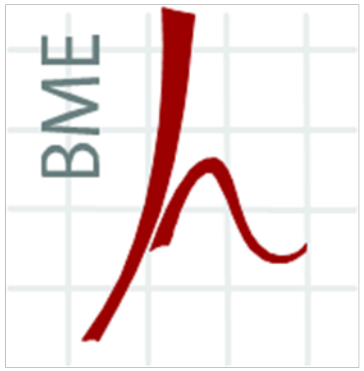


IP – Internet Protocol

IP címzés, routing, IPv6, IP mobilitás

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2014.Március 27.

IP - Áttekintés

- Bevezetés
  - A TCP/IP protokollarchitektúra
  - Az IP feladata és jellemzői
- Címzés
- Az IP csomag szerkezete
- Routing
- IP segédprotokollok
- IPv6
- Mobil IP megoldások

IP – Bevezetés

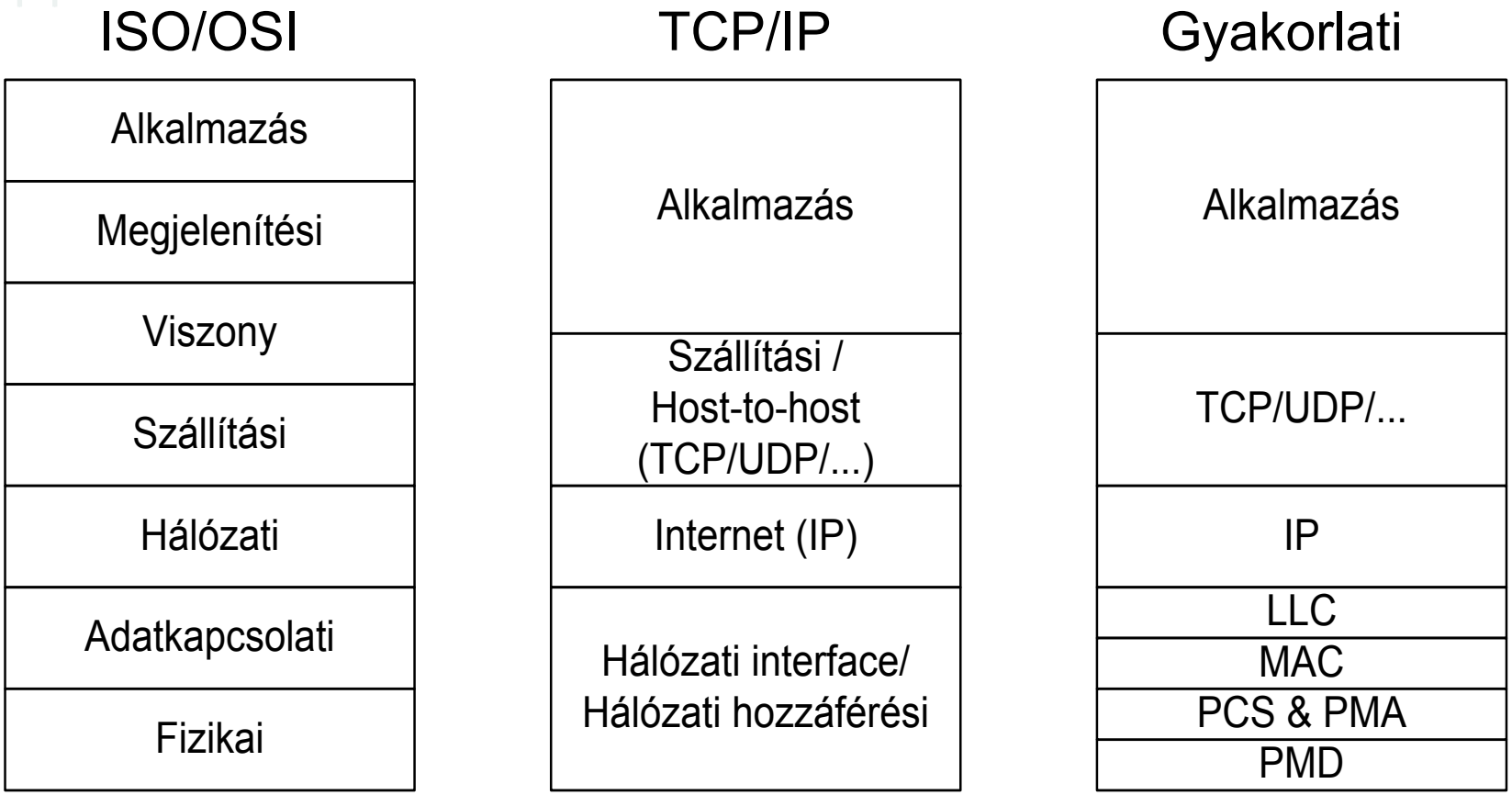
Az IP, mint hálózati protokoll

2013.Április 4.

Az Internet és az IP rövid története

- 1969 – ARPANET, „az Internet őse”
- 1974 – Az IP és a TCP alapgondolata
  - Protokollarchitektúra az NCP kiváltására az ARPANET-en
  - „A Protocol for Packet Network Interconnection” (Vinton G. Cerf, Robert E. Kahn)
  - Az „Internet” kifejezés megjelenése (Internet Protocol Suite)
- 1980 – IPv4 (Internet Protocol version 4)
  - Az első, széles körben használt IP és TCP verziók megalkotása
  - DoD (Department of Defense) által támogatott szabványok: IP (RFC 760), TCP (RFC 761)
  - 1983. január 1: Flag day, NCP → TCP/IP
- Alapvetően ma is ezt használjuk!

TCP/IP architektúra és az ISO/OSI rétegmodell



IP: Internet Protocol
TCP: Transmission Control Protocol
UDP: User Datagram Protocol
LLC: Logical Link Control

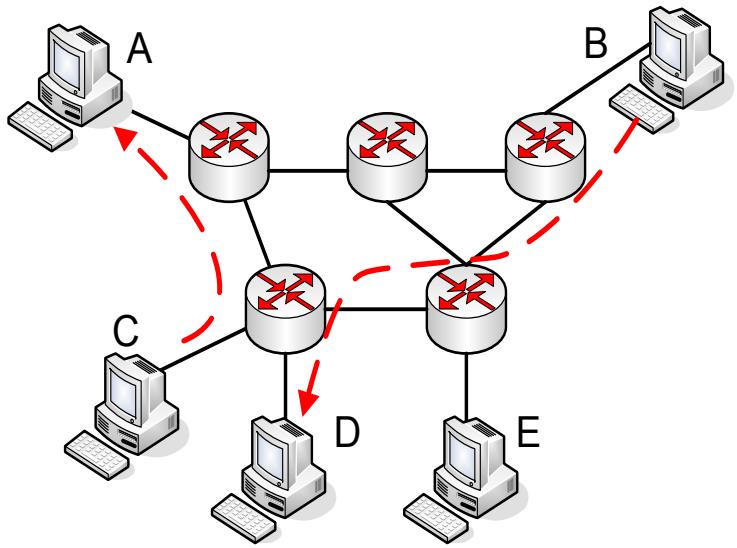
MAC: Medium Access Control
PCS: Physical Coding Sublayer
PMA: Physical Medium Attachment
PMD: Physical Medium Dependent

ISO OSI vs. IETF

- OSI 7 rétegét a gyakorlatban nem szokták megvalósítani
- IETF szerint: Internet protokolljainak és architektúrájának fejlesztése továbbra sem lesz OSI alapú!
- Nálunk: 5 réteg, TCP/IP modell használata, de
  - interface réteg= adatkapcsolati réteg + fizikai réteg

Az IP feladata

- Hálózati protokoll
  - Adattovábbítás a hálózat végpontjai között
- Két fő funkció
  - **Címzés (addressing) és útvonalválasztás (routing)**
  - Tördelés/fragmentálás (fragmentation)



Az IP jellemzői

- Csomagkapcsolt
- Összeköttetés-mentes (connectionless)
- „Best effort” – nincs garancia

} Datagram típusú

  - A csomagokra igaz:
    - Elveszhetnek
    - Duplikálódhatnak
    - Sorrendjük megváltozhat
    - Meghibásodhatnak (nincs hibaészlelés és -javítás)
- Egyéb **NEM** nyújtott szolgáltatások:
  - Torlódáskezelés
  - Ütemezés
  - Titkosítás és hitelesítés

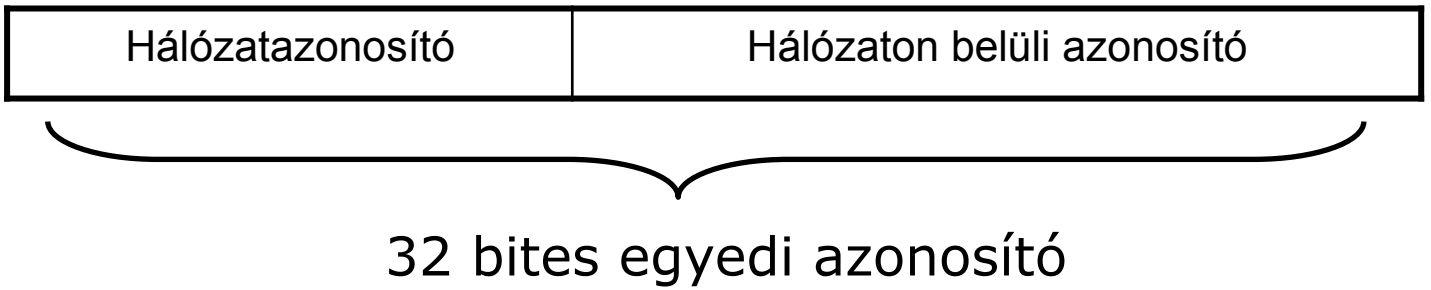
IP – Címzés

IP-címek és címosztályok

2014.Március 27.

Az IPv4-cím felépítése

- 4 bájtos cím (32 bit)
  - $2^{32} \approx 4 \cdot 10^9$ ⇨ kimerült!
- Jelölések
  - Bináris: 10110000 10010011 00111110 11100001
  - „Dotted decimal”: 176.147.62.225
- Cím = Hálózatazonosító + Egyedi azonosító



Címosztályok

- Eleinte első 8 bit hálózati, majd bevezették a címosztályokat (1981)
- Különböző méretű hálózatoknak más méretű IP-cím tartományok

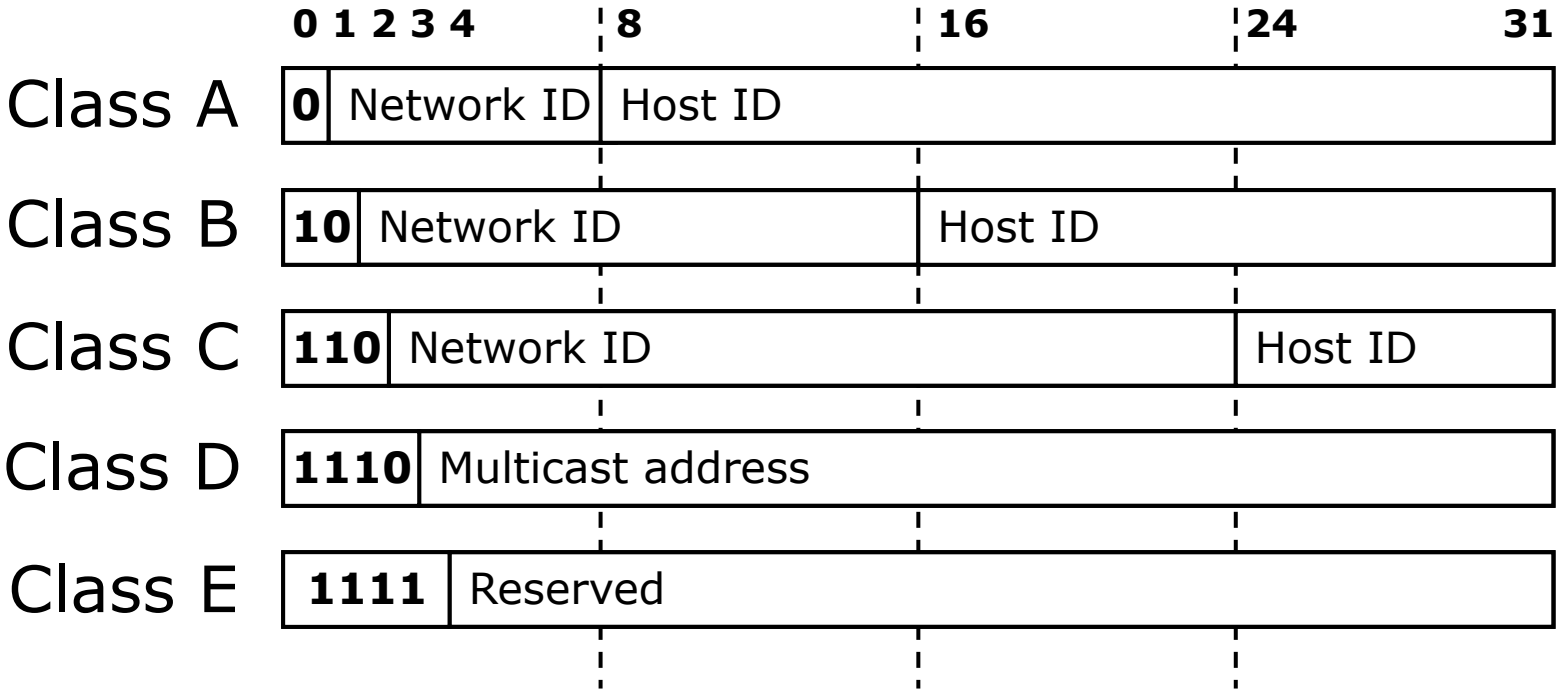
Osztály	Hálózatazonosító bitjeinek száma*	Hálózatok száma*	Hálózaton belüli címek száma*
A	8	126	16777214
B	16	16382	65534
C	24	2097150	254

- Az IP-címeket kezelő szervezet, az IANA: Internet Assigned Numbers Authority osztja ki
- Alatta 5 Regional Internet registry
  - A és B osztályú címet kaphatnak tipikusan:
    - Országok
    - Internet-szolgáltatók (ISP: Internet Service Provider)
    - Egyetemek

Címosztályok

- A, B és C osztályú címek
  - Egyedi címzésre (unicast)
- D osztály
  - Többes címzésre (multicast) (később részletesen)
  - Fenntartott címtartomány:
    224.0.0.0-től 239.255.255.255-ig
- E osztály
  - Fenntartott osztály és címtartomány
    240.0.0.0-től 255.255.255.255-ig

A címosztályok címkiosztása



- Osztályazonosító prefix
- Diszjunkt címtartományok

Speciális címek

- Host ID: csupa 0
  - Hálózat címe
  - Az adott hálózat eszközei opcionálisan kezelhetik
- Host ID: csupa 1
  - Broadcast cím
  - A hálózaton mindenkinek szól

- E kettő miatt csak $2^{N_{host}} - 2$ számú terminált lehet megcímezni

- 127.0.0.0 - 127.255.255.255
  - Loopback cím
  - A helyi gépet azonosítja: localhost
  - Pl. vonaltesztelésre

Problémák a címosztályokkal

- Egyes nagy cégeknek, egyetemeknek *A* osztályú címek, amit nem használtak ki (16 millió cím)
  - B kicsi lett volna nekik
- Ugyanez középcégeknél: *C* túl kicsi, *B* túl nagy
- Gond a routing aggregációval
- Nem skálázható az osztályokon alapuló rendszer!

Privát címtartományok

- Első megoldás az IPv4 címtartomány kimerülésre
- Ötlet: pl. cégen belül nem kell globálisan egyedi IP cím ha nem kommunikál kifelé
- Nem kell hozzá RIR jóváhagyás
- Ha kifelé is szeretne: Network Address Translation (NAT)
- Privát IP-címtartományok (RFC 1918)
  - Csak helyi hálózaton (Interneten nem) érvényes címek
  - 10.0.0.0 – 10.255.255.255 (1 db A osztály)
  - 172.16.0.0 - 172.31.255.255 (16 db B osztály)
  - 192.168.0.0 - 192.168.255.255 (256 db C osztály)

Gondok privát címtartománnyal

- Késleltette az IPv6 bevezezését
- Megsérti a végpont-végpont konnektivitás elvét
  - Sőt egyes Internet protokollokra nem is működik
- Privát hálózatok egyesítése: címduplikáció
  - Újracímezés
  - NAT közöttük
- Privát címek kiszivárgása az Internetre
  - Rosszul konfigurált privát hálózat
  - Komoly terhelés a névszervereknek
    - Blackhole anycast névszerverek
  - Határ routerek szűrik

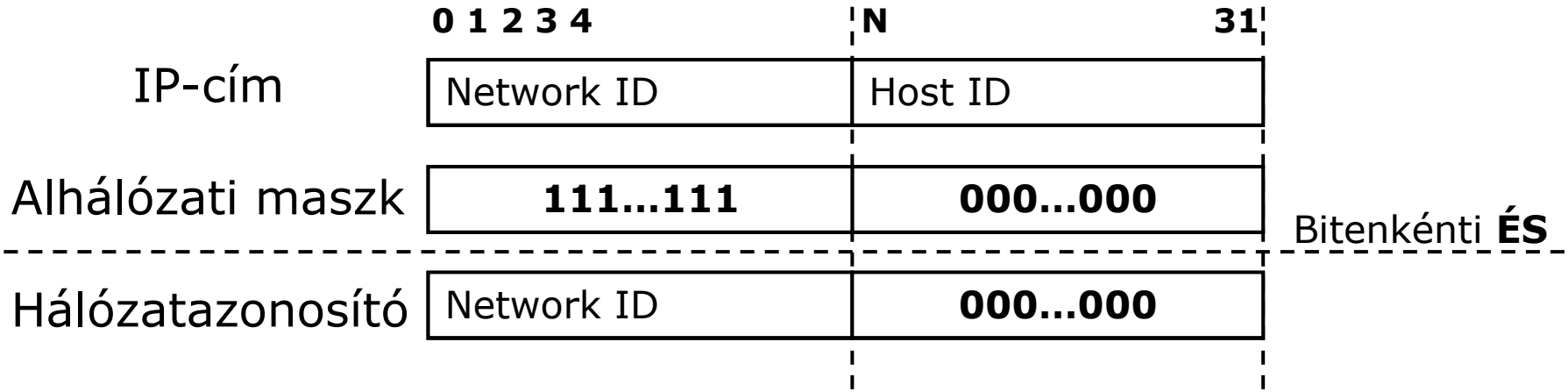
Példák: IP-címek

- Milyen IP-címek az alábbiak?
  - 152.66.115.35 (www.bme.hu)
    - Első bitjei: 10… ⇨ B osztályú cím
    - Utolsó 16 bitje nem csupa 0 vagy 1 ⇨ egyedi cím
  - 195.199.23.255
    - Első bitjei: 110… ⇨ C osztályú cím
    - Utolsó 8 bitje csupa 1 ⇨ broadcast cím
  - 10.97.15.255
    - Első bitje: 0… ⇨ A osztályú cím
    - Utolsó 24 bitje nem csupa 0 vagy 1 ⇨ egyedi cím
    - Privát címtartománybeli ⇨ globálisan nem használható
- Mi legkisebb és legnagyobb C osztályú cím?
  - 11000000 00000000 00000000 00000000 ⇨ 192.0.0.0
  - 11011111 11111111 11111111 11111111 ⇨ 223.255.255.255

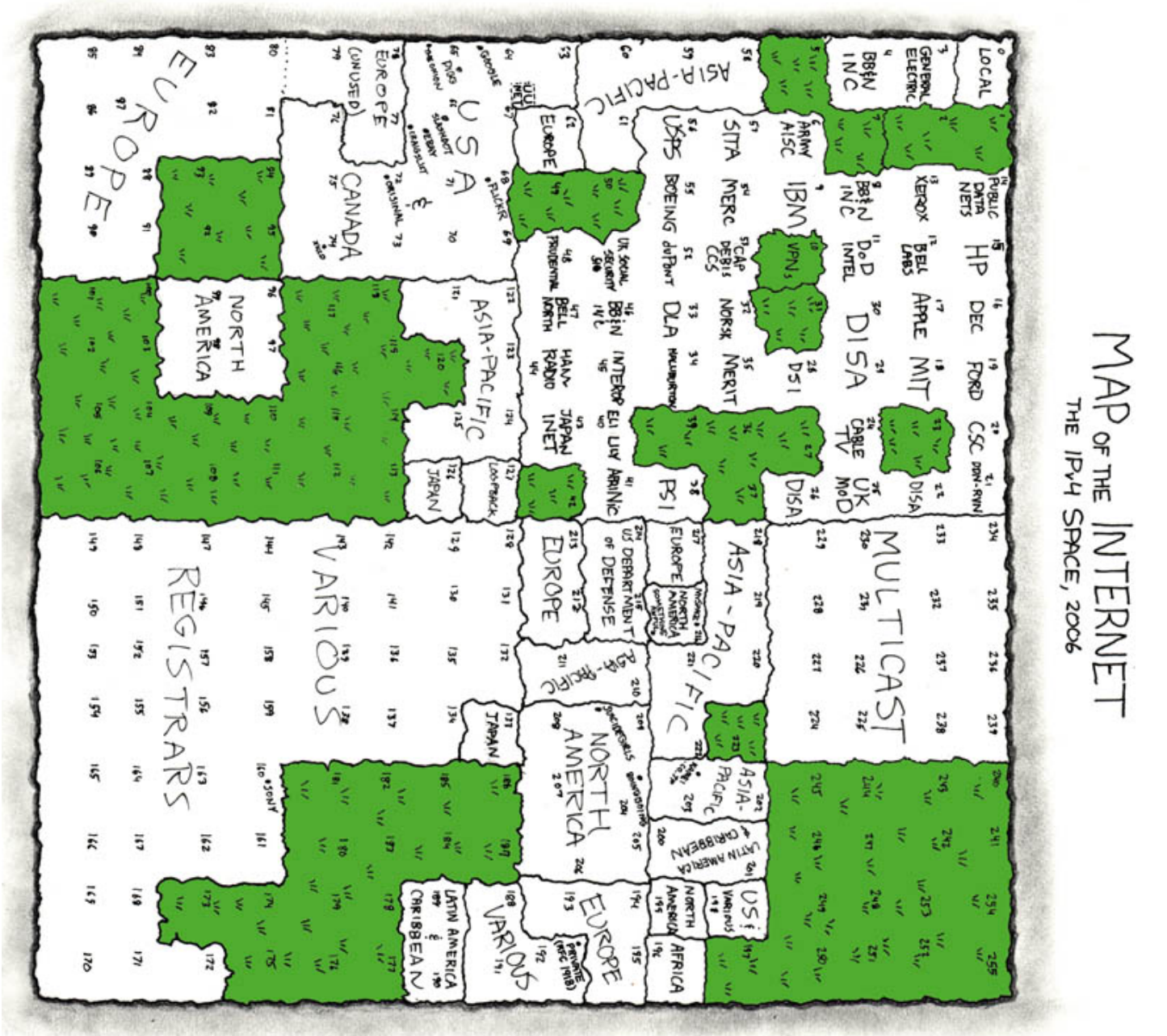
Címosztályok általánosítása

- Osztály alapú nem skálázható, plusz IPv4 címtartomány kimerülés
- CIDR – Classless Inter-Domain Routing
  - VLSM – Variable Length Subnet Mask
    - Osztályok eltörlése: a cím 32 bitje tetszőleges helyen lehet kettéosztva hálózat- és végponti azonosítóra
  - A címből nem lehet megállapítani, hol lett kettéosztva ⇨ alhálózati maszk
- Subnetting
  - Alhálózatokra osztás vagy aggregálás
    - 1 db A osztályú ⇔ 256 db B osztályú
    - 1 db B osztályú ⇔ 256 db C osztályú
  - Már nem lehet a prefix alapján megállapítani

Alhálózati maszk (Subnet mask)

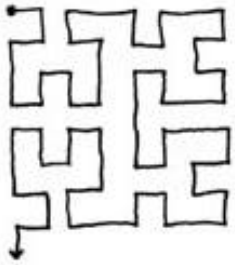


- Jelölés a hálózat egyértelmű azonosítására
  - <hálózat IP-címe> / <alhálózati maszk egyeseinek a száma>
  - Így pl. a BME hálózata: 152.66.0.0 /16 (alhálózati maszkja 255.255.0.0)
  - Hasonlóan:
    - A osztály: /8
    - B osztály: /16
    - C osztály: /24



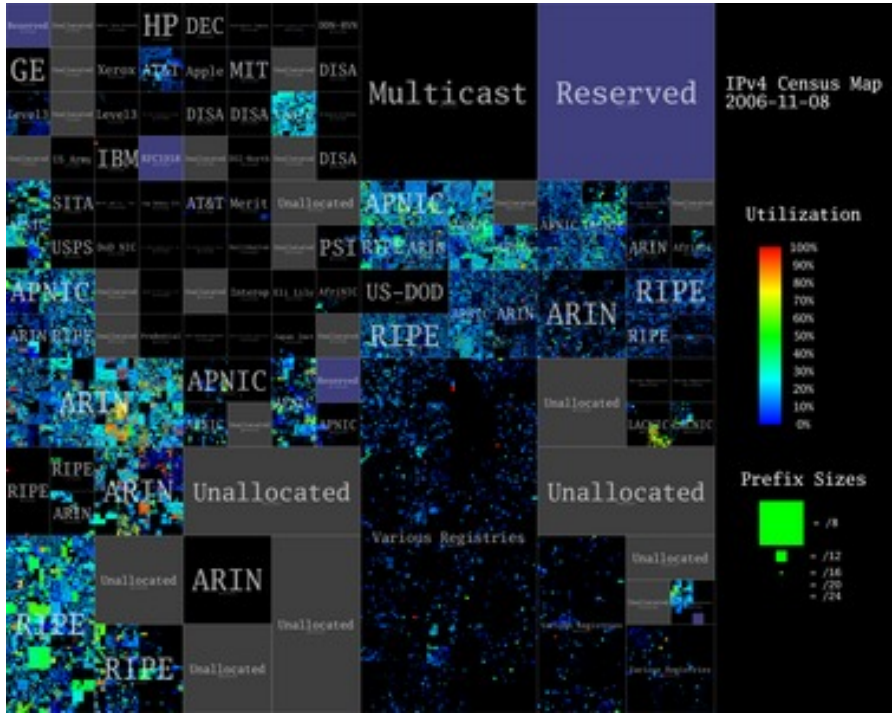
THIS CHART SHOWS THE IP ADDRESS SPACE ON A PLANE USING A FRACTAL MAPPING WHICH PRESERVES GROUPING -- ANY CONSECUTIVE STRING OF IPs WILL TRANSLATE TO A SINGLE COMPACT, CONTIGUOUS REGION ON THE MAP. EACH OF THE 256 NUMBERED BLOCKS REPRESENTS ONE /8 SUBNET (CONTAINING ALL IPs THAT START WITH THAT NUMBER). THE UPPER LEFT SECTION SHOWS THE BLOCKS SOLD DIRECTLY TO CORPORATIONS AND GOVERNMENTS IN THE 1990's BEFORE THE RIRs TOOK OVER ALLOCATION.

0	1	14	15	16	19 →
3	2	13	12	17	18
4	7	8	11		
5	6	9	10		

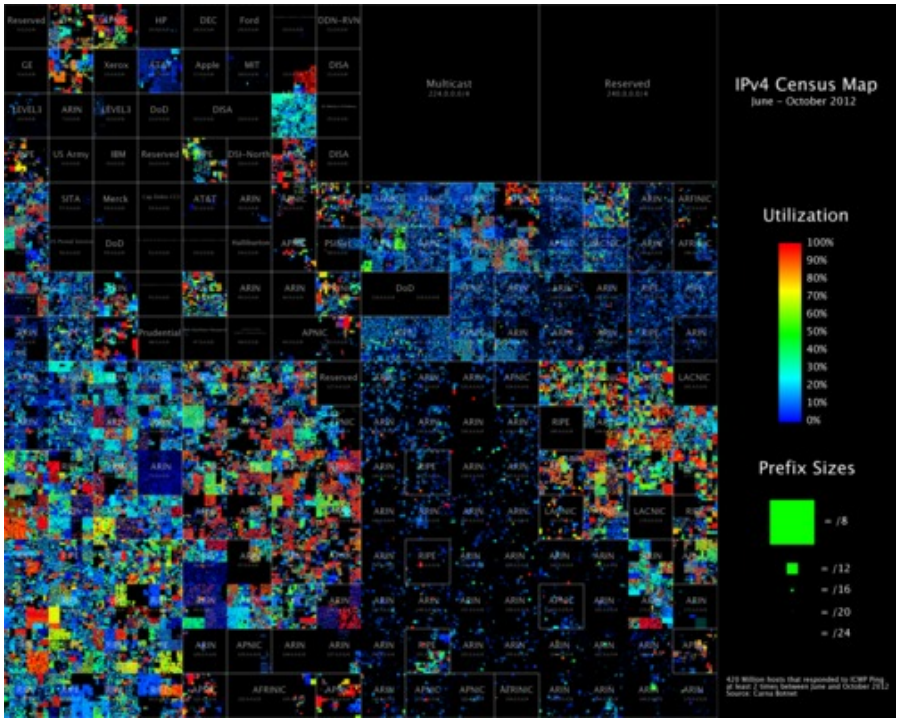


IPv4 címtartomány kimerülése

2006



2012

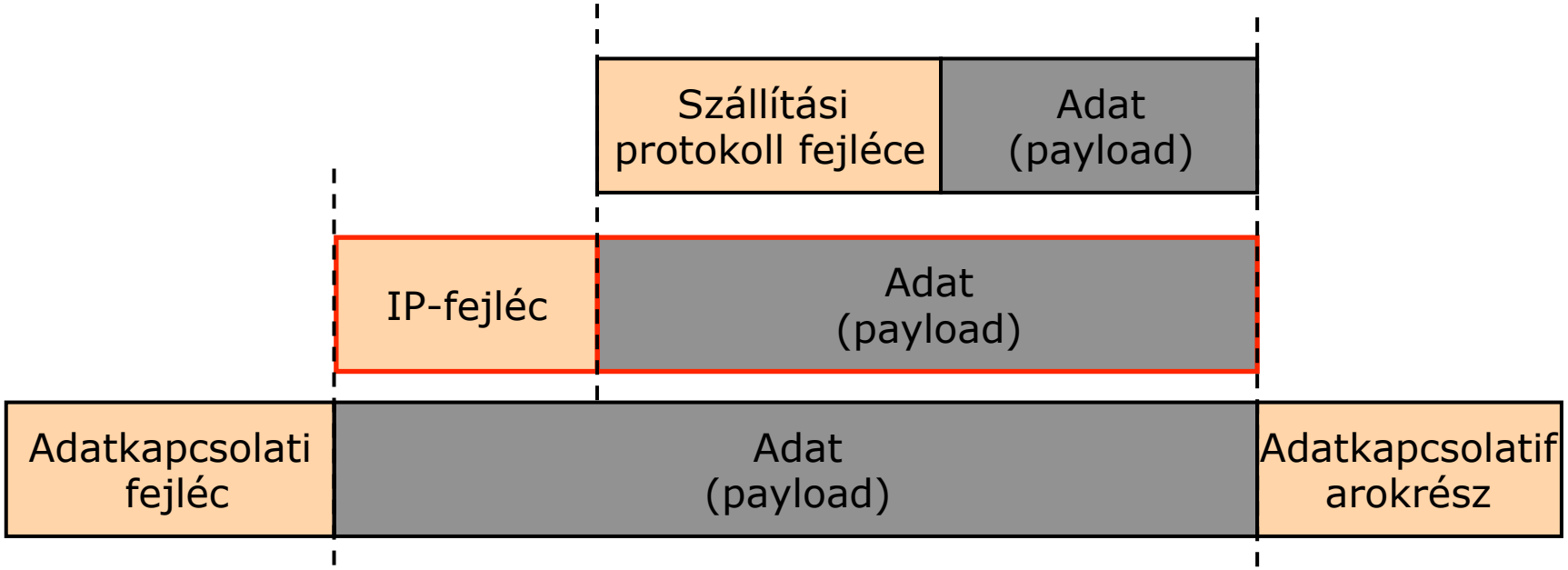


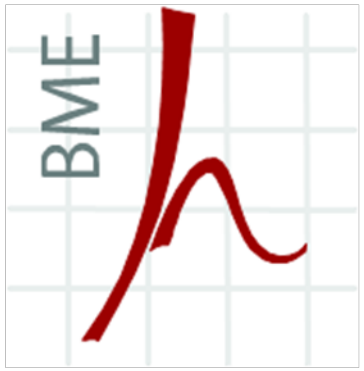
IP - Áttekintés

- Bevezetés
  - A TCP/IP protokollarchitektúra
  - Az IP feladata és jellemzői
- Címzés
- Az IP csomag szerkezete
- Routing
- IP segédprotokollok
- IPv6
- Mobil IP megoldások

Az IP-csomag szerkezete és „helye”

- Az IP-csomag két része:
  - IP-fejléc / fejrész (IP header)
  - Adat (payload)
- Az IP-csomag alsóbb rétegbeli protokoll adatrészébe ágyazódik be
- Az IP-csomag adatrészébe magasabb rétegbeli protokollüzenet (PDU) kerül



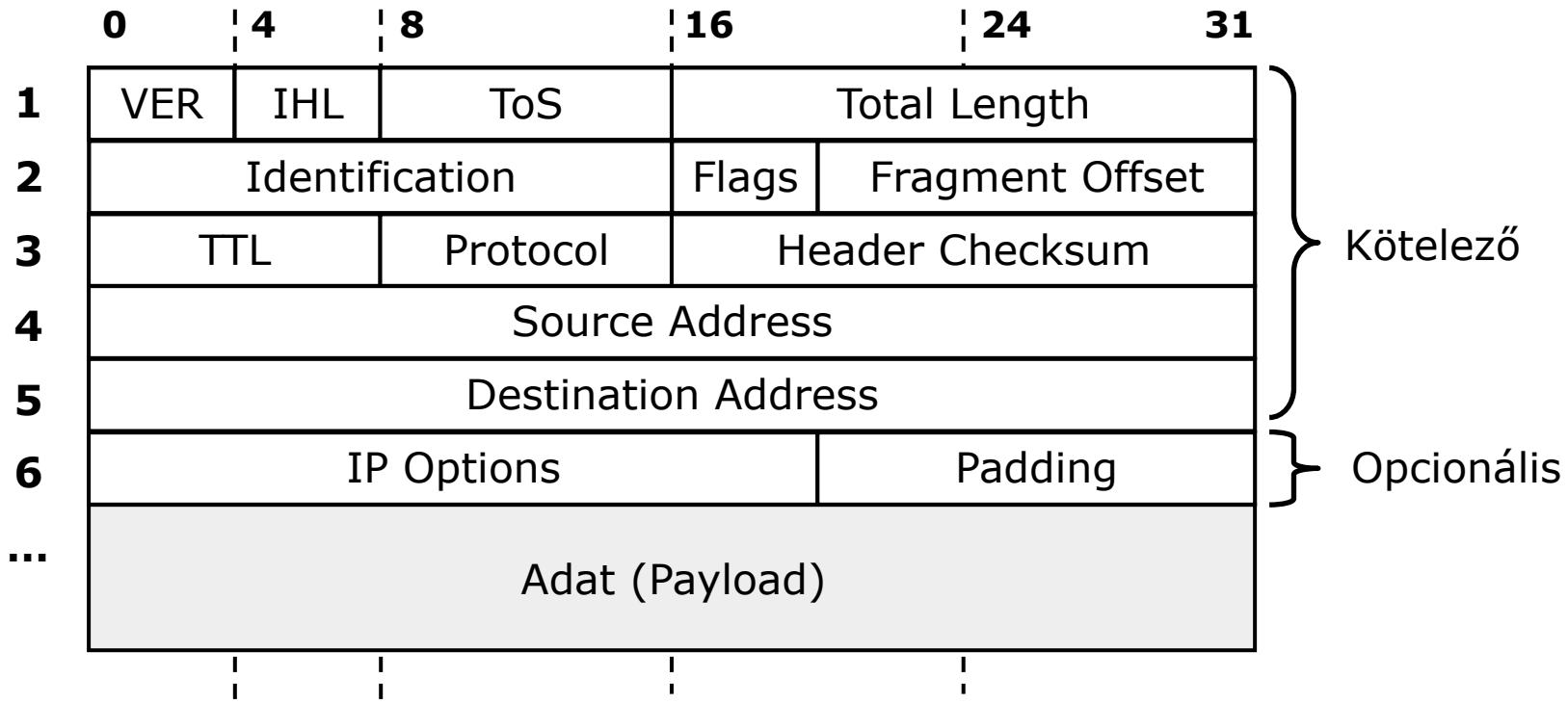


IP - Fejléc

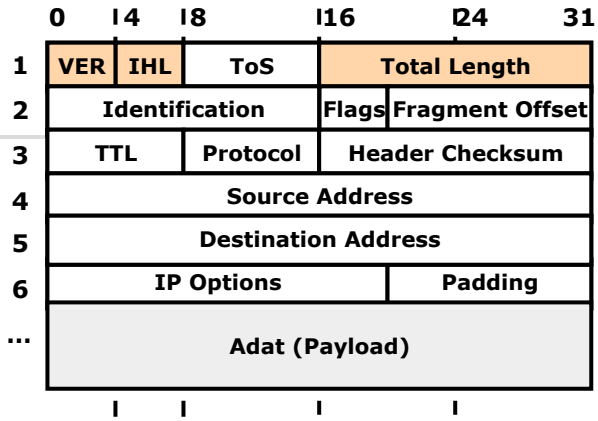
Az IP fejléc szerkezete

2014.Március 27.

Az IP fejléc szerkezete

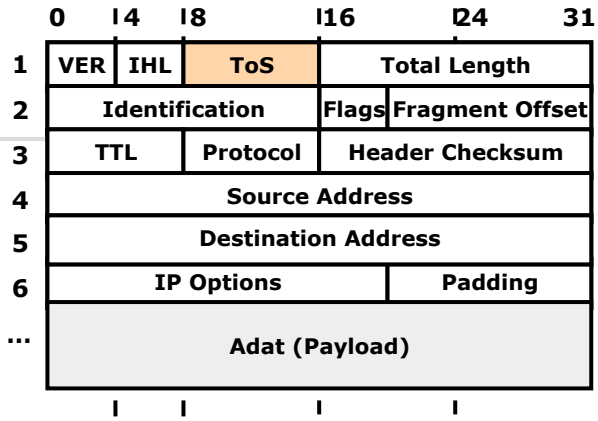


Az IP fejléc mezői I.



- VER – Version (4 bit)
  - Az Internet Protocol verziójának száma
  - Tipikus értéke: IPv4: 4, IPv6: 6
- IHL – Internet Header Length (4 bit)
  - Az IP fejléc mérete **32 bites szavakban**
  - Értéke:
    - Minimum: 5 (20 byte)
    - Maximum: 2^4-1=15 (60 byte)
- Total Length (16 bit)
  - A teljes IP csomag mérete **bájtokban**
  - Értéke:
    - Minimum: 576 (IP-fejléc 20 + 512 adat + 44 IP-opciók és az alsóbb réteg fejlécei)
      Ilyen méretű csomagot tördeletlenül kell továbbítani!
    - Maximum: 2^{16}-1=65535 (max. 65515 bájt adat)

Az IP fejléc mezői II.

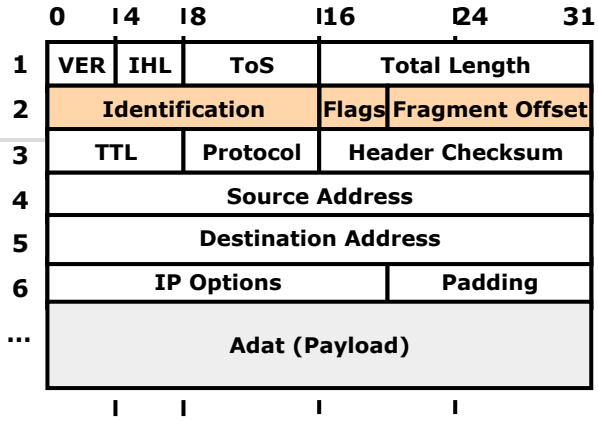


- ToS (Type of Service) (8 bit)
  - QoS osztályok, paraméterek jelzésére (RFC 791)

Bitek	Jelentés
0-2	Precedencia Példák: „network control” „priority” „routine”
3	Késleltetés (normal/low)
4	Throughput (normal/light)
5	Reliability (normal/high)
6-7	Fenntartott

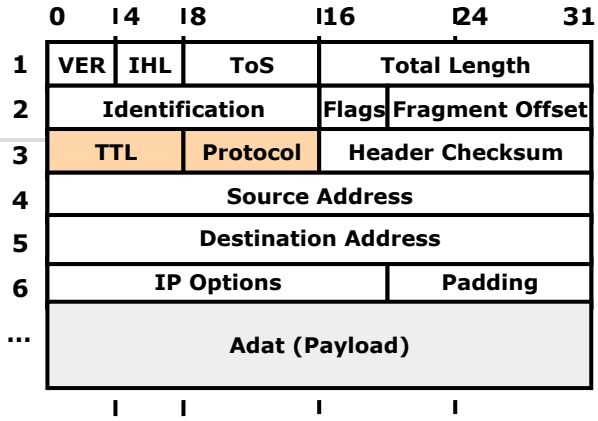
  - A legtöbb router nem támogatja, de jövőben várható
  - Leggyakrabb felhasználás:
    - DSCP – Differentiated Services Code Point (RFC 2474)- 6bit: **DiffServ** (lesz róla szó később!)
    - ECN – Explicit Congestion Notification (RFC 3168)- 2bit

Az IP fejléc mezői III.



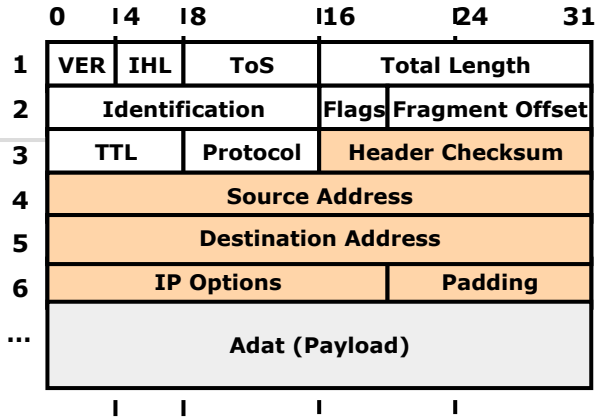
- **Identification (16 bit)**
  - Az IP-töredékek egyedi azonosítása
- **Flags (3 bit)**
  - 0: Fenntartott
    - 0-nak kell lennie
    - „Evil bit” (RFC 3514) (áprilisi tréfa 2003-ban)
  - 1: DF – Don’t Fragment
    - 1: ha tördelni kellene, el kell dobni
    - Ezt használják az MTU (Maximum Transmission Unit) Path Discovery néhány TCP verzióban és az IPv6-ban
  - 2: MF – More Fragment
    - 1: ha nem az utolsó töredék
    - 0: utolsó töredék vagy nem tördelt csomag
- **Fragment Offset (13 bit)**
  - Az eredeti csomagban lévő kezdőpozícióját adja meg e töredékben lévő adatnak **8 bájtos egységekben**
  - $(2^{13}-1)x8=65528 > 65515$ bájt (max. adat)
    (így a maximális méretű csomag is tördelhető)

Az IP fejléc mezői IV.



- **TTL – Time To Live (8 bit)**
  - Csomag élettartama
  - Eredetileg másodpercben
  - Gyakorlatban **hopszámban** mérve (hop count)
    - Minden továbbításnál csökkenteni kell ez értékét legalább 1-gyel
    - Ha a csökkentés után az értéke nem >0, akkor el kell dobni
- **Protocol (8 bit)**
  - Az adatrészben lévő protokoll azonosítója
  - Pl: (kezdeti lista az RFC 790-ben)
    1: Internet Control Message Protocol (ICMP)
    2: Internet Group Management Protocol (IGMP)
    6: Transmission Control Protocol (TCP)
    8: Exterior Gateway Protocol (EGP)
    17: User Datagram Protocol (UDP)
    89: Open Shortest Path First (OSPF)
    132: Stream Control Transmission Protocol (SCTP)
  - Az IANA felügyeli ezeket az azonosítókat

Az IP fejléc mezői V.



- **Header Checksum (16 bit)**
  - Az **IP fejléc** minden 16 bites szavának összegére számolt egyes komplemens
  - A csomag érkezésekor ellenőrizni kell a helyességét, és továbbítás esetén újra kell számítani
    - Új TTL minden lépésben, és fragmentáció is lehet
- **Source / Destination Address (2 x 32 bit)**
  - Az IP-csomag feladójának és címzettjének az IP-címe
    - NAT miatt lehet nem a feladó tényleges címe
- **IP Options**
  - Ritkán használt opcionális része a fejlécnek
  - Segítségével például a csomag útjának lehet kijelölni
    - SSRR: Strict Source (Record) Route – pontos út kijelölése
    - LSSR vagy LSRR: Loose Source (Record) Route – kötelező útba ejtendő csomópontok kijelölése
    - Legtöbb router biztonsági okokból eldobja
- **Padding**
  - Az IP Options részt egészíti ki 4 bájt többszörösére

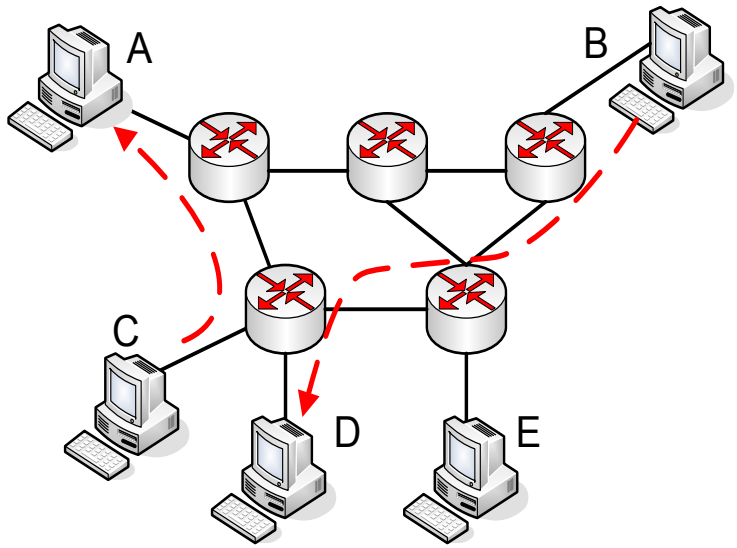
IP - Útvonalválasztás

Az IP routing működése

2014.Március 27.

Az IP feladata

- Hálózati protokoll
  - Adattovábbítás a hálózat végpontjai között
- Két fő funkció
  - **Címzés (addressing) és útvonalválasztás (routing)**
  - Tördelés (fragmentation)



A csomagtovábbítás alapelve

- Az IP jellemzői
  - Csomagkapcsolt
  - Összeköttetés-mentes (connectionless)
  - „Best effort” – nincs garancia

} Datagram típusú

- **„Hot potato”-elv**
  - Minél gyorsabban továbbítsuk
  - Csak a következő csomópontot kell ismernünk (**hop-by-hop**)
  - Előnyei:
    - Kis erőforrásigény
      - Egyszerű, ezért gyorsan továbbítható
      - Gyors, ezért nem kell tárolnunk a csomagokat
        - » kis memóriaigény
    - „Kevés” ismeret a hálózatról
    - Datagram szolgáltatásra tökéletesen alkalmas
      - nem vállal garanciát, nincs szükség nyugtázásra és egyéb hibakezelésre ⇨ gyorsasága megmarad

Továbbításhoz szükséges információk

- Kinek küldjük tovább?
  - Cél címe alapján ⇨ csomag tartalmazza
  - Saját ismeret alapján ⇨ **útvonal-irányítási táblázat** (routing table)
- Hogyan küldjük tovább?
  - Mekkora egységekben?
    - Mekkora érkezik? ⇨ csomag határozza meg
    - Mekkora továbbítható? ⇨ **a következő hálózat MTU-ja** (Maximum Transmission Unit) határozza meg
  - Milyen QoS biztosításával?
    - „Best effort” – nincs garancia
    - Opcionális megoldás: **ToS mező** felhasználásával egyéb protokollok és mechanizmusok

Routing (útválasztás, útvonal-irányítás)

Tördelés (fragmentation)

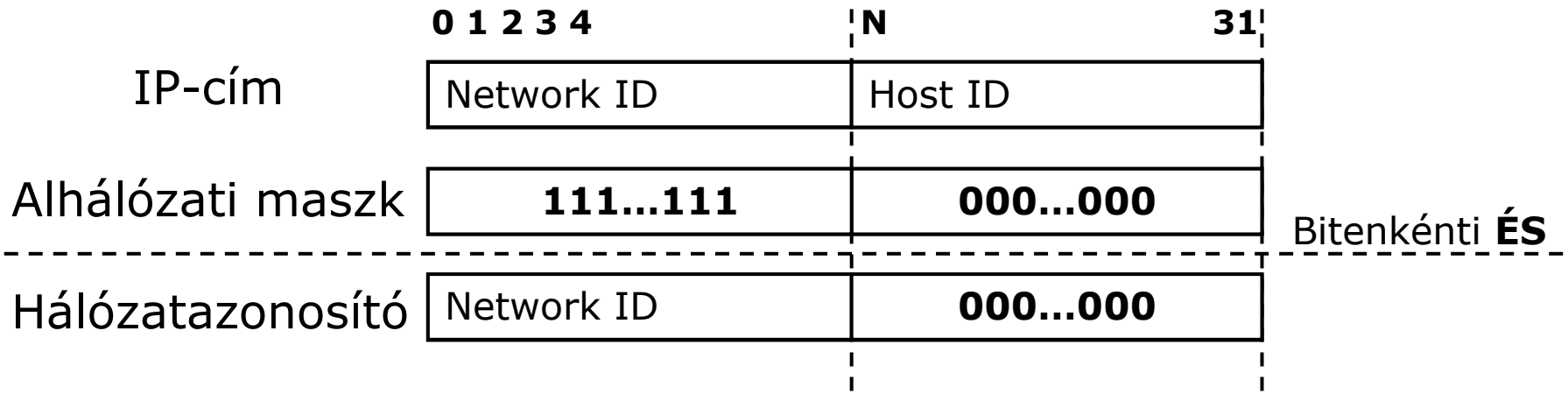
QoS

Útválasztó tábla (Routing table)

- Szükséges információk
  - Hova tart?
  - Merre küldjük tovább?
    - Ki a következő csomópont?
    - Melyik interfészen kell továbbítani?

Hálózat címe	Alhálózati maszk	Interfész	Közvetlenül kapcsolódó	Következő csomópont
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>

Alhálózati maszk (Subnet mask)



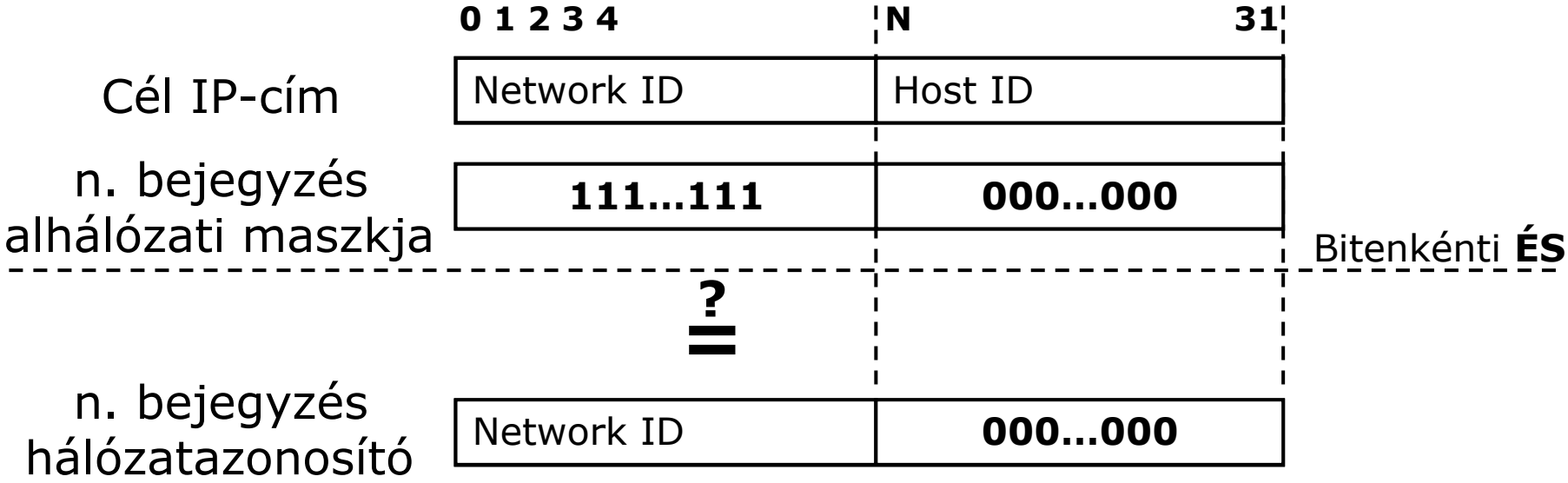
- Jelölés a hálózat egyértelmű azonosítására
  - <hálózat IP-címe> / <alhálózati maszk egyeseinek a száma>
  - Így pl. a BME hálózata: 152.66.0.0 /16 (alhálózati maszkja 255.255.0.0)
  - Hasonlóan:
    - A osztály: /8
    - B osztály: /16
    - C osztály: /24

Útválasztó tábla (Routing table)

Merre megy?		Merre küldjük?		
Hálózat címe	Alhálózati maszk	Interfész	Közvetlenül kapcsolódó	Következő csomópont
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>

- Táblázatban szereplő hálózatok közül melyikbe megy?
- Az ahhoz tartozó interfészre küldjük ki

Hálózat kiválasztása



- A cél IP-cím akkor tartozik egy hálózatba, ha a hálózatazonosító résszel megegyezik

Keresés az útválasztó táblában

- Ha több hálózatra is illeszkedik: **leghosszabb egyezőséget** kell választani!
  - leghosszabb hálózati maszk

- Az útválasztó tábla összes bejegyzését végig kell nézni
  ⇨ rosszul skálázódik
  - Általában bináris fával implementálják
    ⇨ *n* számú bejegyzésben lineáris keresés helyett csak a cím hosszában logaritmikus

- Keresés csak akkor, ha az nem saját cím

Alapértelmezett útvonal (Default route)

- Erre megy a csomag, ha nem ismeri a célhálózatot
- Minden címet tartalmazó hálózat
  - Ehhez a hálózati cím: 0.0.0.0/0
- Az alapértelmezett átjáró és elnevezése
  - A fenti bejegyzéshez tartozó következő csomópont IP-címe
  - **Alapértelmezett átjáró (default gateway, DG)** név nem szerencsés
    - Átjáró: általában alkalmazás rétegbeli továbbító
    - Helyette inkább a router / útválasztó/ útvonalválasztó
- Nem feltétlenül van ilyen
  - Ha másra nem illeszkedik, akkor a célhálózat ismeretlen, így a csomagot eldobja
- Végpontokon gyakran csak két bejegyzés szerepel:
  - Helyi hálózat (helyi végpont közvetlenül elérhető)
  - Alapértelmezett útvonal (minden más távoli hálózaton)

Metrikák szerepe az útválasztásban

- Metrika (mérték): Egy számérték, amely a hálózati utak közötti preferenciát adja meg

- Metrika alapja lehet például:
  - Elérhetőség
  - Terheltség
  - Késleltetés

- Metrika típusa
  - Statikus
    - Manuálisan megadott
  - Dinamikus
    - A link vagy a hálózat állapotától függően automatikusan változó

- A jobb metrikával rendelkező kapcsolaton küldjük ki a csomagot!

Megfelelő bejegyzés kiválasztva

Hálózat címe	Alhálózati maszk	Interfész	Közvetlenül kapcsolódó	Következő csomópont
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>
<IP-cím>	</N>	<azonosító>	<igen/nem>	<IP-cím>

- Mi a stratégia, ha
  - Közvetlenül (nincs következő csomópont!)

  vagy
  - nem közvetlenül

  kapcsolódik a hálózat?

Teendők helyi és távoli hálózat esetén

- Közvetlenül kapcsolódó (helyi) hálózat
  - A címzettnek közvetlenül küldeni
    - Ehhez a címzett **adatkapcsolati rétegbeli címére** van szükség
- Nem közvetlenül kapcsolódó (távoli hálózat)
  - Az útválasztónak kell küldeni
    - Az IP-címet tilos módosítani
    - Csak **adatkapcsolati rétegben** kell az útválasztónak címezni
      - Ehhez szükség van az adatkapcsolati címére

IP – ARP/RARP

Kapcsolat az adatkapcsolati címek felé

2014.Március 27.

ARP – Address Resolution Protocol

- Ha ismerjük az IP-címet, és a hozzá tartozó adatkapcsolati rétegbeli címre van szükség
  - RFC826
- Szinte minden adatkapcsolati és hálózati réteget támogat
  - Token Ring, FDDI, ATM, IEEE802.3, IEEE802.11
- Működése:
  - **Broadcast** kérés: „Kinek az IP-címe a …?”
  - Az IP-cím tulajdonosa válaszol
- Az ARP üzeneteket közvetlenül az adatkapcsolati réteg protokolljának küldjük (**Nem IP-csomag**!)

Az ARP fejléce

0 - 15 bits		16 - 31 bits
Hardware Type		Protocol Type
HLen (1 byte)	PLen (1 byte)	Operation
Sender HA (bytes 1 - 4)		
Sender HA (byte 5- 6)		Sender PA (byte 1 - 2)
Sender PA (byte 3 - 4)		Target HA (byte 1 -2)
Target HA (bytes 3 - 6)		
Target PA (bytes 1 - 4)		
RARP header structure		

- Hardware Type
  - Ethernet: 0x0001
- Protocol Type
  - IP: 0x8000
- HLen
  - Ethernet: 6 byte
- PLen
  - IPv4: 4 byte
- Operation
  - ARP request = 1
  - ARP reply = 2
  - RARP request =3
  - RARP reply =4
- Sender Hardware Address
  - <MAC-cím>
- Sender Protocol Address
  - <IP-cím>
- Target Hardware Address
- Target Protocol Address

Névfeloldás ARP-vel

- „A” szeretné megtudni „B” MAC-címét
  - 1. „A” üzenete
    - Operation: ARP Request = 1
    - Sender HA: <A MAC-címe>
      - **Megegyezik a keret fejlécében** találhatóval
    - Sender PA: <A IP-címe>
    - Target HA: 00:00:00:00:00:00 (ismeretlen)
      - A keret MAC címében FF:FF:FF:FF:FF:FF !!! (broadcast)
    - Target PA: <B IP-címe>
  - 2. „B” válasza
    - Operation: ARP Reply = 2
    - Sender HA: <B MAC-címe>
      - Most már nem 00:00:00:00:00:00 (ismeretlen)
      - Megegyezik a keret fejlécében találhatóval
    - Sender PA: <B IP-címe>
    - Target HA: <A MAC-címe>
    - Target PA: <A IP-címe>

ARP tábla

- Címpárok: Adatkapcsolati rétegbeli és IP-címek
- Két bejegyzéstípus
  - Statikus
    - Manuálisan felvitt bejegyzés
  - Dinamikus
    - ARP címfeloldás eredménye
    - **Gyorsítótár (cache)** funkció: ne kelljen mindig lekérdezni
    - Egy idő után elévül és törlődik

HW-cím	*IP-cím*	*típus*
<MAC-cím1>	<IP-cím1>	statikus
<MAC-cím2>	<IP-cím2>	dinamikus

ARP egyéb alkalmazásai

- **ARP próba (probe)**
  - Mielőtt használni kezdene egy IPv4 címet, felderíti, hogy nem használja-e más
  - Broadcast ARP kérés, küldő IP címe csupa nulla, saját címét kérdezi le
  - *IPv4 Address Conflict Detection* (RFC 5227)

- **ARP hirdetmény**
  - Ha változik MAC vagy IP címe, küld egy *gratuitous ARP* üzenetet, hogy mások tudják frissíteni az ARP táblájukat
  - TPA=SPA és THA=0 (request) vagy TPA=SPA, THA=SHA (reply)
    - Pl. operációs rendszer betöltésnél ha változott a hálózati kártya
    - vagy terheléselosztásra (több hálókártyánál)

RARP – Reverse ARP

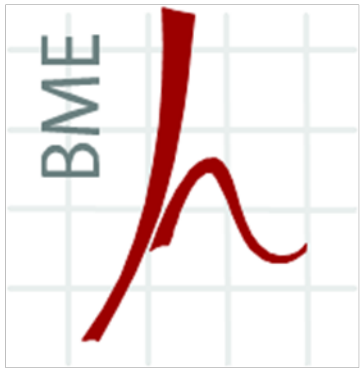
- Adatkapcsolati rétegbeli címből IP-címet

- Felhasználási terület
  - Hálózatmenedzsment
  - Permanens tár nélküli eszközök
    - Semmi ismeretük nincs a hálózatról
    - Hálózatról töltődik be az operációs rendszer
    - IP-cím nélküli kezdeti kommunikáció

- RARP kiszolgálók
  - RARP broadcast üzenetekre válaszolnak
- Nem használják már, helyette:
  - Bootstrap Protocol (BOOTP): hálózatról történő betöltésre
  - Dynamic Host Configuration Protocol (DHCP): IP-cím kérésére

Útválasztó tábla karbantartása

- Manuálisan
  - Minden hálózati operációs rendszer által támogatott
- Automatikusan
  - A tábla felépítéséhez szükséges információkat az útválasztók megosztják egymással
    ⇨ útválasztó protokollok (routing protocols)



IP – Útvonalválasztó protokollok

Útvonalválasztó táblák építése automatikusan

2014.Március 27.

Útválasztó protokollok feladatai

- Útvonal-irányítási információk begyűjtése
- Hurokmentes útvonal-irányítás
- Újabb csomópontok és hálózatok csatlakoztatásának biztosítása
- Csomópontok és hálózatok leválasztásának kezelése

Útvonalválasztó protokollok funkcionális osztályozása

- Ad hoc routing protokollok
  - Kis és gyorsan változó hálózatokra
    - Szenzor és egyéb vezeték nélküli hálózatokra
- Interior Gateway Protocols (IGPs)
  - Autonóm rendszereken (AS) belül
    - Kisebb hálózatokon
- Exterior Gateway Protocols (EGPs)
  - Autonóm rendszerek (AS) között
    - Az Internet routing protokollja

Ad hoc routing protokollok osztályozása

- Működési mód szerint
  - Proaktív
    - Folyamatosan karbantartott táblákkal
  - Reaktív
    - Igény szerinti célfelderítés
  - Hibrid
    - A fenti kettő együttes alkalmazása

- Alkalmazási terület szerint
  - Hierarchikus
  - Földrajzi elhelyezkedés alapján
  - Multicast
  - Energiatakarékos

IGP routing: A „távolság-vektor” módszer

- A gyűjtött információ és a gyűjtés módja:

  A csomópontok elmondják a hálózatról alkotott elképzeléseiket a szomszédaiknak

- Az „elképzelések”:
  - melyik csomópont milyen távol van
  - egy lista (vektor) melynek elemei
    - csomópont azonosító – távolság párok

- A távolság-vektorát közli mindegyik csomópont valamennyi szomszédjával

IGP routing: Az „összekötés-állapot” módszer

- A gyűjtött információ és a gyűjtés módja:

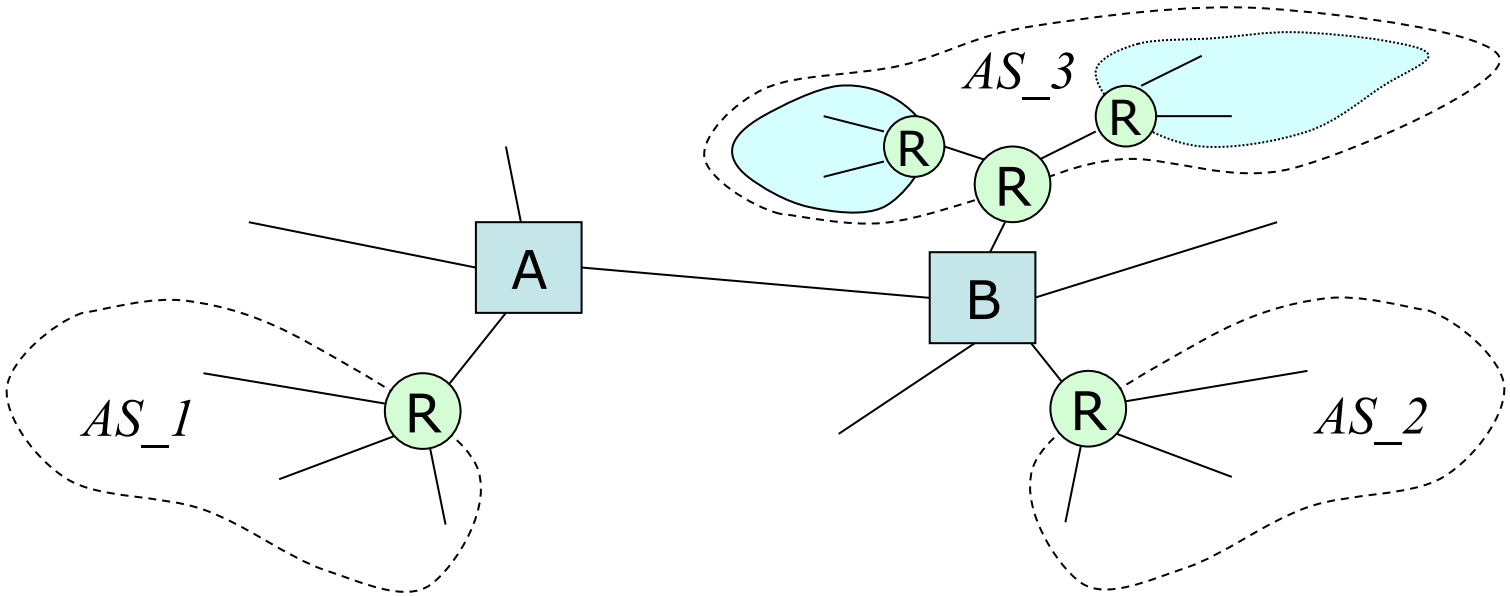
  A csomópontok elmondják mindenkinek a szomszédaikról nyert tapasztalataikat

- A „tapasztalatok”:
  - a szomszédokhoz vezető linkek **aktuális állapota** (ez pontosan ismerhető)

- Az információ elküldése „mindenkinek” a korlátozott (felügyelt) *elárasztás*-sal:
  - elküldik a szomszédokhoz, akik továbbadják
  - kivéve azon a linken, amelyen érkezett

Autonóm rendszerek

- Egy autonóm rendszer a többi autonóm rendszer számára jelzi az általa képviselt csoporto(ka)t
- Az egész csoport egyetlen bejegyzés lesz az útvonal-táblában



Interior Gateway Protocols (IGPs)

- Autonóm rendszereken (AS) belül
  - Távolság-vektor:
    - RIP (Routing Information Protocol)
    - IGRP (Interior Gateway Routing Protocol)
    - EIGRP (Enhanced Interior Gateway Routing Protocol)
  - Összekötés-állapot:
    - OSPF (Open Shortest Path First)
    - IS-IS (Intermediate System to Intermediate System)

Alkalmazásuk

- **RIP:** nem preferált
  - Metrika: **hopszám**
  - Lassan konvergál, nem skálázódik jól (nincs hierarchia)
    - Update: 30s
  - **Hopszám limit** korlátozza a hálózat méretét
    - Max 15
  - De könnyen konfigurálható

- **IGRP**: Cisco specifikus
  - Javította RIP hibáit
    - Max 255 hop
    - **Többféle metrika** figyelembe vétele
    - Update: 90s
    - Már nem használják
- **EIGRP:** IGRP javítása - **Variable Length Subnet Mask** használata
  - Gyorsabb konvergencia
  - Hurokmentes működés
  - **6 metrika kombinációja**

Alkalmazásuk

- Elterjedtebbek az összeköttetés-állapot módszerek
  - OSPF: nagy céges hálózatokban
    - Link state database (LSDB) üzenetek elárasztással
    - Nagyon gyorsan konvergál
    - **Többféle metrika**
    - **Hierarchikus**: gerinc - routing tartományok, köztük határ routerek
  - IS-IS: nagy kiterjedésű ISP hálózatokban

OSPF vs. IS-IS

- OSPF (2-es verzió):
  - IPv4 forgalom irányítására tervezték: népszerűbb
    - több fejlesztés, alkalmazás
  - Át kellett specifikálni IPv6-ra (OSPFv3)

- IS-IS:
  - OSI fejlesztés, nem IP csomagokban
  - Nagyobb kiterjedésű hálózatokra (ISP)
    - Több routert támogat adott területen
  - Mivel nem IPv4 specifikus, könnyebb átállás IPv6-ra

Exterior Gateway Protocols (EGPs)

- Autonóm rendszerek (AS) között
- Példák:
  - EGP (Exterior Gateway Protocol)
    - Sokáig az Internet EGP-je, ma már nem használt
  - BGP (Border Gateway Protocol)
    - 1995-től használt EGP (BGPv4)

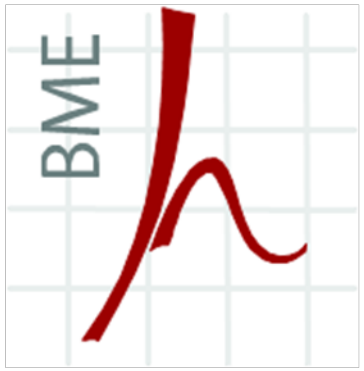
- Címaggregáció (supernet)
  - Kisebb routing táblák
    - Gyorsabb keresés
    - Gyorsabb terjesztés

BGP routing alapelv

- **Path vector protocol (Útvonal-vektor protokoll)**

- Routing tábla bejegyzés:
  - Célhálózat
  - Következő router
  - **Útvonal a célhálózatig**

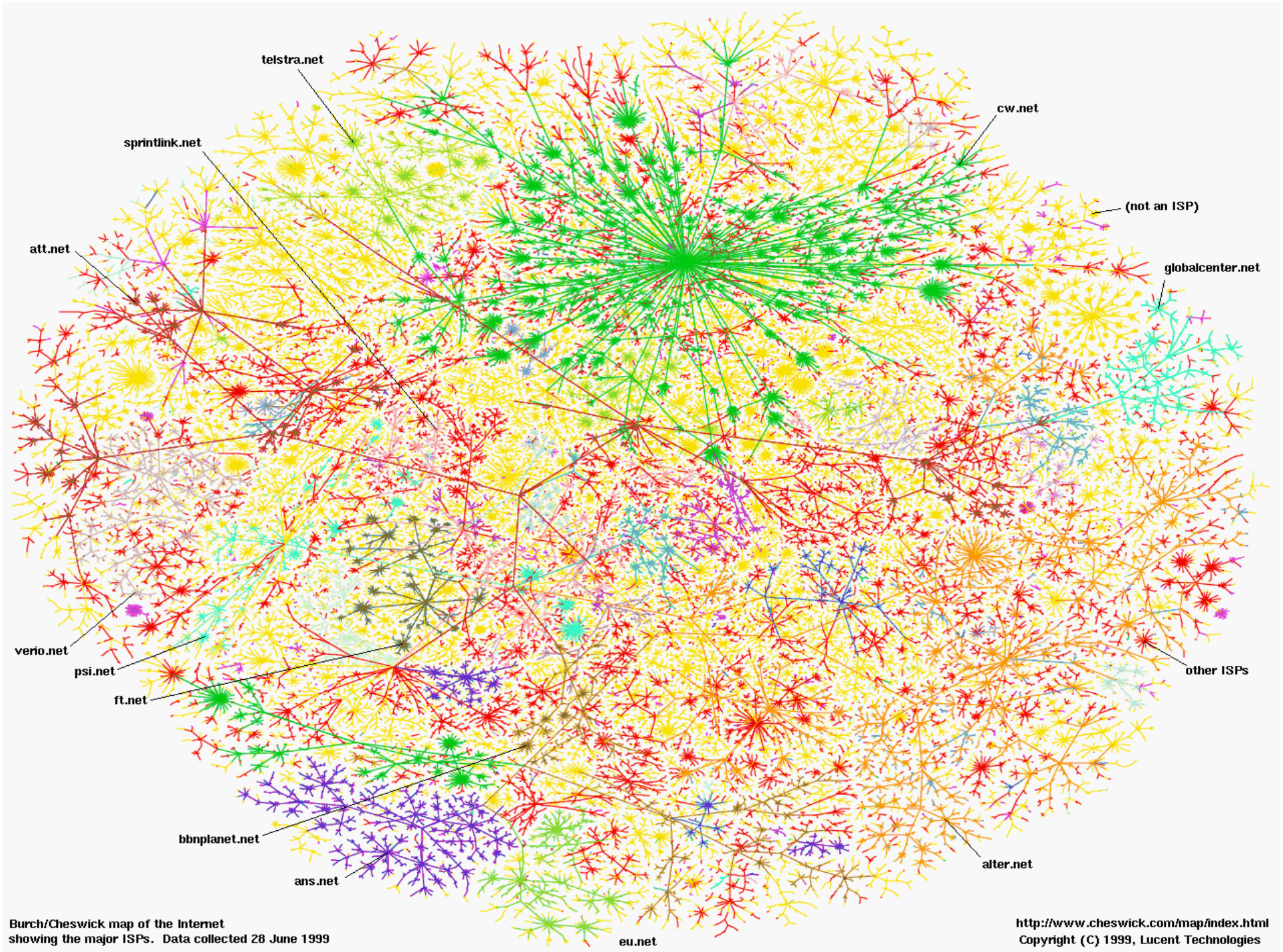
- A **teljes útvonal** dinamikus nyilvántartása
  - AS határ routerek végzik
    - útvonal-vektor üzenetekkel frissítés, hirdetve a többi AS elérhetőségét
  - **Útvonal-vektor üzenet**: **AS-ek sorozata a célhálózat felé**
  - Minden AS határ router fogadáskor megnézi megfelel-e a saját szabályainak
    - Ha igen, módosítja a routing tábláját és az útvonal-vektor üzenetet
    - Következő router: saját ID-je, illetve útvonal-vektorba AS száma



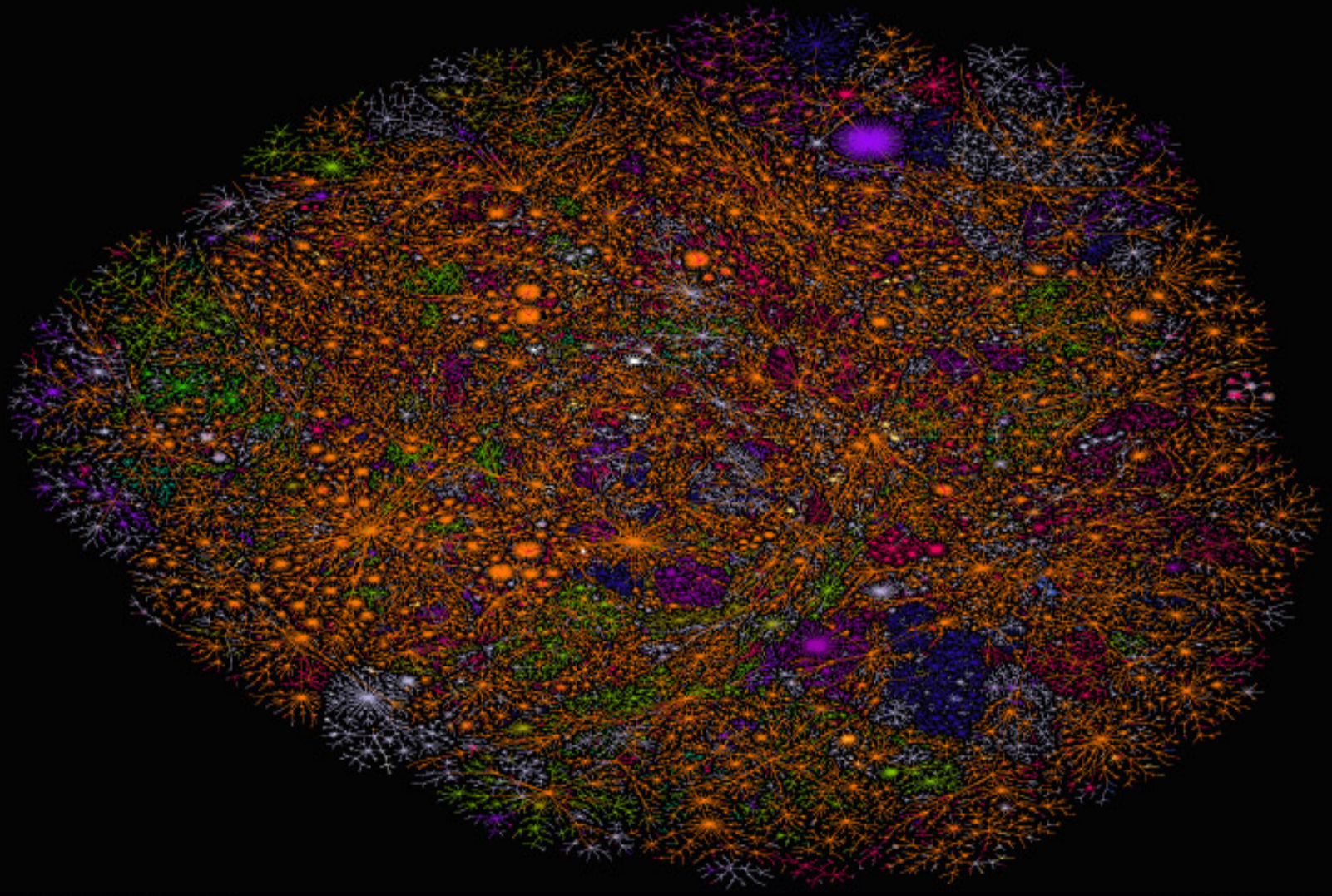
Hogyan néz ki az Internet?

Valóban decentralizált?

2014.Március 27.


telstra.net
cw.net
sprintlink.net
(not an ISP)
att.net
globalcenter.net
verio.net
other ISPs
psi.net
ft.net
bbnplanet.net
alter.net
ans.net
eu.net
Burch/Cheswick map of the Internet
showing the major ISPs. Data collected 28 June 1999
http://www.cheswick.com/map/index.html
Copyright (C) 1999, Lucent Technologies

The Internet: 2002



North America
Central America
South America
Africa
South Africa
Europe
Germany
France
Netherlands
United Kingdom
Asia
Japan
Pacific Islands
Australia
New Zealand
.mil .nato
.gov
.com
.edu
.org
.net
other

Published by
www.peacockmaps.com

IP - Tördelés

Darabolás és összeállítás

2014.Március 27.

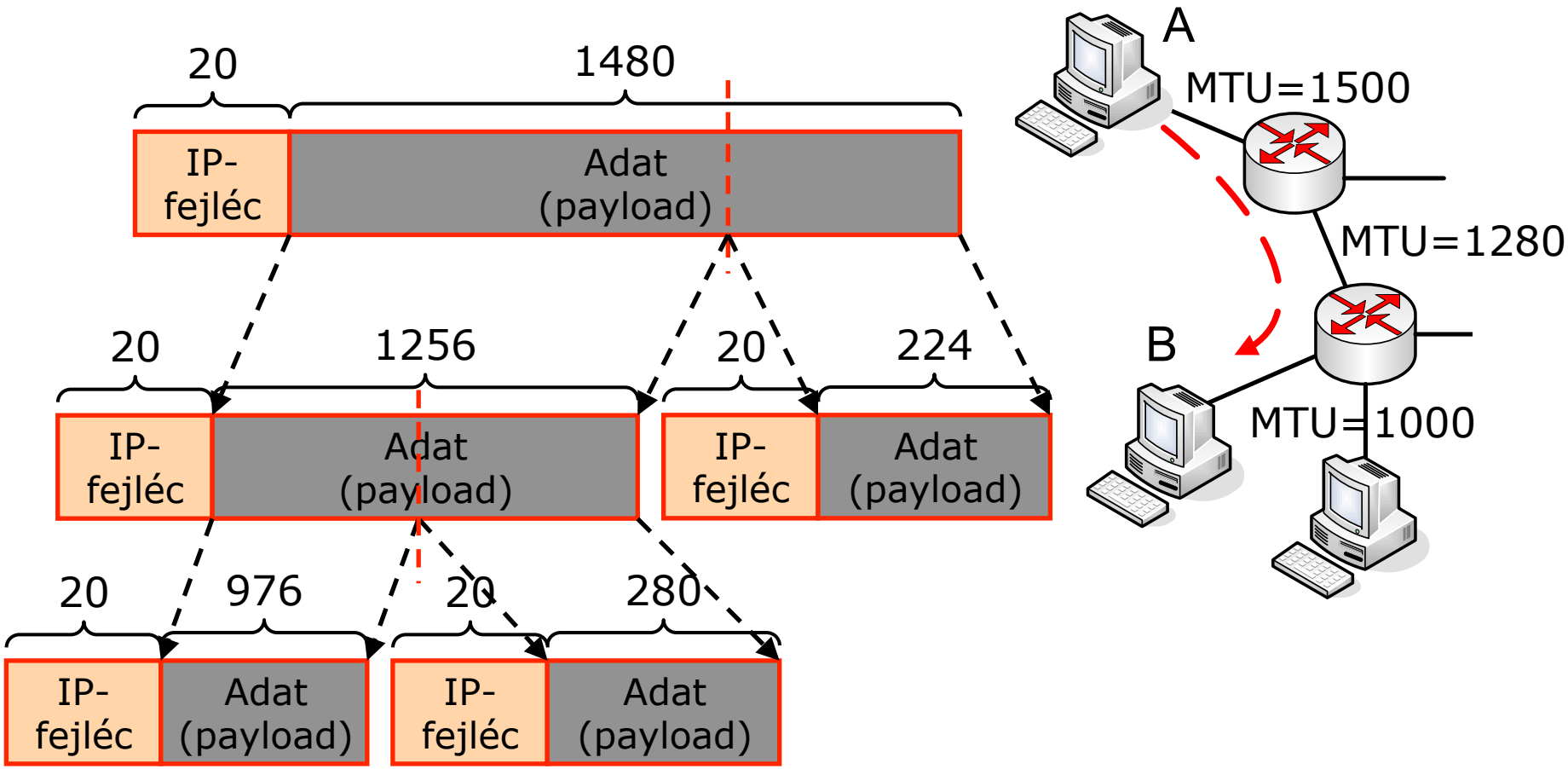
Tördelés szükségessége

- A hálózatok alsóbb rétegei meghatározzák a keret maximális méretét
  - Az adatkapcsolati réteg fej- és farokrészét leszámítva ez az MTU (Maximum Transmission Unit)
    - Ethernetnél 1500 byte

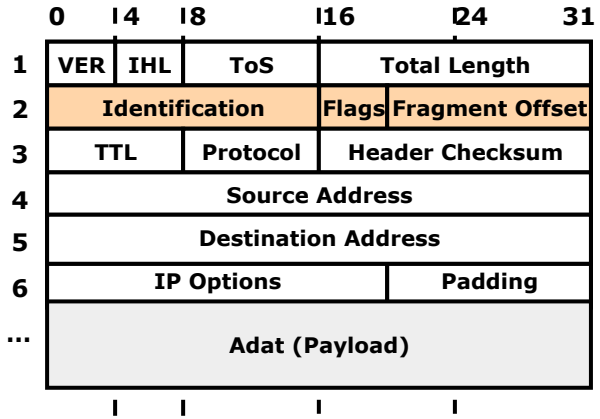
- Eltérő technológiák ⇨ eltérő MTU-jú kapcsolatok ⇨ tördelni kell

A tördelés menete

- Minden csomagnak kell fejléc (itt minimális fejléccel)!



Tördeléssel kapcsolatos mezők az IP-fejlécben



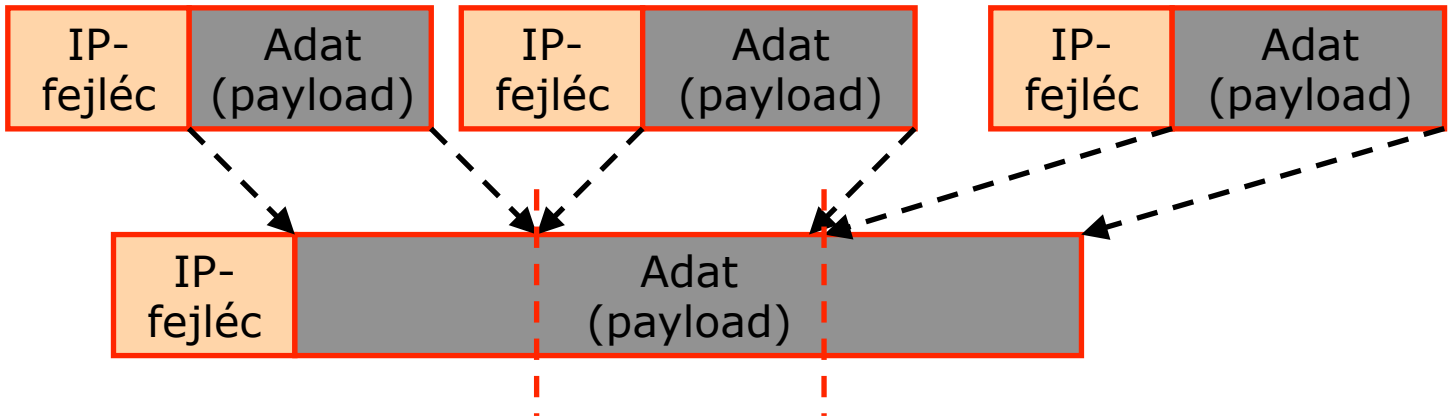
- Identification (16 bit)
  - Az IP-töredékek egyedi azonosítása
- Flags (3 bit)
  - 0: Fenntartott
    - 0-nak kell lennie
    - „Evil bit” (RFC 3514) (áprilisi tréfa 2003-ban)
  - 1: DF – Don’t Fragment
    - 1: ha tördelni kellene, el kell dobni
    - Ezt használják az MTU (Maximum Transmission Unit) Path Discovery néhány TCP verzióban és az IPv6-ban
  - 2: MF – More Fragment
    - 1: ha nem az utolsó töredék
    - 0: utolsó töredék vagy nem tördelt csomag
- Fragment Offset (13 bit)
  - Az eredeti csomagban lévő kezdőpozícióját adja meg e töredékben lévő adatnak **8 bájtos egységekben**
  - $(2^{13}-1)x8=65528 > 65515$ bájt (max. adat)
    (így a maximális méretű csomag is tördelhető)

IP-fejléc változása tördelés közben

- Ha tördelés szükséges:
  - Total Length mezőt a töredék méretére állítani
  - Ha nem volt Identification mező érték, akkor generálni kell
  - Adat tördelése 8 bájtos egységekre
  - A tördelésnek megfelelő fragmentation offset-et beállítani az új csomagokban
  - Minden csomagban az MF bitet 1-re kell állítani, kivéve annak a csomagnak az utolsó töredékénél
  - Ellenőrző összeg változása

A töredékek összeállítása

- Csak a címzett végezheti el
  - Kivéve pl. NAT
- Ha valamely darab nem érkezik meg, akkor a többit is eldobja
- Csak teljesen összeállított csomagokat továbbít a felsőbb réteg felé
- A „fragment offset”-ekből a töredék helye meghatározható



A router feladatai

- Hibás-e a csomag (fejléce)?
- Nekem címezték-e?
- Ismerem-e a címzett hálózatát?
- A TTL érték csökkentés után >0?
- Kell-e tördelni? Lehet-e tördelni?
- Kell-e visszajelzést küldeni?

ICMP és IGMP

Az IP jelzés- és menedzsmentüzenetei

- *ICMP: Internet Control Message Protocol*
- *IGMP: Internet Group Management Protocol*

2014.Március 27.

ICMP - Internet Control Message Protocol

- Jelzés- és menedzsmentüzenetek
  - Visszajelzés a „best effort” hálózattól
- IP felett (0x01-es protokollazonosító)
- Üzenettípusok
  - Hibaüzenetek
  - Kérdések
  - Válaszok
- Gyakran használt ICMP üzenetek:
  - Echo request – Echo reply ⇨ Ping (RTT)
  - Echo request – Echo reply + TTL ⇨ Traceroute

ICMP Message Types	
0	Echo Reply
3	Destination Unreachable
4	Source Quench
5	Redirect/ Change Request
8	Echo Request
9	Router Advertisement
10	Router Selection
11	Time Exceeded
12	Parameter Problem
13	Timestamp Request
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply

ICMP csomag és fejléce

- IP csomagba ágyazva, IPv4 fejléc után következik a fejléce (8 byte)
- Így best-effort szolgáltatás, de különleges elbánást kap a feldolgozásnál
  - Kötelező értesíteni a forrás címet, amely okozta a hibaüzenet generálását

- Fejléc:
  - Típuson belül kód, pl. típus: *címzett elérhetetlen*, ezen belül kód:
    - *címzett hálózat/host/protokoll/port elérhetetlen*
    - *tördelésre van szükség*
    - *letiltott hálózat/host*
  - Ellenőrzőösszeg egész ICMP üzenetre
  - Hibaüzenetnél: a hibás üzenet fejléce és adatmezőjéből az első 8 byte

ICMP – Cél elérhetetlen

- Gyakran előforduló üzenet:
  „Destination unreachable” (ICMP Message Type: 0x03)

  Hibakódok:

0 = net unreachable
1 = host unreachable
2 = protocol unreachable
3 = port unreachable
4 = fragmentation needed and DF set
5 = source route failed
6 = destination network unknown
7 = destination host unknown
8 = source host isolated
9 = communication with destination network administratively prohibited
10 = communication with destination host administratively prohibited
11 = network unreachable for type of service
12 = host unreachable for type of service

Traceroute

- Először TTL=1, majd TTL=2 ...
- Sorban eldobják a routerek és ICMP üzenetet küldenek róla
  - Time Exceeded
- Megadja a csomópontok listáját, amelyeket érint a csomag a célig
- Nem mindig a realitást tükrözi a traceroute lista
- Alkalmazás:
  - Hibaelhárításra
  - Tűzfal felderítésre
  - Letöltésnél (mirror)
  - Penetráció tesztelésre
- Hackerek is szeretik☺

További érdekes ICMP üzenetek

- Source Quench: felszólítás a küldőnek a forgalma visszafogására
- Redirect: felszólítás a küldőnek másik útvonal használatára
- Address Mask Request: host lekéri a router-től az alhálózati maszkot

IGMP – Internet Group Management Protocol

- IP-t futtató csomópont-csoportok kezelésére
  - Pl. online streaming video, játékok
  - IP csomagokban
- Főként **multicast csoportok** kezelése
  - Csoportok lekérdezése
  - Csoporthoz csatlakozás
- A TTL értéke általában 1
  (csak a **helyi hálózaton** érvényes)
- Csak IPv4-hez, IPv6 máshogy oldja meg

Összefoglalás

- Bevezetés
  - A TCP/IP protokollarchitektúra
  - Az IP feladata és jellemzői
- IP címzés
  - Címosztályok
  - Speciális címek
- Az IP fejrész szerkezete
- Routing protokollok
- Routingtáblák, subnet mask
- Továbbítás
- Címfeloldás: ARP, RARP
- MTU, tördelés
- Segédprotokollok: ICMP, IGMP

Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

IPv6

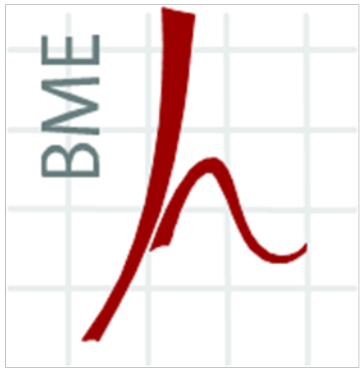
A következő generációs Internet Protocol

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2014.Április 3.

IPv6 - Áttekintés

- Motivációk – az IPv4 hibái
- Címzés
- IPv6-fejléc
- Migráció és együttélés
- IPv6 biztonság



Motivációk

Az IPv4 hibái

2014.Április 3.

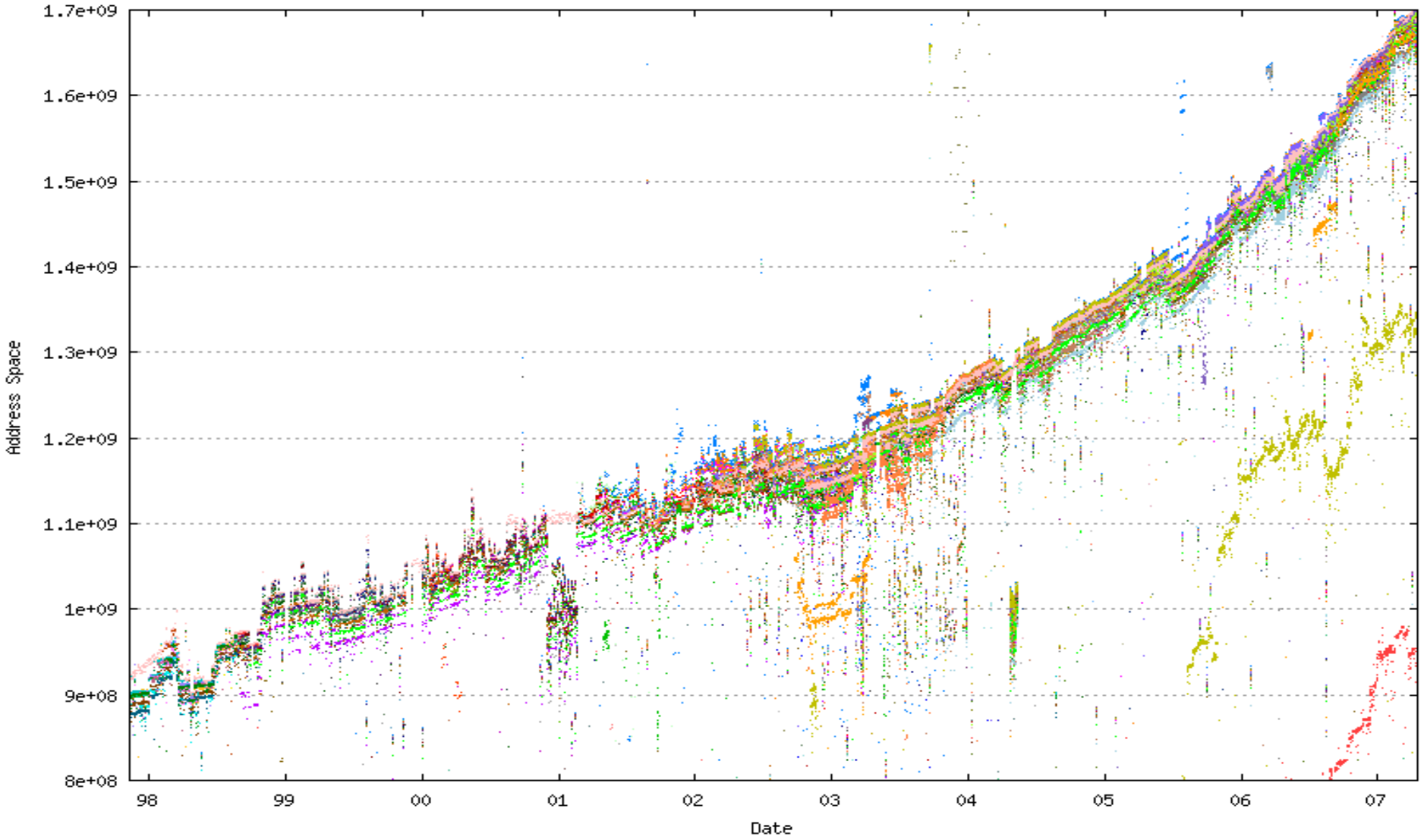
Motiváció – problémák az IPv4-gyel

- 32 bites címtartomány kimerülése
- Erőforrás-igényes
  - Sok felesleges mező a fejlécben
  - Tördelés a köztes csomópontokban
- Nem biztonságos
  - Nem támogatja a hitelesítést és a titkosítást
- Mobilitás támogatás csak külön protokollal

Címtartomány kimerülés okai

- Egyre nagyobb az Internet elterjedtsége
  - Fejlődő népes országok (Kína, India, Afrika)
- Egyre több eszköz
  - Mobiltelefonok: 4G-ben már IPv6!
  - Szenzorok (intelligens otthon, biometrikus ~)
  - TV-készülékek, hűtők, mosógépek, …
- Pazarló címfelhasználás
- Always-on elterjedése
  - dial-up helyett
- Virtualizáció

Kiosztott címtartományok tere



Híres ember híres kijelentése

„32 bits should be enough address space for Internet”

- Vint Cerf, 1977 (Honorary Chairman of IPv6 Forum 2000)

Hol tartunk most? Mennyi IP-cím szabad?

- Már a 80-as végétől felmerült a címkimerítés lehetősége
- Történelmi pillanat: 2011. februárjában osztotta ki a IANA az utolsó 5 blokkot az 5 RIR-nek
  - egyenként 16 millió címmel
- 2011 áprilisára az ázsiai-csendes-óceáni RIR kimerítette ezt a cím blokkját!

- 2001-ben 2007-re a címtartomány kimerülését jósolták
- Miért nem következett be?
  - NAT (Network Address Translation)
  - Privát IP-címtartományok
  - CIDR
    - VLSM
    - Címek újraosztása
  - RIR-ek, ISP-k IP-címgazdálkodása
  - Virtuális hosting
    - Név, IP illetve port alapú

IPv6 címzés

Hatalmas címtartomány újdonságokkal

2014.Április 3.

Milyen hosszú az ideális cím?

- 64 bites
  - Sokkal tovább elég, mint a 32 bites
  - Nem túl nagy növekedés az IP fejlécben
- Változó méretű 160 bitig
  - Kezdetben lehet 64 bites is, majd szükség esetén lehet növelni
- Megállapodás: 128 bites legyen!
  - Kompromisszum
  - 340 282 366 920 938 463 463 374 607 431 768 211 456 ≈ $3{,}4 \cdot 10^{38}$ cím kezelésére
  - Minden lakosra 2^{95} cím

IPv6 címek ábrázolása

- 128 bit
- 8x4 hexadecimális számként ábrázolják
  - FEDC:BA94:7654:3210:FEDC:BA98:7654:3210
- A vezető nullák elhagyhatók
  - FEDC:0094:0004:0000:000C:BA98:7654:3210
  - FEDC:94:4:0:C:BA98:7654:3210
- A 16 bites nullákat tartalmazó részek kihagyhatóak, ha egymás után vannak (maximum egy ilyen blokk hagyható ki):
  - FEDC:0000:0000:0000:000C:BA98:0000:3210
  - FEDC::C:BA98:0:3210
  - Pl. localhost: (0:0:0:0:0:0:0:1) helyett ::1

IPv6 címek ábrázolása

- Egyes IPv6-os címek IPv4-ből származnak. Ekkor megengedett:
  - 0:0:0:0:0:0:A00:1
  - ::10.0.0.1
- Hálózati címek (prefixek) jelölése
  - FEDB:ABCD:ABCD::/48
  - FEDB:ABCD:AB00::/40
- Hivatkozásként
  - http://[FEDC::C:BA98:0000:3210]/index.html
  - RFC 2732

Cím prefixek

0000 0000	Reserved
0000 0001	Unassigned
0000 001	Reserved for NSAP (non-IP addresses used by ISO)
0000 010	Reserved for IPX (non-IP addresses used by IPX)
0000 011	Unassigned
0000 1	Unassigned
0001	Unassigned
001	**Unicast Address Space**
010	Unassigned
011	Unassigned
100	Unassigned
101	Unassigned
110	Unassigned
1110	Unassigned
1111 0	Unassigned
1111 10	Unassigned
1111 110	Unassigned
1111 1110 0	Unassigned
1111 1110 10	**Link Local Use addresses**
1111 1110 11	**Site Local Use addresses**
1111 1111	**Multicast addresses**

Címtípusok az IPv6-ban

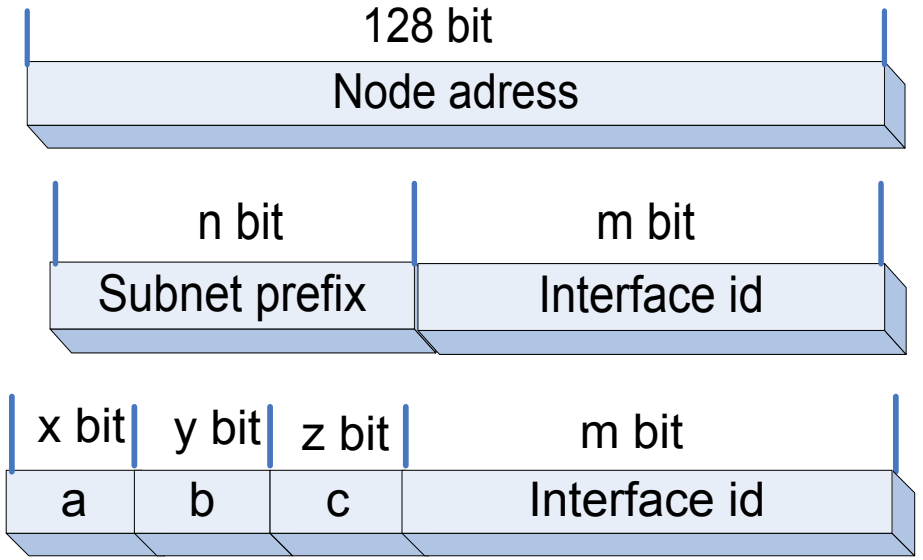
- Unicast
  - Az IPv4-hez hasonlóan
  - Egyedi cím: pontosan egy csomóponthoz tartozik
  - Minden IPv6 csomópontnak legalább egy ilyen címe van
- Multicast
  - Csoportot azonosít
  - Minden csoporton belüli csomópont megkapja az erre a címre küldött adatot
  - Broadcast helyett is ezt használjuk
- Anycast
  - Csoportot azonosít
  - Biztosított, hogy a csoport egy csomópontja megkapja az erre a címre küldött üzenetet
    - pl. a küldőhöz legközelebbi

Unicast címek

- Típusai:
  - Globális
    - Aggregálható

    vagy
    - IPv4 kompatibilis
  - Link local
    - Prefix után a hálózati cím végig nulla, nem route-olható
  - Site local (valószínűleg a gyakorlatban ezt nem fogják használni)
  - Beágyazott IPv4 címet tartalmazó

- Strukturált cím



Globális unicast cím

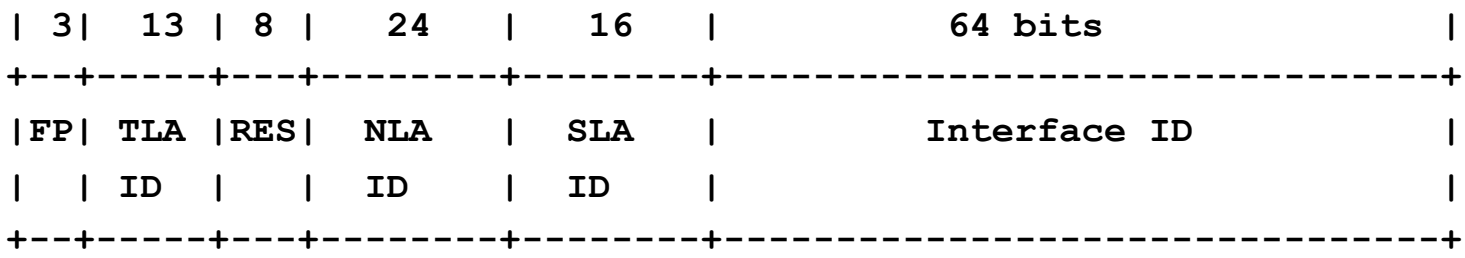
- Felépítése:

- A „ki vagy ” szétválasztása a „hová kapcsolódsz”-tól
  - Routing Prefix
    - Routing topológia
  - Subnet ID
    - Hálózati adminisztrátor
  - Csomópont azonosítás
    - Interface Identifier

- Interface ID
  - Megegyezés alapján 64 bit
  - Az adatkapcsolati rétegbeli címből képződik
    - Egyediséget biztosítja, ha hardverhez kötődik a cím
    - Ethernet MAC címből: 48 bit →64 bit (EUI-64)
  - Vagy DHCP szervertől random

Aggregálható globális unicast címzés

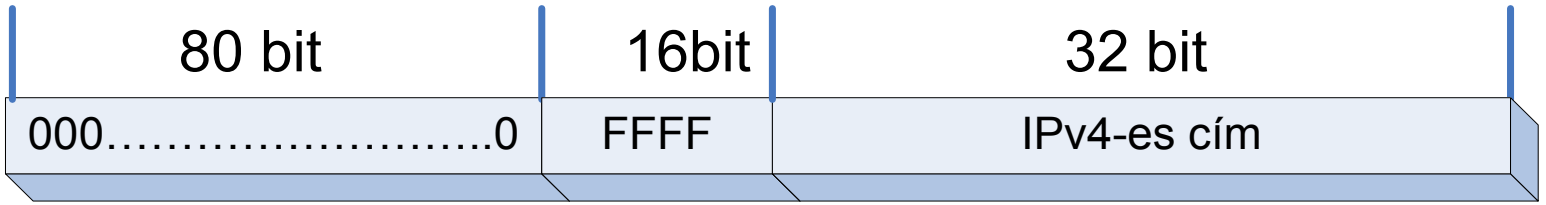
- Aggregatable Global Unicast Address Structure [RFC 2347]



- FP — Format Prefix (001)
- TLA ID — Top-Level Aggregation Identifier (régió)
- RES — Reserved for future use
- NLA ID — Next-Level Aggregation Identifier (szolgáltató)
- SLA ID — Site-Level Aggregation Identifier (előfizető és alhálózat)
- INTERF. ID — Interface Identifier

IPv4 kompatibilis címek

- 000-val kezdődők
  - IPv4 kompatibilis IPv6 címek
    - Pl. ::10.0.0.1
  - IPv6-ra képezett IPv4-os címek
    - Pl: ::FFFF:10.0.0.1

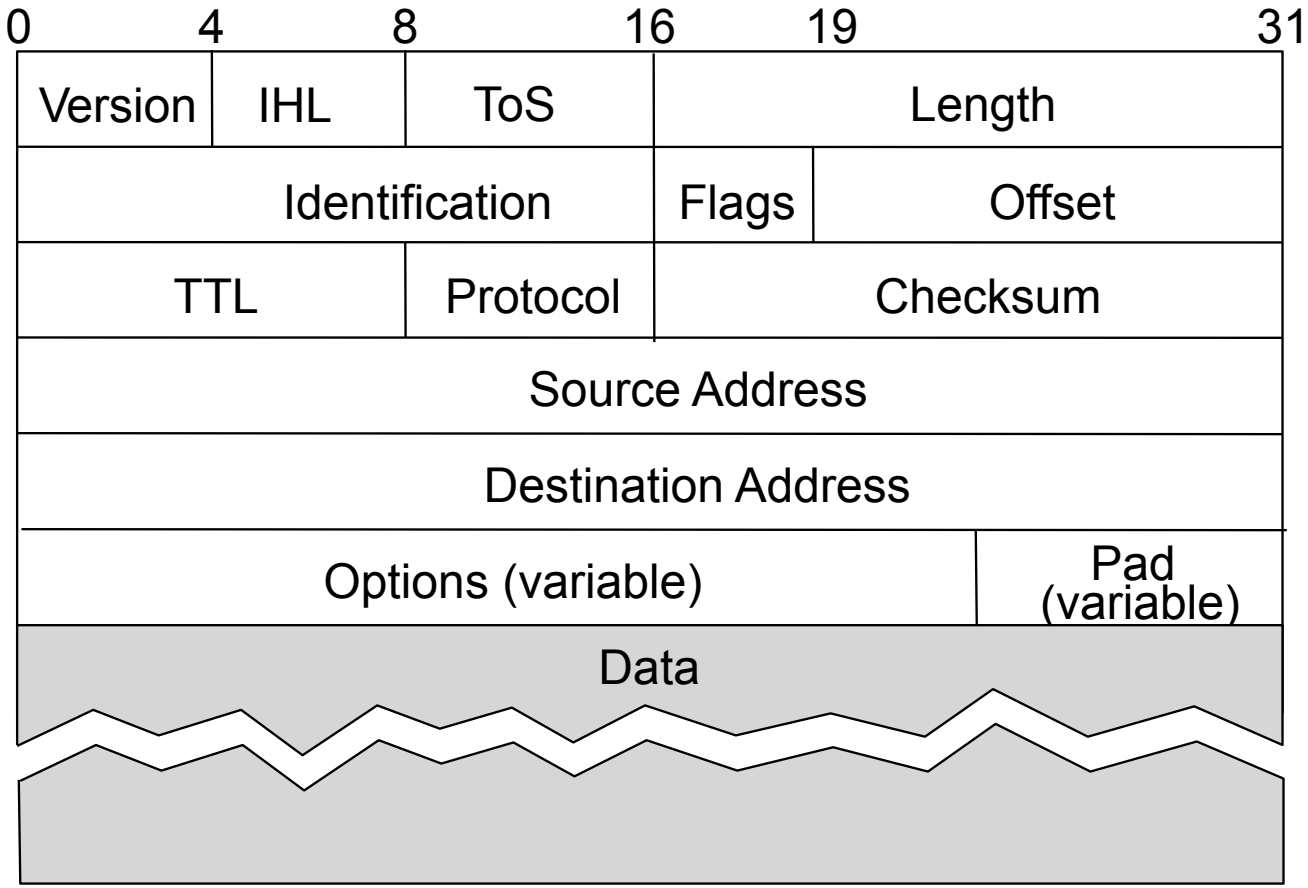


IPv6 csomag szerkezete

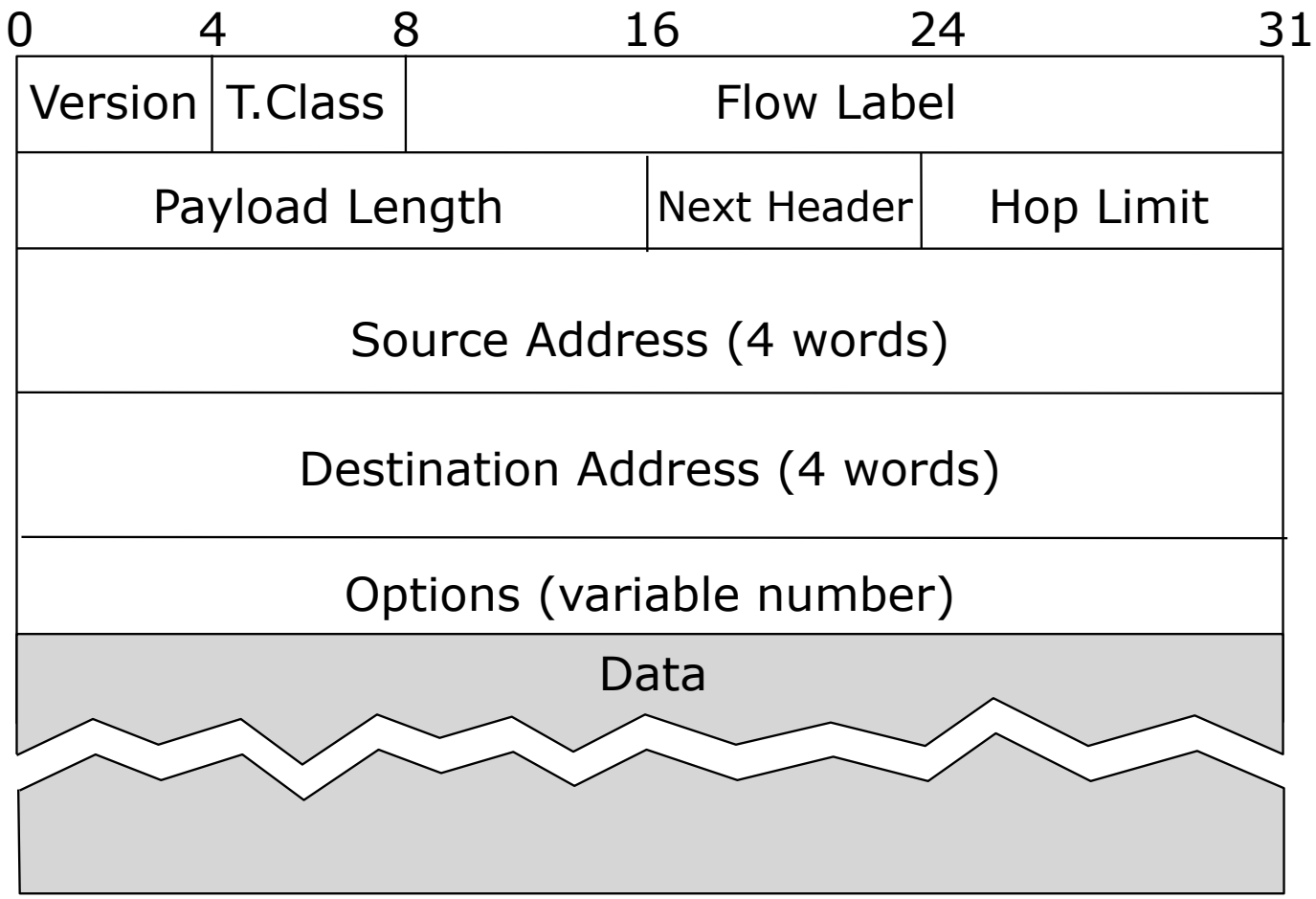
Egyszerűség és univerzalitás

2014.Április 3.

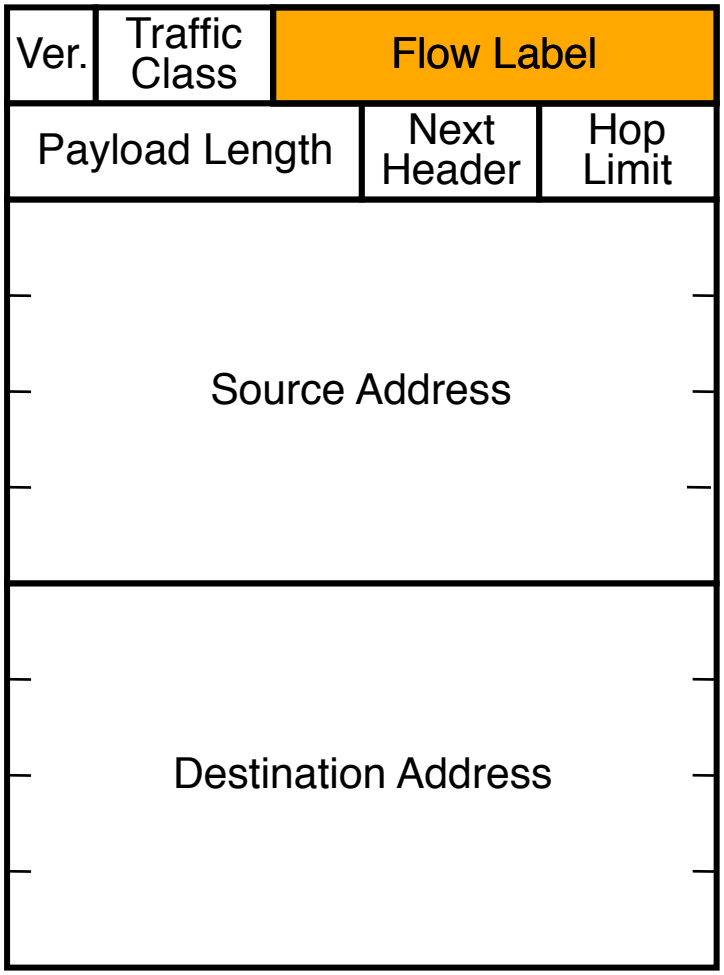
Az IPv4 csomag szerkezete

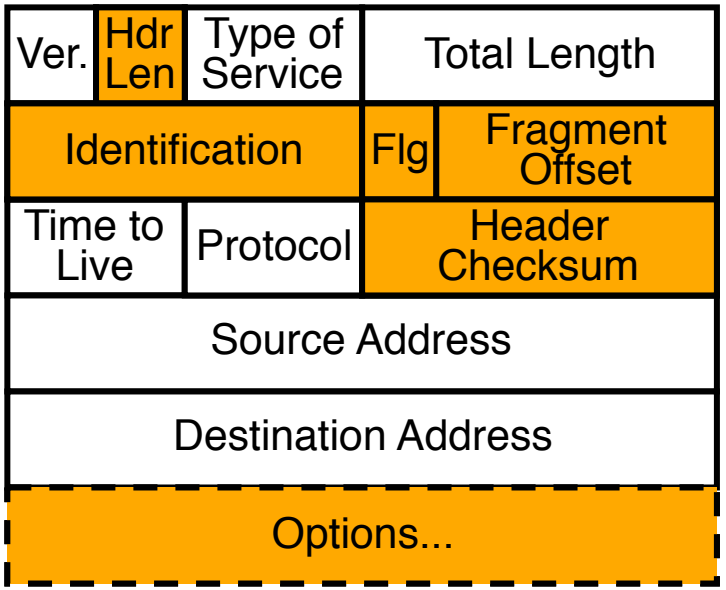


IPv6 csomag szerkezete



IPv6 és IPv4 fejrészek összehasonlítása





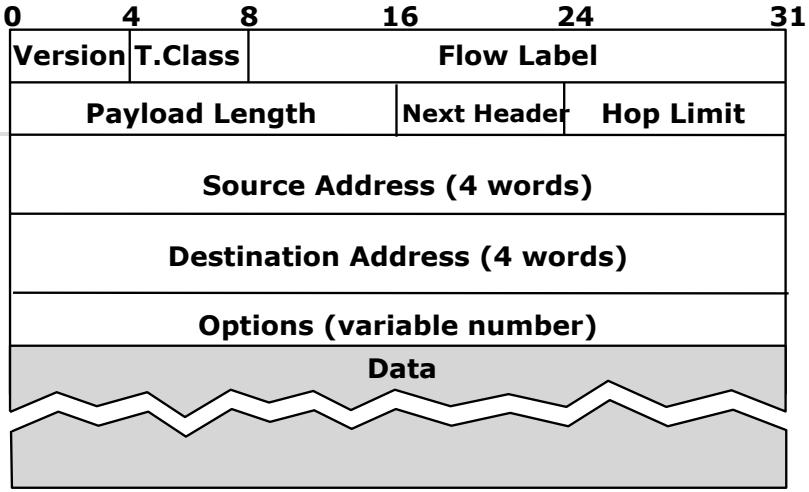
A kiemelt részek a másik verzióban nem találhatók meg!

IPv6 fejrész kétszer hosszabb (40 byte), mint a minimális IPv4-es (20 byte)!!!

Ami az IPv6 fejrészből kimaradt…

- Nincs ellenőrzőösszeg (checksum)
  - Nem kell minden routernek ellenőriznie
    ⇨ nő a feldolgozási sebesség
  - Általában kevés a hiba (jó minőségű kapcsolatok)
- Nem változik a fejrész mérete
  - Rögzített méret ⇨ nő a feldolgozási sebesség
  - Nem kell IHL mező ⇨ kisebb fejrész
- Nincs ugrásonkénti tördelés
  - ⇨ nő a feldolgozási sebesség
  - Nem kell ehhez használt mező ⇨ kisebb fejrész
  - MTU felderítés (Path MTU Discovery) kell!!!

Az IPv6 fejrész mezői I.

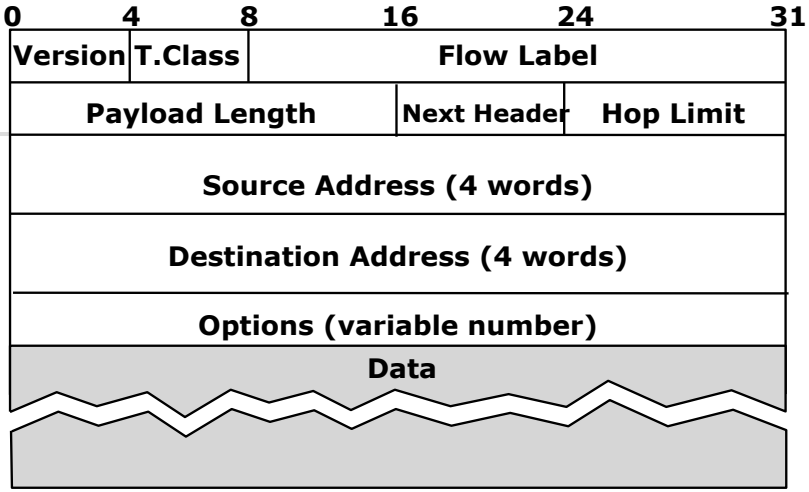


- Version – verzió (4 bit)
  - IPv6 esetén 6 (IPv4: 4)

- Traffic Class – forgalmi osztály
  - QoS lehetőség biztosítása az IPv4 Type of Service (ToS) mezőjével megegyező módon
  - Használt elnevezés még a Priority

- Flow Label – folyam azonosító címke (24 bit)
  - Adott kapcsolatot azonosító generált mező
    - Nem független datagrammok!
  - QoS és igazságos adatsebesség megosztást tesz lehetővé
  - Egy forrás-célállomás pár esetén **több folyam** is lehet
    - Adatfolyamot a **cím és a flow label együtt** azonosítja

Az IPv6 fejrész mezői II.



- Payload Length – adathossz
  - Nem tartalmazza a fejrészt
  - Maximum 64 KB-os csomag
    - de: jumbogram opció (4Gb)
- Hop Limit – ugrás korlát (8 bit)
  - Megegyezik az IPv4 TTL mezőjével
- Címmezők (2 x 128 bit)
  - 2 128 bites cím
  - A fejrész jelentős része
- Next Header – következő fejléc (8 bit)
  - 2 lehetőség:
    - A beágyazott PDU típusát adja meg
      - Hasonlóan az IPv4 Protocol mezőjéhez
    - Az IPv6 fejléc kiterjesztését jelentő „Extension header” típusát adja

IPv6 „opciók” – Header Extensions I.

Célja:

- Az egyszerű fejléc kiegészítése opcionális lehetőségekkel

Típusai:

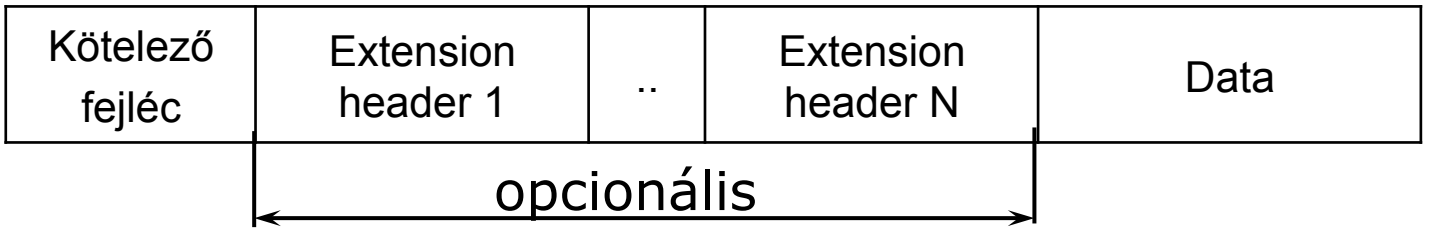
- **Hop-by-hop Options Header** (0)
  - **Minden közbülső** csomópont is meg kell, hogy vizsgálja
  - Jumbogrammok (óriás-datagrammok)
  - QoS támogatása
- **Routing Header** (43)
  - Routerek felsorolása, amelyeket útba kell ejteni
  - Az IPv4 opciókhoz hasonlóan
    - Laza forrás forgalomirányítás
    - Szigorú forrás forgalomirányítás

IPv6 „opciók” – Header Extensions II.

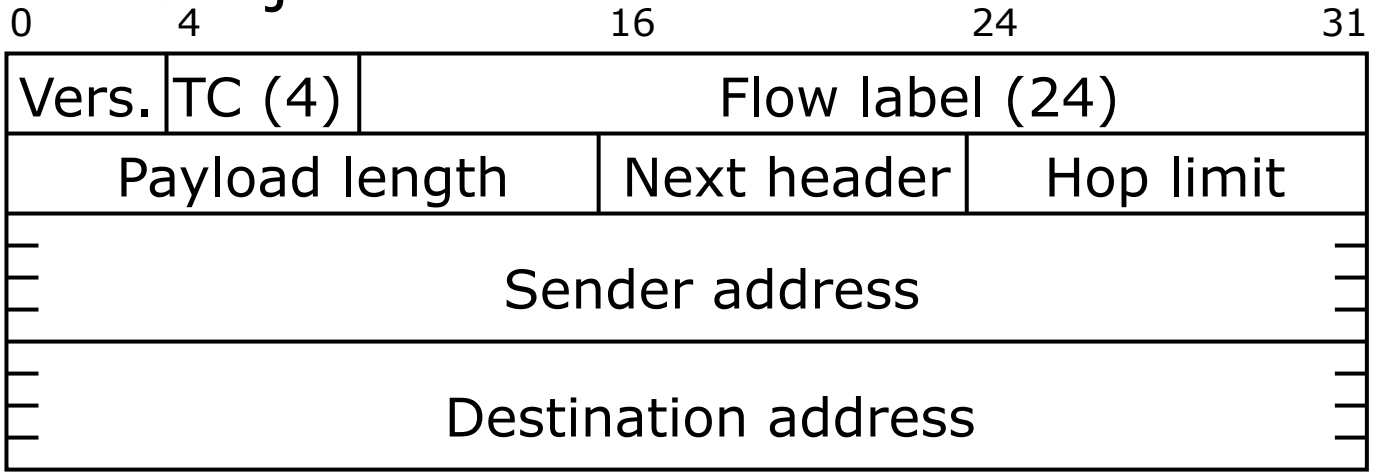
- Fragmentation Header (44)
  - Mint az IPv4-ben, de csak a forrás darabolhat
- Authentication Header (AH) (IPSec-ből) (51)
- Encapsulation Security Payload Header (ESP) (50) (IPSec-ből)

Az IPv6 fejrész kiterjesztése

Általános fejlécformátum

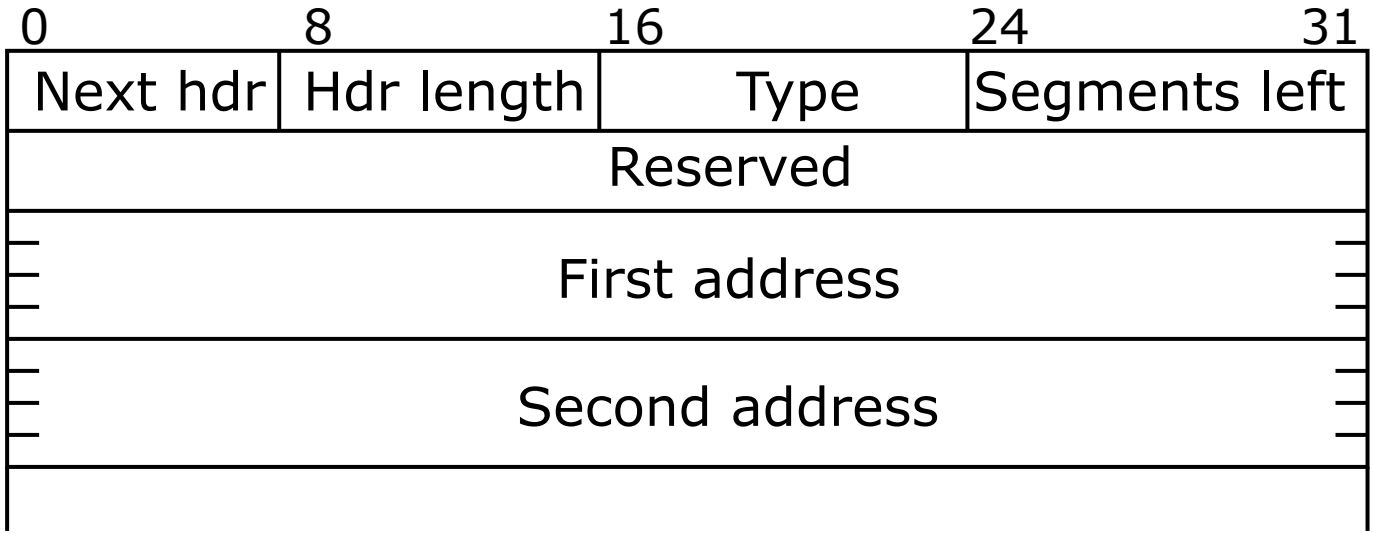


A kötelező fejrész



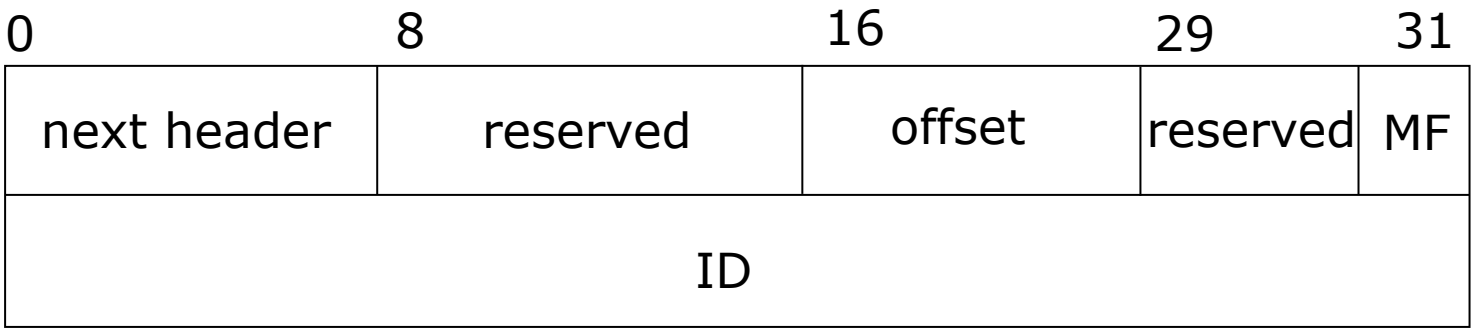
Routing Extension

- Az IPv4 opcióhoz hasonlóan
- Header length – fejrész hossza (64 bites szavakban mérve)
- Type - típus
- Az érintendő csomópontok IPv6 címét tartalmazó lista
  - Maximum 24 cím
  - Következő csomópont megtalálása anycast címzéssel



Fragmentation Extension

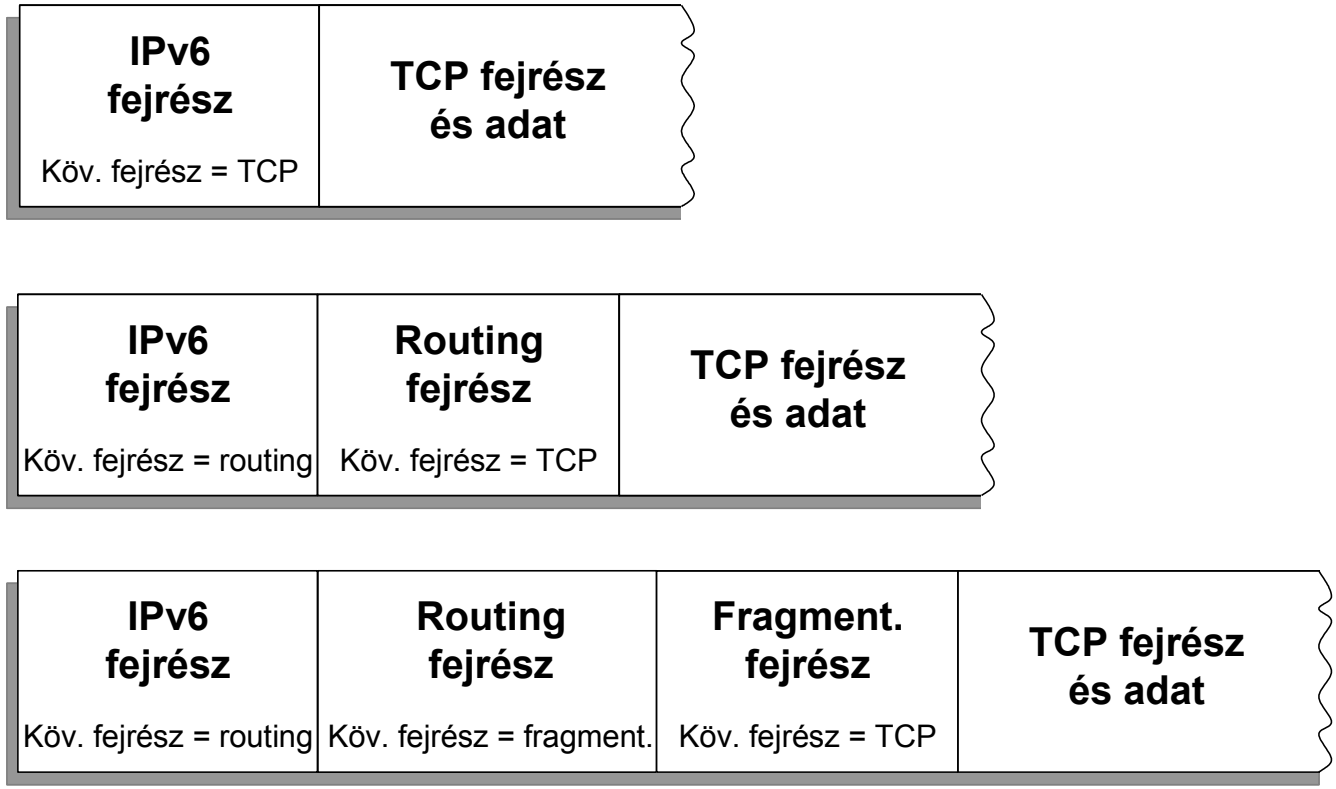
- Hasonló az IPv4-ben megtalálhatóhoz
  - 13 bites Fragmentation offset
  - More fragment bit (MF)
  - Már 2 nem használt bit
  - 16 bites helyett 32 bites Fragment ID mező, amely a töredéksorozatot azonosítja
- Tördelni csak a feladó fog
- Összeállítani csak a fogadó fog



IPsec az IPv6-ban

- Két Extension Header a biztonság növelésére
- IPSec-re minden IPv6 csomópontnak képesnek kell lennie
- *Authentication Header (RFC 2402)*
  - Valóban a látszólagos feladó küldte-e?
  - Lett-e módosítva a csomag?
- *Encapsulating Security Payload Header (RFC 2406)*
  - Titkosított a csomag tartalma

Példák az IPv6 fejlécre



IPv6 koncepció

- Megnövelt címtartomány
- Egyszerűbb és rugalmasan bővíthető fejrészformátum
  - Alapfejrész: kevesebb funkció
  - Bővíthetőség opcionális funkciókkal
  - Gyorsabb feldolgozás a csomópontokban
- Erőforrás-allokáció támogatása
- Biztonságos kommunikáció támogatása
- Mobilitás támogatása

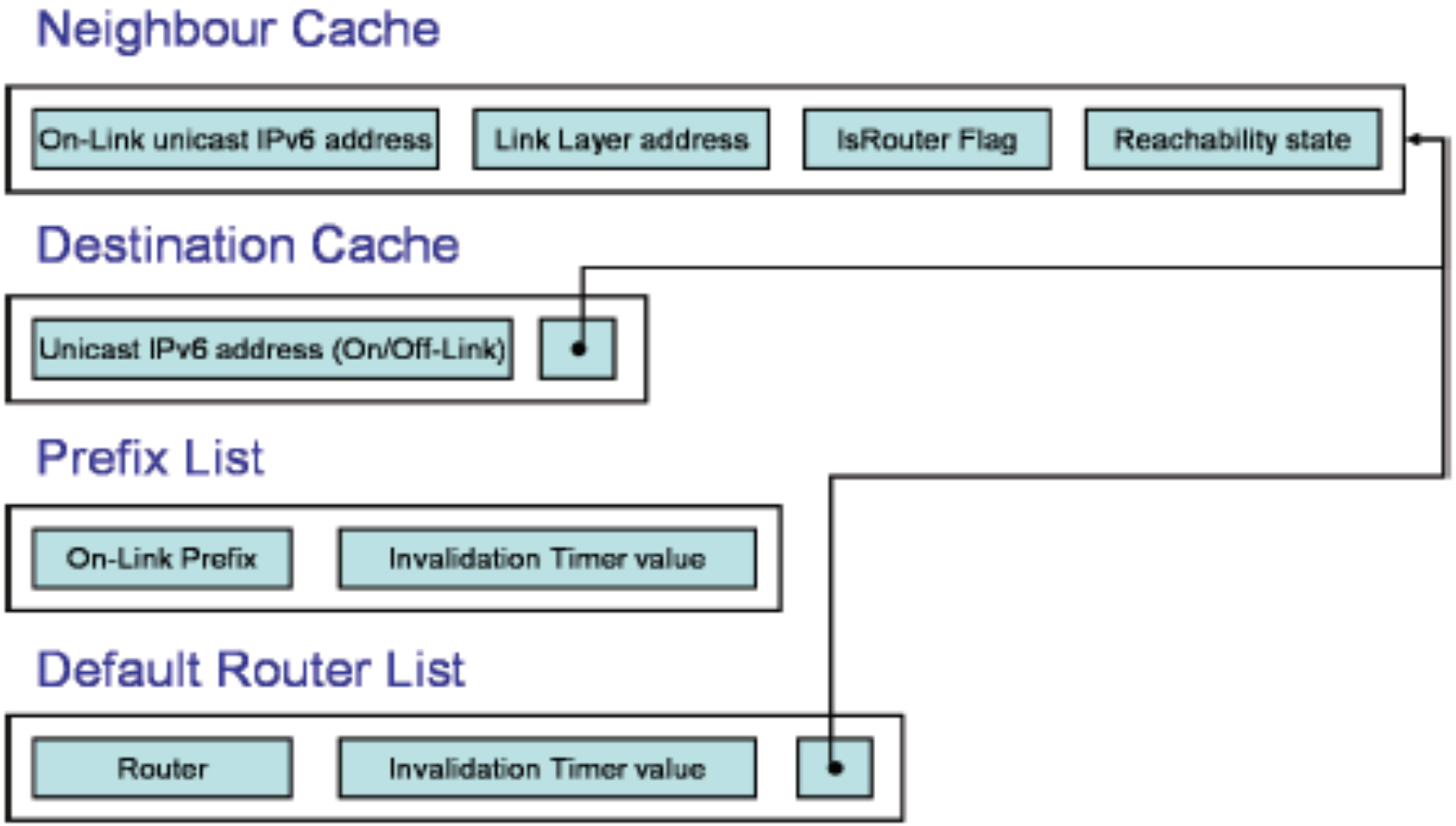
Neighbor Discovery Protocol (NDP)

- IPv4-nél erre szolgált az ARP, ICMP Router Discovery, Redirect
  - Itt egy protokollban minden!
- Szomszédok, routerek felderítése a környezetben
  - ICMPv6 üzenetekkel
- Automatikus címkonfiguráció (Stateless Autoconfiguration)
  - Automatikusan minden beállítást megkap, nem kell DHCP!
  - Stateful: DHCP
- Duplikált cím észlelése

ICMPv6 üzenetek NDP-hez

- Router hirdetés (Router Advertisement)
  - Routerek terjesztik a környezetükben
  - Host automatikusan kiépíti a default router listáját
    - Link prefix, link MTU terjesztése
    - Nem manuális konfiguráció, mint IPv4-ben
- Router kérelmezés (Router Solicitation)
- Redirect
  - Routerek megmondják a jobb útvonalakat a hostoknak
- Szomszéd kérelmezés (Neighbor Solicitation)
  - Szomszéd adatkapcsolati címének lekérése (ARP helyett)
  - Szomszédok elérhetősége/elérhetetlensége
  - Cím duplikáció felderítés
- Szomszéd hirdetés (Neighbor Advertisement)
  - Pl. új adatkapcsolati cím hirdetése

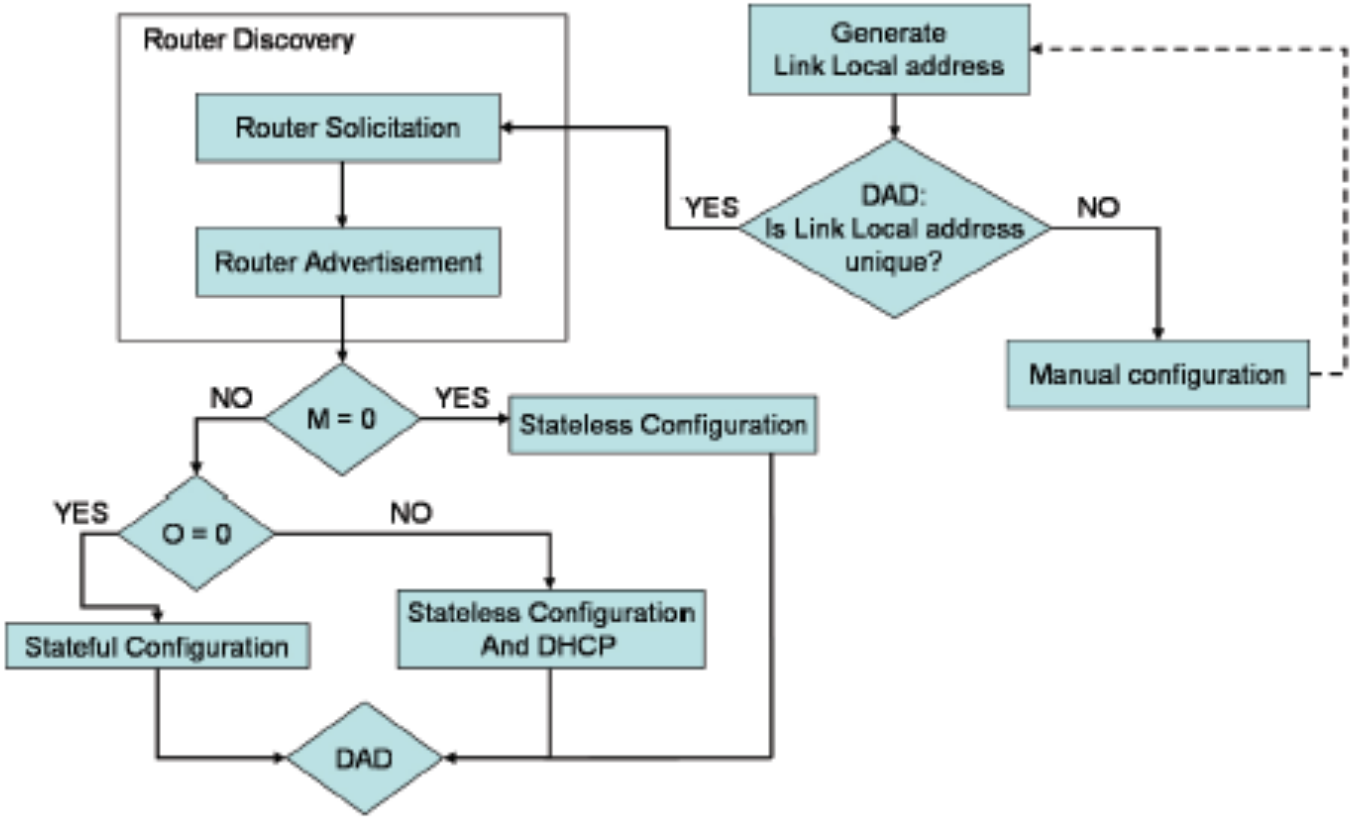
NDP adatbázisok



Állapotmentes automatikus címkonfiguráció

- Először duplikáció ellenőrzése a host link-local címét illetően
  - multicast szomszéd kérelmezéssel
- Ha egyedi a linken: link-local multicast típusú router kérelmezés
- Routerek hirdetéssel felelnek konfigurációs paraméterekkel
- Cím: **routerek által közölt prefix+link local cím**
- Ha alkalmazásnak nem felel meg: lehet állapotfüggő konfiguráció (DHCPv6)

Állapotmentes automatikus címkonfiguráció



A szolgáltatásokat lehetővé tevő „eszközök”

- „Okos” routerek
  - DHCP-funkcionalitás
    - IP-címkiosztás koordinálása
    - Hálózati beállítások nyilvántartása
- Anycast címzés
  - Szolgáltatás elérésére
    - Pl.: „Valamely router mondja meg, hogy…”
- Multicast címzés
  - Azonos szolgáltatásokat nyújtó egységek egymás közötti kommunikációja
    - Pl.: „Én már nem vagyok többé router”

IPv6 Routing

Routing protokollok:

- RIPng
- OSPFv3
- IS-IS Extensions for IPv6
- EIGRP for IPv6

- MP-BGP

Áttérés IPv6-ra

Rengeteg megoldás, mégse megy gyorsan?

2014.Április 3.

Pillanatnyi helyzet

- 2008-ban ünnepelték az IPv6 10 éves születését
- Az IPv6-os forgalom aránya elérte az Interneten 1 % -ot (2012 november), egyes szolgáltatóknál 10-20% felett is már
- Hálózatok több mint 16% támogatja IPv6-ot (2013 vége)

- Okok:
  - Kevés IPv6-os szolgáltató (ISP)
  - Kevés IPv6 kompatibilis alkalmazás és szolgáltatás
  - Még mindig nincs igazi kényszer az áttérésre!
    - Már van☺

- Nagy lökések az áttéréshez:
  - Pekingi olimpia 2008
  - 4G mobil cellás rendszerek: hang VoIPv6

Már létező IPv6 alkalmazások

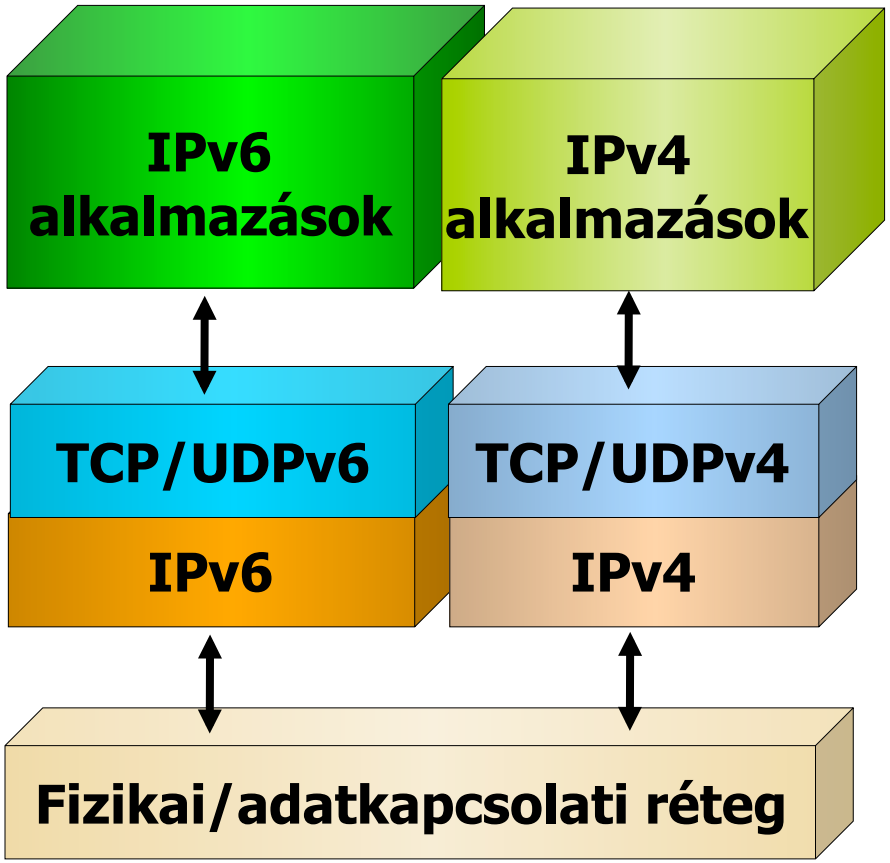
- Hálózatban:
  - Routing protokollok
  - DNS (A6, AAAA és DNAME rekordokkal)
  - DHCP6
  - …
- Operációs rendszerekben régóta (az alábbiakban és ennél újabbaknál):
  - Windows XP SP1, Windows Server 2003
  - Mac OS X 10.2
  - Debian Linux 2.1.8, Redhat 2.1, Solaris 8
  - FreeBSD 4.0, OpenBSD 2.7, NetBSD 1.5, BSD/OS 4.2
- Tűzfalak:
  - ip6fw (FreeBSD)
  - E-Border (kereskedelmi)
  - …
- Alkalmazások:
  - JAVA IPv6 (JDK1.4 óta)
  - 3GPP
  - …

IPv6 kompatibilis IPv4 címzés

- „Compatible”
  - 96 0 bit után az IPv4 cím
- „Mapped”
  - 80 0 bit után 16 1-es bit, majd az IPv4-cím
  - ezt használjuk
- Címfordítás menet közben

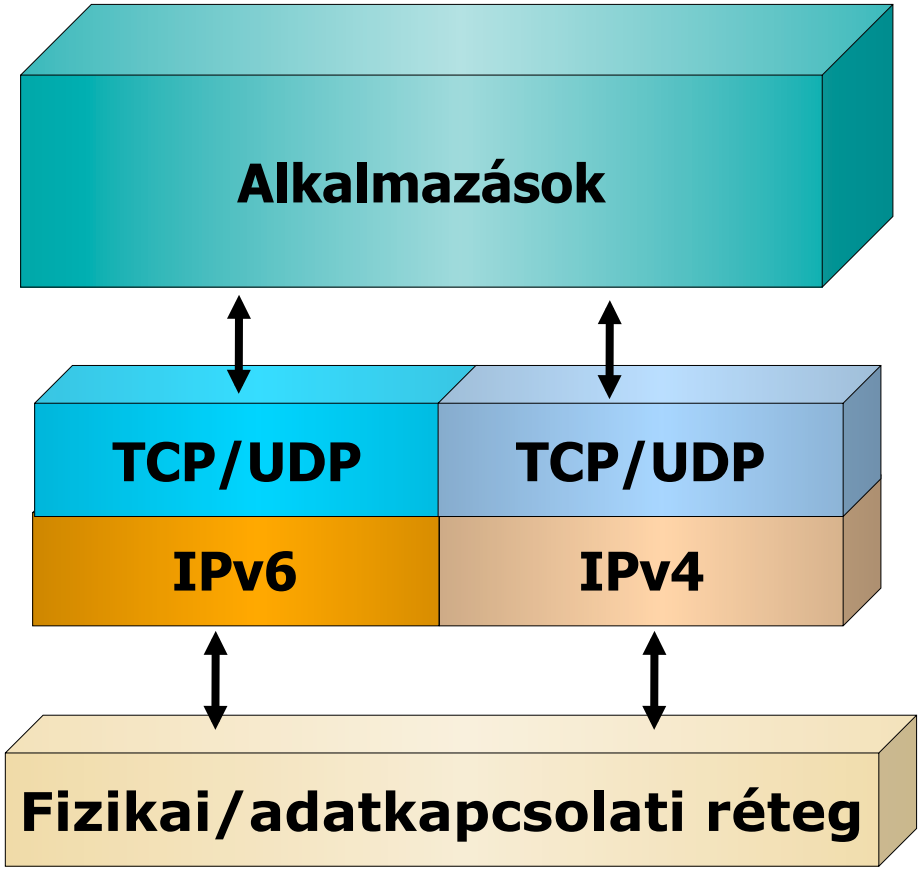
Áttérés Dual Stackkel

- A csomópontok egyszerre IPv4 és IPv6 csomópontok
- IPv6 más IPv6 rendszerekkel való kommunikációhoz, de képesek visszalépni IPv4 módba régi rendszerekhez
- Protokollválasztás: DNS szerver IPv6 vagy IPv4 címmel válaszol
- A legtöbb IPv6 implementáció ezt használja



Áttérés Dual Layerrel

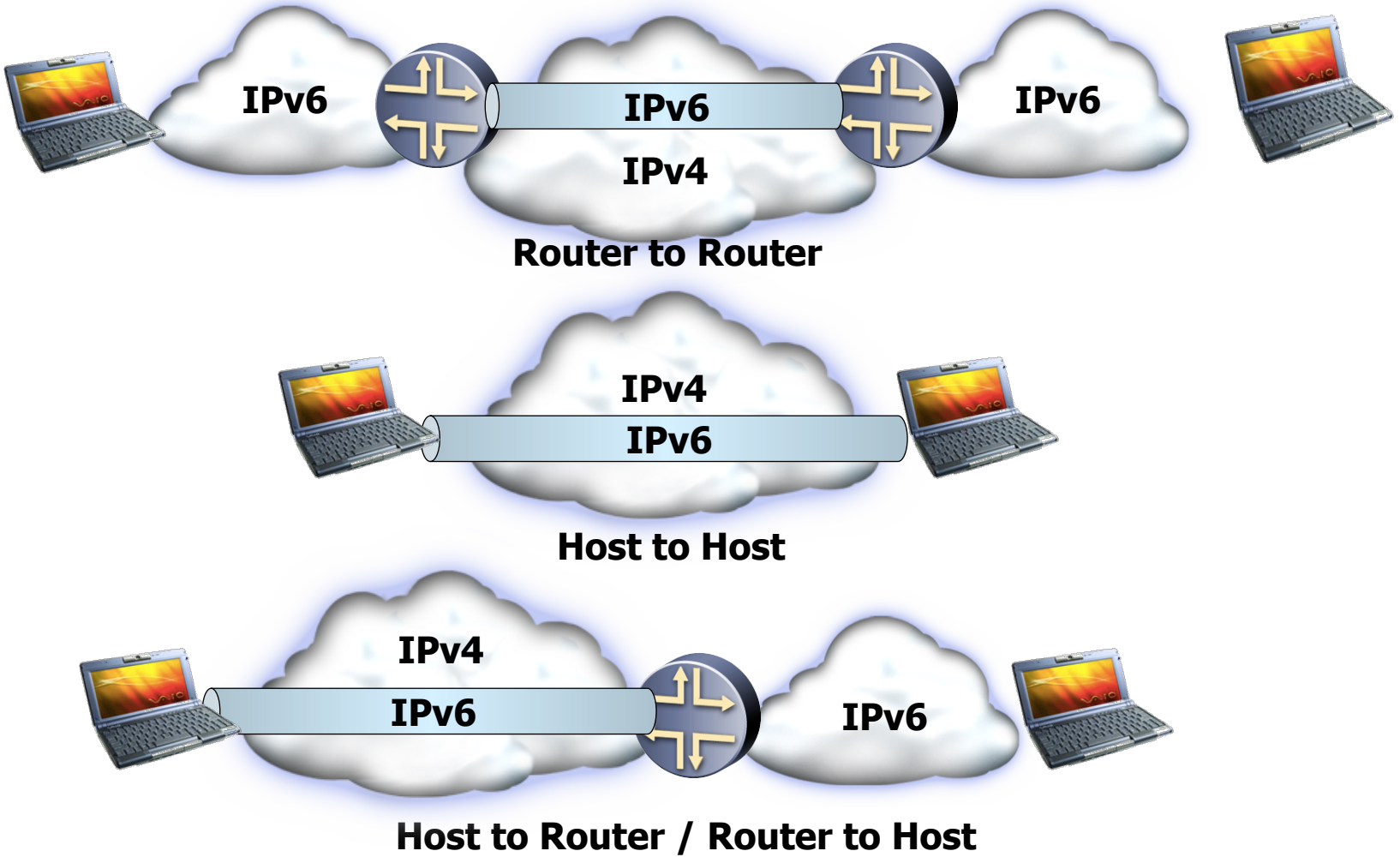
- A csomópontok egyszerre IPv4 és IPv6 csomópontok.
- De a szállítási réteg megmarad.
- Így több alkalmazás működne, ha nem használ IP-címeket.



Áttérés alagutazással (Tunneling)

- Beágyazása az egyik protokoll változatú csomagnak a másikba
  - Mintha adatkapcsolati rétegbeli protokoll lenne az egyik
- IPv4 hálózaton IPv6-os csomagokat szállítunk
  - Végpontok már IPv6-képesek, de a hálózat még nem
  - 6to4: IPv4-es végpont címe az IPv6-os cím hálózati prefixében
    - elterjedt
  - 6in4: statikus alagút-végpont konfiguráció
- IPv6 hálózaton IPv4-es csomagokat szállítunk
  - Hálózat már IPv6-os, de a végpontok még nem

Alagutazás lehetőségei (Tunneling)



Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

IPv6 TRANSITION

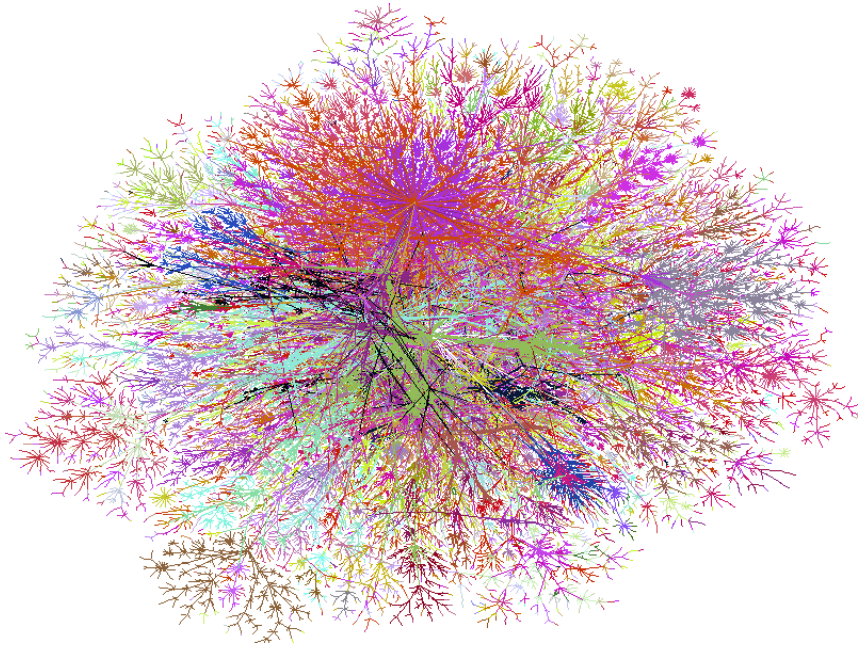
Számítógép-hálózatok (BMEVIHIA215)

Dr. Lencse Gábor
tudományos főmunkatárs
BME Hálózati Rendszerek és Szolgáltatások Tanszék
lencse@hit.bme.hu

2014. április 9.,
Budapest

Tartalom

- Az IPv4 → IPv6 átmenet jellemzői és eszközei
- Kitérő: hálózati címfordítás
- DNS64 + NAT64
- További megoldások



http://cheswick.com/ches/map/gallery/wired.gif

Az IPv4 → IPv6 átmenet jellemzői és eszközei

Az áttérés időzítése

- Áttérés „óraütésre”
  - Az Internet történelmében egyszer sikerült (RFC 801)
    - ARPANET, 1983. 01. 01. áttérés NCP-ről (Network Control Program) TCP/IP-re
  - Ma lehetetlen feladat
    - Több milliárd csomópont (lehetetlen egyszerre „átkapcsolni”)
    - Hibák biztos lennének (jelenleg rengeteg hardver és szoftver eleve alkalmatlan)
    - Hatalmas káosz lenne (pénzügyi, gazdasági,..)
- Hosszú idejű átmenet
  - A két protokoll tartós egymás mellett élésével
    - Bizonyos hardver és szoftver szállítók nem is fogják megoldani az IPv6 kompatibilitást
    - A régi eszközökhöz a felhasználók ragaszkodnak
  - Meg kell oldani az együttműködésüket

IPv4 – IPv6 együttműködése – 1

- Sok alkalmazásunk kliens-szerver konfigurációban működik
  - Mely protokollokra képes a kliens és a szerver?
  - Mely protokollokra képes a hálózat a kettő közötti úton?
- Az egyszerű eset
  - Ha a kliens és a szerver közül bármelyik is képes mindkét protokoll használatára (*dual stack*), akkor a „közös nyelv” használatával a kommunikáció megoldott – feltéve, hogy a hálózat is támogatja. ☺
  - De már nincs elég IPv4 cím! ☹
    - Kényszermegoldás: *Dual-Stack Lite* – bővebben nem foglalkozunk vele.
- IPv6 képes kliens IPv4-only környezetben és IPv6 szerver
  - A kliens képes IPv6-ra de az ISP csak IPv4-címet ad a kliensnek
  - Egy jó megoldás: 6to4 használata
    - Megismerjük

IPv4 – IPv6 együttműködése – 2

- IPv6 kliens és IPv4 szerver
  - Csak IPv6-ra képes kliens (már csak IPv6 cím jutott neki)
  - Csak IPv4-re képes szerver (mert régi, az IPv6-ot nem támogatja)
  - Egy jó megoldás: DNS64 szolgáltatás + NAT64 átjáró használata
    - Megismerjük
- IPv4 kliens és IPv6 szerver
  - Csak IPv4-re képes kliens (régi hardver és/vagy szoftver)
  - Csak IPv6-ra képes szerver (ilyenek is vannak, és számuk nőni fog)
  - Egy megoldás lehet (talán): NAT46 + DNS46
    - Még nem kiforrott, bővebben nem foglalkozunk vele

IPv4 – IPv6 együttműködése – 3

- IPv6 kliens és IPv6 szerver DE útközben csak IPv4 van
  - Tipikus eset, az új IPv6 „szigeteket” össze kell kötni
  - Az IPv6 datagramok szállítása IPv4 fölötti „alagútban” (6in4 tunnel)
    - Röviden megnézzük az elvi megoldást
- IPv4 kliens és IPv4 szerver DE útközben csak IPv6 van
  - Ma még nálunk nem igazán jellemző, de majd lehet
  - Az IPv4 datagramok szállítása IPv6 fölötti „alagútban” (4in6 tunnel)
    - Bővebben nem foglalkozunk vele

KITÉRŐ: HÁLÓZATI CÍMFORDÍTÁS

Network Address Translation – 1

- A TCP/IP protokollcsaládot eredetileg végponttól végpontig való kommunikációra tervezték
- Az alkalmazások arra számítanak, hogy a címek és portszámok a hálózati átvitel során változatlanok
- Az IPv4-címek szűkössége miatt elterjedt megoldás:
  - egy szervezet hálózatában privát IP-címeket használnak
  - a külső kommunikációhoz *címfordítás*t használnak
- A külső kommunikáció feladata lehet:
  1. A privát IP című gépeknek el kell érniük az Internetet
  2. Az Internet felől el kell érni valamely privát IP című gépet
- A feladatok megoldásához rendelkezésünkre áll egy router, amely rendelkezik publikus IP-címmel

Network Address Translation – 2

- Alapötlet az Internet eléréséhez:
  - A célcím alapján küldjük el a kérést a publikus Internet felé
  - A kimenő router cserélje ki a forrás IP-címét a saját publikus IP-címére
  - A csomag megérkezik a címzetthez, és a válasz visszaér a routerhez.
  - A router továbbítsa a választ a forrásnak – DE HOGYAN?

Network Address Translation – 3

- Ahhoz, hogy a router a választ az eredeti feladónak vissza tudja küldeni, nyilván kell tartania, és fel kell ismernie, hogy ki volt az eredeti küldő.
  - Ennek érdekében az IP-címen túl mást is felhasznál. TCP és UDP esetén ezek a forrás portszámok
  - Nem elegendő ezeket megjegyezni, hiszen a forrás portszámok csak *gépenként egyediek*, a forrás IP-címet viszont a sajátjára cseréli
  - A forrás portszámokat kicseréli a *routeren egyedi portszámokra*
  - Az IP-címek és a célportszám mellett ezekkel már egyértelműen azonosítani tudja a kapcsolatokat
  - Kapcsolatonként nyilvántartja, hogy mit mire cserélt ki
  - A bejövő csomagoknál a cél IP-címen kívül a cél portszámot is vissza kell cserélnie ^(változott az irány)^

Network Address Translation – 4

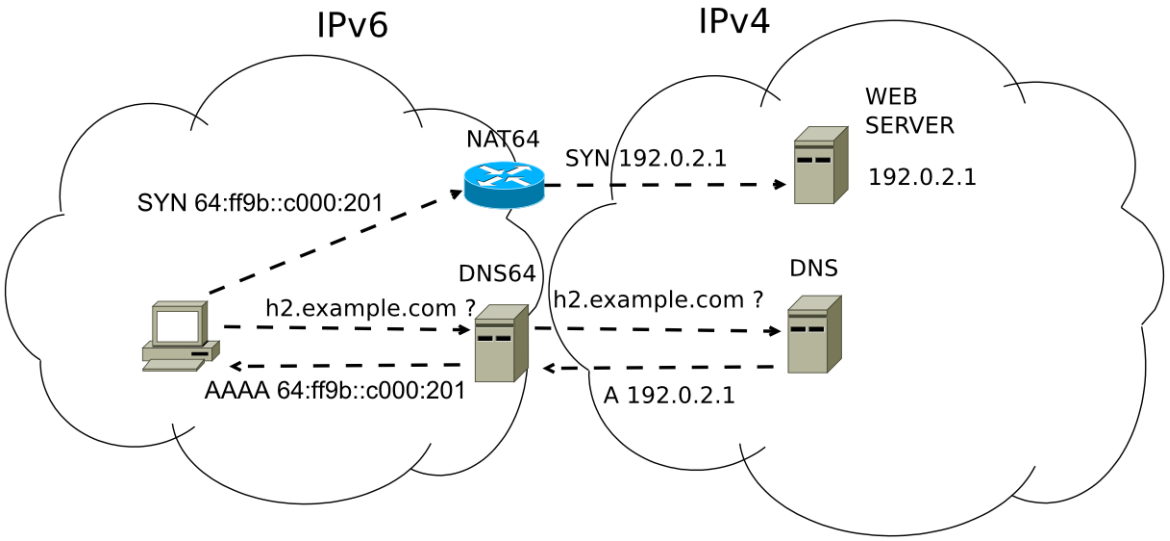
- Az ismertetett megoldást hívjuk *Source NAT*-nak (SNAT) akkor, ha a router publikus IP-címe fix, és Masquerade-nek, ha DHCP-vel kapta az interfésze a címet
- Megjegyzés: terminológia nem egységes
  - Eredetileg a NAT csak az IP-címek cseréjét jelentette, ezt hívják ma *basic NAT*-nak, vagy *one-to-one NAT*-nak.
  - A fenti megoldás precíz neve a *NAPT* (Network Address and Port Translation). Nevezik *many-to-one NAT*-nak is.

Network Address Translation – 5

- A másik irányú feladat az, hogy pusztán privát IP-címmel rendelkező gépeket elérhetővé tegyünk az Internet felől.
  - Erre a megoldás a *Destination NAT* (DNAT) vagy más néven port forwarding, ahol a router az adott portjára érkező datagramokat egy meghatározott privát IP című gépnek továbbítja úgy, hogy a célcímet kicseréli a csomagban.
    - Például a 80-as portra érkező datagramokat a 10.1.1.2 IP-című webszerver, a 25-ösre érkezőket pedig a 10.1.1.3 IP-című SMTP szerver felé továbbítja
- Mi helyzet az ICMP üzenetekkel?
  - ICMP esetén nincs portszám, de segíthet az, hogy egy hibaüzenetben benne van az azt kiváltó TCP vagy UDP adategység első 64 bitje a portszámokkal.
  - Amennyiben nem hibaüzenetről van szó, akkor is van megoldás, az ICMP üzenet valamilyen azonosító jellegű mezőjét használják fel.

Network Address Translation – 6

- Mindkét irányú megoldásnál gondot okozhat, ha az alkalmazások számítanak a címek és portok változatlanságára – ami részükről jogos elvárás.
  - Például egy FTP kliens aktív módban a vezérlő kapcsolaton keresztül megadja a szervernek, hogy mely portján várja, hogy a szerver felépítse az adatkapcsolatot.
  - Privát IP címmel várhatja – hacsak nem segít valaki: *protocol helper*
- Ez a probléma NAT64-nél is előjön.
- Érdeklődőknek:
  - A NAT-ról bővebben az RFC 3022-ben olvashatunk, az IP, TCP, UDP, ICMP fejrészek mezőinek módosításával a 4.1. rész, az ellenőrző összeg hatékony újraszámításával a 4.2. rész foglalkozik.

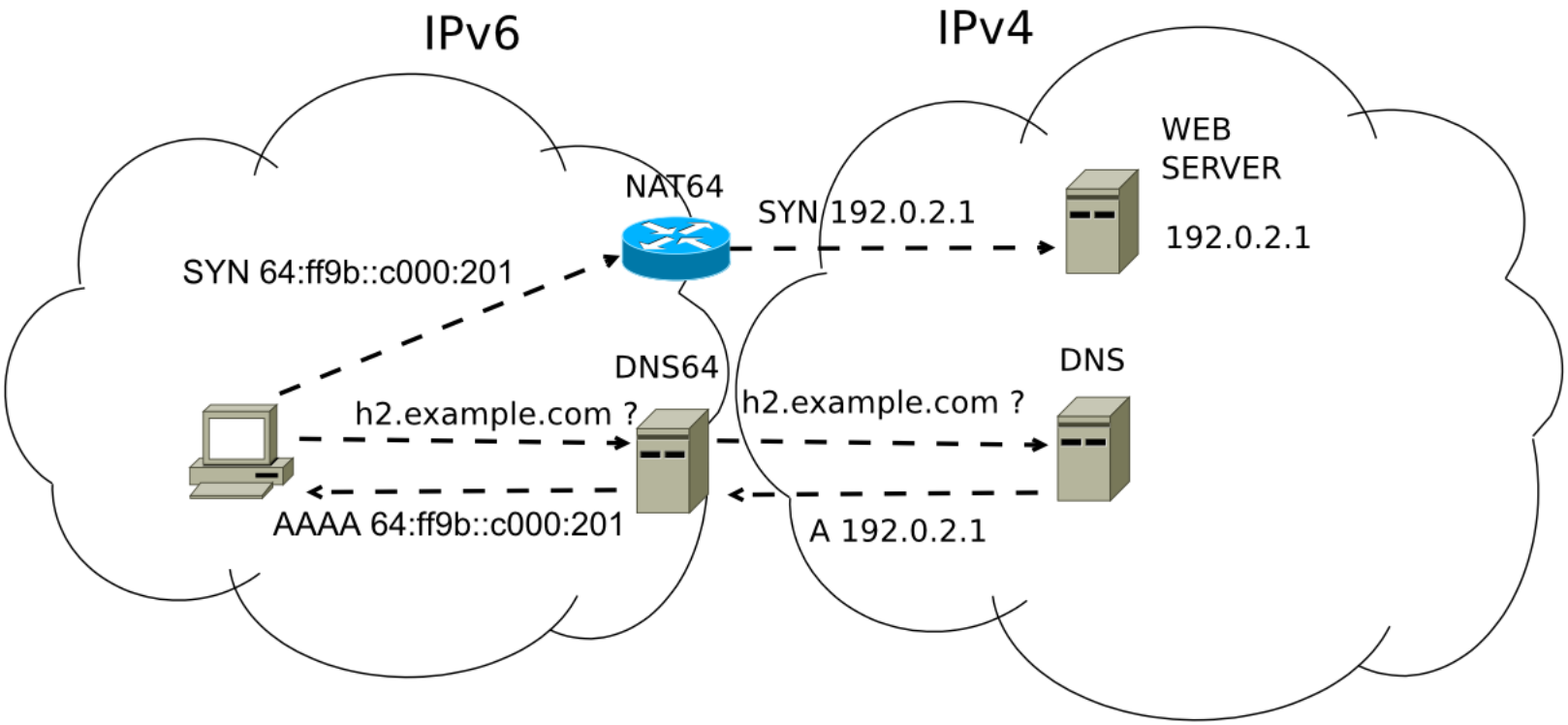


Forrás: http://en.wikipedia.org/wiki/File:NAT64.svg (javítva)

DNS64 + NAT64

IPv6-only kliens és IPv4-only szerver

- Az IPv4 címek kifogyása miatt várhatóan ez lesz az első tipikus megoldandó probléma
  - Az elvi megoldást az alábbi példán mutatjuk be
    forrás: http://en.wikipedia.org/wiki/File:NAT64.svg (javítva)



DNS64 szerver

- Mint egy caching-only name server, *Recursive query*kre válaszol
  - Ha van IPv6 cím, akkor továbbítja a válaszában
  - Ha nincs IPv6 cím, csak IPv4, akkor az IPv4 címből generál egy speciális IPv6 címet, és ezt adja vissza
- A speciális IPv6 cím a példában használatos well-known prefix esetén
  - A 64:ff9b::/96 prefix + az utolsó 32 biten a kapott IPv4 cím

A NAT64 átjáró használata – 1

- A megoldás működéshez szükséges:
  - A kliensben névkiszolgálóként a DNS64 szerver van beállítva
  - Az útválasztási táblázatok szerint a well-known prefix felé az út egy NAT64 átjárón keresztül vezet (Anycast címzés használható)
- Kövessük végig a példát!
  - Az IPv6-only kliens csatlakozni szeretne az IPv4-only szerverhez
  - Lekéri a szerver IPv6 címét a szimbolikus neve alapján
  - Megkapja a szerver IPv4 címét tartalmazó speciális IPv6 címet
  - TCP SYN szegmenst tartalmazó IPv6 csomagot küld a kapott címre
  - Az IPv6 csomag megérkezik a NAT64 átjáróhoz
  - Az átjáró az IPv6 csomag alapján egy IPv4 csomagot készít, benne
    - A célcím az IPv6 cím utolsó 32 bitje
    - A forráscím a NAT64 átjáró IPv4 címe
  - Az átjáró a csomagot elküldi a címzettnek

A NAT64 átjáró használata – 2

- Tovább követjük a példát…
  - Az IPv4-only szerver megkapja a TCP SYN szegmenst tartalmazó IPv4 csomagot és a megszokott módon válaszol (SYN+ACK)
    - A kapott IPv4 csomagban a forráscím a NAT64 átjáró IPv4 címe volt
    - A válasz címzettje is a NAT64 átjáró IPv4 interfésze lesz
  - A választ megkapja a NAT64 átjáró és elkészíti a neki megfelelő IPv6 csomagot, melybe
    - Forráscímként az a speciális IPv6 cím kerül, amit a DNS64 szerver generált
    - Célcímként az IPv6-only kliens IPv6 címe kerül
    - Mindez a NAT által korábban is használt módon, kapcsolattábla alapján történik
  - Az IPv6 csomagot a NAT64 átjáró elküldi az IPv6-only kliensnek
  - Az IPv6-only kliens megkapja a csomagot
  - A kommunikáció a fentiek szerint tovább folytatódik…

A NAT64 átjáró használata – 3

- A fentiekben bemutatott megoldás az RFC 6052 szerinti 64:ff9b::/96 előre lefoglalt, ún. *Well-Known Prefix*et használja.
- A gyakorlatban számos hátránnyal járna, ha világszerte mindenki ezt a prefixet használná (RFC 6052 3.1 és 3.2)
- A NAT64 átjáró megvalósításakor a gyakorlatban az egyes IPv6 hálózatokból szoktak erre a célra lefoglalni egy részt, ezt *hálózat specifikus prefix*nek (network specific prefix) nevezik.
- Az ilyen IPv4 címeket tartalmazó IPv6 címeket úgy hívják, hogy *IPv4-Embedded IPv6 Addresses*

IPv4-Embedded IPv6 címek fajtái

- Az IPv4-címeket beágyazó IPv6-címeket még két további névvel illetik a felhasználásuk céljától függően, bár szerkezetük és előállításuk azonos.
  - Azokat az IPv6 címeket, amiket arra használunk, hogy IPv4 állomásokat képviseljenek IPv6 hálózatokban, úgy nevezik, hogy IPv4-Converted IPv6 Address. (Jelenleg pontosan erről van szó.)
  - Az IPv4-Translatable IPv6 Address nevet pedig akkor használjuk, ha a cím egy IPv6 állomáshoz tartozik, és a fordítás célja, hogy a csak IPv4-re képes eszközök is el tudják érni az IPv6 állomást. (Ezzel az esettel most nem foglalkozunk.)

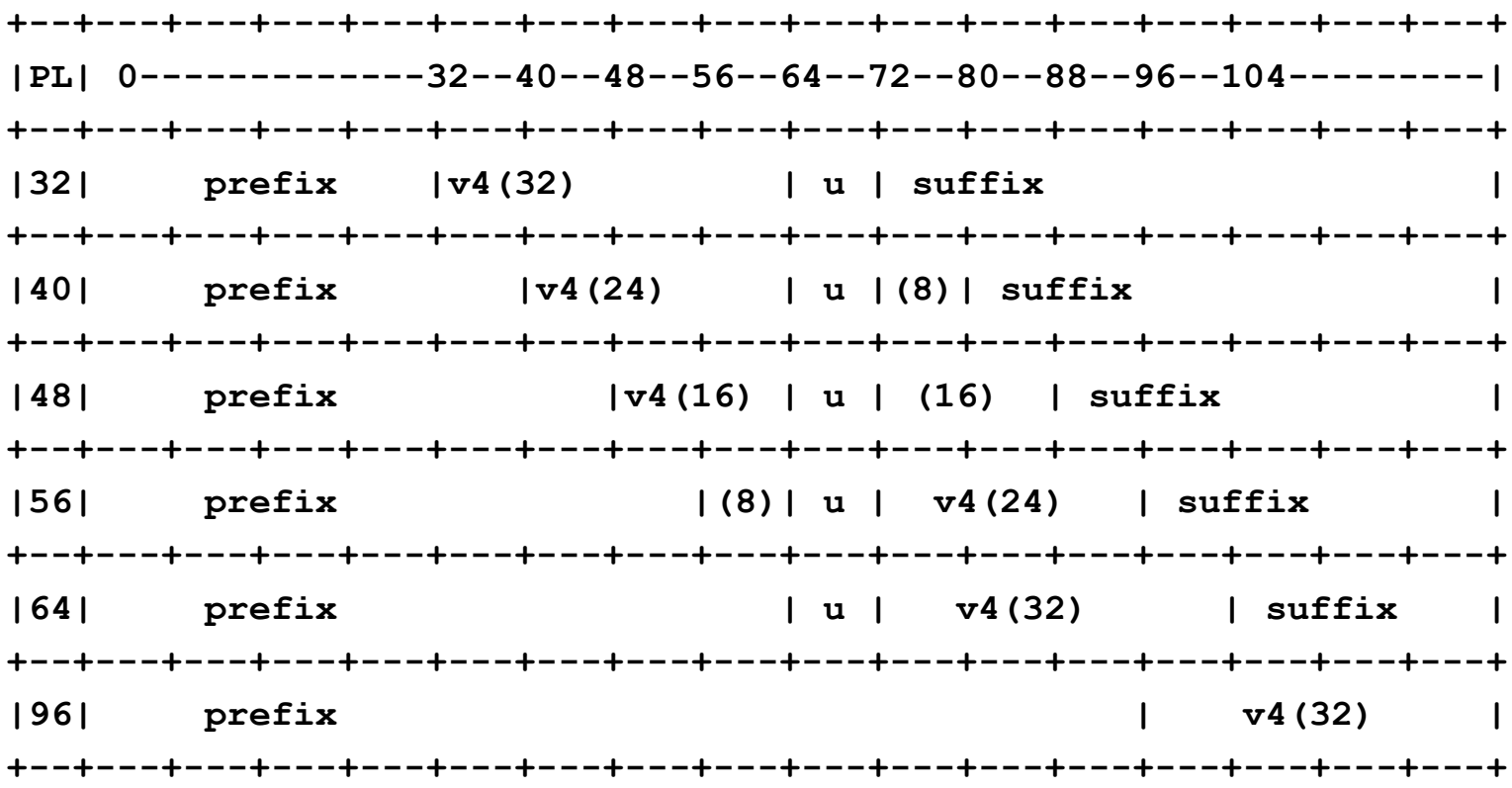
IPv4-Embedded IPv6 címek képzése – 1

- Az RCF 6052 definiálja, hogy a prefix méretétől függően hogyan kell az IPv4-címeket beágyazó IPv6-címeket képezni.
  - A prefix mérete szigorúan csak 32, 40, 48, 56, 64 vagy 96 lehet
  - Az IPv6 címben 64-71 biteknek 0-nak lenniük
  - Az IPv6 cím 32 bitjét általában a prefix után írjuk, de a fenti követelmény kielégítése érdekében a szigorúan 0 értékű bitek helyét „átugorjuk”
  - A cím végét szükség esetén ugyancsak 0 értékű bitekkel töltjük ki

(Ábra a következő oldalon)

IPv4-Embedded IPv6 címek képzése – 2

- A címek lehetséges formátuma
  - PL: prefix length, v4: IPv4 cím bitjei, u és suffix: 0 értékű bitek



TOVÁBBI MEGOLDÁSOK

A 6in4 tunnel elve

- A feladat:
  - IPv6 szigetek csak IPv4 hálózaton keresztül tudnak egymással kommunikálni
- IPv6-over-IPv4 (RFC 4213)
  - Az IPv6 csomagokat IPv4 csomagokba csomagolja be
  - Az IPv4-ben az 41-es protokoll azonosítót használja az IPv6 csomagok azonosítására
- A be- és kicsomagolást az IPv6 szigetek határán levő átjárók végzik

A 6to4 megoldás – 1

- A megoldandó feladat:
  - IPv6 képes eszköz IPv4-only környezetben van
  - IPv6 protokollal egy másik IPv6-os eszközt szeretne elérni
    - akár az is lehet IPv4-only környezetben
- A 6to4 megoldás (RFC 3056) egy „automatikus” tunnel, ami az IPv6 csomagokat IPv4 csomagokba csomagolja be
  - A 6in4-hez hasonlóan a 41-es protokoll azonosítót használja
- Megvalósítás szempontjából kétféle konfiguráció lehetséges
  - Egyetlen host, amin IPv6 kliens fut, és a becsomagolást is a host végzi – a 6to4 pseudo-interface a hoston van (*6to4 host*)
  - Egy IPv6 hálózat, aminek a routere végzi a becsomagolást – a *6to4 (border) router* rendelkezik 6to4 pseudo-interface-szel.
- A kicsomagolás a *6to4 relay* feladata
  - Ilyen több is lehet, legközelebbi a 192.88.99.1 anycast címen

A 6to4 megoldás – 2

- A megoldás működésének lépései, ha a cél natív IPv6
  - A kliens a szerver felé IPv6 csomagot küld
  - A becsomagolás elvégzője (host vagy router) az IPv6 csomagot IPv4 csomagba ágyazza, és az IPv4 segítségével elküldi egy általa választott *6to4 relay*nek (saját döntése, hogy kinek címzi)
  - A 6to4 relay megkapja az IPv4-be ágyazott IPv6 csomagot, kicsomagolja és továbbítja a natív IPv6 hálózatba
  - A natív IPv6 hálózatban a csomag megérkezik a címzetthez
  - A címzett válaszol
  - A válasz visszaér vagy az előbbi vagy egy másik 6to4 relayhez
  - Visszafele a 6to4 relay az IPv6 csomagokat IPv4-be csomagolva küldi a korábban a kliens IPv6 csomagját IPv4-be csomagoló eszköznek (host vagy router). – Ennek publikus IPv4 címe van!
  - A host vagy router kicsomagolja az IPv4 csomagból az IPv6 csomagot és eljuttatja a kliensnek.

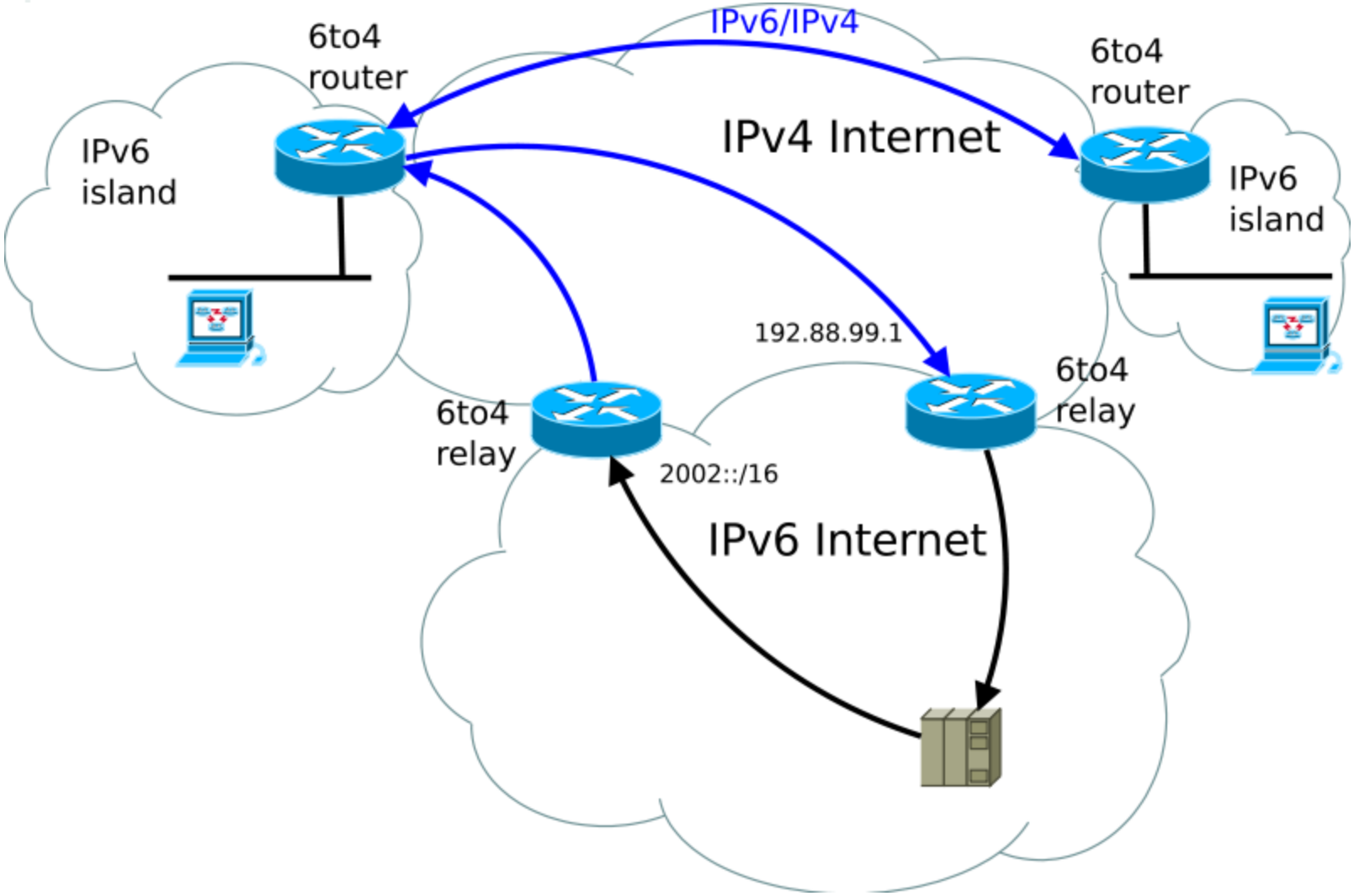
A 6to4 megoldás – 3

- Működéséhez szükséges feltétel: a kliensnek legyen publikus IPv4-címe
  - Ha nincs neki, akkor *Teredo*t kell helyette használni. (RFC 4380)
  - A 6to4 továbbfejlesztett változata a *6rd* (RFC 5969)
- Példánkban egy IPv4 környezetben működő IPv6 host natív IPv6 Interneten levő szervert ért el.
- A 6to4 segítségével IPv6 szigeteket is összeköthetünk IPv4 fölött.
  - Ebben az esetben a becsomagolást végző eszköz a cél IPv6-címéből (6to4 cím) tudja meg, hogy nem egy IPv6 relaynek, hanem a megfelelő IPv4 című pseudo-interfésszel rendelkező eszköznek kell IPv4 szinten címeznie és küldenie a csomagot.

A 6to4 megoldás címzése

- A 6to4 címzéshez 2002::/16 prefixet foglalták le
- A címek képzése
  - Hálózatcím:
    - 2002::/16 prefix + publikus IPv4 cím 32 bitje + 16 bit subnet ID
    - Host esetén a subnet ID egy generált véletlenszám
    - Ha a 6to4 mechanizmust router használja, akkor akár több IPv6 hálózat is lehet mögötte, ekkor hasznos a subnet ID.
  - Gépcím:
    - A szabványos módosított EUI-64 azonosító
- Ilyen módon a 6to4 minden publikus IPv4 címhez egy 2002::/16 kezdetű, /48 méretű IPv6 címtartományt rendel
  - Mindegyik „mögött” elérhető lehet egy ilyen méretű IPv6 hálózat

A 6to4 megoldás lehetőségei



Forrás: http://en.wikipedia.org/wiki/File:6to4.svg

Összefoglalás

- Az IPv4 és IPv6 még sokáig együtt fog élni.
  - Az együttműködésnek számos fajtája van/lesz.
  - Még nagyobb a megoldások száma.
- Megismertük:
  - DNS64+NAT64: IPv6-only kliens eléri az IPv4-only szervert
  - 6to4: IPv4 környezetben levő IPv6 eszköz eléri a natív IPv6 Internetet vagy egy másik hasonló eszközt
  - 6in4 tunnel (csak elvi szinten): IPv6 szigetek IPv4 fölött képesek egymással kommunikálni

Kérdések?

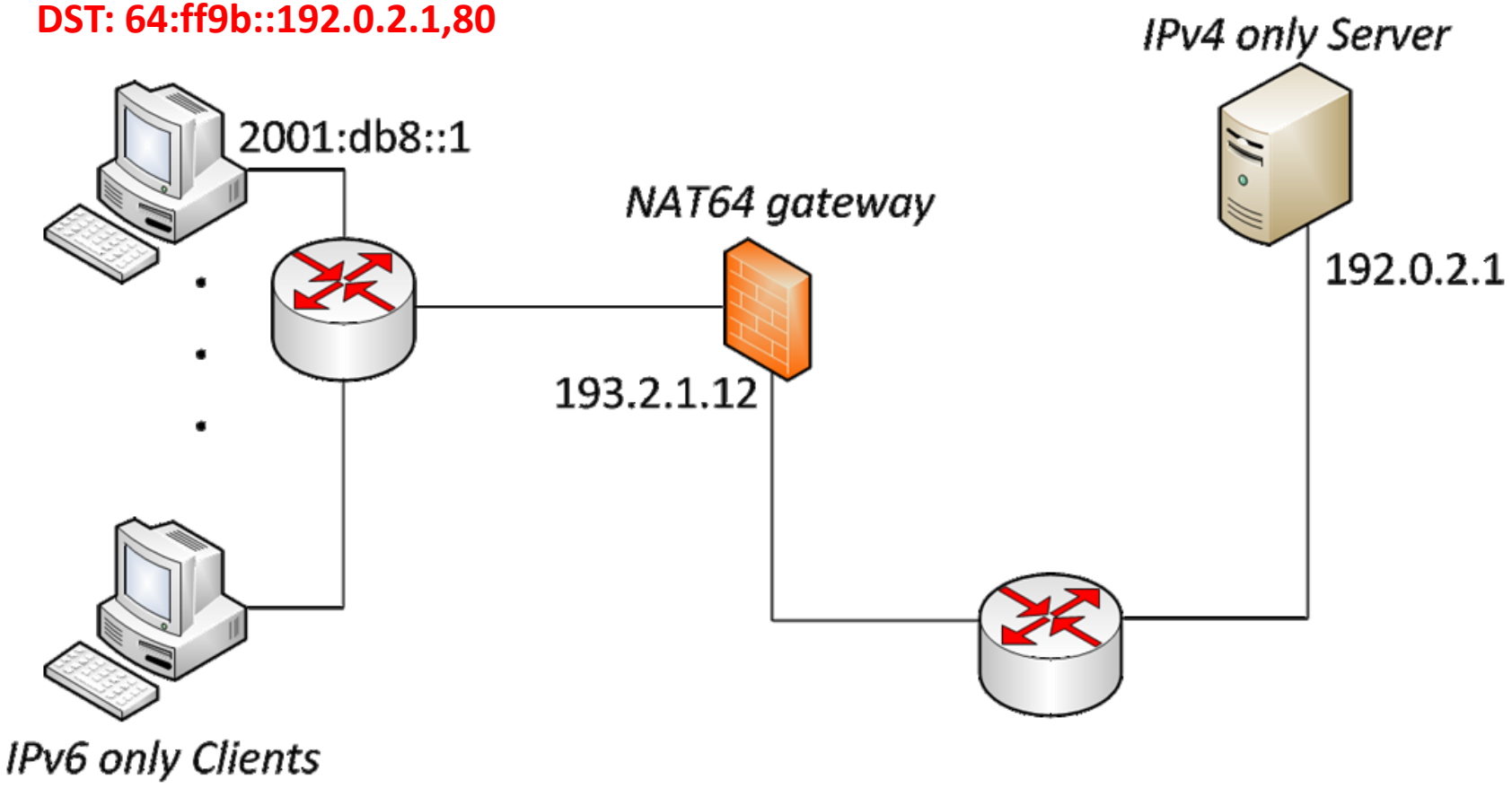
KÖSZÖNÖM A FIGYELMET!

Hálózati Rendszerek és Szolgáltatások Tanszék

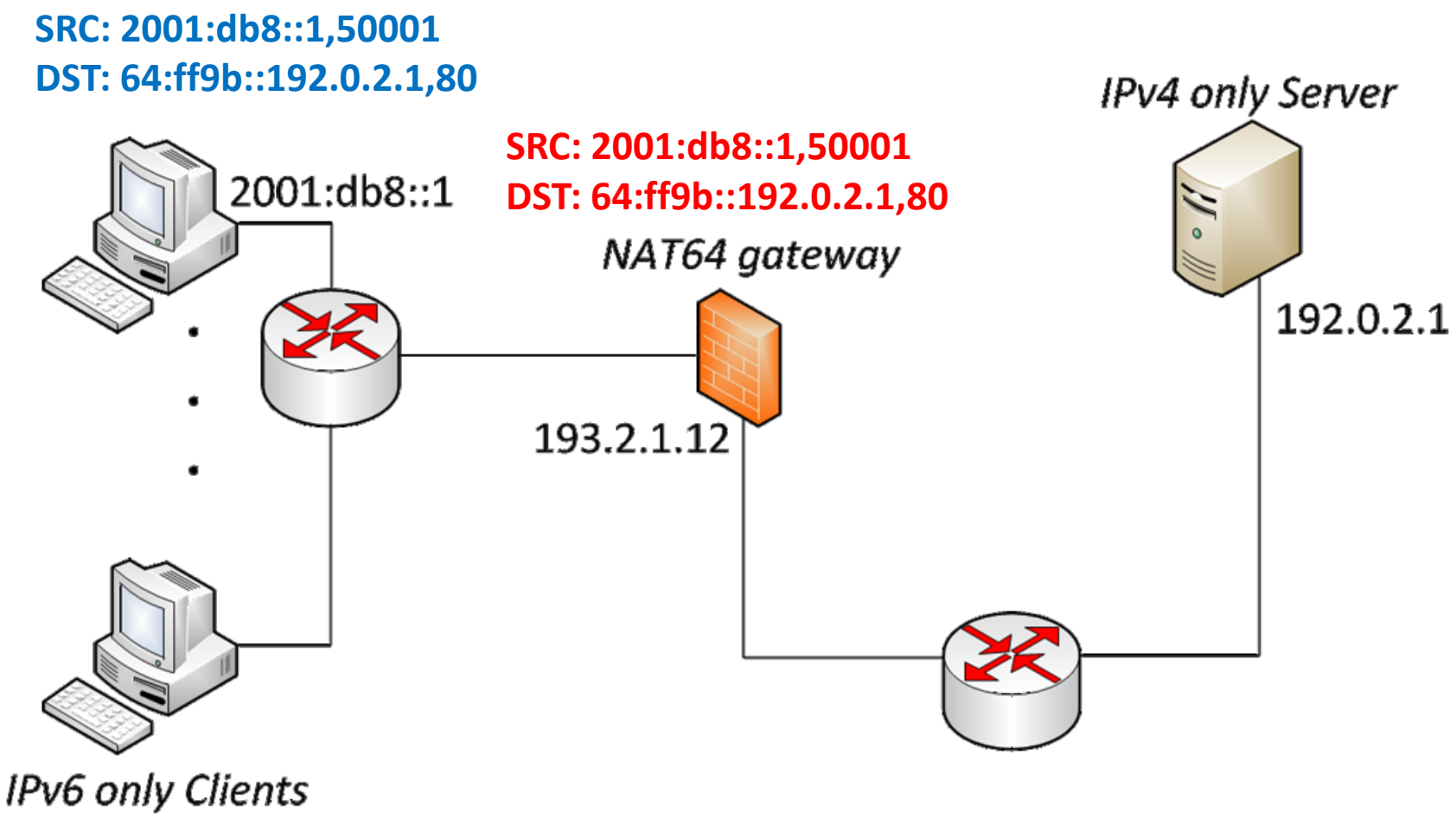
Dr. Lencse Gábor
tudományos főmunkatárs
BME Hálózati Rendszerek és Szolgáltatások Tanszék
lencse@hit.bme.hu

The Operation of NAT64

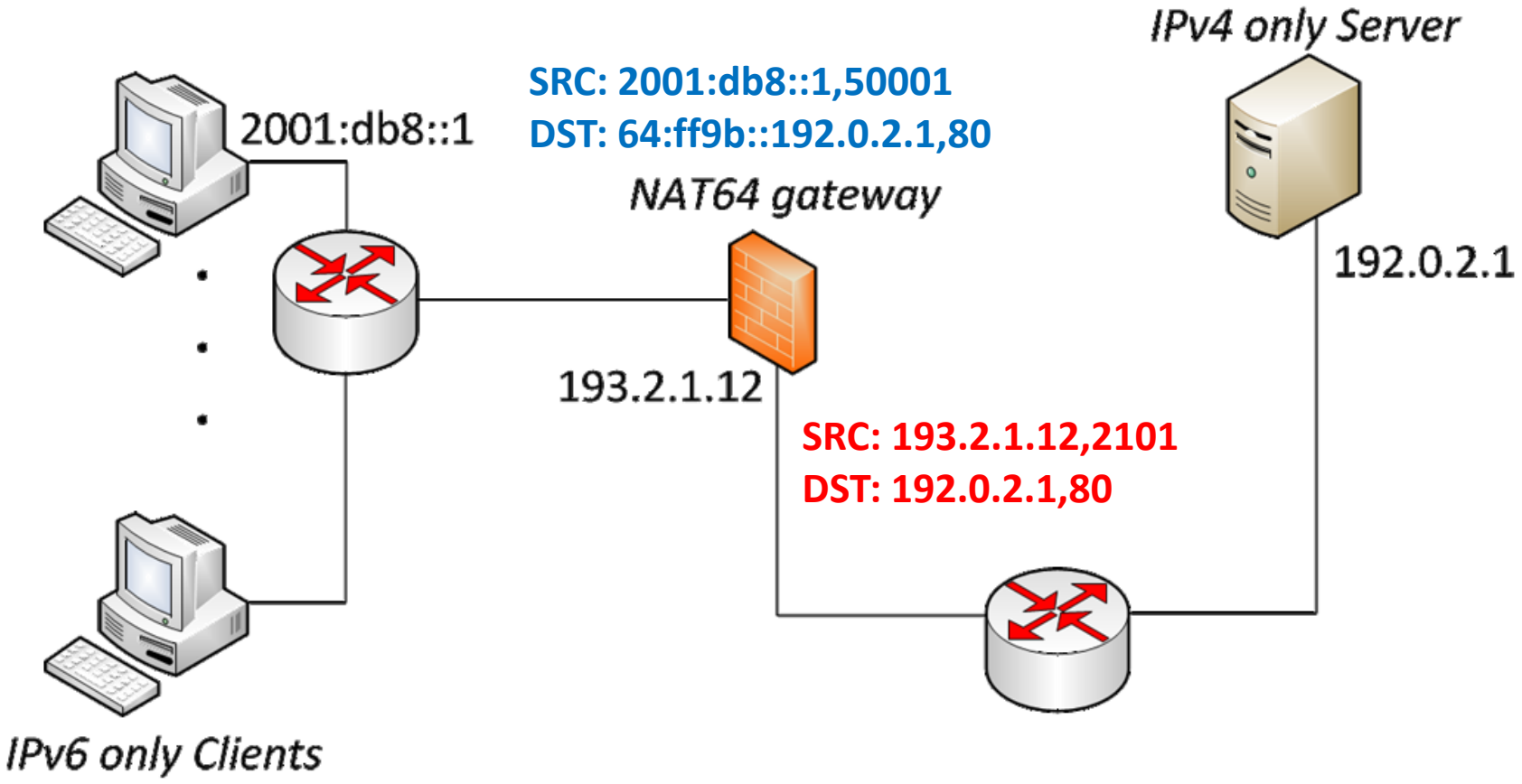
SRC: 2001:db8::1,50001
DST: 64:ff9b::192.0.2.1,80



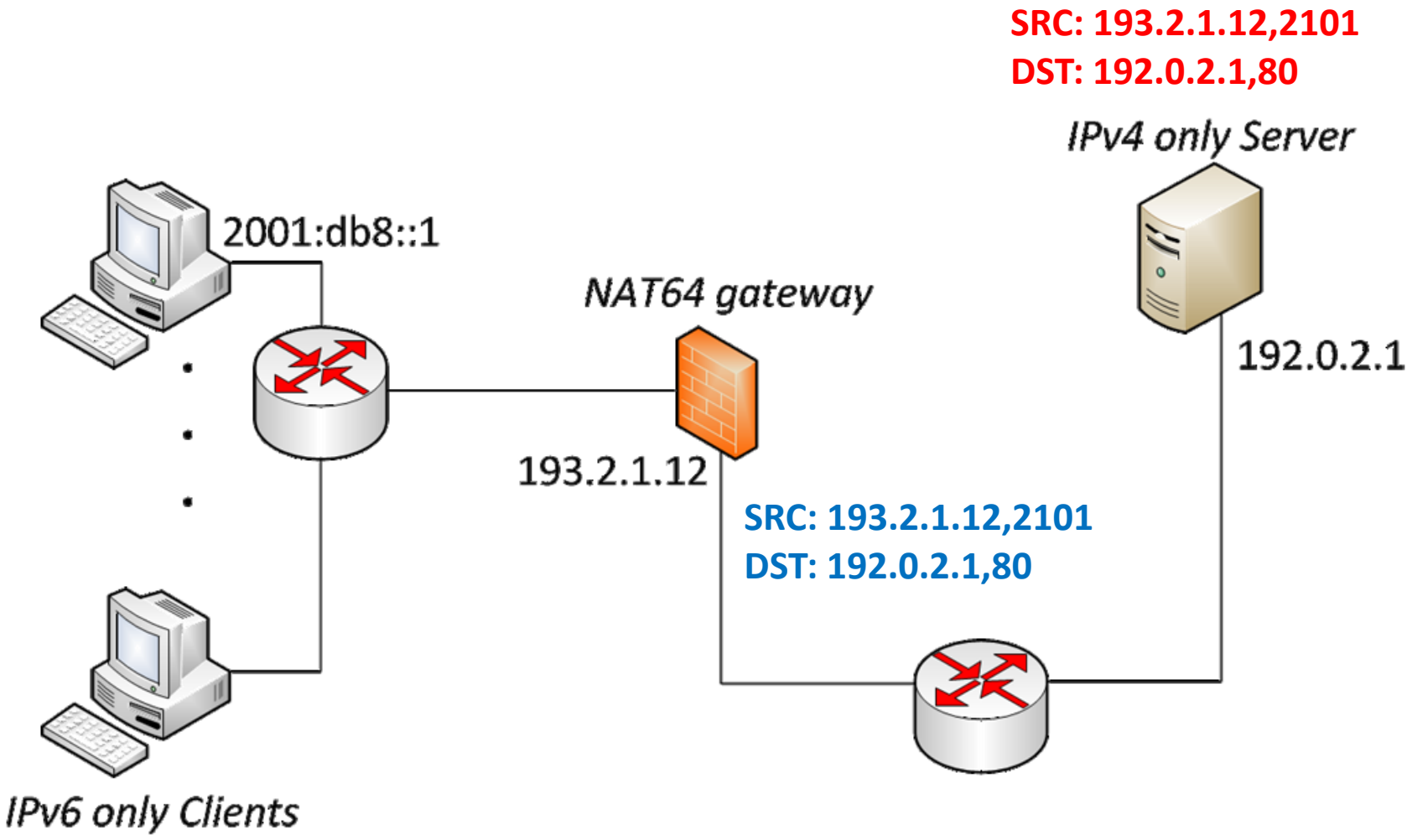
The Operation of NAT64



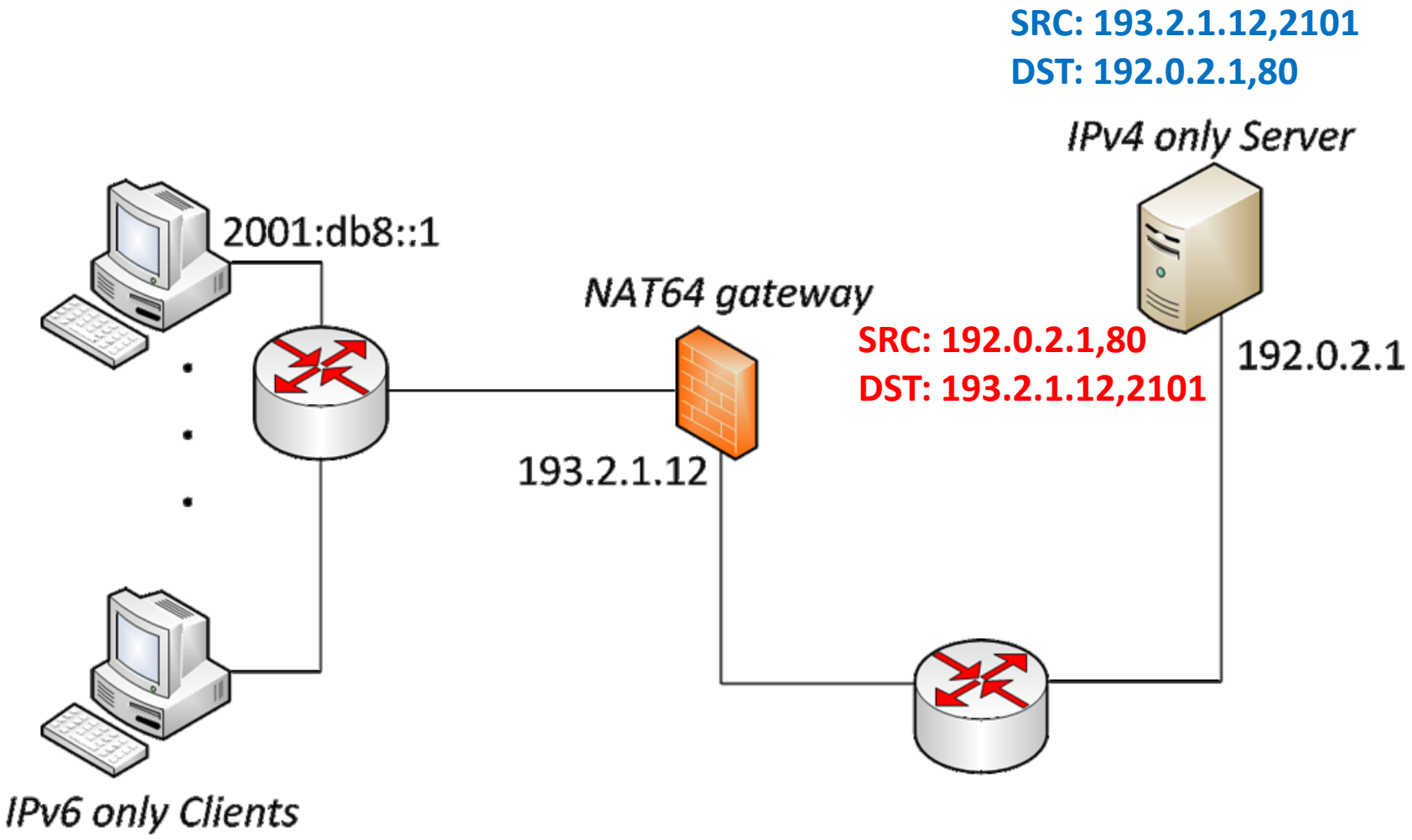
The Operation of NAT64



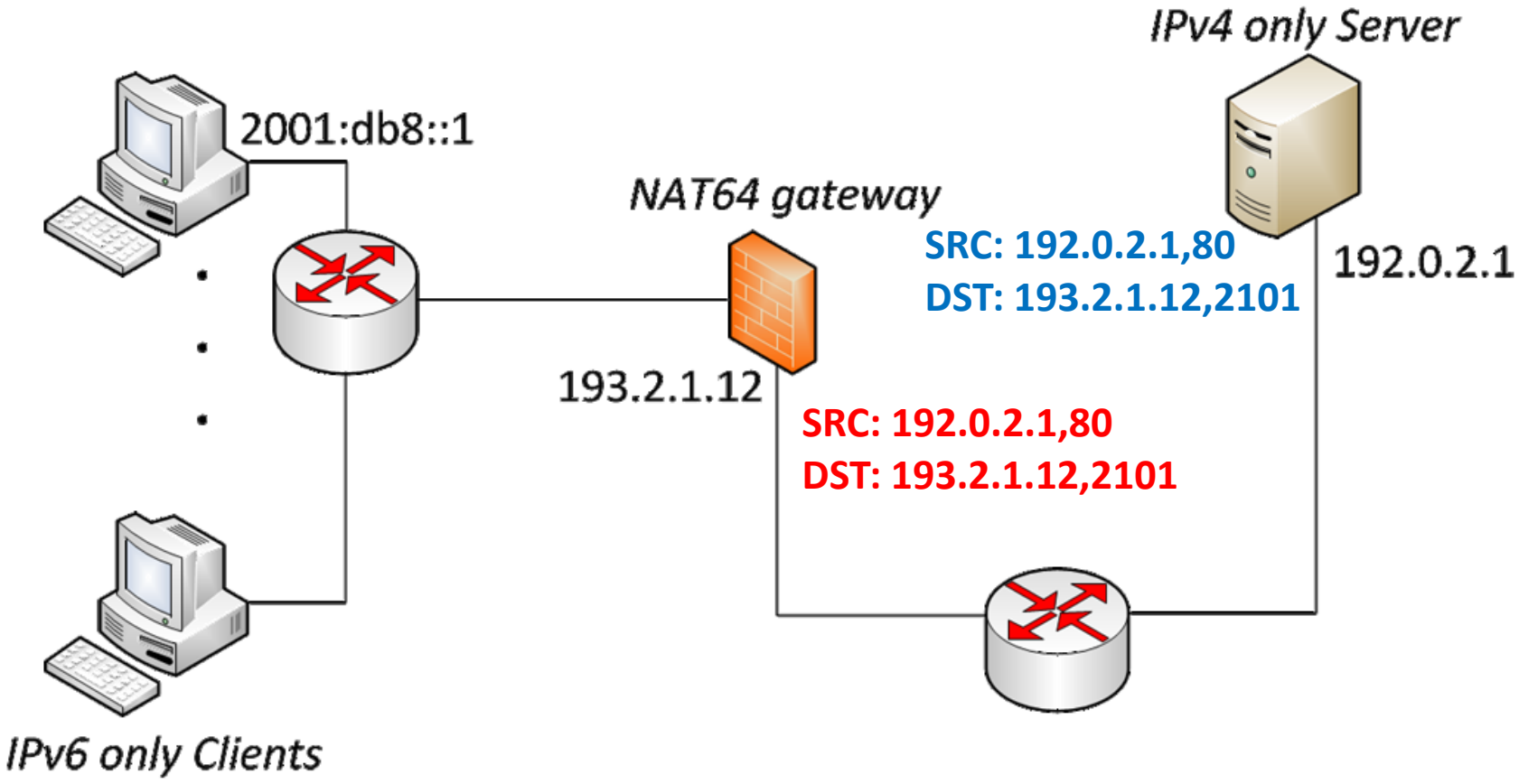
The Operation of NAT64



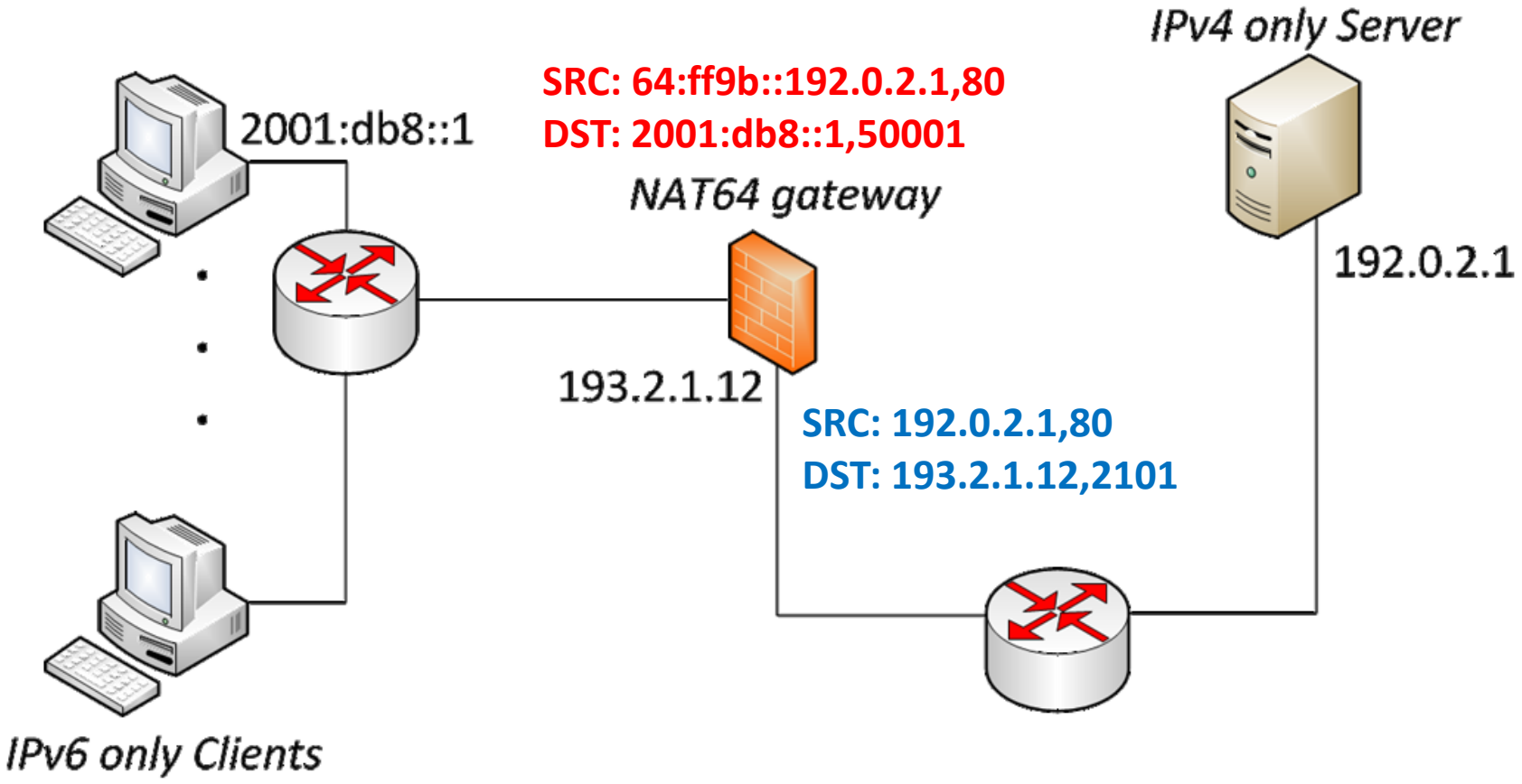
The Operation of NAT64



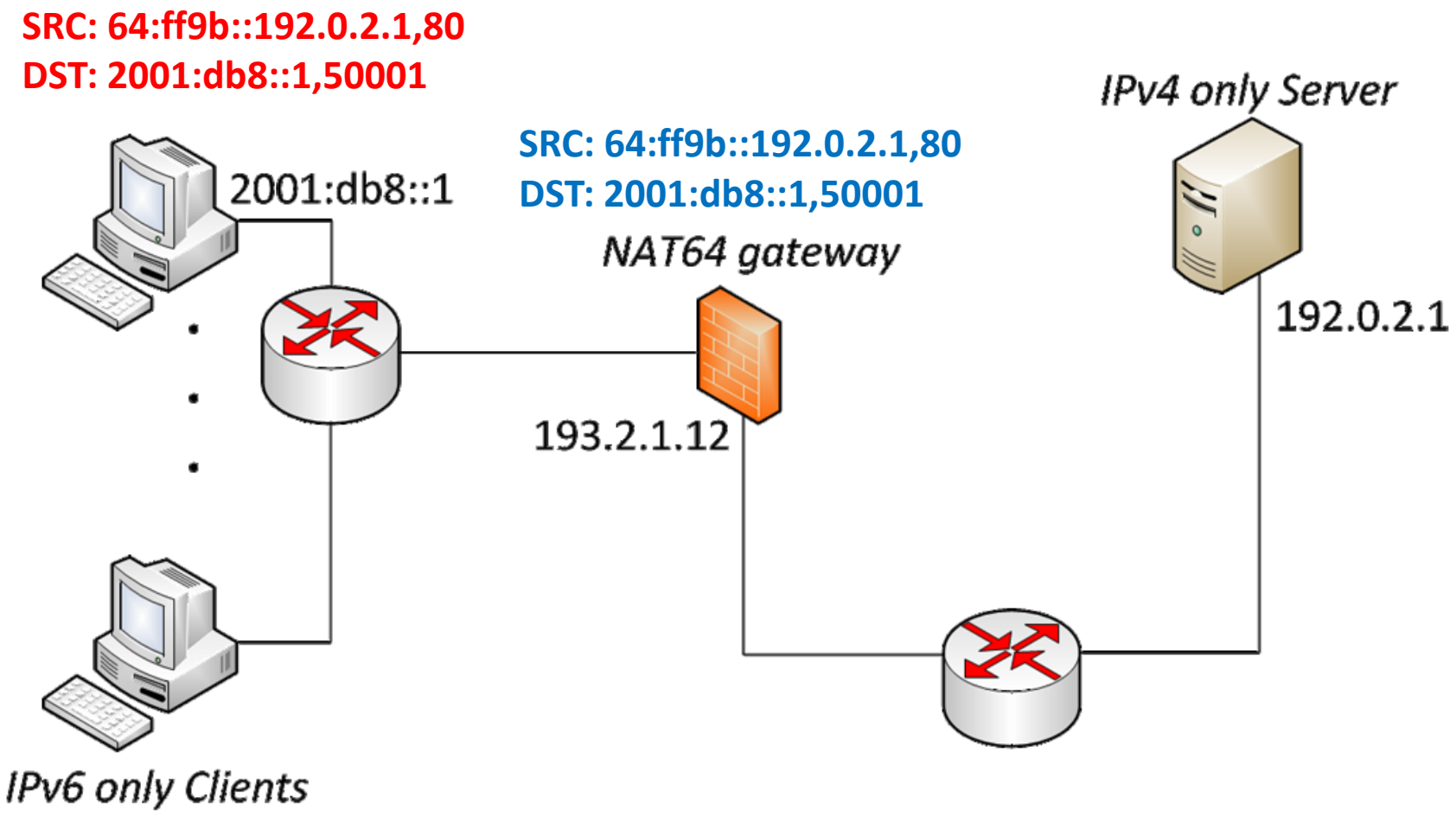
The Operation of NAT64



The Operation of NAT64



The Operation of NAT64



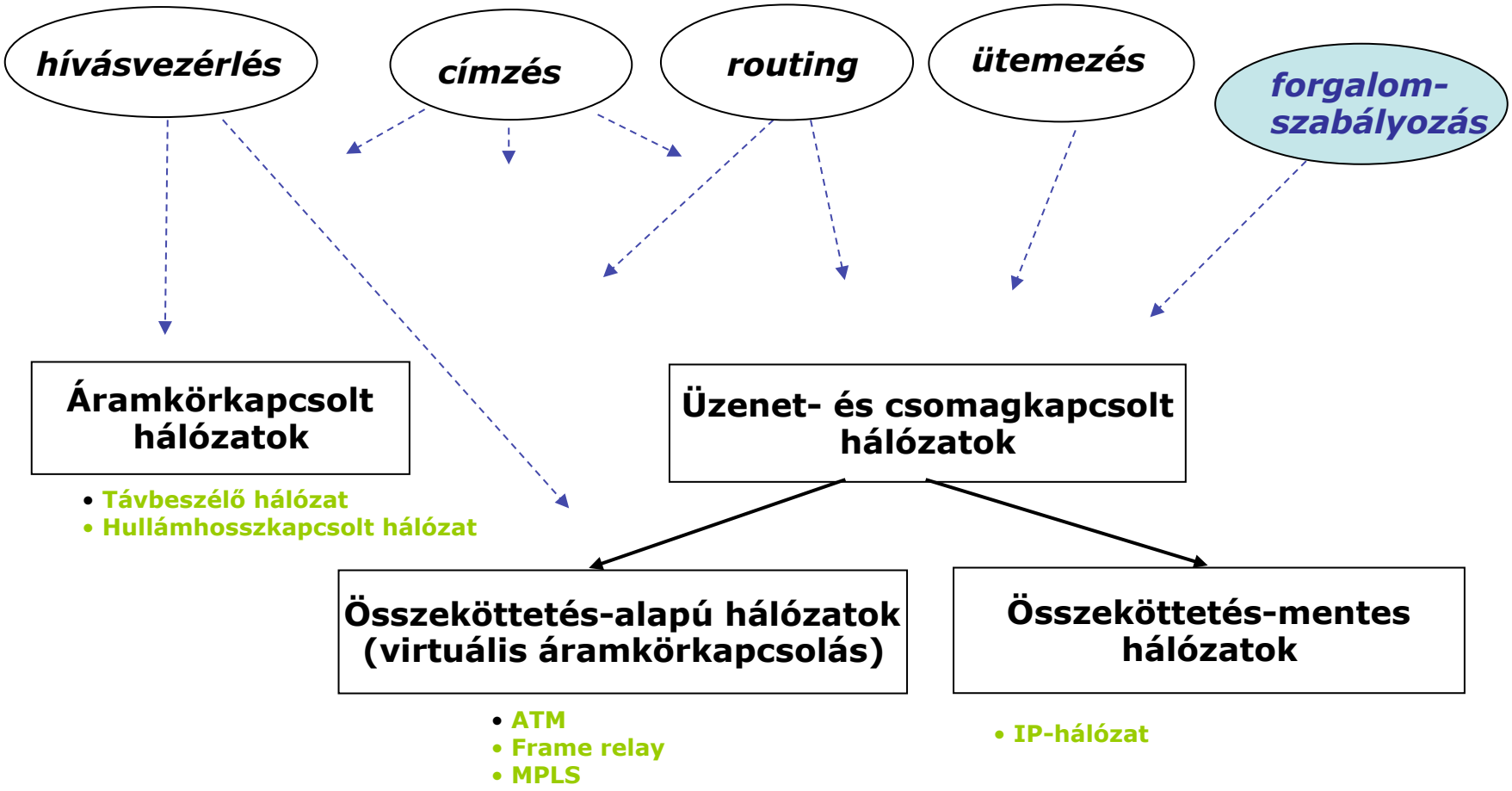
FORGALOMSZABÁLYOZÁS

Flow control

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2014.Április 15.

Újabb fontos funkciók és alapelvek



Definíció és a feladat meghatározása

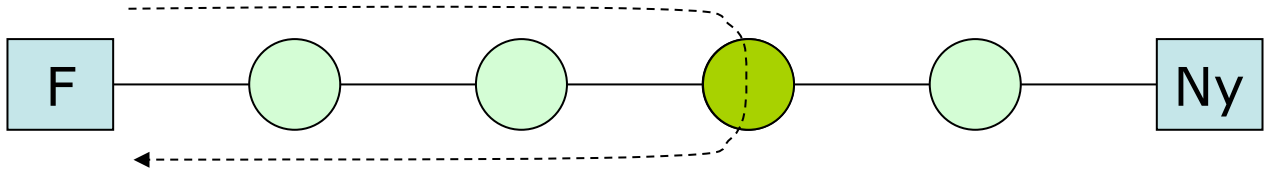
- Két rokon feladat:

- Forgalomszabályozás (*flow control*):
  - Módszerek, amelyek lehetővé teszik, hogy egy adatforrás az **aktuális átviteli sebességét illessze** a vevőnél és a hálózatban rendelkezésre álló kiszolgálási sebességhez

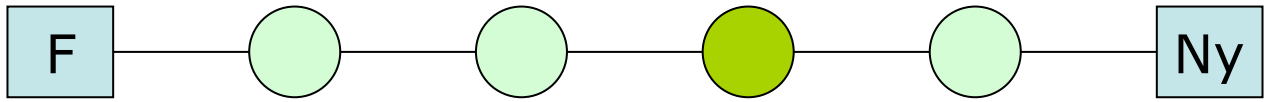
- Torlódásvezérlés (*congestion control*):
  - Azok a módszerek, amelyekkel linkek, csomópontok időszakos túlterheltségét megkíséreljük megszüntetni
    - A torlódásvezérlést értelmezhetjük általánosabban is: azok a módszerek, amelyekkel meg is lehet előzni a torlódásokat
    - Ebben az értelemben a *flow control* tekinthető a *congestion control* eszközének
      - előzzük meg a torlódást, mielőtt az bekövetkezne

A működés egyszerűsített vizsgálata

- Korlátozódjunk egyetlen forrásra:
  - A csomagok több csomóponton jutnak a nyelőhöz



  - Az út mentén valahol kialakul egy szűk keresztmetszet, „bottleneck”
  - A többi csomópont elfelejthető a forgalomszabályozás modellezése szempontjából



(A késleltetéshez természetesen valamennyi csomópont hozzájárul)

A forgalomszabályozás kívánatos jellemzői

- Az alapvető cél megvalósításán túl:
  - Legyen **egyszerűen** megvalósítható
  - **Lehető legkevesebb** hálózati erőforrást vegyen igénybe
  - Hatékonysága ne függjön a szabályozott **források számától**
  - Biztosítsa a szabályozott források **igazságos részesedését** az igénybe vett erőforrásokon
  - **Stabil** legyen a működése

A forgalomszabályozás fajtái

- Két jelentős csoport:
  - „Nyílthurkú” – nincs visszacsatolás
  - „Zárthurkú” – van visszacsatolás
- Lehetséges a kettő kombinációja is (hibrid)

- Nyílthurkú:
  - A kommunikáció előtt a felhasználó és a hálózat forgalmi paramétereket egyeztet
  - A hálózat dönt az új összeköttetés elfogadásáról
    - *admission control, beengedésszabályozás*
  - Ennek megfelelően a hálózat **erőforrásokat dedikál**
  - A működés során a **paraméterek ellenőrzése**

Nyílthurkú szabályozás

- **Forgalomleírók (traffic descriptors):**
  - Egy paraméterkészlet, amely jellemzi az adatforrás viselkedését, továbbá
  - alapját képezi egy *szolgáltatási szerződés forgalmi részének*

- Egyszerű példák forgalomleírókra:
  - csúcssebesség
  - átlagsebesség

- A forgalomleírók bemenő adatai:
  - a forgalomszabályozónak (regulator), valamint
  - a felügyelőnek (policer)

A *regulátor* tipikusan késlelteti a túlzott forgalmat, míg a *policer* inkább eltávolítja

Zárthurkú szabályozás

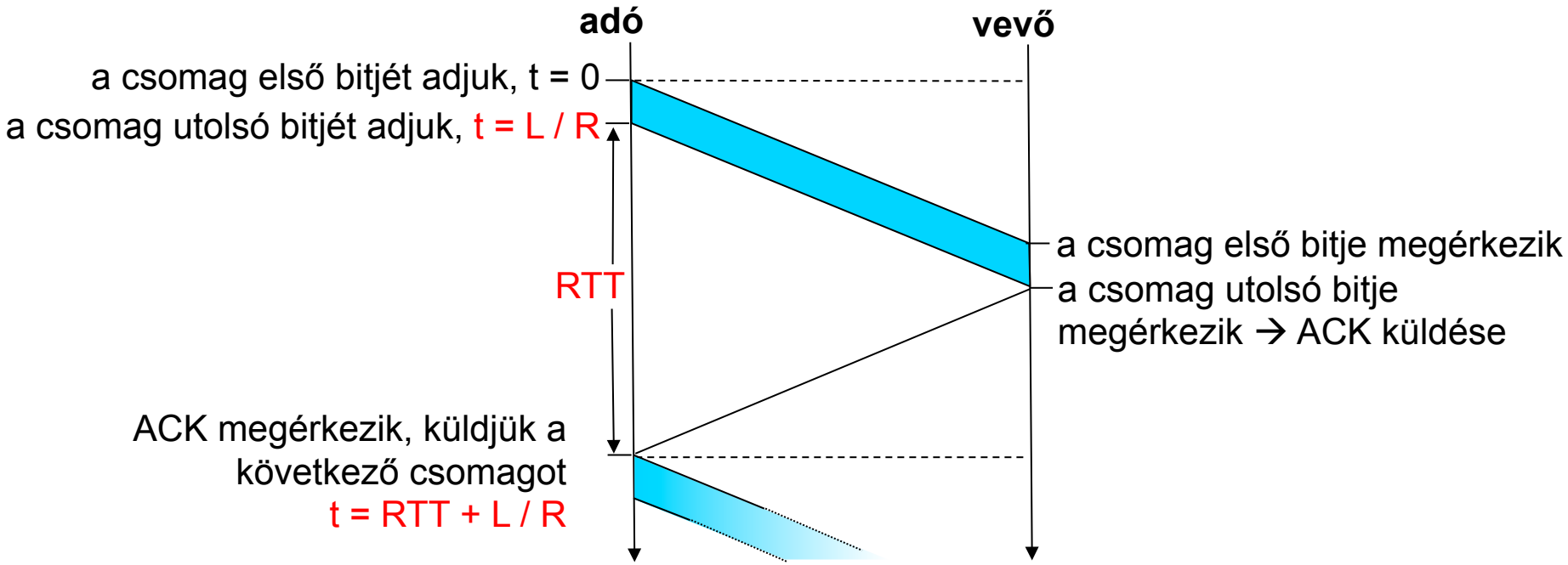
- Feltétlenül szükséges, ha:
  - nincs erőforrás-foglalás
  - túlfoglalást (*overbooking*) alkalmazunk statisztikus nyereség elérése érdekében
- Típusai:

1. generáció (csak a nyelő képessége)	**ki – bekapcsolás (on-off)**		
	stop-and-wait		
	statikus ablak (static window)		
2. generáció (a nyelő és a hálózat képessége)	**állapot vizsgálat**	**vezérlés módja**	**vezérlés helye**
	explicit	**din. ablak**	**végpont**
	implicit	**din. seb.**	**lépések**

A felsorolt módszerek

- *On-off:*
  - a nyelő engedélyezi az adást
- *Stop-and-wait:*
  - a küldő egy csomag után vár a nyugtára
- *Statikus ablak:*
  - a küldő az ablak méretével megadott számú csomag elküldése után vár csak nyugtára
- *Második generációs módszerekre a protokolloknál látunk majd példát*

A stop-and-wait működése



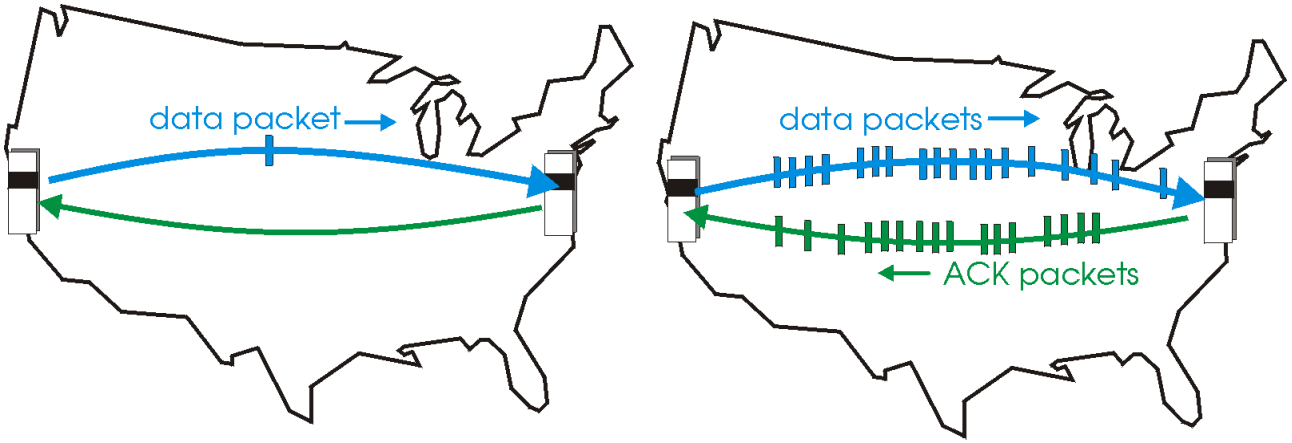
A kihasználtság: $U = \frac{L / R}{RTT + L / R}$

RTT: round-trip-time (teljes körbefordulási idő)

Statikus ablak: több egymás utáni, még nem nyugtázott csomag adása

Az adónak lehet több, adásban lévő, még nem nyugtázott csomagja

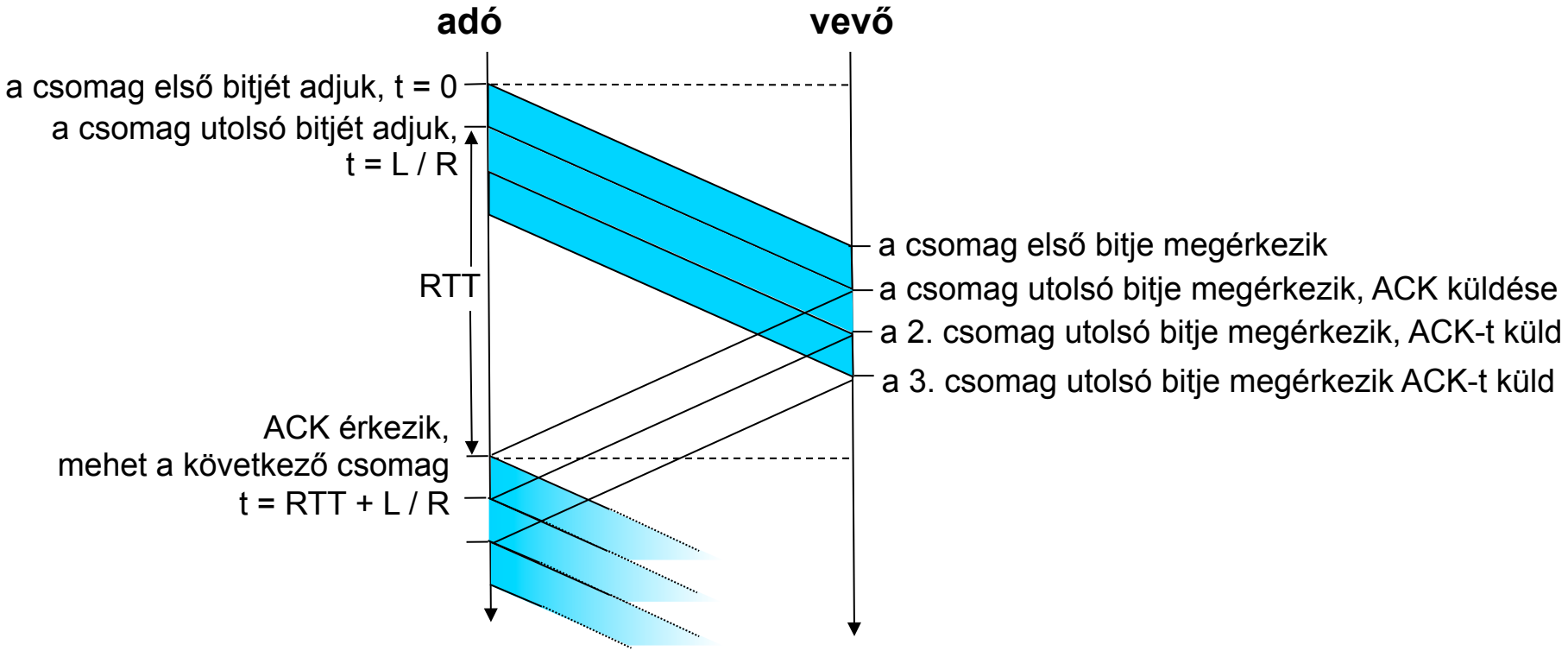
- sorszámozásra van szükség
- tárolásra van szükség az adóban



(a) a stop-and-wait protocol in operation

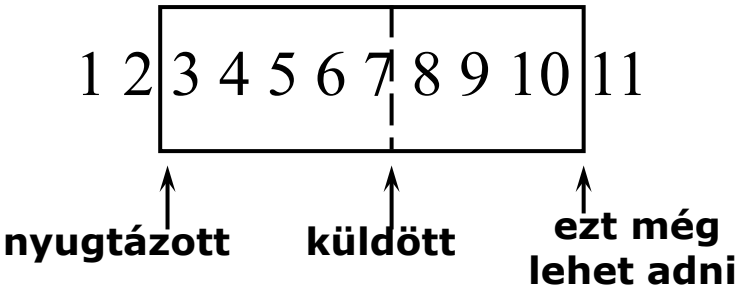
(b) a pipelined protocol in operation

Nagyobb kihasználtság a stop-and-wait-hez képest



$$U_{sender} = \frac{3 * L / R}{RTT + L / R}$$

Csúszóablakos („sliding window”) forgalomszabályozás



Várakozás ACK-ra: **time–out**

- Mekkorára válasszuk a time–out-ot?
- Probléma a túl kicsivel és a túl naggyal
- Megoldás: a teljes körbefordulási időhöz (RTT) igazítani
  - **adaptívvá** tenni
- Szabályok arra, hogy mit tegyünk, ha nem jön ACK a time-out alatt

Összefoglalás

- A küldő által kibocsátott adatátviteli sebességet kordában kell tartani, mert
  - a vevő nem győzi annak feldolgozását
  - a hálózatban képződő szűk átviteli keresztmetszetek torlódáshoz vezethetnek

- Két alapvetően eltérő megközelítés:
  - az információtovábbítást megelőzi egy szerződéskötés
  - folyamatosan befolyásoljuk a küldőt

Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

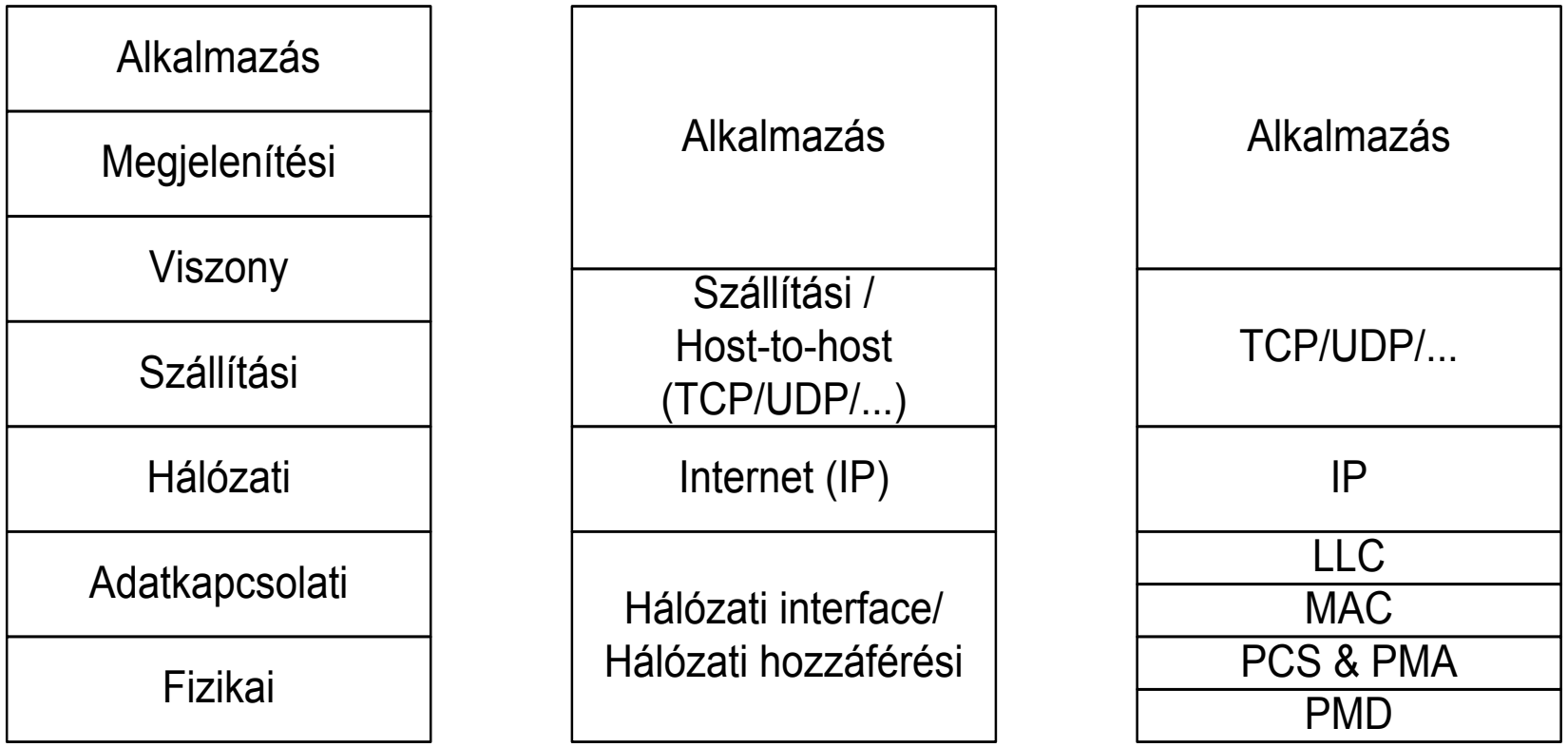
SZÁLLÍTÁSI (TRANSPORT, HOST-TO-HOST) PROTOKOLLOK

UDP és TCP

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2014.Április 15.

A TCP/IP architektúra és az ISO/OSI rétegmodell



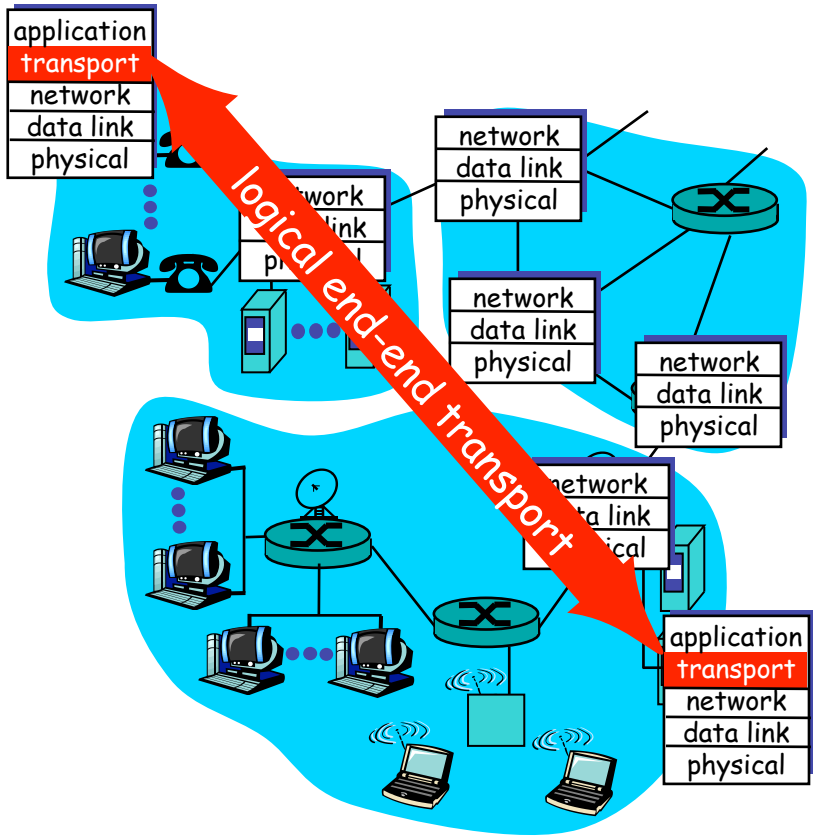
IP: Internet Protocol
TCP: Transmission Control Protocol
UDP: User Datagram Protocol
LLC: Logical Link Control

MAC: Medium Access Control
PCS: Physical Coding Sublayer
PMA: Physical Medium Attachment
PMD: Physical Medium Dependent

A hálózati és a szállítási réteg

- *hálózati réteg:* végpontok („host”-ok) közötti logikai kapcsolatok
- *szállítási réteg:* alkalmazások (process) közötti logikai kapcsolatok
  - a hálózati réteg szolgáltatásainak igénybevételére alapozva **megbízható** átvitel a **transzport entitások** között (transport entity)
  - Feladatai:
    - forgalom szabályozás
    - multiplexelés
    - hibadetektálás, javítás (pl. Automatic Repeat reQuest – ARQ)
    - sorrendhelyes átvitel
    - csomagképzés (szegmensek) és csomagok visszaállítása a felsőbb rétegek számára
      – byte alapú átvitel

A szállítási réteg



Logikai kapcsolatok a végpontokban futó **alkalmazások között**

A szállítási protokollok a **végpontokban futnak**, a csomópontokban nem

Az alkalmazások adategységeit szállítási protokoll-adategységekbe tördeljük, a kapottakból pedig összerakjuk

Szállítási protokollok: UDP és TCP

- **UDP – User Datagram Protocol**
- **TCP – Transmission Control Protocol**

- Az UDP és TCP közös képességei:
  - Portok kezelése
  - Multiplexelési képesség

- Alapvető különbség az UDP és a TCP között:
  - UDP összeköttetésmentes (connectionless),
  - TCP összeköttetés-alapú (connection–oriented)

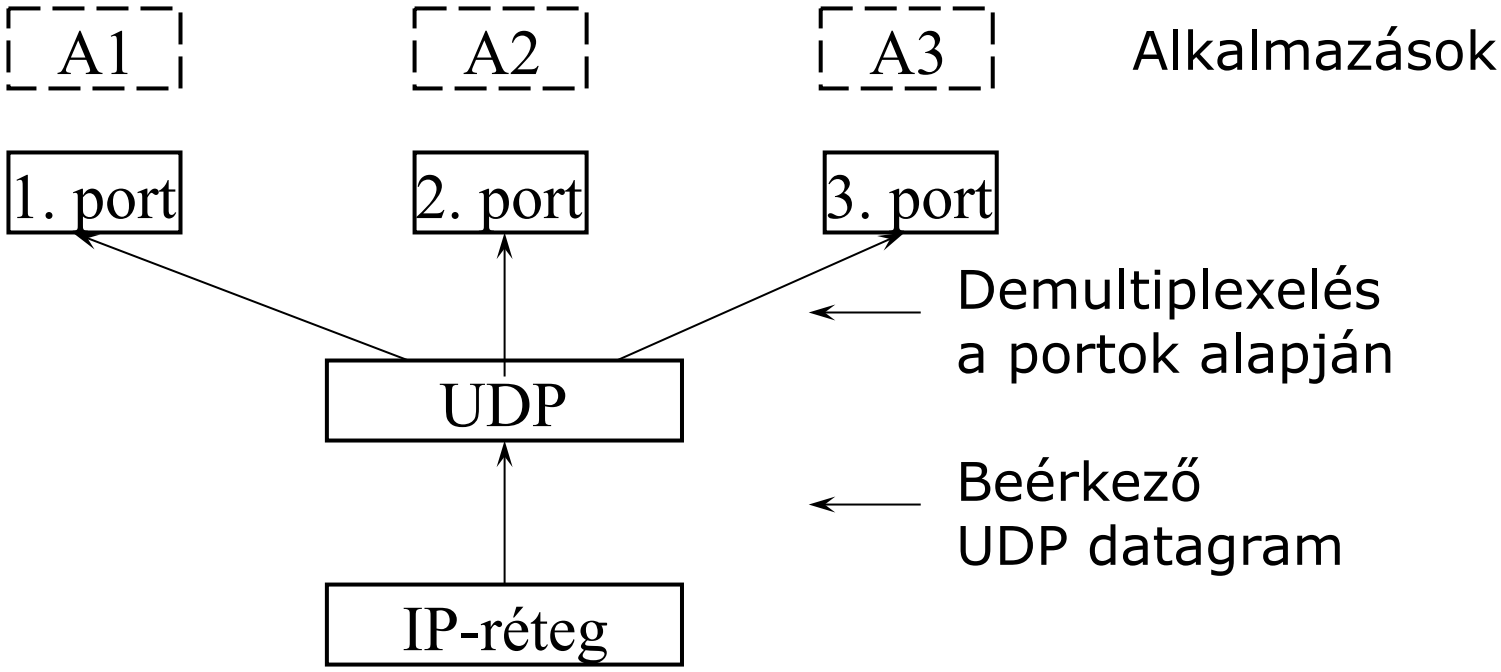
  transzport-szolgáltatást nyújt

Az UDP és TCP közös képességei (1)

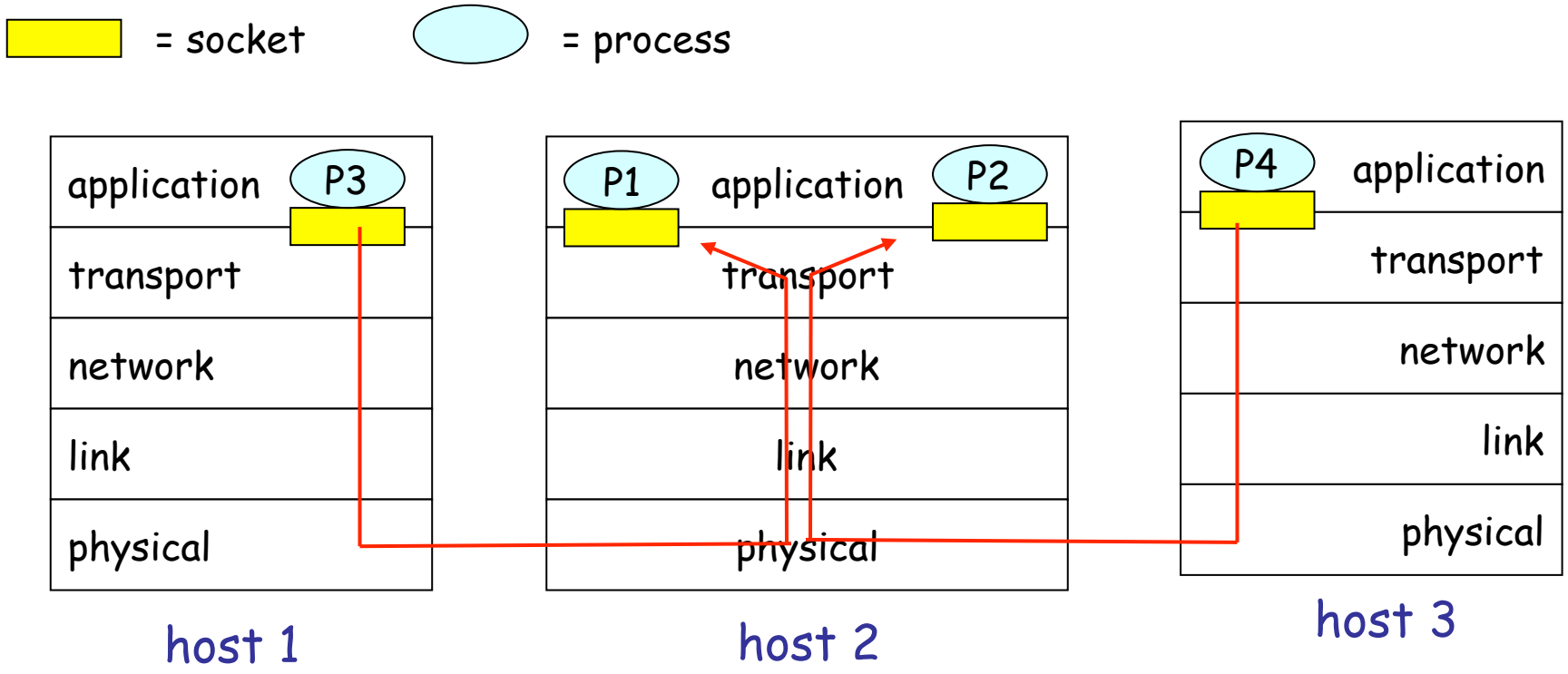
- Portok kezelése:
  - Az IP-rétegben a csomagok végpontnak, „host”-nak vannak címezve
  - A végpontokon belül: több alkalmazás, folyamat
  - Megkülönböztetésük: portok használatával
  - **Foglalt** (reserved, „well-known”) és **rendelkezésre álló** (available) portszámok
  - Foglalt portok: ide mindig lehet küldeni datagrammokat
    - 0...1023 közötti portszámok
    - pl. 80: HTTP, 21: FTP, 69: TFTP (ezek főként TCP-re)
  - Az UDP-en belül megállapításra kerülnek az alkalmazandó portszámok
- Multiplexelés /demultiplexelés
  - A portmechanizmus segítségével

Az UDP és TCP közös képességei (2)

Multiplexelés és demultiplexelés
Példa:



Multiplexelés-demultiplexelés



Socket: magyarázat

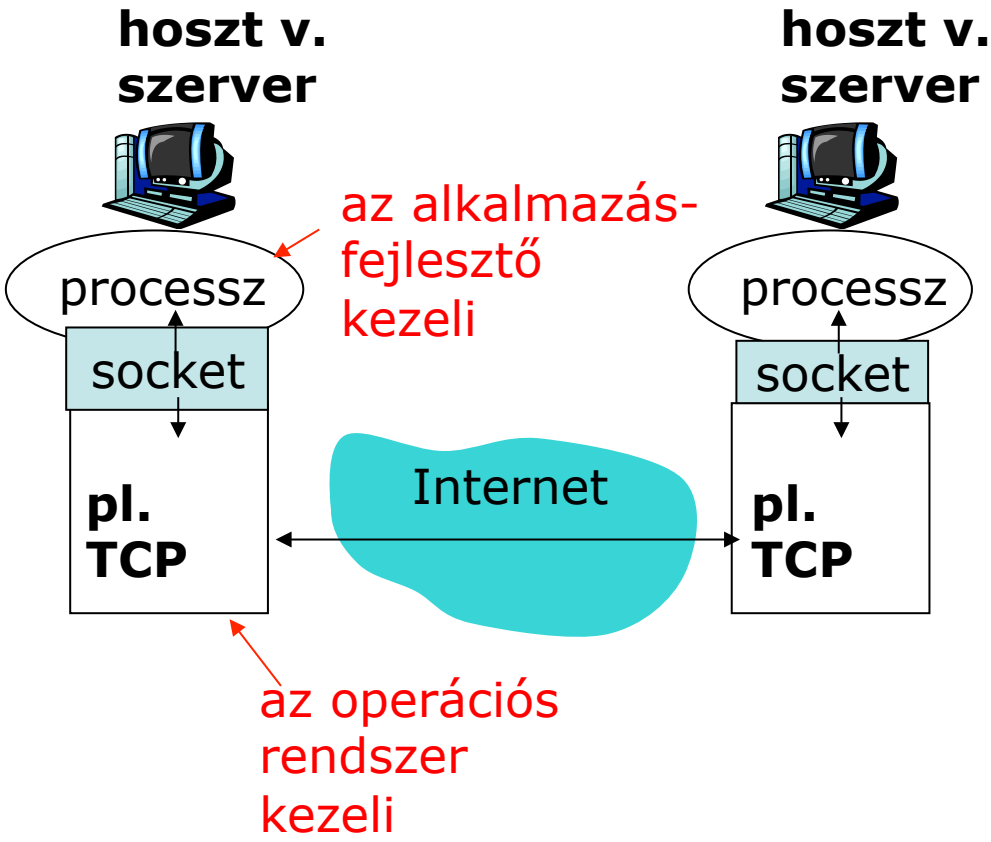
- Socket: interfész, „ajtó” az alkalmazás és a hálózat között
- A socket-et az alábbiak jellemzik:
  - transzportprotokoll (UDP v. TCP)
  - saját IP cím
  - saját portszám

    } helyi socket cím
  - *(opcionális) távoli host IP címe*
  - *(opcionális) távoli alkalmazás portszáma*

    } távoli socket cím

  } socket pár
- Leegyszerűsítve: socket = IP cím + portszám
- Az operációs rendszer a bejövő IP csomagokat a fentiek alapján továbbítja az alkalmazásnak, kiszedve ezeket a megfelelő PDU-kból

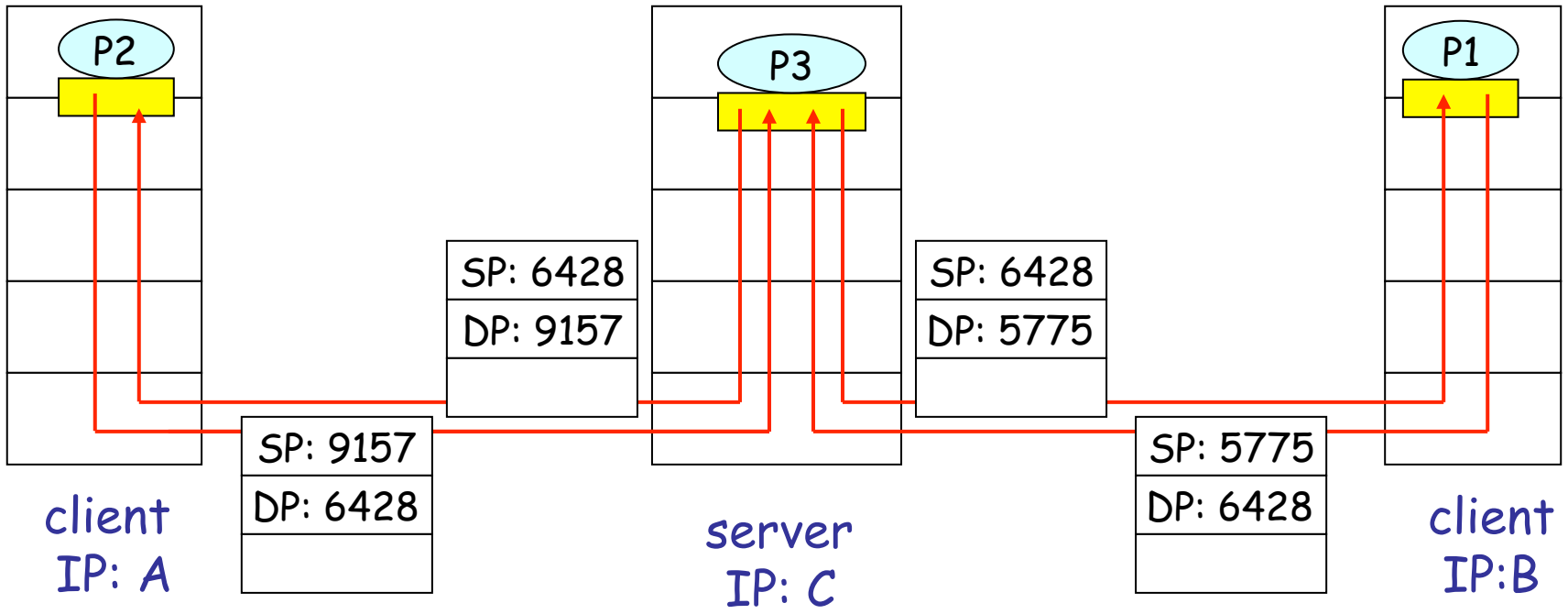
Socket: illusztráció



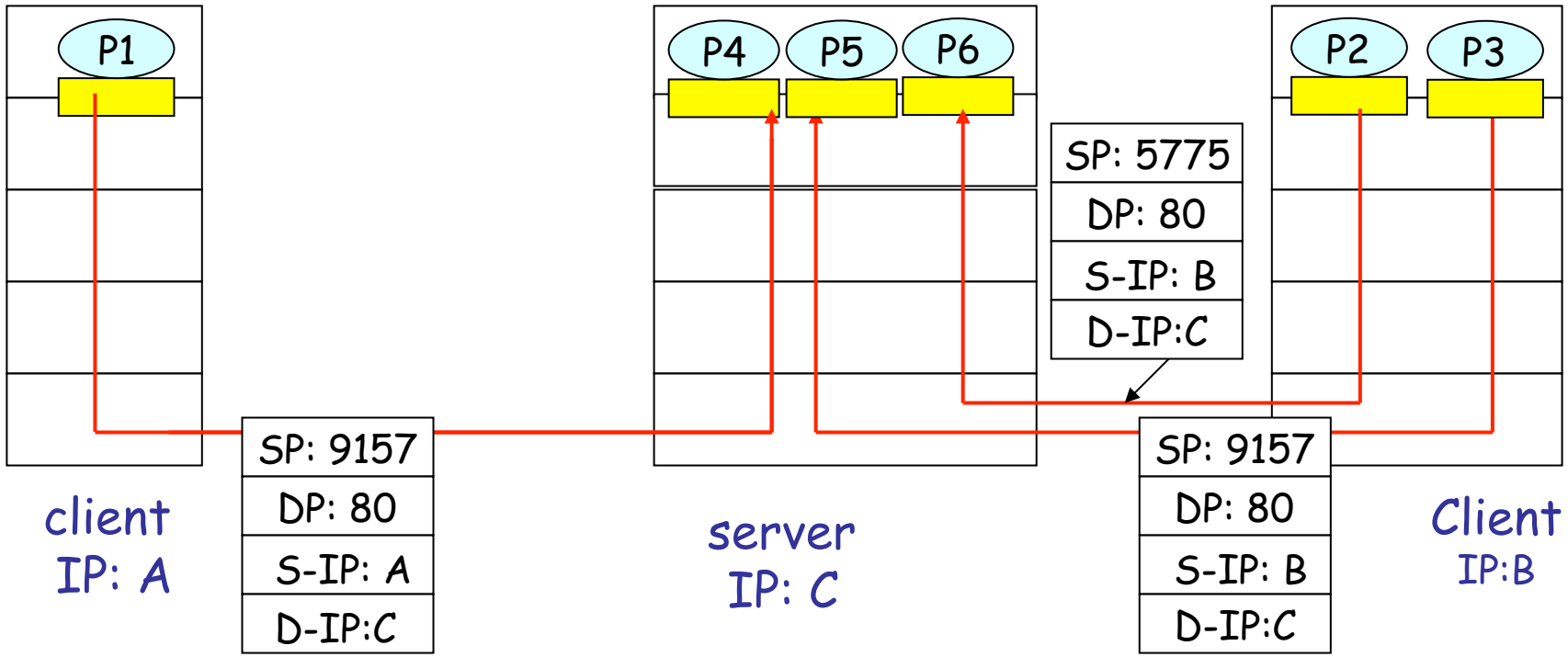
Socket típusok

- Datagramm socket
  - Összeköttetésmentes socket, UDP használja
- Stream socket
  - Összeköttetés-alapú socket, TCP használja
- Raw (IP) sockets
  - Routerekben és hálózati eszközökben
  - Pl. ICMP, IGMP, OSPF

Multiplexelés-demultiplexelés, összeköttetésmentes eset (UDP)



Multiplexelés-demultiplexelés, összeköttetés-alapú eset (TCP)



TCP kliens-szerver socket kezelés

- Szerver létrehoz **„hallgató” módban** lévő socketeket
  - várja a kliensek kapcsolatfelvételét

- Kapcsolatfelvétel után: **dedikált socket** minden kapcsolathoz
  - Socket-socket közötti virtuális áramkörkapcsolás: TCP kapcsolat, duplex byte folyam

- Szerver különböző TCP socketeket hozhat létre ugyanazzal a helyi IP címmel és port számmal
  - mivel a távoli host IP címével, portjával **más socket párt alkot**
  - szerver gyerek processz összerendelése a kliens processzével

- UDP-ben **nincs dedikált socket**
  - Nincs külön gyerek processz minden távoli processzhez, egy processz kommunikál velük

És ahol nincs transzport réteg implementálva?

- Egyes hálózati elemekben nincs implementálva a transzport réteg
  - Router hálózati rétegben
  - Switch adatkapcsolati rétegben
- De tűzfalak, NAT-ok és proxy szerverek figyelik az aktív socket párokat és fenntartanak socket interfészeket
- Továbbá: ütemezéshez, QoS támogatáshoz routerekben a csomagfolyamokat a socket párokkal lehet azonosítani

Megoldás: Egyszerű (raw) socket

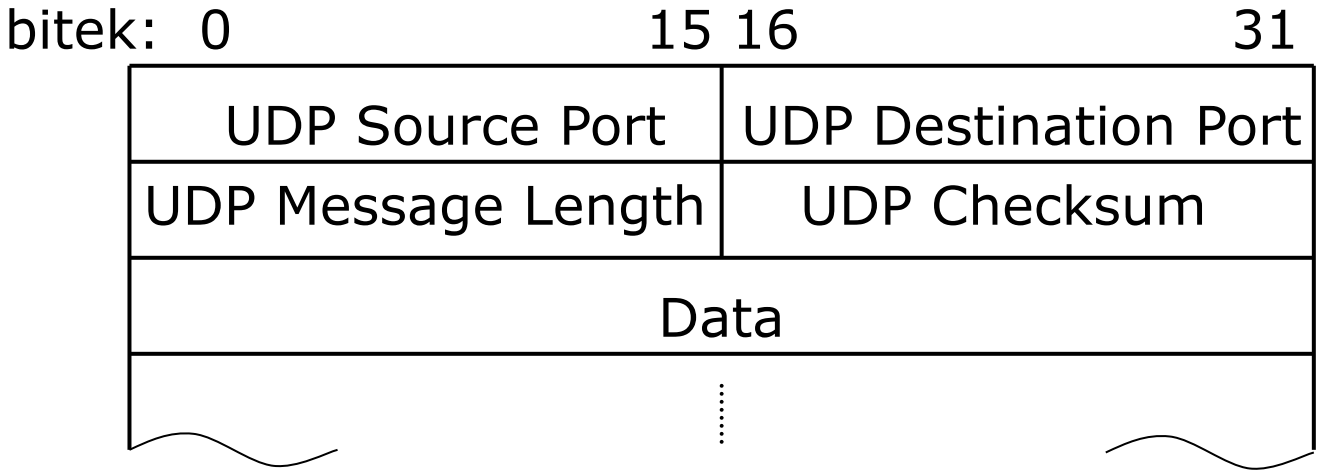
- **Közvetlenül** az alkalmazásnak továbbítja a csomagot a **fejléccel**
  - Nincs TCP/IP feldolgozás, alkalmazás látja el fejléccel/veszi le a fejlécet

- Windows XP-ben 2001-ben jelent meg: biztonsági aggályok (Unixban régóta)
  - Félelem **TCP reset támadástól**: „lelövi” a TCP kapcsolatot
    - +:gyanús kapcsolatok megszakítása
    - -: harmadik fél beékelődve megszakítja a kapcsolatot
    - +/- ? Peer-to-peer forgalom szűrése (Comcast-NNSquad eset)

UDP (User Datagram Protocol)

- Nincs kapcsolatfelépítés, kézfogás
- Semmi garancia:
  - a csomagok sorrendjére
  - adatintegritására
  - elvesztésének detekciójára
- A megbízható átvitel garantálását az alkalmazási rétegre bízza
- *Unreliable* Datagram Protocol ☺
- Multiplexálást és opcionálisan adatintegritás ellenőrzést nyújt

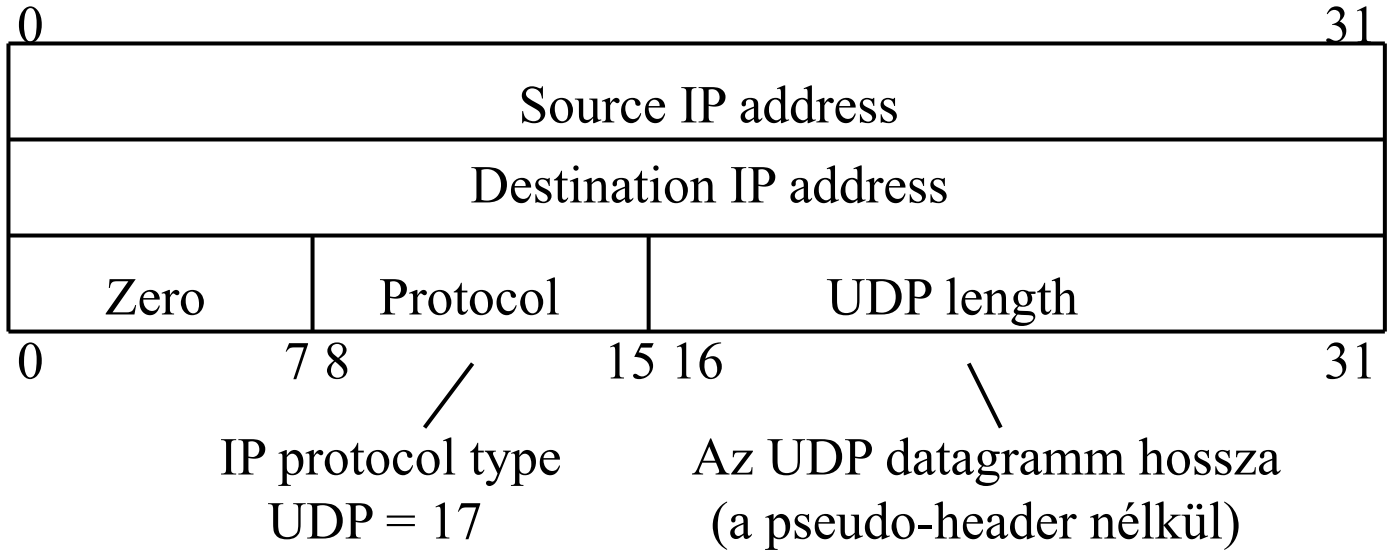
UDP – User Datagram Protocol (2)



- **Source port** opcionális
- **Length**: oktettben, max (65 535-8-20)
- **Checksum**: opcionális (nincs: 0, IPv6-nál már nem opcionális)
- Az UDP checksum az egyetlen lehetőség annak ellenőrzésére, hogy a datagram helyesen érkezett-e meg
  - **adatra és fejlécre!**
  - az IP-csomag ellenőrzése csak a fejrészt fogja át

UDP – User Datagram Protocol (3)

- A checksum számítási elve
  - 1-es komplemens 16 bites szavakra
- Tartalmazza az IP-címeket is
  - annak ellenőrzésére, hogy a datagramm elérte a helyes címzettet, nemcsak a helyes portot
- „Pseudo-header” hozzáadásával (IPv4-re):



UDP – User Datagram Protocol
Összefoglalás: funkcionalitás és a költsége

- Portkezelés
  - a különböző alkalmazások/folyamatok megkülönböztethetők
- Több alkalmazás egyidejű kezelése
  - port-hozzárendeléssel és multiplexeléssel/demultiplexeléssel
- Hibajelzés az UDP datagram tartalmára és az IP csomag további részeire
- A fentiek költsége:
  minimum 8 oktettnyi overhead

UDP alkalmazása

- Broadcast, multicast (TCP nem képes rá)
- Média: streaming, valós idejű játékok, VoIP, IPTV
- Gyors és rövid lekérdezések:
  - DNS
  - DHCP
  - RIP
- Tipikusan a hálózati forgalom pár százaléka
  - De növekszik a részaránya és nincs torlódás szabályozás: aggályok!
  - Datagram Congestion Control Protocol (DCCP)

TCP –Transmission Control Protocol
Fő jellemzői

- **Célkitűzés:** megbízható szállítási szolgáltatás nyújtása az IP nem megbízható datagram-szolgáltatásán

- **Jellemzői:**
  - ***Virtuális összeköttetések:*** összeköttetés épül fel és marad fenn a kommunikáció tartamára
  - ***Stream–típusú szolgáltatás:*** byte- (oktett-) streamek sorrendhelyes átvitele
  - ***Strukturálatlan stream:*** nincsenek határolók a streamen belül
  - ***Pufferelt átvitel:*** a streamből a datagramm megtöltéséhez szükséges mennyiséget várja össze
  - ***Duplex kapcsolatok:*** két független stream
  - ***Vezérlő információk küldése:*** az ellenkező irányban folyó streambe ágyazva (**piggybacking**)

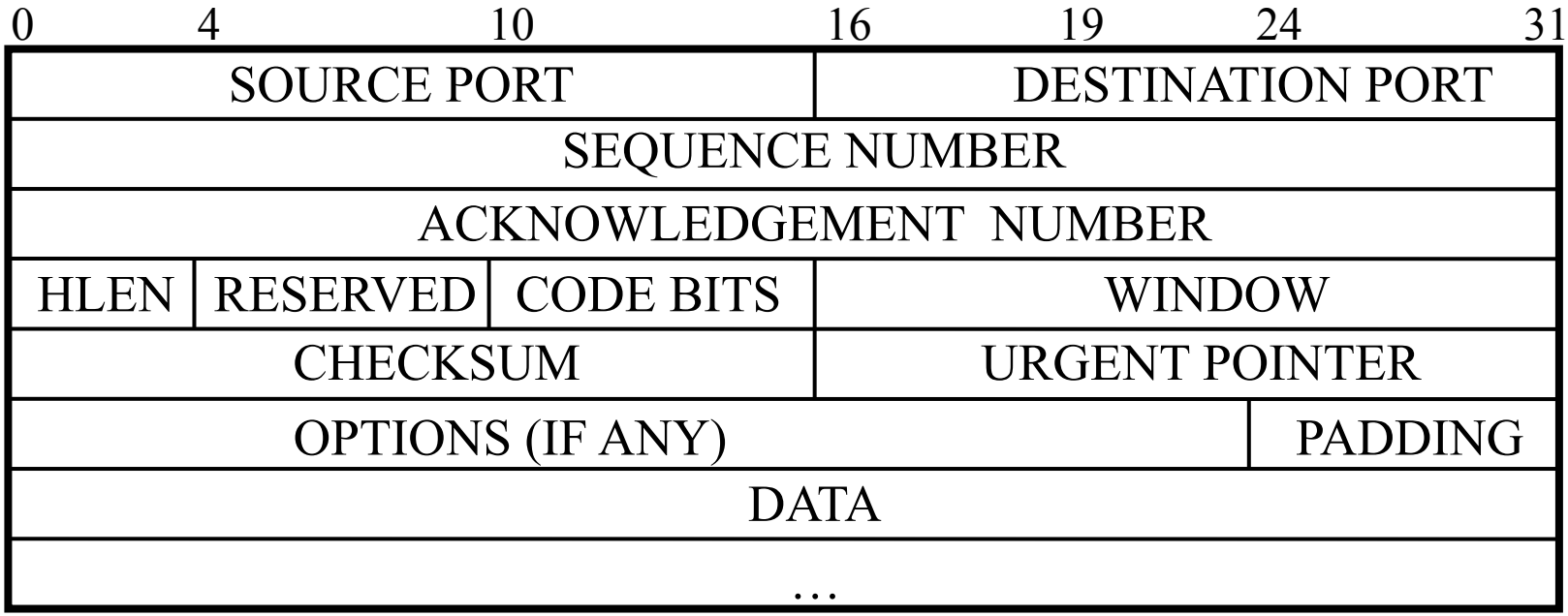
TCP – Transmission Control Protocol
Miről lesz szó?

- **Adategység**: szegmens; szegmensstruktúra
- **Megbízható átvitel** sorszámozás és pozitív nyugtázás segítségével
- Összeköttetés-alapú kommunikáció: **kapcsolatfelépítés és -lebontás**
- **Forgalomszabályozás** (flow control) ablakmechanizmus segítségével
- **Torlódásvezérlés** (congestion control)

TCP kialakulása

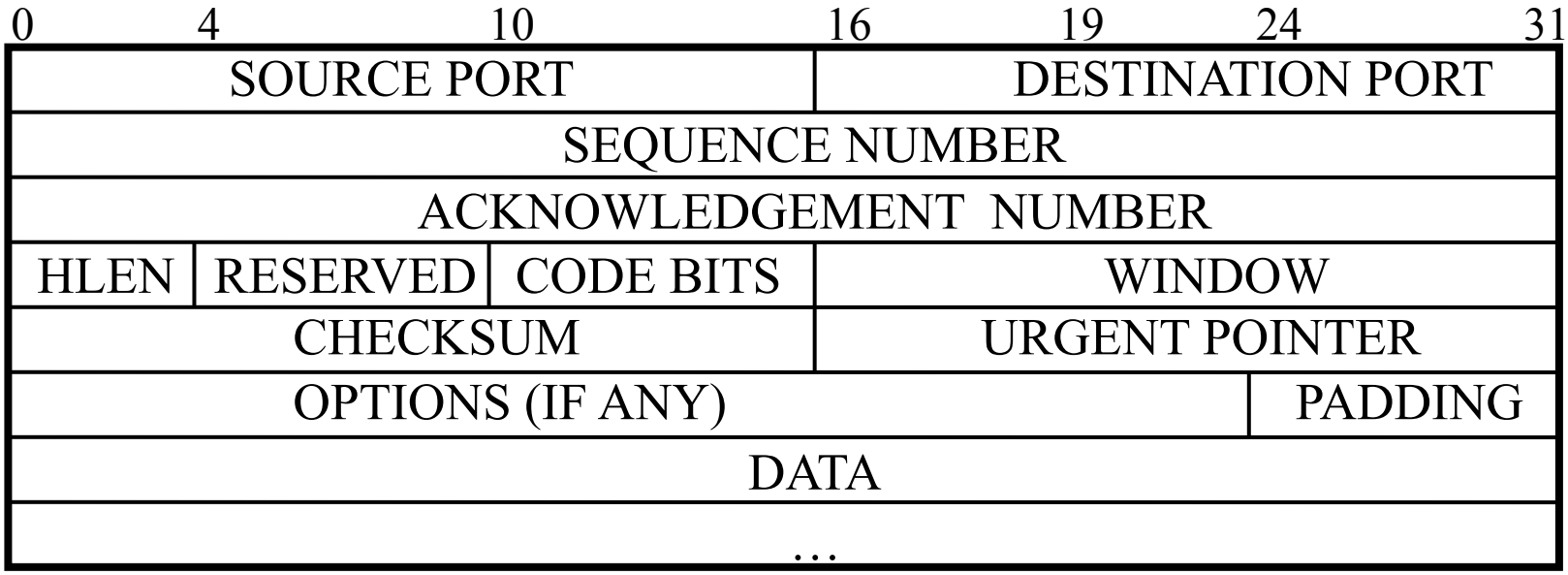
- 1974: "*A Protocol for Packet Network Interconnection.*" Vint Cerf és Bob Kahn
- Itt vezetik be a *Transmission Control Program*-ot, ez válik szét később TCP-re és IP-re
- Eleinte Internet Protocol Suite, majd TCP/IP elnevezés

A PDU – Protocol Data Unit (a TCP-ben: „segment”) struktúrája



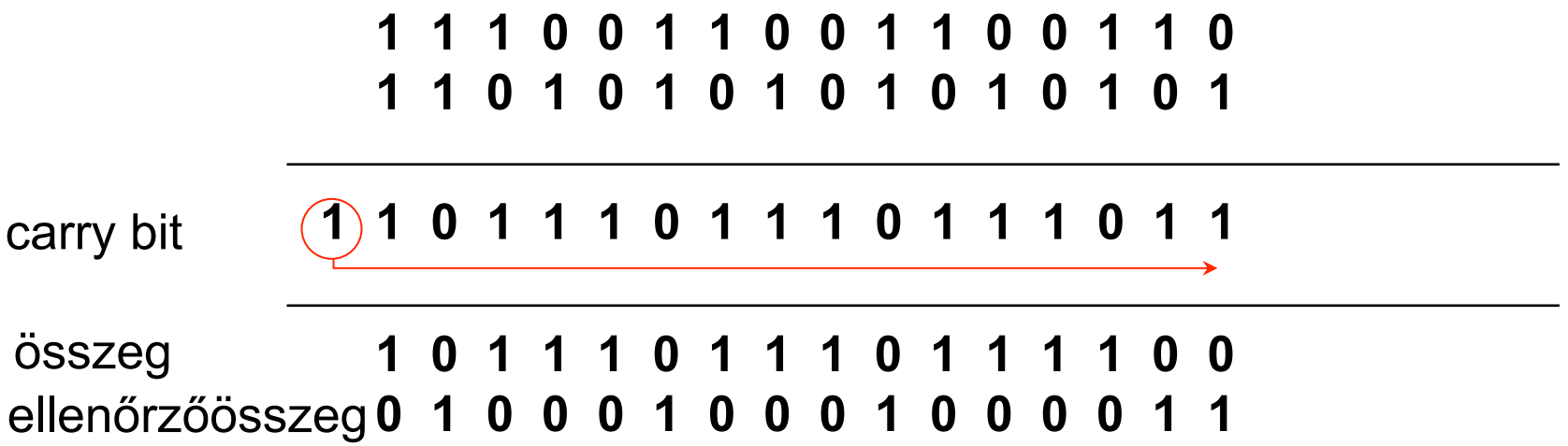
- **Sequence no**.: a szegmensben levő adat első byte-jának pozíciója a küldő byte stream-jében
- **Ack no**.: annak a byte-nak a sorszáma, amelyet a forrás legközelebb vár

TCP – Transmission Control Protocol
Szegmensformátum (folyt.)



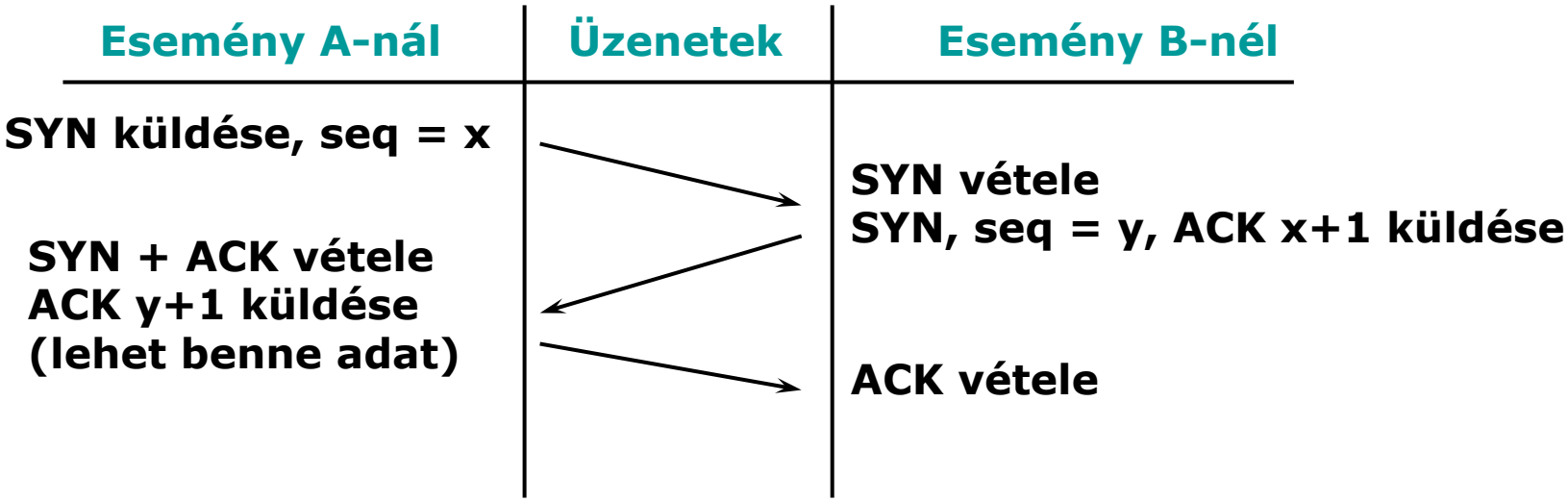
- **HLEN**: fejléc mérete, minimum 20 byte, max 60 byte
- **Code bits** (flags): URG, PSH, ACK, RST, SYN, FIN bitek
  a kapcsolat kezeléséhez használt jelzőbitek
- **Window**: a küldő ismertté teszi a **vételi pufferének méretét**
- **Checksum**: mint az UDP-ben (pszeudorandom)
- **Urgent pointer**: ha URG=1, a szegmens „urgent” részt tartalmaz, ilyenkor a végére mutat (pl. jelszóküldés)

Checksum – ellenőrző összeg



Hívásfelépítés a TCP-ben

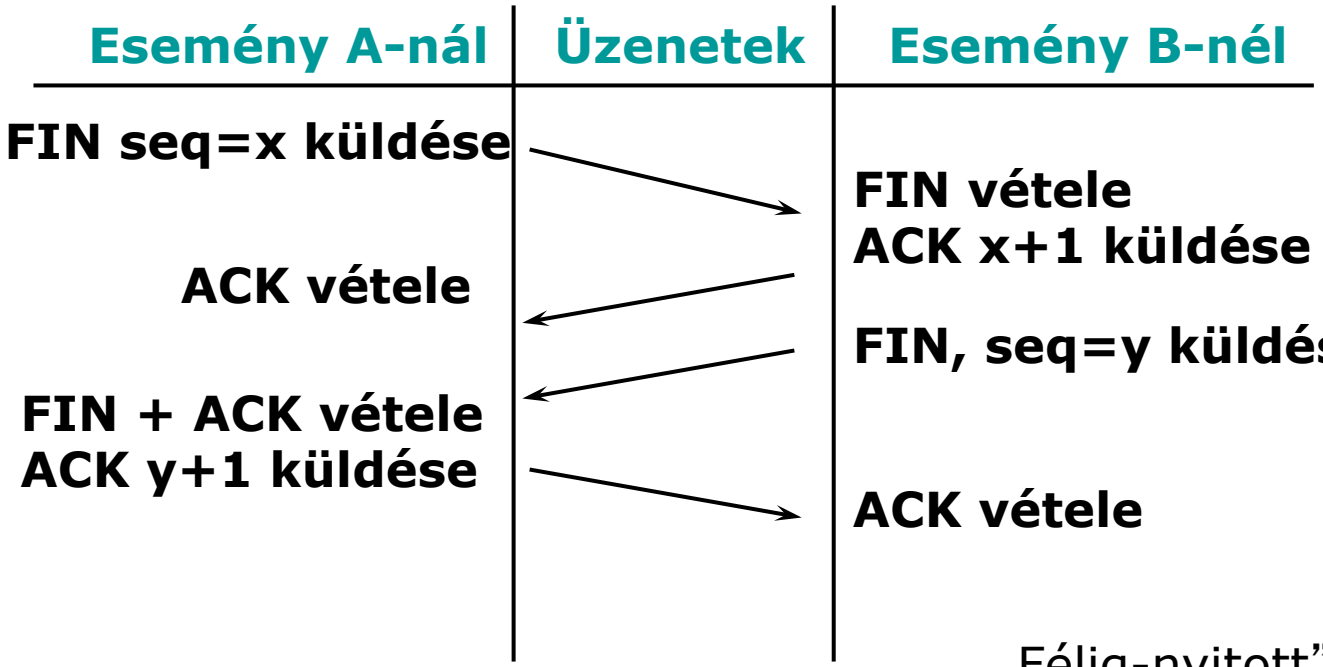
„3-way handshake” eljárás (3 utas kézfogás)



Elején véletlenszerű sorszám: a TCP sorszám predikciós támadás elkerülése

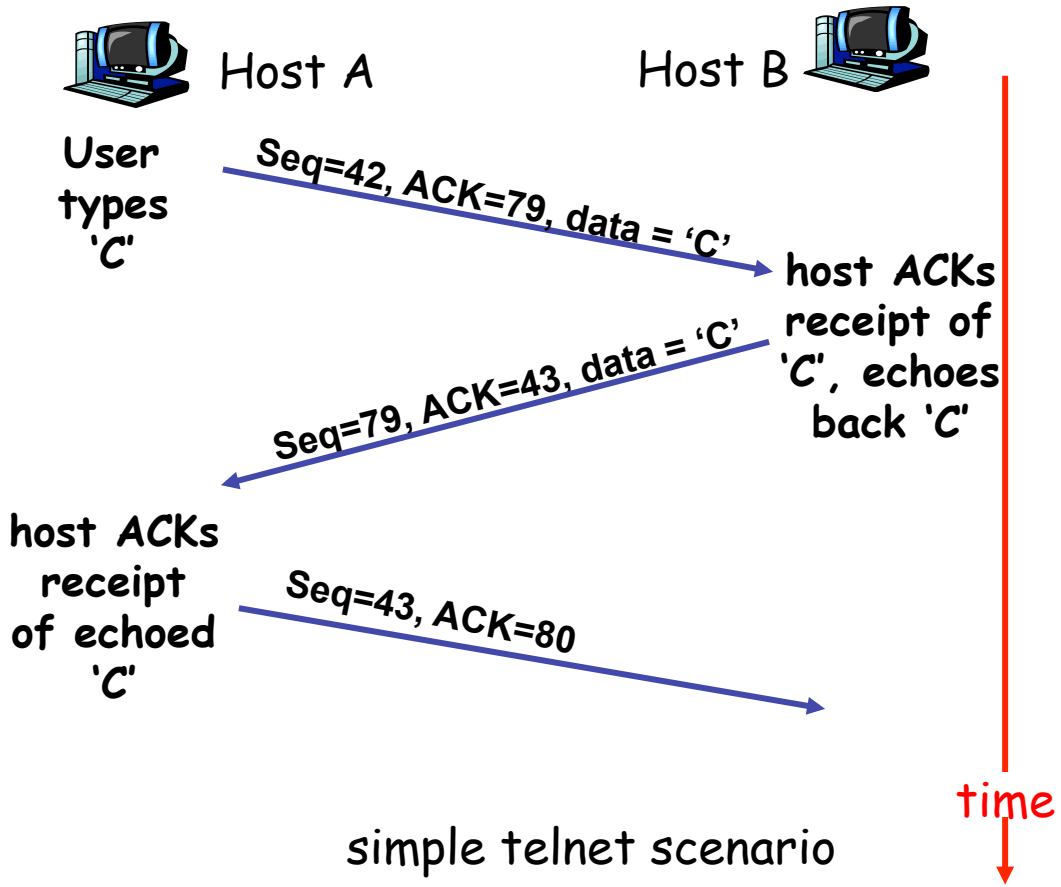
Híváslebontás a TCP-ben

„Modified 3-way handshake” eljárás



„Félig-nyitott” kapcsolat elkerülése

Sorszámok és nyugtaszámok használata a TCP-ben



simple telnet scenario

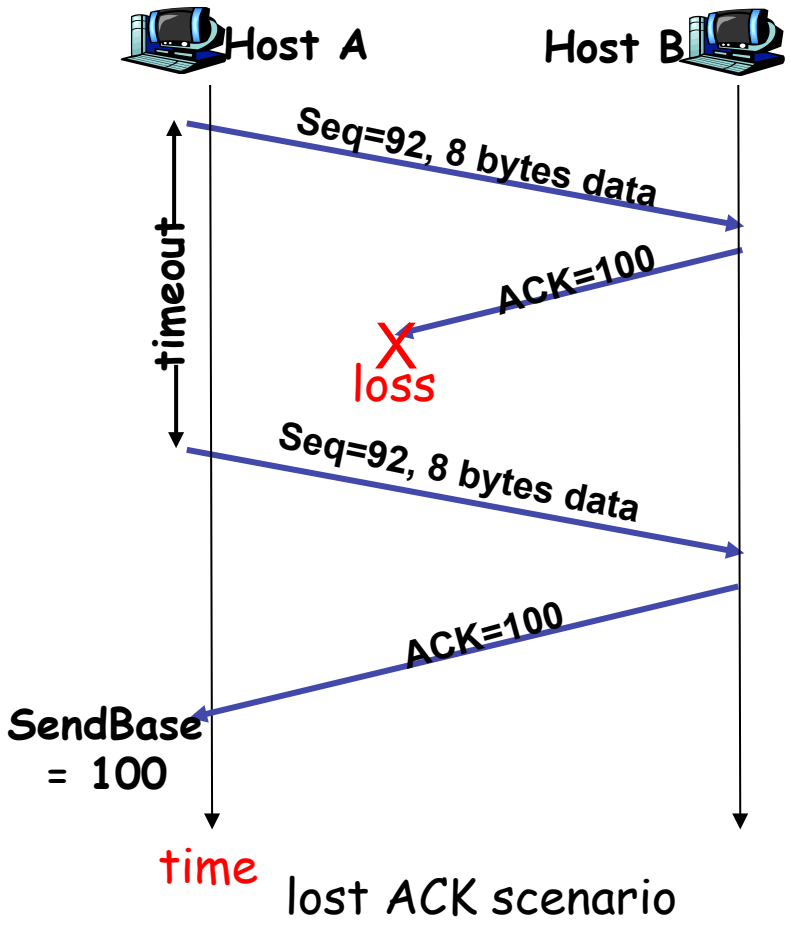
Várakozás nyugtázásra

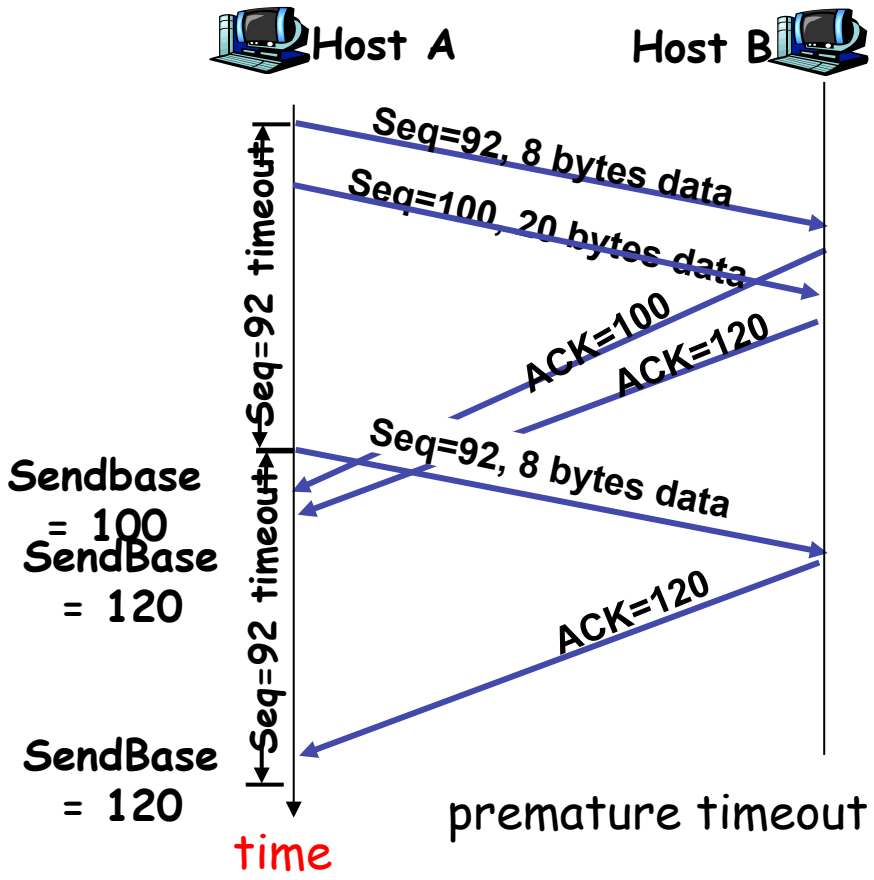
- Várakozás ACK-ra: **time–out**
- Mekkorára válasszuk a time–out-ot?
- Probléma a túl kicsivel és a túl naggyal

- Megoldás:
  a teljes terjedési időhöz
  (RTT - round-trip time) igazítani, **adaptívvá** tenni

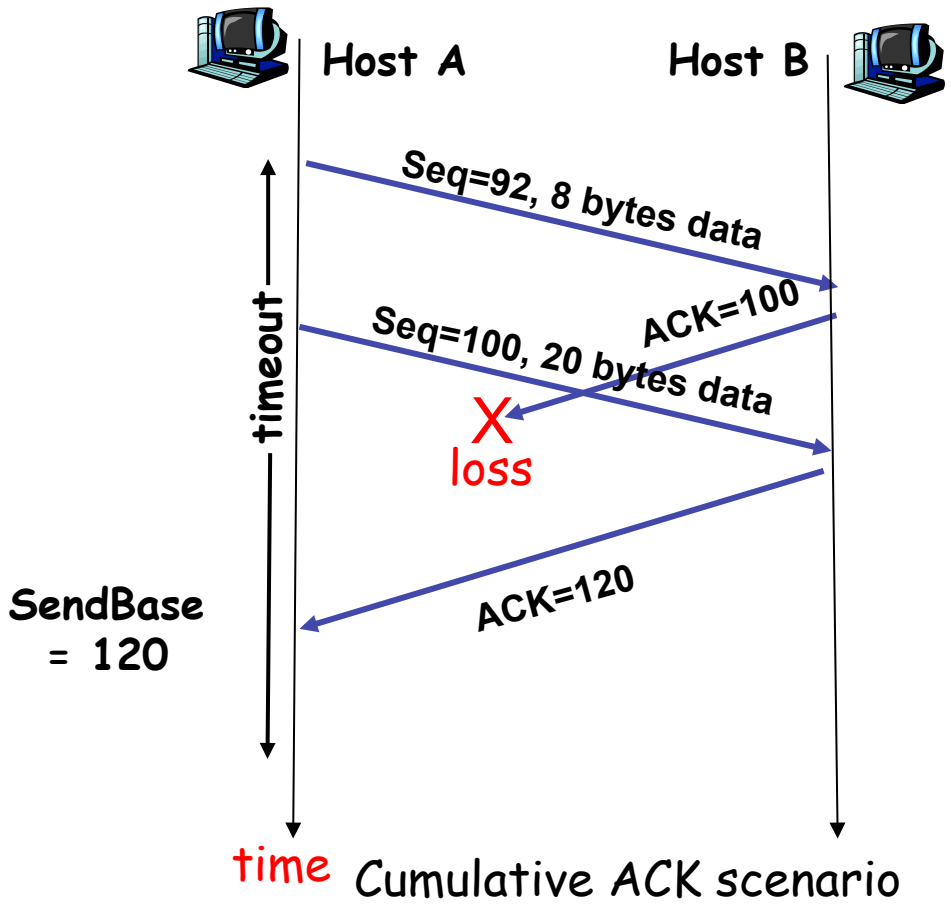
- Szabályok arra, hogy mit tegyünk, ha nem jön ACK a time-out alatt

Újraküldési esetek a TCP-nél: elveszett nyugta és korai timeout





Újraküldési esetek a TCP-nél: összevont nyugta



Gond összevont nyugtával

- Pl. 1000 bájt kerül elküldésre 10 szegmensben
  - Ha elveszik az első szegmens, a fogadó nem tudja visszajelezni, hogy 100-999 megjött, de 0-99 nem
  - Újra el kell küldeni a teljes 1000 bájtot

- Megoldás: **szelektív nyugtázás**
  - SACK-ban meg tudja mondani, hogy 100-999 megjött sikeresen
  - Opcionális fejrészmezőben
  - Népszerű, minden TCP implementációban

Gond sorrend keveredéssel

- Sorrendkeveredés miatt elveszett csomagnak hiszi a küldő ->újraküldés->forrás visszafogás

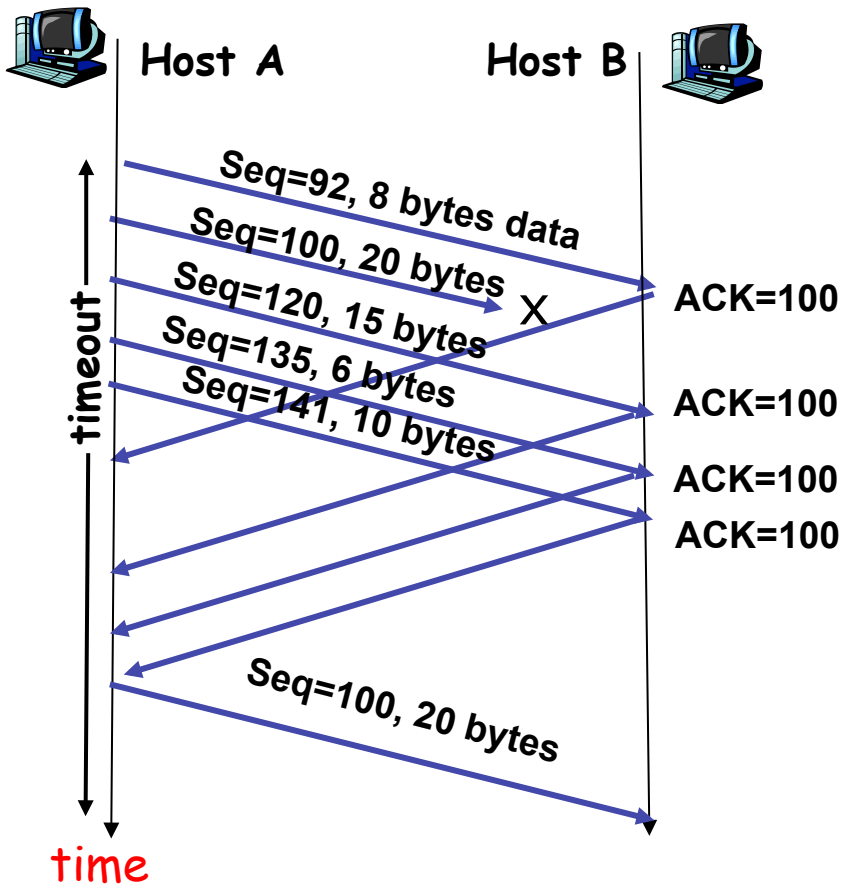
- Megoldás: **D-SACK (duplikált nyugta)**
  - Fogadó szól, hogy az újraküldött csomag duplikátum

  ->visszagyorsul a forrás

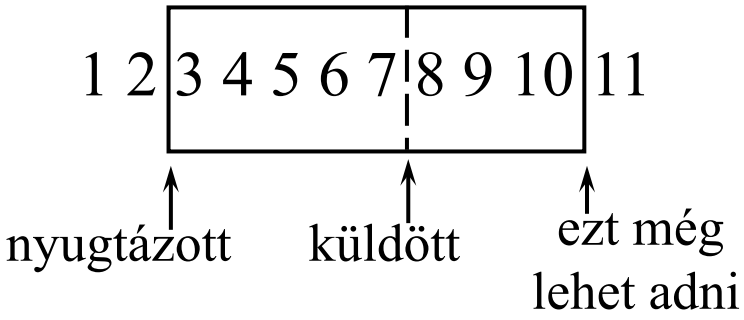
Fast retransmit (1)

- A time-out idő gyakran túl hosszú:
  - nagy késleltetés mielőtt az elveszett csomagot az adó újra tudná küldeni.
- De hogy értesülhetne az elveszett csomagról előbb, mintsem hogy letelt volna a timeout?
  - Az elveszett szegmensekre utalhatnak a duplikált ACK-ok
  - Ha a vevő hézagot vesz észre a vett szegmensek sorozatában (elveszett csomag) akkor újból lenyugtázza a megelőző helyesen vett szegmenst.
  - Több egymást követő duplikált ACK érkezhet.
- Fast retransmit szabály: ha az **adó 3 egymást követő ACK-t kap** (plusz az eredeti ACK) ugyanarra a szegmensre, feltételezi, hogy az azt követő szegmens elveszett és
  - **újraküldi azt még mielőtt lejárna a timeout**.

Fast retransmit (2)

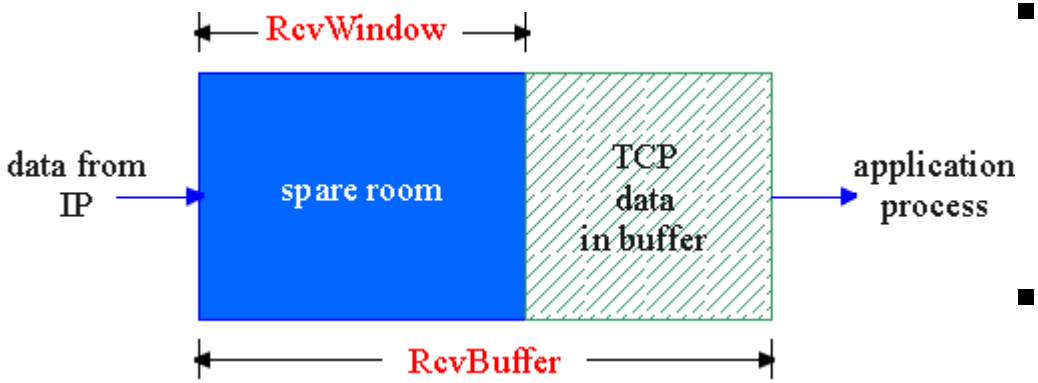


Flow control: a „sliding window” módszer elve



- Csúszóablakos („sliding window”) mechanizmus
  - az ablak mérete megadja a „kintlevő”, nyugtázatlan csomagok max. számát. (Pl.: w=8)
- A TCP-ben: az ablak-mechanizmus oktetteken működik

TCP Flow control: hogy működik?



- Spare room:

= **RcvWindow**

= **RcvBuffer-[LastByteRcvd - LastByteRead]**

- A vevő közli a szabad helyének a méretét (**RcvWindow**) a küldött szegmensben
- Az Adó legfeljebb **RcvWindow** mennyiségű nyugtázatlan adatot küld
- Nem szabad összekeverni a torlódási ablakkal (**CongWin**)

MSS (Maximum Segment Size)

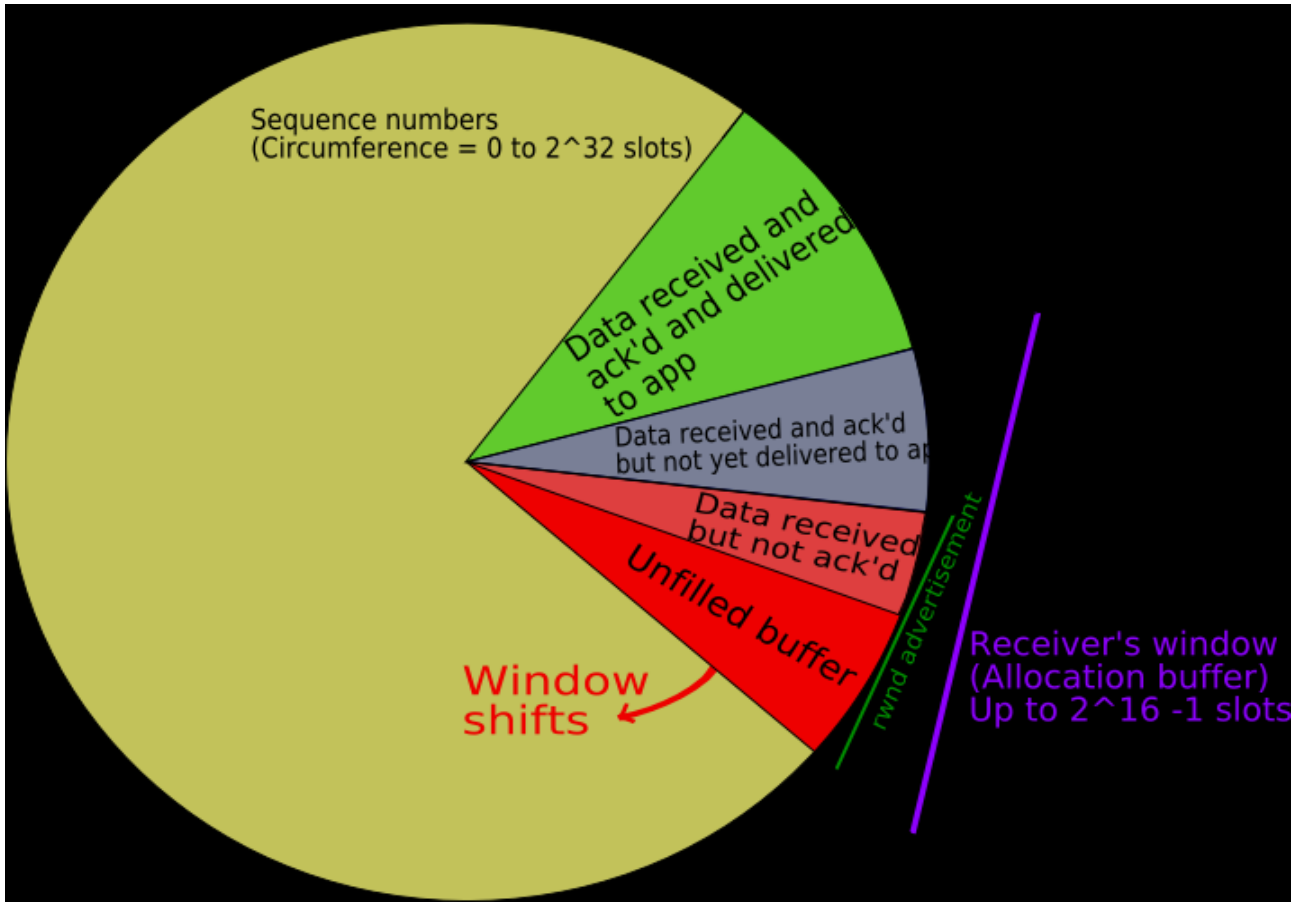
- A legnagyobb adatméret bájtban, amit a TCP hajlandó küldeni egy szegmensben
- Össze kell egyeztetni az adatkapcsolati réteg MTU-jával
  - elkerülendő a tördelést
- TCP kapcsolatfelépítésnél kell egyeztetni, **MSS opció a fejlécben**
- TCP adó használhat Path MTU discovery-t is: dinamikus MSS változtatás

Csúszóablakos problémák

I. Ha a vevő **nullás csúszóablak** méretet hirdet: adó leáll a küldéssel
- Ha **elveszik a vevő csomagja az új csúszóablak méretről**: adó vár hiába
- Megoldás: adó egy **timert** indít, lejárta után felkéri a vevőt, hogy küldjön ACK-ot az új ablakméretről

II. **Buta ablak jelenség**
- Ha a vevő oldalon kicsi (akár 1 byte) szabadul fel, lehet 1 byte az ablak
- A küldő 1 byte-ot küld, a vevő megint 1 byte-tal nyit
- Erőforráspocsékolás: kisebb az adat mint a fejléc!
- Megoldás:
  - A vevő nem nyitja az ablakot, csak akkor, **ha MSS nagyságrendűt nyithat**
- A küldő nem küld, hacsak nem
  - MSS-nyit küldhet
  - Mindent küldhet, amit az alkalmazás kért

A TCP „óra”


Sequence numbers
(Circumference = 0 to 2^32 slots)
Data received and ack'd and delivered to app
Data received and ack'd but not yet delivered to ap
Data received but not ack'd
Unfilled buffer
Window shifts
rwnd advertisement
Receiver's window
(Allocation buffer)
Up to 2^16 -1 slots

Torlódásvezérlés a TCP-ben

- A torlódásvezérlésről általában:
  - Azok a módszerek, amelyekkel linkek, csomópontok időszakos túlterheltségét megkíséreljük megszüntetni.

- Két fő módszer:
  - „Hálózat által segített” (**network assisted**) torlódásvezérlés
    - A hálózati elemek szolgáltatnak információt a túlterhelésről az adónak
    - Pl. TCP/IP ECN, ATM: külön jelzéscsomagok ehhez
  - Végpontok közötti (**end-to-end**) torlódásvezérlés
    - nincs visszacsatolás a hálózatból, a végpont következtet arra, hogy torlódás léphetett fel

- A TCP az utóbbit csinálja!

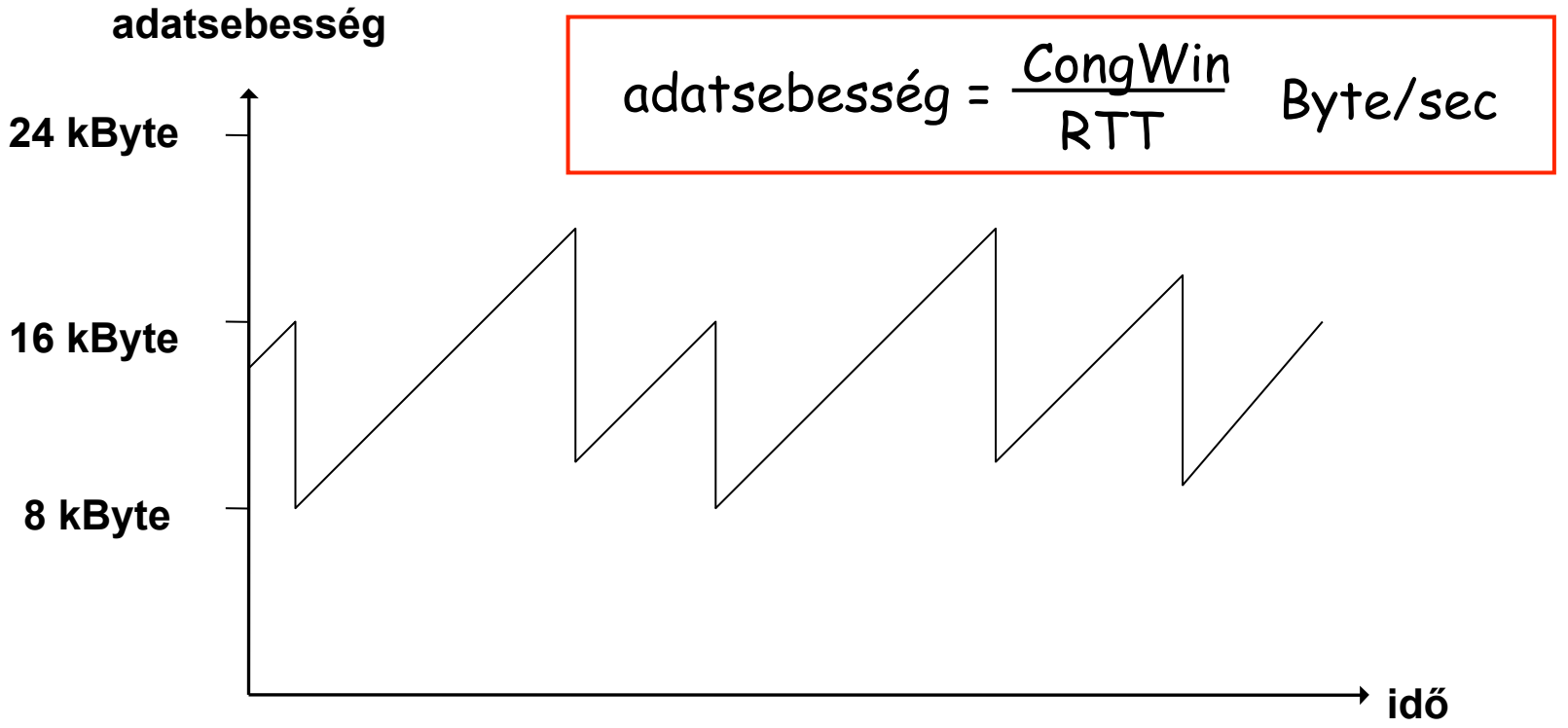
Torlódásvezérlés a TCP-ben

- Módszer:
  - **növeljük az adatsebességet**, ha úgy érzékeljük, van elég átbocsátóképesség
  - amíg **torlódásra utaló jeleket** nem tapasztalunk, ha igen, csökkentsük.

- Hogyan:
  - congestion window, **CongWin**
  - növeljük a CongWin-t minden RTT alatt MSS-sel, amíg vesztést nem érzékelünk
  - csökkentsük a CongWin-t felére minden vesztéskor
  - = **Additive increase – multiplicative decrease (AIMD)** módszer

- Hogyan érzékeljük a torlódást (vesztést)?
  - timeout letelt, nem jött nyugta
  - többszörös nyugta érkezett ugyanarra a szegmensre

Az adatsebesség alakulása AIMD esetén



$$\text{adatsebesség} = \frac{\text{CongWin}}{\text{RTT}} \quad \text{Byte/sec}$$

```
LastByteSent – LastByteAcked ≤ min {CongWin, RcvWin}
```

TCP torlódásvezérlés: további részletek

- Az AIMD kiegészítései:

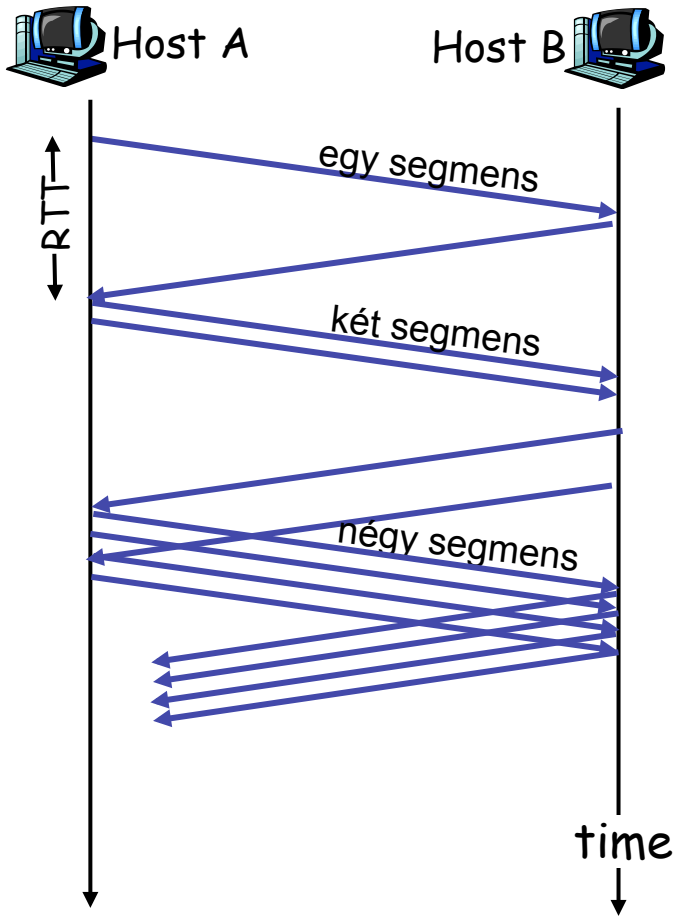
- **„Slow Start”**
  - az összeköttetés kezdetén a sebesség gyors (exponenciális) növelése az első vesztésig
  - utána AIMD

  - Gyors növelés: minden ACK után kétszerezzük

- Eltérő viselkedés timeout és többszörös (3-szoros) nyugták esetén

TCP Slow Start

- Az elején: **`CongWin`** = 1 MSS
  - Példa: MSS = 500 byte & RTT = 200 msec
  - kezdeti sebesség = 20 kbps
- Mivel a rendelkezésre álló átb. képesség >> CongWin/RTT lehet,
  - célszerű gyorsan elérni, ezért
  - induláskor exponenciálisan növelni az első vesztésig



Csomagvesztés

- 3 duplikált ACK után:
  - **`CongWin`** felére csökken
  - Utána lineáris növekedés
    - **torlódás elkerülés**
- De timer lejárta esetén:
  - **`CongWin`** 1 MSS lesz;
  - Utána exponenciálisan nő,
  - Egy korlátig, majd onnan lineáris

Filozófia:

- timer lejárta rosszabb torlódási helyzetet jelez, mint a 3 duplikált ACK

Összefoglaló: TCP torlódás szabályozás

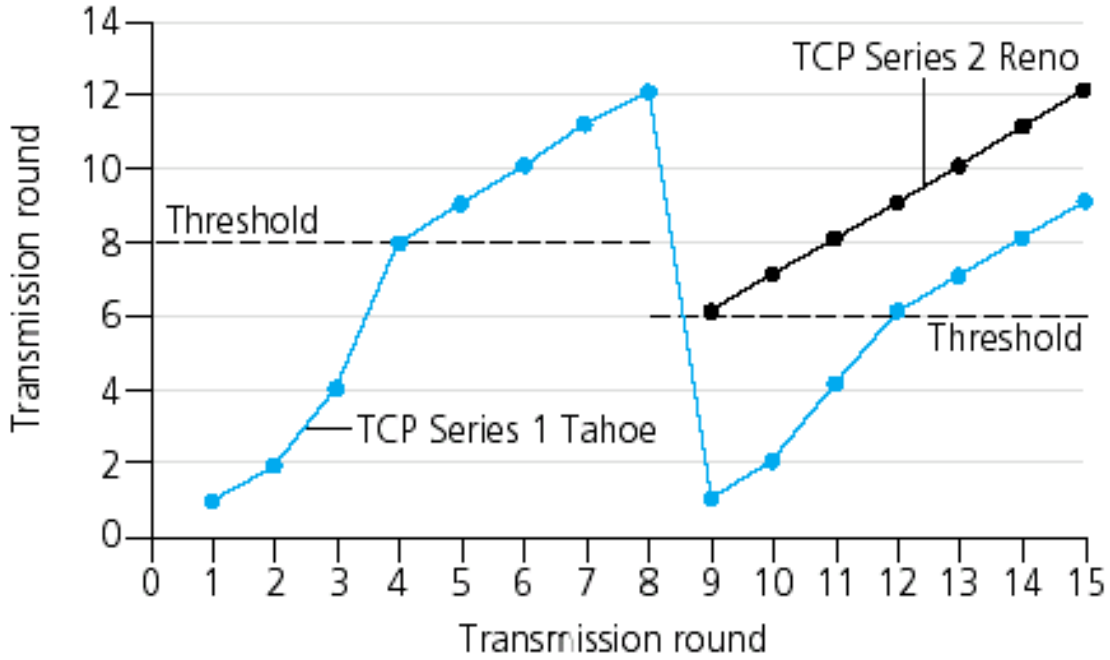
- Ha a **`CongWin`** egy korlát alatt van, adó slow-start fázisban, exponenciális ablak növelés
- Ha **`CongWin`** a korlát felett, az adó torlódás elkerülési fázisban, lineáris ablak növelés
- Ha 3 duplikált ACK, a korlát **`CongWin/2`** lesz és a **`CongWin`** pedig a korlát
- Ha timer lejárt, a korlát **`CongWin/2`** lesz, míg a **`CongWin`** 1 MSS lesz

TCP sender congestion control *

State	Event	TCP Sender Action	Commentary
Slow Start (SS)	ACK receipt for previously unacked data	CongWin = CongWin + MSS, If (CongWin > Threshold) set state to “Congestion Avoidance”	Resulting in a doubling of CongWin every RTT
Congestion Avoidance (CA)	ACK receipt for previously unacked data	CongWin = CongWin+MSS * (MSS/ CongWin)	Additive increase, resulting in increase of CongWin by 1 MSS every RTT
SS or CA	Loss event detected by triple duplicate ACK	Threshold = CongWin/2, CongWin = Threshold, Set state to “Congestion Avoidance”	Fast recovery, implementing multiplicative decrease. CongWin will not drop below 1 MSS.
SS or CA	Timeout	Threshold = CongWin/2, CongWin = 1 MSS, Set state to “Slow Start”	Enter slow start
SS or CA	Duplicate ACK	Increment duplicate ACK count for segment being acked	CongWin and Threshold not changed

TCP implementációk

- Korlát 8MSS
- 3 duplikált ACK a 9.átvitelnél:
  - **Tahoe**: nincs fast recovery, így 1MSS-ról indul újra (mint ha timer járt volna le), exp. a korlátig, onnan lineáris
  - **Renoe**: CongWin/2-ről lineáris



Vegas TCP

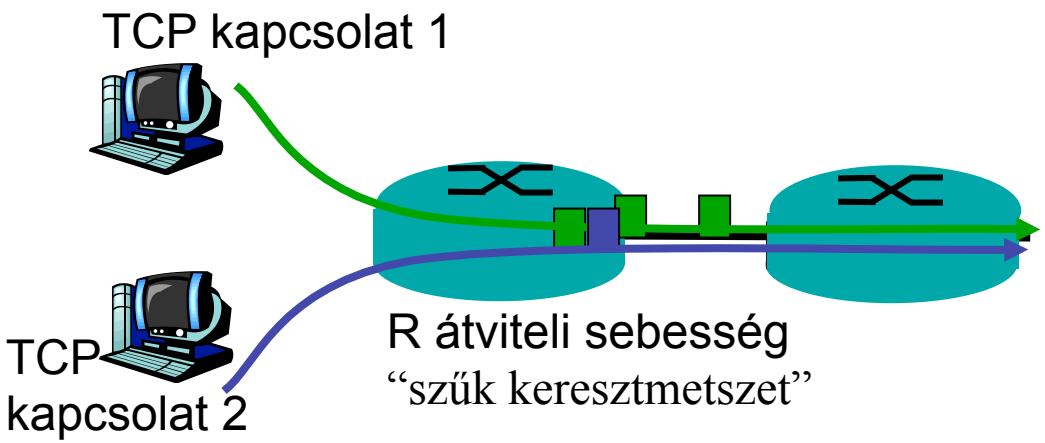
- megpróbálja **előre jelezni a torlódást**, mielőtt elveszne csomag
  - RTT alapján jósolja meg, előre csökkenti az ablakot ha kritikus az RTT
  - Pl. TCP Reno elnyomhatja, ha ugyanabban a hálózatban más csomópontok azt használják

TCP átbocsátóképessége

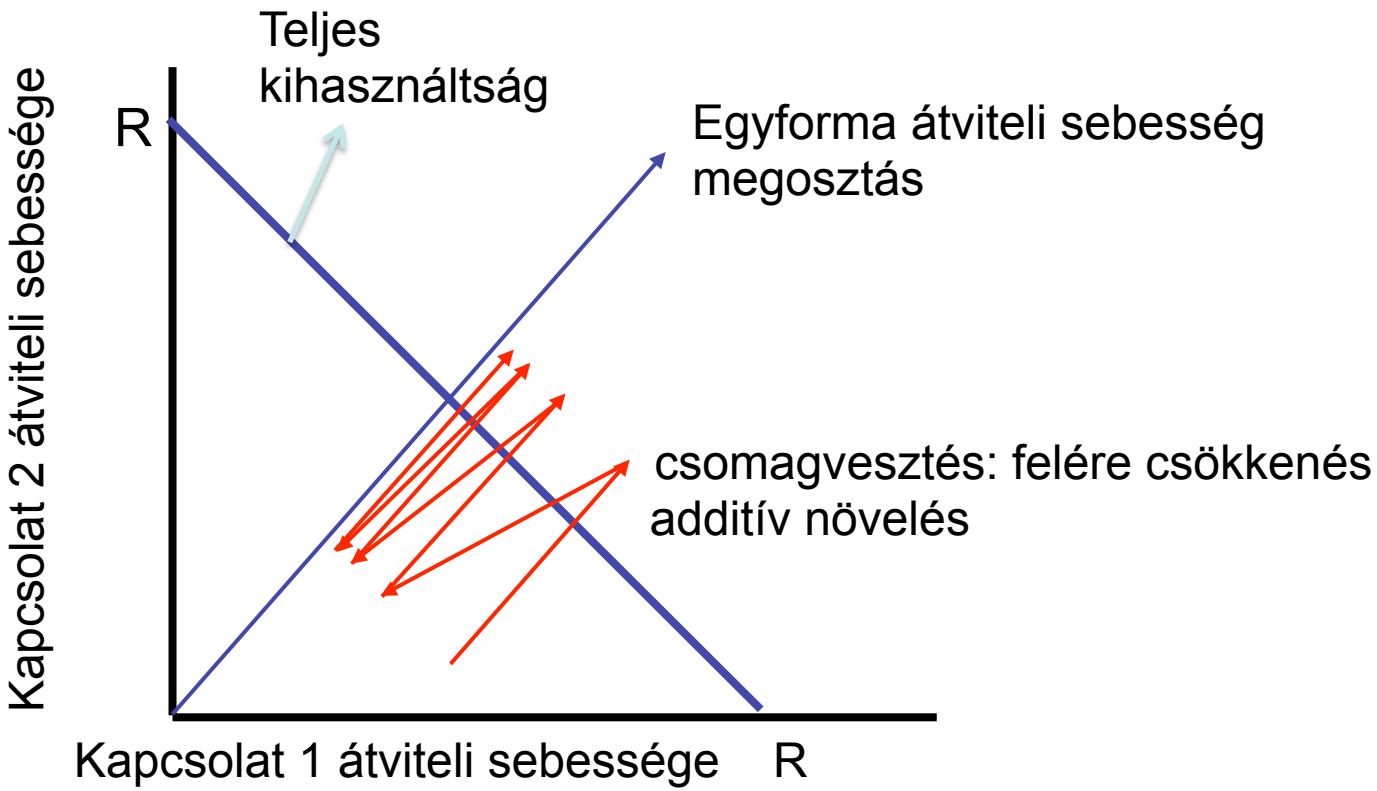
- TCP átlagos átbocsátó képessége az ablak méret és RTT függvényében?
  - Nem foglalkozunk slow-start-al
- *W* az ablakméret a csomagvesztés pillanatában
- Csomagvesztés előtt az átbocsátás: ***W/RTT***
- Vesztés után az ablak *W/2* lesz, átbocsátás: ***W/2RTT***.
- Átlagos átbocsátás: ***0.75 W/RTT***

TCP fairness

- Idealizált eset:
  - Mindkét kapcsolatnál egyforma MSS és RTT
  - Nincs más TCP vagy UDP kapcsolat
  - AIMD eset (nincs slow-start)



Két versenyző kapcsolat



Valóságban…

- Kisebb RTT-vel rendelkező kapcsolatok gyorsabban növelik a CongWin-t ⟶ nagyobb átviteli sebesség!

- UDP nem fair, kiszorítja a TCP-t

- Párhuzamos TCP kapcsolatok
  - Böngészők több párhuzamos TCP kapcsolatot építenek fel a webszerverhez
  - Minél több kapcsolat, annál nagyobb átviteli sebesség!
- Példa: R átv. seb., kliens-szerver, 9 kapcsolattal
  - Új alkalmazás kér 1 TCP kapcsolatot, sebessége: R/10
  - Új alkalmazás 11 TCP kapcsolat kér: több mint R/2!

TCP és UDP: összefoglalás

- Mindkettő host layer/transport layer protokoll
- Mindkettő portokat kezel
  - multiplexelés/demultiplexelés
  - ezáltal interface nyújtása az alkalmazói folyamatok felé
- Az UDP összeköttetés-mentes, best effort szolgáltatás
  - nem garantál célba juttatást, csak hibajelzést nyújt
  - gyorsan célba juttat
- A TCP összeköttetés-alapú, megbízható transzport szolgáltatás
  - sorrendhelyes, hibamentes szállítást nyújt
  - ára: késleltetés

Néhány gyakori alkalmazás és az használt transzportprotokollok

Alkalmazás	*Alkalmazási rétegbeli protokoll*	*Használt transzport-protokoll*
E-mail	SMTP	TCP
Távoli elérés	Telnet	TCP
Web-elérés	HTTP	TCP
File-átvitel	FTP	TCP
Routing	RIP	UDP
Hálózatmenedzsment	SNMP	UDP, TCP
VoIP, média-streaming	Többnyire nem szabványos	UDP

Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

MULTIMÉDIA TOVÁBBÍTÁSA AZ IP FELETT

1. rész
Bevezető áttekintés
Médiakezelő protokollok (RTP, RTCP, RTSP)

Dr. Szabó Csaba Attila
egy. tanár
BME Hálózati Rendszerek és Szolgáltatások Tanszék
szabo@hit.bme.hu

Multimédia alkalmazások

- *A multimédia alkalmazások fő csoportjai:*
  - Tárolt audió- és videó-streaming
  - Élő audió- és videó-streaming
  - Interaktív real-time audió és videó

- *Szolgáltatásminőség-igényeik:*
  - Késleltetés-érzékenyek (késleltetés és késleltetés-ingadozás)
  - Adatvesztésre általában nem érzékenyek
  - Éppen ellentétesen az adatkommunikációval

Médiastreaming fogalma

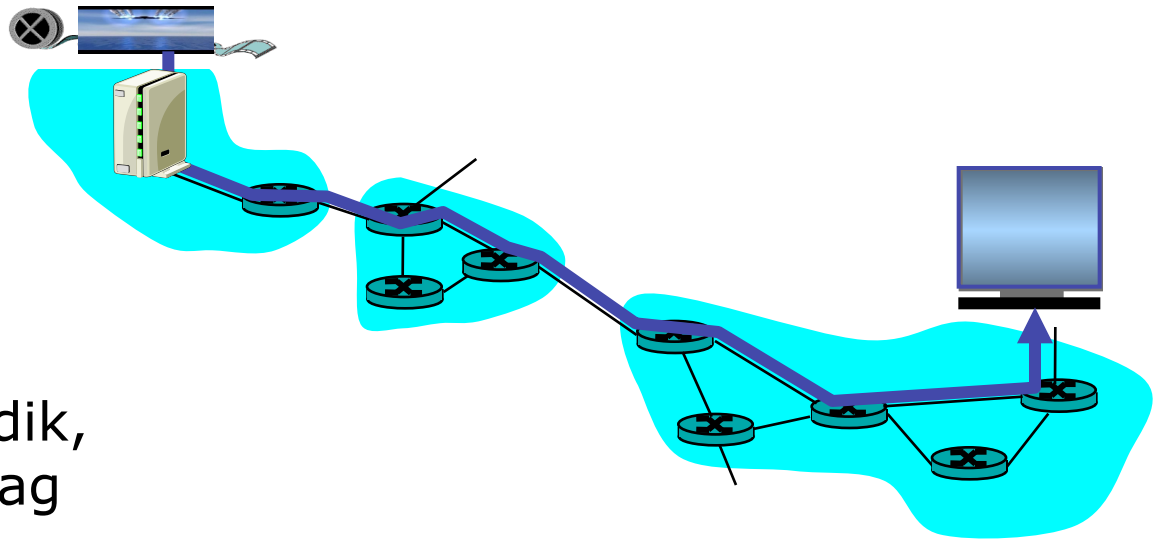
- „Azonnali” adatfolyam
  - általában tömörített multimédiás információ
  - azonnaliságra összpontosít, nem a multimédiás tartalom teljes hűségű visszaállítására
- Hagyományos média fájlok: egyetlen nagy csomagban letöltés az értelmezéshez
  - Ezzel szemben itt folyamatos csomagözön
- Elterjedéséhez kellett:
  - megnövekedett CPU és operációs rendszer képességek csomópontokban
  - magas adatátviteli ráták, különösen „utolsó mérföld”
  - hatékonyabb tömörítési algoritmusok

Tárolt médiastreaming

A tartalom a forrásnál tárolódik (médiaszerver)

Lejátszásra kerül a kliensnek

A lejátszás elkezdődik, mielőtt a teljes anyag megérkezik



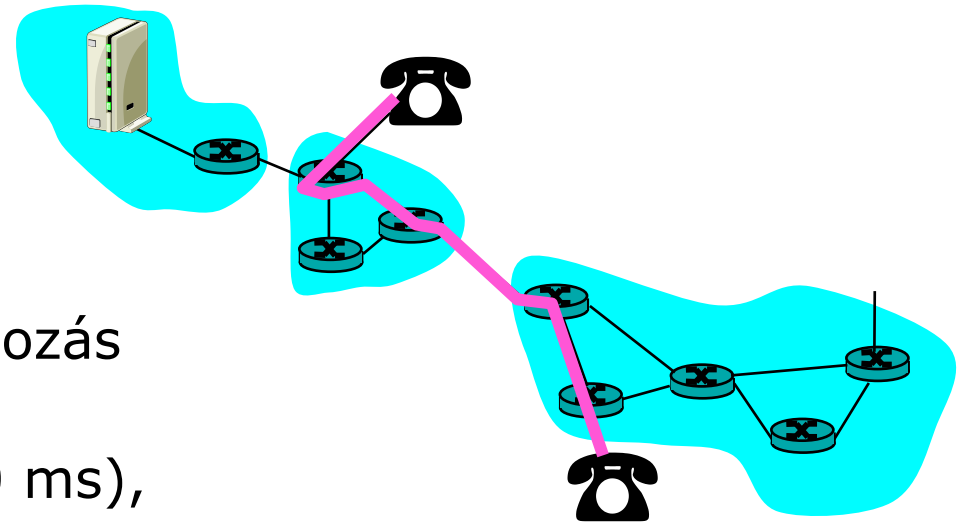
Interaktivitás: videomagnó-szerű funkciók szükségesek (előre-hátra, szünet, állj stb.)

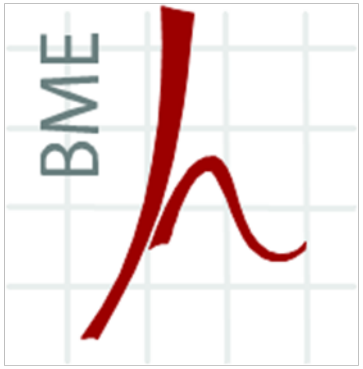
Élőmédia-streaming

- Alkalmazási példák:
  - rádió-, TV-állomások az interneten
  - élő előadások („webinar”)
  - élő sportesemények
- Streaming:
  - playback-tár
  - lejátszás késleltetéssel
- Interaktivitás:
  - néhány, a tárolt média lejátszásánál használt funkció (persze pl. gyors előre nincs)

Interaktív real-time audió és videó

- Alkalmazási példák:
  - VoIP, IP feletti telefonálás
  - videokonferencia
- Igények:
  - kis késleltetés,
  - kis késleltetésingadozás
  - pl. beszédnél:
    150 ms (max. 300 ms),
    ingadozás pár 10 ms
- Új funkció: hívásvezérlés





Példa: mit kell tudni a beszédről a VoIP-hez?

Beszédjel feldolgozása beszédcsomag-kommunikációs rendszerekben

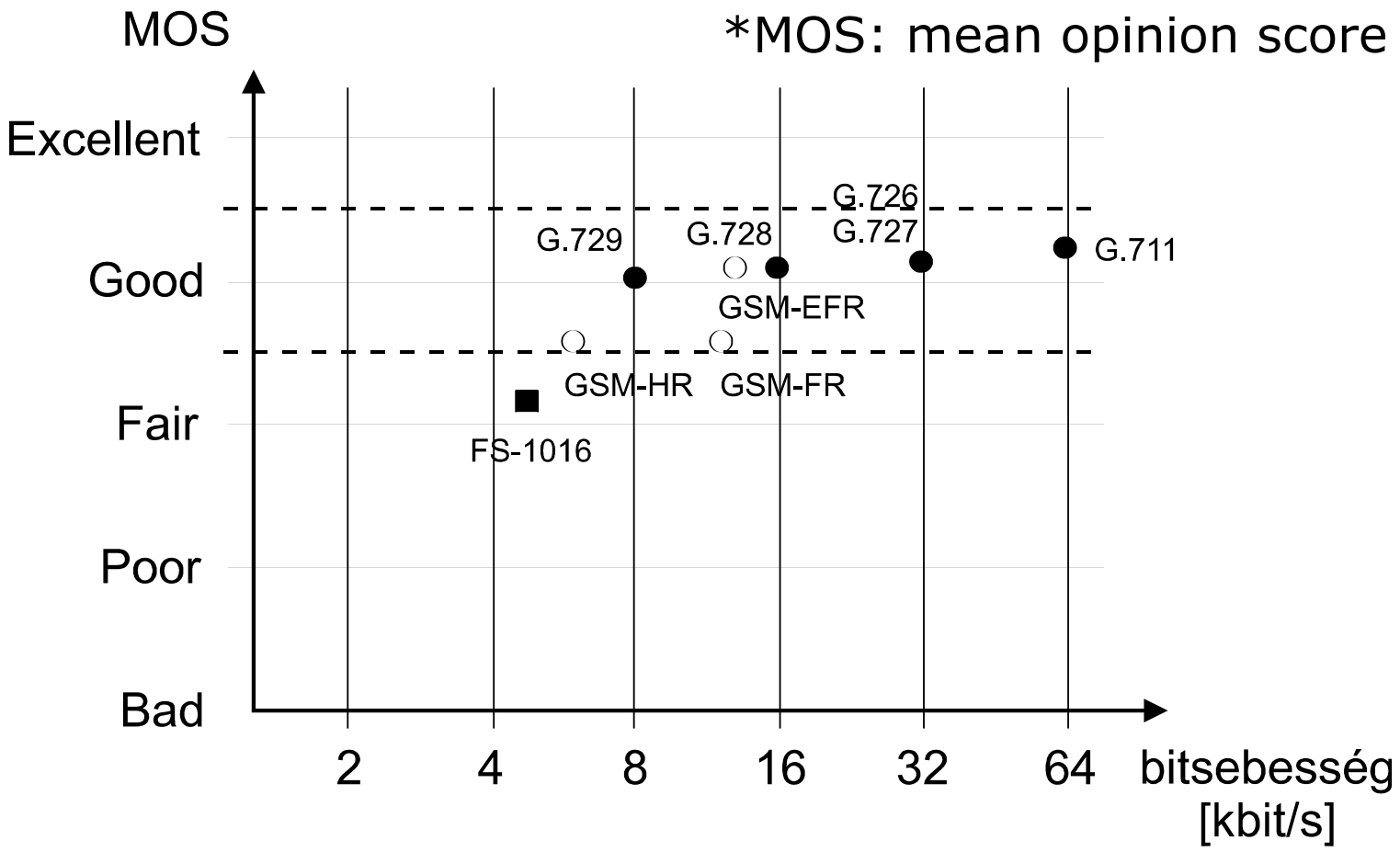
- Analóg-digitális átalakítás
  - A 64 kbit/s-os beszéd:
    - mintavétel 8 kHz-cel, azaz 125 mikroszekundumonként
    - a minták ábrázolása 8 biten (kerekítése 256 szintre)
    - tehát 8 bit 125 μs-nként
- Az inaktív szakaszok kivonása
  - Szünet/aktivitás detektálás (VAD - Voice Activity Detection)
- Redundanciák kivonása a beszédminta-folyamból, tömörítés
  - A kiiindulási 64 kbit/s forrássebesség csökkentése felére, negyedére,...
- Csomagokká alakítás
  - A beszédcsomag mérete függ a forrássebességtől és hogy milyen hosszú beszédszakaszt teszünk bele egy csomagba.

Néhány beszédtömörítési szabvány összehasonlítása

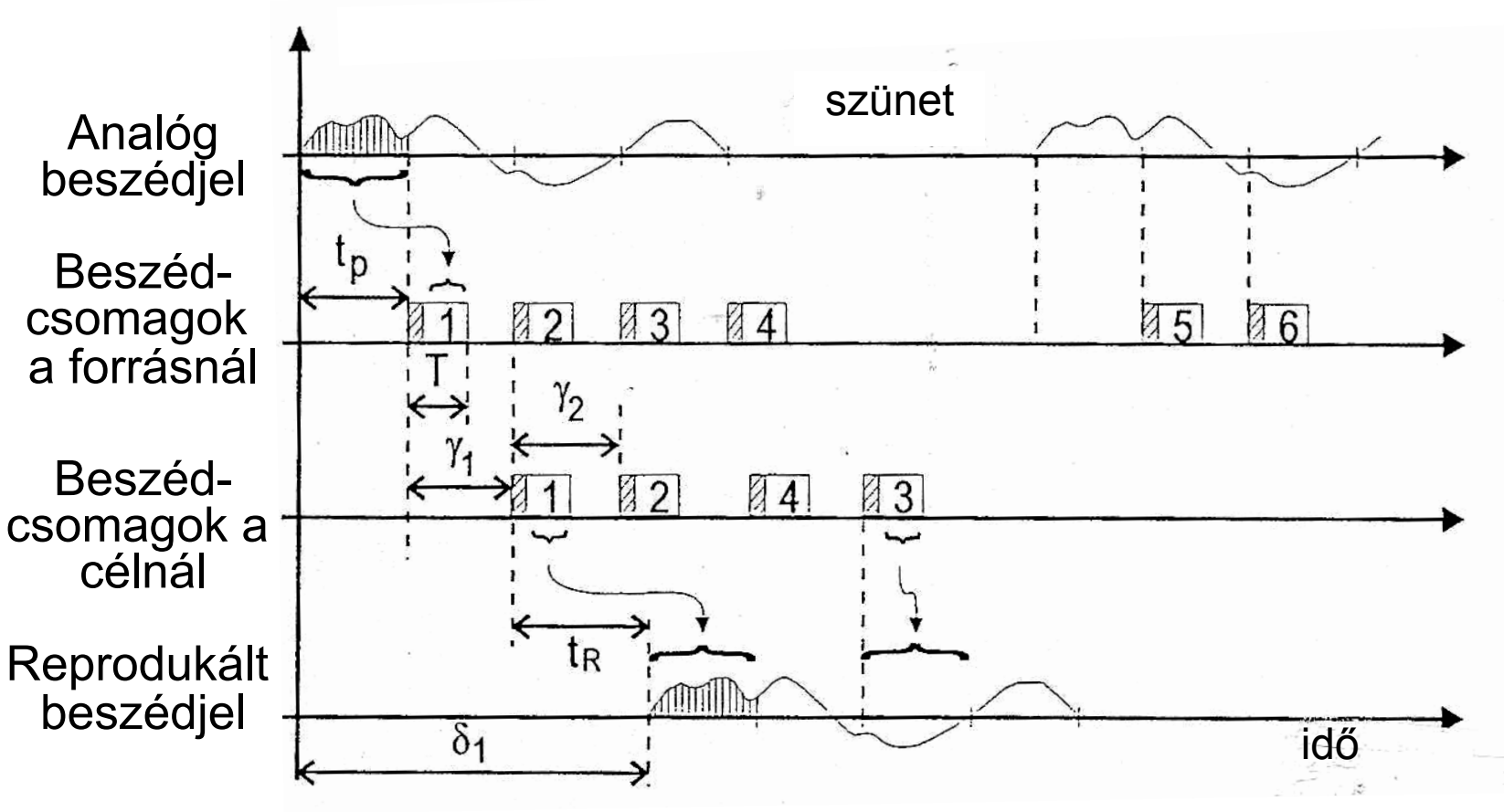
Szabvány*	Kódolási módszer	Bitsebesség [kbit/s]	Bonyolultság	Késleltetés [ms]
G.711	PCM	64	1	0,125
G.726	ADPCM	32	10	0,125
G.728	Low Delay-CELP	16	50	0,625
G.729	CS-ACELP	8	30	15
G.723.1	ACELP	6,3/5,3	25	37,5

* ITU-T szabványok

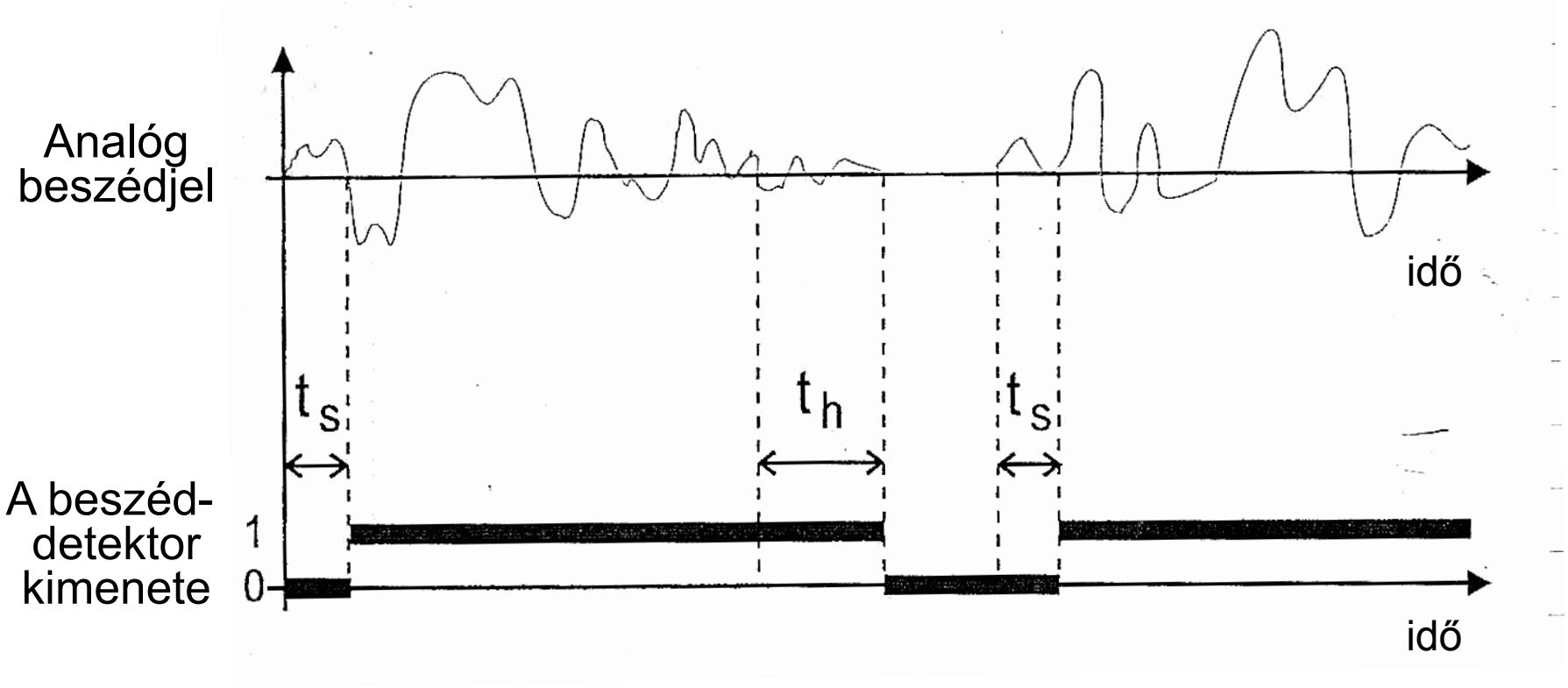
Néhány beszédtömörítés-szabvány szubjektív minősége (perceptual quality) - MOS



Beszédcsomagolás és -visszaállítás



Szünet/aktivitás-detektálás



Tipikus beszédcsomag-méretek

- Pontosabban: payload-méretek.

- Tipikus beszédszakasz, amelyet egy csomaggá alakítunk át: néhány ms-tól néhány 10 ms-ig.
  - A csomagolás fix kezdeti késleltetési komponenst okoz.
  - Ha a csomag elveszik (nincs pótlás): viszonylag rövid szakasz essen csak ki.
  - Gyakorlati értékek: 10 ms, 20 ms.

- 1. példa: 10 ms és tömörítés nélkül 80 byte.
- 2. példa: 20 ms és 2-szeres tömörítés esetén 80 byte.
- 3. példa: 6 ms és tömörítetlen beszéd esetén 48 byte.

Videó-tömörítési formátumok

- MPEG-1
  - Kb. VHS videó, CD audió minőség, 1,5 Mbit/s
- MPEG-2
  - DVD, digitális tv-adások, 3…10 Mbit/s (SD minőség), 10…20 Mbit/s (HD minőség)
- MPEG-4
  - Nagyobb tömörítés, jobb minőség az MPEG-2-hez képest
    - Tipikusan 1,5 Mbit/s (SDTV), 8 Mbit/s (HDTV)
  - MPEG-4 AVC (H.264), Blue-ray-kben is

Médiakommunikáció IP felett?

Mire van még szükségünk az eddig megismert protokollokon túl?

- IP:
  - „Best effort”-szolgáltatás, datagram-átvitel a hálózaton a végpontok között.
  - Nincs késleltetésgarancia.

- TCP vagy UDP:
  - Kommunikáció a végpontok alkalmazásai között.
  - UDP összeköttetésmentes, nincs lényeges plusz az IP-hez képest.
  - TCP összeköttetés-alapú, megbízható átvitel, késleltetés.
  - Eddig még a payload bármi lehetett, nem kezeltük, hogy milyen.

Mire van még szükségünk?

Feladatcsoportok és jellegzetes protokollok:

- Médiakezelés
  - RTP, RTCP, RTSP (1. rész)
- Hívásvezérlés
  - (H.323) és SIP (jövő félévben)
- Szolgáltatásminőség biztosítása
  - A DiffServ módszer (2. rész)

Médiakezelő protokollok - tartalom

- Bevezetés
- RTP – Real-time Transport Protocol
- RTCP – Real-Time Transport Control Protocol
- RTSP – Real-Time Streaming Protocol
- Összefoglalás

RTP és RTCP

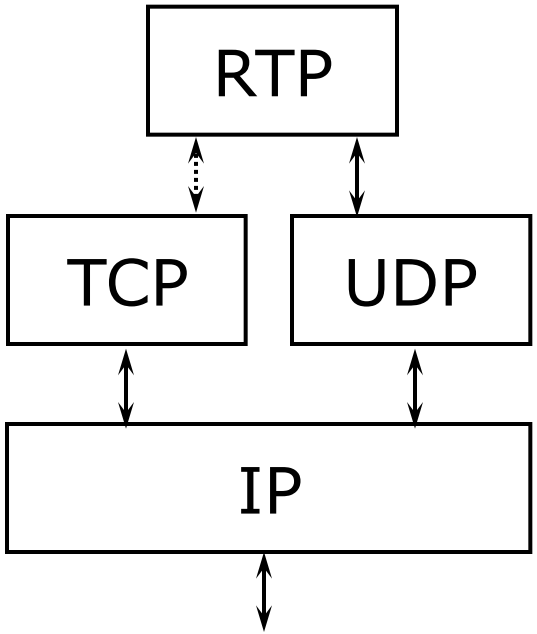
- Mi az RTP és mi nem:
  - A payloadjában média-, pl. beszédinformációt hordoz,
  - szállítási protokoll felett működik,
  - támogatja a médiafolyamok (streamek) szállítását,
  - de nem nyújt QoS-garanciát.

- RTCP:
  - A végpontok közötti QoS-**monitorozás** eszköze.

- Különböző UDP portokat használnak párban
  - RTP: páros számú portot
  - RTCP: a következő páratlant

- RTP, RTCP: az RFC-t 1889-ben specifikálták először (1996), majd felváltotta az RFC 3550 (2003)

Az RTP helye az IP protokollcsaládban

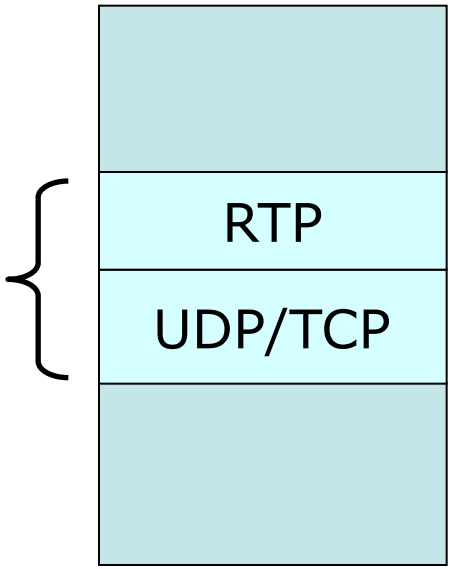


Rétegek

Alkalmazási

Szállítási
(Host-to-host)

Hálózati



* az RTP-t néha a szállítási rétegbe sorolják

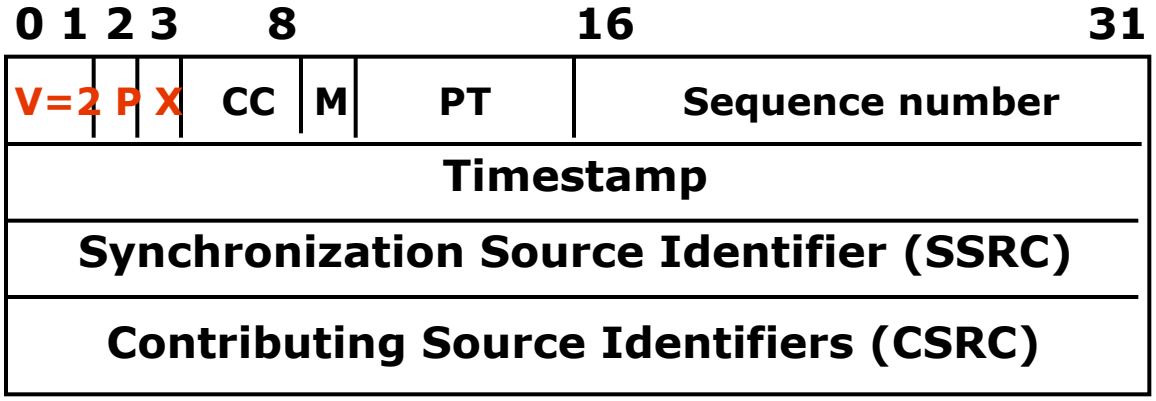
Az RTP szolgáltatásai

- **Payload-típusok (médiatípusok, tartalomtípusok) kezelése**
- **Sorszámozás**
- **Időbélyeg**

- RTCP:
  - QoS szolgáltatáshoz ad segítséget
  - A QoS paramétereknek végpontok közötti monitorozása
  - Médiastream-ek közötti szinkronizálás

- 5% RTCP forgalom, a többi RTP

RTP: csomagfejrész-formátum (1)



Version (V, 2 bits)

Az RTP verziószáma; az RFC 1889-ben definiált módon

Padding (P, 1 bit)

Ha 1-es: van padding az RTP csomag végén

Padding - utolsó byte: hányat kell figyelmen kívül hagyni

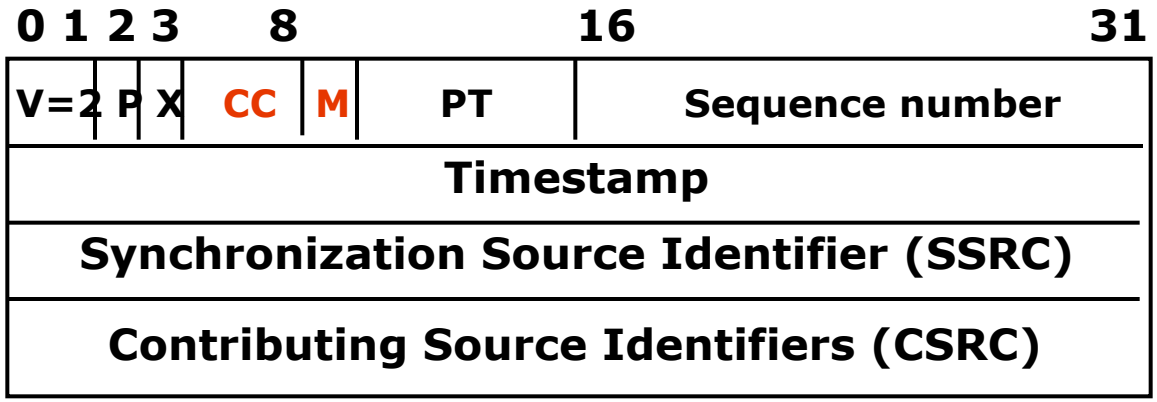
Extension (X, 1 bit)

Ha 1-es: a fejrész után változó hosszúságú fejrész kiterjesztés.

Ha van kiterjesztés: első 2 byte a hosszát adja meg

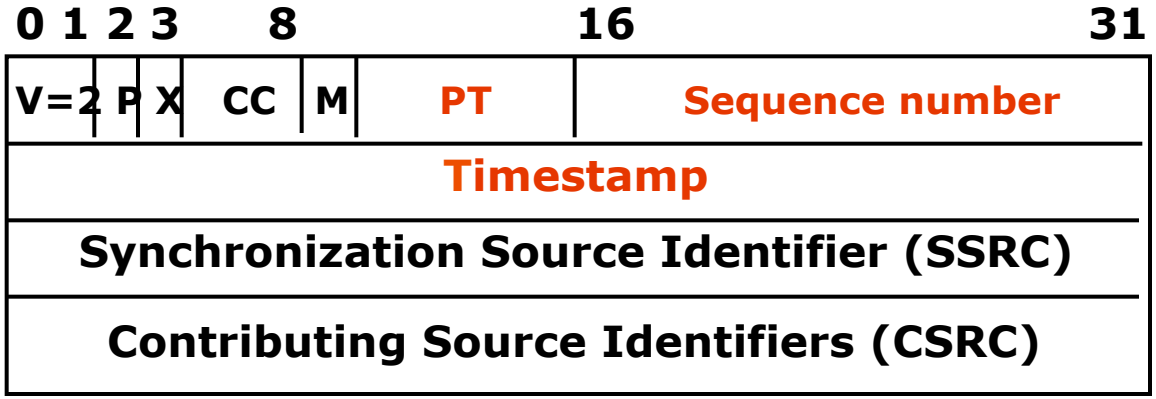
A kiterjesztés a fix fejrész utolsó érvényes mezője után következik

RTP: csomagfejrész-formátum (2)



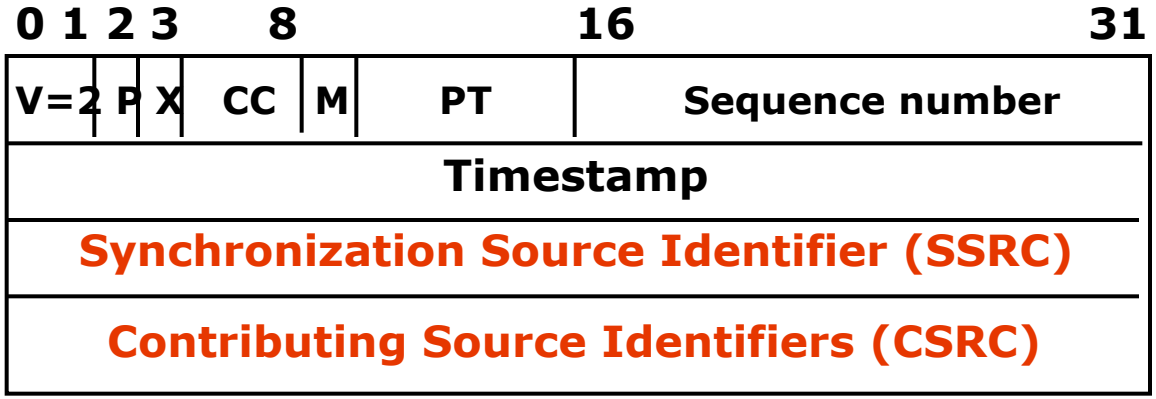
- CSRC count (CC, 4 bits)
  - a CSRC azonosítók száma = a multiplexált források száma (a források megadása: a CSRC mezőben)
  - csak egy forrás: CC = 0
- Marker (M, 1 bit)
  - a csomagfolyam szignifikáns eseményeinek megjelölése
  - példák:
    - kerethatárok a különféle kódolási módszereknél
    - a beszéd aktív időszakainak kezdete/vége
  - az aktuális interpretációt a “profile” adja meg

RTP: csomagfejrész-formátum (3)



- Payload type (PT, 7 bit)
  - “profile”, amely legtöbbször a médiakódolási típusokat kezeli, azoknak payload-formátumokat feleltet meg
- Sequence number (16 bit)
  - lehetővé teszi az elveszett csomagok detektálását és a csomagsorrend helyreállítását
  - kezdőértéke véletlen szám (l. később); minden elküldött RTP csomag után eggyel növelődik
- Timestamp (32 bit)
  - Az RTP csomag első oktettjének megfelelő pozíció valódi ideje a médiafolyamban

RTP: csomagfejrész-formátum (4)



- SSRC (32 bit)
  - Az RTP csomagfolyam forrását azonosítja, az RTCP rendeli hozzá, véletlenszerűen
- CSRC (0…15-ször 32 bit)
  - contributing source: az „RTP mixer” által létrehozott kombinált csomagfolyam komponensét azonosítja

RTP „profile”-ok

= A médiakódolást felelteti meg payload-formátumoknak
Példák beszédátvitelnél (továbbiakat specifikál az RFC):

Média kódolás	Mintavételi seb., kHz	Adatsebesség, kbit/s	RTP payload type
G.711	8	64	0
G.722	16	48-64	9
G.728	8	16	15
GSM	8	13	3
„Comfort noise”	-	-	19

Videó-payload-ok pl.: 26: JPEG, 33: MPEG2

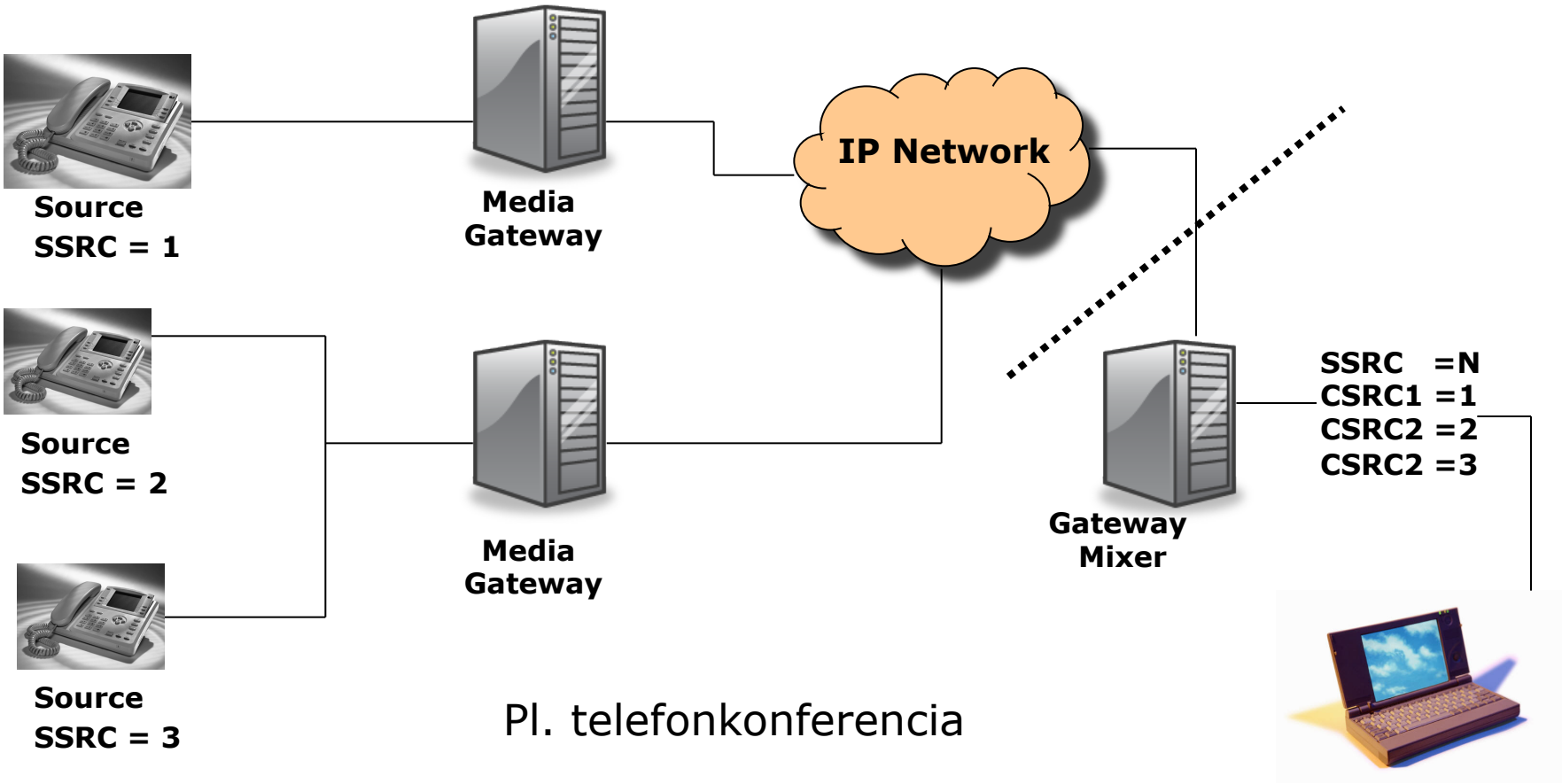
SSRC, Synchronization source

- A stream forrását azonosítja, **független a hálózati címtől**
  - Véletlen szám, egyedi kell hogy legyen a session-on belül
  - Pl. mikrofon, kamera
- Ha több stream-et generál ugyanazon forrás egy session-on belül: **mindegyiknek más SSCR**
  - Pl. videójel több kamerától
- A vevő ez alapján válogatja egy csoportba a csomagokat pl. playback esetén
- Változtathatja az adatformátumot időben

RTP mixer

- Közbülső rendszer, amely fogadja az RTP-csomagokat egy vagy több forrásból
  - megváltoztathatja az adatformátumot
  - kombinálja a csomagokat
  - új RTP-csomagként továbbítja
  - kombinált streamre új időzítés
- Mixer által összerakott új csomagok SSRC-je a mixer lesz
- Viszont a CSRC-ben felsorolja az eredeti forrásokat

RTP Mixer



Pl. telefonkonferencia

RTP csomag megjelenítése hálózatmonitoron

```
RTP Header

RTP: Version = 2
RTP: P Bit = 0 (Padding Does Not Exist)
RTP: X Bit = 0 (No Extension Header Follows)
RTP: CSRC Count = 0
RTP: Marker Bit = 0
RTP: Payload Type = MU-Law Scaling (PCMU) (0)
RTP: Sequence Number = 19382
RTP: Time Stamp = 7241.899 seconds
RTP: Synchronization Source Indentifier = 0x1C1A054A
RTP: 160 Bytes Of PCMU Payload Data

00 90 a0 00 00 73 00 90   a0 00 00 95 08 00 45 00  .....s.. ......E.
00 c8 00 37 00 00 78 11   a1 cb 0a 01 46 11 0a 01  ...7..x. ....F...
46 10 04 02 04 05 00 b4   00 00 80 00 4b b6 03 74  F....... ....K..t
05 58 1c 1a 05 4a ff ff   ff ff ff ff ff ff ff ff  .x...J.. ........
...
```

RTP-t használó alkalmazások

- *QuickTime (Apple)* – audio és video streaming
- *RealAudio, RealVideo* (RealNetworks) – média streaming
- *NetMeeting (Microsoft)* – IP telefónia, whiteboard, text chat, alkalmazásmegosztás
- *CU-SeeMe* – a NetMeeting-hez hasonló
- *IP/TV (Cisco)* – jó minőségű videó/audió, élő, VoD

RTCP – Real-time Transport Control Protocol

- Végpontok között információt szolgáltat a minőségről a kapcsolat résztvevőinek:
  - késleltetés
  - jitter
  - vett csomagok, elveszett csomagok stb.
  - RTT
- RTCP = nem jelzésátviteli protokoll!
- Hatására az alkalmazás kodeket válthat, vagy folyamot korlátozhat
- 3. fél is monitorozhat végig

Gondok az RTCP-vel

- Ha sok felhasználó és hosszú sessionok: ritkán jön RTCP jelentés (az 5% korlát miatt)
  - Pl. IPTV esetén, több perc vagy akár óra után csak, miközben 10 mp az elfogadható
- Elavult jelentések miatt a forrás rossz intézkedéseket hozhat a QoS szempontjából
- Megoldás: hierarchikus aggregáció
  - Több jelentés összevonása egy összefoglaló jelentésbe
  - Összevonás: node-ok hierarchiája, fastruktúra

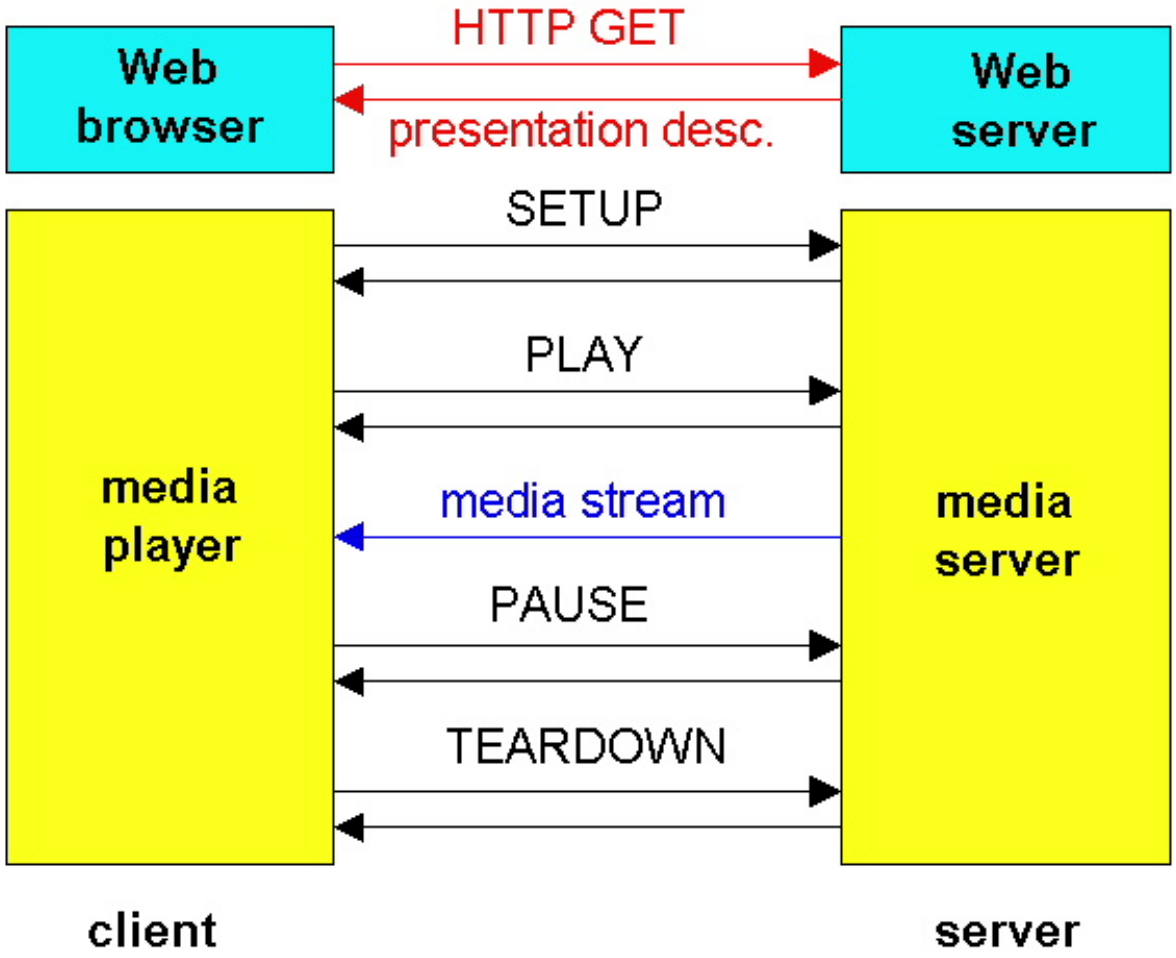
RTSP – Real Time Streaming Protocol

- Kapcsolat felépítése és ellenőrzése a session végpontok között

- VCR-jellegű funkciók, azaz lejátszásvezérlés:
  - elindítás tetszőleges pozícióból
  - előre, hátra, megállás, folytatás
  - különböző lejátszási sebesség kérése
  - különböző bitsebesség beállítása

- Az RTSP-szerverek nagy része RTP-t használ

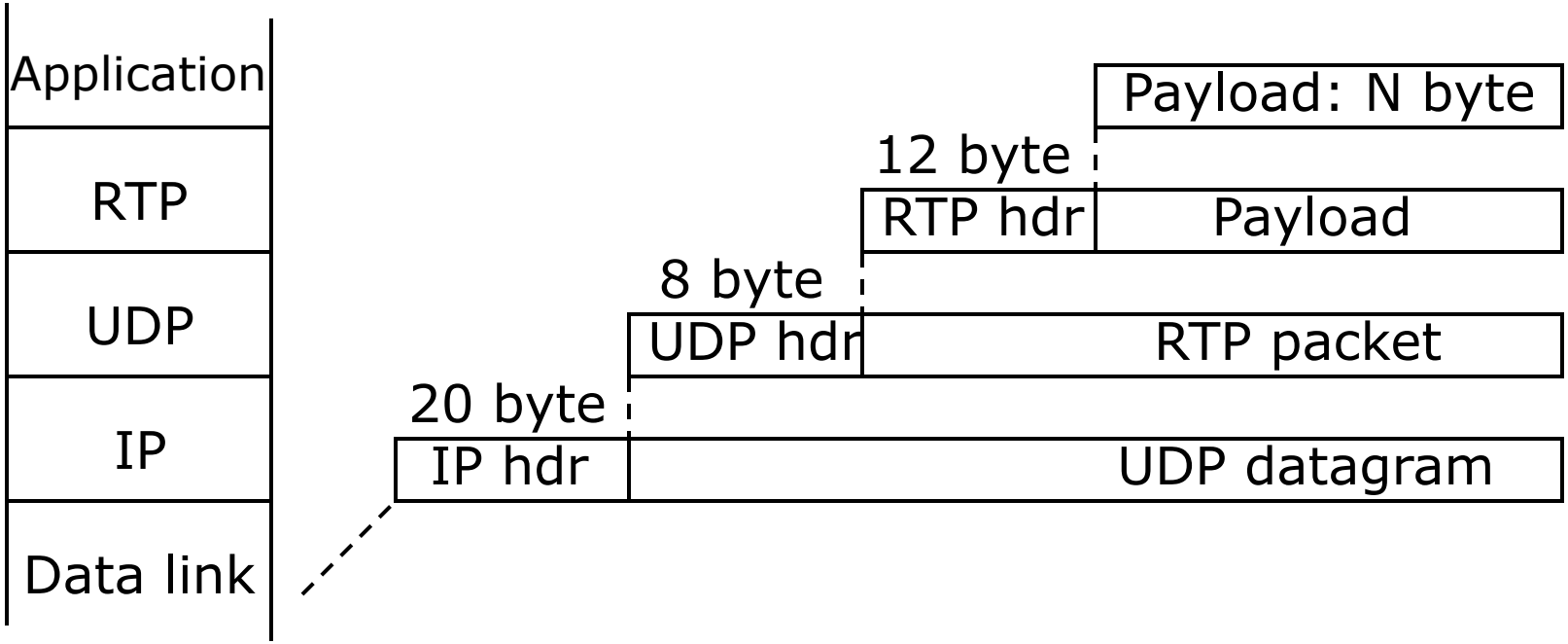
Az RTSP működése



RTSP

- Hasonló mint a HTTP, de ez állapotfüggő protokoll, több kérés típussal
  - kliens-médiaszerver kapcsolat
- Pl. kérésekre:
  - SETUP: specifikálja hogy kell átvinni az adott média stream-et (PLAY kérés előtt)
    - Portszámok RTP-re és RTCP-re
    - Szerver: jóváhagyja és elküldi saját portszámait
  - PLAY: egy vagy több médiastream lejátszása
    - Megadható a lejátszási intervallum is
- Mit nem csinál az RTSP?
  - Nem kezeli a médiaformátumokat
  - Nem nyújt semmilyen támogatást a szolgáltatásminőség megvalósításához

Most már túl sok protokollunk van ☺
Átvitelisebesség-igény a fejrész-halmozódás miatt?



Overhead = 40 byte plusz az adatkapcsolati réteg fejrésze
Egy médiacsomag hossza = 40+N+ link layer header

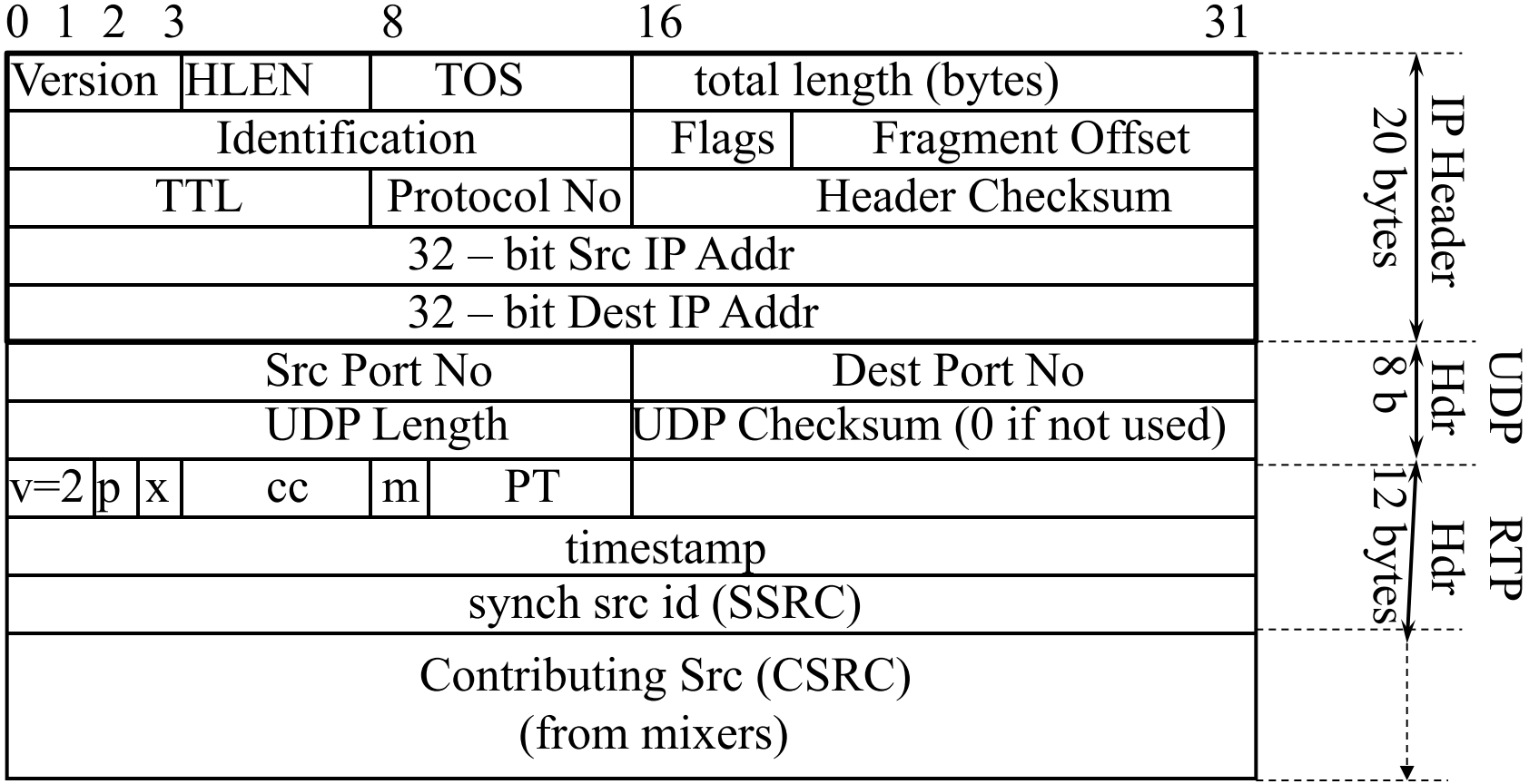
Fejrésztömörítés (1)

- Cél: az overhead csökkentése
- Kezeljük együtt mindhárom protokollt (IP + UDP + RTP)!
- Elv: a kapcsolat kezdete után kis változások a fejrészekben

1. Redundáns információ (pl. megvan az adatkapcsolati fejlécben)
2. Sok mezőben nincs változás a kapcsolat alatt
3. Vannak mezők, amelyek változnak, de jósolható módon, pl. sequence number

- Először tömörítetlen és teljes fejrész elküldése
- Utána a kompresszor és a dekompresszor megállapodnak egy Context Session ID-ben (CID);
  (8/16 bit hosszú) és a tömörítés formátumában
- CID: IP címek, UDP portok, RTP SSCR halmazát azonosítja

Fejrésztömörítés (2)
Nézzük meg az együttes fejrészt!



Fejrésztömörítés (2)

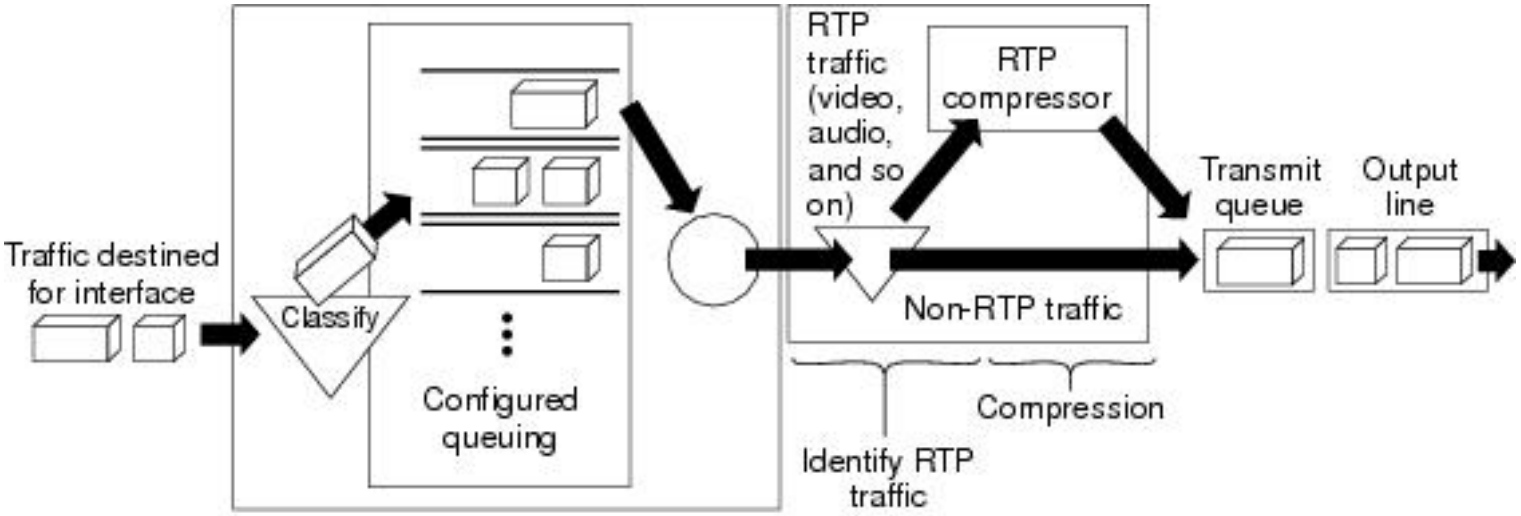
- Csak a következők változnak:
  - IP-nél:
    - Teljes hossz: ezt az adatkapcsolati fejrész is tartalmazza, nem kell átvinni
    - Header checksum: elhagyható, adatkapcsolatira bízva
    - Identification: inkrementálódik, IPv6-nál már nincs
  - UDP:
    - Ugyanaz az első kettő, mint IP-nél
  - RTP:
    - Seq. number: inkrementálódik
    - Időbélyeg: mintavételi időkkel növekszik
    - M bit
    - CSRC lista ritkán, csak akkor kell átvinni

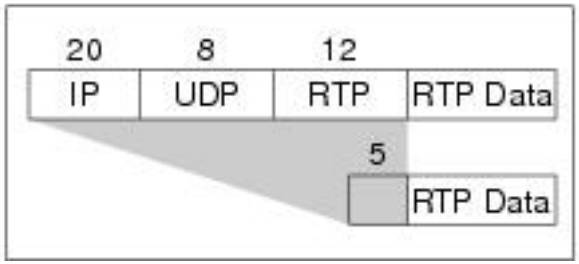
Fejrésztömörítés (4)

- Compressed RTP (cRTP) protokoll

- 4 formátum
  - FULL_HEADER: teljes tömörítetlen fejrészek plusz CID és Sequence No
    - Seq. No: csomagvesztés ellen kom. és dekom. között
  - COMPRESSED_UDP:
    IP + UDP tömörített (2 v 6 byte), RTP tömörítetlen
    - Ha változik az RTP payload típus
  - COMPRESSED_RTP:
    normál eset, minden fejrész tömörítve
    - Delta kódolás
  - COMPRESSED_NON_TCP:
    IPv4 ID mező tömörítetlen
    - védekezés nagy csomagvesztés esetén

Fejrésztömörítés (5)





Traffic type	Payload	Packet size reduction*
VoIP G.729	20 bytes	~ 58%
VoIP G.711	128 bytes	~ 21%

*Also ~5-ms reduction in serialization delay at 64 kbps

13473

A fejrésztömörítés alkalmazása

- Kis sebességű linkeken érdemes csak használni
- Ahol alacsony a hibaarány
  - Pl. vezeték nélküli környezetben nem
- Szolgáltatások:
  - Interaktív: videokonferencia, IP-telefónia

Voice over RTP adatátviteli-sebesség számolása (adatkapcsolati réteg nélkül)

Payload-formátum	Névleges sebesség	Csomagolási idő [ms]	Payload méret [byte]	Átvitelisebesség-igény, tömörítetlen fejrészek [kbit/s]	Átvitelisebesség-igény, tömörített fejrészek [kbit/s]
G.711	64 kbit/s	20	160	80	64.8
G.711		10	80	96	65.6
G.729	8 kbit/s	20	20	24	8.8
G.729		10	10	40	9.6

RTP/RTCP, RTSP: összefoglalás

- RTP - médiakezelő protokoll
  - UDP felett működik
  - Különböző médiakódolási formátumokat támogat
  - Az átvitelért felelős, a QoS-ért nem
  - A fejrészt (az UDP+IP-vel együtt) kompresszióval lehet csökkenteni
- RTCP az RTP társprotokollja
  - Nem jelzésprotokoll
  - Nem QoS protokoll
  - Hasznos segédeszköz a QoS megvalósításához
- RTSP: VCR-jellegű funkciók

Kérdések?

KÖSZÖNÖM A FIGYELMET!

QoS IP-hálózatokban: túl a Best Effort-on

DiffServ

Dr. Szabó Csaba Attila
egy. tanár
BME Hálózati Rendszerek és Szolgáltatások Tanszék
szabo@hit.bme.hu

Mi a QoS?

- QoS - Quality of Service, szolgáltatásminőség = végpontok közötti garanciák
  - adott időszakra és adott forgalom mellett, adott QoS-paraméterekre, pl.:
    - Rendelkezésreállás,
    - átviteli sebesség, „sávszélesség”,
    - késleltetés, késleltetésingadozás,
    - Csomagvesztés.
  - Pl. **beszédre**: legyen állandó 64 kbit/s a bitsebesség, a késleltetés ne legyen nagyobb, mint 100 ms, csomagvesztés kisebb, mint 1%.
- Milyen hálózatokban könnyű QoS-t biztosítani?
- A QoS biztosításának eszközei csomagkapcsolt hálózatokban:
  - Forgalmi méretezés
  - Protokollválasztás
  - Hálózati architektúra megválasztása, hálózati biztonság
  - Tárak menedzselése

A QoS biztosításának módszerei IP-hálózatokban

- „Nyers erő” (over-provisioning)
- Folyamonkénti (per-flow) QoS-biztosítás
  - IETF’s Integrated Services (**IntServ**) módszer
  - QoS az egyedi csomagfolyamokra: **„finom felbontású”** módszer
- Forgalomosztály-alapú (class-based) QoS-biztosítás
  - IETF’s Differentiated Services (DiffServ) módszer
  - QoS folyamosztályokra: **„durva felbontású”** módszer

Integrated Services (IntServ)

- Integrated Service Working Group
  - Szolgáltatás-osztályok specifikálása
  - az RSVP (Resource reSerVation Protocol) használata
  - Több RFC-ben specifikálták: 1633, 2211, 2212, 2215, 2216

- Alkalmazások és szolgáltatásosztályok:
  - **Guaranteed Quality** (RFC 2212):
    - Az ún. „Real-time intolerant (RTT)” alkalmazások számára
    - Garantált korlátokat nyújt bármely csomag **késleltetésére és a sávszélességre.**
  - **Controlled-Load** (RFC 2211):
    - Az ún. „Real-time tolerant (RTT)” alkalmazások számára
    - Törekszik **kb. ugyanolyan** szolgáltatást nyújtására, mint amilyet a folyam kapna, ha terheletlenek lennének a hálózati csomópontok.
    - Beengedés-szabályozást használ.
  - Eszköze: ütemezés!
  - **Best Effort**

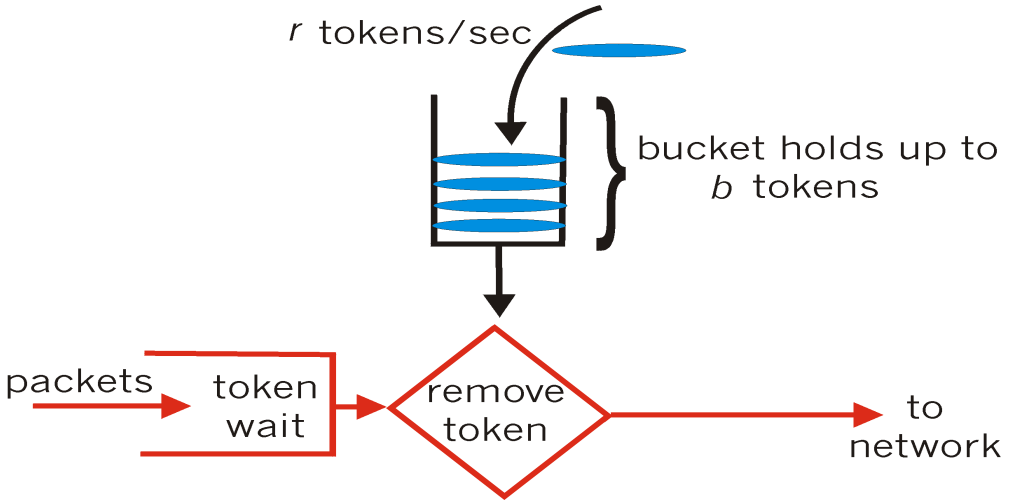
Az IntServ-ben használt mechanizmusok

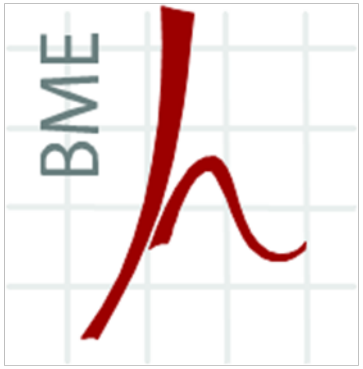
- A forgalom és a kért szolgáltatásminőség leírása
- Ennek alapján a hálózat **beengedés-szabályozást (***admission control)* alkalmaz
- Ha beengedi a folyamot, akkor **erőforrás-foglalás** történik **jelzésátviteli protokoll segítségével** *(RSVP – ReSource reserVation Protocol)*
- A forrás forgalmának **ellenőrzése és formálása** *(traffic policing)*
- A csomópontokban **ütemezés** (*scheduling)*

Forgalomleírás és policing

A policing és a forgalomleírás eszköze: a **token bucket** (tokenvödör)
A vödör formálja a forgalmat: küldhetünk *b* méretű börsztöt, de az átlagsebesség csak *r* lehet

- A vödörbe ***r* sebességgel** töltődnek a tokenek
- Legfeljebb ***b* token** lehet benne, ha már tele van, a beérkező tokenek elvesznek, túlcsordulnak
- Ha egy *n* hosszú csg érkezik, kivesz *n* tokent a vödörből (ha van annyi) és továbbításra kerül
- Ha kevesebb mint *n* token van: nem kerül kiszolgálásra





Differentiated Services (DiffServ)

A DiffServ alapötlete (1)

- A gerinchálózatra kell megoldást találni, ahol nagy volumenű, aggregált forgalmat viszünk át

- A DiffServ módszer nem egyedi folyamokhoz, hanem kisszámú **forgalomosztály**-hoz rendel erőforrásokat:
  - osztály, pl.:
    - *Premium*
    - *Regular*

- Ahelyett, hogy külön jelzésekkel (pl. RSVP-vel) értesítenénk a routereket egy folyam pl. *Premium* igényéről,

  egyetlen bit a csomagfejben megteheti ezt!

DiffServ alapötlete (2)

- Két kérdés az előző ötlethez:
  - Ki állítsa be a „premium” bitet, és mikor?
    - történjék a beállítás az ***„adminisztratív” határon***
  - Mit csinál a router másképpen, ha egy olyan csomagot kap, ahol ez a bit be van állítva?
    - különféle router-viselkedéseket ***követ***, meghatározta az *IETF DiffServ WG*
- Előbb nézzük meg, hogyan működik a DiffServ!

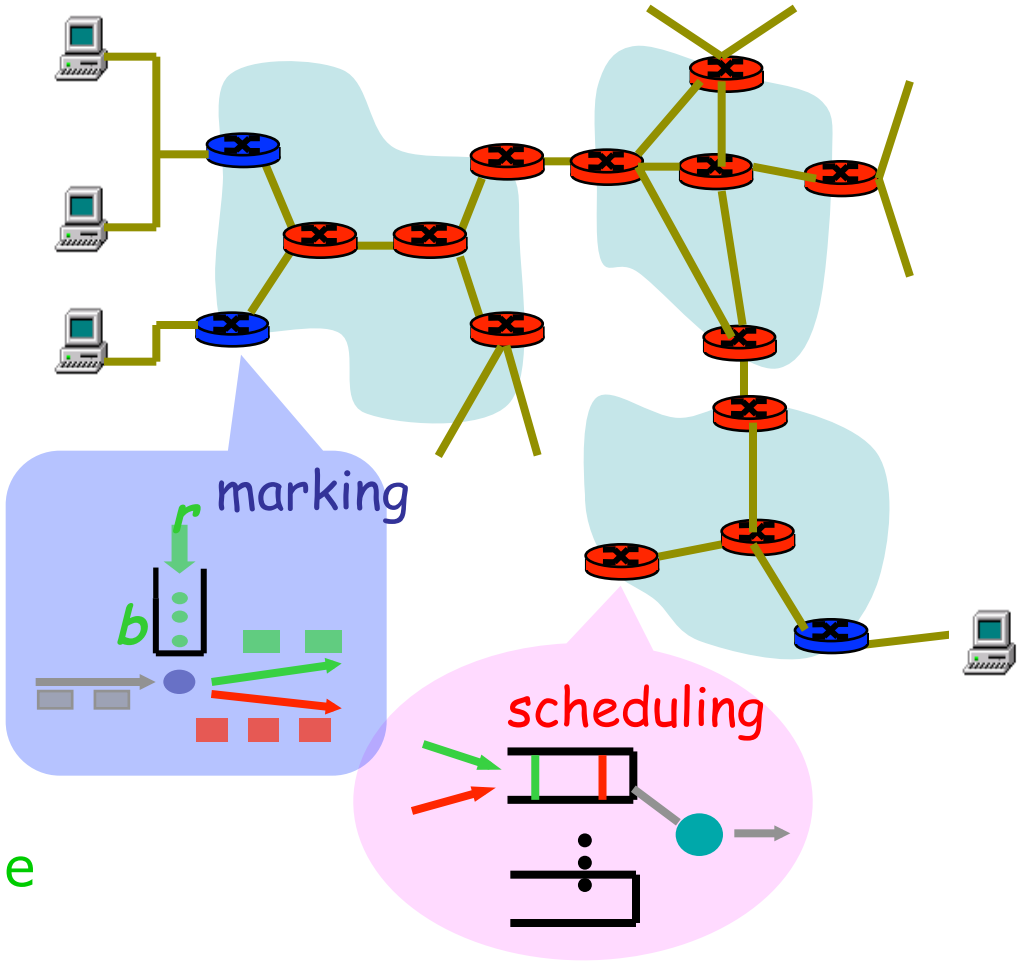
A DiffServ működése

Edge router:

- folyamonkénti forgalom-menedzselést végez
- megjelöli a csomagokat in-profile ill. out-profile -ként

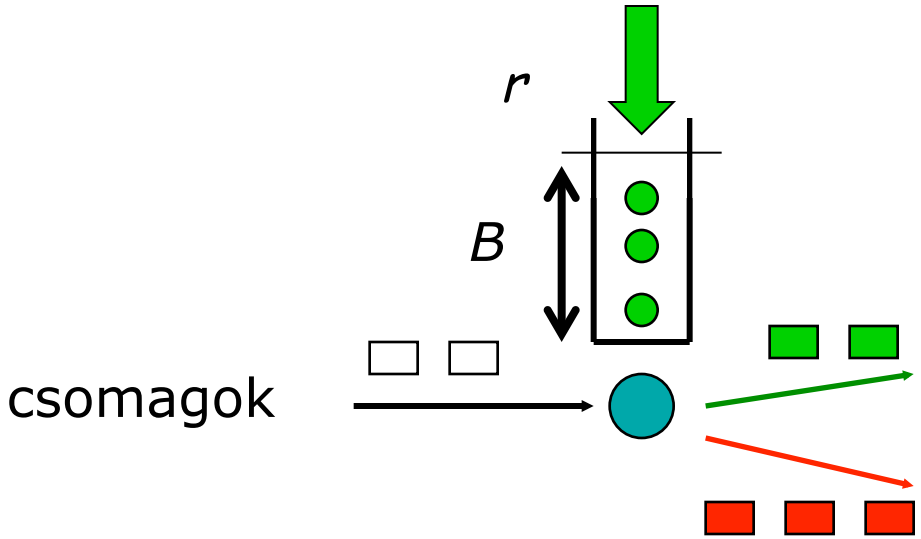
Core router:

- osztályonkénti forgalom-menedzselést végez
- pufferelés és ütemezés a széleken történt megjelölésnek megfelelően
- elsőbbség adása az in-profile csomagoknak



A csomagok megjelölése az edge-routerben

- profil: egyeztetett *r* sebesség és *B* vödörméret
- a csomagok megjelölése a széleken a folyamonkénti profil szerint

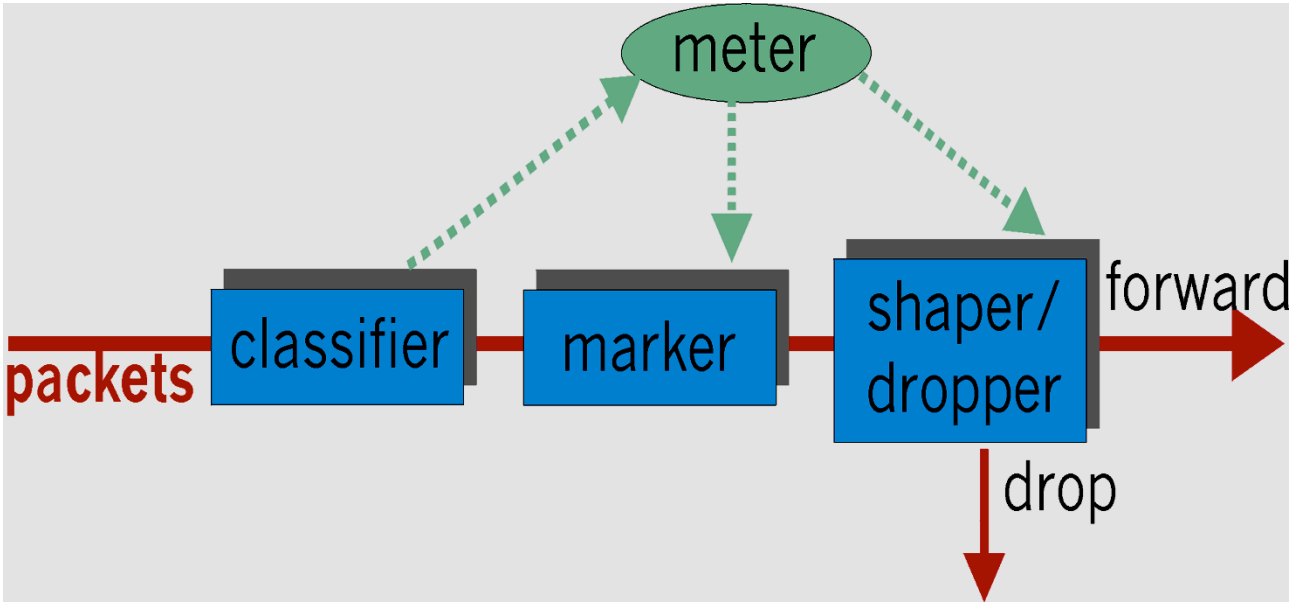


Megjelölés:

- osztály szerinti: a különböző osztályokhoz tartozó csomagokat eltérően jelöljük
- osztályon belüli: a folyam konform (a profilnak megfelelő) és nem konform részei eltérő jelölést kapnak

Mérés, osztályozás, megjelölés, és „kondícionálás”

- megfigyeljük a forgalmat (meter) és a nem konform csomagokat
  - formáljuk (shaping)
  - eldobjuk (dropping)

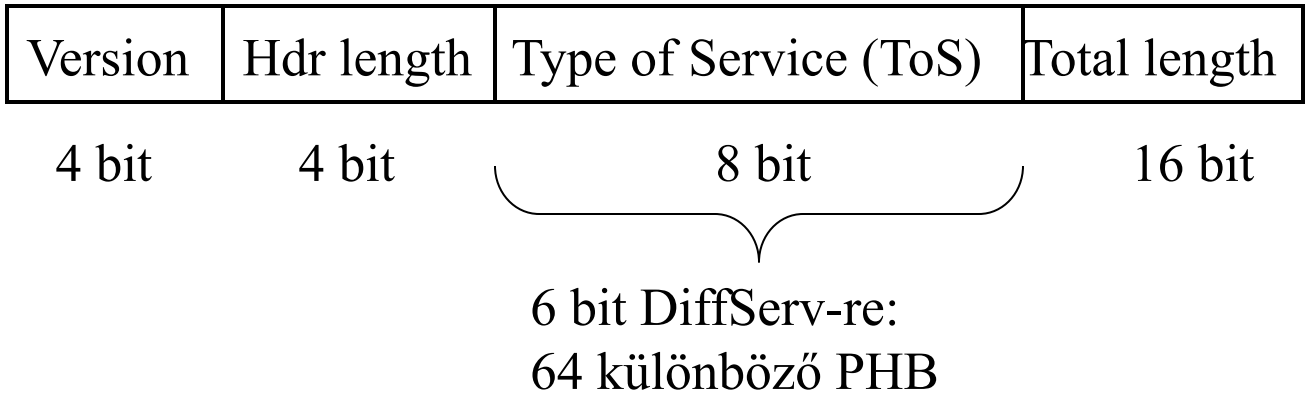


DiffServ – a csomópontok viselkedése

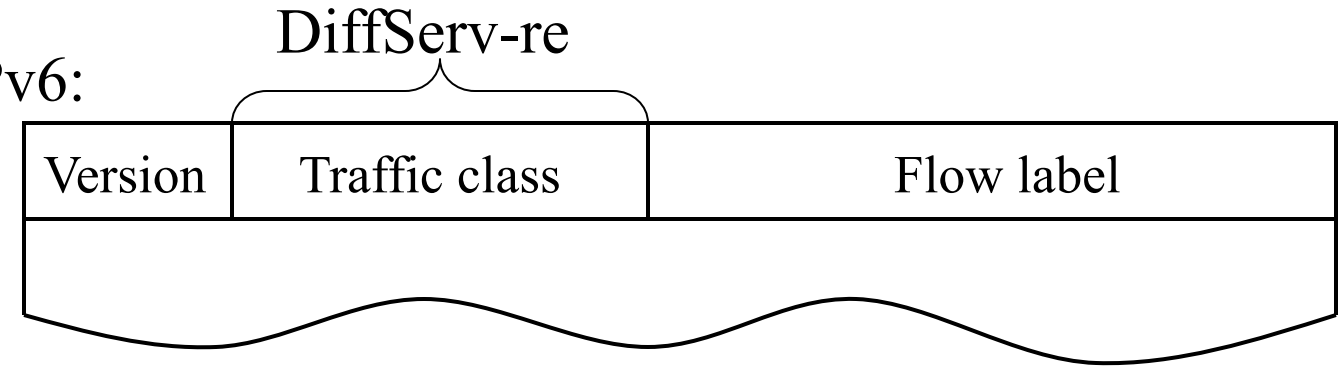
- *Természetesen nem csak 2 osztály van*
- ***Per-Hop-Behavior (PHB):*** *a forgalomosztályhoz tartozó csomagtovábbítási elveket definálja*
- Nincs együttműködés a végpontok között, mint az IntServ-nél
- A különböző PHB-k megadására:
  - IPv4-nél: IP fejrész ToS mezőjében 6 bit
  - IPv6-nál: ugyanaz Traffic Class néven
- Ez a 6 bit: “DiffServ Code Points” (DSCP)
  - A mező további 2 bitje: Explicit Congestion Notification (torlódásjelzés)
- Két alapvető PHB-típus:
  - **“*expedited forwarding*” (EF)**
  - **“*assured forwarding*” (AF)**
- Plusz default PHB: **best effort**
  - minden idekerül ami nem felel meg a fenti kettőnek

A “DiffServ Code Points”-ok (DSCP-k) elhelyezése

IPv4:



IPv6:



DiffServ – *expedited forwarding* (EF)

- Ez a legegyszerűbb PHB:
  - a csomagok továbbítása
    - minimális késleltetéssel
    - kis csomagvesztéssel történik
- Célszerű ha az EF-forgalom érkezési ütemét csak a csomópontok link-sebessége korlátozza
  - Ütemezőben: olyan sorhoz rendeljük hozzá, amelynek a kiszolgálási üteme legalább a csomagok beérkezési ütemével egyezik meg
    - Ezért elhanyagolható lesz a késleltetés és csomagvesztés
- Az EF („expressz” továbbítás) más típusú forgalommal szembeni szigorú prioritással garantálható
- Alkalmazások: beszéd, videó

DiffServ – *expedited forwarding* (EF)

- Gond az EF-fel: túl nagy EF forgalom **sorbanállási késleltetést okoz**
- Osztályon belül: késleltetés és jitter leromlása
- Szigorú beengedés-szabályozás kell erre az osztályra!
- A szolgáltató korlátozza, pl. max. 30%-a lehet a link kapacitásának

DiffServ – *assured forwarding* (AF)

- A legnépesebb osztály: 12 elemmel (**AF$_{xy}$**)
  - 4 csoport, az *x* (1 ÷ 4) sort választ
    - ezekben 3 – 3 elem, *y* (1 ÷ 3) „eldobási stílust” határoz meg

- Így az összesen 14 kiszolgálás közül a DSCP mező 6 bitje választja a megfelelőt:
  - - A*ssured Forwarding* DSCP: a köv. tábl. szerint
  - - *Expedited Forwarding* DSCP: 101 110
  - - *Best Effort (Default PHB):* 000000

- **Prioritásos sorok és eldobási jellemzők** valósítják meg a megfelelő kiszolgálást

Assured forwarding (AF) alosztályok

Drop Precedence	Class 1	Class 2	Class 3	Class 4
Low *Medium* *High*	010 000 … …	… … …	… … …	… … …

Osztályok közötti torlódás: az osztály prioritása dönti el

Osztályon belüli torlódás: a legkisebb eldobási valószínűségű élvez előnyt

Eldobási valószínűség szolgáltatónál: elég alacsony mindegyik eleinte, amelyik csomagok túllépik a „szerződött” átviteli rátát, azoknál növelik

Multimédia IP-hálózatokban, összefoglalás

- *Feladatcsoportok és*
  - *jellegzetes protokollok*
- Médiakezelés
  - RTP és RTCP
  - RTSP
- Szolgáltatásminőség-biztosítás
  - DiffServ

TCP/IP SOCKET INTERFACE PROGRAMOZÁSA

Számítógép-hálózatok (BMEVIHA215)

Dr. Lencse Gábor
tudományos főmunkatárs
BME Hálózati Rendszerek és Szolgáltatások Tanszék
lencse@hit.bme.hu

2014. április 24.,
Budapest

TCP/IP Socket Interface fogalma

- Socket: kommunikációs végpont
- Socket és fájl formális hasonlósága és különbsége
  - Kezelésük részben hasonló: megnyit, lezár, ír, olvas
  - Feladatuk más:
    - Fájl: adattárolás (helyben)
    - Socket: információcsere egyik végpontja
- Két megfelelő módon egymáshoz rendelt socket segítségével kétirányú kommunikáció valósítható meg
- TCP/IP socket interfész: a TCP/IP protocol stack programozási felülete
- Számos nyelvhez létezik: pl. C, C++, java, perl, stb.
- A C nyelvét fogjuk megismerni, elsősorban Linux alatt.

A legfontosabb lépések

- Socket létrehozása: **`socket`**
- Helyi IP-cím és portszám megadása: **`bind`**
- Kapcsolat orientált esetben
  - a kapcsolat létrehozása: **`listen, accept, connect`**
  - kommunikáció: **`write`** és **`read`** vagy **`send`** és **`recv`**
- Kapcsolatmentes esetben
  - kommunikáció: **`sendto`** és **`recvfrom`**
- Socket lezárása: **`close`**

Socket létrehozása

```
int socket(int domain, int type,
   int protocol);
```

- **`domain`**: protokollcsalád (lásd következő fólia)
- **`type`**: kommunikáció jellege, például:
  - **`SOCK_STREAM`**: megbízható kétirányú kapcsolat, megvalósítás: TCP-vel
  - **`SOCK_DGRAM`**: datagram szolgálat (kapcsolatmentes, nem megbízható), megvalósítás: UDP-vel
  - **`SOCK_RAW`**: hozzáférés a hálózatai szintű protokollhoz
- **`type`**: milyen protokoll valósítja meg, tipikusan csak 1 van
- **`return`**: socket descriptor: socketet azonosító egész szám (mint file descriptor), hiba esetén: -1

Példák protokollcsaládra

- **`AF_INET`**: IPv4 protokollcsalád
- **`AF_INET6`**: IPv6 protokollcsalád
- **`AF_UNIX, AF_LOCAL`**: helyi kommunikáció

Nevezéktan: *Protocol Family* más néven *Address Family*

- Régebbi BSD rendszerekben **`PF_`** előtagok
- Modern BSD és Linux rendszerekben **`AF_`** előtagok
- Érdekesség: **`/usr/include/bits/socket.h:`**

```
#define AF_INET PF_INET
```

Helyi IP-cím és portszám megadása

```
int bind(int sockfd,
   struct sockaddr *my_addr,
   socklen_t addrlen);
```

- **`sockfd`**: socket descriptor
- **`my_addr`**: mutató egy **`sockaddr`** típusú címstruktúrára, ami a beállítani kívánt helyi IP-címet és portszámot tartalmazza. Típusa **`AF_INET`** esetén **`sockaddr_in`**, **`AF_INET6`** esetén pedig **`sockaddr_in6`**. (Ld. köv. fóliák) A függvény hívásakor type cast: **`(struct sockaddr *)`**
- **`addrlen`**: megadja a címstruktúra méretét bájtokban
- **`return`**: hiba esetén -1, egyébként 0

IPv4: struct sockaddr_in

```
struct sockaddr_in {
  sa_family_t sin_family; /*address family: AF_INET*/
  u_int16_t  sin_port; /*port in network byte order*/
  struct in_addr sin_addr;  /* IPv4 address */
/* Pad to size of 'struct sockaddr'. */
};
/* IPv4 address: */
struct in_addr {
  u_int32_t s_addr; /*address in network byte order*/
};
```

Megjegyzés: hálózati bájtsorrend: MSB: Most Significant Bit first, big-endian. A legnagyobb helyi értékű bájt van elöl.

De a számítógép bájtsorrendje eltérő lehet! Vannak konverziós függvények, lásd később.

IPv6: struct sockaddr_in6

```
struct sockaddr_in6 {
  u_int16_t sin6_family; /* address family: AF_INET6 */
  u_int16_t sin6_port; /* port in network byte order */
  u_int32_t sin6_flowinfo;  /* IPv6 flow information */
  struct in6_addr sin6_addr; /* IPv6 address */
  u_int32_t sin6_scope_id;   /* Scope ID */
};

/* IPv6 address: */
struct in6_addr {
  unsigned char s6_addr[16]; /* address in netw. b.o.*/
};
```

Kapcsolat orientált eset

- A kapcsolat létrehozása során a szerver és a kliens szerepe aszimmetrikus
  - A szerver csatlakozásra vár, bárki csatlakozhat hozzá
    `listen`, **`accept`**
  - A kliens dönti el, hogy melyik szerverhez csatlakozik
    `connect`
- A létrehozott kapcsolaton keresztül a kommunikáció a „másik féllel” történik, nem kell megadni, hogy kivel
  `write` , **`read`** , (**`send`** , **`recv`** , sőt van még **`writev`** , **`readv`** is)

Szerver csatlakozásra kész

```
int listen(int sockfd, int
  backlog);
```

- **`sockfd`**: socket descriptor
- **`backlog`**: hagyományosan a létrehozás alatt álló kapcsolatok (TCP 3 utas kézfogás) számának felső korlátja volt. Ha ezt kimerítették, a szerver a további kapcsolatokat visszautasítja. A 2.2-es Linux kernelben ennek jelentését megváltoztatták, a már létrejött, de még **`accept()`**-tel el nem fogadott kapcsolatok maximális számát jelenti. (man 2 listen)
- **`return`**: hiba esetén -1, egyébként 0

Megjegyzés: ez egy nem blokkoló függvényhívás.

Szerver fogadja a kliens csatlakozását

```
int accept(int sockfd,
   struct sockaddr *addr,
   socklen_t *addrlen);
```

- **`sockfd`**: socket descriptor
- **`addr`**: mutató egy címstruktúrára. Ide kerül be a kapcsolódó fél IP-címe és portszáma.
- **`addrlen`**: mutató egy kétirányú paraméterre: híváskor megadja a rendelkezésre álló hely méretét, visszatéréskor pedig a kapott címadatok tényleges méretét bájtokban.
- **`return`**: siker esetén a létrejött kapcsolat eléréséhez a socket descriptor, hiba esetén -1

Megjegyzés: ez egy blokkoló függvényhívás.

Kliens csatlakozik a szerverhez

```
int connect(int sockfd, const
   struct sockaddr *serv_addr,
   socklen_t addrlen);
```

- **`sockfd`**: socket descriptor
- **`serv_addr`**: mutató egy **`sockaddr`** típusú címstruktúrára, ami a kívánt cél IP címet és portszámot tartalmazza. Ez **`AF_INET`** esetén **`sockaddr_in`** típusú, **`AF_INET6`** esetén pedig **`sockaddr_in6`**, mint a **`bind()`**-nál is láttuk.
- **`addrlen`**: megadja a címstruktúra méretét bájtokban
- **`return`**: ha sikerült a kapcsolódás, akkor 0, hiba esetén pedig -1

Megjegyzés: ez is blokkoló függvényhívás.

Adatok küldése kapcsolaton át

```
ssize_t write(int fd,
   const void *buf, size_t count);
```

- **`sockfd`**: socket descriptor
- **`buf`**: mutató arra a memóriaterületre, ahol az adatok találhatók. Ezek, úgy ahogy vannak, átvitelre kerülnek.
- **`count`**: az átvinni kívánt bájtok száma
- **`return`**: hiba esetén -1, egyébként az elküldött bájtok száma.

Megjegyzések:

- Azonos az (alacsony szintű) fájlba írással.
- A két félnek összhangban kell használnia a **`write()`** és a **`read()`** függvényeket!

Adatok fogadása kapcsolaton át

```
ssize_t read(int fd,
  void *buf, size_t count);
```

- **`sockfd`**: socket descriptor
- **`buf`**: mutató arra a memóriaterületre, ahova az adatok kerülnek.
- **`count`**: a fogadni kívánt bájtok száma
- **`return`**: hiba esetén -1, egyébként a fogadott bájtok száma.

Megjegyzések:

- Szintén az (alacsony szintű) fájlból való olvasás függvénye.
- A két félnek összhangban kell használnia a **`write()`** és a **`read()`** függvényeket!

Kapcsolatmentes eset

- Minden egyes küldésnél megmondjuk, hogy kinek szeretnék datagramot küldeni: **`sendto()`**
- Minden egyes vételnél megtudjuk, hogy kitől kaptuk a datagramot: **`recvfrom()`**

Adatok küldése kapcsolat nélkül

```
ssize_t sendto(int sockfd, const
  void *buf, size_t len, int flags,
  const struct sockaddr *dest_addr,
  socklen_t addrlen);
```

- **`sockfd`**: socket descriptor
- **`buf`**: mutató arra a memóriaterületre, ahol az adatok találhatók
- **`len`**: az átvinni kívánt bájtok száma
- **`flags`**: megadható számos opció (lásd: man 2 sendto), ha nem kívánunk élvi vele, írhatunk egyszerűen 0-t.
- **`dest_addr`**: a korábban megismert struktúra a cél megadására
- **`addrlen`**: a fenti struktúra mérete
- **`return`**: hiba esetén -1, egyébként az elküldött bájtok száma.

Adatok fogadása kapcsolat nélkül

```
ssize_t recvfrom(int sockfd,
  void *buf, size_t len, int flags,
  struct sockaddr *src_addr,
  socklen_t *addrlen);
```

- **`sockfd`**: socket descriptor
- **`buf`**: mutató arra a memóriaterületre, ahova az adatok kerülnek
- **`len`**: a fogadni kívánt bájtok száma
- **`flags`**: megadható számos opció, hasonlóan, mint **`sendto()`**
- **`dest_addr`**: a korábban megismert struktúra a forrás címadatainak fogadására
- **`addrlen`**: a fenti struktúra mérete – mindkét irányban!
- **`return`**: hiba esetén -1, egyébként a fogadott bájtok száma.

A kapcsolat bontása / socket lezárása

```
int close(int fd);
```

- **`sockfd`**: socket descriptor
- **`return`**: hiba esetén -1, egyébként a fogadott bájtok száma.

Mind a **`socket()`** függvényhívással megnyitott, mind az **`accept()`**-tel „kapott” socketeket ezzel a függvénnyel kell lezárni.

FONTOS, hogy a függvények visszatérési értékét mindig ellenőrizzük! Vannak a lehetséges hibák kiírására szolgáló könyvtári függvények…

Bájtsorrend konverziós függvények

- Alapvető 16 és 32 bites bájtsorrend konverziós függvények.

  Névkonvenció: <honnan>to<hova><hossz>: {h|n}to{n|h}{s|l}

  h=host, n=network; s=short, l=long (az adott helyen értve)

```
uint32_t htonl(uint32_t hostlong);
uint16_t htons(uint16_t hostshort);
uint32_t ntohl(uint32_t netlong);
uint16_t ntohs(uint16_t netshort);
```

  Bővebben: man 3 byteorder

IP-cím konverziók

```
in_addr_t inet_addr(const char *cp);
```

A **cp** stringben megadott IP-címet adja vissza binárisan, hálózati bájtsorrendben. Problémát okozhat, hogy a -1 értéket használja hiba jelzésére, de ez megegyezik az érvényes 255.255.255.225 érték ábrázolásával. Helyette:

```
int inet_aton(const char *cp, struct in_addr
   *inp);
```

Az eredménynek külön van helye (inp), és az érvénytelen inputot a 0 visszatérési értékkel jelzi. A másik irány:

```
char *inet_ntoa(struct in_addr in);
```

Bináris, hálózati bájtsorrendben megadott IP-címet szövegesen ad vissza. FIGYELEM: statikus puffert használ, tehát ki kell másolni, mert legközelebb felülírja!

Ha valaki programozni szeretne…

`int gethostname(char *name, size_t len);`

A saját gépünk nevének lekérdezése a **`name`** stringbe, ami **`len`** hosszú.

Példaprogramokban előfordulnak:

`gethostbyname(); gethostbyaddr();`

De ne használjuk, mert elavultak (obsolete)!

Helyette:

`getaddrinfo(); getnameinfo();`

Kicsit bonyolultak, de ott a man. ☺
Cserében reentránsak, IPv4-hez és IPv6-hoz is használhatók.

Példaprogram: http://whale.hit.bmehu/socket

HÁLÓZATI ALKALMAZÁSOK

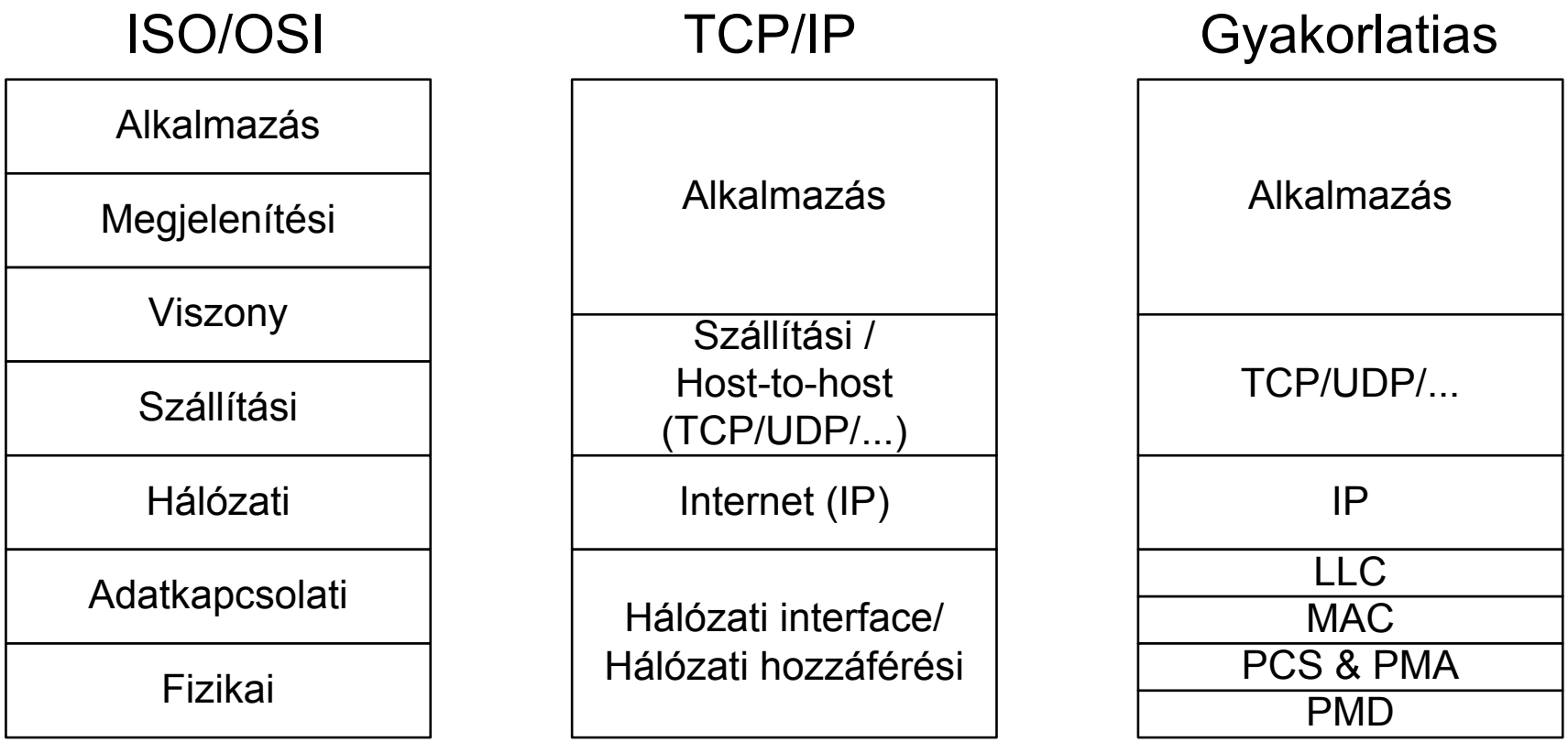
Dr. Simon Vilmos
docens
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2014.Május 6.

A tartalomból

- Hálózati alkalmazások
- Alkalmazásprotokollok
- Infrastrukturális szolgáltatások
  - Névfeloldási szolgáltatás
  - Címkonfiguráció
- Levelezési rendszerek
- Webes rendszerek

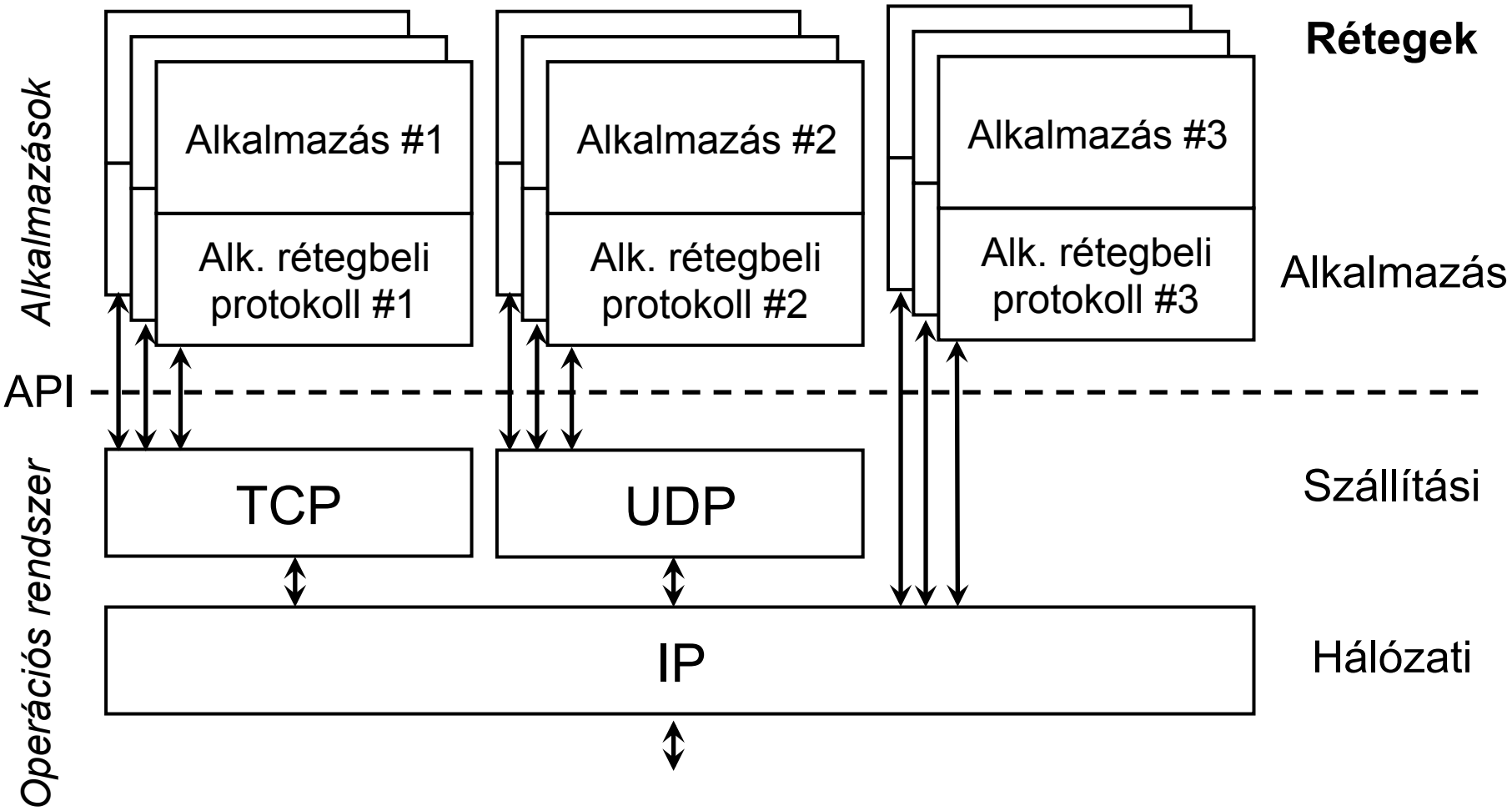
TCP/IP architektúra és az ISO/OSI rétegmodell



IP: Internet Protocol
TCP: Transmission Control Protocol
UDP: User Datagram Protocol
LLC: Logical Link Control

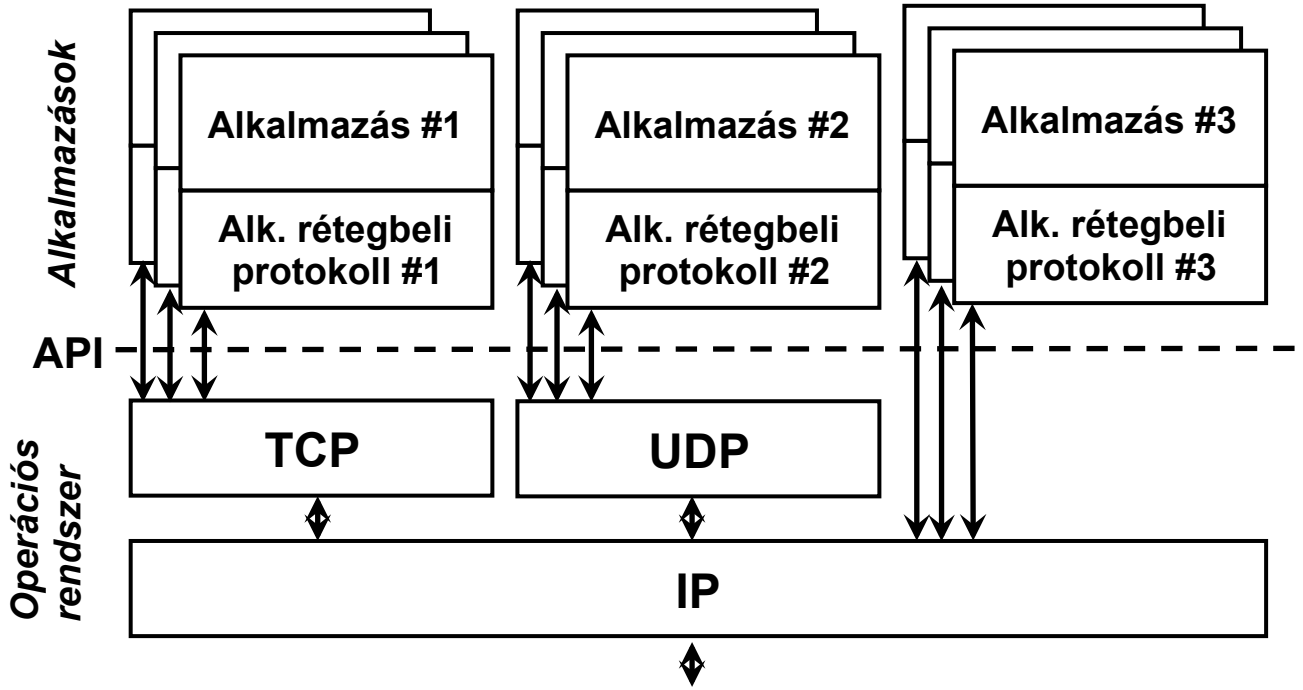
MAC: Medium Access Control
PCS: Physical Coding Sublayer
PMA: Physical Medium Attachment
PMD: Physical Medium Dependent

Alkalmazások kapcsolata az alsóbb rétegekkel



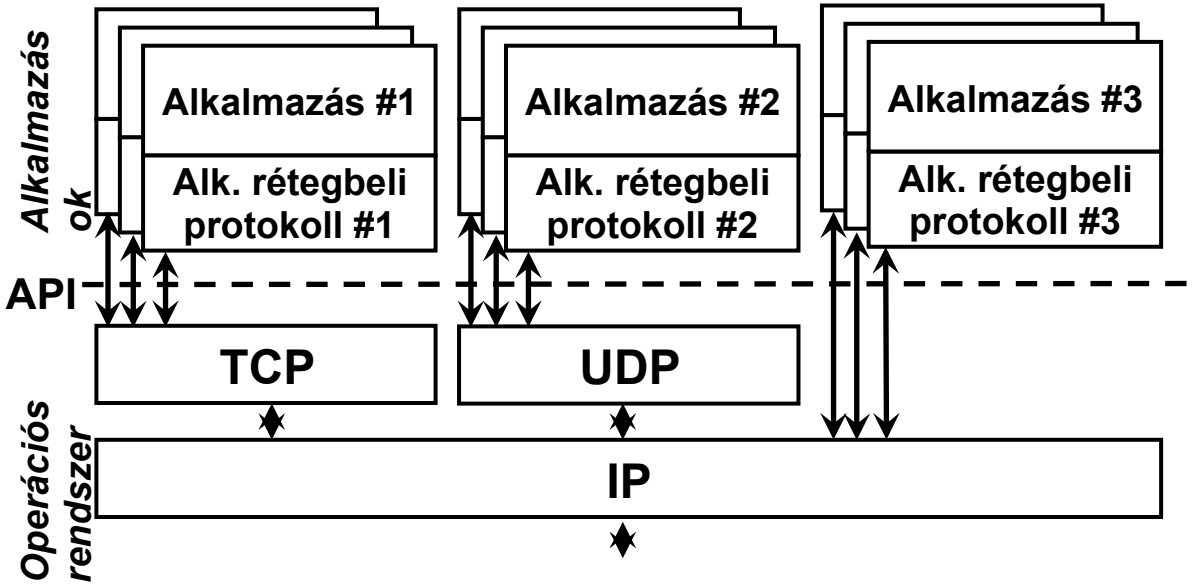
Alkalmazás-rétegbeli protokollok

- Legtöbbször az alkalmazásban kerül implementálásra
  - Alkalmazás logikájához szorosan kapcsolódik
  - Egy alkalmazás-rétegbeli protokollt tipikusan kevés alkalmazás használja
- Mégis szükséges szabványosítani (lásd RFC-k)
  - Alkalmazások együttműködése



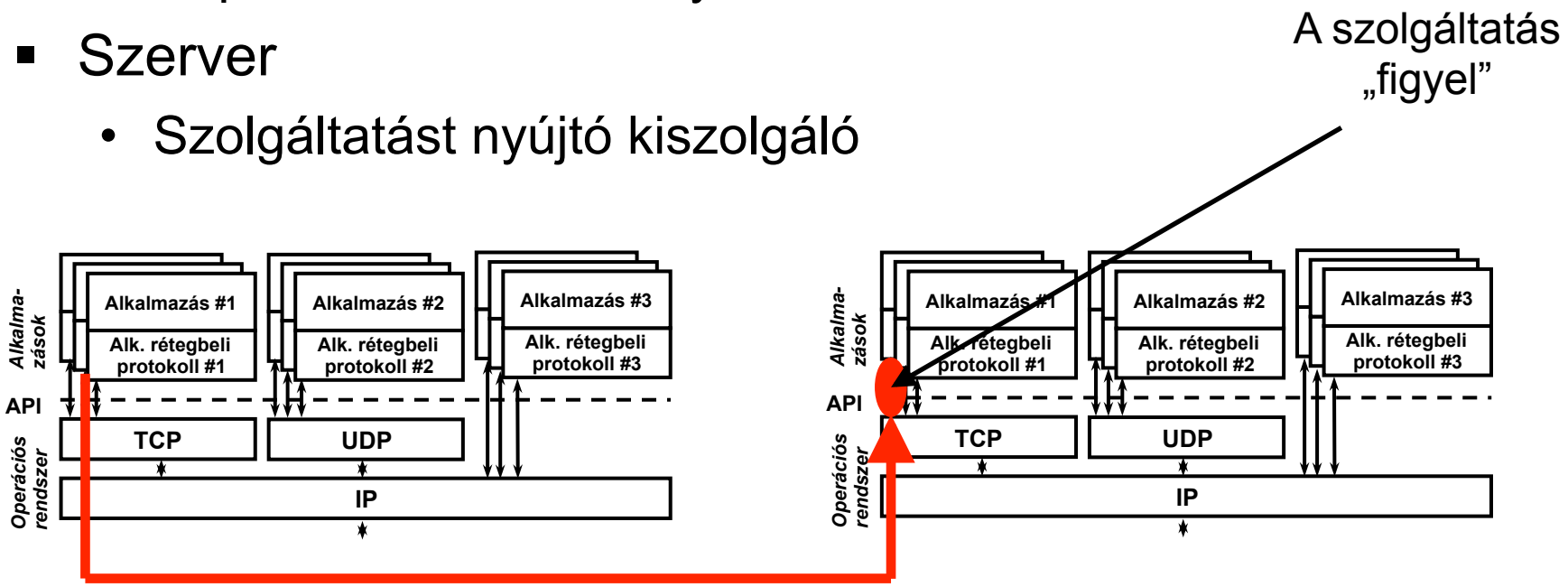
Alkalmazások környezete

- **Alsóbb rétegeket** – mint szolgáltatásokat – az **operációs rendszer** biztosítja
- **Elfedi** a tényleges rétegeket
- Csak egy interfészt (API: Application Programming Interface) biztosít
  - SAP (Service Access Point)
- Ennek rendszerhívásait használva létrehozható a kívánt kommunikációs csatorna
  - az alkalmazás által használható végződése: **socket**



Kliens-szerver architektúra

- Kliens
  - Kapcsolatot kezdeményező fél
- Szerver
  - Szolgáltatást nyújtó kiszolgáló



- Kliensnek a szolgáltatást meg kell címeznie
  - IP-cím (vagy DNS név) + szállítási protokoll + portszám

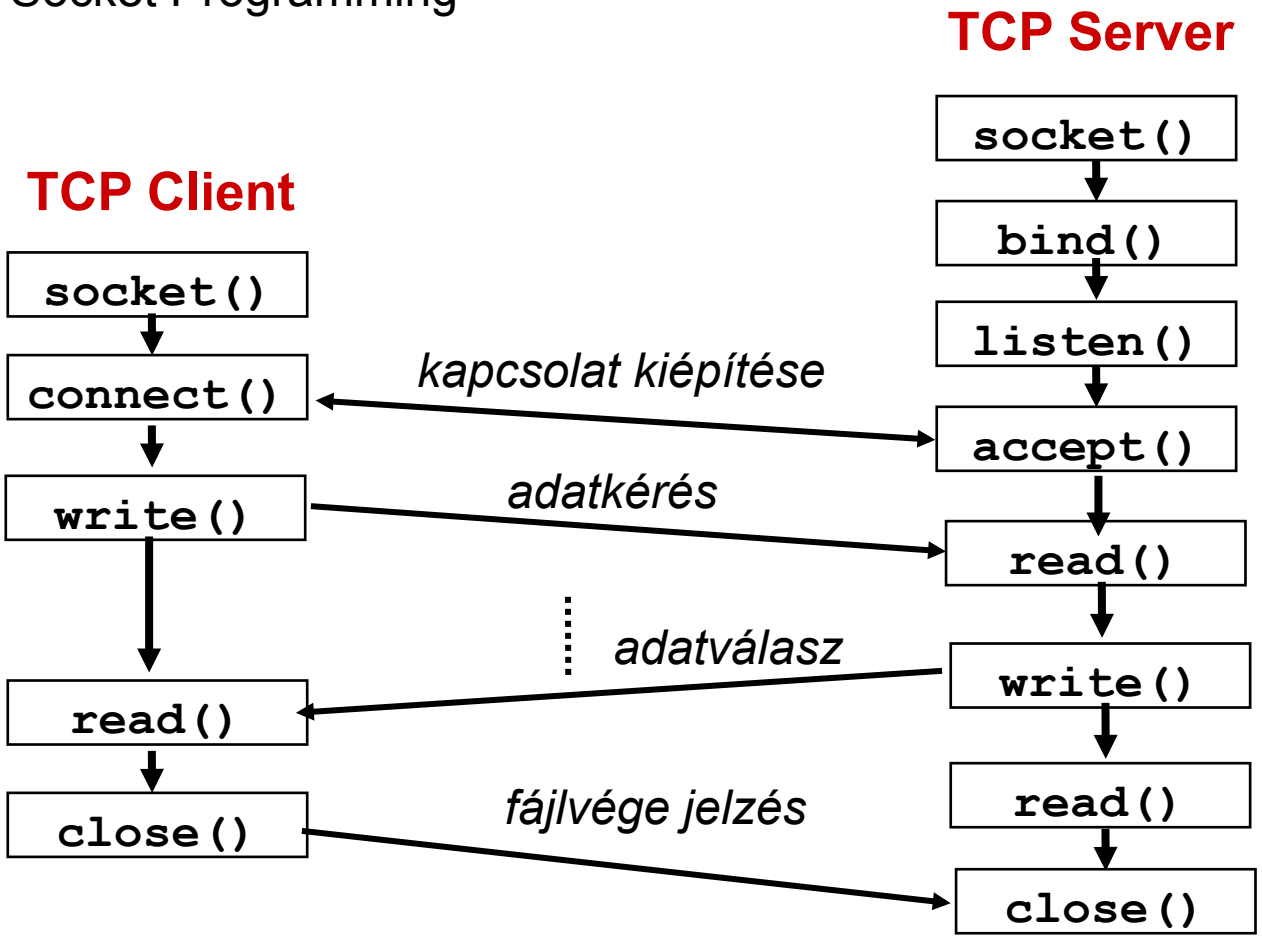
Port-hozzárendelés

- Szerveren
  - Szolgáltatást azonosítja
    - egy port maximum egy szolgáltatáshoz lehet hozzárendelve
  - **Statikus**
  - 1-65536 tartományból tipikusan 1-1023-ig
    - ***well-known ports ("jól ismert" portok)***
    - privilegizált szerver programok kapcsolódhatnak csak

- Kliensen
  - Dinamikusan kerül kiosztásra a még nem használtak közül
  - 1-65536 tartományból 1024-65535-ig
    - **Regisztrált** (1024-49151) illetve **dinamikus** portok (49152-65535)

TCP kommunikáció sockethívásokkal

Socket Programming



UDP: egyszerűbb, kevesebb hívás

Milyen alkalmazás milyen szállítási rétegbeli protokollon?

	Natív IP	UDP	TCP
Kapcsolatorientált	✗	✗	✓
Megbízható	✗	✗	✓
Üzenetméret (tipikus)	rövid	rövid	hosszú
Adattovábbítás jellege	datagram	datagram	bitstream pipe
Portkezelés	✗	✓	✓
Overhead	minimális	kicsi	nagy
Alkalmazások	vezérlési és menedzsment •ICMP, IGMP •Routing	multimédia -átvitel, névfeloldás	fájlátvitel, web, levelezés

Alkalmazások, protokollok és portszámok – natív IP felett

- Natív IP szolgáltatást igénybevevő protokollok és az IP Protocol mező értéke*:
  - Vezérlés:
    1: ICMP (Internet Control Message Protocol)
    2: IGMP (Internet Group Management Protocol)
  - Routing:
    8: EGP (Exterior Gateway Protocol)
    89: OSPF (Open Shortest Path First)
  - Egyéb:
    132: SCTP (Stream Control Transmission Protocol)
  - Valódi szállítási rétegbeli protokollok:
    6: TCP (Transmission Control Protocol)
    17: UDP (User Datagram Protocol)

* kezdeti lista az RFC 790-ben

* a mező értékeit **az IANA** (Internet Assigned Numbers Authority) felügyeli

Alkalmazások, protokollok és portszámok – UDP felett

- UDP szolgáltatásait igénybevevő protokollok és a tipikus **szerveroldali** UDP portszám értéke*:
  - **53 DNS (Domain Name System)**
    - névfeloldás
  - **67** BOOTP (Bootstrap Protocol) (Server)
    DHCP (Dynamic Host Configuration Protocol)
    - konfigurálás
  - **68** BOOTP (Bootstrap Protocol) (Client)
    DHCP (Dynamic Host Configuration Protocol)
    - konfigurálás
  - 69 TFTP (Trivial File Transfer Protocol)
    - fájlátvitel
  - 123 NTP (Network Time Protocol)
    - időszinkronizáció
  - 161 SNMP (Simple Network Management Protocol)
    - hálózatmenedzsment
  - 520 RIP (Routing Information Protocol)
    - routing

* a portszámokat az IANA (Internet Assigned Numbers Authority) felügyeli

Alkalmazások, protokollok és portszámok – TCP felett

- TCP szolgáltatásait igénybevevő protokollok és a tipikus szerveroldali TCP portszám értéke*:

20 és 21 FTP (File Transfer Protocol)
- Ugyanahhoz a protokollhoz két port is!

22 SSH (Secure Shell)

23 Telnet

25 SMTP (Simple Mail Transfer Protocol)

53 DNS (Domain Name System)
- Ugyanaz a protokoll UDP-n és TCP-n is!

80 HTTP (HyperText Transfer Protocol)

110 POP3 (Post Office Protocol version 3)

143 IMAP4 (Internet Message Protocol version 4)

443 HTTPS (HTTP Secure)

465 SMTPS (SMTP Secure)

993 IMAP4S (IMAP4 Secure)

995 POP3S (POP3 Secure)

* a portszámokat az IANA (Internet Assigned Numbers Authority) felügyeli

DNS – Domain Name System

DHCP – Dynamic Host Configuration Protocol

INFRASTRUKTURÁLIS SZOLGÁLTATÁSOK

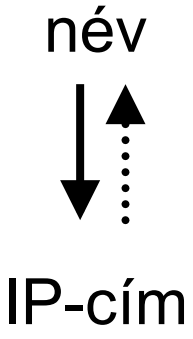
DNS – Domain Name System

NÉVFELOLDÁSI RENDSZER

A névfeloldás szerepe és követelményei

- Eltérő reprezentációk közötti megfeleltetés
  - Humán: könnyen megjegyezhető, beszédes nevek
  - Gépi: IP-címek

  → névfeloldás
- Követelmények:
  - Jó skálázhatóság
  - Hibatűrés
  - Aktuális (friss) információk

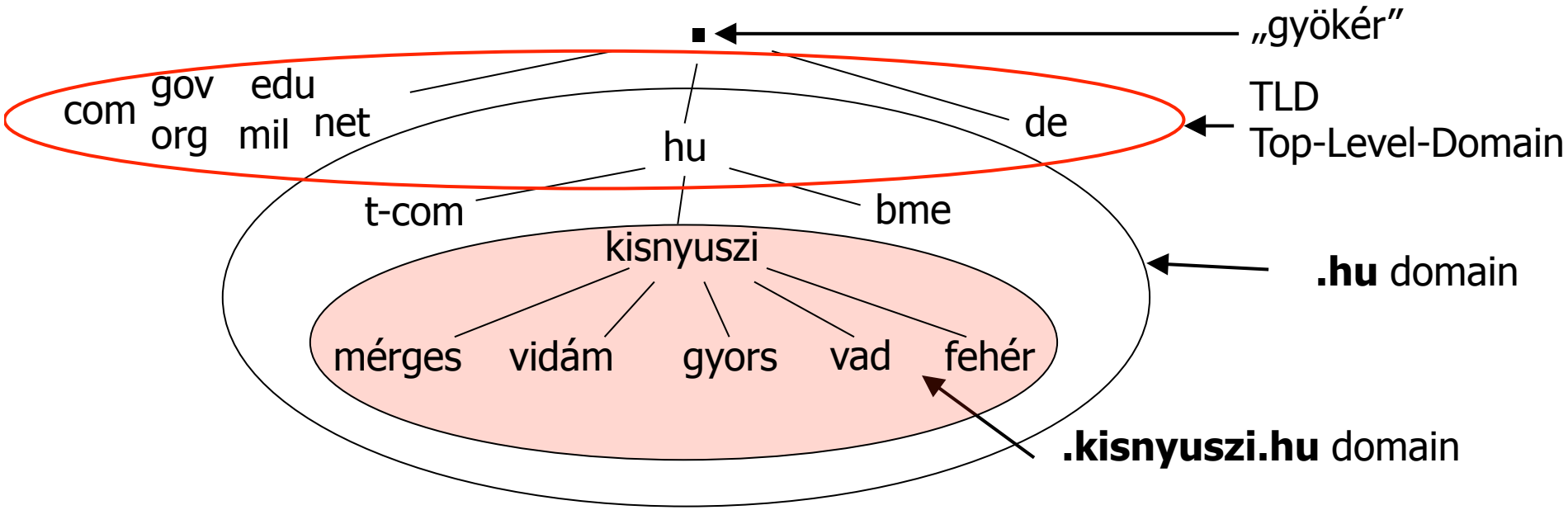


A DNS (Domain Name System) névtere

- Hierarchikus
- Állomások azonosítása: FQDN (Fully Qualified Domain Name)

vad.kisnyuszi.hu.

host neve — tartomány neve (FQDN suffix)

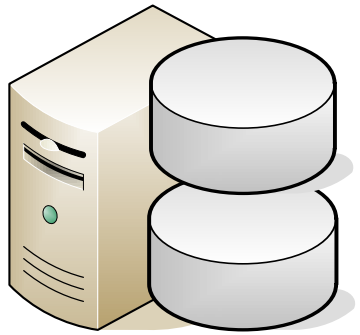


A DNS névtér „csúcsa”

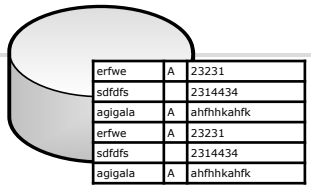
- Root (gyökér):
  - „.” – ponttal jelölik
  - Elméletileg minden FQDN emiatt ponttal zárul
  - **Gyökérzóna**: legmagasabb szintű DNS zóna
    - Valójában világszerte 13 DNS szerver klaszter szolgáltatja
    - *a.root-servers.net*-től *m.root-servers.net*-ig terjednek
  - **Anycast címzés** előnye!
- Top level domains: mindegyik a root domain-ben van
  - Az IANA adminisztrálja
    (Internet Assigned Numbers Authority)
  - country code top-level domains (ccTLD): pl. .hu (284 db)
  - generic top-level domains (gTLD, 20db):
    pl. .org, .edu, .net, .com, .gov, .mil
  - infrastructure top-level domains: egy van, az .arpa

A DNS zóna és névszerverek

- Zóna
  - Minden tartomány csúcsának vagy egészének adatait tároló adatbázis
  - A DNS nevével azonosítjuk (pl. kisnyuszi.hu.)
- DNS szerver
  - Egy vagy több zónát tárol, szolgál ki
- **Elsődleges/másodlagos** DNS szerver
  - Egy adott zónára vonatkozóan
    - Elsődleges: írható és olvasható
    - Másodlagos: csak olvasható
  - Minden esetben pontosan 1 elsődleges és legalább 1 másodlagos kell
    - → hibatűrés
    - → terheléselosztás és skálázhatóság
  - Szinkronizálás monoton növekvő verziószám alapján
    - Egyre elfogadottabb a verziószám: YYYYMMDDnn



A zóna elemei: rekordok



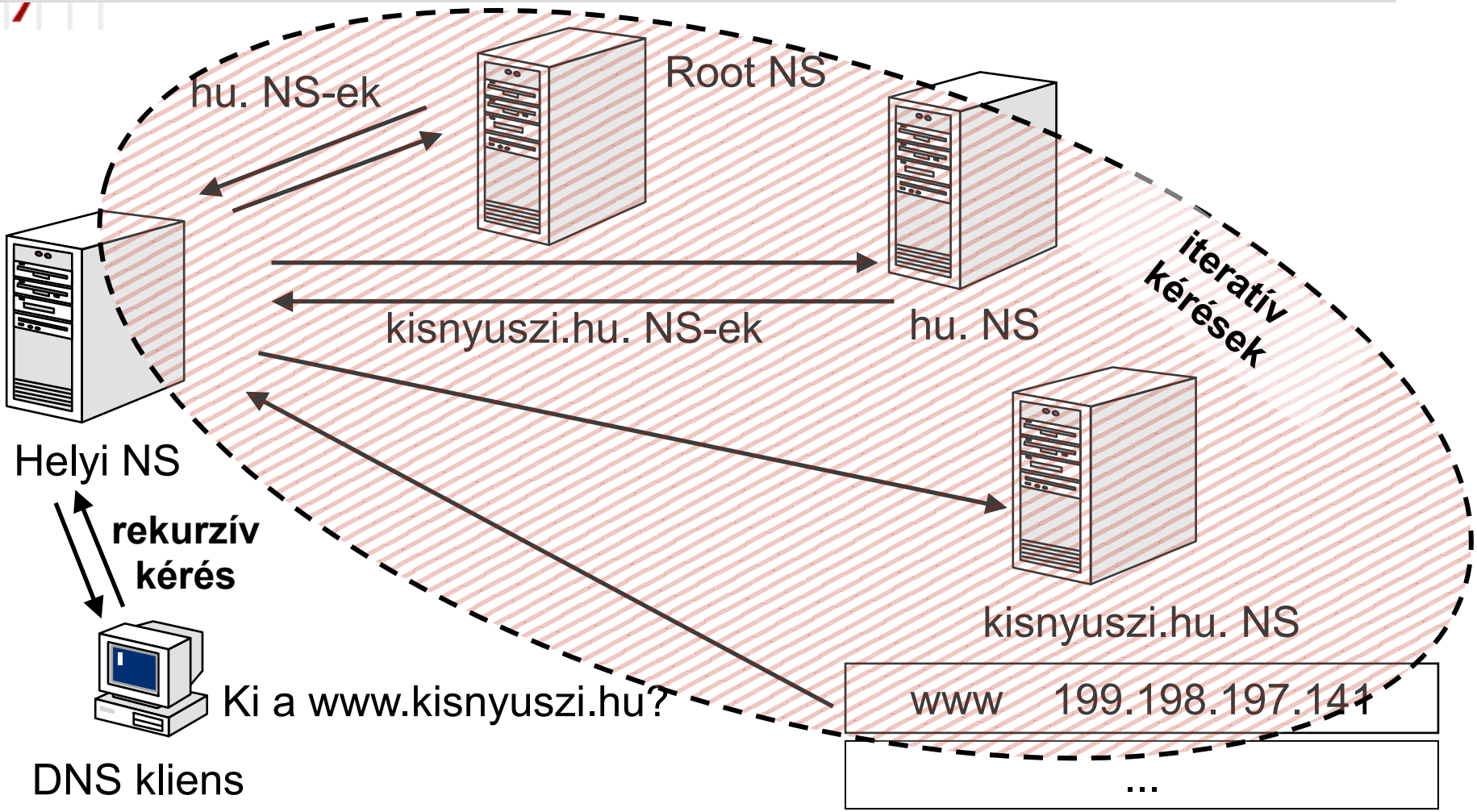
Az erőforrásrekordok (**Resource Records - RR**) gyakori típusai:

- SOA (Start of Authority)
  - adminisztratív adatok
    - az elsődleges DNS szerver neve
    - zóna verziószáma (ez alapján a szinkronizálás)
    - kapcsolattartó e-mail címe
- A (Address)
  - név – IP-cím
  - a legtipikusabb felhasználás
- CNAME (Canonical Name)
  - más néven „alias”
  - név – név összerendelés
- PTR (Pointer)
  - IP-cím – név
  - ún. reverse zónában
- NS (Name Server)
  - az adott zónát kiszolgáló DNS szerverek
  - legalább kettő kell
- MX (Mail Exchange)
  - SMTP kiszolgálót azonosít
  - Több is megadható **preferenciával** (prioritással)
- SRV (Service Locator)
  - MX általánosítása
  - tetszőleges szolgáltatásra (pl. SIP)

Altartományok (subdomain)

- delegálhatóság

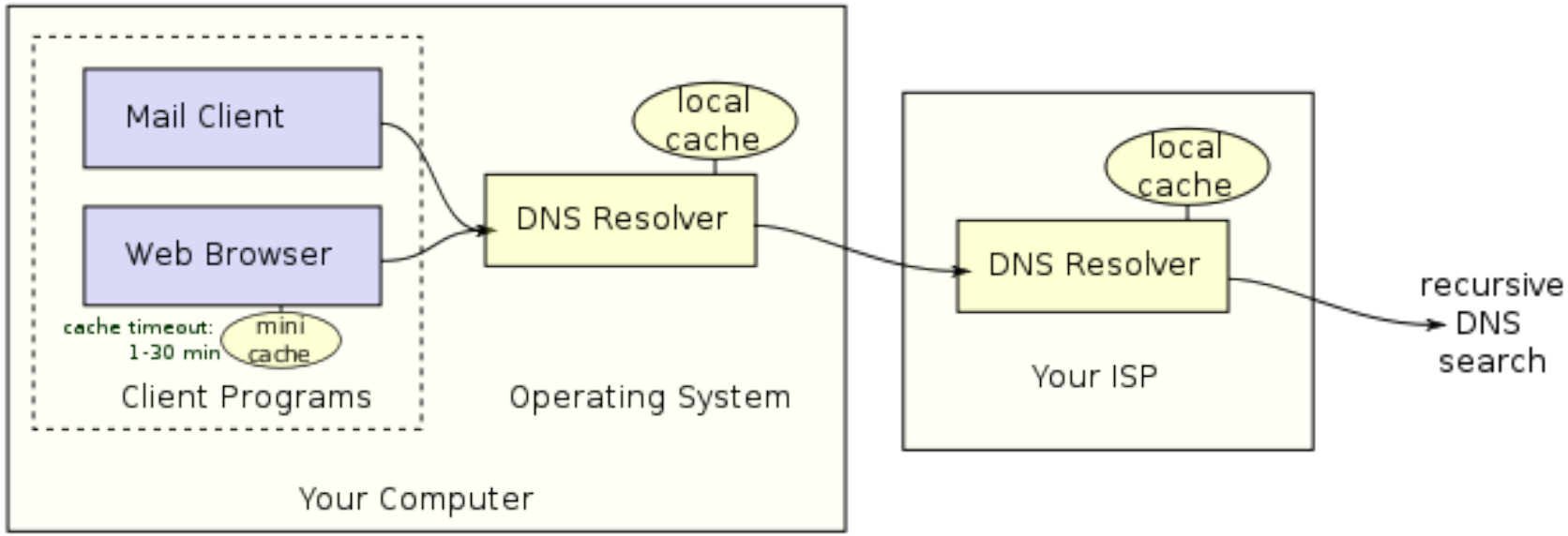
Névfeloldás menete



- **Rekurzív kérés**: a konkrét (vagy a negatív) választ várja, a címzettnek a feladata a névfeloldás
- **Iteratív kérés**: a minél közelebbi felelős megtalálása, arra egy hivatkozást ad vissza: **referral válasz**

Névfeloldás gyorsítótárral

- A DNS-kéréseket a helyi gépen az operációs rendszer oldja fel egységesen
- **DNS gyorsítótár (cache)** a helyi gépen és a DNS szerveren



- Minden rekordnak **TTL-je (Time To Live**) valódi másodpercben megadva → elévülés

- **Autoritatív válasz**: ha a rekordért felelős szerverek valamelyikétől származik a válaszol
- **Nem autoritatív**: ha gyorsítótárból származik

Névfeloldás menete

(0. Böngésző gyorsítótára)

1. Helyi gép gyorsítótára
2. Helyi gépen „hosts” fájl
3. Lekérdezés DNS szerverektől
   - Ha van DNS szerver megadva
     - Lekérdezés az elsődleges DNS szervertől, ha elérhető (rekurzív)
       - Az a cache-ből kiszolgál vagy névfeloldást végez (iteratív)
     - Lekérdezés a másodlagos DNS szervertől, ha meg van adva és az elsődleges nem érhető el (rekurzív)
   - Ha nincs DNS szerver megadva vagy nem elérhetők, akkor lekérdezés valamely root NS-től, majd a hivatkozott NS-ektől (iteratív)

DNS mint protokoll: kapcsolatok

- A DNS protokoll felhasználási területei:
  - Lekérdezés:
    - DNS kliens ⇔ DNS szerver
    - UDP 53 ← rövid, gyors üzenetváltás
      (TCP 53)
  - Zónaletöltés
    - Elsődleges DNS szerver ⇨ másodlagos DNS szerver
    - TCP 53 ← hosszabb, megbízhatóbb

DNS mint protokoll: üzenetek

- **Kérés** elemei
  - Kért rekord típusa(i)
  - Feloldandó név vagy IP-cím
  - **Rekurzív kérés esetén** RD (Recursion Desired) bit beállítva
- **Válasz**
  - Pozitív válasz: egy vagy több elemű lista
    - Ebből „véletlenszerűen” (round-robin) választ
  - **Authoritatív válasz esetén** az AA (Authoritative Answer) bit beállítva
  - **Referral** válasz (egy vagy több elemű lista)
    - Kiegészítő hivatkozásokat tartalmaz, mely közelebb visz a feloldáshoz
      - pl.: illetékesebb NS; A rekord helyett azonos nevű CNAME rekord
  - Negatív válasz: nem található bejegyzés, nem oldható fel
    - Nincs válasz ≠ negatív válasz

Dynamic Host Configuration Protocol

DHCP

Általában a DHCP-ről

- Mi a DHCP?
  - IP-beállításokat oszthatunk ki vele dinamikusan
- Előnyei:
  - Kliensek egyszerű beállítása
  - Módosítások központilag
  - Mobilitás hálózatok között (eltérő beállítások)

IP cím igénylése

DISCOVER és OFFER
Bérlet kérése és felajánlása

- Kérés:

  0.0.0.0-tól → 255.255.255.255-nek
  (még nincs IP-címe) (bárkinek)

- Ajánlat:

  255.255.255.255-nek ← 192.168.1.1-től
  (bárkinek; nem címezhető) (egy DHCP szerver címe)

- Ajánlat tartalma
  - IP-cím
  - Alhálózati maszk
  - Bérleti idő
  - DHCP szerver IP címe
  - **Igénylő MAC címével!**

REQUEST és ACK
Bérlet kiválasztása és nyugtázása

- Kiválasztás
  - az első ajánlatot (pl. ha több DHCP szerver)
  - Kiválasztási üzenet:
    - 0.0.0.0-tól 255.255.255.255-nek
    - Az üzenet tartalma
      - **kért IP-cím**
      - **DHCP szerver IP-címe**
- Nyugta
  - 192.168.1.1-től, 255.255.255.255-nek
  - Nyugtázó üzenet tartalma
    - **kiosztott IP-cím**
    - **alhálózati maszk**
    - **bérleti idő**

Bérleti idő (TTL)

- Bérlet időbeli kezelése
  - Félidőben hosszabbítási kérés
  - 7/8 TTL-nél új igénylése

- Hosszú vagy rövid TTL?
  - Rövid mellett
    - Ha a kliens szabálytalanul távozik a hálózatból
      - a bérletét nem adja vissza
    - Ha a kliens szabálytalanul újraindul
      - nem adja vissza a bérletét, és még újat is igényel
    - A beállításváltozások gyorsan életbe lépjenek
  - Hosszú mellett
    - Ne legyen nagy hálózati forgalom

Ami DHCP-vel beállítható…
DHCP opciók, paraméterek

- **0x01 Subnet Mask (alhálózati max)**
- **0x0F Domain Name (FQDN suffix)**
- **0x03 Router (alapértelmezett átjáró(k))**
- **0x06 DNS (DNS szerver(ek))**
- **0x0C Host Name (gép neve is kioszthető)**
- 0x1F Router Discovery
- 0x21 Static Route
- 0x2B Vendor Specific (gyártófüggő beállítások)
- 0x2C WINS
- 0x2E NBT
- 0x2F Node Type
- **0x32 Requested Address (igényelt IP-cím)**
- **0x33 Lease Time (TTL)**
- **0x36 DHCP Server (DHCP szerver IP-címe)**
- **0x37 Parameter Request List (igényelt paraméterek listája)**
- **0x3A Renewal Time (megújítási idő)**
- 0x3B Rebinding Time
- 0x3C Client Class Information
- 0x4D User Class Information
- 0xF9 Static Route CIDR

Stb…

DHCP egyéb alkalmazása

- DHCP hibatűrés
  - Több DHCP használata egy hálózatban, de **diszjunkt IP-címtartományok osztása**!
- DHCP kiterjesztése több hálózati szegmensre
  - A routerek nem engedik át a DHCP üzeneteket
  - A routerekre ún. „**DHCP Relay Agent**”-et telepítve az továbbítja a DCHP forgalmat a DHCP szerverek és kliensek között

- IPv6-ban minden router egyben DHCP szerver is

Telnet

FTP – File Transfer Protocol

SZÖVEG- ÉS FÁJLÁTVITEL

Telnet

- Egyik legrégebbi alkalmazás
- Távoli parancssor
  - Parancsok elküldése
  - Visszajelzések megjelenítése
- Még ma is alkalmazzák főként hálózati eszközök egyszerű hálózati adminisztrációjára
- Nem biztonságos (jelszavak védelem nélkül)
  - SSH (Secure Shell) helyette

FTP – File Transfer Protocol

- Az egyik legelső fájlátvitelre tervezett protokoll
  - RFC 959
- TCP 21-es port
  - Ha a TCP 20-as portot használjuk adatcsatornaként, akkor ez csak vezérlés
- Parancsok
  - open – kapcsolat létrehozása
  - ls – aktuális könyvtár listázása
  - put – feltöltés
  - get – letöltés
  - delete – törlés
  - bye – kapcsolat lebontása
  - …

SMTP – Simple Mail Transfer Protocol

POP3 – Post Office Protocol v3

IMAP4 – Internet Message Protocol v4

LEVELEZŐ RENDSZEREK

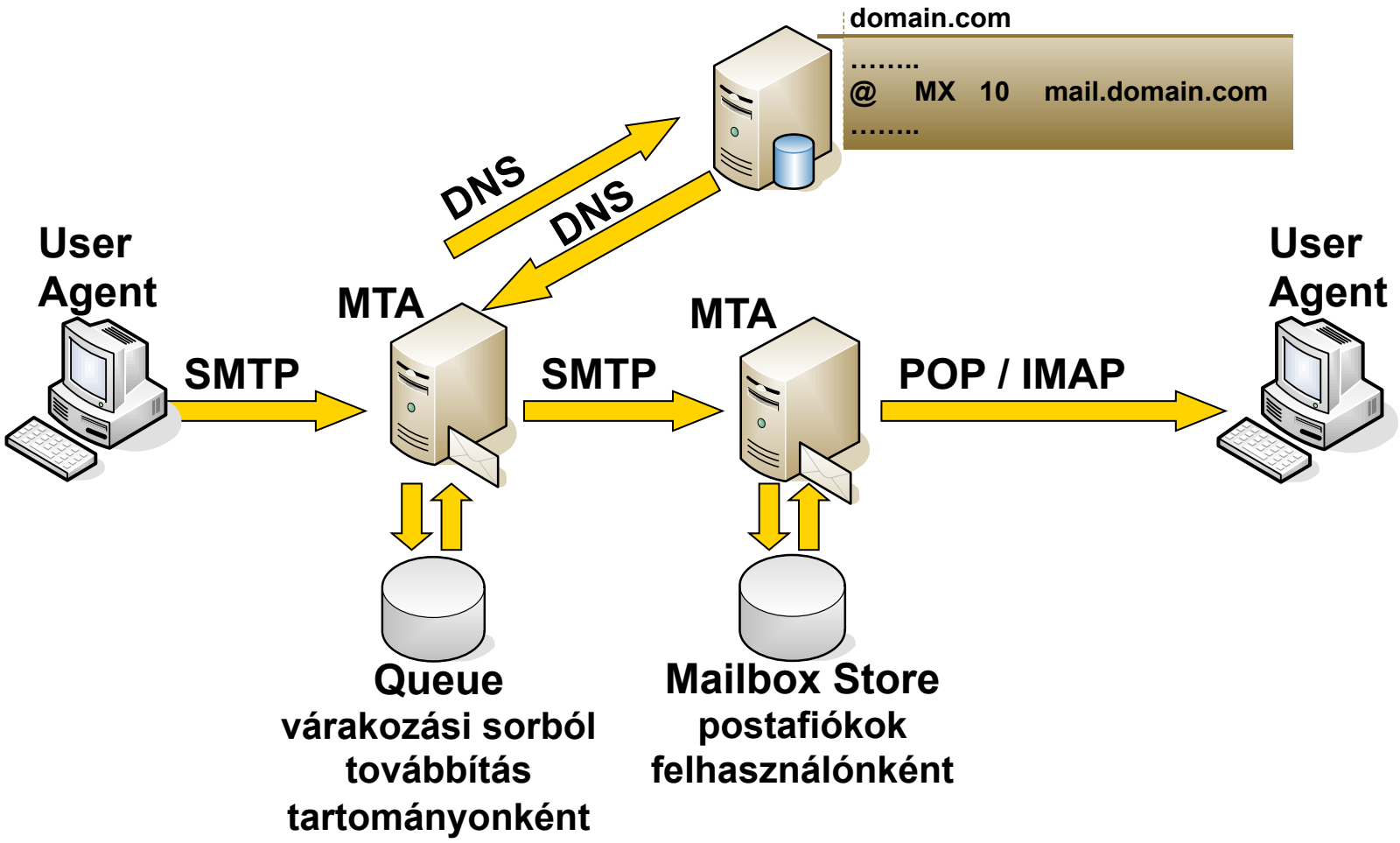
Levelező rendszerek

- Komponensek
  - (Mail) User Agent ((M)UA) (levelező kliens)
  - Mail Transfer Agent (MTA) (SMTP szerver)

- Használt protokollok
  - SMTP: levél továbbításra
  - POP3: levelek lekérdezése
  - IMAP4: levelek lekérdezése

- Címzett meghatározása
  - DNS segítségével (MX rekord)

Levelező rendszerek



Levelek lekérdezése – POP3 és IMAP4

POP3

- Post Office Protocol version 3
- RFC 1939, 1957, 1725
- Parancsorientált
- TCP 110-es port
- Levelek lekérdezésére
- POP3S
  - POP3 TLS tikosítással
  - TCP 995

IMAP4

- Internet Message Protocol version 4
- RFC 2060, 1731, 1730
- Parancsorientált
- TCP 143-as port
- Levelek lekérdezésére
- IMAP4S
  - IMAP4 TLS tikosítással
  - TCP 993
- Intelligensebb a POP3-nál:
  - Könyvtárstruktúra támogatása
  - Keresés támogatása
  - Nem törli automatikusan a szerveren tárolt leveleket

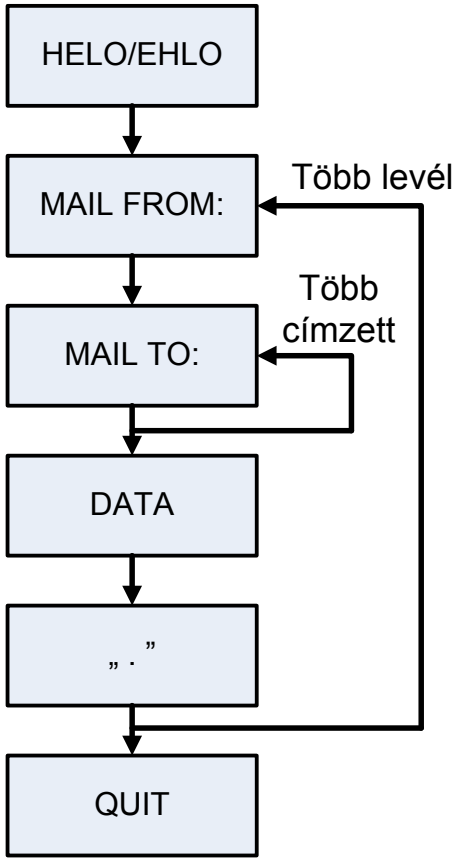
SMTP – Simple Mail Transfer Protocol

- RFC 821, 822
- Levelek továbbítására (első „killer application”)
- Parancsorientált állapotkódokkal
- TCP 25-ös port
- SMTP relay
  - Nem közvetlen továbbítás
  - Egy (vagy több) SMTP (relay) szerver közbeiktatásával
- SMTPS (SMTP Secure)
  - SMTP TLS csatornában
  - TCP 465

A leggyakoribb SMTP parancsok

- HELO
  - Üdvözlés
  - ESMTP esetén EHLO
- MAIL FROM:<feladó e-mail címe>
- RCPT TO:<címzett e-mail címe>
- DATA
  - Adat következik
- <CR><LF>.<CR><LF>
  - Adat vége
- QUIT
  - SMTP kapcsolat bontása
- VRFY <e-mail cím>
  - Létezik-e az adott e-mail cím
- HELP
- NOOP
  - Kapcsolat ellenőrzése, fenntartása

Tipikus kapcsolat folyamatábrája



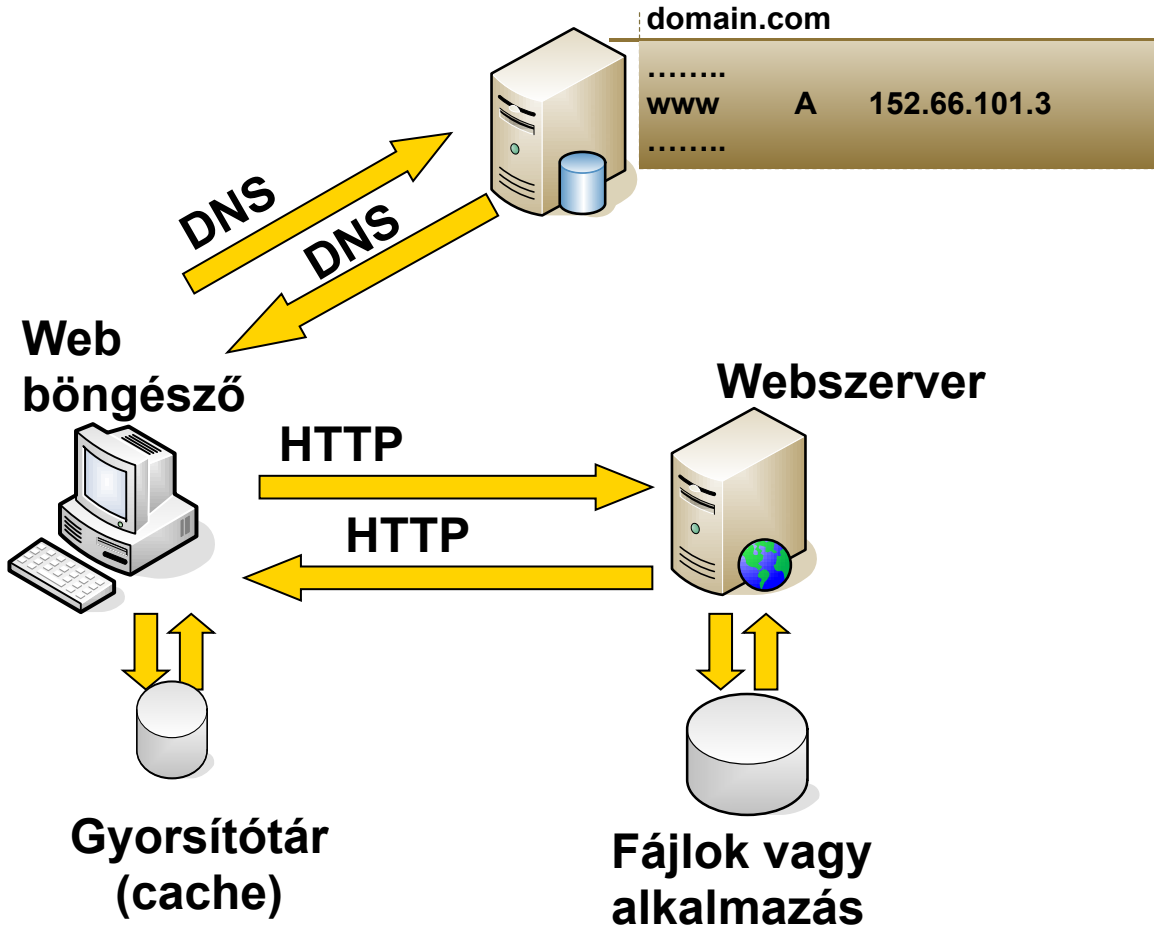
Példa egy SMTP kommunikációra

S: 220 lappfold.fi
C: HELO bme.hu
S: 250 Hello bme.hu, pleased to meet you
C: MAIL FROM: <jogyerek@bme.hu>
S: 250 jogyerek@bme.hu... Sender ok
C: RCPT TO: <mikulas@lappfold.fi>
S: 250 mikulas@lappfold.fi ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Kedves Mikulás!
C: Jó gyerek voltam. Hozzál sok csokit!
C: Köszönöm,
C: Jógyerek Jóska
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 lappfold.fi closing connection

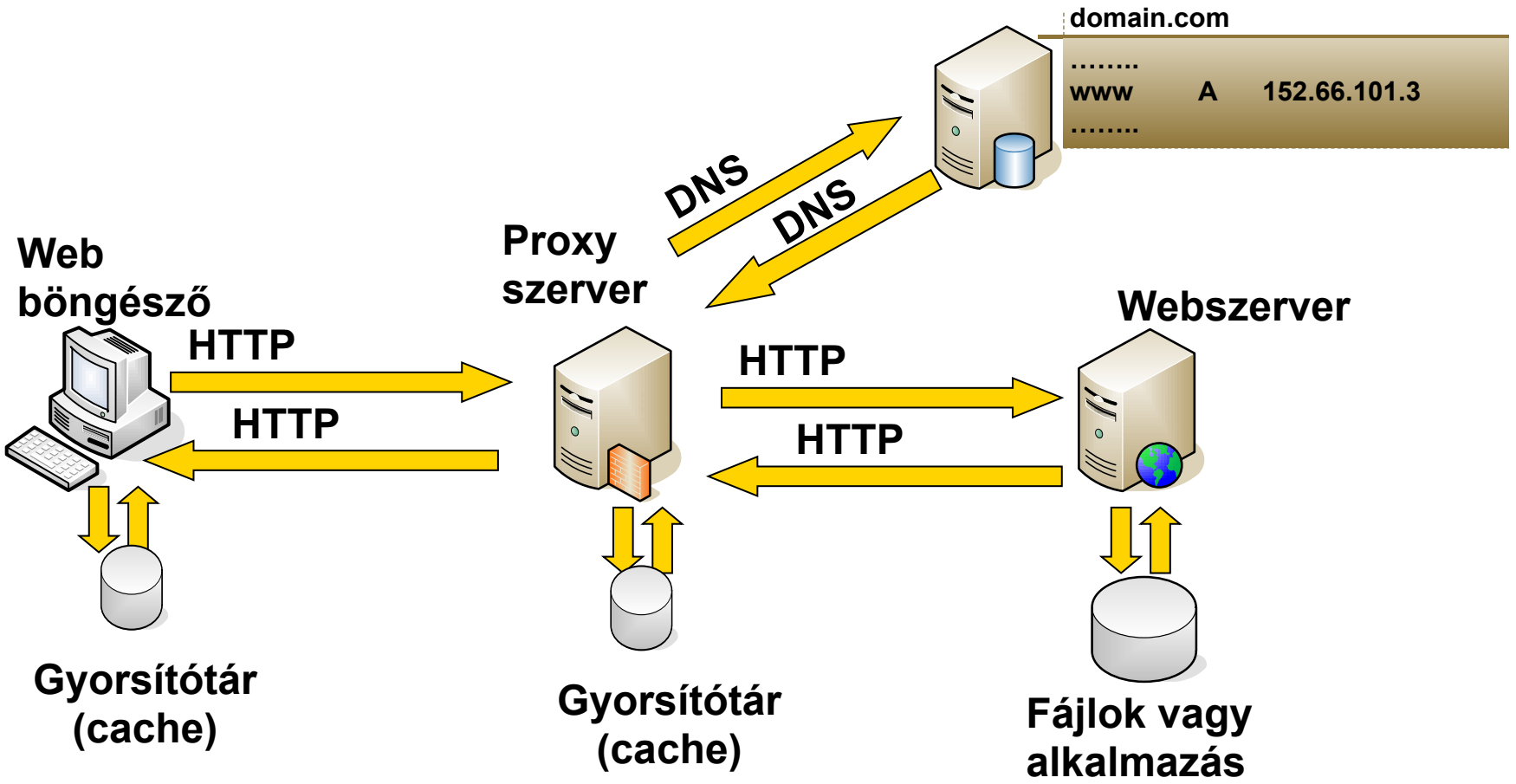
HTTP – HyperText Transfer Protocol

WEBES RENDSZEREK

Webes rendszerek



Webes rendszerek proxyval



HTTP – HyperText Transfer Protocol

- RFC 2068
- Web
  - Tim Berners-Lee, CERN
  - a második „killer application”

- Parancsorientált állapotkódokkal

- Speciális fejlécek

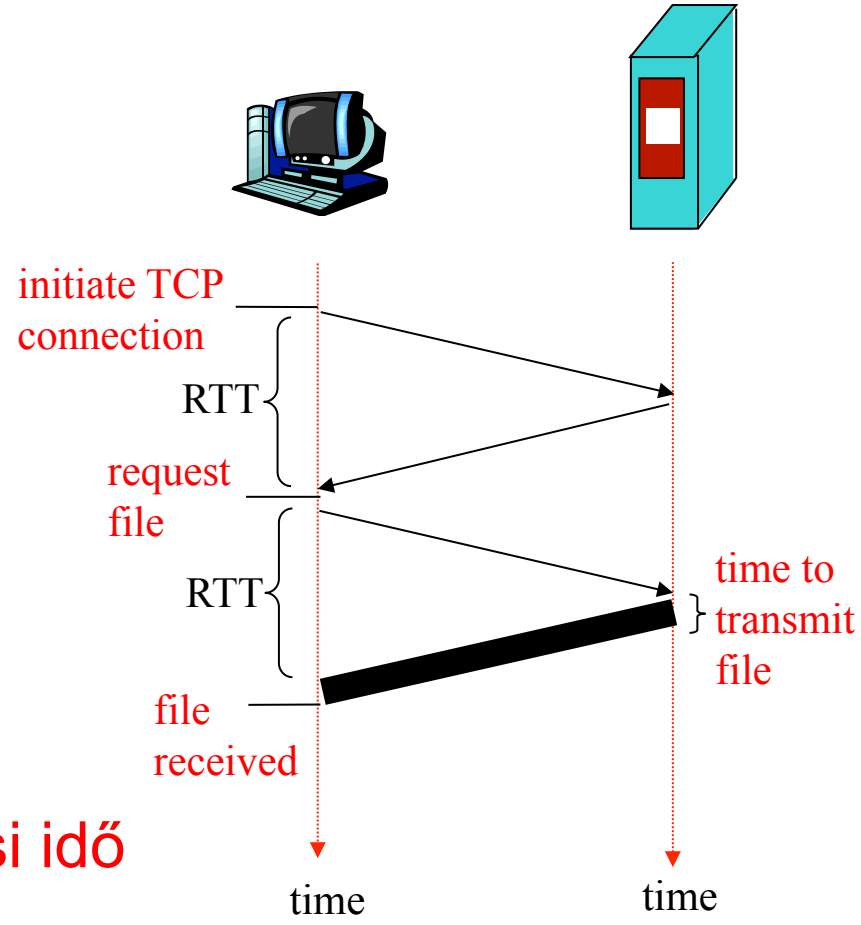
- Állapotmentes protokoll
- TCP 80
- Proxy
  - Kliens nevében jár el
  - Főként a hatékony gyorsítótárazás miatt

Perzisztens HTTP

- *Nem-perzisztens*:
  - minden kérés/válasz pár egy **külön** TCP kapcsolaton keresztül
  - Legfeljebb egy objektumot visz át egy TCP kapcsolaton
  - HTTP/1.0
- *Perzisztens*:
  - **egy TCP kapcsolaton belül** több kérés/válasz pár
  - Alapbeállítás HTTP1.1-nél

© Dr. Simon Vilmos, Hálózati Rendszerek és Szolgáltatások Tanszék

Késleltetés nem perzisztens kapcsolatnál



Teljes késleltetés = 2RTT+adási idő

© Dr. Simon Vilmos, Hálózati Rendszerek és Szolgáltatások Tanszék

Perzisztens HTTP

Nonperzisztens HTTP:

- 2 RTT/objektum
- + TCP kapcsolat overhead
- Böngészők használhatnak **párhuzamos TCP kapcsolatokat** a különböző objektumok letöltésére

Perzisztens HTTP

- Szerver **nyitva hagyja a kapcsolatot** a válasz elküldése után
- HTTP üzenetek sorozata ugyanazon kliens és szerver között **egy kapcsolaton belül**

Perzisztens pipelining nélkül:

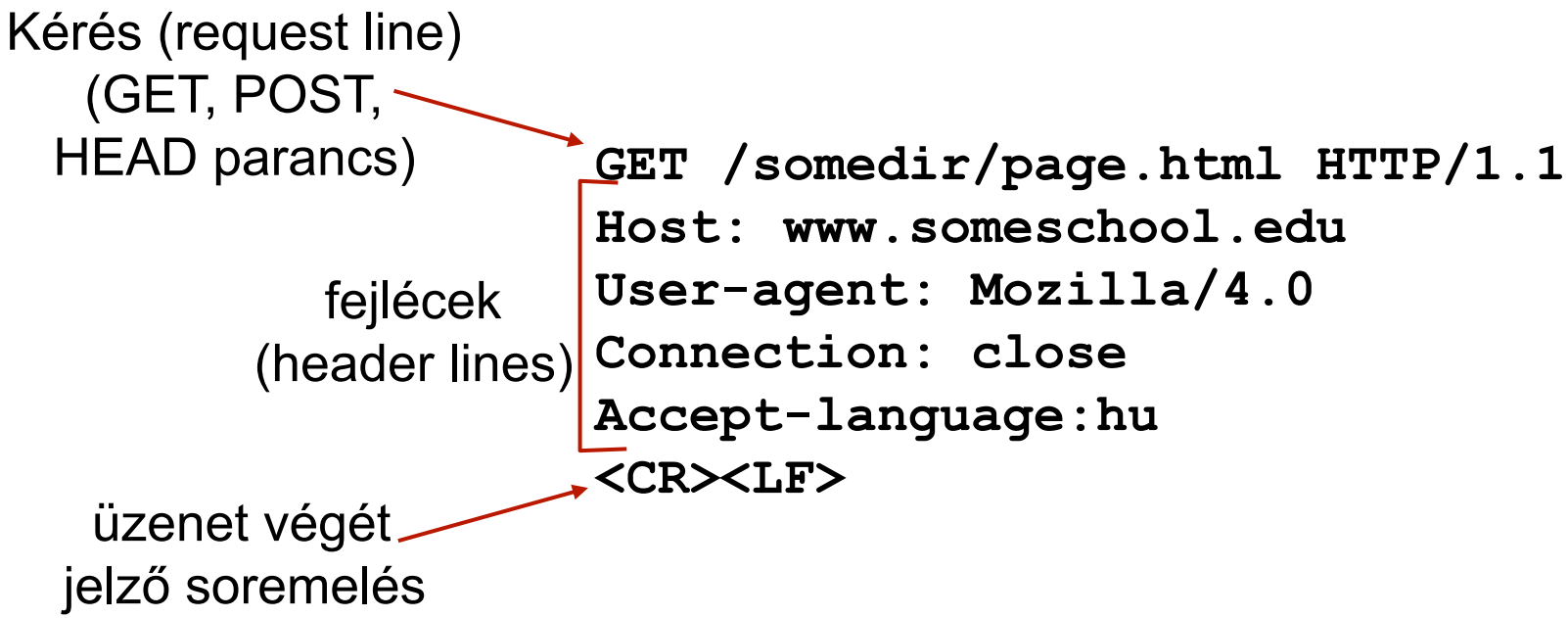
- Kliens csak akkor küld új kérést, ha előzőre **megjött a válasz**
- 1 RTT/objektum

Perzisztens + pipelining:

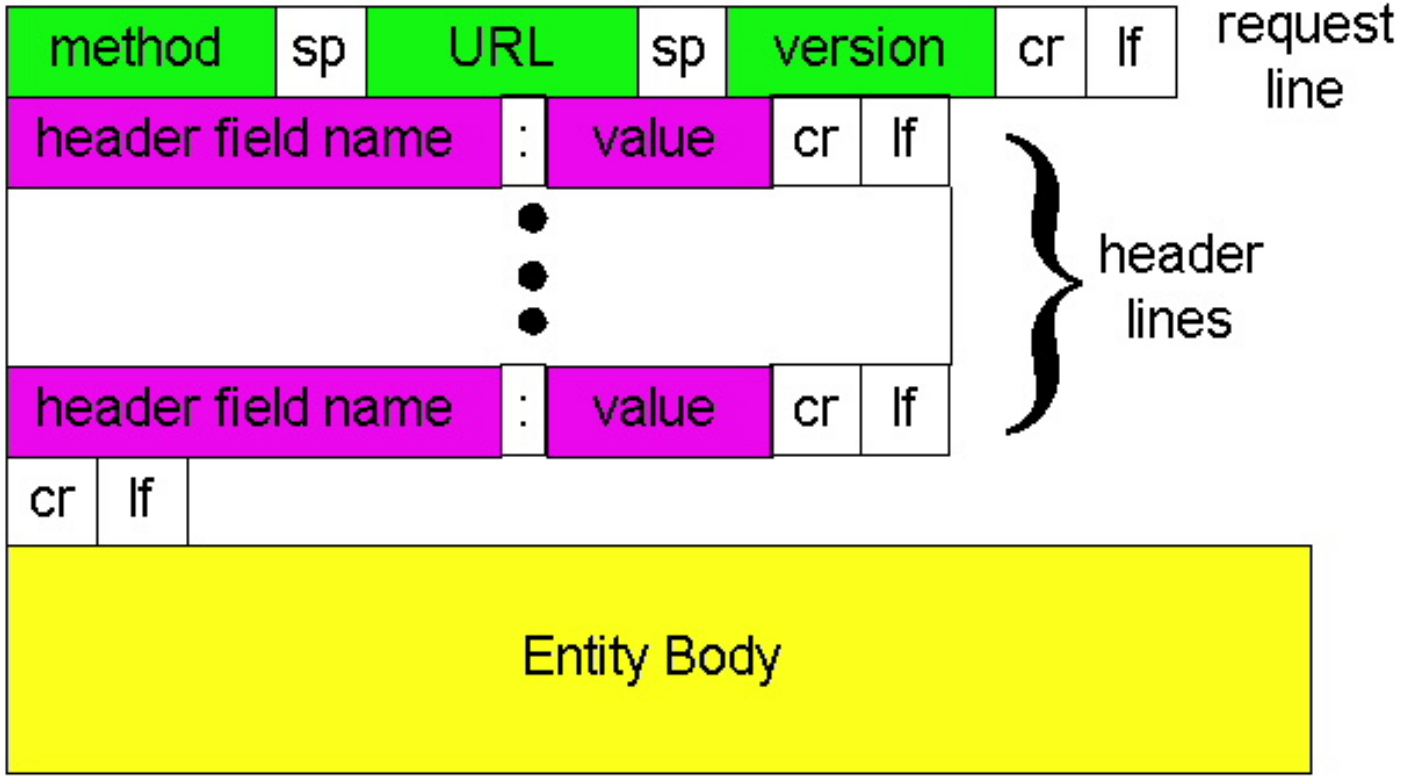
- Alapbeállítás HTTP/1.1-nél
- Kliens elküldhet több kérést is egymás után **anélkül, hogy megvárná a választ**
- 1 RTT minden objektumra

© Dr. Simon Vilmos, Hálózati Rendszerek és Szolgáltatások Tanszék

Példa egy HTTP kérésre



Általános HTTP kérés üzenet formátum



Entity Body: POST-nál van értelme!

Gyakori HTTP parancsok

- GET <URL> HTTP/1.1
  - adott URL tartalmának lekérése
- HEAD
  - mint a GET, de csak a metaadatokat adja vissza
- POST
  - a kliens ezzel tud adatokat küldeni a szervernek
- PUT
  - a POST-hoz hasonló, fájlfeltöltésre alkalmas
- DELETE
  - adott URL tartalmának törlése

Gyakori HTTP fejlécek

- Accept: elfogadható MIME típus
- Accept-Charset: elfogadható karakterkészlet
- Allow: szerver által támogatott parancsok
- Authorization: támogatott hitelesítési módok
- Content-Encoding: tömörítés típusa
- Content-Length: tartalom mérete
- Content-Type: MIME típus
- Date: lekérés dátuma és ideje
- From: a látogató e-mail címe (nem authentikációra)
- Pragma: nem meghatározott paraméter (pl. „no-cache”)
- Referer: a hivatkozó oldal URL-je
- Retry-After: 503 utáni újrapróbálkozási idő
- Server: szerver neve és verziója
- User-Agent: böngésző neve és verziója
- WWW-Authenticate: Hitelesítési információk (credentials)

Példa egy HTTP válaszüzenetre

állapotkód

fejlécek

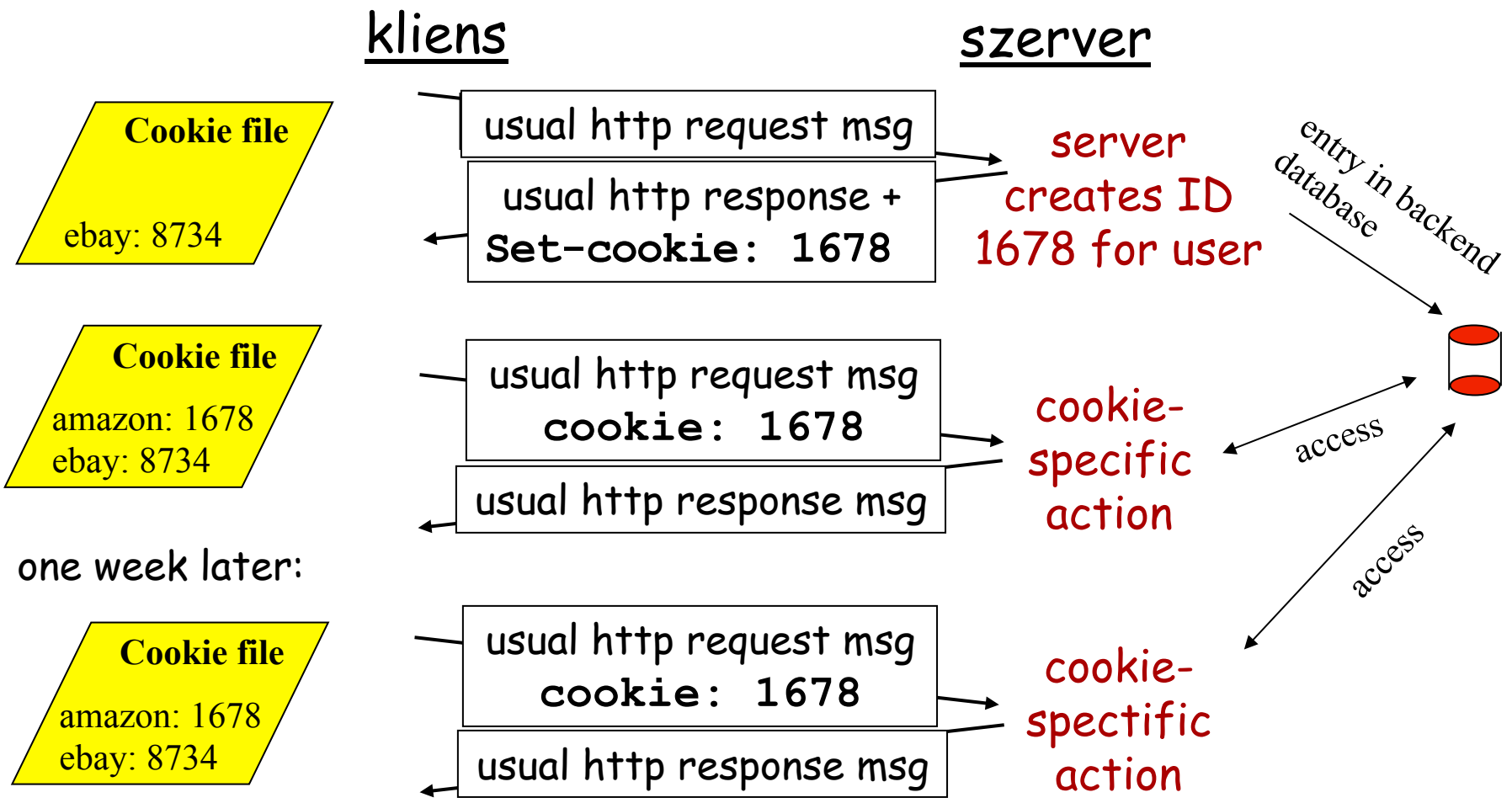
adat
(pl. a kért
HTML fájl)

```
HTTP/1.1 200 OK
Connection close
Date: Thu, 06 May 2014 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 2013 …...
Content-Length: 6821
Content-Type: text/html

data data data data data ...
```

Gyakori HTTP állapotkódok

Kód	Jelentés	Leírás
200	OK	
201	Created	POST sikeres
202	Accepted	Kérés elfogadva
204	No content	Nincs semmi a kliensnek
400	Bad request	Hibás kérés
401	Unauthorized	Hitelesítés szükséges
403	Forbidden	Hozzáférés megtagadva
404	Not found	Nem található
500	Internal Server Error	Belső szerver hiba
503	Service Unavailable	Pillanatnyilag nem szolgálható ki

Cookie-k használata



HTTP jövője

- Következő verzió: HTTP 2.0, IETF fejlesztésben
- A Google SPDY protokolljának lényeges elemeit használja
  - De facto szabvány?
- Csökkenti a web oldalak betöltési idejét:
  - multiplexált stream-ek egy TCP kapcsolaton belül
  - kérések priorizálása
  - kérés és válasz fejrészek tömörítése
    - a statikusok nincsenek újraküldve
  - szerver push és hint funkció

Összefoglalás

- Alkalmazások hálózati kapcsolata
  - Portok és socketek használata
- Infrastrukturális szolgáltatások
  - DNS – Névfeloldási szolgáltatás
  - DHCP
- Szöveg- és fájlátvitel
  - Telnet, FTP
- Levelezési rendszerek
  - SMTP, POP3, IMAP4
- Webes rendszerek
  - HTTP

Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

MŰEGYETEM 1782

BME

HÁLÓZATI ALKALMAZÁSOK II.

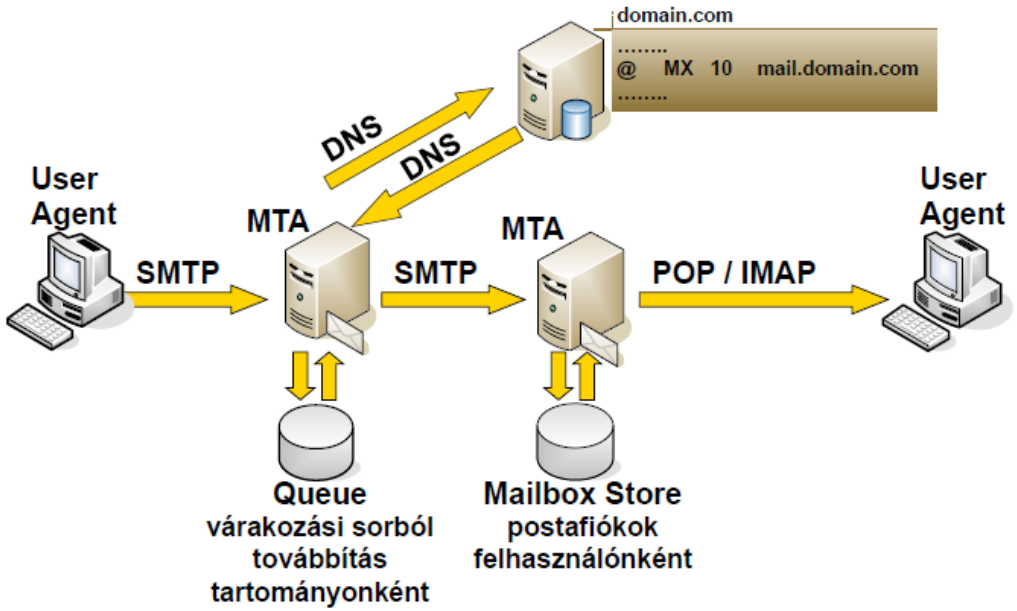
Számítógép-hálózatok (BMEVIHIA215)

Dr. Lencse Gábor
tudományos főmunkatárs
BME Hálózati Rendszerek és Szolgáltatások Tanszék
lencse@hit.bme.hu

2014. május 15.,
Budapest

Tartalom

- Levelezés gyakorlati kérdései
- FTP működése
- Virtuális webszerverek
- DHCP működése



Simon Vilmos: Számítógép-hálózatok tárgy / Hálózati alkalmazások előadás

LEVELEZÉS GYAKORLATI KÉRDÉSEI

Message Submission Protocol

- SMTP képes ellátni:
  - (M)UA – MTA
  - MTA – MTA
- A kéretlen levelek miatt célszerű az első feladatra egy új protokollt használni, ami megoldja a „beengedés” korlátozását
- Message Submission Protocol (RFC 6409)
  - Hasonló, mint SMTP, de
  - Eltérő portszám: 587
  - Autentikáció (pl. username, password)
  - További funkciók (pl. FQDN ellenőrzése, szintaxis ellenőrzése, hibák naplózása)
- MTA helyett: MSA – Message Submission Agent

Címek a levélben

- Egy postai levél esetén két helyen van címzés:
  - levélpapíron
  - borítékon
- Elektronikus levélben is így van.
  - A levelező kliensben látható mezők (**`From`**, **`To`**, **`Cc`**, esetleg **`Bcc`**) tartalma a levél törzsében utazik.
  - A kézbesítés pedig az ún. envelope recipient mező alapján történik, illetve hiba esetén az envelope sender alapján talál vissza a feladóhoz a hibaüzenet
- Ezeket a küldő MUA állítja be (konzisztens módon)
  - a feladót a saját beállítása alapján („Saját e-mail cím: ”)
    → **`From`** mező, és envelope sender
  - a címzettet(ek)et a felhasználó rendelkezése alapján
    → **`To`**, **`Cc`**, **`Bcc`** mezők, illetve envelope recipient

Címek megadása

- Hogyan adja meg a címeket a MUA az MTA/MSA-nak?
- A levelező kliensben látható címek megadása
  - A **`DATA`** paranccsal történik
  - Először a fejrész mezők (**`From`**, **`To`**, **`Cc`**, **`Bcc`**, **`Date`**, **`Subject`**), majd egy üres sor után a levél szövege (végül egy sorban csak egy pont zárja le)
- Az envelope sender és envelope recipient megadása
  - A **`MAIL FROM:`** és az **`RCPT TO:`** parancsokkal történik
- Ezeket a MUA természetesen konzisztens módon tölti ki.
  - Szemléltetésként bemutatunk egy példát, ahol nem ☺

(Tanórán élő demonstráció, a következő fóliák csak emlékezetőnek, illetve a „levelező” hallgatók számára.)

Levélküldés példa – 1

```
lencse@dev:~$ telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 dev.tilb.sze.hu ESMTP Postfix (Debian/GNU)
helo localhost
250 dev.tilb.sze.hu
mail from: lencse@dev.tilb.sze.hu
250 2.1.0 Ok
rcpt to: lencse@hit.bme.hu
250 2.1.5 Ok
```

Levélküldés példa – 2

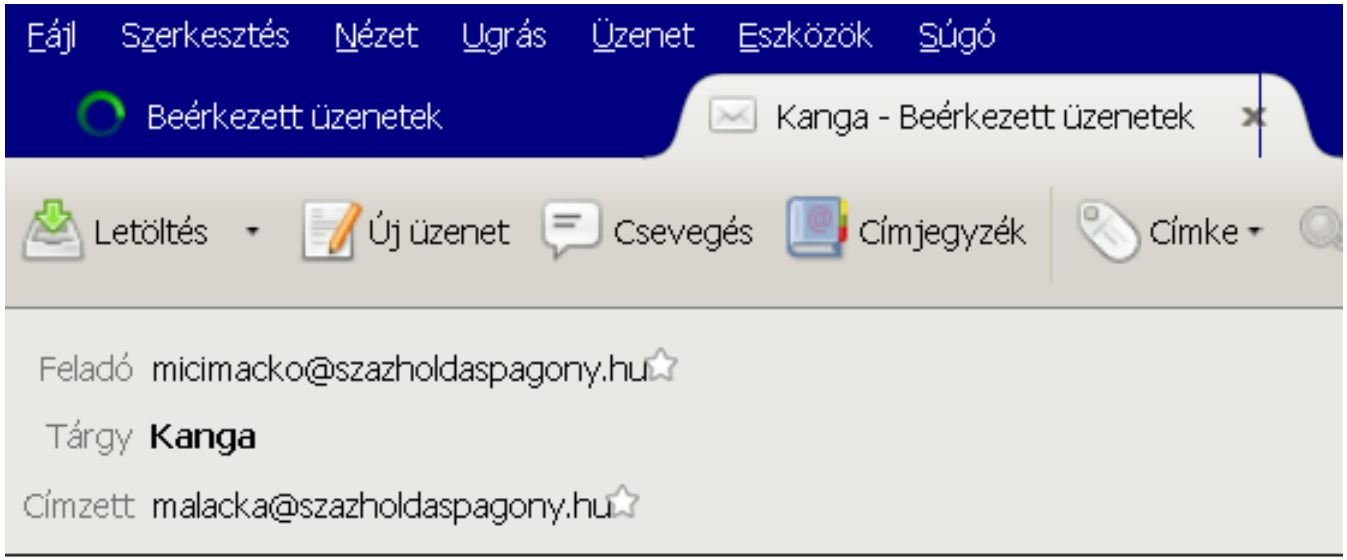
```
data
354 End data with <CR><LF>.<CR><LF>
From: micimacko@szazholdaspagony.hu
To: malacka@szazholdaspagony.hu
Subject: Kanga

Szia Malacka!

Hogy lehet Kanga? Keressuk fel!

Puszi:

Micimacko
.
```

Levélküldés példa – 3

```
250 2.0.0 Ok: queued as 255B41B210F
quit
221 2.0.0 Bye
Connection closed by foreign host.
lencse@dev:~$
```

Levélküldés példa – 4

A levelező kliens programban ez látható:




```
Szia Malacka!

Hogy lehet Kanga? Keressuk fel!

Puszi:

Micimacko
```

Levelek letöltése POP3-mal

- POP3: Post Office Protocol version 3
  - Postafiók távoli elérésére használható
  - Szabványos portszáma: 110
- A POP3 protokoll legfontosabb parancsai
  - USER username – felhasználó nevének megadása
  - PASS password – jelszó megadása (nyílt szövegként!)
  - STAT – lekérdezi a levelek számát és összesített méretét
  - LIST – levelek lekérdezése (sorszám + méret)
  - RETR n – az n. levél letöltése
  - DELE n – az n. levél kijelölése törlésre
  - RSET – törlésre való kijelölés(ek) megszüntetése
  - QUIT – postafiók aktualizálása (törlések véglegesítése) és kilépés

  (Tanórán élő demonstráció, fóliákon emlékeztető.)

Levél letöltés példa – 1

```
lencse@dev:~$ telnet localhost 110
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
+OK Hello there.
user lencse
+OK Password required.
pass ********
+OK logged in.
stat
+OK 4 2160
```

Levél letöltés példa – 2

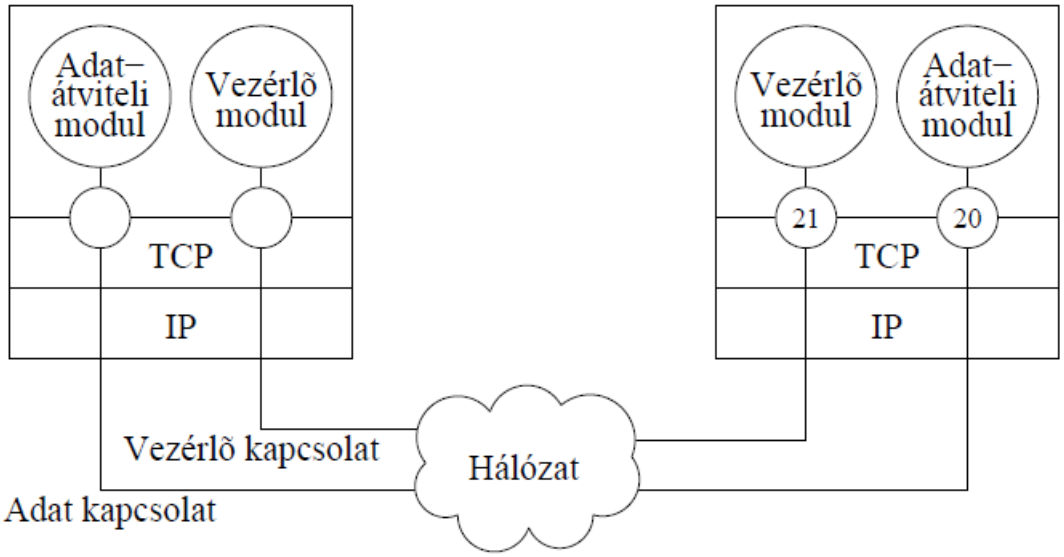
```
list
+OK POP3 clients that break here, they violate
  STD53.
1 423
2 442
3 665
4 630
.
```

Levél letöltés példa – 3

```
retr 1
+OK 423 octets follow.
Return-Path: <root@dev.tilb.sze.hu>
X-Original-To: lencse
Delivered-To: lencse@dev.tilb.sze.hu
Received: by dev.tilb.sze.hu (Postfix, from userid 0)
        id 66A53171D7E; Fri, 15 Sep 2006 10:59:11 +0200 (CEST)
To: viktor@dev.tilb.sze.hu
Subject: proba
Cc: lencse@dev.tilb.sze.hu
Message-Id: <20060915085911.66A53171D7E@dev.tilb.sze.hu>
Date: Fri, 15 Sep 2006 10:59:11 +0200 (CEST)
From: root@dev.tilb.sze.hu (root)


.
dele 1
+OK Deleted.
```

Levél letöltés példa – 4

```
list
+OK POP3 clients that break here, they violate STD53.
2 442
3 665
4 630
.
rset
+OK Resurrected.
list
+OK POP3 clients that break here, they violate STD53.
1 423
2 442
3 665
4 630
.
quit
+OK Bye-bye.
Connection closed by foreign host.
```



Lencse Gábor: Hálózati alkalmazások (elektronikus jegyzet)

FILE TRANSFER PROTOCOL

FTP kliens-szerver kommunikációja

- Az FTP szerver a 21-es porton várja a kliens csatlakozását
- Csatlakozáskor létrejön a **vezérlő kapcsolat** (TCP)
  - A kliens parancsokat ad ki rajta, a szerver (röviden) válaszol
- Fájlok, könyvtárlisták átviteléhez: **adat kapcsolat** (TCP)
  - Szerver oldalon a 20-as portot használja
  - Szükség esetén létrejön, majd lebomlik
  - Létrehozásának iránya kétféle lehet
    - Aktív mód esetén: szervertől a kliens felé
    - Passzív mód esetén: klienstől a szerver felé
  - Kliens oldali tűzfal vagy privát IP-cím+NAT esetén passzív módot használunk. Miért? Mert aktív módban:
    - A tűzfal nem engedné meg a kapcsolódást befele
    - Privát IP-cím alapján nem jönne meg a SYN
      - De létezik protocol helper! ☺

FTP kliens-szerver parancsok

- Néhány alapvető kliens-szerver parancs
  - USER username – felhasználói név megadása
  - PASS password – jelszó megadása
  - PORT h1,h2,h3,h4,p1,p2 – IP-cím és portszám megadása a szerver által kezdeményezett adatkapcsolathoz (aktív mód)
  - PASV – IP-cím és porszám kérése a szervertől, ennek alapján építi fel az adatkapcsolatot a kliens (passzív mód)
  - LIST [directory] – könyvtár tartalmának listázása
  - RETR filename – fájl letöltése
  - STOR filename – fájlt feltöltése
  - HELP [command] – argumentum nélkül parancslistát ad
  - SYST – információk a szerverről
  - QUIT – kilépés

Telnettel FTP-zünk – 1

- Kliens-szerver kommunikáció demonstrációja, melyben
  - Csatlakozunk,
  - Authentikálunk
  - Lekérjük a szerver típusát
  - Könyvtárat váltunk, majd lekérdezzük, hol vagyunk
  - Passzív módhoz portot kérünk
  - Közben egy MÁSIK terminálról csatlakozunk
  - Lekérünk egy könyvtárlistát (a másik terminálon lejön)
  - Passzív módhoz portot kérünk ismét
  - A másik terminálról ismét csatlakozunk
  - Lekérünk egy ASCII fájlt
  - Átnevezünk egy fájlt (két parancs kell!)
  - Letörlünk egy másikat
  - Kilépünk

Telnettel FTP-zünk – 2

```
lencse@dev:~$ telnet localhost 21
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 ProFTPD 1.3.3a Server (Debian) [::ffff:127.0.0.1]
user lencse
331 Password required for lencse
pass ********
230-Welcome, archive user lencse@localhost !
230-
230-The local time is: Tue May 13 21:14:21 2014
230-
230-This is an experimental FTP server.  If you have
   any unusual problems,
230-please report them via e-mail to <root@localhost>.
230-
230 User lencse logged in
```

Telnettel FTP-zünk – 3

```
syst
215 UNIX Type: L8
cwd ftptest
250 CWD command successful
pwd
257 "/home/lencse/ftptest" is the current
  directory
pasv
227 Entering Passive Mode (127,0,0,1,139,40).
list
150 Opening ASCII mode data connection for
  file list
226 Transfer complete
```

Telnettel FTP-zünk – másik terminal

```
lencse@dev:~$ telnet localhost $((139*256+40))
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
-rw-r--r--   1 lencse   lencse          0 May 13 19:06 atnevezendo
-rw-r--r--   1 lencse   lencse          0 May 13 19:07 torlendo
-rw-r--r--   1 lencse   lencse         84 May 13 19:05 Unixos-szoveg.txt
Connection closed by foreign host.
```

(A könyvtárlista betűméretét csökkentettük.)

Telnettel FTP-zünk – 4

```
pasv
227 Entering Passive Mode (127,0,0,1,179,154).
retr Unixos-szoveg.txt
150 Opening ASCII mode data connection for Unixos-
  szoveg.txt (84 bytes)
226 Transfer complete
rnfr atnevezendo
350 File or directory exists, ready for
  destination name
rnto atnevezett
250 Rename successful
dele torlendo
250 DELE command successful
quit
221 Goodbye.
Connection closed by foreign host.
```

Telnettel FTP-zünk – másik terminal

```
lencse@dev:~$ telnet localhost $((179*256+154))
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Ez egy szoveges fajl unixos sorveg jelekkel.
Masodik sor.
Harmadik sor.
Utolso sor.
Connection closed by foreign host.
```

Konverzió az átvitel során

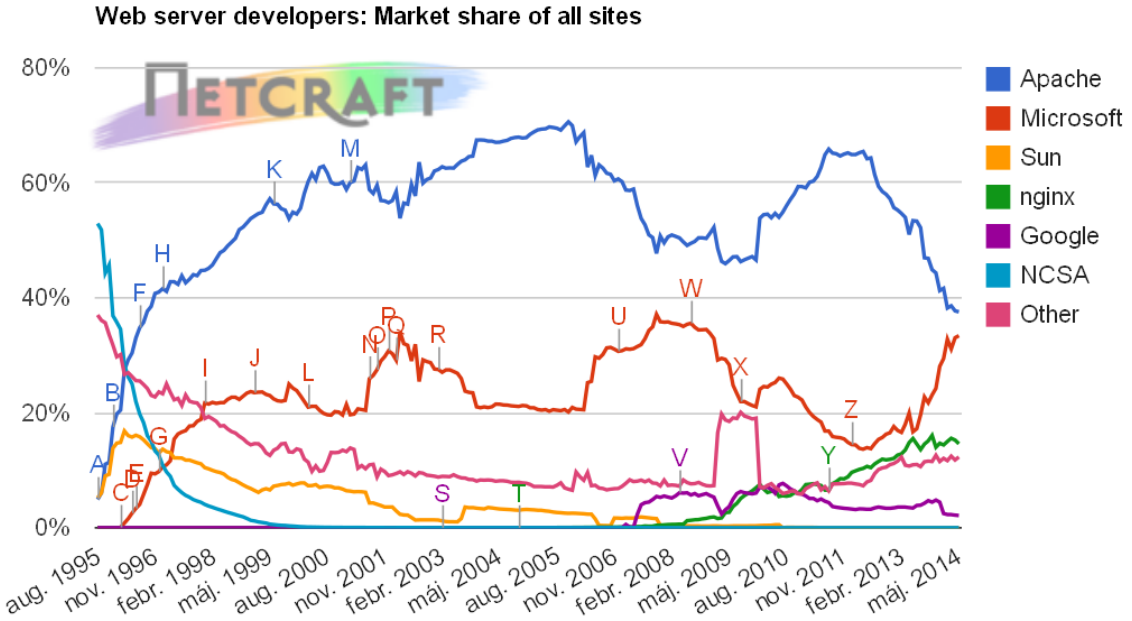
- Emlékeztető: az OSI referenciamodellben van megjelenítései réteg: adatábrázolással foglalkozik.
- Ez a TCP/IP modellből hiányzik
- ASCII szövegben a sorvégek jelzése
  - Unix/Linux alatt: LF, azaz 10-es kódú karakter
  - DOS/Windows alatt: CR-LF, azaz 13-as és 10-es kódú karakterek
- ASCII szövegek átvitelekor konverzió szükséges:
  - Linux → Windows: minden LF elé be kell szúrni egy CR-t
  - Windows → Linux: minden CR-LF párból csak az LF marad
- Egyéb fájlok esetén TILOS ilyen konverziót végezni!
- A konverzió létét a felhasználó beállíthatja:
  - ASCII mód: van konverzió; bináris mód: nincs konverzió

Karakteres FTP-kliens parancsok

- Különböző platformokra számos FTP kliens létezik
- Általában elérhető a hagyományos karakteres kliens is
- Ennek néhány alapvető parancsa
  - get filename – fájl letöltése
  - put filename – fájl feltöltése
  - ascii – ASCII módra váltás
  - bin – bináris módra váltás
  - ls / dir – könyvtár listázása
  - cd – könyvtárváltás
  - quit / bye – kilépés

Anonymous FTP

- A Word Wide Web megjelenése előtt a fájlmegosztás alapvető módszere volt az anonymous FTP szerver
- Bárki be tud lépni, mert
  - a felhasználói név: Anonymous
  - a jelszó az e-mail címünk, de legalább egy @ karakter
    (20 éve még volt bizalmunk megadni.)
- Tipikus könyvtárak
  - pub – itt találhatók a nyilvános anyagok
  - incoming – ide lehet feltölteni (ha engedélyezett)
- A böngészők ma is használják az anonymous FTP-t, ha az URL-ben nem http-t, hanem ftp-t adunk meg



http://news.netcraft.com/archives/2014/

VIRTUÁLIS WEBSZERVEREK

Virtuális webszerverek problémája

- Webtárhely-szolgáltatók szeretnék egy szerveren több ügyfél weblapját kiszolgálni.
- Nem igazán jó megoldások:
  - Külön könyvtárba tenni, könyvtár neve az URL-ben:
    - **`www.webhostingkft.hu/bogyoesbaboca`**
    - **`www.webhostingkft.hu/micimacko`**

    NEM jó, mert a felhasználók saját domaint szeretnének, ilyeneket:
    - **`www.bogyoesbaboca.hu`**
    - **`www.micimacko.hu`**
  - Minden weblapnak külön IP-cím, ugyanahhoz az interfészhez mindet hozzárendelni

    NEM jó, mert sok IP-cím kell, ami IPv4-ben szűkös erőforrás!

Virtuális webszerverek alapötlete

- Mi a probléma?
  - Szimbolikus név → IP-cím leképzésnél a hostnév elveszik
- Megoldás: HTTP/1.0 helyett HTTP/1.1
  - A hostnév elküldhető (és sok egyéb is)
- Összes ügyfél esetén CNAME rekord, például:

```
www.bogyoesbaboca.hu CNAME www.webhostingkft.hu
www.micimacko.hu CNAME www.webhostingkft.hu
```

- A **`GET`** parancs után a **`Host:`** adja meg a hostnevet, pl.:

```
GET / HTTP/1.1
Host: www.bogyoesbaboca.hu

GET / HTTP/1.1
Host: www.micimacko.hu
```

Virtuális webszerverek megvalósítása

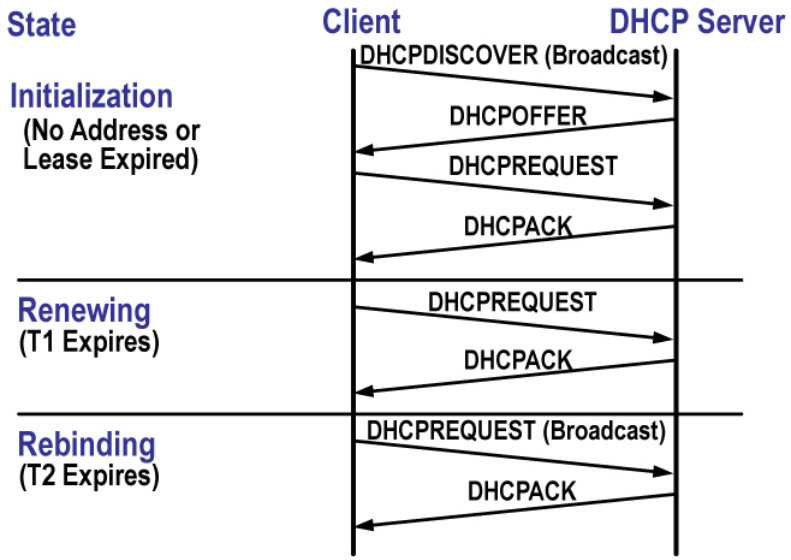
- Az egyes webszerver implementációktól függő módon be kell állítani, hogy melyik virtuális webszerverhez milyen tartalmat rendelünk.

  Például Apache2 esetén:

```
cat /etc/apache2/sites-available/bogyoesbaboca
<VirtualHost *:80>
        Servername www.bogyoesbaboca.hu
        DocumentRoot /var/www/bogyoesbaboca
</VirtualHost>
```

  A további beállításokat nem részletezzük, és a fentieket sem kell megtanulni.

  (A működés élő demonstrációja kizárólag a jelenlevőknek.)



Forrás: http://wiki.hill.com/wiki/index.php?title=Image:DHCP.jpg

DHCP MŰKÖDÉSE

A DHCP kommunikációja

- A kliens és a szerver *DHCP üzenetek*kel kommunikálnak
- A DHCP üzenetek BOOTP üzenetekben opcióként jelennek meg
- A BOOTP üzenetek IP fölött, UDP-be ágyazva haladnak
  - Amíg a kliensnek nincs érvényes IP címe, addig 0.0.0.0-t használ
  - Broadcast esetén IP szinten természetesen 255.255.255.255 címre küldi az üzenetet (Ethernet szinten pedig FF:FF:FF:FF:FF:FF-re)
  - UDP-ben a kliens portszáma: 68, a szerveré: 67
- A továbbiakban a DHCP üzenetek neve mellett feltüntetjük, hogy ki küldi kinek: „küldő → címzett” formában. Jelölések:
  - K: kliens
  - S: szerver
  - B: broadcast (IP és Ethernet szinten is)

A DHCP üzenetei (ismétlés)

- Cím beszerzése, ha minden jól megy:
  - DHCPDISCOVER K→B
  - DHCPOFFER S→K
  - DHCPREQUEST K→B
  - DHCPACK S→K
- De előfordulhatnak ilyenek is:
  - DHCPNAK S→K
  - DHCPDECLINE K→S
- Megújításnál csak:
  - DHCPREQUEST K→S
  - DHCPACK S→K
- Idő előtt visszaadás:
  - DHCPRELEASE K→S

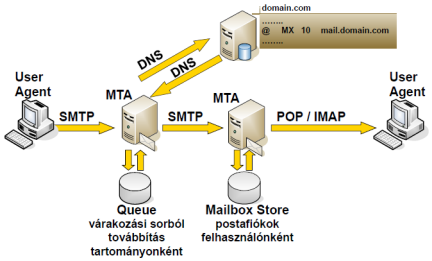
A DHCP üzenetei (demonstráció)

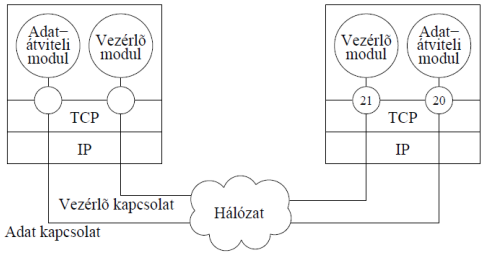
- Figyeljük meg az IP-cím beszerzésének 4 lépését Wireshark segítségével a **`DHCP.pcap`** fájl alapján
  - Ellenőrizzük a MAC- és IP-címeket, UDP portszámokat
- Nézzük meg az IP-cím megújítását a **`DHCP-renew.pcap`** fájl alapján

(Előadáson közös tanulmányozás, otthon önállóan folytatható.)

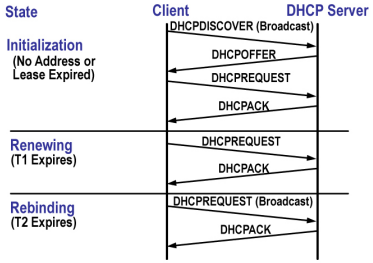
Összefoglalás

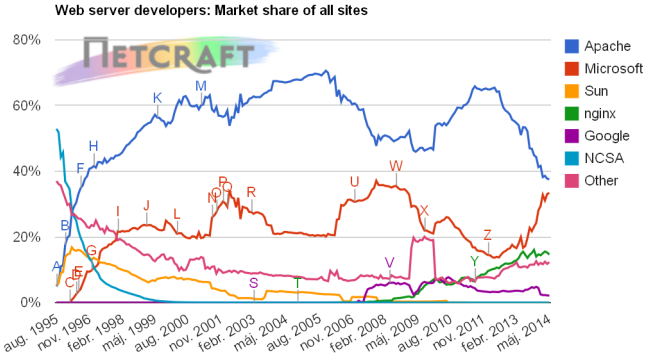
- Levelezés gyakorlati kérdései
  - Message Submission Protocol
  - Látható/envelope sender/recipient
  - POP3
- FTP működése





- Virtuális webszerverek
- DHCP működése



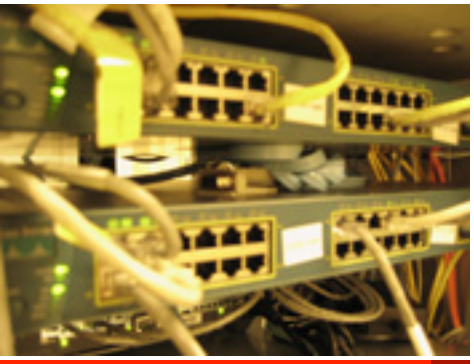


Kérdések?

KÖSZÖNÖM A FIGYELMET!

Hálózati Rendszerek és Szolgáltatások Tanszék

Dr. Lencse Gábor
tudományos főmunkatárs
BME Hálózati Rendszerek és Szolgáltatások Tanszék
lencse@hit.bme.hu



Introduction

Security

Mark Felegyhazi
assistant professor
CrySyS Lab.

BME Department of Networked Systems and Services
(Hálózati Rendszerek és Szolgáltatások Tanszék)
mfelegyhazi(atat)crysys(dot)hu

First look – quick overview

- What is security?
- Prevent or detect/block?
- Security techniques
  - crypto
  - protocols
- Why do we care about economics here?
  - How much is enough to invest in defense?
  - We work together, or not?
- Who is responsible?

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

- Concepts and definitions
- Attacks
- Cryptography
- Security defense mechanisms
- Security economics

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

biztonság

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

biztonság

safety

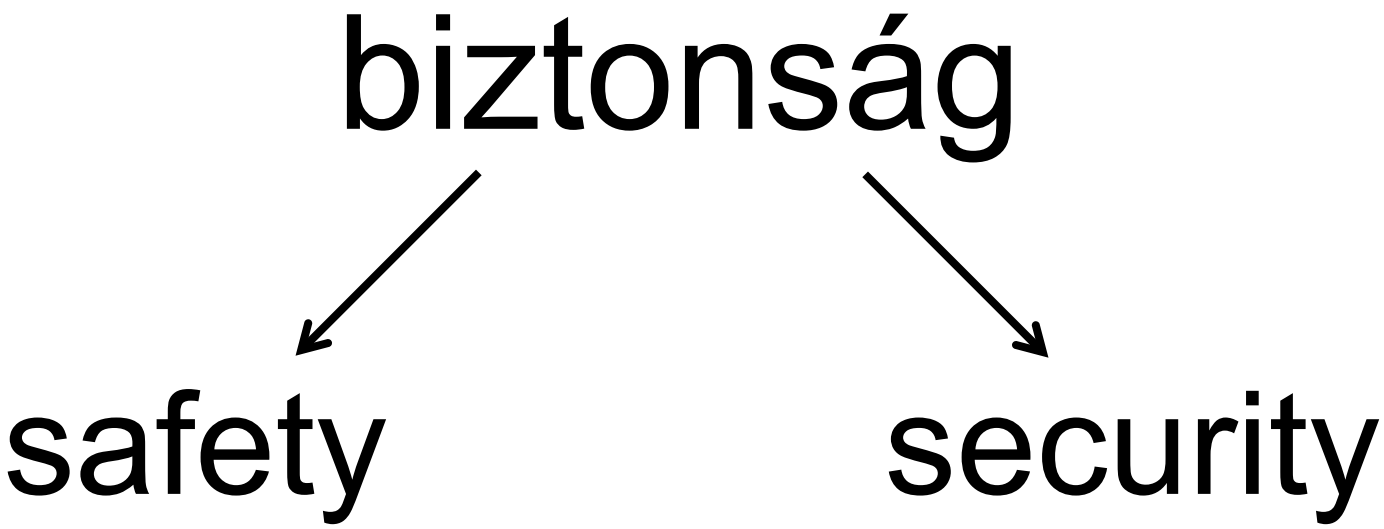
the condition of being protected against the consequences of failure, damage, error, accidents, harm or any other event, which could be considered non-desirable

security

the **process** of delaying, preventing, and otherwise protecting against external or internal defects, dangers, loss, criminals, and other individuals or actions that threaten the steady state of a system

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

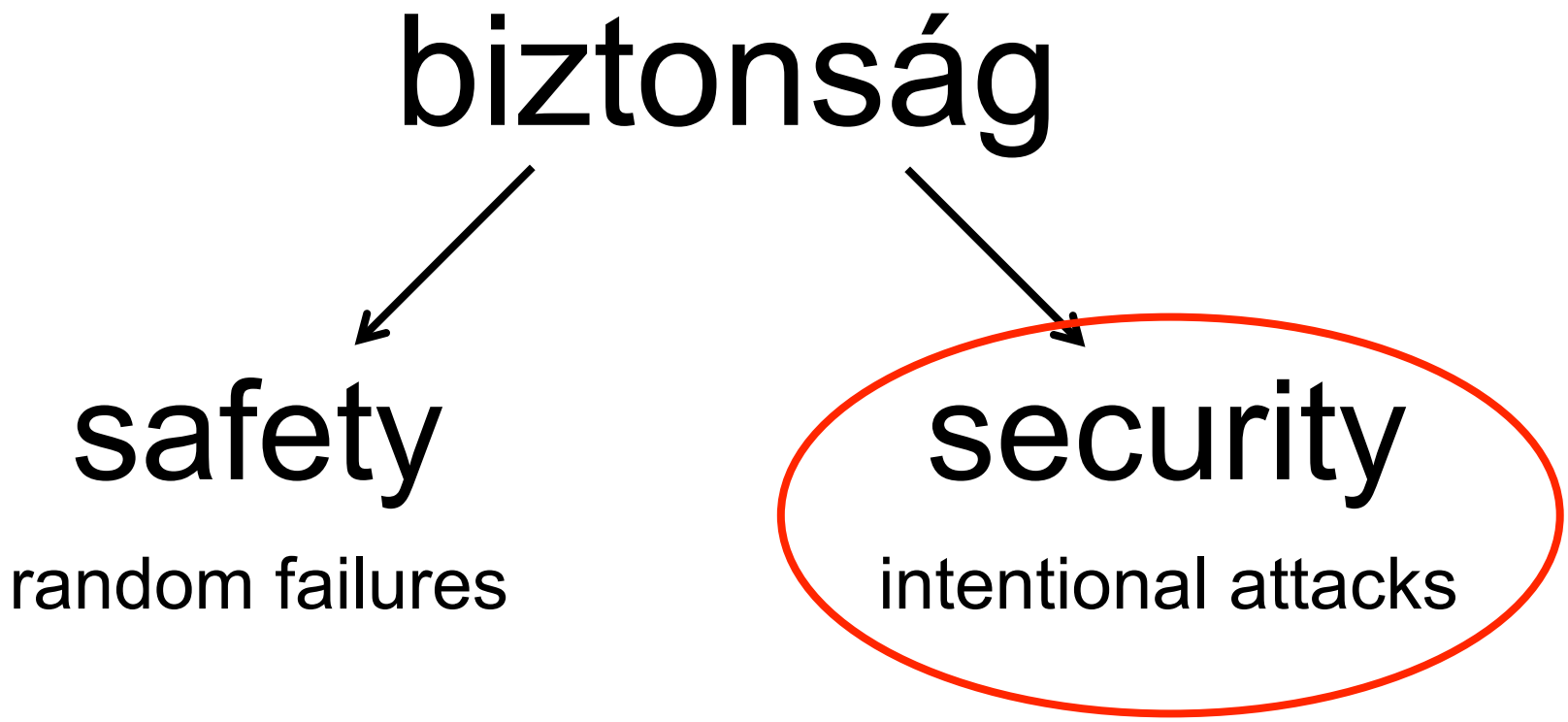
Concepts



achieve an acceptable level of risk

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

megbízhatóság

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

megbízható

reliable

able to perform and maintain its functions in routine, as well as in unexpected circumstances

trustworthy

does what people expect it to do – and not something else – despite environmental disruption, human user and operator errors, and attacks by hostile parties

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Concepts

megbízható

reliable

random failures

trustworthy

+ intentional attacks

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

adatbiztonság

vs.

adatvédelem

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Concepts

adatbiztonság

↓

data **security**

the practice of **protecting** information from unauthorized access, use, disclosure, disruption, modification, or destruction

adatvédelem

↓

data **privacy**

the ability to **control** what information one reveals about oneself, and who can access that information

control can be preserved by protection against disclosure

↓

data security mechanisms can be used to achieve data privacy

but, in general, privacy is a broader (social) concept

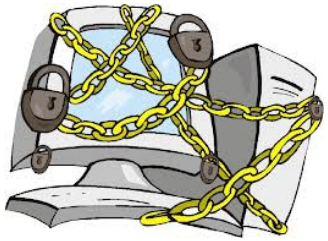
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Concepts

information security?

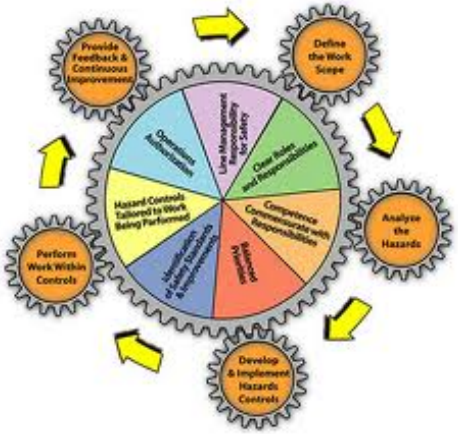
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

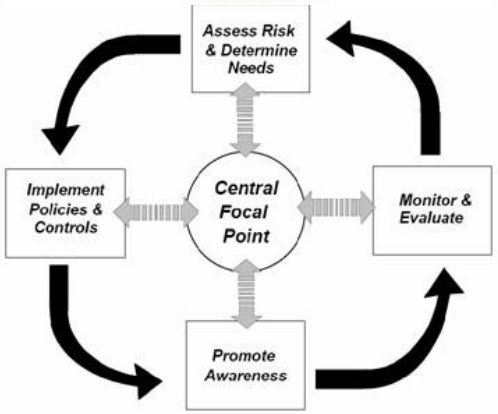
... is about protecting data (transmitted and stored)



... is protecting data processing systems (computers and networks)

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME





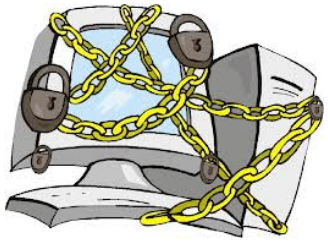
... is a process (aiming at reducing risk)

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

= cryptography

κρυπτός = "hidden, secret";
γράφειν = "writing"

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME



= computer and network security

user authentication (e.g., passwords)
protection against malware (e.g., virus scanners)
vulnerablity detection and patching
network perimeter defense (e.g., firewalls)
network intrusion detection systems (IDS)
DoS resistance
...

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Security

- Merriam-Webster, [4b, (1)]:
  - measures taken to guard against espionage or sabotage, crime, attack, or escape
- Dictionary.com, [1]:
  - freedom from danger, risk, etc.; safety
- Wikipedia.com:
  - **Security** is the degree of protection against danger, damage, loss, and crime.
  - **Information security** means protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.
  - **Communications security** is the discipline of preventing unauthorized interceptors from accessing telecommunications in an intelligible form, while still delivering content to the intended recipients.
  - **Computer security** can focus on ensuring the availability and correct operation of a computer system without concern for the information stored or processed by the computer.

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Our definition

- **security**: prevention or detection of an attack on the computer system
  - attack: deliberate attempt to compromise the intended use of a computer system

- a few important points
  - attacker: a malicious entity whose aim is to prevent the users of the computing system from achieving their goal (primarily privacy, integrity, and availability of data)
  - **security vs. safety**
- Why economics of computer security?
  - strategic adversary: rational, profit-seeking
  - in general: see “Security Protocols” course from Prof. Levente Buttyan (Hírközlõ rendszerek biztonsága szakirány)

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Secure protocols

- in a very general sense, secure protocols are distributed algorithms – involving message passing between participants – that try to reach a certain goal, even in the presence of attackers
- examples:
  - secure communication protocols (for wired and wireless networks)
  - secure key exchange protocols
  - secure routing protocols
  - secure neighbor discovery protocols (in wireless networks)
  - …
- security of a protocol is always evaluated w.r.t. an attacker model
- different types of protocols call for different attacker models

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

More definitions

- vulnerability
  - attacks usually exploit vulnerabilities
  - a vulnerability is a flaw or weakness in the system's design, implementation, or operation and management
  - most systems have vulnerabilities, but not every vulnerability is exploited
  - whether a vulnerability is likely to be exploited depends on the difficulty of the attack and the perceived benefit of the attacker

- threat
  - a possible way to exploit vulnerabilities
  - a potential attack

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

More on attacks

- passive attack
  - requires no intervention into the operation of the system
  - typically consists in the passive acquisition of some information that should not be available to the attacker
  - typical examples:
    - eavesdropping message contents
    - traffic analysis
      - gaining knowledge of data by observing the characteristics of communications that carry the data
      - even if message content is encrypted, an attacker can still
        - » determine the identity and the location of the communicating parties
        - » observe the frequency and length of the messages being exchanged
        - » guess the nature of the communication
  - difficult to detect, should be prevented

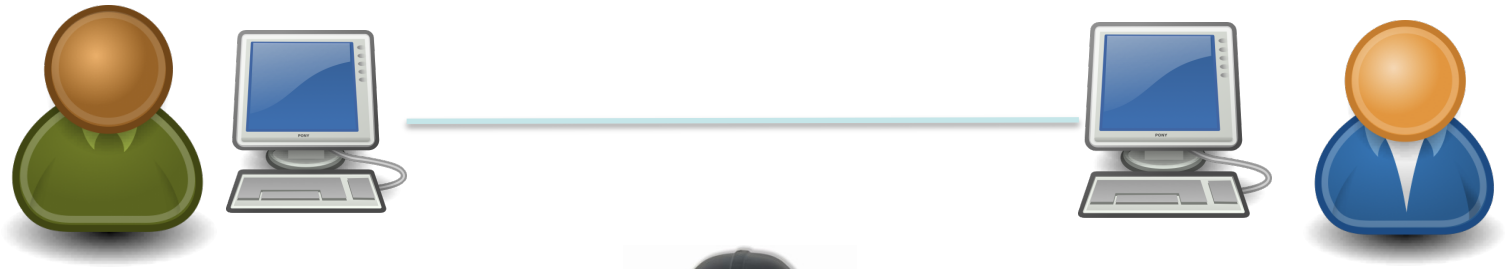
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

More on attacks

- active attack
  - requires an active intervention into the operation of the system
  - typical examples:
    - masquerade (spoofing)
      - an entity pretends to be a different entity
    - replay
      - capture and subsequent retransmission of data
    - modification (substitution, insertion, destruction)
      - (some parts of the) legitimate messages are altered or deleted, or fake messages are generated
      - if done in real time, then it needs a “man in the middle”
    - denial of service
      - normal use or management of the system is prevented or inhibited
      - e.g., a server is flooded by fake requests so that it cannot reply normal requests
  - difficult to prevent, should be detected

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Communication security – a simple view



Alice

Malice

Bob

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

(Bob is not) living next door to Alice

- the motivation, operation, and analysis of security protocols are often presented as tales about two strange characters, Alice and Bob, and their “friends”
- Alice and Bob
  - they live far from each other and communicate only via Internet, e-mail, or telephone
  - they have actually never met, but for some reason, they frequently need to conduct all sorts of business with each other
  - they rarely trust anyone else, sometimes not even each other
  - their history of interactions include exchanging secret e-mails, playing poker over the phone, using electronic coins to buy digital content from each other, remotely signing contracts, running auctions and elections over the Internet, …

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Friends

- Carol / Carlos / Charlie is a third participant in communications
- **Eve** is an eavesdropper (a passive attacker)
- Gordon is a government agent
- Isaac is an Internet Service Provider (ISP)
- Justin / Julian is from the justice system
- **Mallory** is a malicious attacker; unlike Eve, Mallory can modify messages, substitute her own messages, replay old messages, and so on (active attacker)
- Oscar is an opponent, usually taken as equivalent to Mallory
- Pat / Peggy is a prover and Victor is a verifier; in their interactions, Peggy always tries to convince Victor that she knows some information without actually revealing that information (zero-knowledge protocols)
- Trent is a trusted arbitrator, some kind of neutral third party, whose exact role varies with the protocol under discussion
- Trudy, is an intruder; another alternative to Mallory
- Zoe, often the last party to be involved in a cryptographic protocol

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Communication security services

CIA principles

- confidentiality
  - protection of information from unauthorized disclosure
  - information can be
    - content of communications → (content) confidentiality
    - meta-information (derived from observation of traffic flows) → traffic flow confidentiality
- integrity protection
  - aims to detect message modification and replay
  - provides assurance that data received are exactly as sent by the sender
    - in case of a stream of messages (connection oriented model), integrity means that messages are received as sent, with no duplication, modification, insertion, deletion, reordering, or replays
- availability
  - the service is reachable for the users

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Communication security services

- authentication
  - aims to detect masquerade (spoofing)
  - provides assurance that a communicating entity is the one that it claims to be
    - peer entity authentication
    - data/message origin authentication
- non-repudiation
  - provides protection against denial by one entity involved in a communication of having participated in all or part of the communication
    - non-repudiation of message origin
    - non-repudiation of message delivery

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Placement of security services

- some services can more naturally be implemented at the application layer (e.g., non-repudiation)
- some services better fit in the link layer (e.g., traffic flow confidentiality)
- but many services can be provided at any layer (e.g., authentication, confidentiality, integrity)
  - lower layer (e.g., link-by-link encryption):
    - services are generic, can be used by many applications
    - protection mechanisms are transparent to the user
  - higher layer (e.g., end-to-end authentication):
    - services are more application specific
    - more user awareness

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Attacks

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Examples of attacks

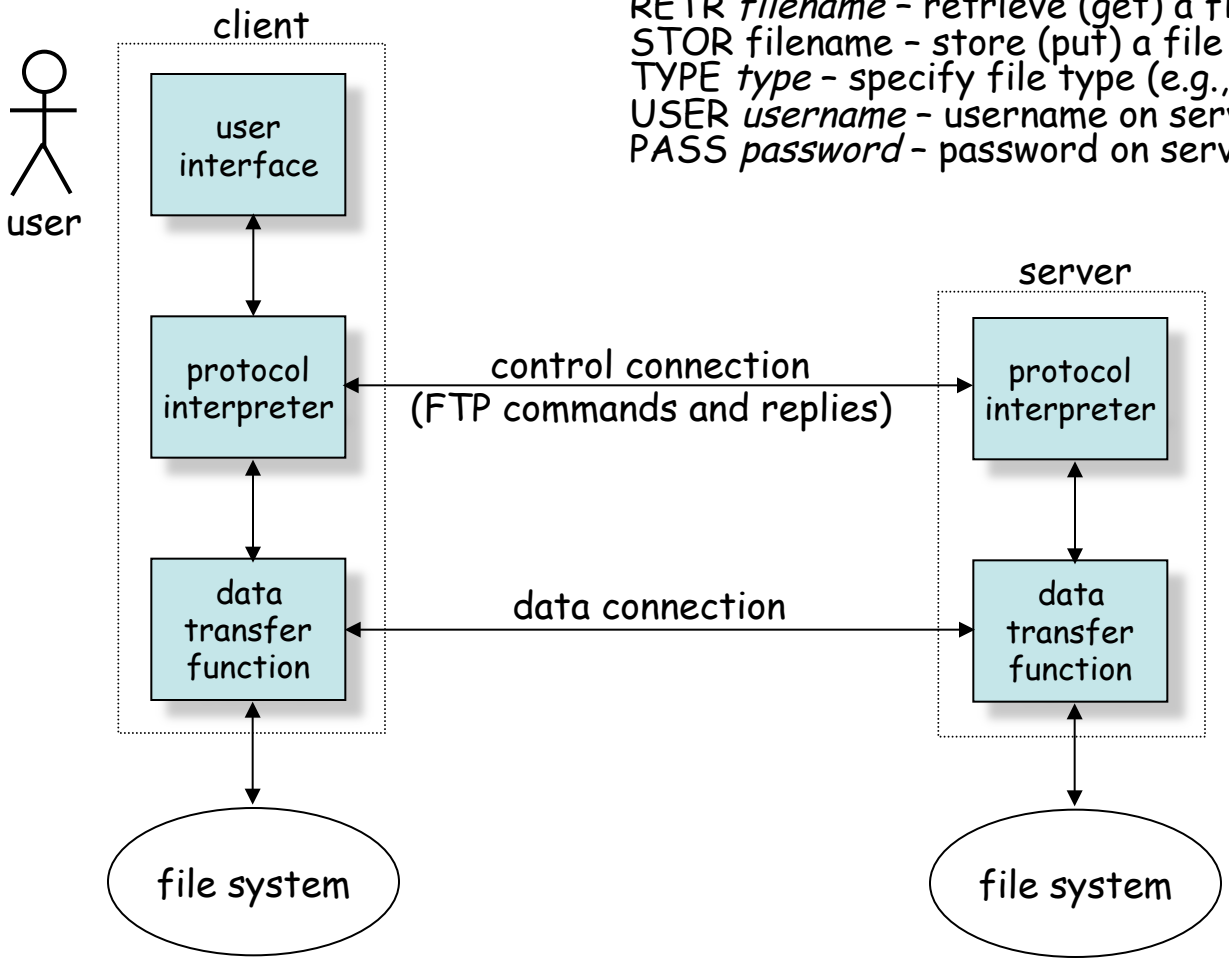
- password sniffing in FTP
- password sniffing in TELNET
- mail forging with SMTP
- ARP spoofing
- DoS against a web server
- spam

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

FTP – File Transfer Protocol

typical FTP commands:

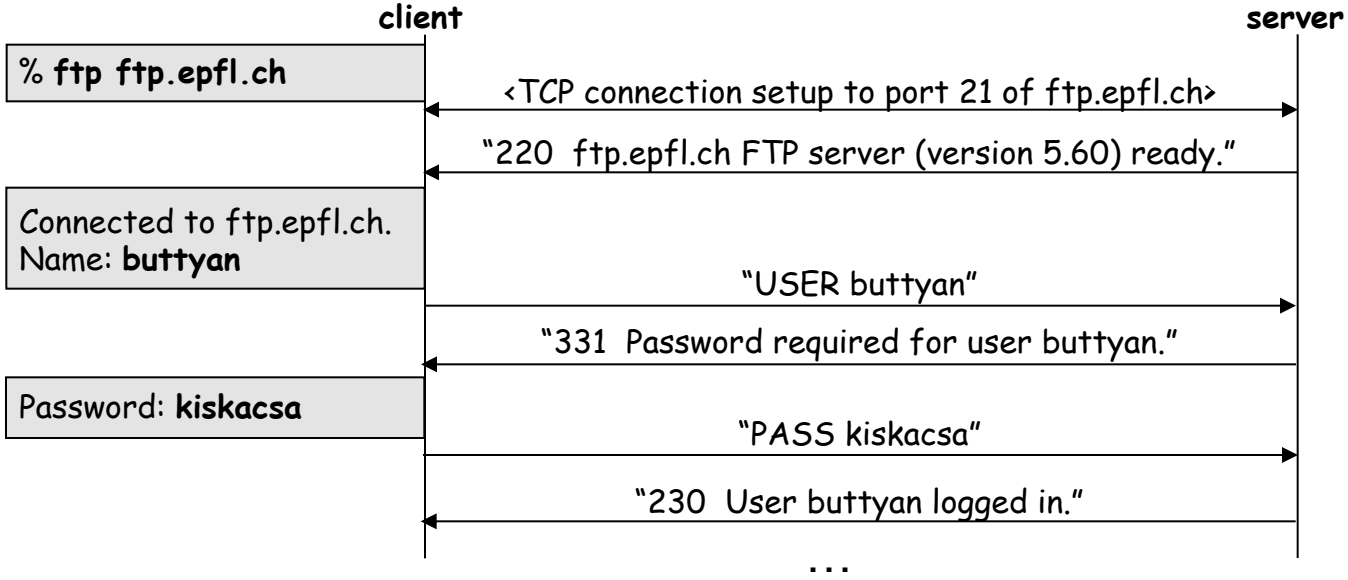
RETR *filename* – retrieve (get) a file from the server
STOR filename – store (put) a file on the server
TYPE *type* – specify file type (e.g., A for ASCII)
USER *username* – username on server
PASS *password* – password on server



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

FTP security problems

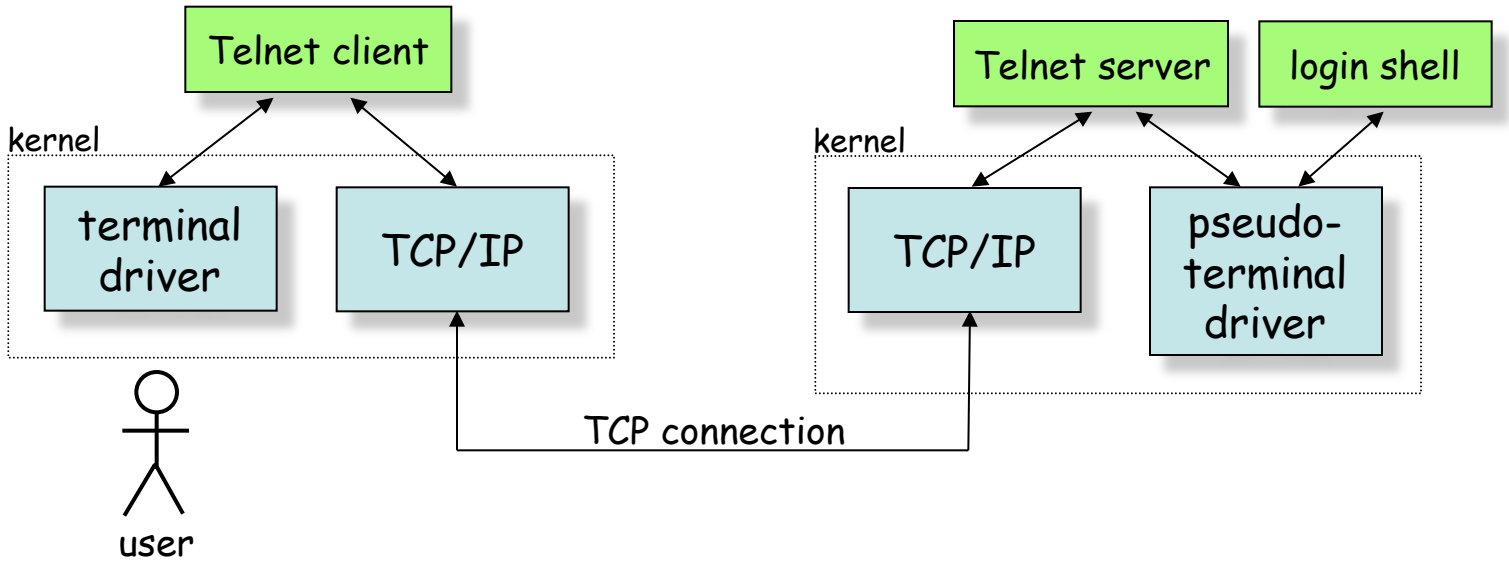
- neither the control nor the data connection is protected
  - passwords can be eavesdropped
    - FTP is a text(ASCII) based protocol, which makes password sniffing even easier
  - files transmitted over the data connection can be intercepted and modified



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Telnet

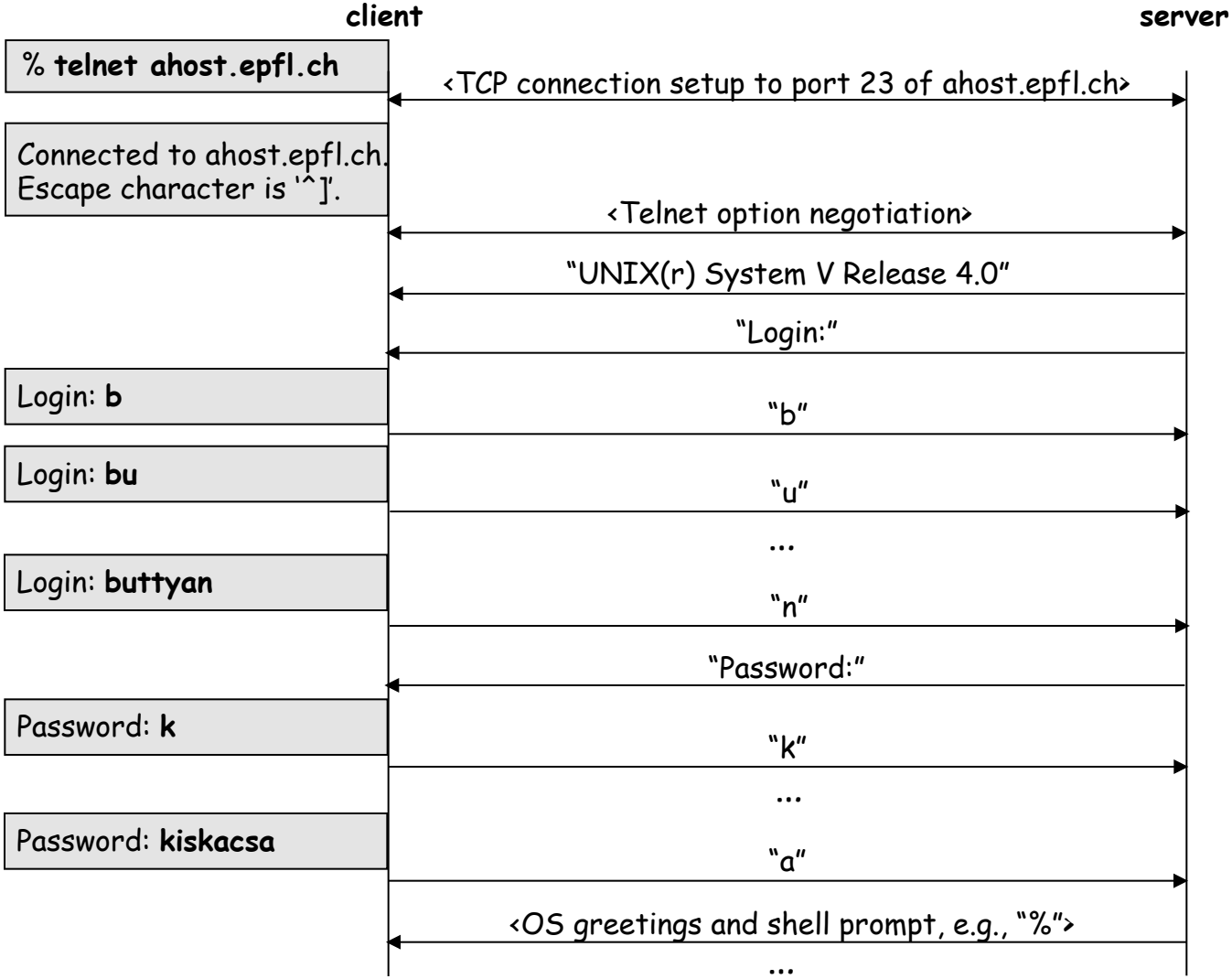
- provides *remote login* service to users
- text (ASCII) based protocol



© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

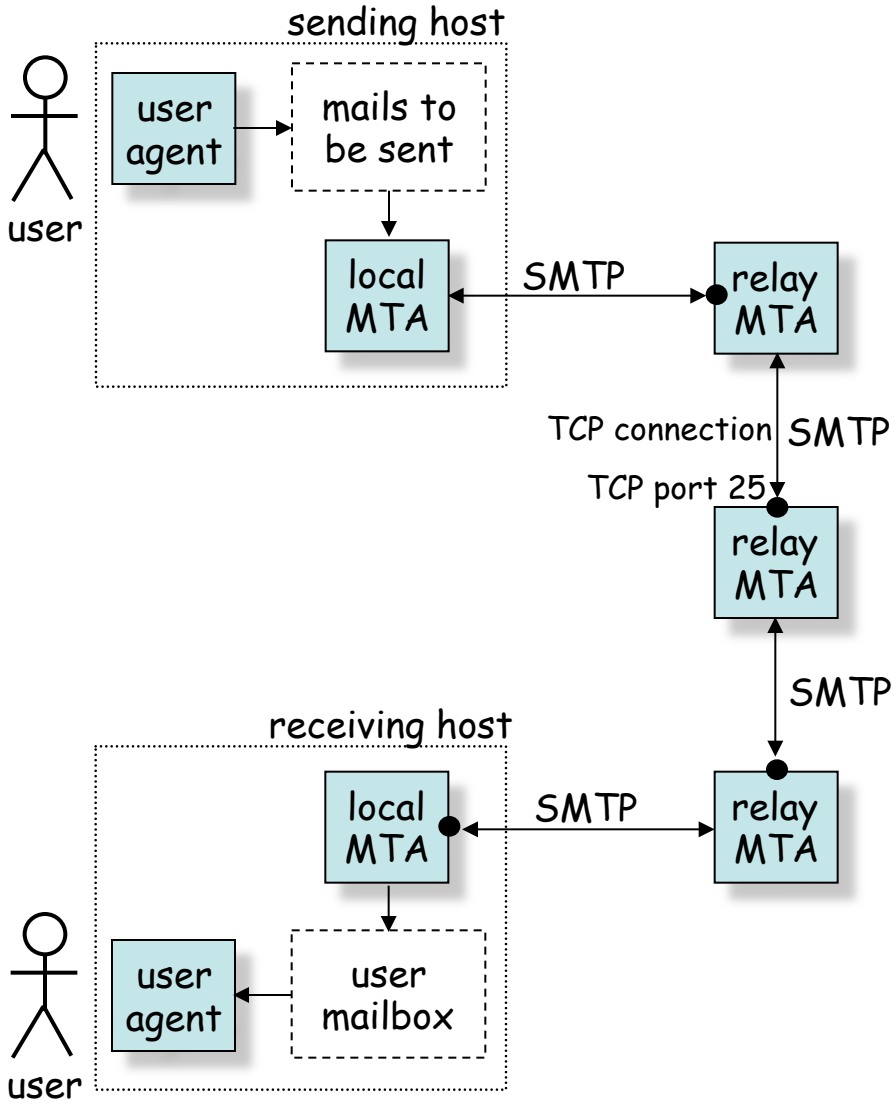
Telnet security problems

- passwords are sent in clear



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

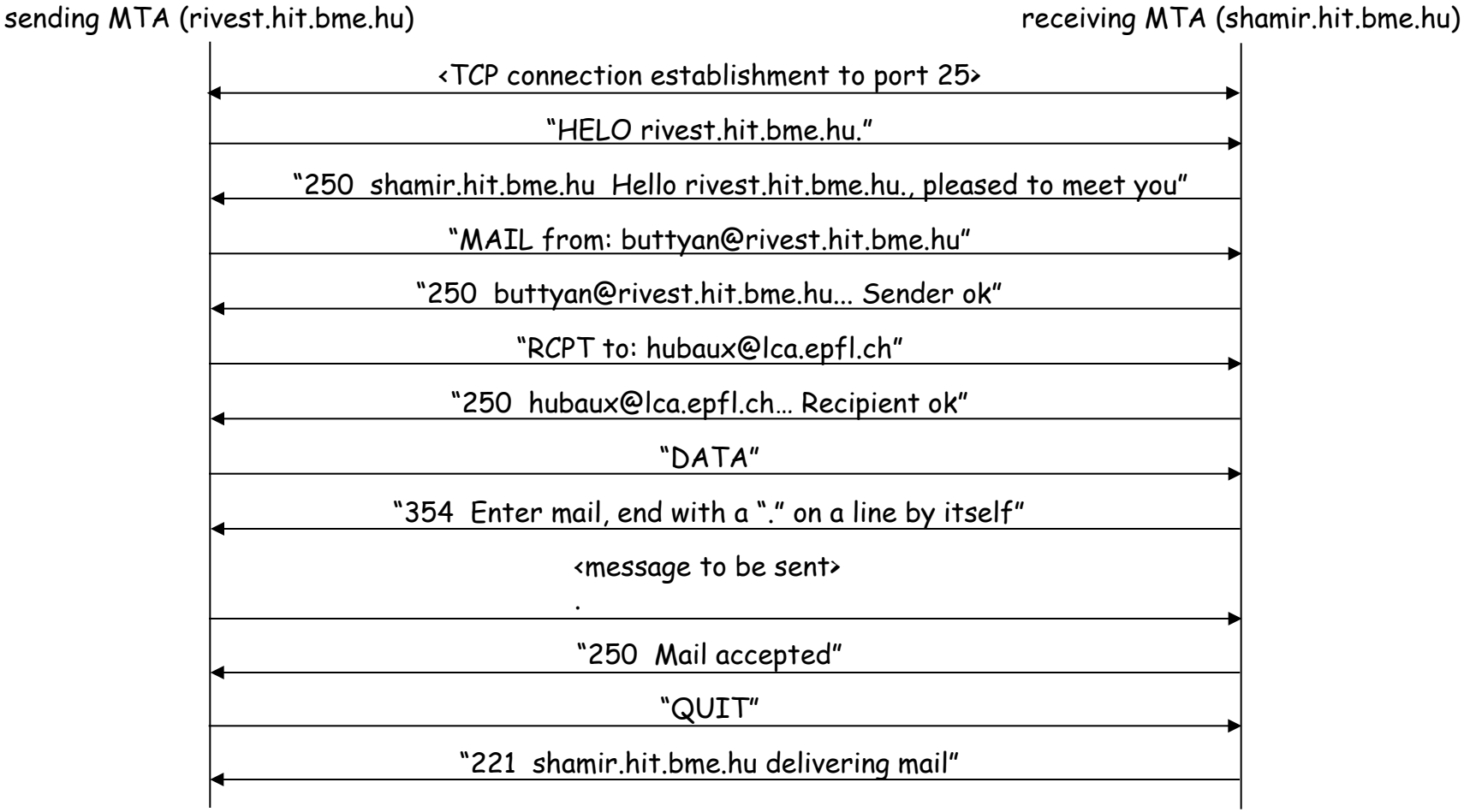
SMTP – Simple Mail Transfer Protocol



© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

SMTP cont'd

- SMTP is used by MTAs to talk to each other
- SMTP is a text (ASCII) based protocol



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

SMTP security problems

- SMTP does not provide any protection of e-mail messages
  - messages can be read and modified by any of the MTAs involved
  - fake messages can easily be generated (e-mail forgery)
- Example:

```
% telnet frogstar.hit.bme.hu 25
Trying...
Connected to frogstar.hit.bme.hu.
Escape character is '^['.
220 frogstar.hit.bme.hu ESMTP Sendmail 8.11.6/8.11.6;
Mon, 10 Feb 2003 14:23:21 +0100
helo abcd.bme.hu
250 frogstar.hit.bme.hu Hello [152.66.249.32], pleased to meet you
mail from: bill.gates@microsoft.com
250 2.1.0 bill.gates@microsoft.com... Sender ok
rcpt to: buttyan@hit.bme.hu
250 2.1.5 buttyan@ebizlab.hit.bme.hu... Recipient ok
data
354 Enter mail, end with "." on a line by itself
Your fake message goes here.
.
250 2.0.0 h1ADO5e21330 Message accepted for delivery
quit
221 frogstar.hit.bme.hu closing connection
Connection closed by foreign host.
%
```


© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Be careful, though!

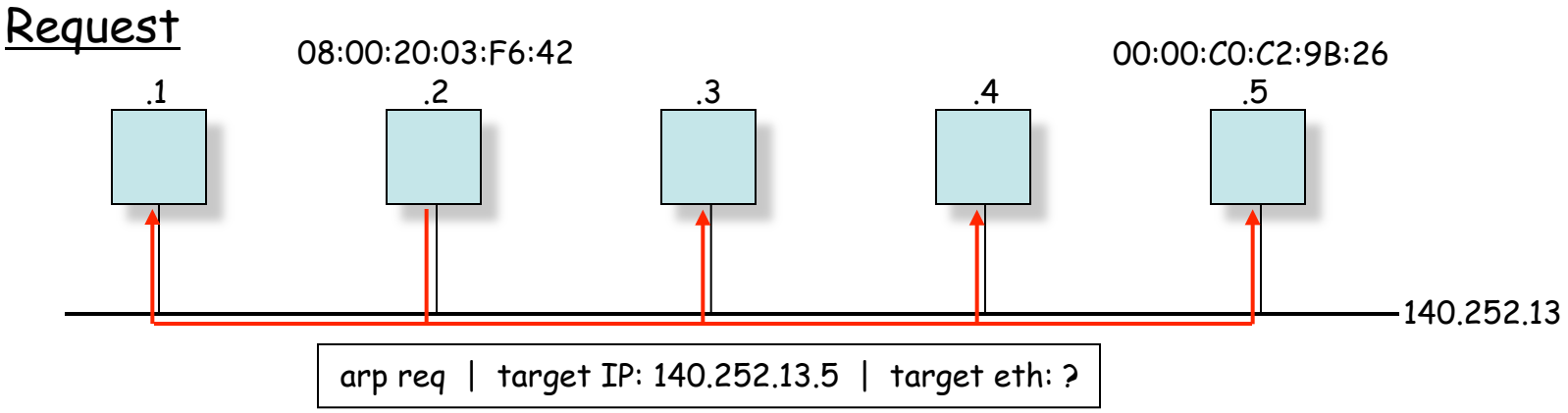
Return-Path: <bill.gates@microsoft.com>
Received: from frogstar.hit.bme.hu (root@frogstar.hit.bme.hu [152.66.248.44])
by shamir.ebizlab.hit.bme.hu (8.12.7/8.12.7/Debian-2)
with ESMTP id h1ADSsxG022719
for <buttyan@ebizlab.hit.bme.hu>; Mon, 10 Feb 2003 14:28:54 +0100
Received: from abcd.bme.hu ([152.66.249.32])
by frogstar.hit.bme.hu (8.11.6/8.11.6) with SMTP id h1ADO5e21330
for buttyan@ebizlab.hit.bme.hu; Mon, 10 Feb 2003 14:25:41 +0100
Date: Mon, 10 Feb 2003 14:25:41 +0100
From: bill.gates@microsoft.com
Message-Id: <200302101325.h1ADO5e21330@frogstar.hit.bme.hu>
To: undisclosed-recipients:;
X-Virus-Scanned: by amavis-dc
Status:

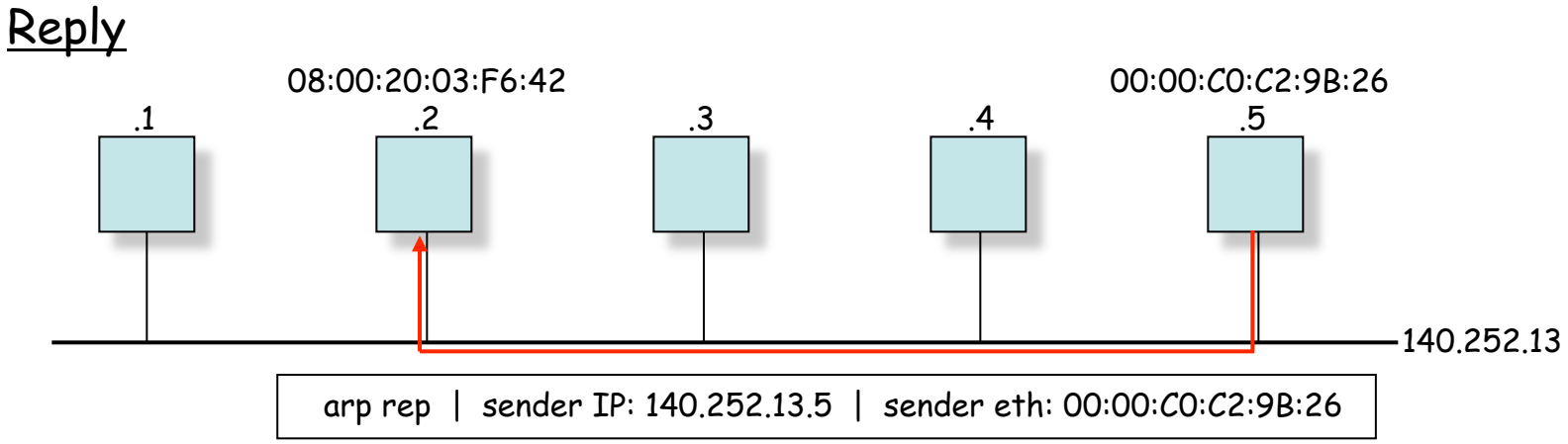
Your fake message goes here.

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

ARP

- mapping from IP addresses to MAC addresses

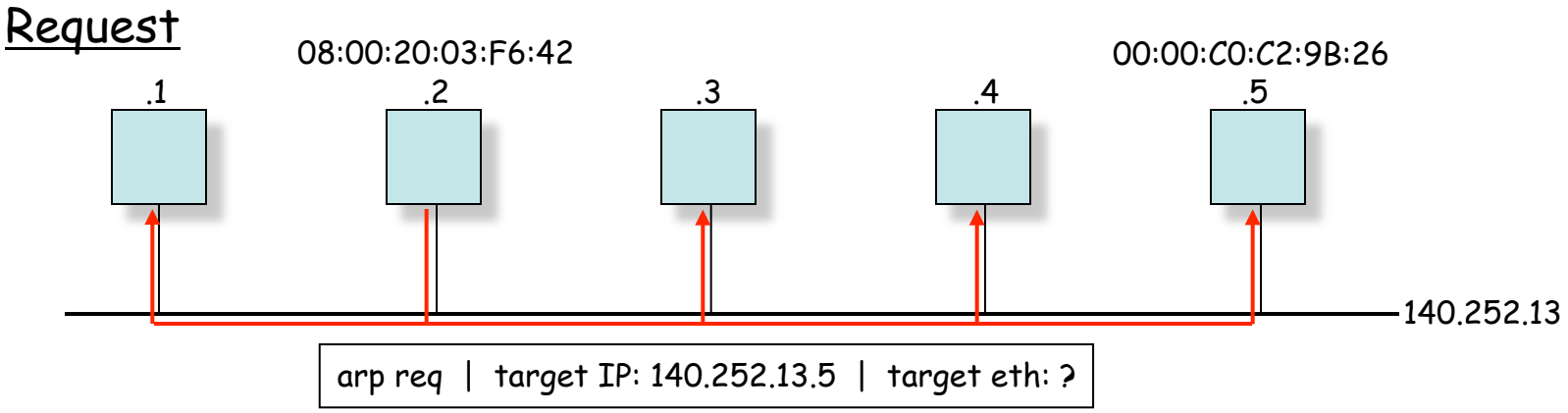


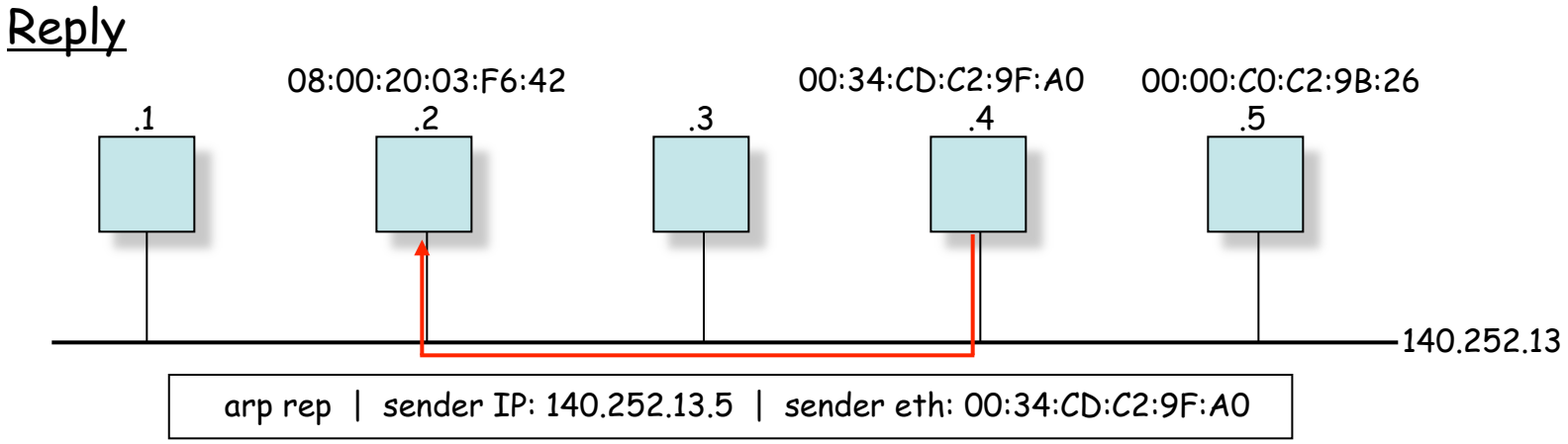


© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

ARP spoofing

- an ARP request can be responded by another host



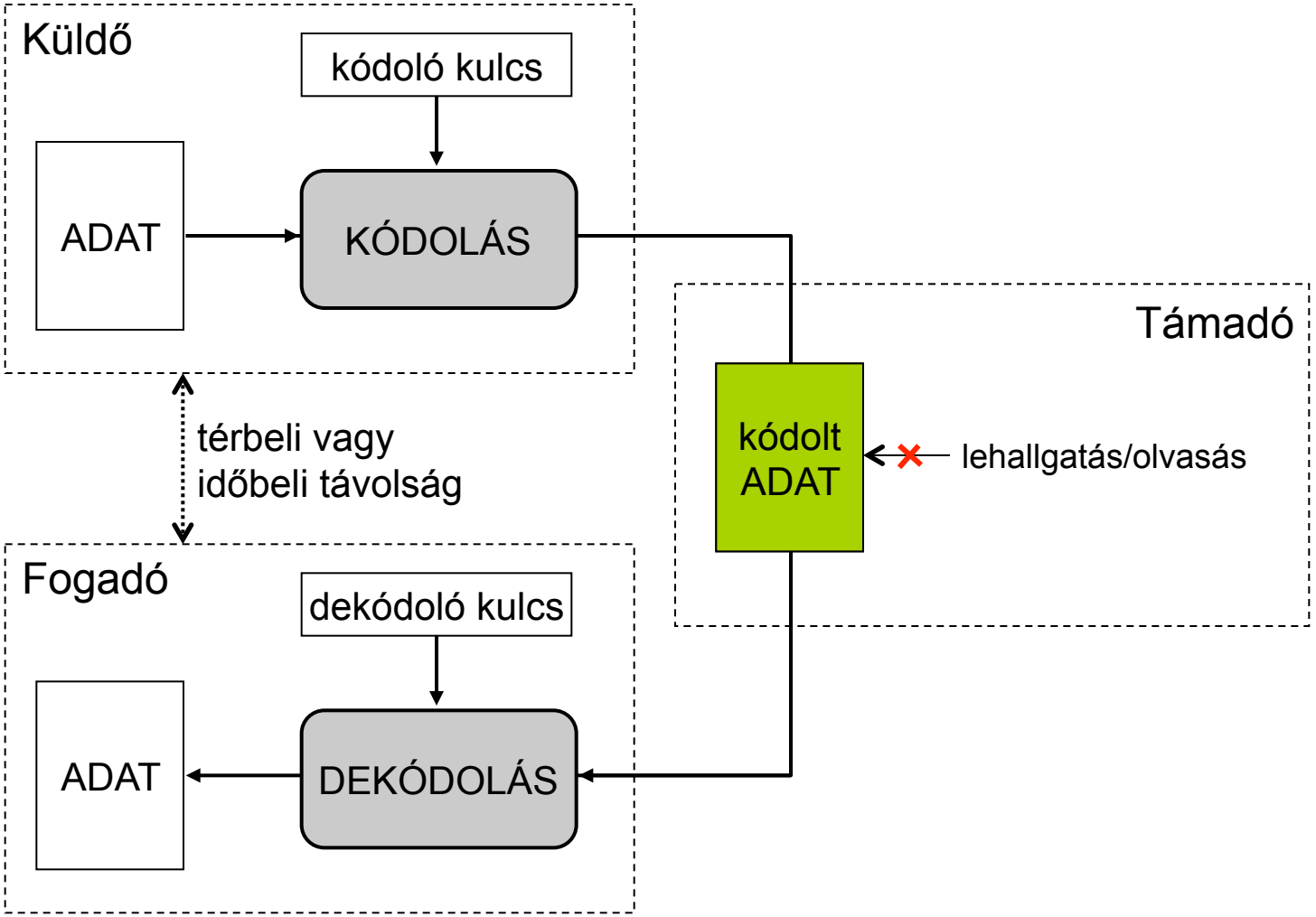


© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Crypto

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Encryption model



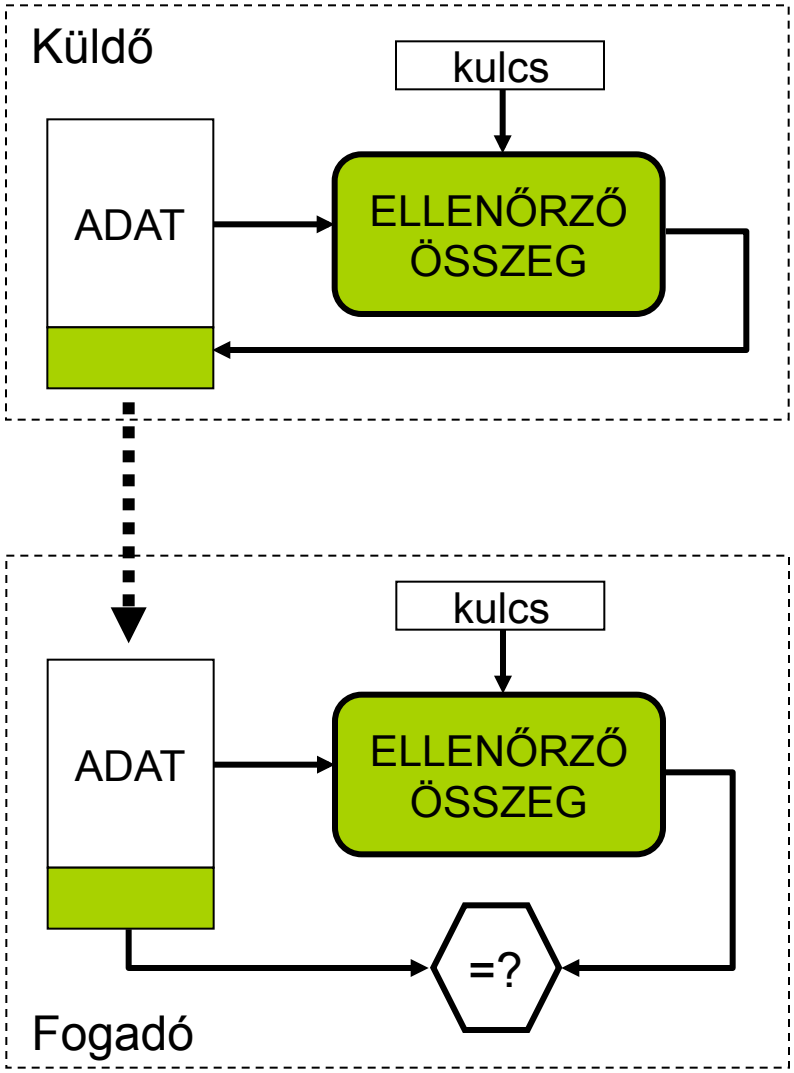
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Algorithms for symmetric and asymmetric crypto

- symmetric key
  - DES
  - 3DES
  - AES
  - RC4
- asymmentric (public) key
  - RSA (factorization)
  - elliptic curve
- key agreement
  - Diffie-Hellman
  - PKI and CAs

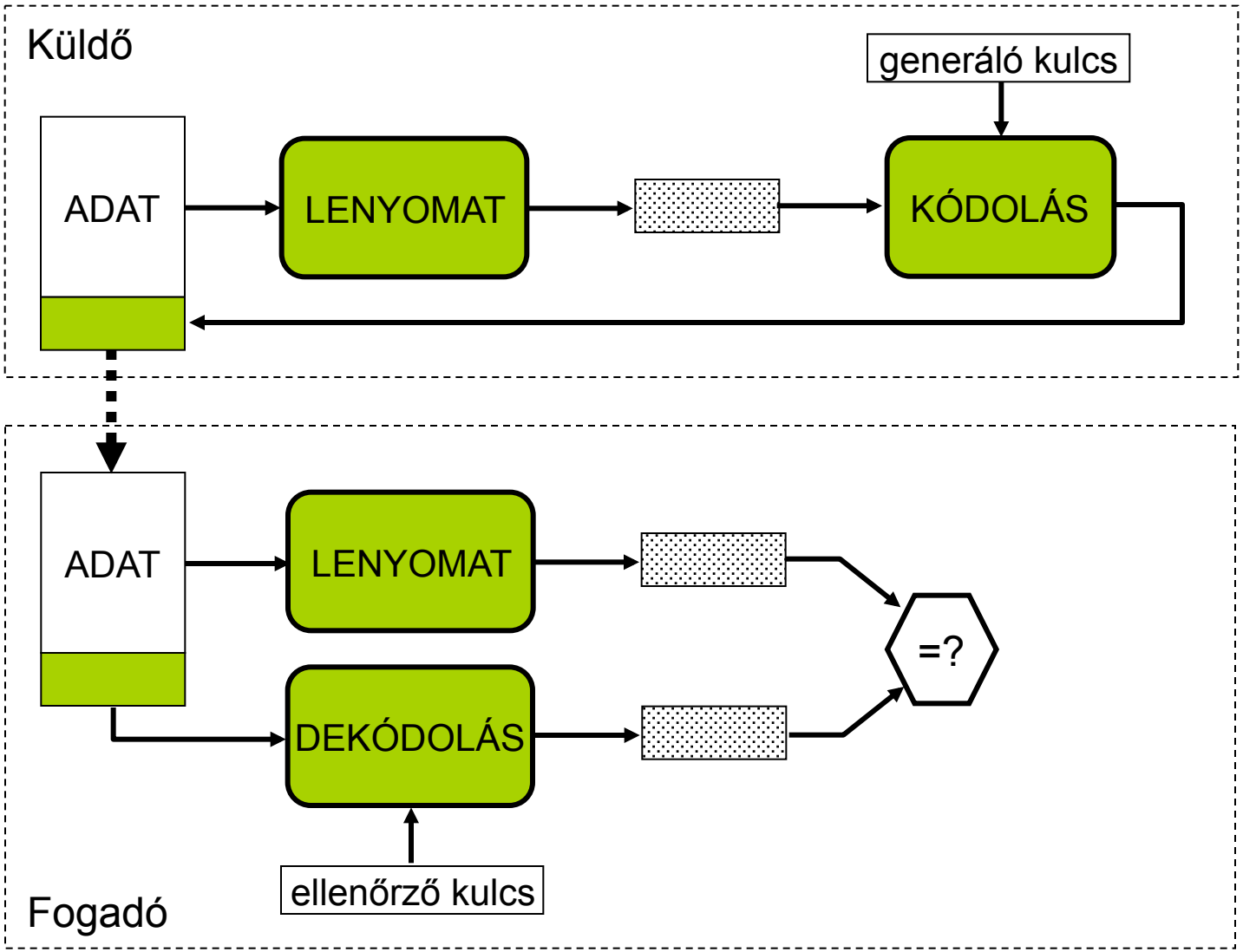
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Authentication and integrity protection



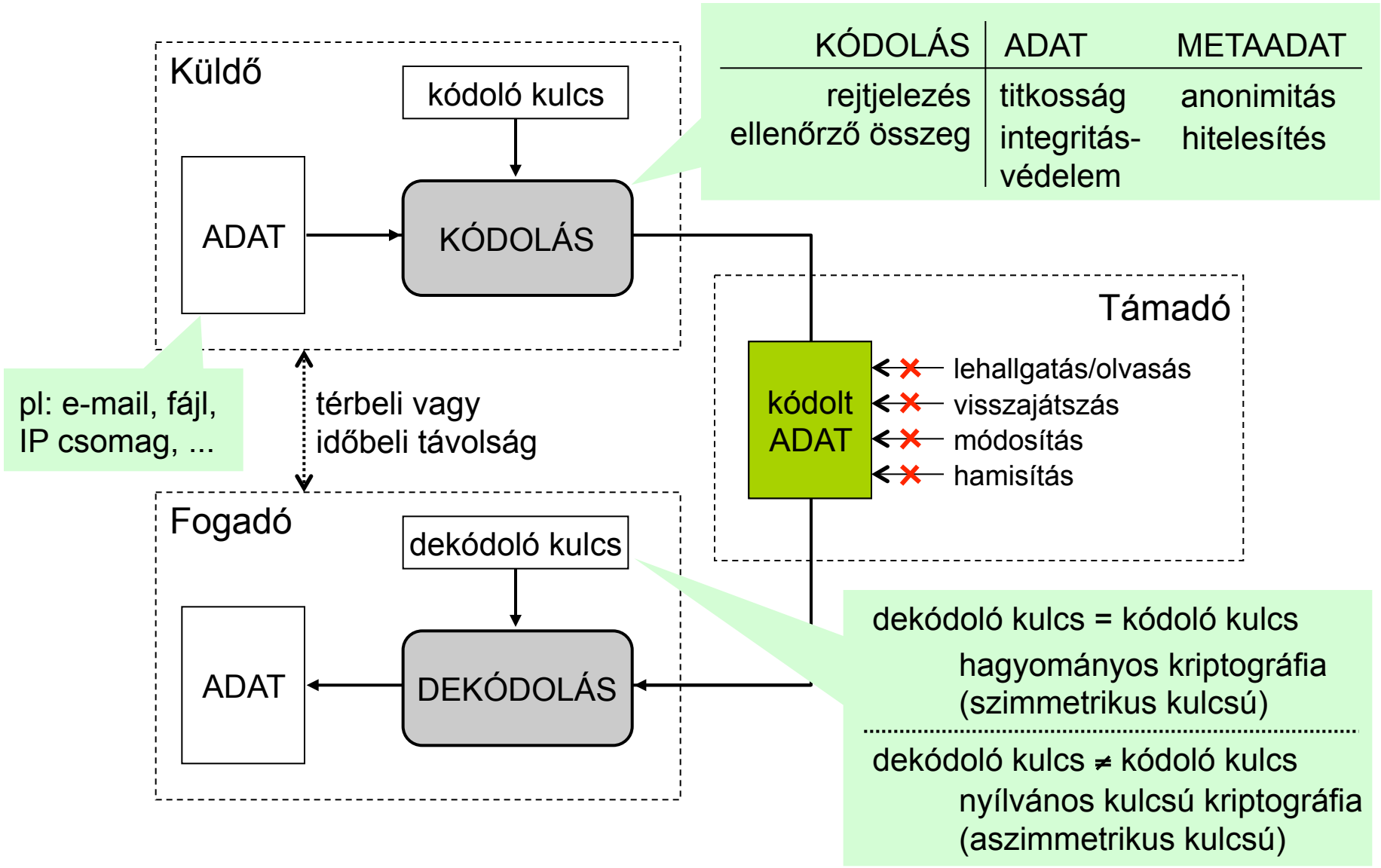
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Digital signature



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Crypto summary



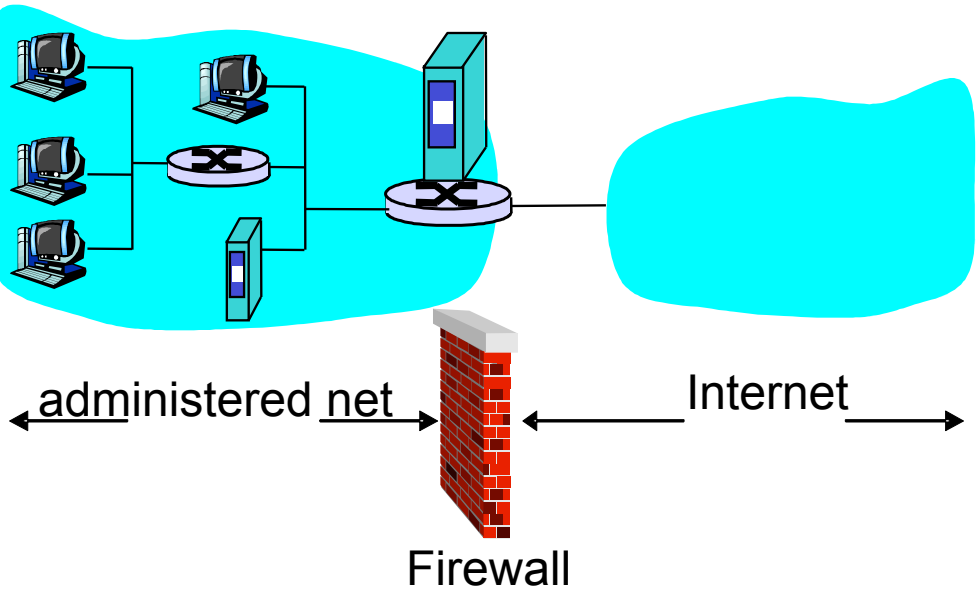
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Network defenses

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Firewalls

- separates the internal network from the Internet
- some packets can pass, other are dropped



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Firewalls: Why?

Denial-of-Service attack defense:

- SYN-Flooding: attacker initiated several useless TCP connections, not enough resources for legit connections

illegal access or manipulation if internal data:

- website defacement

unauthorized access to the internal network

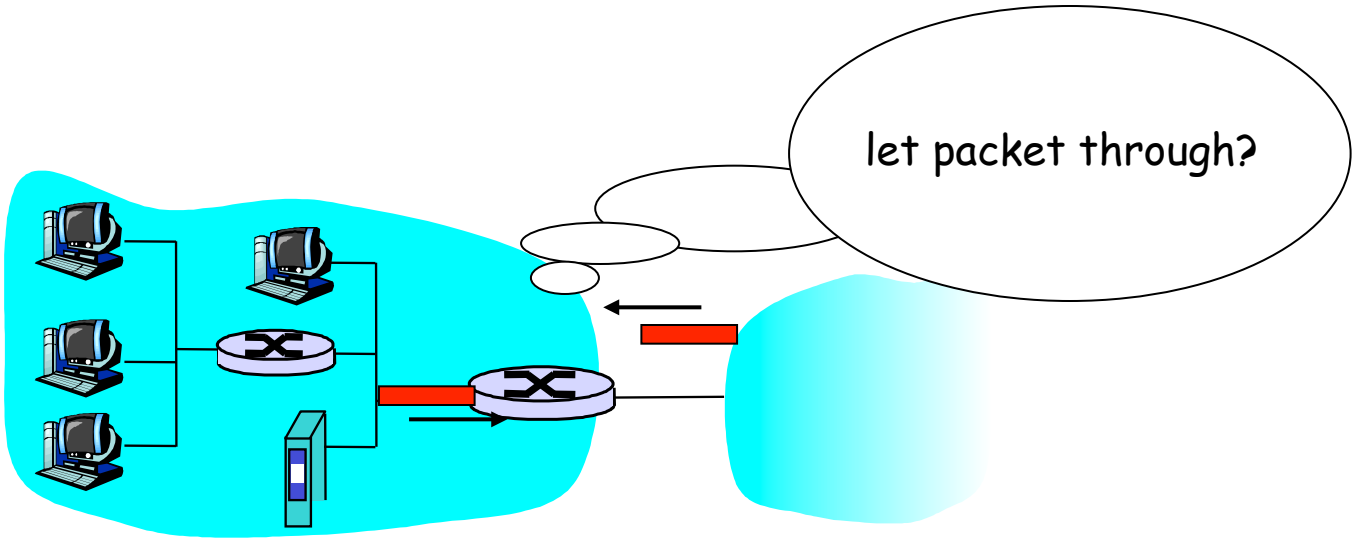
- passing through access control, privilege escalation

3 firewall types (depending on the filtering layer)

- stateless packet filter (L3)
- stateful packet filter (L4)
- application gateway (L7)

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Stateless packet filter



- Router-Firewall separates the internal network and the public Internet
- the router processes each packet, and decides to let it through based on:
  - source and dest IP
  - TCP/UDP source and dest port
  - ICMP data type
  - TCP-SYN- and ACK-Bits

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Stateless packet filter - examples

goal	Firewall rule
no web access to the outside	drop all outcoming packets to all IP addresses to port 80
no incoming TCP connections unless they go to the own webserver	drop all incoming TCP-SYN packets, except if they go to the IP address 130.207.244.203, Port 80.
avoid that a web radio consumes all bandwidth	drop all UDP packets, except for DNS and Router-Broadcasts.
protect network from Smurf-DoS attacks	drop all incoming ICMP broadcast packets (ex. 130.207.255.255)
block net discovery by traceroute	drop outgoing ICMP-TTL-Expired packets

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Access Control Lists

- *ACL:* list of rules applied to all packets on an interface
- in or out
- criteria - action

action	source IP	dest IP	protocoll	source port	dest port	Flags
allow	222.22/16	not in 222.22/16	TCP	> 1023	80	any
allow	not in 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	not in 222.22/16	UDP	> 1023	53	---
allow	not in 222.22/16	222.22/16	UDP	53	> 1023	----
drop	all	all	all	all	all	all

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Stateful packet filter

- stateless packet filters often let useless traffic pass:
  - ex. packets with port 80, ACK flag set, but no TCP connection exists

action	source IP	dest IP	protocoll	source port	dest port	Flags
allow	not in 222.22/16	222.22/16	TCP	80	> 1023	ACK

- *stateful packet filter:* follow TCP connections
  - read SYN and FIN packets
  - timeout for inactive connections on the firewall

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Stateful packet filter

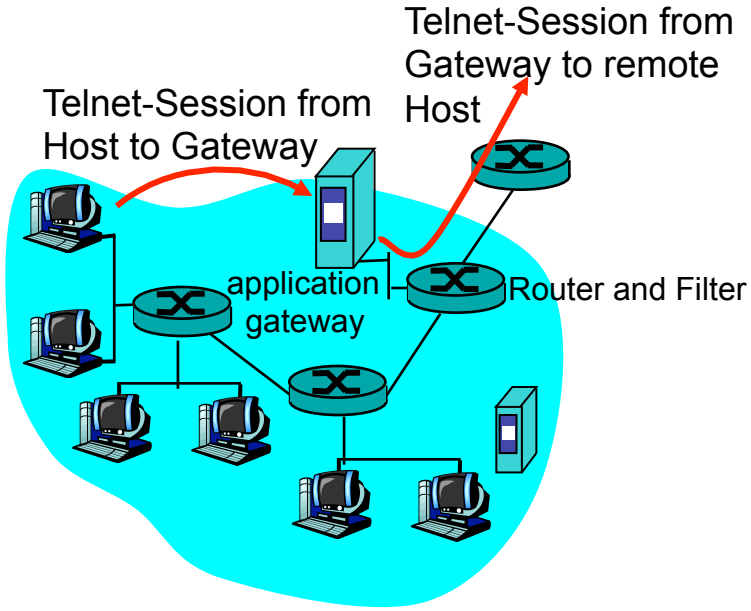
- ACL is extended to check the connection status

action	source address	dest address	proto	source port	dest port	flag bit	check conn
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any	
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK	X
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---	
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----	X
deny	all	all	all	all	all	all	

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Application gateway

- filtering packets based on application info and TCP/ UDP/IP fields
- example: only some users can Telnet to the outside



1. require each Telnet connection to pass the gateway
2. establish a Telnet session for authorized users with the remote host; Gateway used as a proxy to relay data
3. router firewall blocks all Telnet connection not originating from the gateway

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Limits of firewalls and gateways

- IP-Spoofing: a router cannot know if the data comes from the source IP
- we need an application gateway for each special application
- Client software has to know which gateway to access:
  - need to set an address of the web proxy
- "all or nothing" rules for UDP
- communication possibilities vs. security level
- even well protected networks are targeted by attacks

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

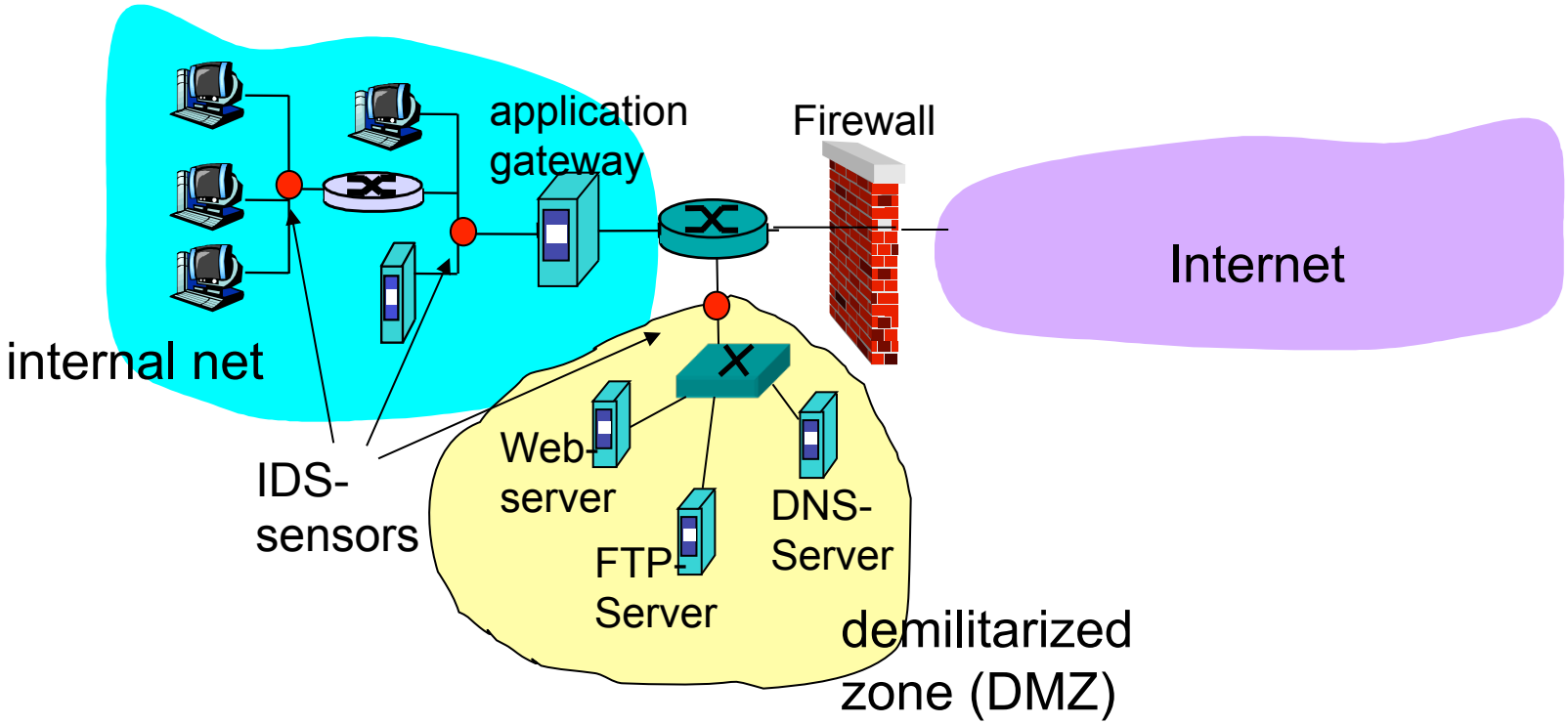
Intrusion detection systems (IDS)

- packet filters:
  - applied only on TCP/IP headers
  - data from different sessions cannot be correlated
- Intrusion detection systems (IDS)
  - *Deep Packet Inspection:* analyze the content of packets (ex. for patterns signalling a well-known attack)
  - correlation of multiple packets
    - Port-Scanning
    - network mapping
    - DoS attacks

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Intrusion detection systems

- multiple IDS: various tests at different locations



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Security concepts summary

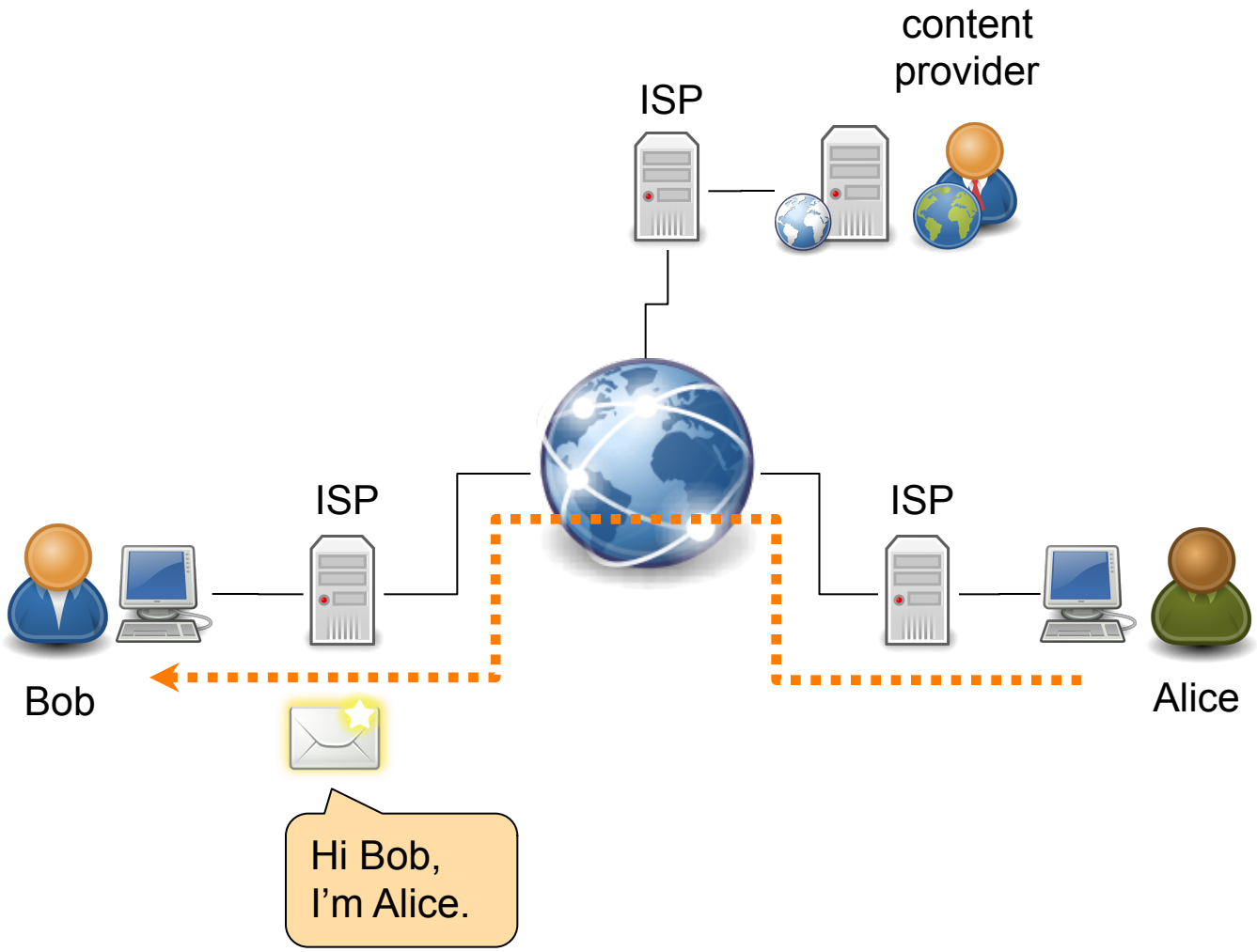
- basic concepts
  - security, attack, vulnerability, threat
  - passive vs. active attacks
  - eavesdropping, traffic analysis, masquerade (spoofing), modification, replay, denial of service
  - main communication security services: confidentiality, integrity, availability, authentication, non-repudiation
- some real world attacks
  - ARP spoofing, e-mail forgery, eavesdropping Telnet and FTP passwords, DDoS attacks
- network defenses

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Economics of Security

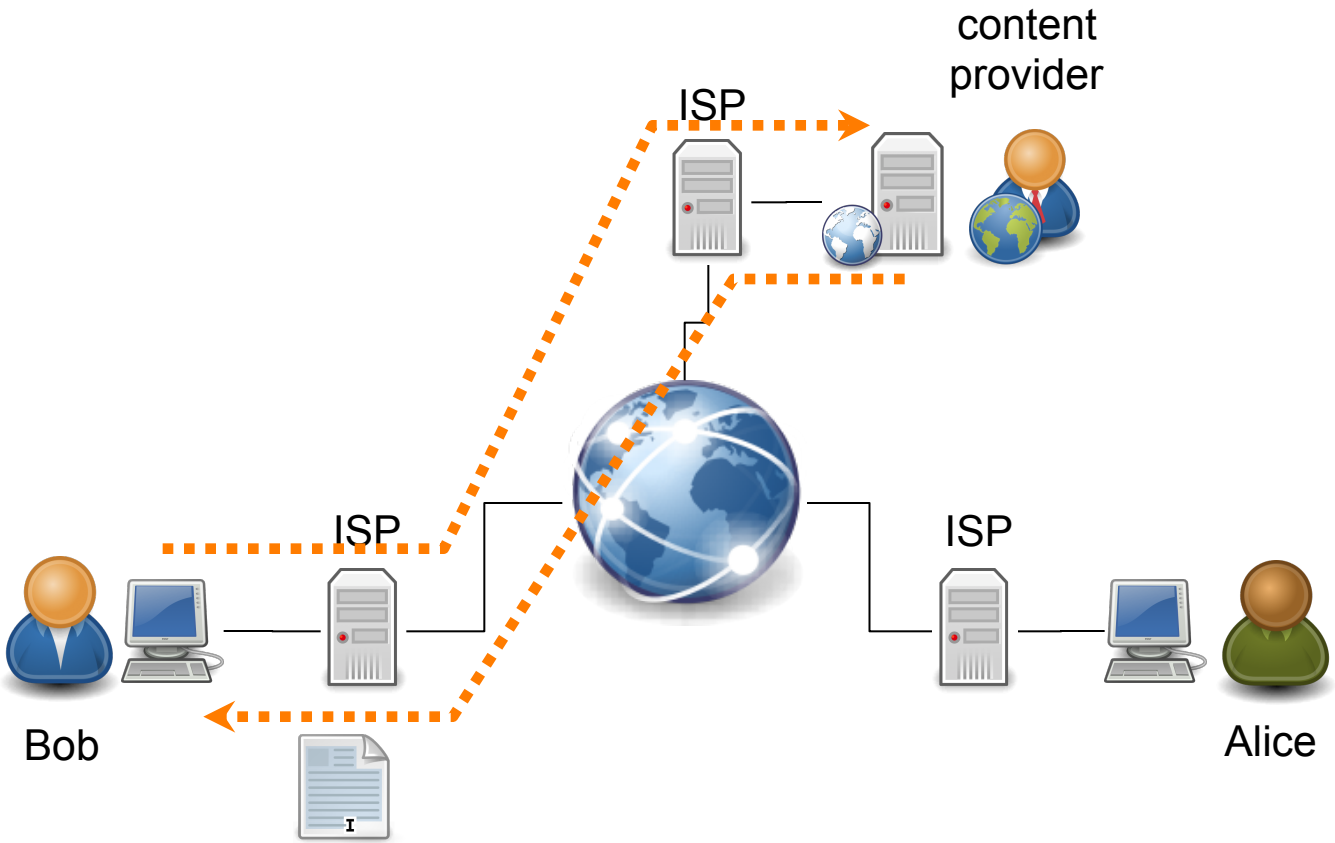
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Internet communication



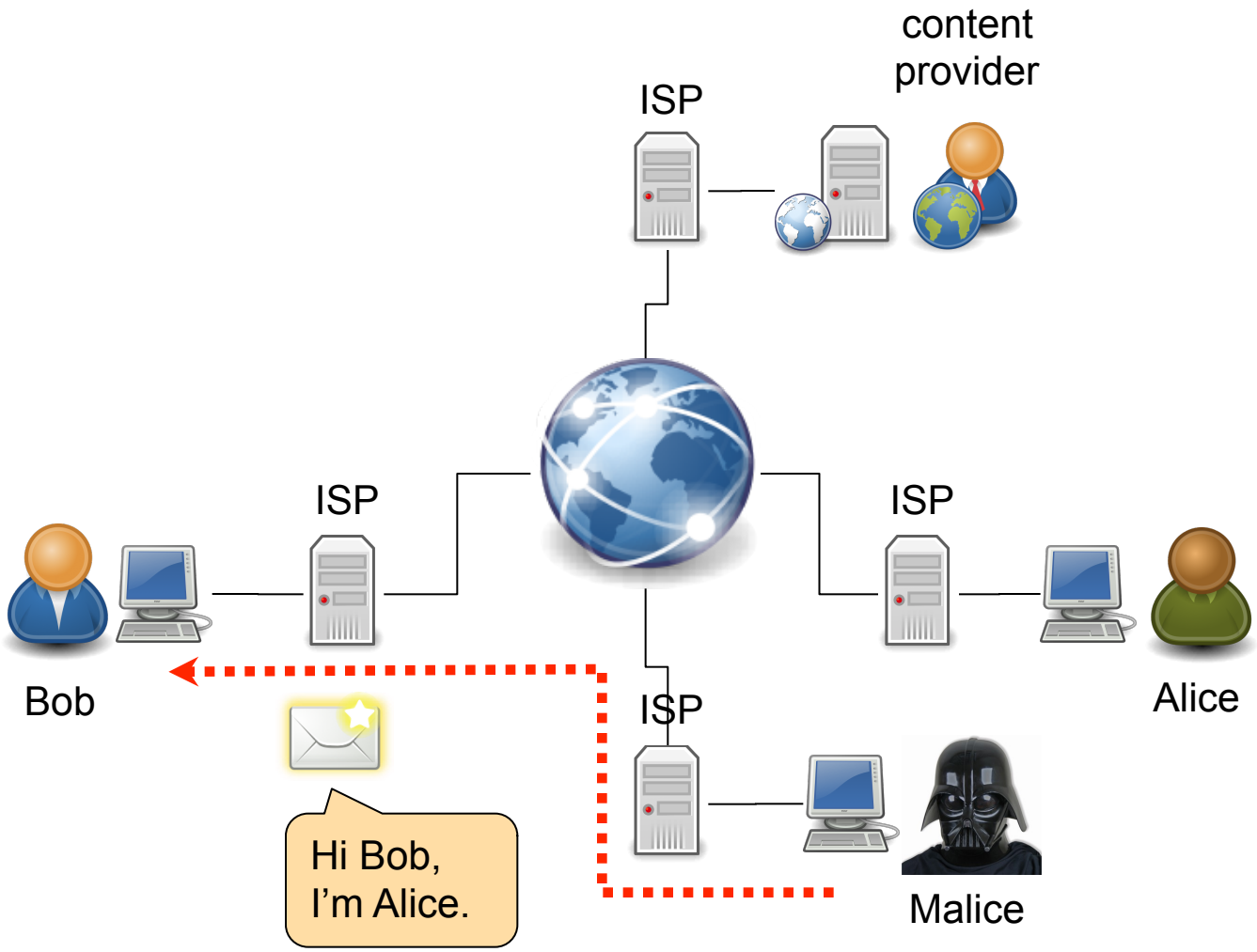
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Internet communication



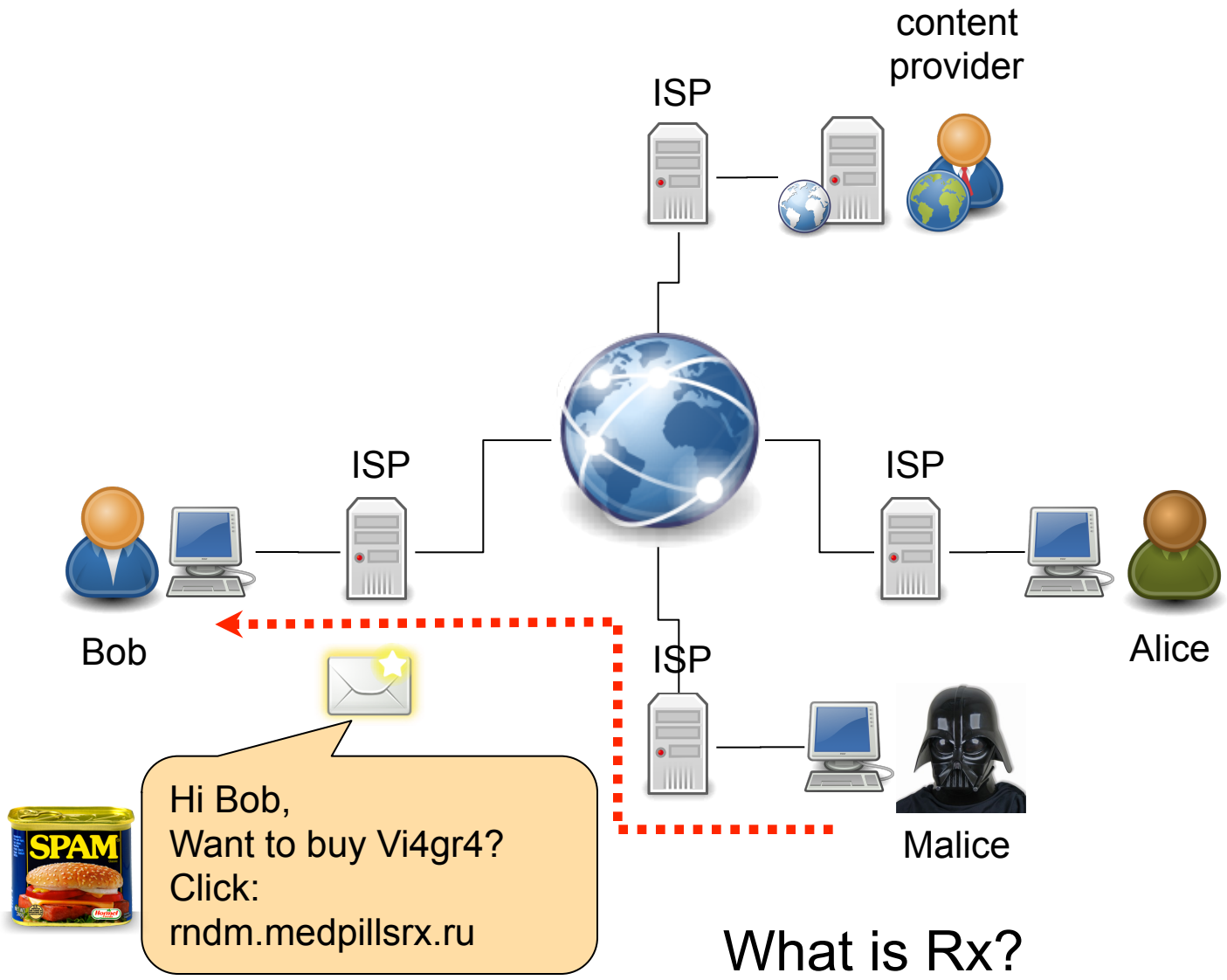
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

When things go wrong



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

When things go wrong



© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

The engineering solution

- develop secure software
- design a public key infrastructure for authentication and data confidentiality
- raise firewalls to block dangerous connections
- compile blacklists to block malicious IP addresses (from sending emails)
- …

So why do we still have security issues?

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security simplified

- attacker's advantage

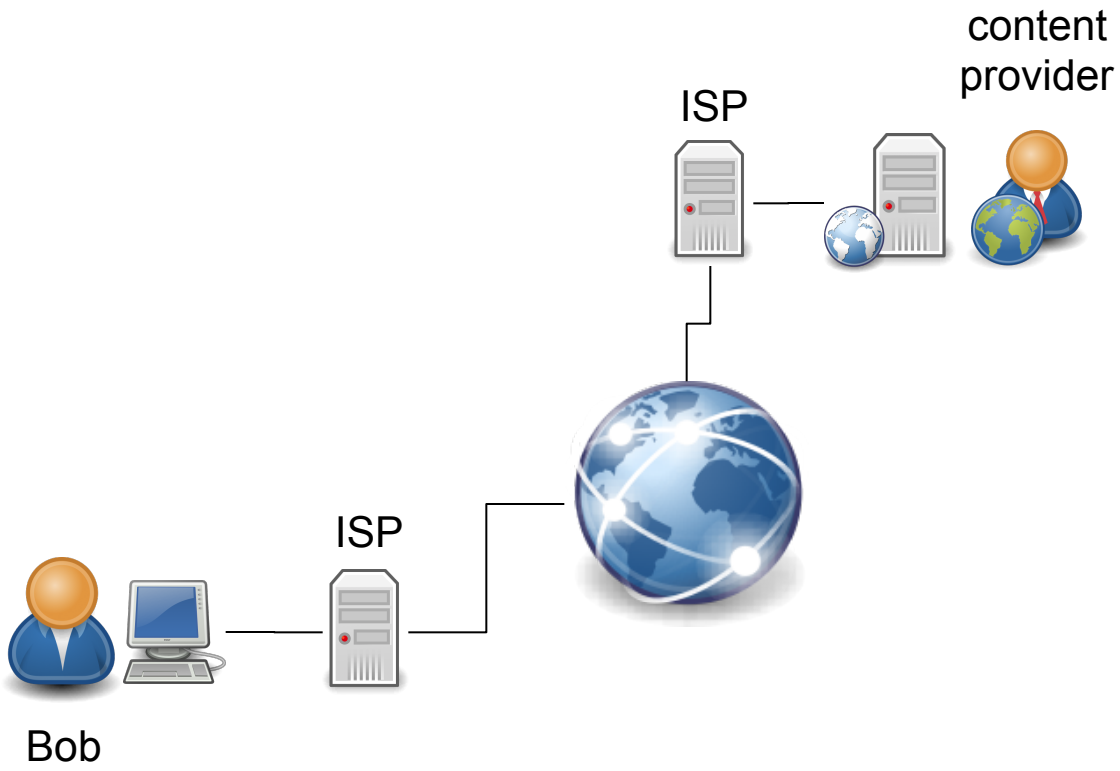
attack	*defense*
• cheap	– expensive
• proactive	– reactive
• easy to measure	– hard to measure
• illegal	– must be lawful

Malice

Bob

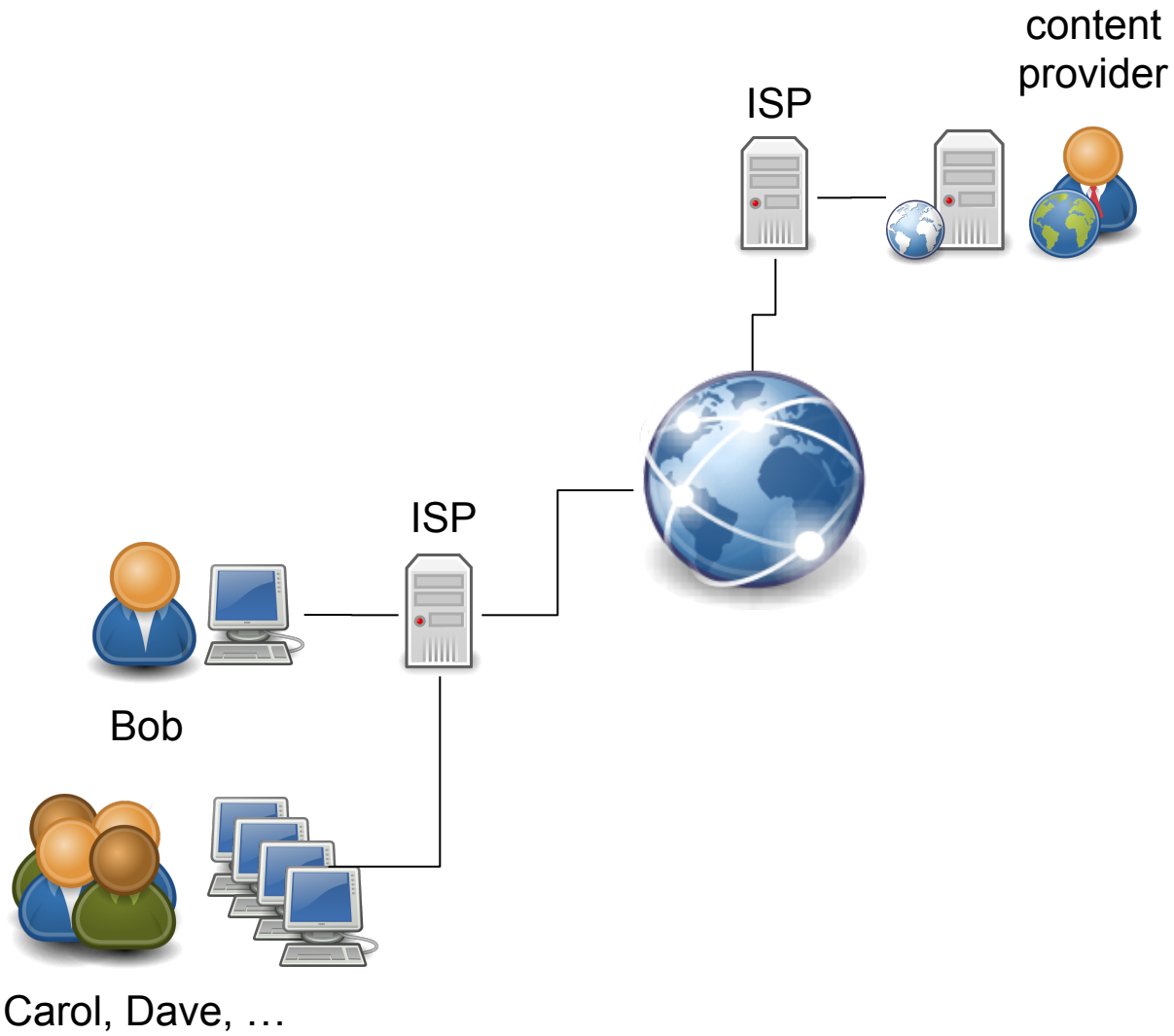
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Information security: the big picture



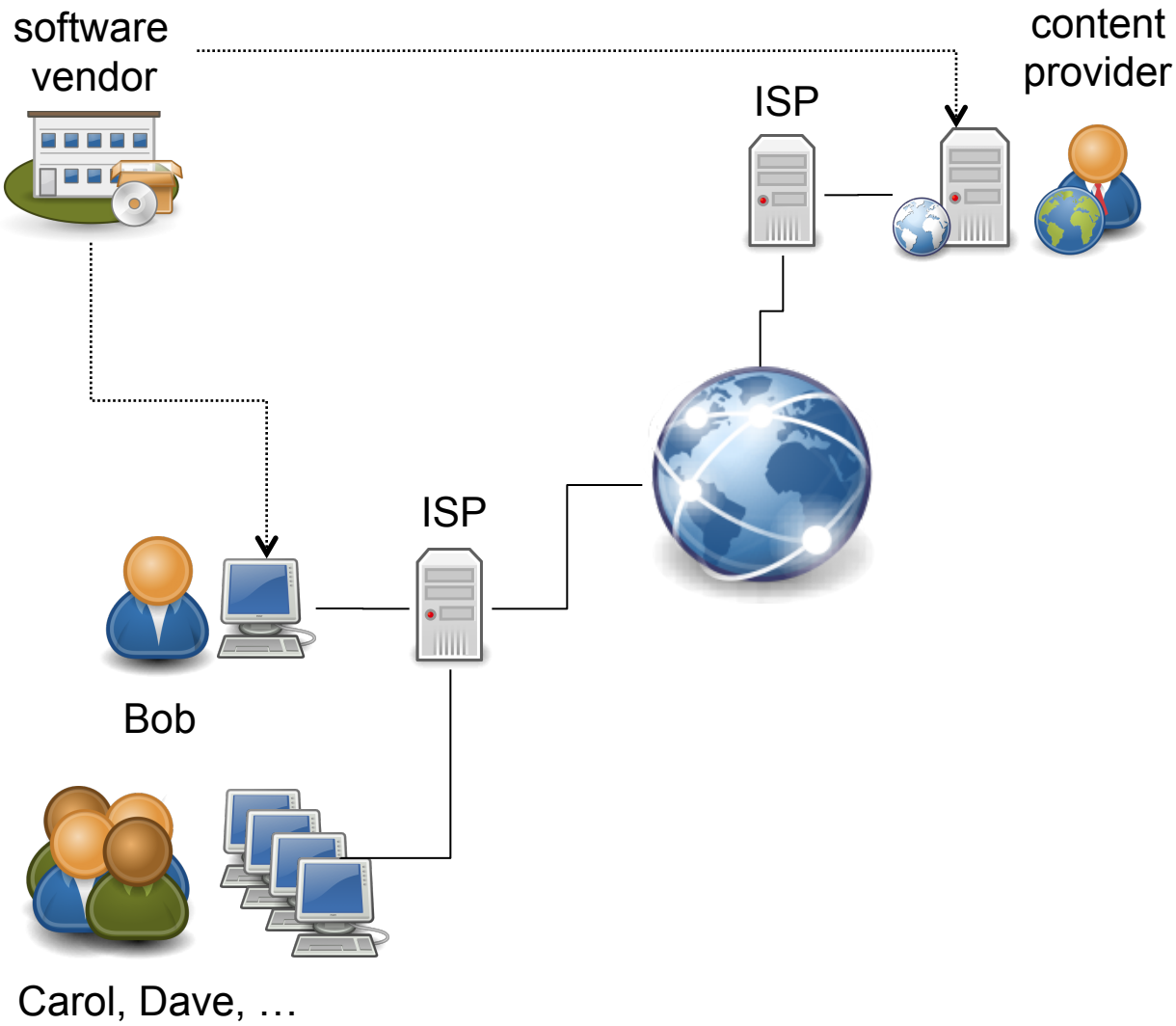
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Information security: the big picture



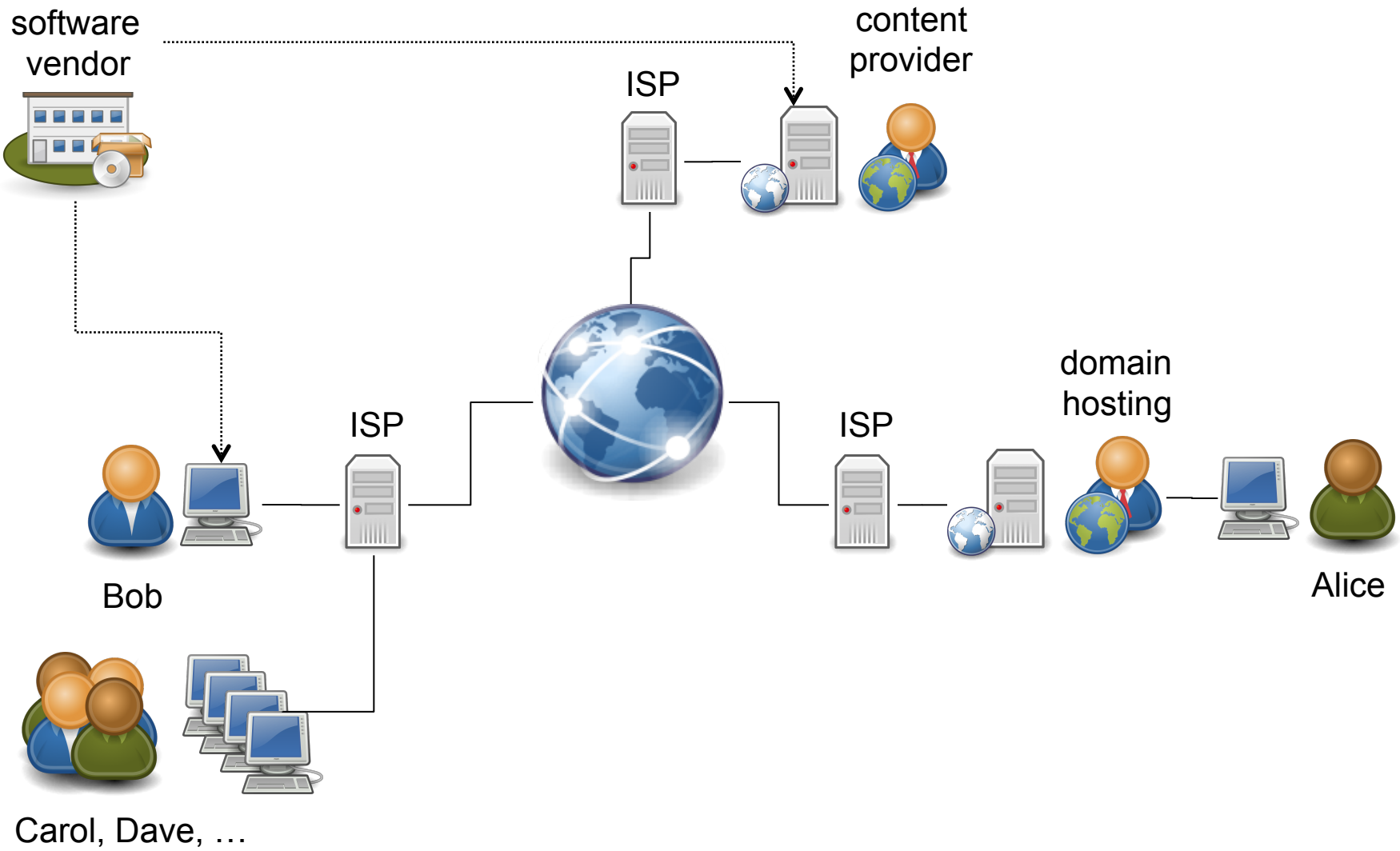
© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Information security: the big picture



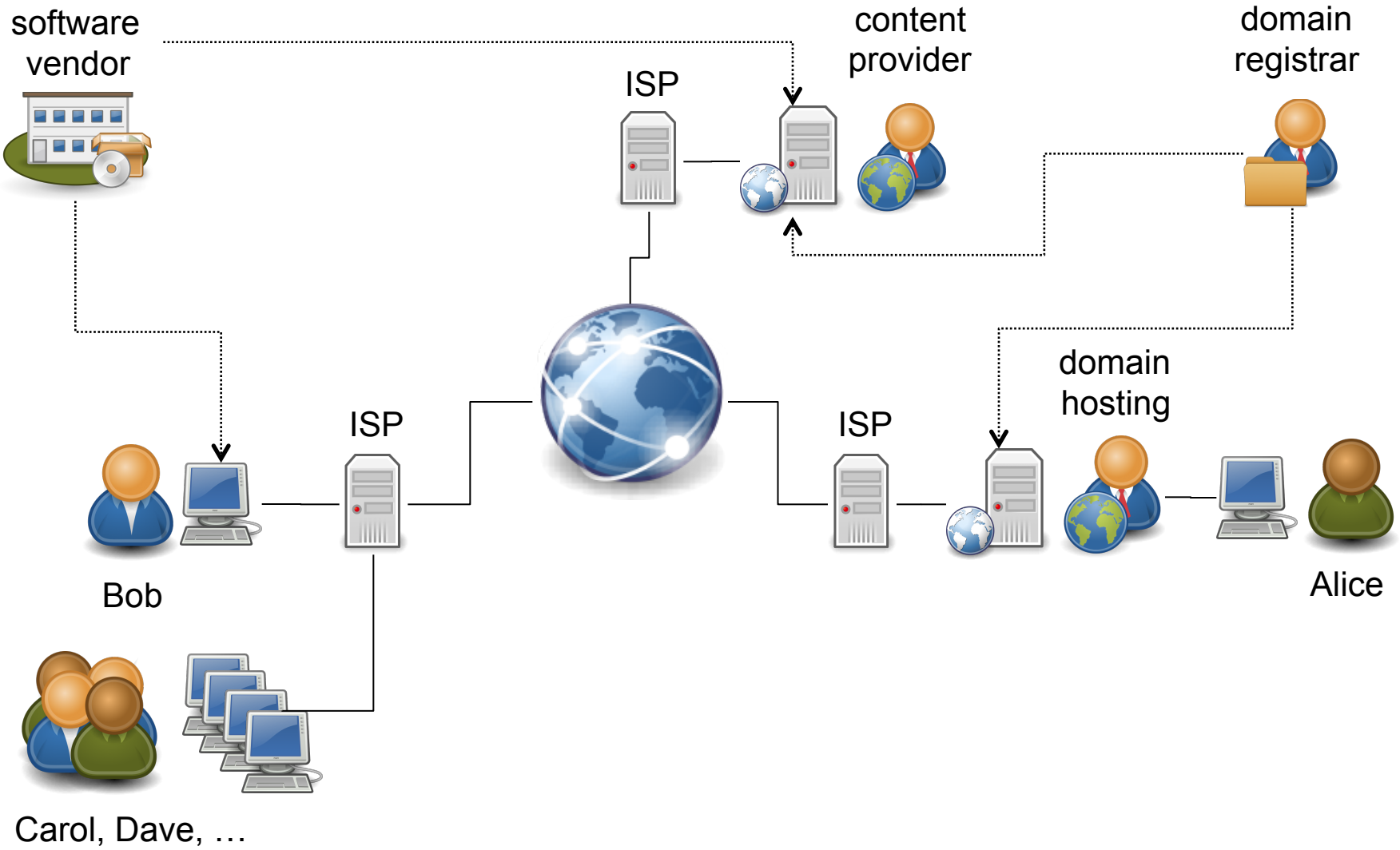
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security: the big picture



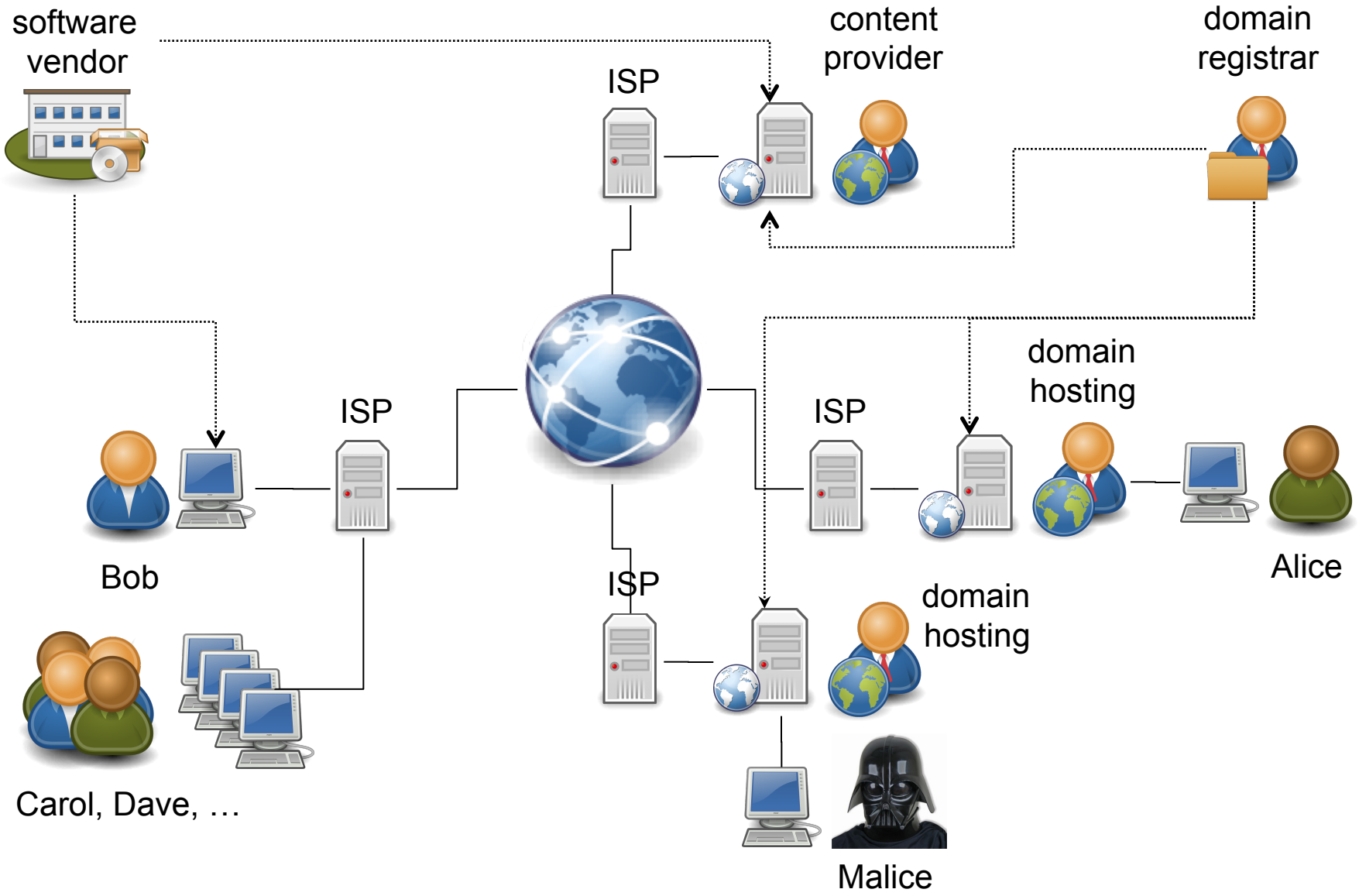
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security: the big picture



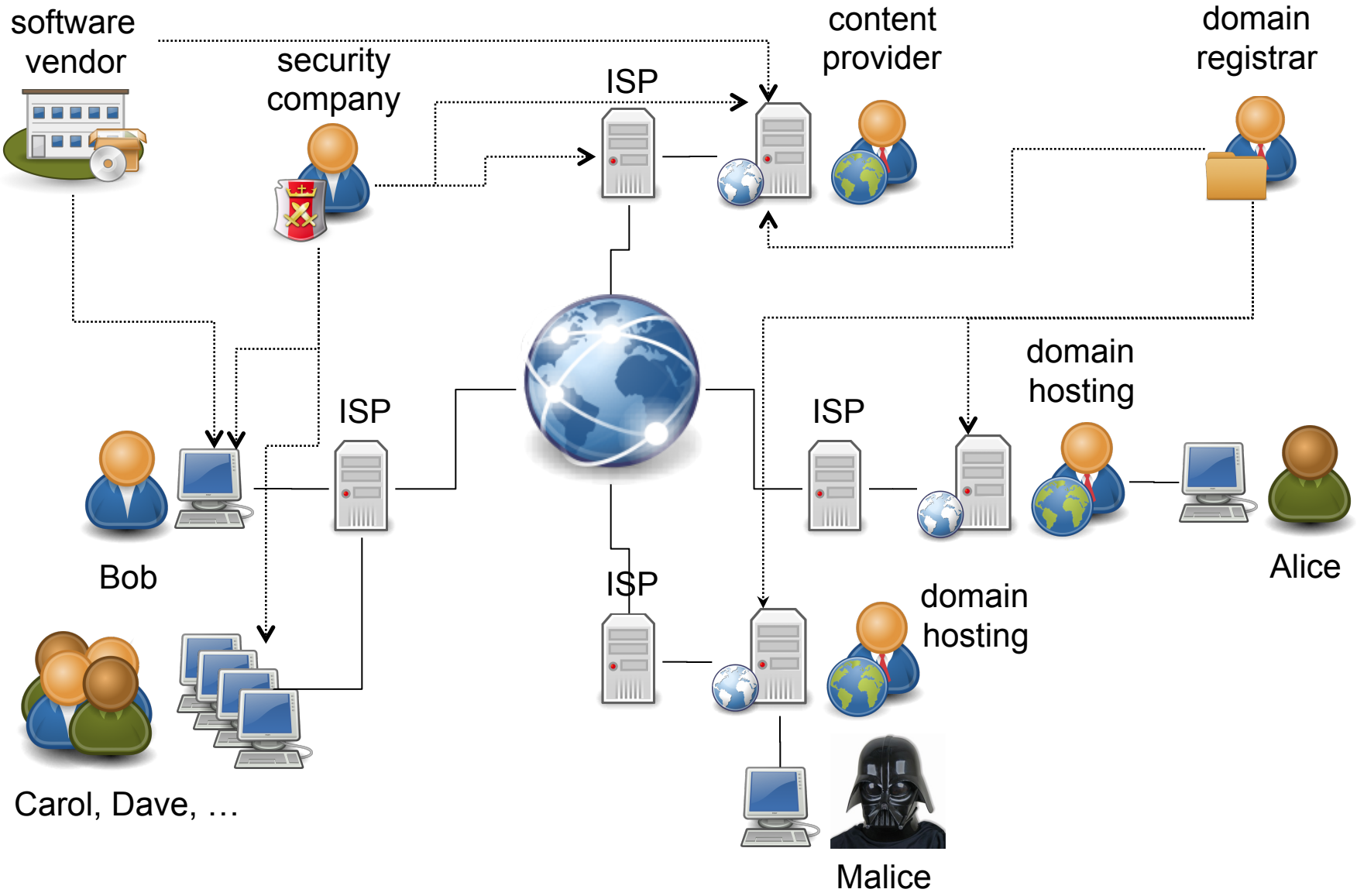
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security: the big picture



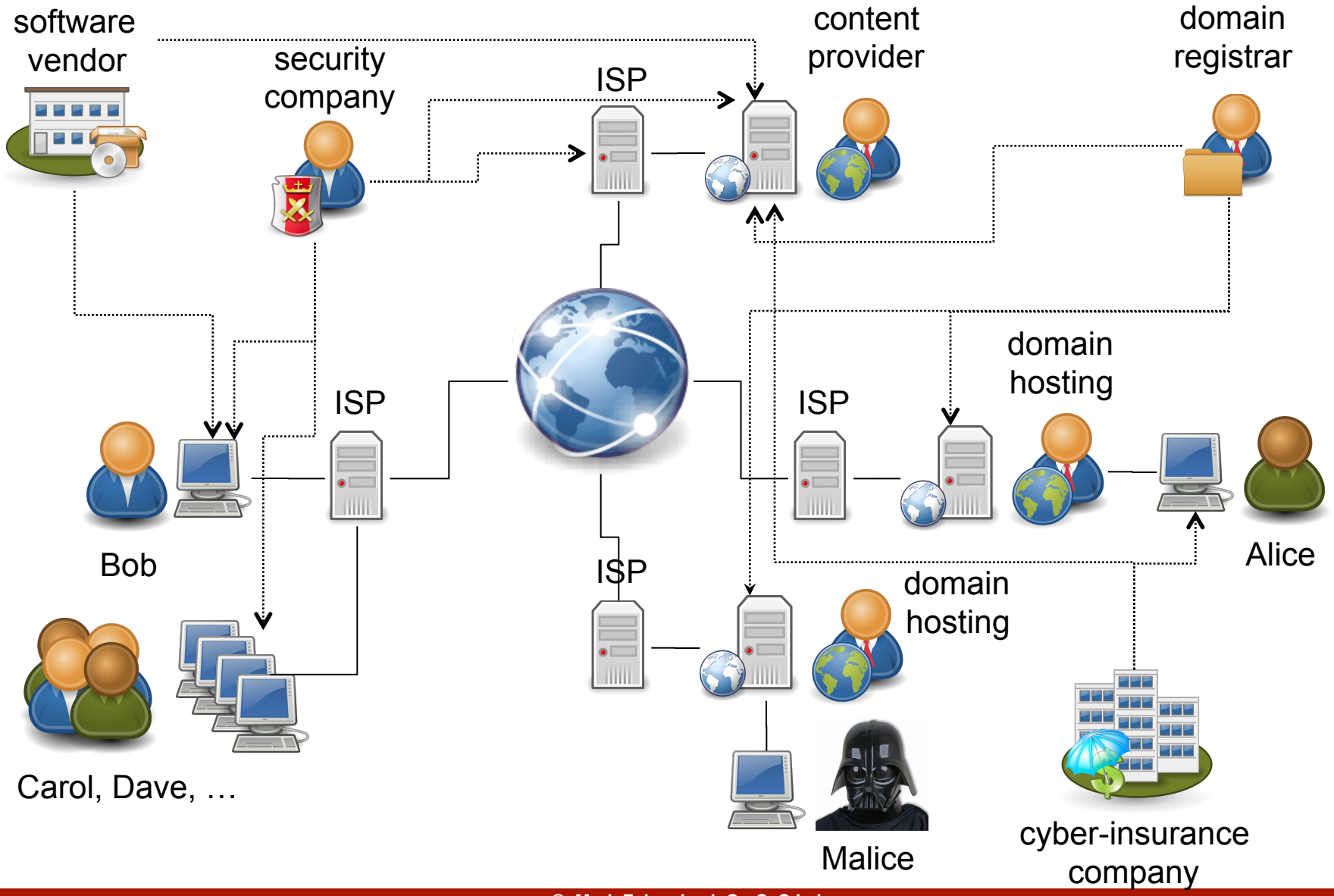
© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security: the big picture



© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security: the big picture



© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Information security incentives

- Common view: technology solves the problem
  - buy better software
  - frequent updates to fix vulnerabilities
  - hire a security expert for audit

Why are system still unsecure?

Answer: **misaligned incentives**

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Information security incentives

- incentives for the parties
  - Bob: system functional (email, web)
    - security is not the main goal for users
  - Malice: money (exploit resources)
- Bob's information security investment
  - buy better software
  - frequent updates to fix vulnerabilities
  - hire a security expert for audit
- What is the optimal security investment for Bob?
  - given: service infrastructure, threat model, security budget, user population, time horizon

Bob

Malice

Economics analysis gives the answer

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Misaligned incentives

Examples:

- fraud against ATMs
- medical payment system (supported by insurers)
- Common Criteria evaluation (by vendor commissioned third –party)

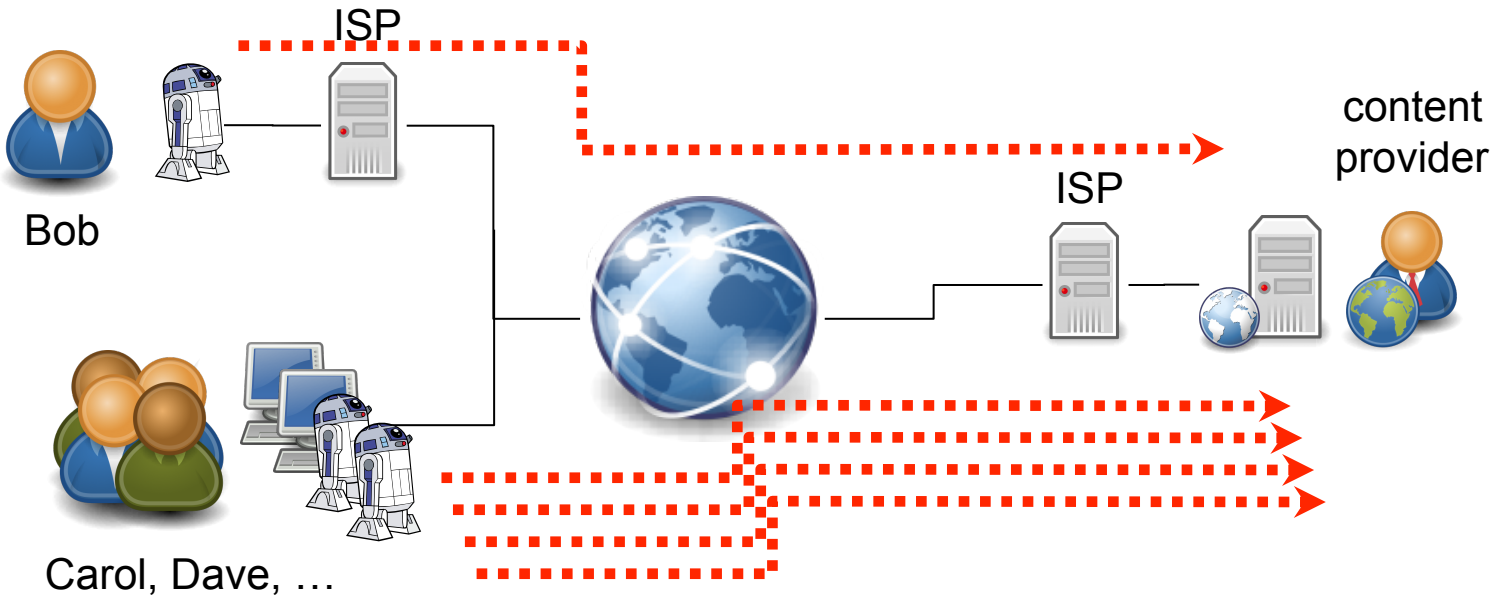
party to implement protection ≠ party to suffer !

1. incentive issue: liability

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Misaligned incentives: DDoS

- One more example: DDoS attack

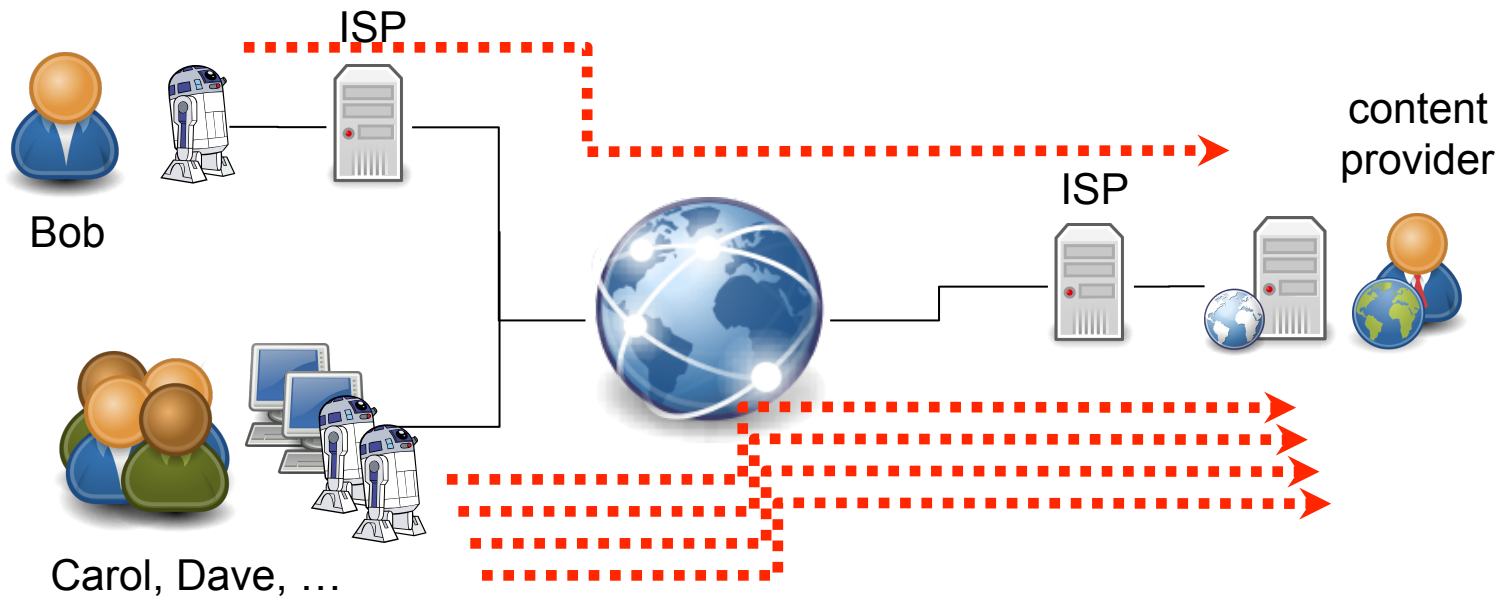


© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Misaligned incentives: DDoS

- One more example: DDoS attack
  - variant of the Tragedy of the Commons

    G. Hardin, “The tragedy of the commons,” Science, 1968

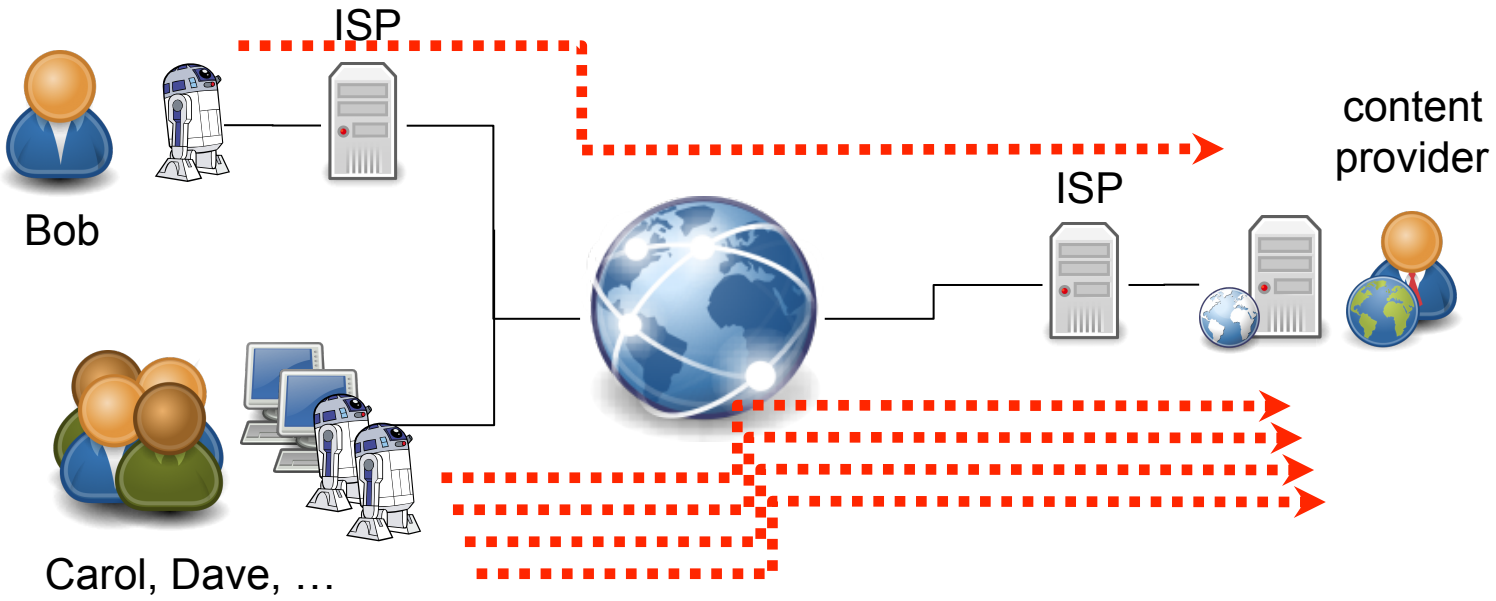


© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Misaligned incentives: DDoS

- One more example: DDoS attack

  (Tragedy of the Commons)



2. incentive issue: free-riding

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Possible solution

- transfer liability to the ISPs
  - technically capable
  - aggregate traffic
  - legally accessible
- more on the topic in Chapter 8

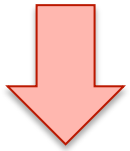
Any thoughts?

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

IT markets

Three properties:

- networks effect – value grows with user base
- high fixed costs and low marginal costs
- lock-in effect



quickly to the market
monopoly products

3. incentive issue:
networks effects and monopolistic markets

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Reminder: attacker's advantage

- Attack is favored against defense
  - attacker need one vulnerability to attack
  - defender needs to find all vulnerabilities
- Remedies
  - not all flaws are critical
  - one patch can fix many flaw
- BUT: when you discover a bug
  - use it for defend own or attack others?
  - attack is more visible

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME

Asymmetric info: Lemon markets

- The theory of asymmetric information

  G. A. Akerlof. “The market for ‘lemons’: quality uncertainty and the market mechanism.” In Quarterly Journal of Economics 84, 488 (1970)

- if the market is not transparent
  - bad products drive out good products
- examples of security products:
  - secure USB sticks, firewall products
- need good signals to increase transparency

4. incentive issue: asymmetric information

© Mark Felegyhazi, CrySyS Lab, Department of Telecommunications, BME

Economics of information security

Causes	***Consequences***
no liability	no security investments
monopolistic software	correlated events
asymmetric information	lemon market
interdependence	tragedy of the commons

© Mark Felegyhazi, CrySyS Lab,
Department of Telecommunications, BME