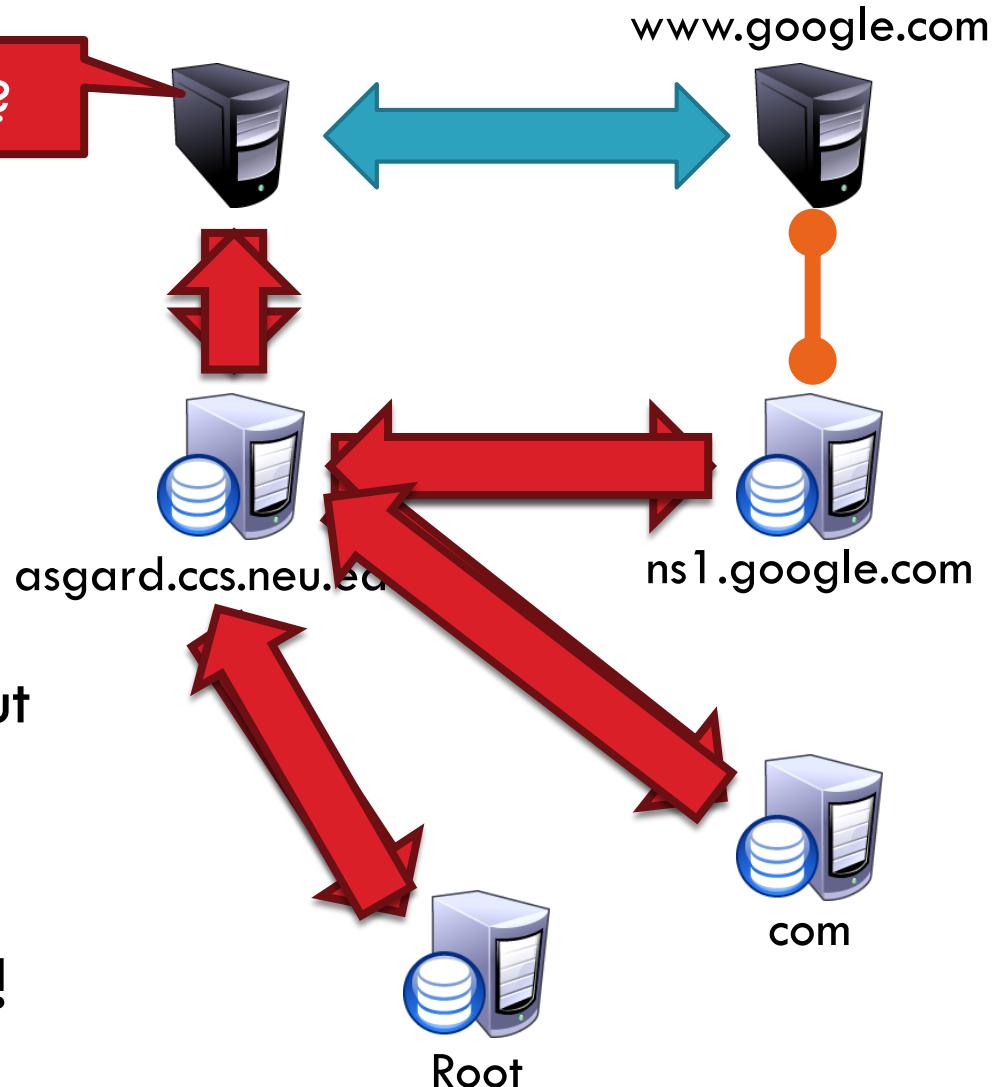
- A lokális szerver terhet rak a kérdezett névszerverre (pl. root)
- Honnan tudja a kérdezett, hogy kinek továbbítsa a választ?
 - Random ID a DNS lekérdezésben



Iteratív DNS lekérdezés

72

Hol van a www.google.com?

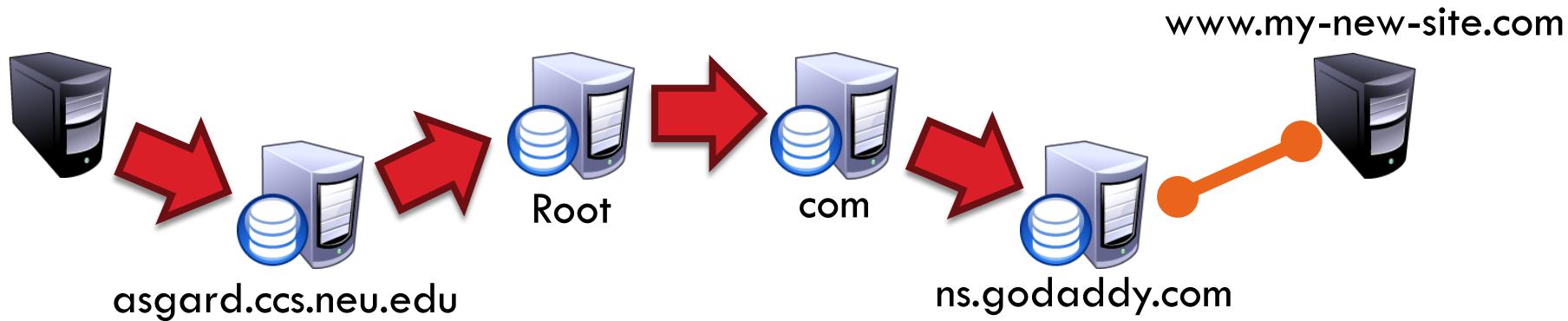


- A szerver mindenkor a következő kérdezendő névszerver adataival tér vissza
- “I don’t know this name, but this other server might”
- **Napjainkban iteratív módon működik a DNS!!!**

DNS bejegyzés elterjedése

73

- Van-e a teremben olyan, aki vásárolt már domén nevet?
 - Észrevettétek-e, hogy kb. 72 óra kell ahoz, hogy elérhető legyen a bejegyzés után?
 - Ez a késés a DNS Propagáció/DNS bejegyzés elterjedése



- Miért nem sikerül ez egy új DNS név esetén?

Cachelés VS frissesség

74

- DNS elterjedés késését a cache okozza



- A zóna fájlok 1-72 órig élnek a cache-ben



DNS Erőforrás rekordok (Resource Records)

75

- A DNS lekérdezéseknek két mezőjük van
 - `name` és `type`
- Az erőforrás rekord válasz egy DNS lekérdezésre
 - Négy mezőből áll: (`name`, `value`, `type`, TTL)
 - Egy lekérdezésre adott válaszban több rekord is szerpelhet
- Mit jelent a `name` és a `value` mező?
 - Ez a lekérdezés típusától (`type`) függ

DNS lekérdezés típusok

76

- Type = A / AAAA
 - Name = domén név
 - Value = IP cím
 - A = IPv4, AAAA = IPv6

- Type = NS
 - Name = rész domén
 - Value = a rész doménhez tartozó DNS szerver neve
 - "Menj és küldd a kérésed ehhez a szerverhez"

Query Name: www.ccs.neu.edu
 Type: A

Resp. Name: www.ccs.neu.edu
 Value: 129.10.116.81

Query Name: ccs.neu.edu
 Type: NS

Resp. Name: ccs.neu.edu
 Value: 129.10.116.51

DNS lekérdezés típusok

77

- Type = CNAME
 - Name = domén név
 - Value = kanonikus név
 - Alias nevek használatához
 - CDN használja

Query Name: [foo.mysite.com](#)
 Type: CNAME

Resp. Name: [foo.mysite.com](#)
 Value: [bar.mysite.com](#)

- Type = MX
 - Name = emailben szereplő domén név
 - Value = mail szerver kanonikus neve

Query Name: [ccs.neu.edu](#)
 Type: MX

Resp. Name: [ccs.neu.edu](#)
 Value: [amber.ccs.neu.edu](#)

Fordított lekérdezés (PTR rekord)

78

- Mi a helyzet az IP→név leképezéssel?
- Külön hierarchia tárolja ezeket a leképezéseket
 - Gyökér pont: in-addr.arpa és ip6.arpa
- DNS rekord típusa (**type**): PTR
 - Name = IP cím
 - Value = domén név
- Nincs garancia arra, hogy minden IP címre működik

Query

Name: 129.10.116.51
Type: PTR

Resp.

Name: 129.10.116.51
Value: ccs.neu.edu

DNS as Indirection Service

79

- DNS számos lehetőséget biztosít
 - Nem csak a gépekre való hivatkozást könnyíti meg!
- Egy gép IP címének lecserélése is triviális
 - Pl. a web szervert átköltöztetjük egy új hosztra
 - Csak a DNS rekord bejegyzést kell megváltoztatni!

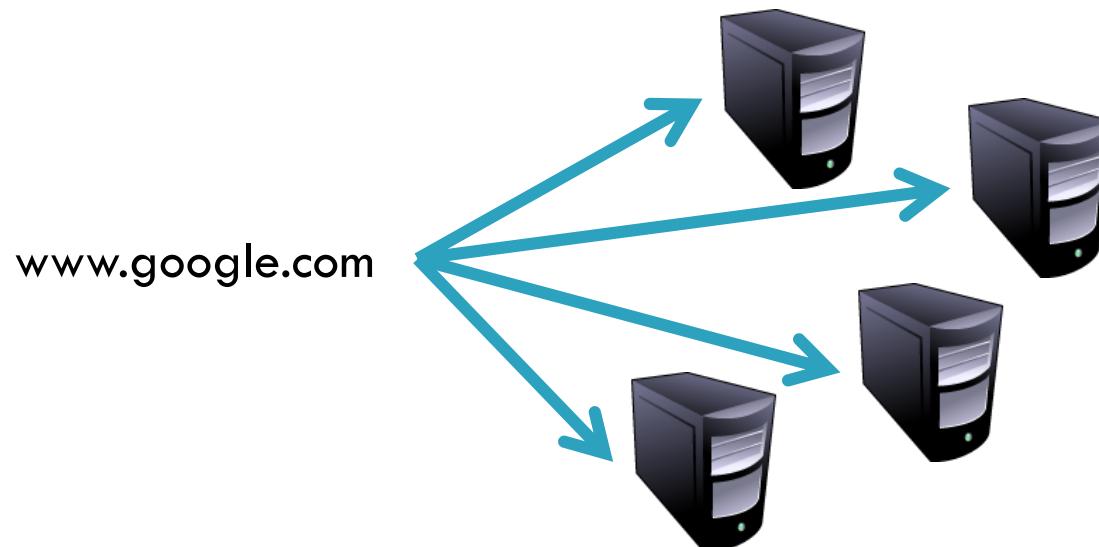
Aliasing/Kanonikus nevek és Load Balancing/Terhelés elosztás

80

- Egy gépnak számos alias neve lehet

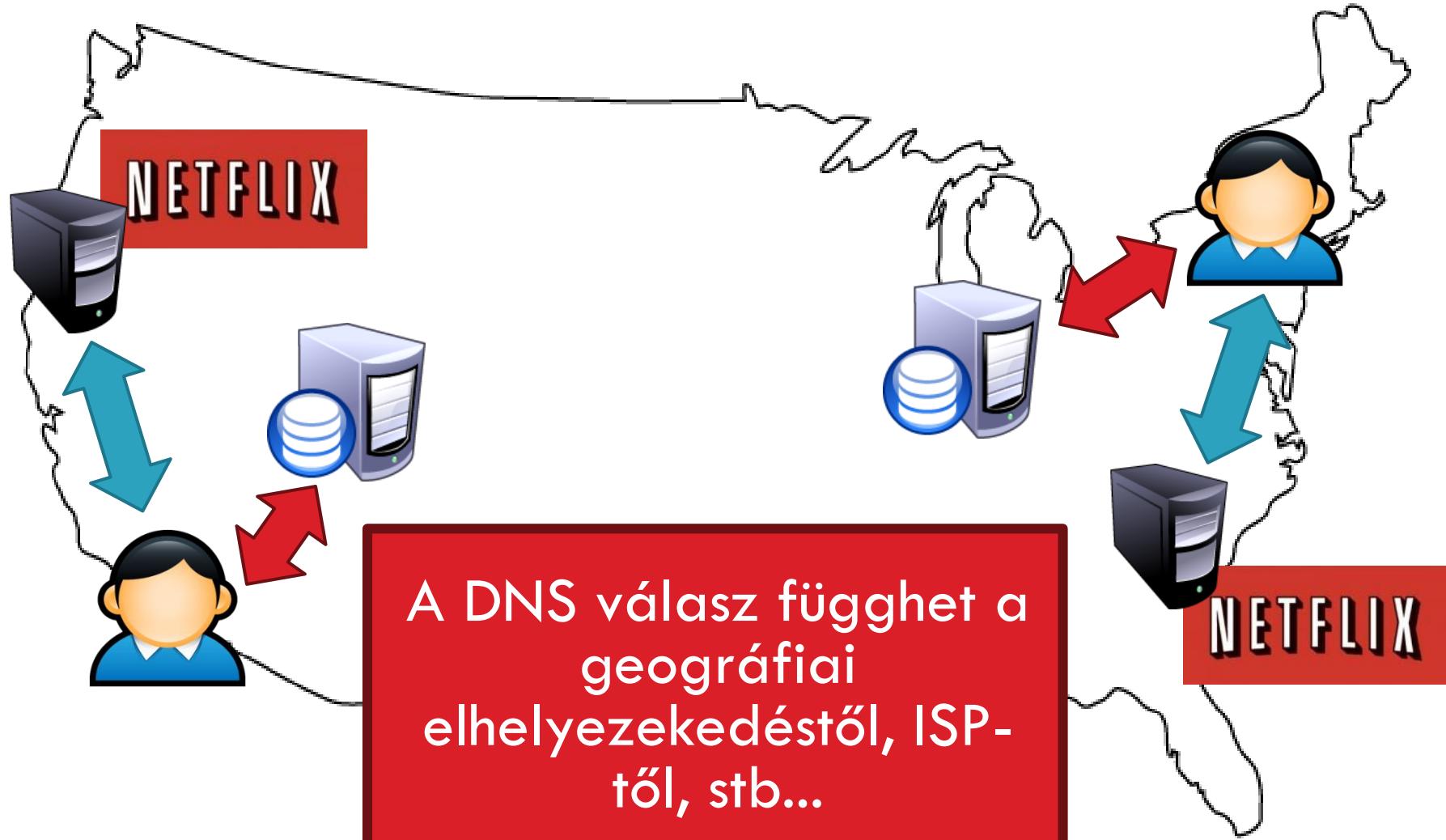


- Egy domén névhez számos IP cím tartozhat



Content Delivery Networks

81



A DNS fontossága

82

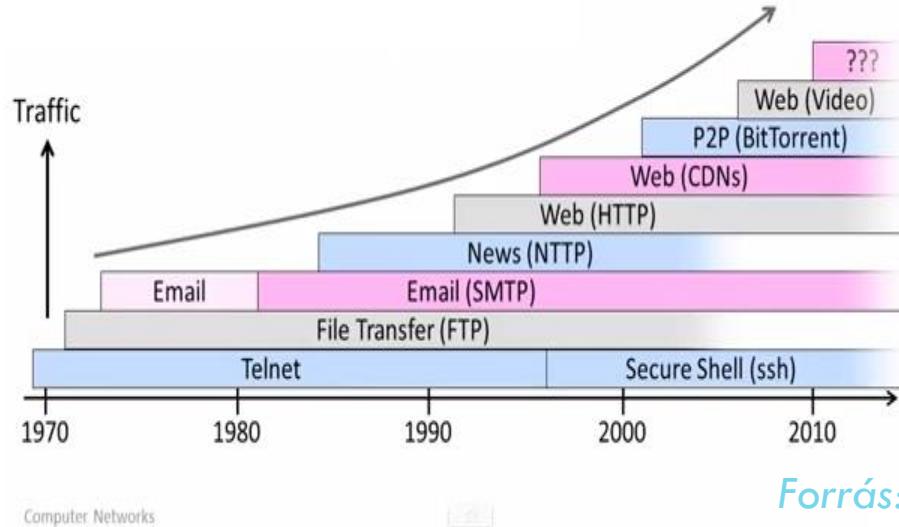
- DNS nélkül...
 - Hogyan találunk meg egy weboldalt?
- Példa: a mailszervered azonosít
 - Email címet adunk meg weboldalakra való feliratkozásnál
 - Mi van, ha valaki eltéríti a DNS bejegyzést a mailszerveredhez?
- DNS a bizalom forrása a weben
 - Amikor a felhasználó begépeli a www.bankofamerica.com címet, azt várja, hogy a bankja honlapja jelenjen meg.
 - Mi van, ha a DNS rekordot meghackelték?

Köszönöm a figyelmet!

Számítógépes Hálózatok

11. Előadás: Alkalmazási réteg
DNS, HTTP, CDN

Internetes alkalmazások evolúciója



Forrás: [1]

- Folyamatosan változik, növekszik...
 - www.evolutionoftheweb.com

3

DNS

„8. réteg” (A szénalapú csomópontok)

4

□ Ha...

- Fel szeretnél hívni valakit, akkor el kell kérned a telefonszámát

- Nem hívhatod csak úgy “P I S T Á T”

- Levelet küldenél valakinek, akkor szükséged van a címére

□ Mi a helyzet az Internettel?

- Ha el akarod érni a Google-t, szükséges annak IP címe

- Tudja valaki a Google IP címét???

□ A probléma bennünk van:

- Az emberek nem képesek IP címek megjegyzésére

- Ember számára értelmes nevek kellenek, melyek IP címekre képezhetők

Internetes nevek és címek

5

- Címek, pl. 129.10.117.100
 - Számítógépek által használt címkék a gépek azonosítására
 - A hálózat szerkezetét tükrözi
- Nevek, pl. www.northeastern.edu
 - Ember számára értelmes címkék a gépeknek
 - A szervezeti struktúrát tükrözi
- Hogyan képezzünk az egyikről a másikra?
 - Domain Name System (DNS)

Réges régen...

6

- A DNS előtt minden név-IP leképezés egy *hosts.txt*-ben volt
 - /etc/hosts - Linuxon
 - C:\Windows\System32\drivers\etc\hosts - Windowson
- Központosított, manuális rendszer
 - A változásokat emailben kellett beküldeni a SRI-nek
 - SRI=Stanford Research Institute
 - A gépek periodikus időközönként letöltötték (FTP) a *hosts.txt* fájlt
 - minden név megengedett volt – nem volt benne hierarchia („flat” (sík) felépítés)
 - alans_server_at_sbu_pwns_joo_lol_kthxbye

A DNS felé

7

- Végül a *hosts.txt* alapú rendszer szétesett
 - Nem skálázható, SRI nem bírt a terheléssel/igényekkel
 - Nehéz volt a nevek egyediségének biztosítása
 - PI. MIT
 - Massachusetts Institute of Technology?
 - Melbourne Institute of Technology?
 - Számos gép rendelkezett nem naprakész *hosts.txt*-vel
- Ez vezetett a DNS megszületéséhez...

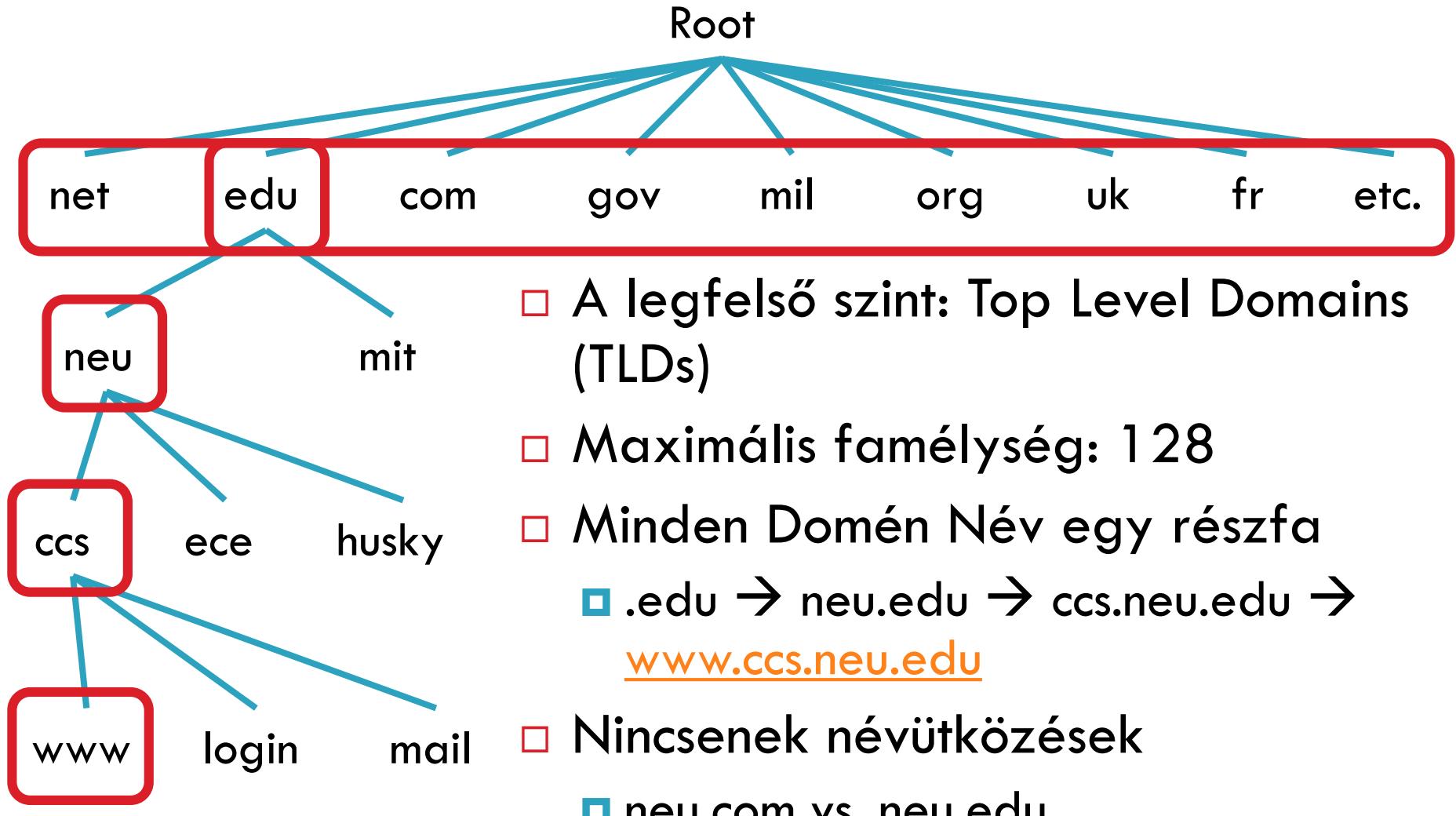
DNS általánosságban

8

- Domain Name System
- Elosztott adatbázis
 - Nem központosított
- Egyszerű kliens-szerver architektúra
 - UDP 53-as port, vannak TCP implementációk is
 - Rövid kérések – rövid válaszok; kérés-válasz típusú kommunikáció
- Hierarchikus névtér
 - Szemben a hosts.txt alapú flat megoldással
 - pl. .com → google.com → mail.google.com

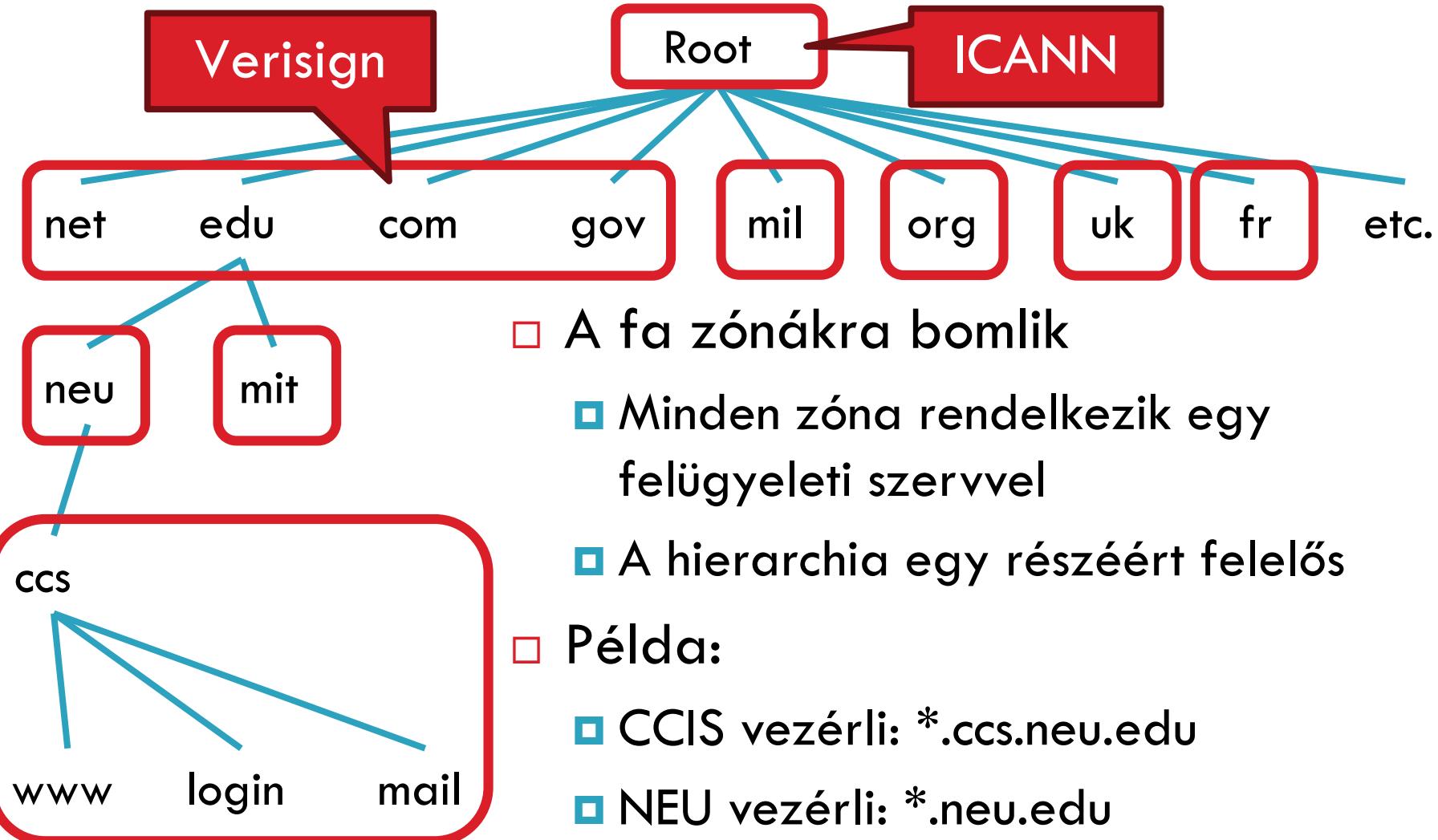
Név hierarchia

9



Hierarchikus adminisztráció

10



Szerver hierarchia

11

- Egy DNS szerver funkciói:
 - A hierarchia egy részét felügyeli
 - Nem szükséges minden DNS nevet tárolnia
 - A zónájához tartozó összes hoszt és domén rekordjainak tárolása
 - Másolatok lehetnek a robosztusság növelés végett
 - Ismeri a root szerverek címét
 - Ismeretlen nevek feloldása miatt kell
- A root szerverek minden TLD-t ismernek
 - Azaz innen indulva fel lehet tárni...

Top Level Domains

- Internet Corp. Assigned Names and Numbers (1998)
- 22+ általános TLDs létezik
 - klasszikusok: .com, .edu, .gov, .mil, .org, .net
 - később keletkeztek: .aero, .museum, .xxx
- ~250 TLDs a különböző ország kódoknak
 - Két betű (mint például .au, .hu), 2010-től plusz nemzetközi karakterek (például kínai)
 - Több elüzletisedett, például a .tv (Tuvalu)
 - Példa domén hack-ekre: instagr.am (Örményország), goo.gl (Grönland)

Root Name Servers

13

□ A Root Zone Fájlért felelős

- Listát vezet a TLD-kről és arról, hogy ki felügyeli őket.
- ~272KB a fájl mérete
- Pl. bejegyzése:

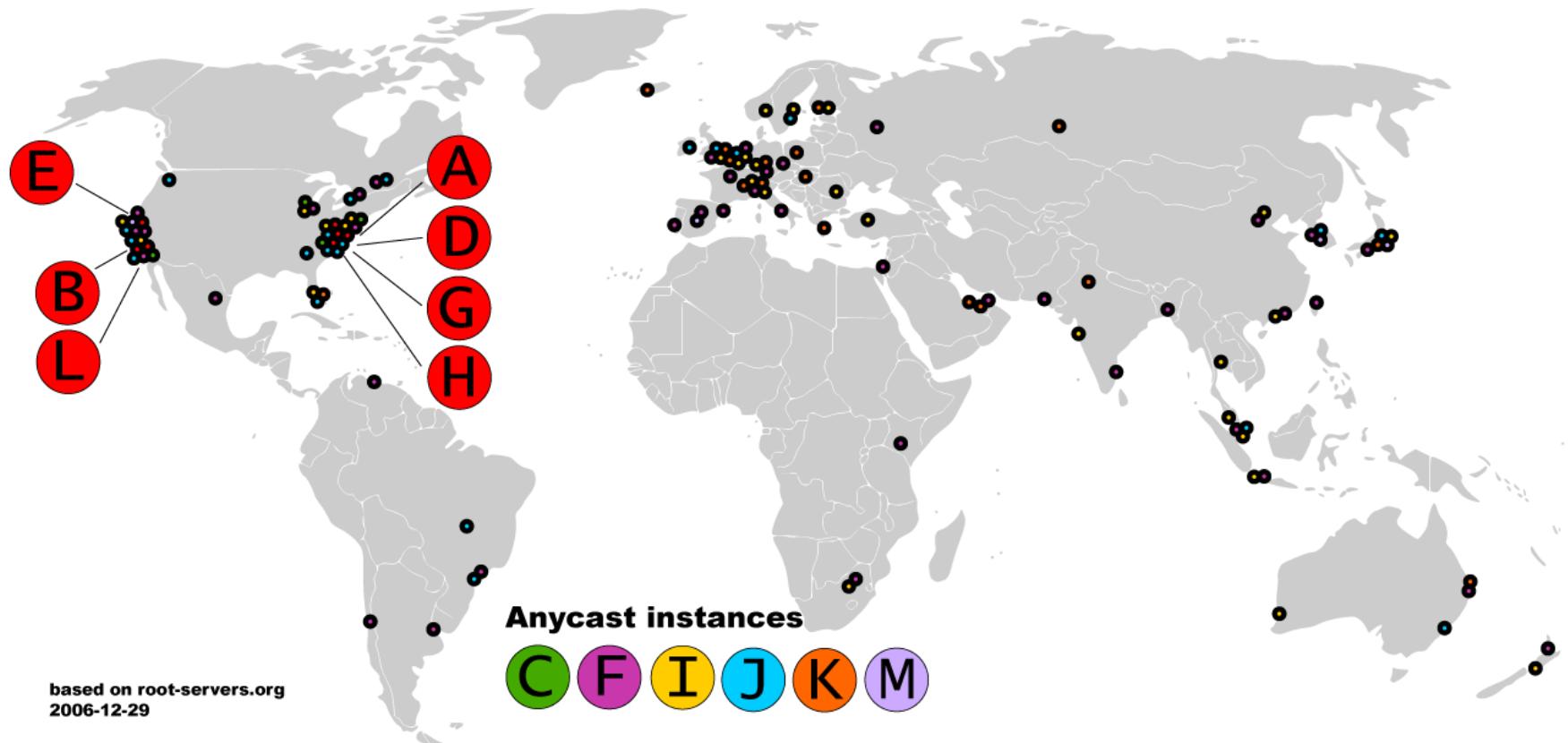
com.	172800	IN	NS	a.gtld-servers.net.
com.	172800	IN	NS	b.gtld-servers.net.
com.	172800	IN	NS	c.gtld-servers.net.

□ Az ICANN adminisztrálja

- 13 root szerver, címkék: A→M
 - Pl.: i.root-servers.net
 - 6 db ezek közül „anycastolt”, azaz globálisan számos replika létezik
- ## □ Ha név nem feloldható (lokálisan), akkor hozzájuk kell fordulni
- A gyakorlatban a legtöbb rendszer lokálisan tárolja ezt az információt (cache)

Map of the Roots

14



Lokális névszerverek

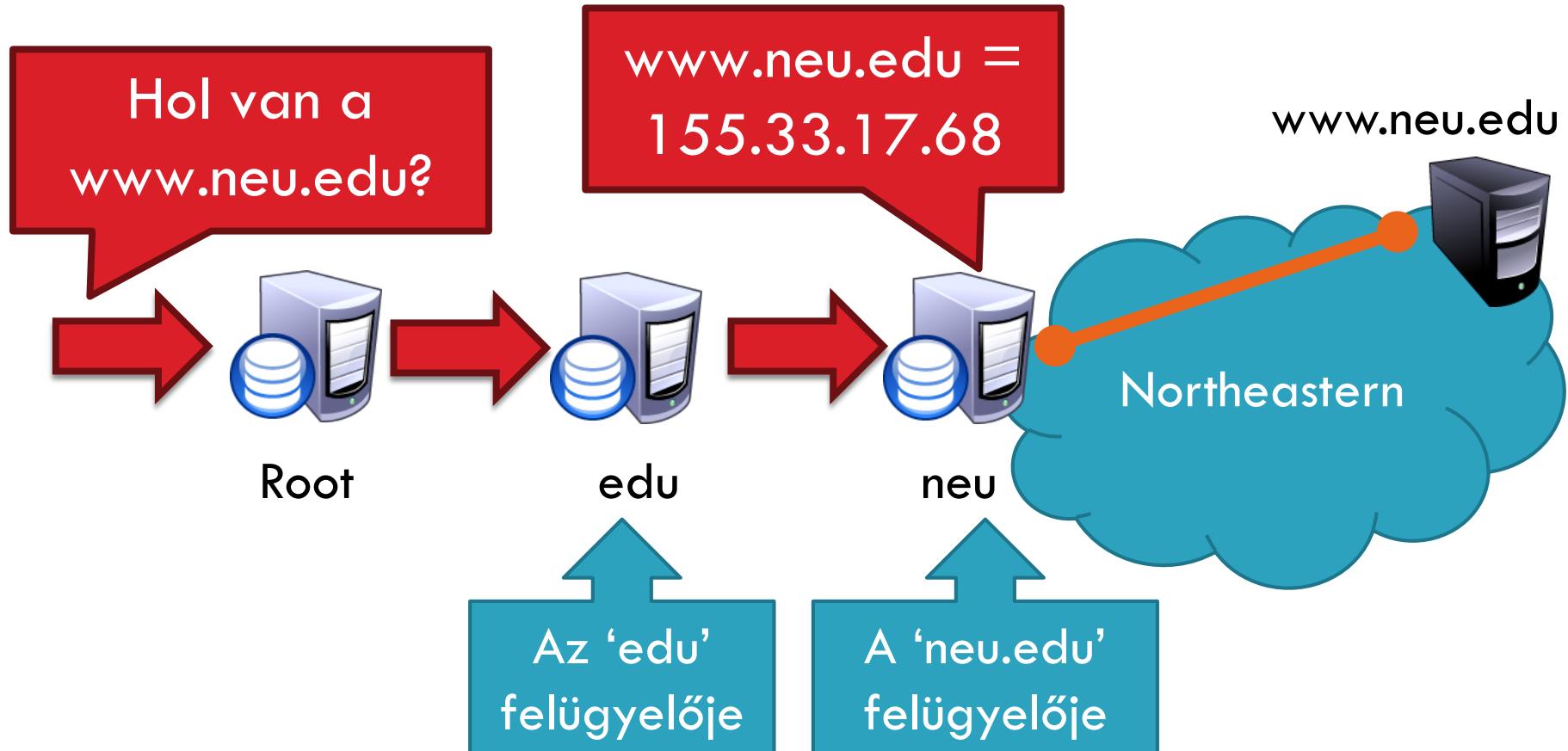
15



- minden ISP/cég rendelkezik egy lokális, default névszerverrel
- Gyakran a DHCP konfigurálja fel
- A DNS lekérdezések a lokális névszervernél kezdődnek
- Gyakran cache-be teszik a lekérdezés eredményét

Authoratív Névszerverek

16



- ❑ név → IP leképezéseket tárolja egy adott hoszthoz

Egyszerű doménnév feloldás

17

- minden hoszt ismer egy lokális DNS szervert
 - minden kérést ennek küld
- Ha a lokális DNS szerver tud válaszolni, akkor kész...
 1. A lokális szerver a felügyelő szerver az adott névhez
 2. A lokális szerver cache-ében van rekord a keresett névhez
- Különben menjünk végig a teljes hierarchián felülről lefelé egészen a keresett név felügyeleti szerveréig
 - minden lokális DNS szerver ismeri a root szervereket
 - Cache tartalma alapján bizonyos lépések átugrása, ha lehet
 - Pl. ha a root fájl tárolva van a cache-ben, akkor egyből ugorhatunk az „.edu” szerverére.

Lekérdezések

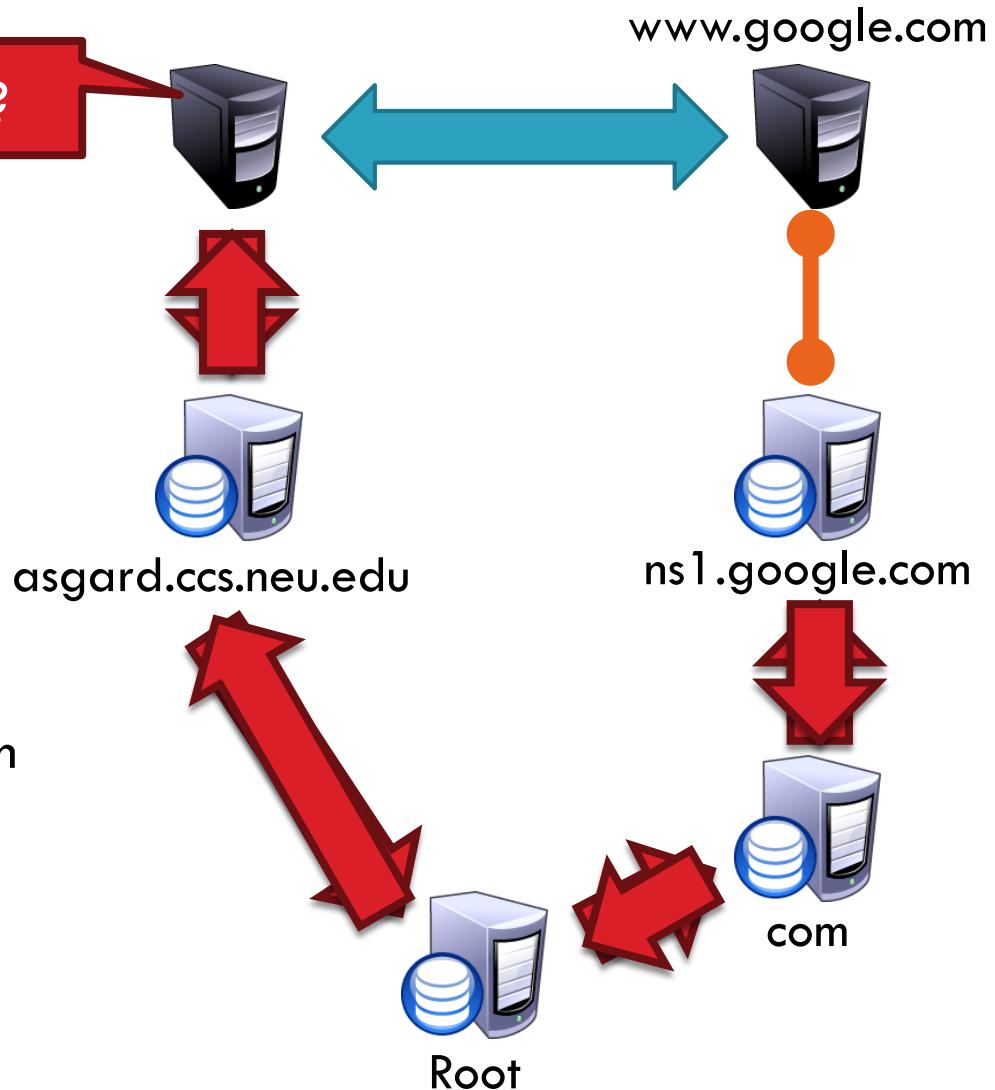
- A lekérdezésnek két fajtája van:
 - ▣ *Rekurzív lekérdezés* – Ha a névszerver végzi el a névfeloldást, és tér vissza a válasszal.
 - ▣ *Iteratív lekérdezés* – Ha a névszerver adja vissza a választ vagy legalább azt, hogy kitől kapható meg a következő válasz.
- Melyik a jobb?
 - ▣ *Rekurzív jellemzői*
 - Lehetővé teszi a szervernek a kliens terhelés kihelyezését a kezelhetőségért.
 - Lehetővé teszi a szervernek, hogy a kliensek egy csoportja felett végezzen cachelést, a jobb teljesítményért.
 - ▣ *Iteratív jellemzői*
 - Válasz után nem kell semmit tenni a kéréssel a névszervernek.
 - Könnyű magas terhelésű szervert építeni.

Rekurzív DNS lekérdezés

19

Hol van a www.google.com?

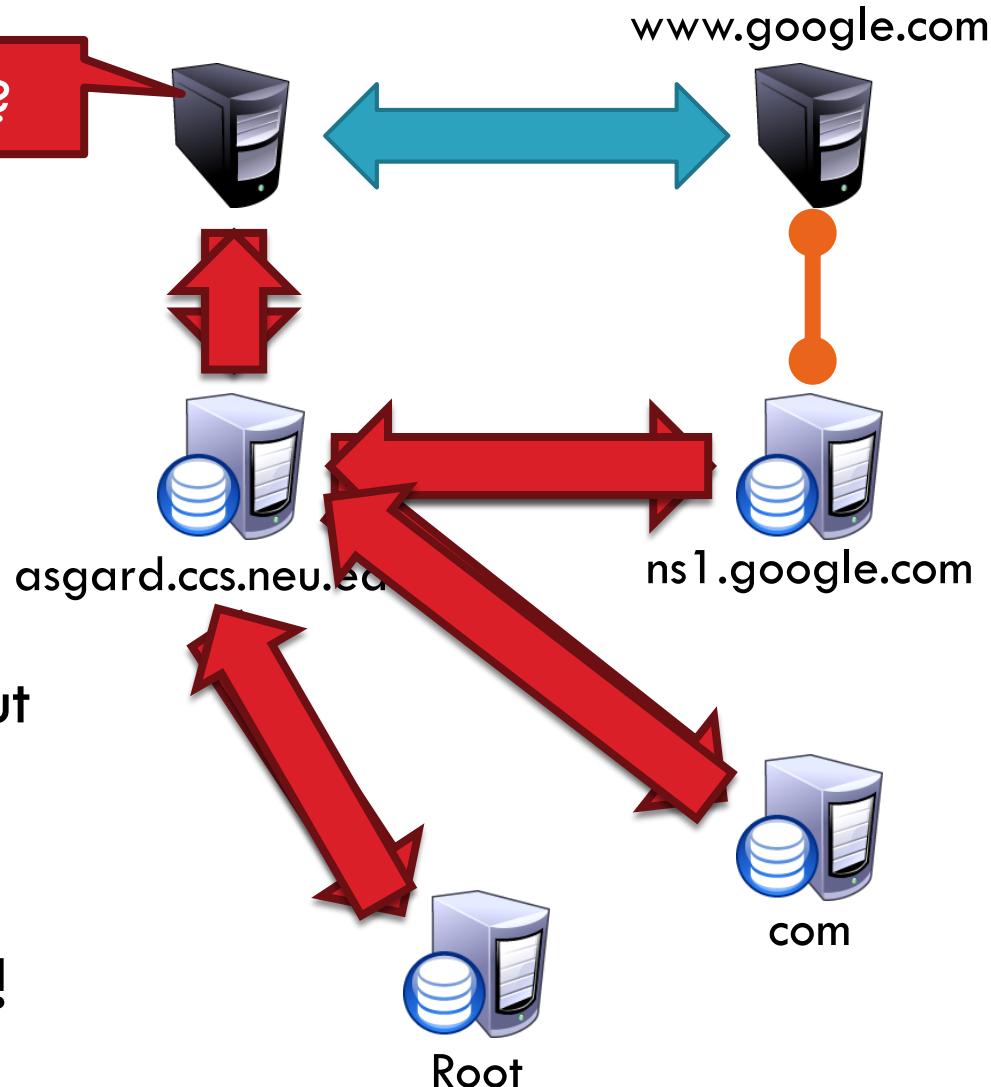
- A lokális szerver terhet rak a kérdezett névszerverre (pl. root)
- Honnan tudja a kérdezett, hogy kinek továbbítsa a választ?
 - Random ID a DNS lekérdezésben



Iteratív DNS lekérdezés

20

Hol van a www.google.com?

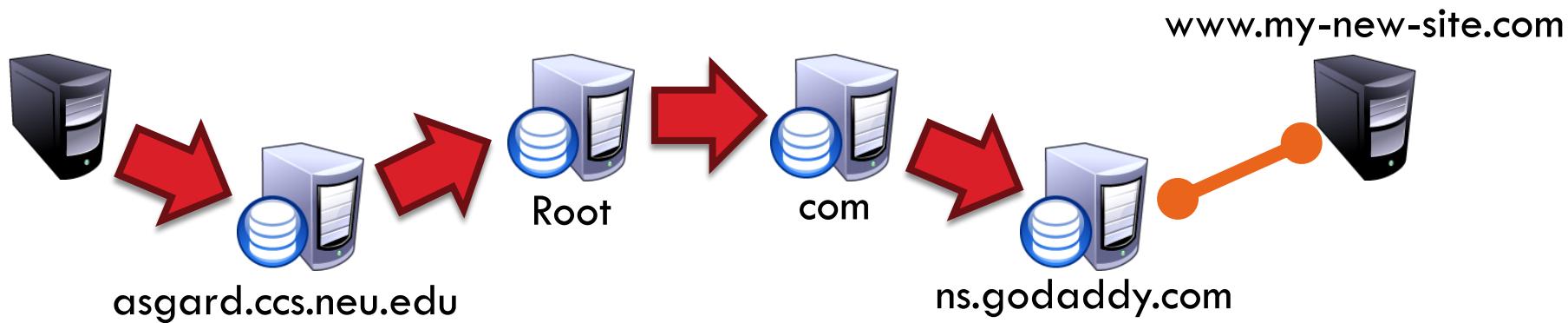


- A szerver mindenkor a következő kérdezendő névszerver adataival tér vissza
 - ▣ “I don’t know this name, but this other server might”
- Napjainkban iteratív módon működik a DNS!!!

DNS bejegyzés elterjedése

21

- Van-e a teremben olyan, aki vásárolt már domén nevet?
 - Észrevettétek-e, hogy kb. 72 óra kell ahoz, hogy elérhető legyen a bejegyzés után?
 - Ez a késés a DNS Propagáció/DNS bejegyzés elterjedése



- Miért nem sikerül ez egy új DNS név esetén?

Cachelés VS frissesség

22

- DNS elterjedés késését a cache okozza



- A zóna fájlok 1-72 órig élnek a cache-ben



DNS Erőforrás rekordok (Resource Records)

23

- A DNS lekérdezéseknek két mezőjük van
 - `name` és `type`
- Az erőforrás rekord válasz egy DNS lekérdezésre
 - Négy mezőből áll: (`name`, `value`, `type`, TTL)
 - Egy lekérdezésre adott válaszban több rekord is szerpelhet
- Mit jelent a `name` és a `value` mező?
 - Ez a lekérdezés típusától (`type`) függ

DNS lekérdezés típusok

24

- Type = A / AAAA
 - Name = domén név
 - Value = IP cím
 - A = IPv4, AAAA = IPv6

- Type = NS
 - Name = rész domén
 - Value = a rész doménhez tartozó DNS szerver neve
 - "Menj és küldd a kérésed ehhez a szerverhez"

Query Name: www.ccs.neu.edu
 Type: A

Resp. Name: www.ccs.neu.edu
 Value: 129.10.116.81

Query Name: ccs.neu.edu
 Type: NS

Resp. Name: ccs.neu.edu
 Value: 129.10.116.51

DNS lekérdezés típusok

25

- Type = CNAME
 - Name = domén név
 - Value = kanonikus név
 - Alias nevek használatához
 - CDN használja

Query Name: [foo.mysite.com](#)
 Type: CNAME

Resp. Name: [foo.mysite.com](#)
 Value: [bar.mysite.com](#)

- Type = MX
 - Name = emailben szereplő domén név
 - Value = mail szerver kanonikus neve

Query Name: [ccs.neu.edu](#)
 Type: MX

Resp. Name: [ccs.neu.edu](#)
 Value: [amber.ccs.neu.edu](#)

Fordított lekérdezés (PTR rekord)

26

- Mi a helyzet az IP→név leképezéssel?
- Külön hierarchia tárolja ezeket a leképezéseket
 - Gyökér pont: in-addr.arpa és ip6.arpa
- DNS rekord típusa (**type**): PTR
 - Name = IP cím
 - Value = domén név
- Nincs garancia arra, hogy minden IP címre működik

Query

Name: 129.10.116.51
Type: PTR

Resp.

Name: 129.10.116.51
Value: ccs.neu.edu

DNS as Indirection Service

27

- DNS számos lehetőséget biztosít
 - Nem csak a gépekre való hivatkozást könnyíti meg!
- Egy gép IP címének lecserélése is triviális
 - Pl. a web szervert átköltöztetjük egy új hosztra
 - Csak a DNS rekord bejegyzést kell megváltoztatni!

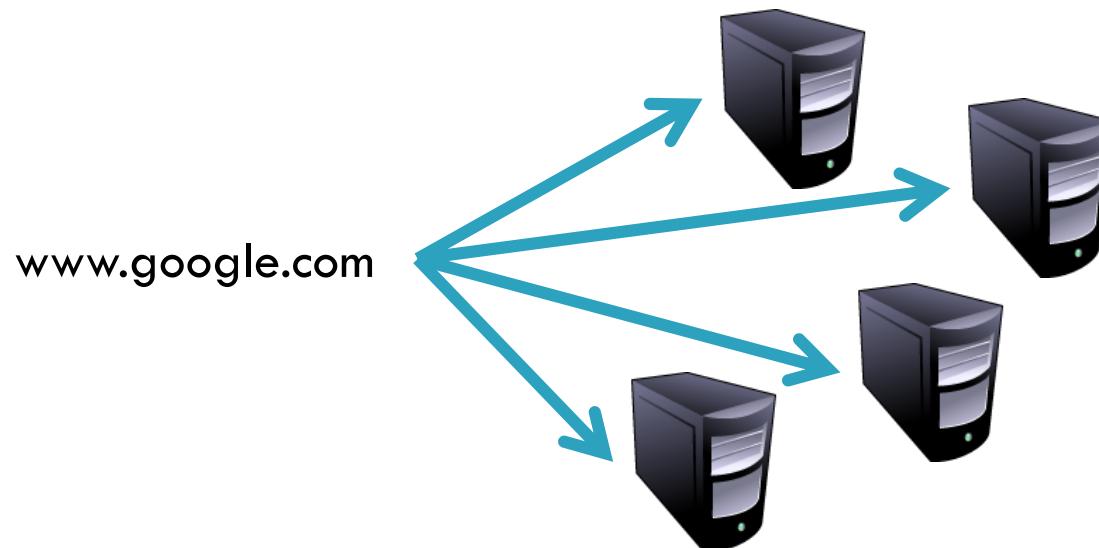
Aliasing/Kanonikus nevek és Load Balancing/Terhelés elosztás

28

- Egy gépnak számos alias neve lehet

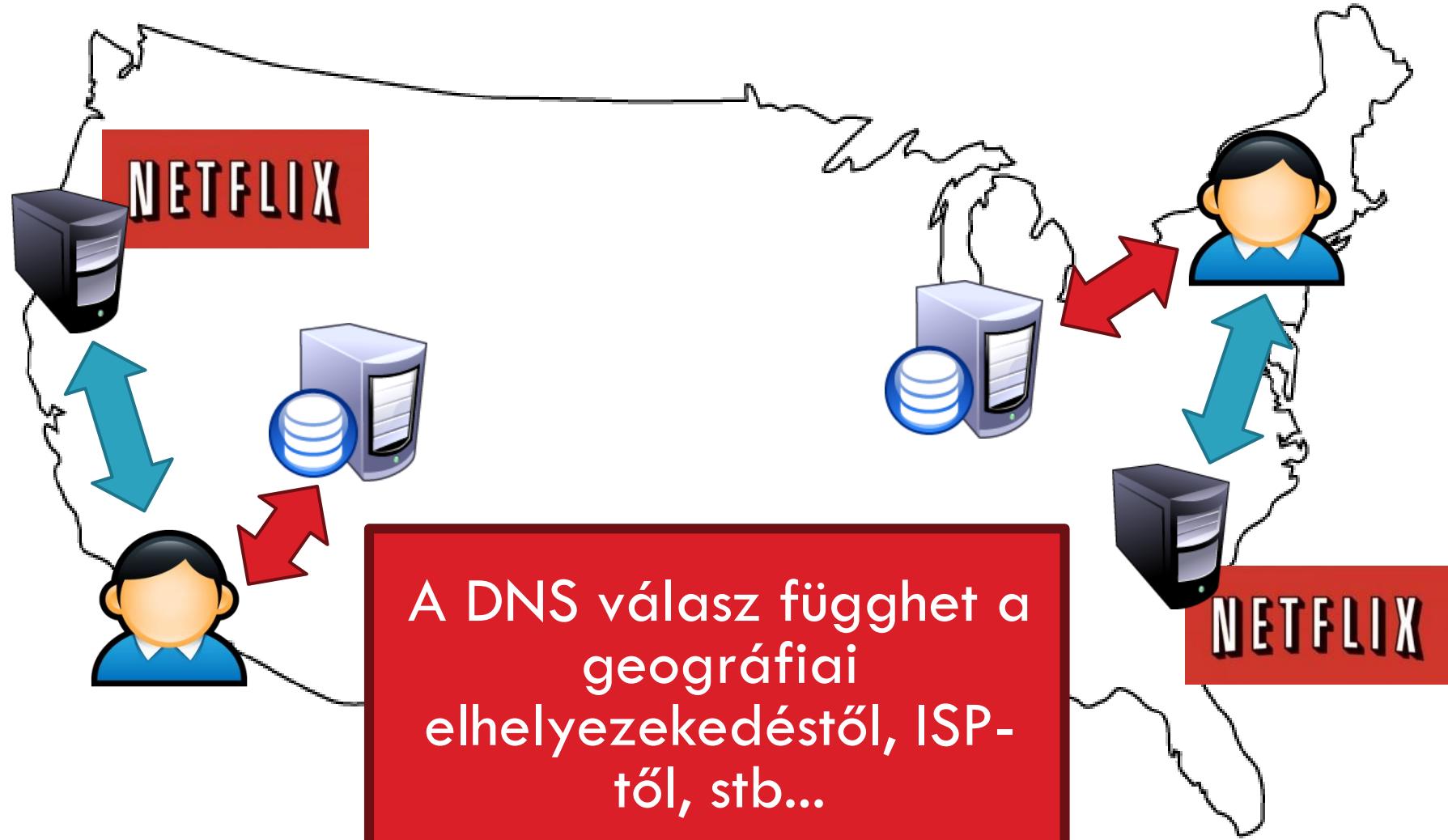


- Egy domén névhez számos IP cím tartozhat



Content Delivery Networks

29



A DNS fontossága

30

- DNS nélkül...
 - Hogyan találunk meg egy weboldalt?
- Példa: a mailszervered azonosít
 - Email címet adunk meg weboldalakra való feliratkozásnál
 - Mi van, ha valaki eltéríti a DNS bejegyzést a mailszerveredhez?
- DNS a bizalom forrása a weben
 - Amikor a felhasználó begépeli a www.bankofamerica.com címet, azt várja, hogy a bankja honlapja jelenjen meg.
 - Mi van, ha a DNS rekordot meghackelték?

Denial Of Service (DoS)

31

- DNS szerverek túlterhelése lekérdezésekkel, amíg össze nem omlanak
- 2002 Október: masszív DDoS támadás a root szerverek ellen
 - Mi volt a hatása?
 - ... a felhasználók észre se vették
 - Root zónák mindenhol cache-elve vannak
- Célzottabb támadás hatékonyabb lenne
 - Lokális DNS szerver → elérhetetlen DNS
 - Authoritative szerver → elérhetetlen domén

DNS Hijacking (eltérítés)

32

- OS vagy browser megfertőzése (virus/trojan)
 - Pl. Számos trójai megváltoztatja a bejegyzéseket a /etc/hosts fájlnan
 - *.bankofamerica.com → evilbank.com

□ Man-in-the-middle



- Válasz hamisítás (spoofing)
 - Kérések lehallgatása
 - Válaszok megversenyeztetése

D

Hol van a
bankofamerica.com?

123.45.67.89

33

Honnan tudjuk, hogy a név → IP
leképezés helyes?



Hol van a
bankofamerica.com?

66.66.66.93

ank of America



123.45.67.89



dns.evil.com



66.66.66.93

D
Hol van a

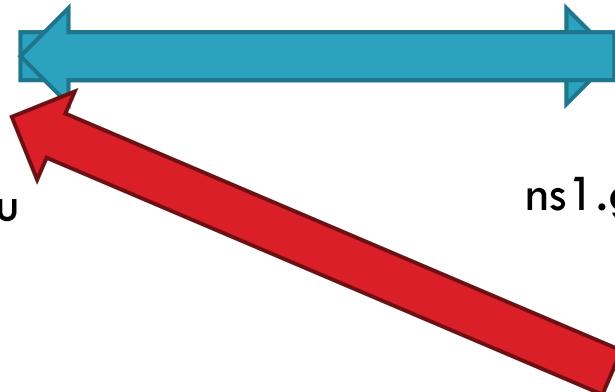
Hol van a
bankofamerica.com?

34



spoofer

www.google.com =
74.125.131.26



- Amíg TTL lejár, a BofA összes kérése, amit a dns.neu.edu-nak küld, hamis/fertőzöt vissza

- Sokkal rosszabb, mint a spoofing/man-in-the-middle
 - Egy teljes ISP-t érinthet!

bankofamerica.com =
66.66.66.92

Hogyan éri el a támadó a fertőzött bejegyzés tárolását?

35

- 1. Azt mondjuk a feloldónak, hogy az áldozathoz tartozó NS a támadó IP-jén érhető el
 - ▣ Kiváltó lekérdezés: subdomain.attacker.example IN A
 - ▣ Támadó válasza:
- Válasz: (nincs válasz a lekérdezésre), de
 - ▣ Authority Section: attacker.example. 3600 IN NS ns.target.example.
 - ▣ Additional Section: ns.target.example. IN A w...

A támadó azt mondja, „hogy a doménem autoratív szervere a ns.target.example és mellesleg itt van az IP-je”...

Hogyan éri el a támadó a fertőzött bejegyzés tárolását?

36

- 2. NS rekord átirányítása a támadó doménébe
 - Kiváltó lekérdezés: subdomain.attacker.example IN A
 - Válasz: (nincs válasz)
 - Authority section:
 - Target.example. 3600 IN NS ns.attacker.example.
 - Additional section:
 - Ns.attacker.example. IN A w.x.y.z



A támadó nem releváns információt szűr be, melyet a szerver el fog tárolni a cacheben...

Megoldás: DNSSEC

37

- Kritikus rekordokat kriptografikus aláírással látjuk el
 - A feloldó ellenőrzi az aláírást
 - Két új erőforrástípus
 - Type = DNSKEY
 - Name = Zóna domén
 - Value = A zóna publikus kulcsa
 - Type = RRSIG
 - Name = (type, name) páros, pl. a lekérdezés maga
 - Value = A lekérdezés eredményének kriptografikus aláírása
 - Elterjedése
 - Root szerverek 2010 Július óta
 - Verisign lehetővé tette a .com és .net doménekben 2011 Január
- Bizalmi hierarchiát hoz
létre a zónák között
- Megelőzi az eltérítést
és a hamisítást

DNSSEC 2

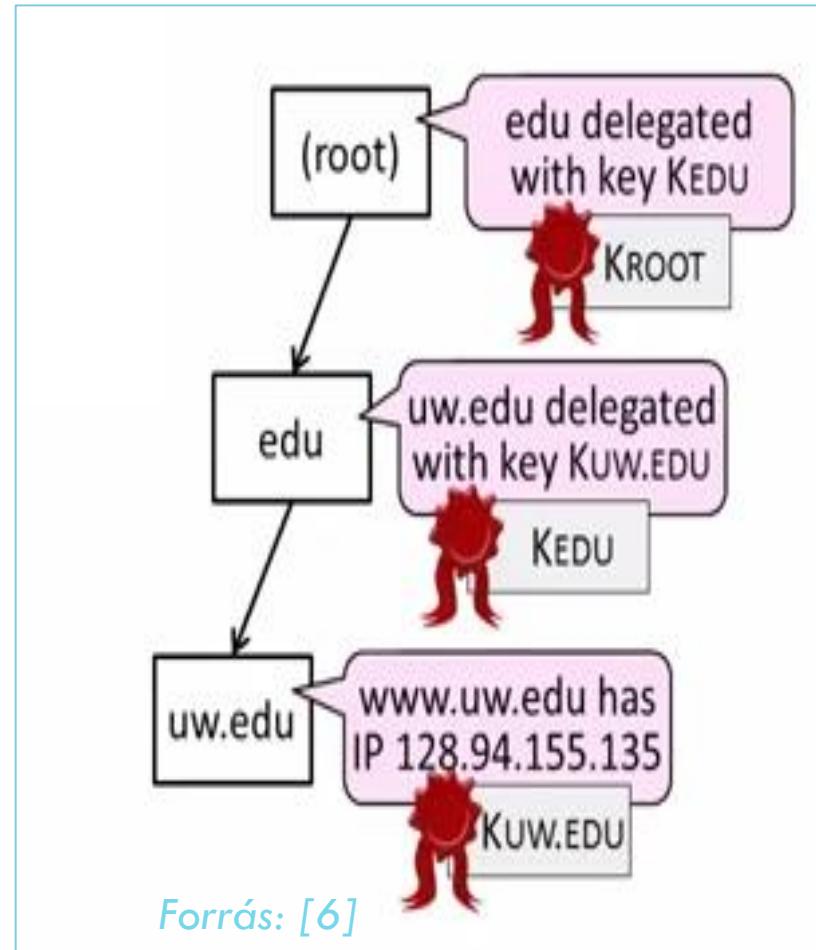
38

- Új rekordtípusokat vezettek be:
 - RRSIG a rekordok digitális aláírására.
 - DNSKEY publikus kulcsok használatához a validáció során.
 - DS publikus kulcsok használatához a delegációhoz.
 - NSEC/NSEC3 a létezés hitelesített visszautasítása.
- Első változat 1997-ben jelent meg. 2005-ben újraírták.
- Ez a fejlesztés viszont a kliensek és szerverek frissítését vonja maga után. (2010 - „Root” szerverek frissítése.)
- A kliensek a szokott módon kérdezik le a DNS-t, és később ellenőrzik a válasz hitelességét.
- A *Trust Anchor* a publikus kulcsok gyökere. (DNS kliens konfigurációjának a része)
- A biztonság leköveti a DNS hierarchiát.

DNSSEC 3

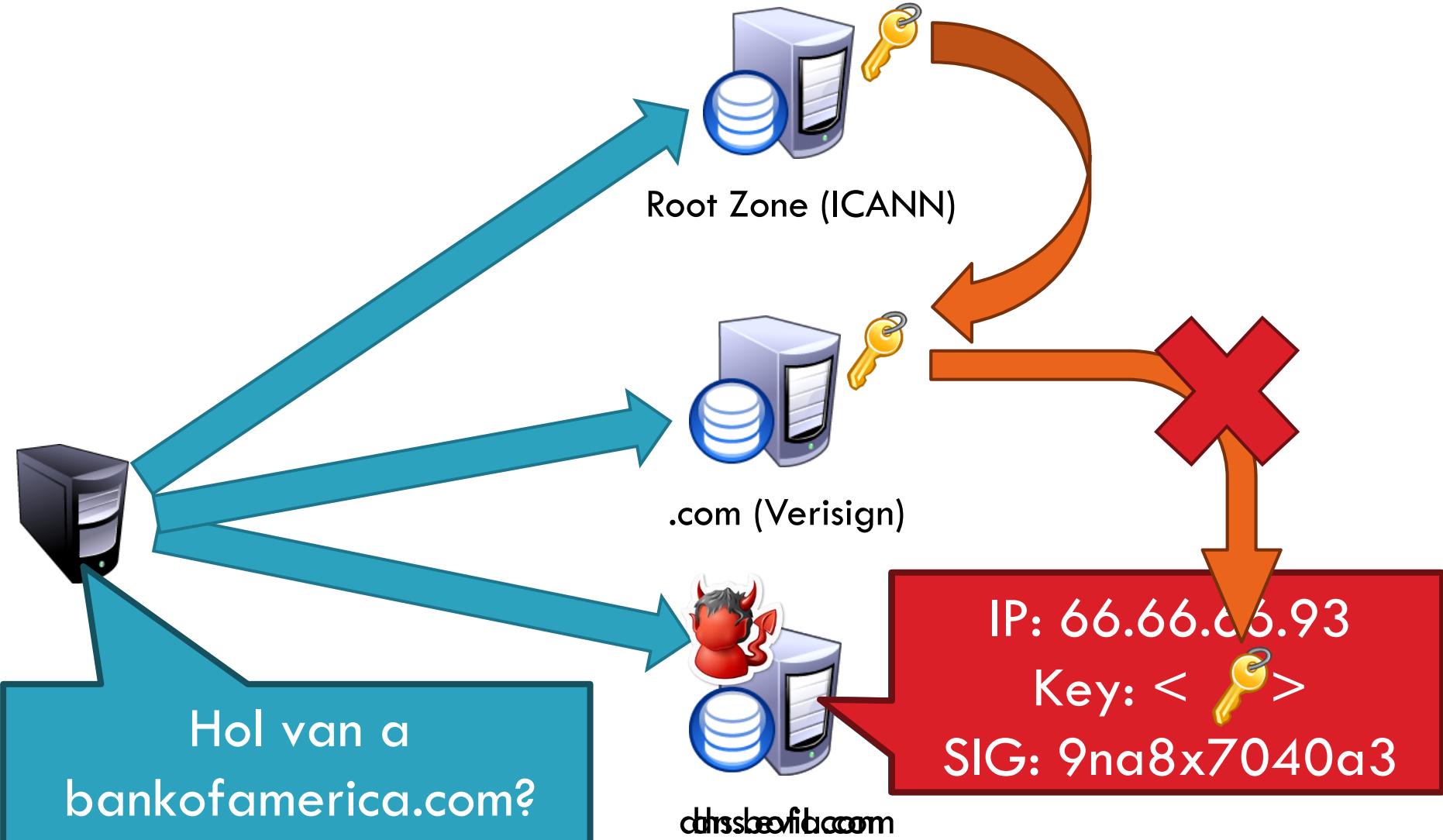
39

- Példa a www.uw.edu lekérdezése egy klienssel.
- Kliens hitelesíti a választ:
 1. KROOT egy *Trust Anchor*.
 2. KROOT-ot használja a KEDU ellenőrzésére.
 3. KEDU-t használja a KUW.EDU ellenőrzésére.
 4. KUW.EDU-t használja a IP cím ellenőrzésére.



DNSSEC Bizalmi hierarchia

40



Does DNSSEC Solve all our problems?

41

- No.
- DNS still vulnerable to reflection attacks + injected responses

DNS Reflection

42

- Very big incident in 2012
 - ▣ (<http://blog.cloudflare.com/65gbps-ddos-no-problem/>)
 - ▣ 65 Gbps DDoS
 - ▣ Would need to compromise 65,000 machines each with 1 Mbps uplink
 - How was this attack possible?
- Use DNS reflection to amplify a Botnet attack.
- Key weak link: Open DNS resolvers will answer queries for anyone <http://openresolverproject.org/>

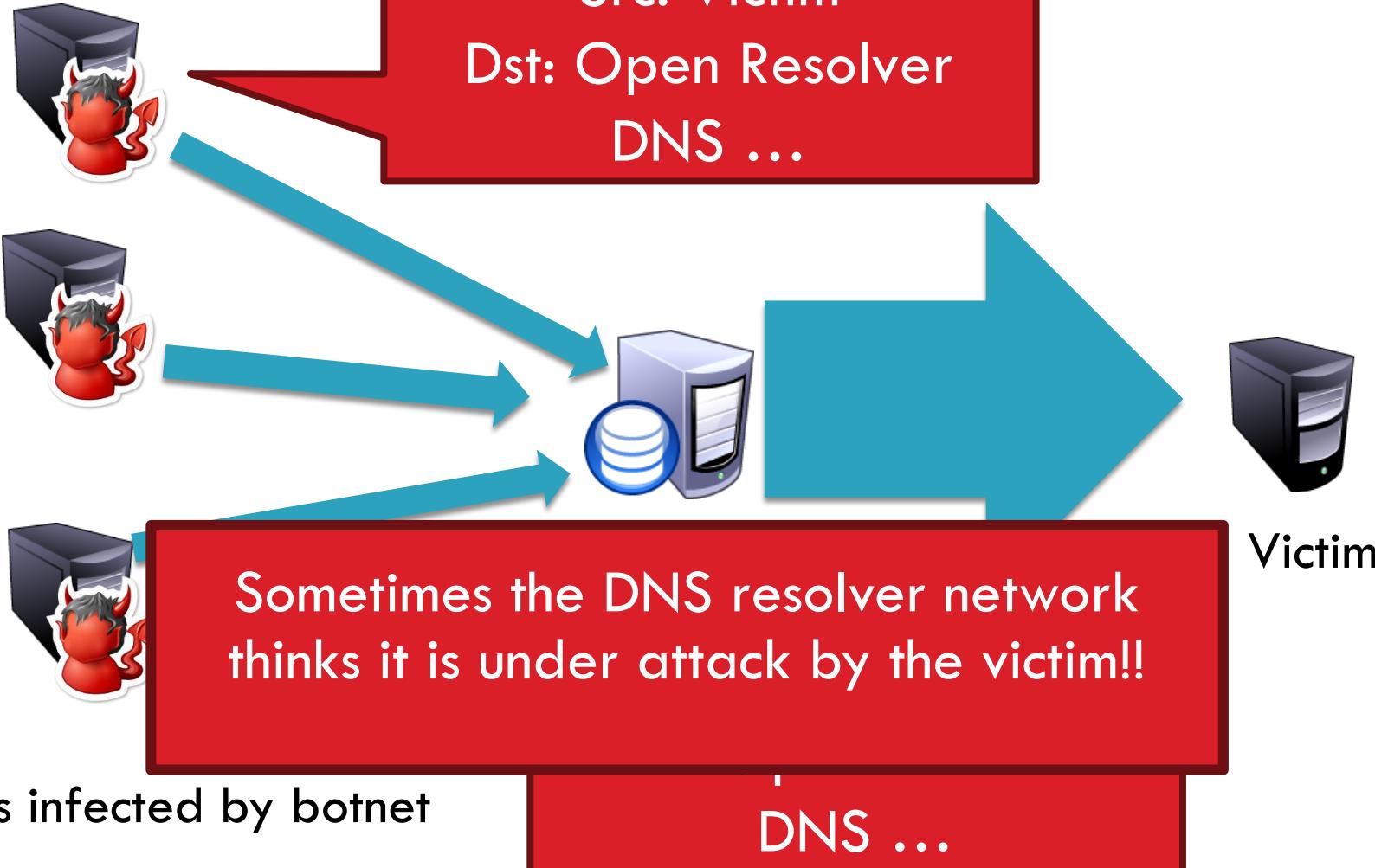
So how does this work?

43

- Remember: DNS is UDP
- No handshaking between endpoints
- I can send a DNS query with a forged IP address and the response will go to that IP address
 - ▣ **Secret sauce:** a small request that can elicit a large response
 - ▣ E.g., query for zone files, or DNSSEC records (both large record types).
- Botnet hosts spoof DNS queries with victim's IP address as source
 - ▣ Resolver responds by sending massive volumes of data to the victim

DNS amplification illustrated

44



HTTP

Web and HTTP

2-46

First, a review...

- **web page** consists of **objects**
- object can be HTML file, JPEG image, Java applet, audio file,...
- web page consists of **base HTML-file** which includes **several referenced objects**
- each object is addressable by a **URL**, e.g.,

www.someschool.edu/someDept/pic.gif

host name

path name

HTTP overview

2-47

HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server model
 - **client:** browser that requests, receives, (using HTTP protocol) and "displays" Web objects
 - **server:** Web server sends (using HTTP protocol) objects in response to requests



HTTP overview (continued)

2-48

uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

HTTP is “stateless” (in theory...)

- server maintains no information about past client requests

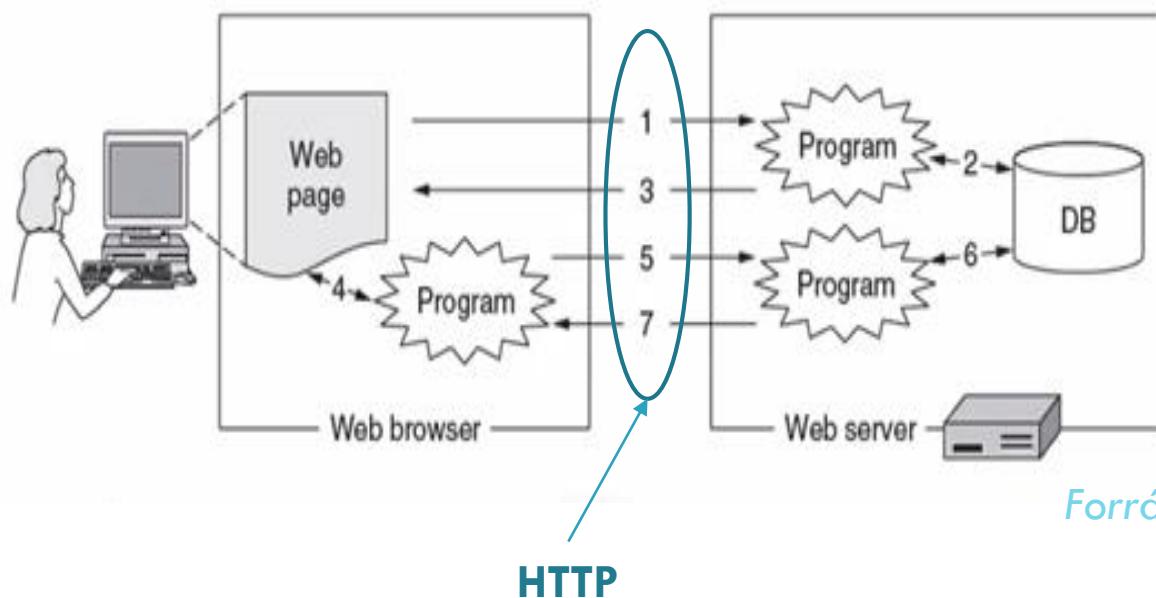
aside

protocols that maintain “state” are complex!

- ❖ past history (state) must be maintained
- ❖ if server/client crashes, their views of “state” may be inconsistent, must be reconciled

Statikus és dinamikus weboldalak

- A *statikus weboldal* tartalma nem változik csak manuális átszerkesztéssel.
- A *dinamikus weboldal* valamilyen kód végrehajtásaként keletkezik, mint például: *javascript*, *PHP*, vagy minden kettő egyszerre.



Forrás: [4]

HTTP connections

2-50

non-persistent HTTP

- at most one object sent over TCP connection
 - connection then closed
- downloading multiple objects required multiple connections

persistent HTTP

- multiple objects can be sent over single TCP connection between client, server

Example Web Page

51

page.html

Harry Potter Movies

As you all know,
the new HP book
will be out in June
and then there will
be a new movie
shortly after that...



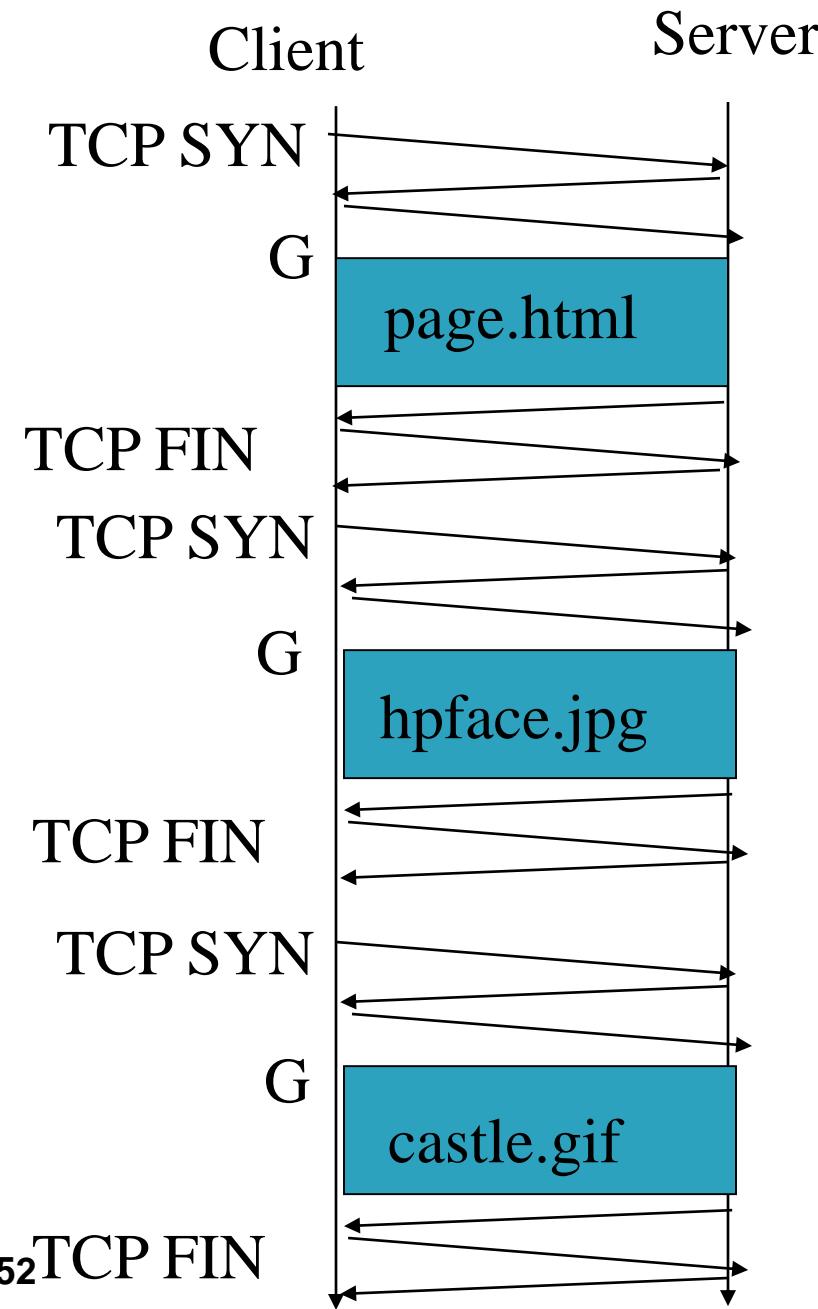
hpface.jpg



castle.gif

“Harry Potter and
the Bathtub Ring”

Non-Persistent HTTP



The “classic” approach in HTTP/1.0 is to use one HTTP request per TCP connection, serially.

Client

Server

TCP SYN

G

page.html

TCP FIN

F

C
S

G

hpface.jpg

S

S

F

C

S
G

castle.gif

S

S

F

Concurrent (parallel) TCP connections can be used to make things faster.

Persistent HTTP

2-54

non-persistent HTTP issues:

- requires 2 RTTs per object
- OS overhead for each TCP connection
- browsers often open parallel TCP connections to fetch referenced objects

persistent HTTP:

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

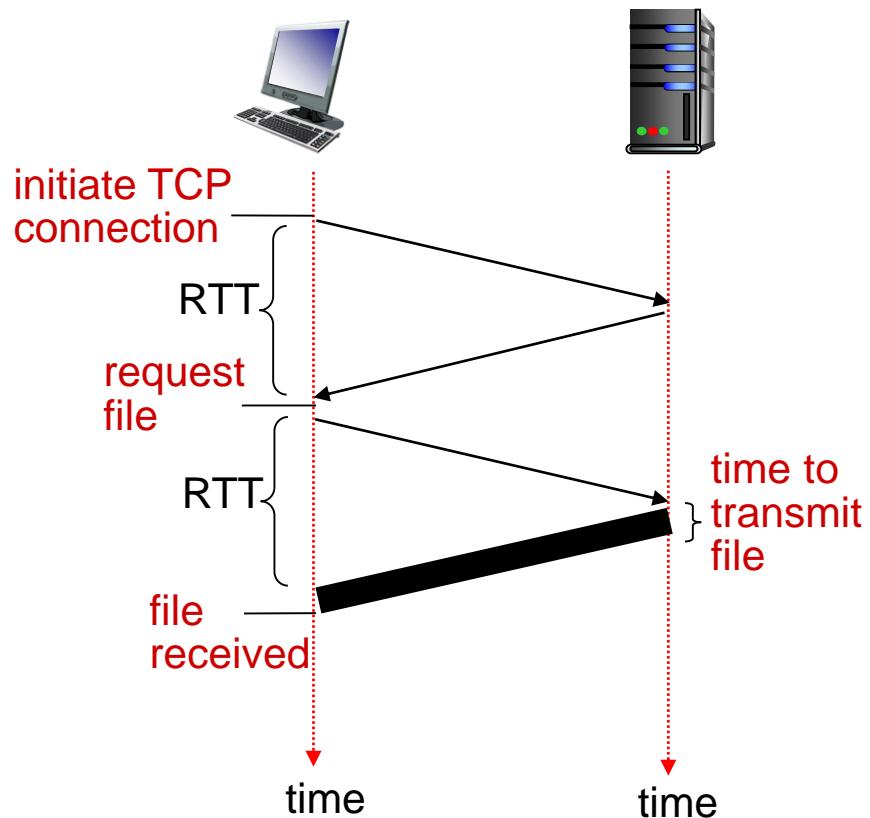
Non-persistent HTTP: response time

2-55

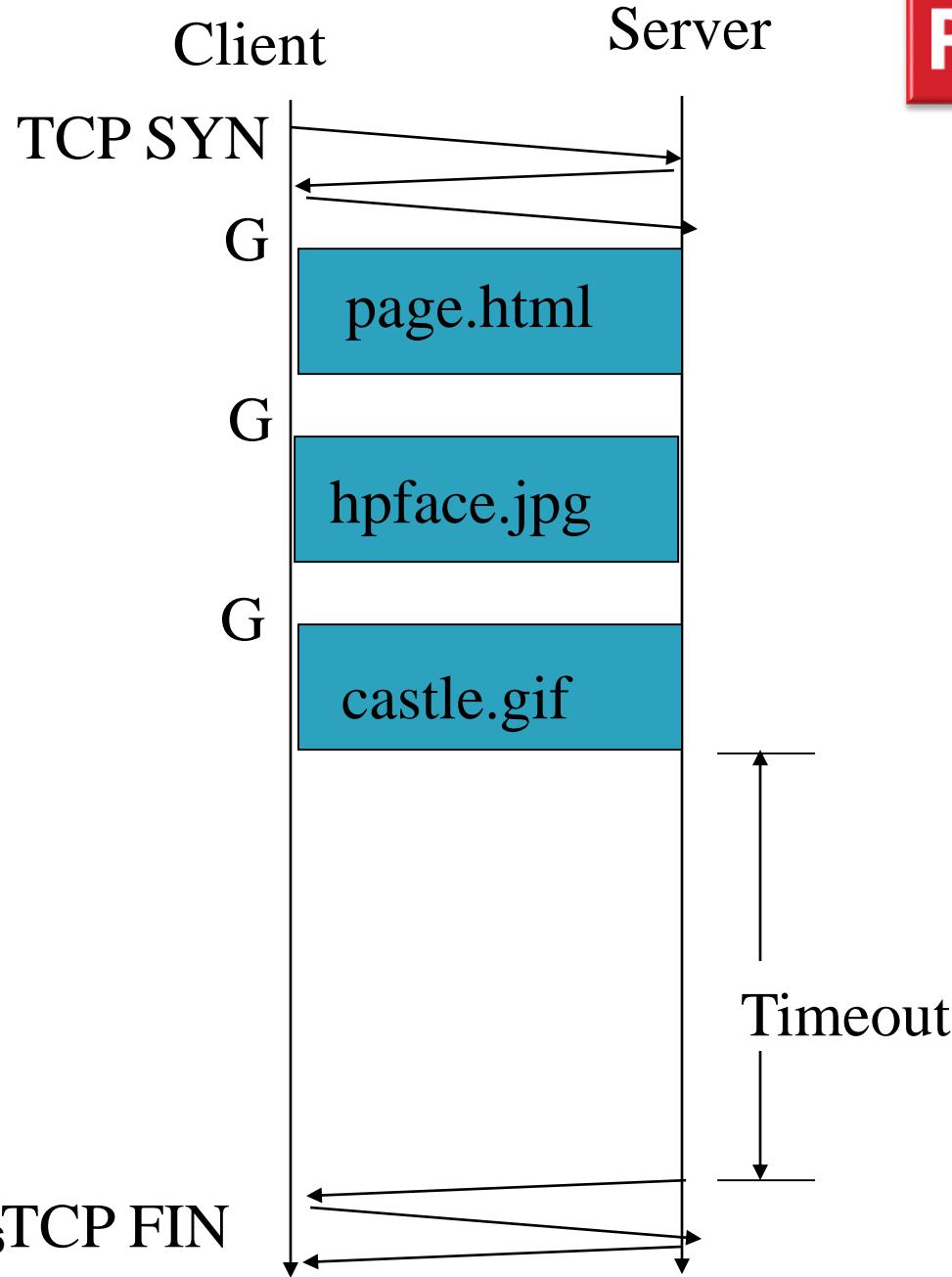
RTT: time for a packet to travel from client to server and back

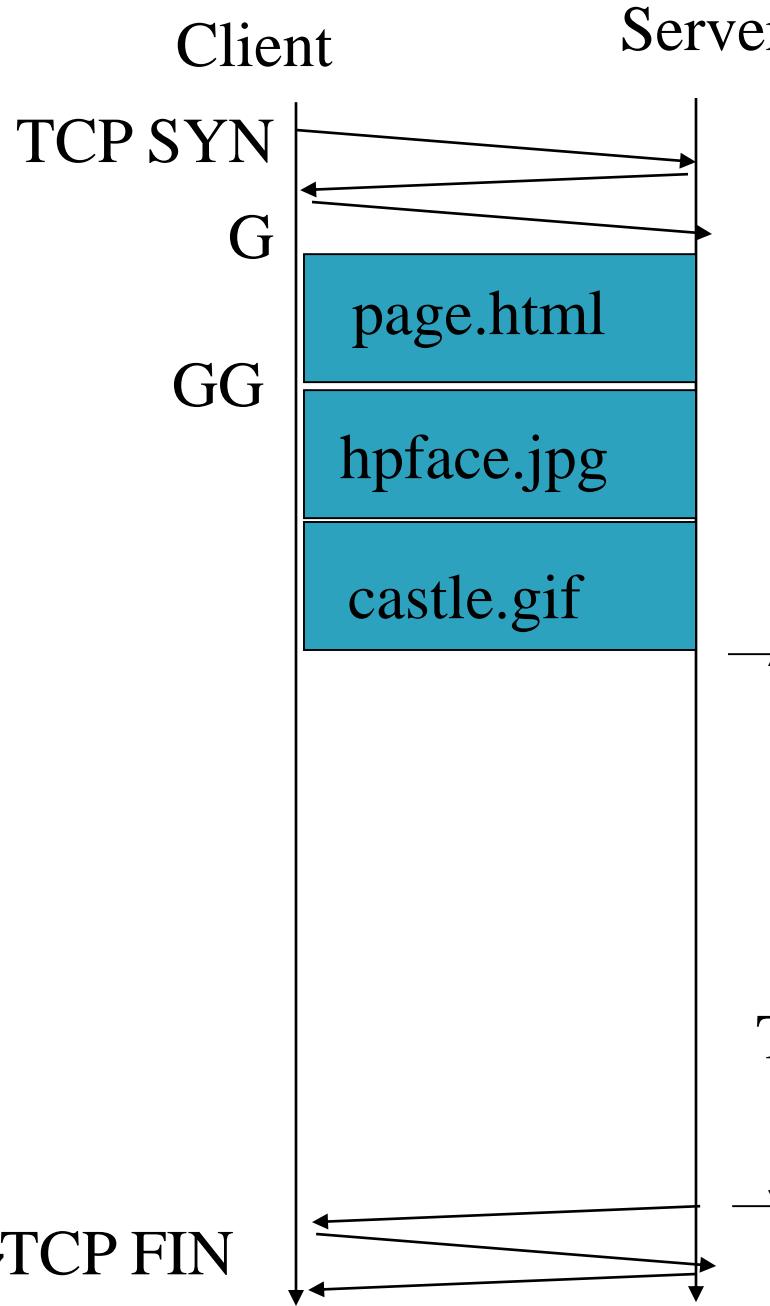
HTTP response time:

- one RTT to initiate TCP connection
- one RTT for HTTP request and first few bytes of HTTP response to return
 - This assumes HTTP GET piggy backed on the ACK
- file transmission time
- non-persistent HTTP response time =
$$2\text{RTT} + \text{file transmission time}$$



Persistent HTTP





The “pipelining” feature in HTTP/1.1 allows requests to be issued asynchronously on a persistent connection. Requests must be processed in proper order. Can do clever packaging.

Outline

- HTTP Connection Basics
- HTTP Protocol
- Cookies, keeping state + tracking

HTTP request message

2-59

- two types of HTTP messages: **request, response**
- **HTTP request message:**
 - ASCII (human-readable format)

request line
(GET, POST,
HEAD commands)

```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

header
lines

carriage return,
line feed at start
of line indicates
end of header lines

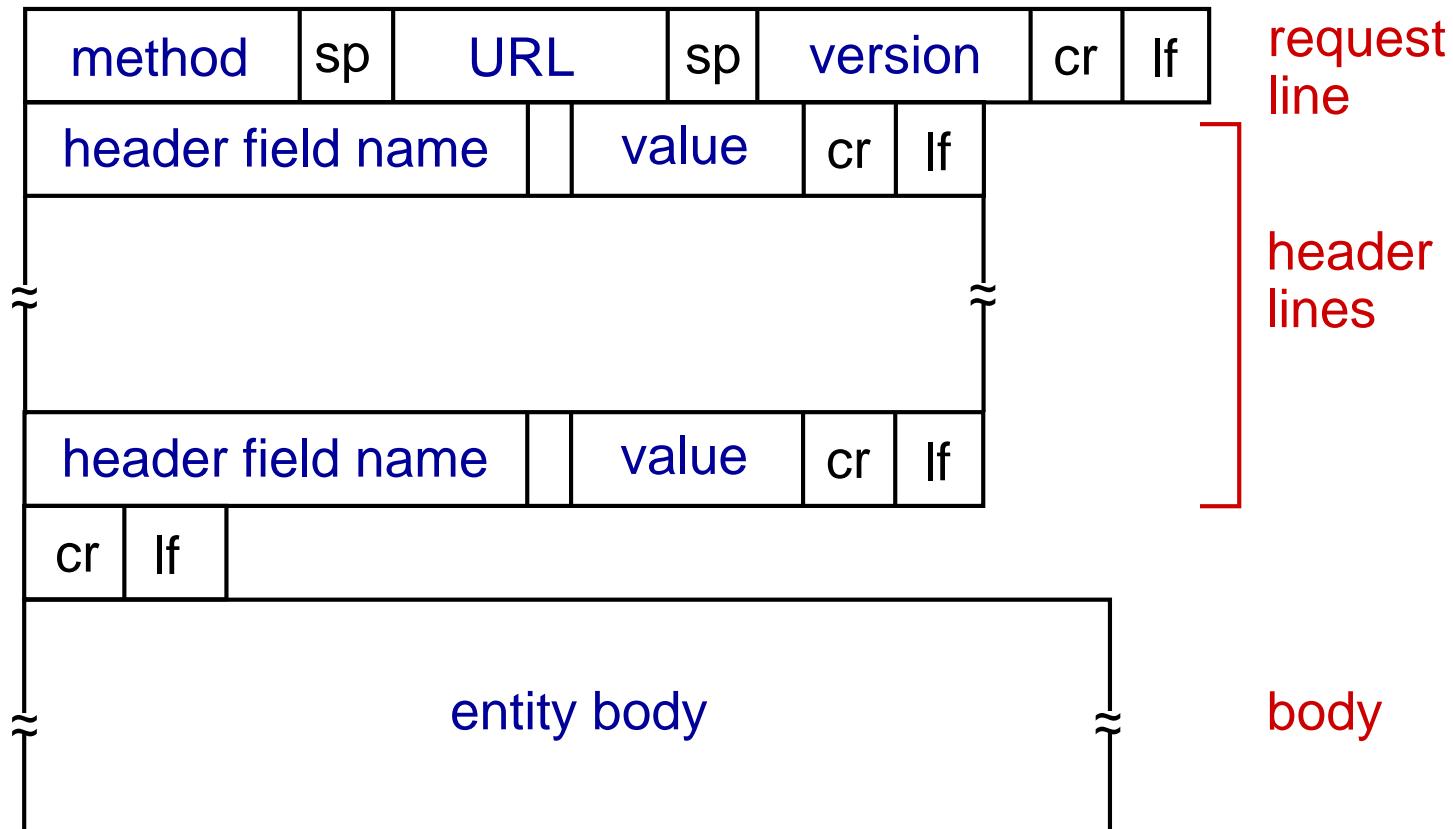
Application Layer

carriage return character

line-feed character

HTTP request message: general format

2-60



Uploading form input

2-61

POST method:

- web page often includes form input
- input is uploaded to server in entity body

URL method:

- uses GET method
- input is uploaded in URL field of request line:
`www.somesite.com/animalsearch?monkeys&banana`

Method types

2-62

HTTP/1.0:

- GET
- POST
- HEAD
- asks server to leave requested object out of response

HTTP/1.1:

- GET, POST, HEAD
- PUT
- uploads file in entity body to path specified in URL field
- DELETE
- deletes file specified in the URL field

HTTP response message

2-63

status line

(protocol

status code

status phrase)

header
lines

data, e.g.,
requested
HTML file

```
HTTP/1.1 200 OK\r\nDate: Sun, 26 Sep 2010 20:09:20 GMT\r\nServer: Apache/2.0.52 (CentOS)\r\nLast-Modified: Tue, 30 Oct 2007 17:00:02  
GMT\r\nETag: "17dc6-a5c-bf716880"\r\nAccept-Ranges: bytes\r\nContent-Length: 2652\r\nKeep-Alive: timeout=10, max=100\r\nConnection: Keep-Alive\r\nContent-Type: text/html; charset=ISO-8859-  
1\r\n\r\n
```

data data data data data ...

Application Layer

HTTP response status codes

2-64

- ❖ status code appears in 1st line in server-to-client response message.
- ❖ some sample codes:

200 OK

- ❑ request succeeded, requested object later in this msg

301 Moved Permanently

- ❑ requested object moved, new location specified later in this msg
(Location:)

400 Bad Request

- ❑ request msg not understood by server

404 Not Found

- ❑ requested document not found on this server

505 HTTP Version Not Supported

Trying out HTTP (client side) for yourself

2-65

1. Telnet to your favorite Web server:

telnet cis.poly.edu 80

opens TCP connection to port 80
(default HTTP server port) at cis.poly.edu.
anything typed in sent
to port 80 at cis.poly.edu

2. type in a GET HTTP request:

**GET /~ross/ HTTP/1.1
Host: cis.poly.edu**

by typing this in (hit carriage
return twice), you send
this minimal (but complete)
GET request to HTTP server

3. look at response message sent by HTTP server!

(or use Wireshark to look at captured HTTP request/response)

- HTTP Connection Basics
- HTTP Protocol
- Cookies, keeping state + tracking

User-server state: cookies

2-67

many Web sites use cookies

four components:

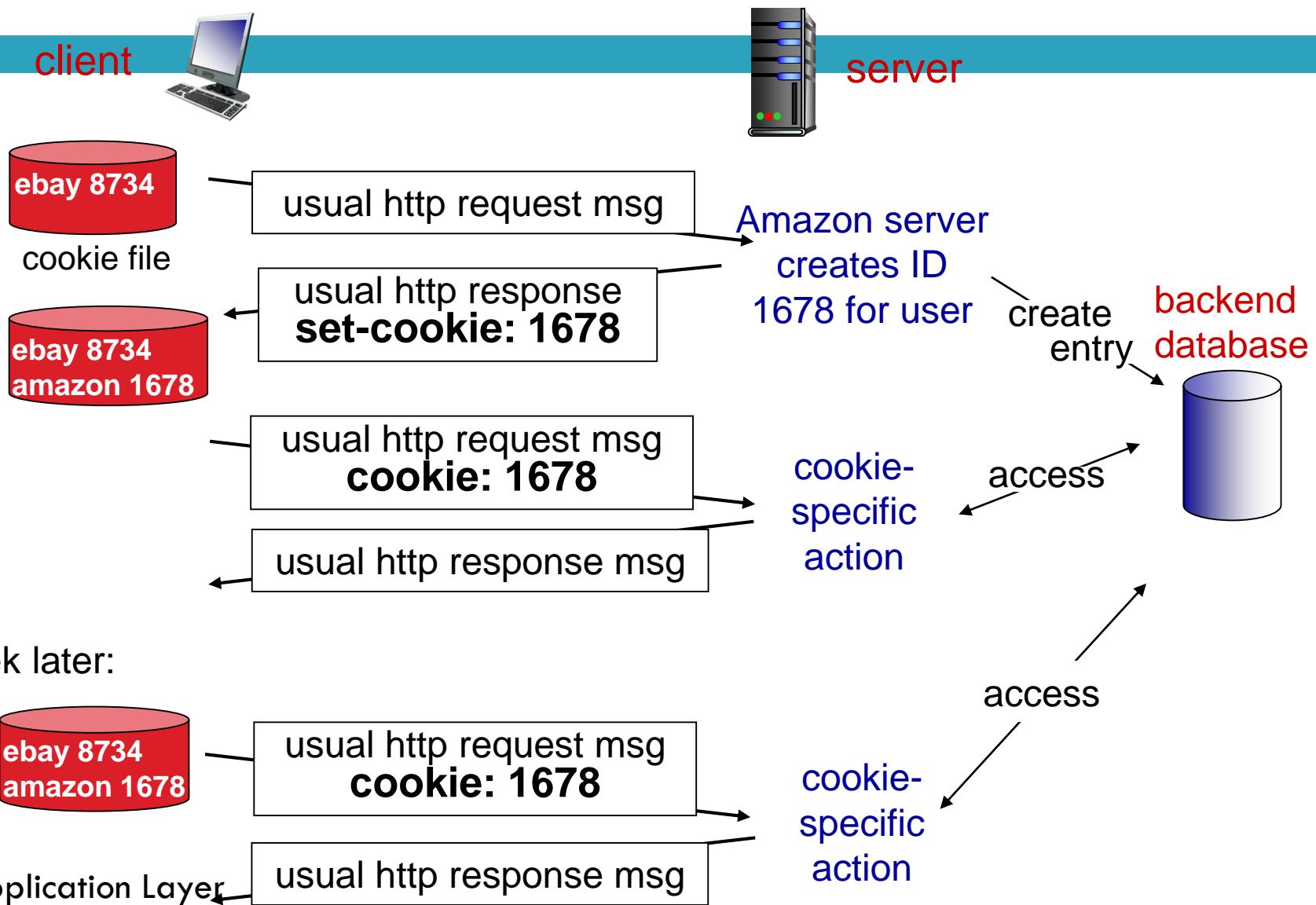
- 1) cookie header line of HTTP response message
- 2) cookie header line in next HTTP request message
- 3) cookie file kept on user's host, managed by user's browser
- 4) back-end database at Web site

example:

- Susan always access Internet from PC
- visits specific e-commerce site for first time
- when initial HTTP requests arrives at site, site creates:
 - unique ID
 - entry in backend database for ID

Cookies: keeping “state” (cont.)

2-68



Cookies (continued)

2-69

*what cookies can be used
for:*

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

aside

cookies and privacy:

- ❖ cookies permit sites to learn a lot about you
- ❖ you may supply name and e-mail to sites

how to keep “state”:

- ❖ protocol endpoints: maintain state at sender/receiver over multiple transactions
- ❖ cookies: http messages carry state

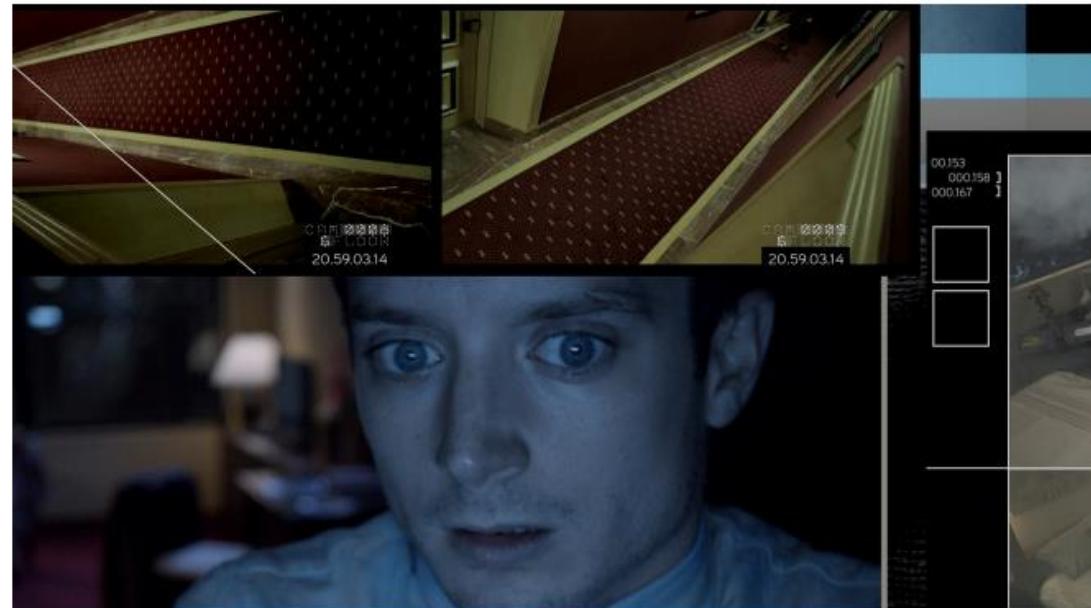
Cookies + Third Parties

70

- Example page (from Wired.com)

Elijah Wood's New Movie Is a Prophetic Thriller About Celebrity Hacking

CUTTER 10.02.14 | 6:30 AM | PERMALINK



Elijah Wood in *Open Windows*. courtesy Cinedigm

How it works

71

And it's not just Facebook!



Wi

GET article.html



GET sharebutton.gif
Cookie: FBCOOKIE

Facebook now knows you visited this Wired article.
Works for all pages where 'like'/'share' button is embedded!

CDN

Content in today's Internet

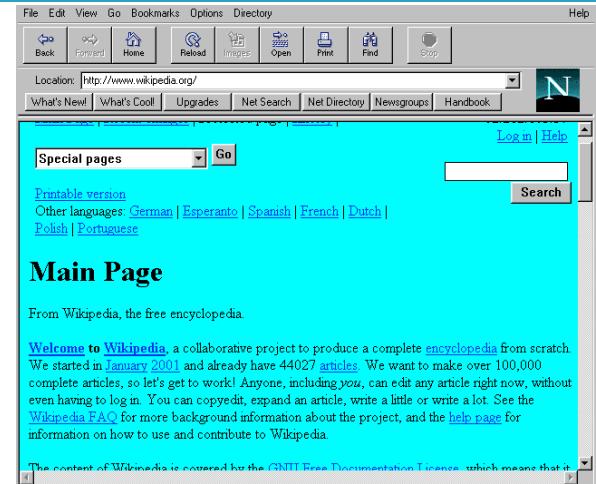
73

- Most flows are HTTP
 - Web is at least 52% of traffic
 - Median object size is 2.7K, average is 85K (as of 2007)
- HTTP uses TCP, so it will
 - Be ACK clocked
 - For Web, likely never leave slow start
- Is the Internet designed for this common case?
 - Why?

Evolution of Serving Web Content

74

- In the beginning...
 - ...there was a single server
 - Probably located in a closet
 - And it probably served blinking text
- Issues with this model
 - Site reliability
 - Unplugging cable, hardware failure, natural disaster
 - Scalability
 - Flash crowds (aka Slashdotting)



Replicated Web service

75

- Use multiple servers
- Advantages
 - Better scalability
 - Better reliability
- Disadvantages
 - How do you decide which server to use?
 - How to do synchronize state among servers?



Load Balancers

76

- Device that multiplexes requests across a collection of servers
 - All servers share one public IP
 - Balancer transparently directs requests to different servers
- How should the balancer assign clients to servers?
 - Random / round-robin
 - When is this a good idea?
 - Load-based
 - When might this fail?
- Challenges
 - Scalability (must support traffic for n hosts)
 - State (must keep track of previous decisions)
 - RESTful APIs reduce this limitation



Load balancing: Are we done?

77

□ Advantages

- Allows scaling of hardware independent of IPs
- Relatively easy to maintain

□ Disadvantages

- Expensive
- Still a single point of failure
- Location!

Where do we place the load balancer for Wikipedia?

Popping up: HTTP performance

78

- For Web pages
 - RTT matters most
 - Where should the server go?
- For video
 - Available bandwidth matters most
 - Where should the server go?
- Is there one location that is best for everyone?

Server placement

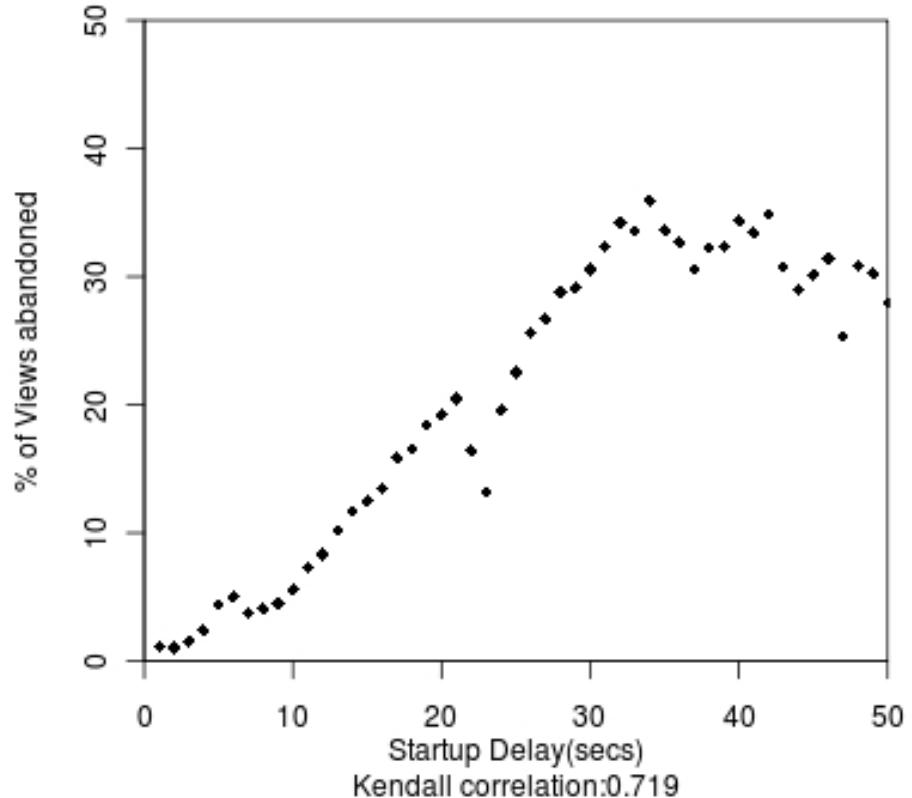
79



Why speed matters

80

- Impact on user experience
 - Users navigating away from pages
 - Video startup delay



Why speed matters

81

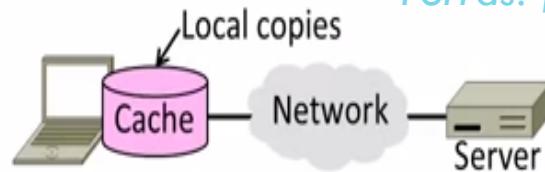
- Impact on user experience
 - Users navigating away from pages
 - Video startup delay
- Impact on revenue
 - Amazon: increased revenue 1% for every 100ms reduction in page load time (PLT)
 - Shopzilla: 12% increase in revenue by reducing PLT from 6 seconds to 1.2 seconds
- Ping from BOS to LAX: ~100ms



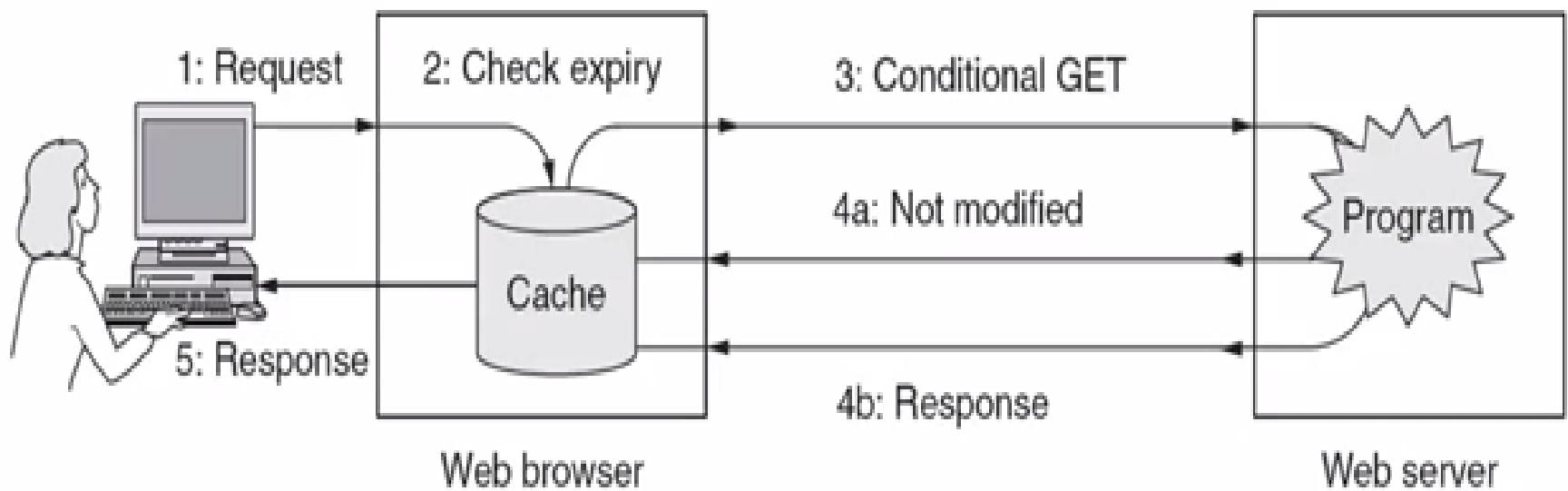
Web „caching” 1/2

- Felhasználók gyakran újra látogatják az oldalakat.
 - Használható a lokális másolata a lapnak. Ezt nevezzük „caching”-nek.
- **Kérdés:** Mikor használhatjuk a lokális másolatot?
 - Lokálisan meghatározni, hogy valid-e a kópia.
 - Lejárati információk alapján, mint például az „Expires” fejléc alapján a szervertől.
 - Heurisztikák használatával megtippelni.
 - **Előny:** A tartalom azonnal elérhető.
 - A másolat újra validálása a szerverrel.
 - Másolatban lévő időbényező alapján. (a *Last-Modified* fejléc)
 - A másolat tartalma alapján. (az *Etag* fejléc a szervertől)
 - **Előny:** A tartalom eav RTT-nvi késleltetéssel elérhető.

Forrás: [5]



Web „caching” 2/2

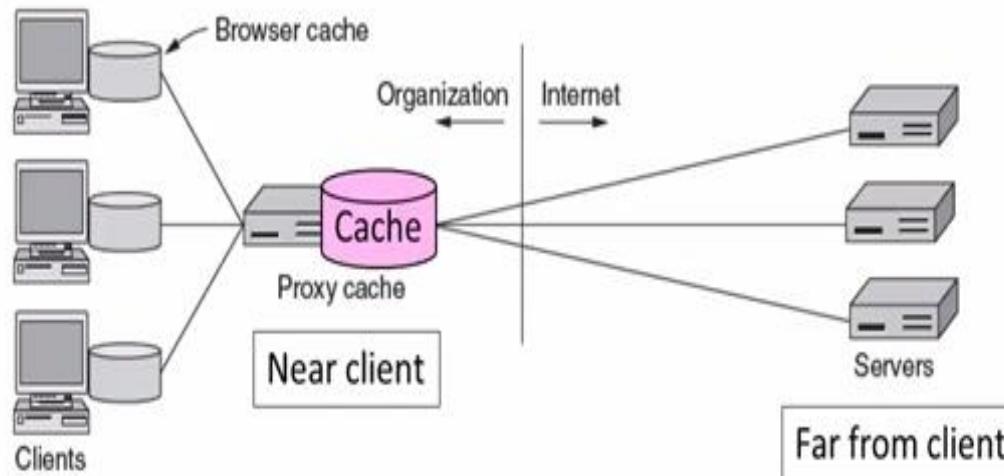


Forrás: [5]

Web proxy

- A kliensek csoportja és a külső web szerverek közé egy közbelső elem elhelyezése, ez lesz a proxy.
 - Előnyök a klienseknek: nagyobb cache és biztonsági ellenőrzés.
 - A szervezeti hozzáférés szabályozás.
- Proxy cache egy elosztott, nagy gyorsító tárat biztosít a klienseknek.
 - Korlátozások a tárolásra: biztonságos tartalmak, dinamikus tartalmak.

Forrás: [5]



Strawman solution: Web caches at ISPs

85

- ISP uses a middlebox that caches Web content
 - Better performance – content is closer to users
 - Lower cost – content traverses network boundary once
 - Does this solve the problem?
- No!
 - Size of all Web content is too large ☹
 - Web content is **dynamic** and **customized**
 - Can't cache banking content
 - What does it mean to cache search results?

What is a CDN?

86

- Content Delivery Network
 - Also sometimes called Content Distribution Network
 - At least half of the world's bits are delivered by a CDN
 - Probably closer to 80/90%
- Primary Goals
 - Create replicas of content throughout the Internet
 - Ensure that replicas are always available
 - Directly clients to replicas that will give good performance

Key Components of a CDN

87

- Distributed servers
 - Usually located inside of other ISPs
 - Often located in IXPs (coming up next)
- High-speed network connecting them
- Clients (eyeballs)
 - Can be located anywhere in the world
 - They want fast Web performance
- Glue
 - Something that binds clients to “nearby” replica servers

Examples of CDNs

88

- Akamai
 - 147K+ servers, 1200+ networks, 650+ cities, 92 countries
- Limelight
 - Well provisioned delivery centers, interconnected via a private fiber-optic connected to 700+ access networks
- Edgecast
 - 30+ PoPs, 5 continents, 2000+ direct connections
- Others
 - Google, Facebook, AWS, AT&T, Level3, Brokers

Inside a CDN

89

- Servers are deployed in clusters for reliability
 - Some may be offline
 - Could be due to failure
 - Also could be “suspended” (e.g., to save power or for upgrade)
- Could be multiple clusters per location (e.g., in multiple racks)
- Server locations
 - Well-connected points of presence (PoPs)
 - Inside of ISPs

Mapping clients to servers

90

- CDNs need a way to send clients to the “best” server
 - The best server can change over time
 - And this depends on client location, network conditions, server load, ...
 - What existing technology can we use for this?
- DNS-based redirection
 - Clients request www.foo.com
 - DNS server directs client to one or more IPs based on request IP
 - Use short TTL to limit the effect of caching

CDN redirection example

91

```
choffnes$ dig www.fox.com
```

;; ANSWER SECTION:

www.fox.com.	510	IN	CNAME	www.fox-rma.com.edgesuite.net.
www.fox-rma.com.edgesuite.net.	5139	IN	CNAME	a2047.w7.akamai.net.
a2047.w7.akamai.net.	4	IN	A	23.62.96.128
a2047.w7.akamai.net.	4	IN	A	23.62.96.144
a2047.w7.akamai.net.	4	IN	A	23.62.96.193
a2047.w7.akamai.net.	4	IN	A	23.62.96.162
a2047.w7.akamai.net.	4	IN	A	23.62.96.185
a2047.w7.akamai.net.	4	IN	A	23.62.96.154
a2047.w7.akamai.net.	4	IN	A	23.62.96.169
a2047.w7.akamai.net.	4	IN	A	23.62.96.152
a2047.w7.akamai.net.	4	IN	A	23.62.96.186

DNS Redirection Considerations

92

□ Advantages

- Uses existing, scalable DNS infrastructure
- URLs can stay essentially the same
- TTLs can control “freshness”

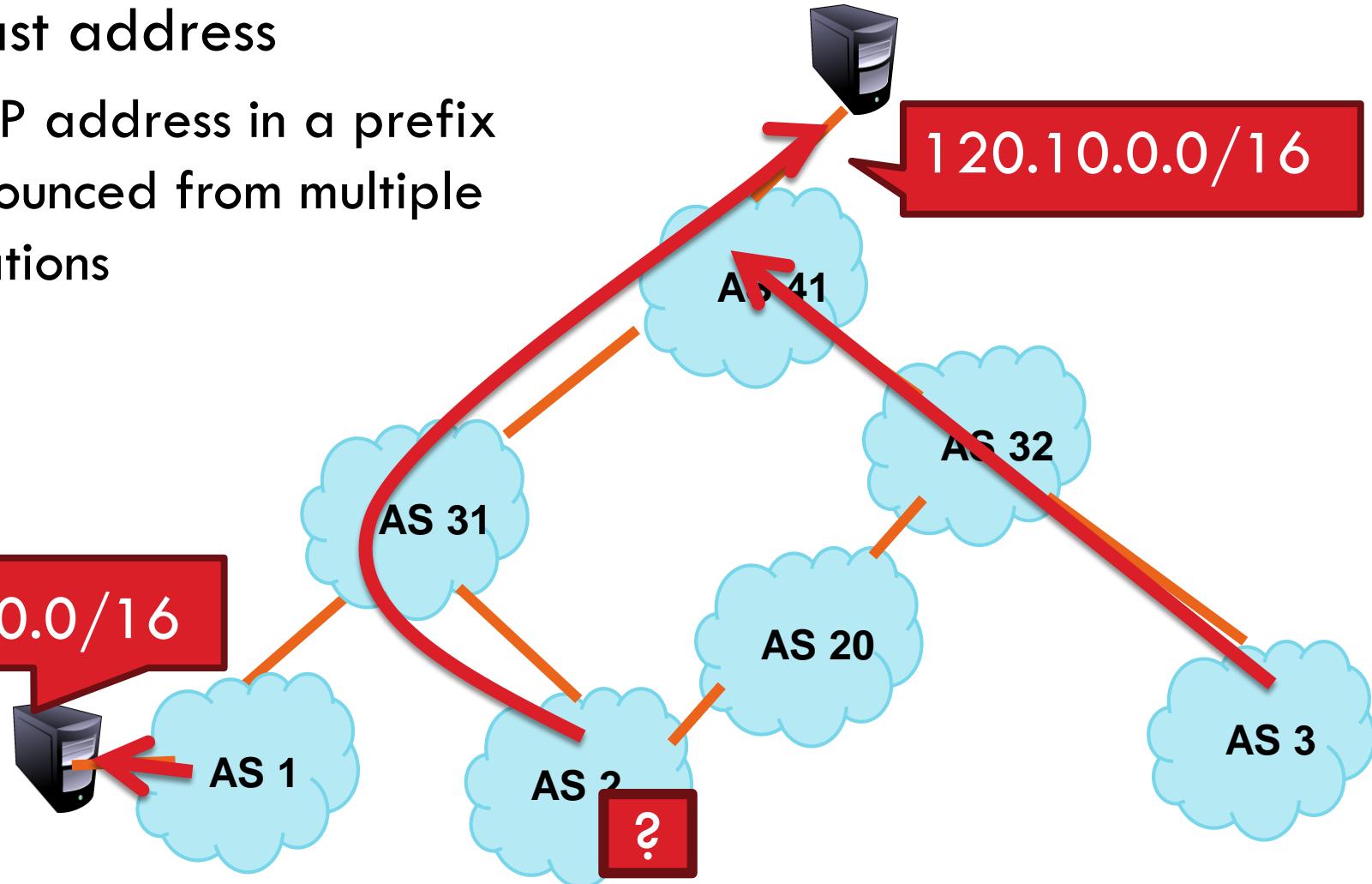
□ Limitations

- DNS servers see only the DNS resolver IP
 - Assumes that client and DNS server are close. Is this accurate?
- Small TTLs are often ignored
- Content owner must give up control
- Unicast addresses can limit reliability

CDN Using Anycast

93

- Anycast address
 - An IP address in a prefix announced from multiple locations



Anycasting Considerations

94

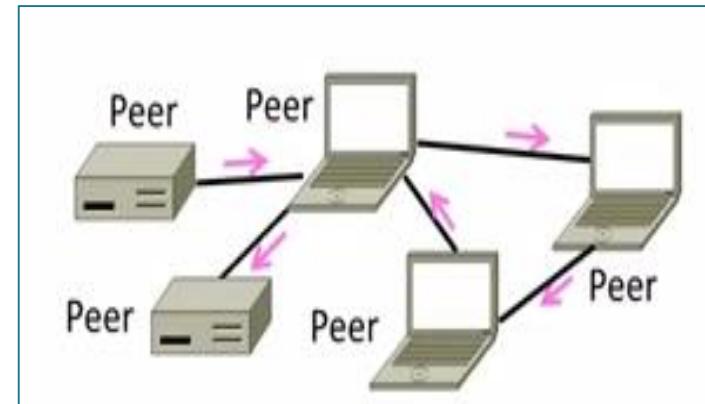
- Why do anycast?
 - Simplifies network management
 - Replica servers can be in the same network domain
 - Uses best BGP path
- Disadvantages
 - BGP path may not be optimal
 - Stateful services can be complicated

p2p tartalom megosztás 1/3

- CDNs hátrányai
 - Dedikált infrastruktúrát igényelnek.
 - Centralizált vezérlés/felügyelet kell.
- **Cél:** Olyan dedikált infrastruktúra és központi felügyelet mentes kézbesítés megvalósítása, ami még mindig hatékony és megbízható.
- **Kulcs:** a résztvevők, avagy peer-ek, segítsenek magukon
 - Napster 1999 zenei tartalomra,
 - BitTorrent 2001 bármilyen tartalomra.

P2P HÁLÓZAT JELLEMZŐI

- Nincs szerver.
- A kommunikáció peer-ek között folyik és önszerveződő.
- Skálázási problémák merülnek fel.

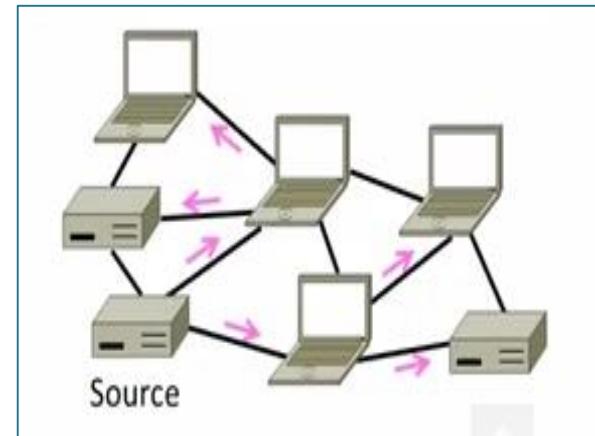


p2p hálózat
Forrás: [2]

p2p tartalom megosztás 2/3

P2P HÁLÓZAT KIHÍVÁSOK

1. Korlátozott lehetőségek
 - Hogyan képes egy peer tartalmat továbbítani az összes többi peer-nek?
2. Részvétel ösztönzése
 - Miért segítenének egymásnak a peer-ek?
3. Decentralizálás
 - Hogy találják meg a peer-ek a tartalmat?
- Peer-ek kettős szerepe:
 1. Feltöltés a többiek segítésére.
 2. Letöltés saját magának.

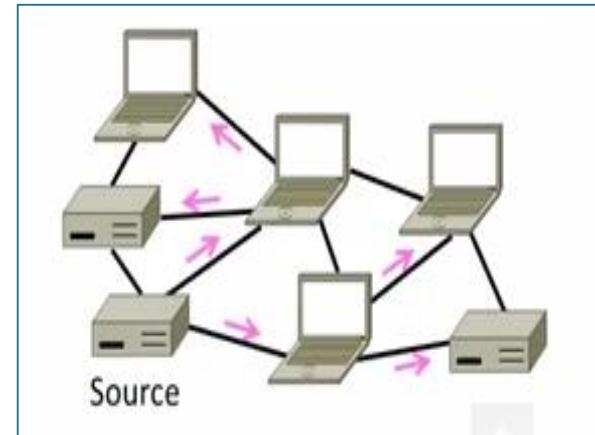


p2p megosztási fa
Forrás: [2]

p2p tartalom megosztás 3/3

P2P DECENTRALIZÁLÁS MEGVALÓSÍTÁSA

- A peer meg kell tanulja a tartalom helyét.
(Distributed Hash Tables)
- A DHTs teljesen decentralizált, hatékony algoritmusok egy elosztott indexhez.
 - Az index a peer-ek között terjed.
 - Az index listázza a tartalommal rendelkező peer-eket.
 - Bárminely peer kikeresheti az indexet.
 - Tudományos munkaként indult 2001-ben.



p2p megosztási fa
Forrás: [2]

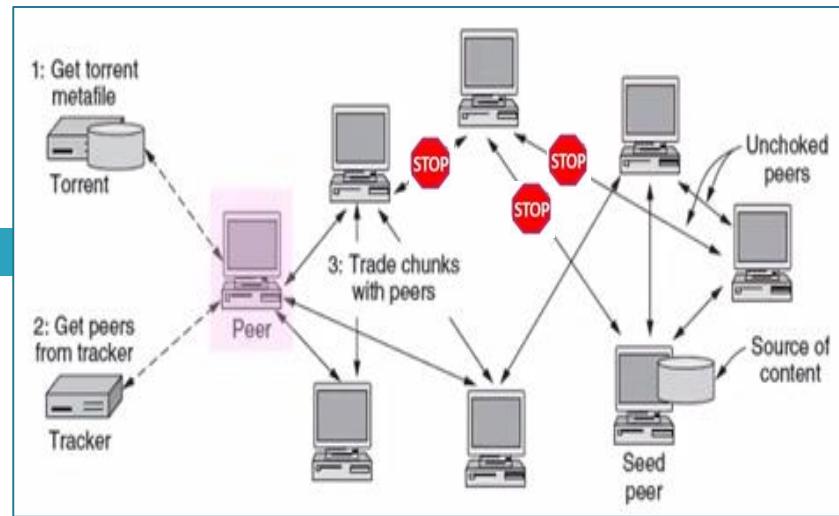
BitTorrent

- A legelterjedtebb p2p rendszer, amit:
 - 2001-ben fejlesztette ki Bram Cohen.
 - Nagyon gyorsan elterjedt, és jelenleg az Internet forgalom nagy részét teszi ki a BitTorrent forgalom.
 - Legális és illegális tartalmak megosztására is használnak.
- Az adatkézbesítés „torrent”-ek segítségével történik:
 - A fájlok megosztása darabonként történik.
 - Az ösztönzés céljára figyelemremélő módszert alkalmaz.
 - Tracker-ek vagy decentralizált indexek használata.

BitTorrent protokoll

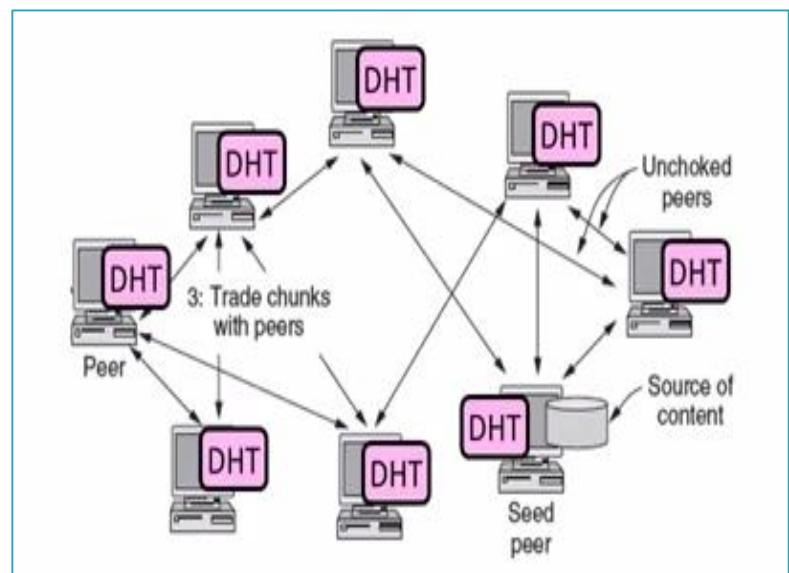
EGY TORRENT LETÖLTÉSÉNEK LÉPÉSEI

1. A torrent leírásával kezdődik.
2. Két lehetőség van:
 - a) Kapcsolatba lépni a tracker-rel a csatlakozáshoz, és elkérni a peer-ek listáját, amelyen legalább egy seed peer is van. (régi)
 - **seed peer** – Olyan speciális peer, aki rendelkezik a letöltendő fájl összes darabjával.
 - **leech peer** – Olyan peer, aki fel és le is tölt, azaz nem rendelkezik az összes darabbal.
 - b) Vagy DHT index használata a peer-ekhez. (új)
3. A különböző peer-ekkel forgalom lebonyolítása.
4. Előnybe részesítjük azon peer-eket, akik gyorsan töltenek fel a részünkre.
 - **choke peer** – Olyan peer, aki korlátozza a letöltést más peer-ek részére.



BitTorrent tracker-rel (példa)

Forrás: [6]



BitTorrent DHT-val (példa)

Forrás: [2]

100

Köszönöm a figyelmet!