Beágyazott rendszerek illesztése információs rendszerekhez BMEVIMIM343

dr. Kovácsházy Tamás

TCP/IP protokoll család

Méréstechnika és Információs Rendszerek Tanszék

TCP/IP protokoll család

- „A protokoll, ami átalakította a világot”
- Az Internet alapját képző protokoll család
- Privát hálózatokban is alkalmazható
- Egyeduralkodó a magasabb hálózati rétegekben (adatkapcsolati réteg felett)
  - Hány „alternatívája” volt
    - ISO/OSI (Európa kormányai és az USA)
    - IBM SNA, Novell IPX, Microsoft próbálkozások
  - Mennyi pénzt öltek bele az alternatívákba?
- Beágyazott rendszerekben is széles körben használják
  - Nagyon flexibilisen konfigurálható
  - Erőforrás igények és a funkciók kompromisszumát kell megtalálni
- Nagyon érdekes a története is
  - Műszaki szempontból
  - Gazdasági, társadalmi szempontból

TCP/IP, the Internet Protocol Suite

- Összeforr az Internet és elődeinek a történetével
  - TCP/IP, the Internet Protocol Suite
  - Elsődleges célalkalmazás a fejlesztők számára
- A 70-es évek végétől összeforr az Ethernet fejlődésével is
  - Az elsődleges hordozó (ezen fejlesztik)
- Katonai kutatásból fejlődik ki
  - 1960-as évek eleje
    - Az USA védelmi minisztériuma felismeri, hogy a kommunikációs rendszerei védtelenek (atomháború)
    - Az akkoriban használt vonalkapcsolt rendszerek túlélőképesség alacsony
    - Csomagkapcsolt rendszerekkel kezdődnek kísérletek
    - Minta a postai szolgáltatás:
      - A postai szolgáltatás túlélőképessége nagy
      - A leveleket nagyon sok minden (alacsonyabb réteg) továbbíthatja
      - A továbbítás során a továbbító közeg változhat
      - Hierarchikus rendszer lokálisan döntő központokkal

A TCP/IP története, indulás

- A nagy telekommunikációs beszállítók passzívak
  - El akarják adni a termékeiket, nem ismerik fel a lehetőséget
- 1960-as évek második fele
  - Kutatások a DARPA/ARPA finanszírozásával
  - Modell:
    - Több, alternatív, párhuzamos kis projektet
    - Fogdoss össze egy halom jó MSc és PhD hallgatót, fiatal kutatót PhD-vel egyetemenként
    - Vess fel nekik egy valós problémát
    - Bízd meg őket gyors prototípus fejlesztéssel
    - Szervezz találkozókat közöttük
      - Ami jó, a többi projekt is gyorsan átveszi az eredményeket
    - A működő prototípusok alapján szabványosítás
    - A csoportokat azonnal kösd össze az eredménnyel, hogy még gyorsabban tudjanak kommunikálni
      - Ez újabb problémákat vet fel, amiket majd megoldanak
  - A modell nagyon hatékony
    - Persze ha van pénz...

TCP/IP története 1.

- 1969 ARPANET kísérlet (3 node)
  - Request for Comments, RFC a szabvány
    - Jelenleg legmagasabb sorszámú az RFC6138 LDP IGP Synchronization for Broadcast Networks
    - 2009 őszén a legmagasabb sorszámú az RFC 5734 Extensible Provisioning Protocol (EPP) Transport over TCP volt
    - Az IETF a szabványosítási szervezet (http://www.ietf.org)
- 1973 Robert E. Kahn és Vinton Cerf (ARPANET egyik tervezője, jelenleg a Google-nél)
  - Open-architecture interconnection
  - Az ARPANET következő generációjának kifejlesztés
  - Router (útvonalválasztó fogalom megjelenése)
  - UNIX OS és a C nyelv a fejlesztőplatform
- 1982 US Department of Defense bevezeti a TCP/IP-t
  - A protokoll verzió számot azóta nem váltott, minden felmerülő problémát megoldottak az elmúlt 29 évben (IPv4 és TCPv4)
    - Kiterjesztésekkel és opciókkal
    - Új protokollokkal

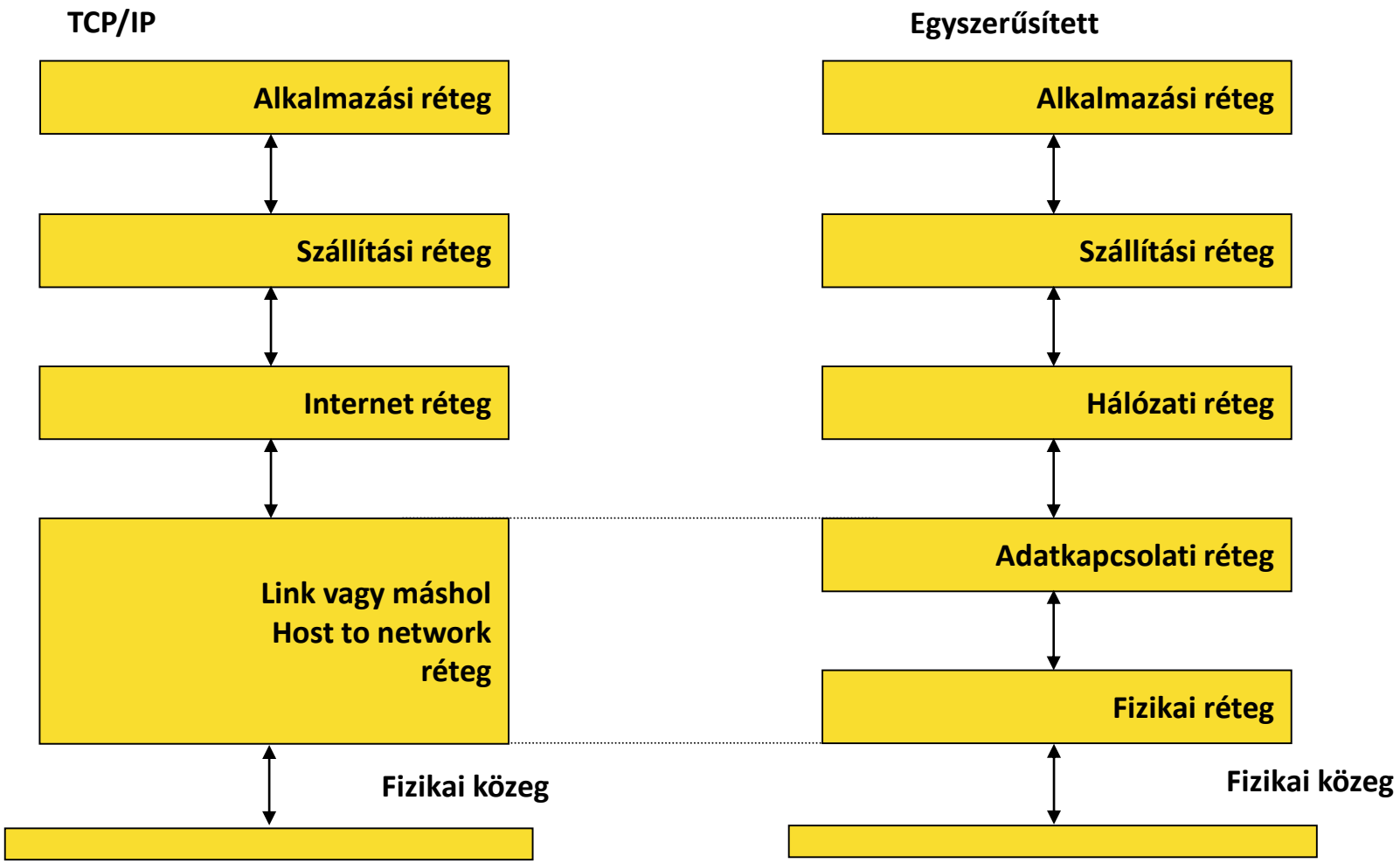
TCP/IP története 2.

- 1988 Nyitás a piac felé, első Internet szolgáltatók
- 1988 november 2. Morris/Internet WORM, elszabadult kísérlet, CERT/CC, biztonság
- 1991 augusztus 6. A CERN publikálja a WWW/HTTP első verzióját (Tim Berners-Lee)
- 1993 Mosaic browser (grafikus, beágyazott képet megjeleníti, stb., Marc Andreessen)
- 1996 Széleskörű elterjedés, teljes piacosítás kezdete
- 1998 IPv6 szabvány elfogadása
  - Fő ok: Az IPv4 címtér kimerül, első becslések szerint 2000-2002-ben, ma 2011 körül
- 2009 1.5-2 milliárd felhasználó, <1% IPv6
- 2009 IPv6 adaptáció gyorsul, címtér kimerülése tényleg közel van már (end game)
- 2011 IPv4 címek elfogynak a Internet Assigned Numbers Authority (IANA) címterében, már csak regionálisan van szabad cím

TCP/IP jövője?

- Hogyan tovább?
  - Internet of Things
    - A jövő Internetjén az embereket körülvevő eszközök kommunikációja lesz a jellemzőbb
  - IPv6
    - Az IPv4 tapasztalatai alapján egy sokkal jobban átgondolt, megtervezett protokoll
    - Bevezetés vontatottan halad.
    - Sokkal erőforrás-igényesebbnek tűnik, de…
  - Interplanetary Internet
    - Delay-tolerant networking

Ismétlés: referencia modell



Ismétlés: Link vagy Host to network réteg

- Minden egyes esetben ki kell dolgozni, hogy az IP réteg hogyan tudja használni
  - Beágyazás (embedding)
  - Dedikált pont-pont link réteg (pl. soros port)
    - A pont-pont alacsony szintű kapcsolat azonosítja a feleket
    - Adatkapcsolati réteg cím nem szükséges
  - Üzenetszórásos közeg
    - Címzés az adatkapcsolati rétegben
    - Az adatkapcsolati és hálózati réteg címek között meg kell teremteni a kapcsolatot

Ismétlés: IP réteg (Hálózati réteg)

- Feladata:
  - Információ küldése nagyobb, de korlátozott méretű egységekben (**csomag, packet**) a hálózatban **tetszőleges (akár nem szomszédos) állomásnak**
    - Eltérő adatkapcsolati réteg keret és hálózati réteg csomag méret
    - Eltérő technológiákat alkalmazhat a két állomás az alsóbb rétegekben a kommunikáció során (köztes állomás szükséges)
      - Sokféle fizikai és adatkapcsolati réteg technológia van!
- Címzés kötelező ezen a szinten, de az adatkapcsolati réteg cím erre nem használható
    - Nem minden eszköznek van címe az adatkapcsolati rétegben
      - Pont-pont kapcsolatok
    - Az adatkapcsolati réteg címek kiosztása véletlenen múlik
      - Gyártásban kiadják sorfolytonosan, aztán valaki megveszi.
      - Ez alapján hogyan találjuk meg egy nagy hálózatban az állomást?
      - Nagy rendszerekben hierarchikus címzés a kedvező.
    - Altérő címzési megoldások vannak az adatkapcsolati rétegben
- Interfész a szállítási réteg felé
  - Kereteket tartalmazó adatstruktúrák küldése és fogadása, send() és receive()

Ismétlés: Szállítási réteg (TCP v. UDP tipikusan)

- Feladatok
  - Nagyobb információs egységek továbbítása **alkalmazások között**
    - Információ csomagokra bontása (adó) vagy összeállítása (vevő)
  - Hálózati hibák elleni védelem a nagyobb információs egységek szintjén
    - Keret vagy csomag elveszhet, sorrendjük megváltozhat, duplikálódhatnak, stb.
  - Egy állomáson belül több alkalmazás akarja használni a hálózatot
    - Alkalmazások címzése az állomáson belül
  - Adatfolyam vezérlés (folyam szabályzás, flow control)
    - Gyors adó lassú vevő, szűk keresztmetszet a hálózatban, ne alakuljon ki torlódás
  - Torlódásvezérlés (congestion control)
    - A torlódás felismerése és megszüntetése
  - Kapcsolatorientált vagy kapcsolat mentes szolgáltatások
    - Analóg telefon rendszer ⇔ papír alapú levél
  - Folyam alapú vagy csomag alapú szolgáltatások
    - Pl. byte folyam (aszinkron soros port) ⇔ C struktúra érték szerinti átadása
- Interfész az alkalmazási réteg felé
  - Majd beszélünk róla részletesen…

Ismétlés: Alkalmazási réteg

- Feladata
  - Kommunikáció az alkalmazások speciális igényeinek megfelelően
  - Az információ magas szinten történő értelmezésének biztosítása
  - Biztonság és megbízhatóság megvalósítása az alacsonyabb rétegeknél összetettebb módon
- Nagyon sokféle van
  - Elsődleges (pl. HTTP, FTP, SSH) és segítő (pl. DNS, SSL) protokollok
  - Az elsődleges alkalmazás réteg protokollok többnyire az alkalmazásokban kerül megvalósításra
  - A segédalkalmazások sok esetben az operációs rendszerben kerülnek megvalósításra
    - Több alkalmazás használja őket, rendszerszintű kedvezőbb működés érhető el így (pl. DNS cache)

IPv4 protokoll

- Az Internet hálózati réteg protokollja
- Fő feladat: Internetworking
  - A különböző szervezetek hálózatainak összekapcsolása
  - Különböző adatkapcsolati rétegeket felett kell működnie
    - Ethernet, GPRS/3G, PPP (soros linkek), PPPoE, VPN, füstjelek, galambposta, stb.
    - Különböző szolgáltatások, műszaki jellemzők stb.
- Működés a végfelhasználó szempontjából (felsőbb rétegek kifejtése nélkül):
  - A szállítási réteg által küldött információt esetleg szétdaraboljuk
    - Analógia: Csak kis borítékunk van, és a küldendő dokumentum nem biztos, hogy belefér ebbe a kis borítékba
  - Körbevesszük IP fejrésszel
    - Analógia: Borítékra rakjuk, és a borítékra ráírjuk a továbbításához szükséges össze információt
  - Továbbítás az IP rétegben
    - Analógia: Bedobjuk a borítékot a postaládába, vagy ha a szomszéd utcában lakik a címzett, akkor egyből a címzett postaládájába
    - Néha a borítékot még tovább darabolják útközben (eltérő a borítékméret)
  - A címzett összerakja az információt (ha szükséges)
  - Továbbítja a szállítási rétegnek

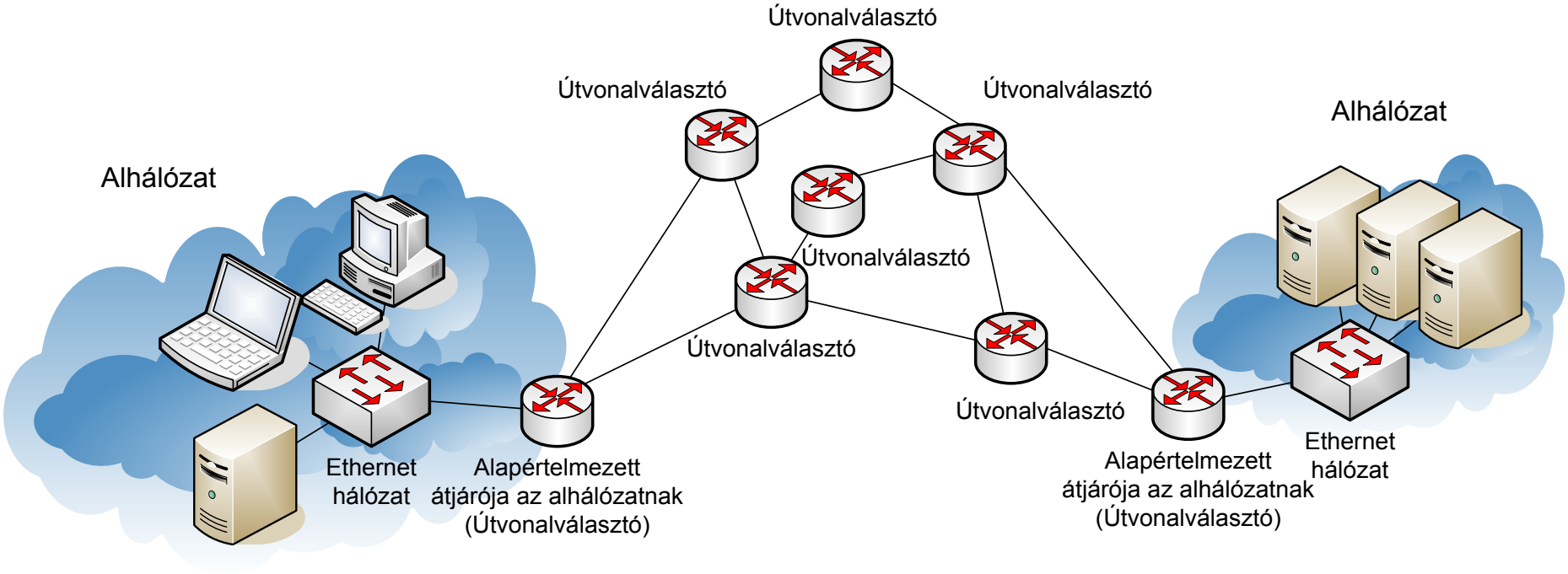
IPv4 szolgáltatás

- Az IPv4 által nyújtott szolgáltatás
  - Csomagkapcsolt
    - Az információ csomagokra darabolva kerül továbbításra
    - A csomagok tetszőleges úton haladhatnak két állomás között
      - Többnyire egy időintervallumban csak 1 úton!
      - A csomagok sorrendje felcserélődhet
      - Egy csomag több példányban is megérkezhet
  - Nem megbízható
    - Csomagok elveszhetnek, vagy megsérülhetnek, rosszindulatú módosítás is lehetséges
    - Bizonyos hibák jelzésre kerülnek
      - ICMP üzenet érkezik a feladóhoz
    - Minden hibát nem lehet jelezni (CRC hiba pl., mi volt a jó üzenet?)
  - Nincs nyugtázás
    - A két végpont közötti hibákat többnyire nem jelezik, a magasabb rétegekben kell kezelni azokat
  - Működéséhez segédprotokollok szükségesek (ARP, ICMP, IGMP)

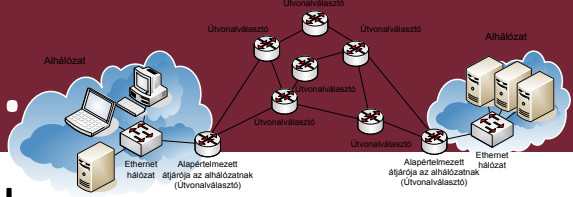
A megoldás előnyei/hátrányai

- A nyújtott szolgáltatás előnyei
  - A hálózat egyszerű, csupán a csomagok továbbításával foglalkozik
    - A csomagok tetszőleges, alkalmas után eljuthatnak a feladótól a címzettig (útvonalválasztás, routing)
      - Az útvonalválasztás konfigurálása az alkalmazási rétegben történik
    - Túlélőképesség jó
      - Dinamikus, és részben automatikus útvonal választás
  - Az állomások felelősek minden más szolgáltatásért
    - Azok könnyen megváltoztathatóak, telepíthetőek, stb.
- A nyújtott szolgáltatás hátrányai
  - A szolgáltatásminőség biztosítása nem lehetséges, vagy bonyolult
  - Az információ biztonsága hálózati szinten nem garantálható
    - Az információ pl. elterelhető, megváltoztatható, stb.
- Az élet bizonyított:
  - A problémák máshol megoldhatók (szállítási és alkalmazási rétegben)…

Internet felépítése



Internet felépítése..

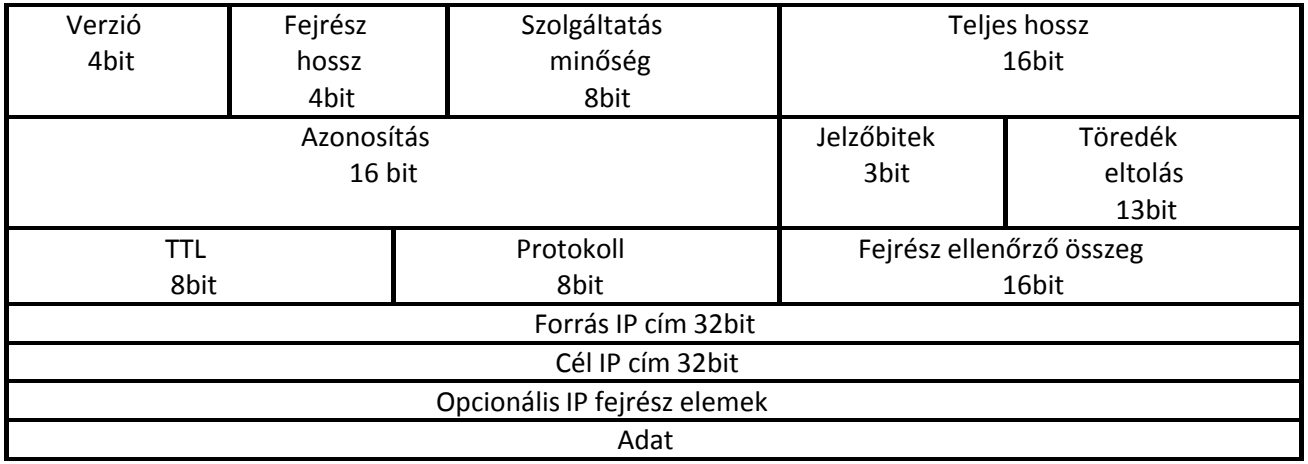


- Alhálózaton (subnet) belül adatkapcsolati rétegben lehet kommunikálni
- Az alapértelmezett átjáró (default gateway) a legközelebbi (az első az úton) útvonalválasztó (router) azon interfésze, amely része az alhálózatnak
  - A többi alhálózattal ezen keresztül lehet kommunikálni
- Az útvonalválasztók (router) továbbítják az információt
  - Ismerik a hálózat topológiáját (mi merre van)
  - Ezt az információt egymás között spec. alkalmazási réteg protokollokkal terjesztik (routing protocol, nem foglalkozunk velük)
  - SW és HW eszközök nagyon eltérő kialakítással
    - Pl. Cisco CRS-1 Carrier Routing System és egy Broadband router eltérései

IPv4 csomagformátum 1.

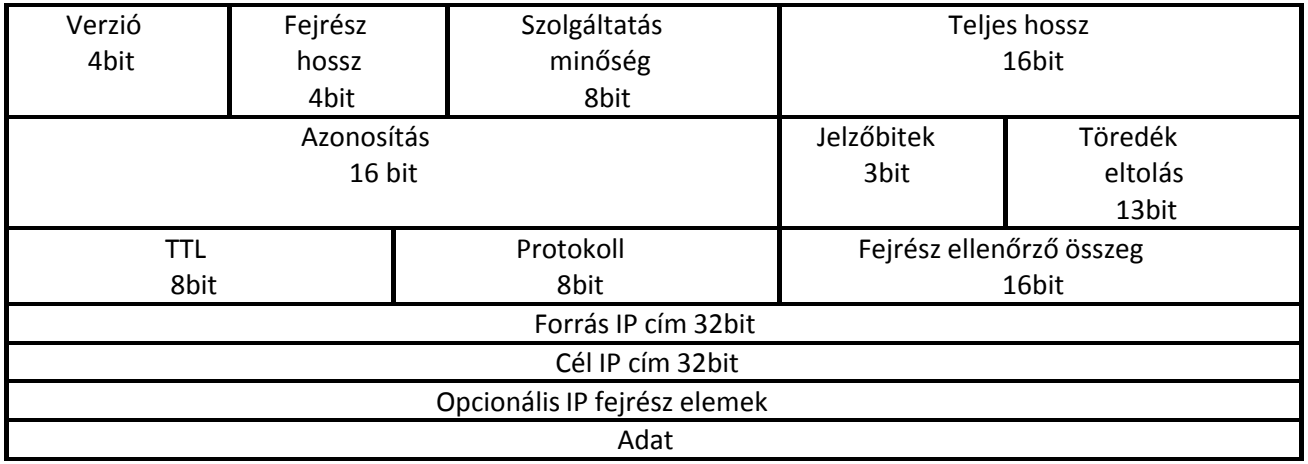
Verzió 4bit	Fejrész hossz 4bit	Szolgáltatás minőség 8bit	Teljes hossz 16bit	
Azonosítás 16 bit			Jelzőbitek 3bit	Töredék eltolás 13bit
TTL 8bit	Protokoll 8bit	Fejrész ellenőrző összeg 16bit		
Forrás IP cím 32bit				
Cél IP cím 32bit				
Opcionális IP fejrész elemek				
Adat				

IPv4 csomagformátum 2.



- Verzió (Version): 4 bit, IPv4 esetén 4
- Fejrész hossz (Internet Header Length): 4 bit, A fejrész hossza 32 bites szavakban
  - Min. 5 (160bit-es fejrész)
  - Max. fejrész méret: 15 (480bit-es fejrész)
  - Opcionális fejrész elemek megadására
- Szolgáltatás minőség (Differentiated Services): 8 bit, eltérő értelmezések, a gyakorlatban nem használják
- Teljes hossz (Total Length): 16 bit, A teljes adatcsomag (datagram) mérete byte-ban

IPv4 csomagformátum 3.



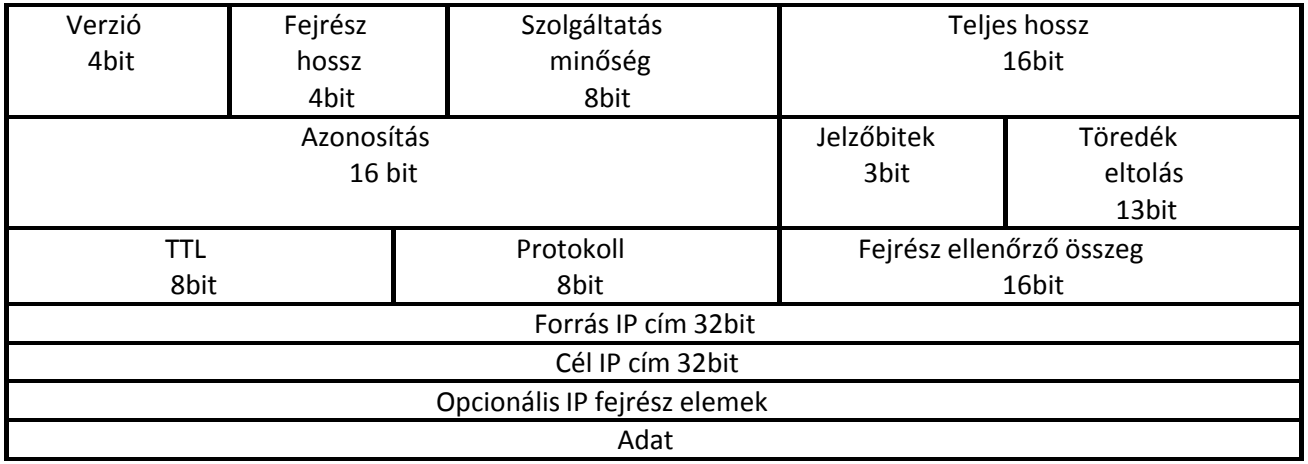
- Azonosítás (Identification): 16 bit, a csomag azonosítása, szekvenciálisan növekszik
- Jelzőbitek (Flags): 3 bit
  - Foglalt (Reserved): 0 érékűnek kell lennie
  - Tördelés tiltott (Do not fragment)
  - További töredékek jönnek (More fragments)
- Töredék eltolás (Fragment offset): 13 bit, 8 byte-os blokkokban a töredék kezdete az eredeti nem tördelt IP csomagban

IPv4 csomagformátum 4.

Verzió 4bit	Fejrész hossz 4bit	Szolgáltatás minőség 8bit	Teljes hossz 16bit	
Azonosítás 16 bit			Jelzőbitek 3bit	Töredék eltolás 13bit
TTL 8bit	Protokoll 8bit	Fejrész ellenőrző összeg 16bit		
Forrás IP cím 32bit				
Cél IP cím 32bit				
Opcionális IP fejrész elemek				
Adat				

- TTL (Time To Live): 8 bit, csomag élettartama, külön fólia van róla
  - A csomagoknak a maradék élettartamát adja meg
    - Hop count (nem valódi idő)
      - Pontosan hány útvonalválasztón haladhat még át a csomag?
    - A rendszerben kialakult útvonal választási hurkokból származó problémák kivédésére szolgál
      - Az útvonalválasztók állapotmentesek, nem tudják, milyen csomagok haladtak át rajtuk
      - A csomagok így véges számú útvonalválasztón haladhatnak csak át

IPv4 csomagformátum 4. TTL folytatás



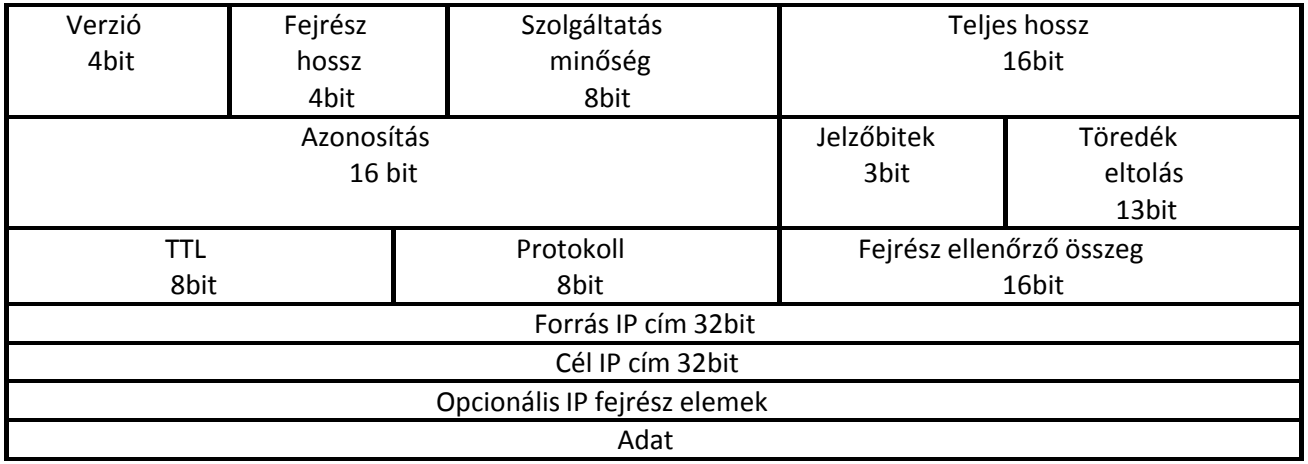
- TTL alkalmazása
  - A küldő állomás egy megadott értékre állítja be (implementáció függő)
  - Minden egyes útvonalválasztó az értéket eggyel csökkenti
    - Az IP fejrész ellenőrző összegét is módosítani kell
    - Az algoritmus okos, lényegében azt is elég eggyel csökkenteni
  - Ha az értéke 0-ra csökken, akkor a csomagot eldobja az az útvonalválasztó, ahol ez megtörténik
    - A hibáról a küldő értesítést kaphat (opcionális)
    - Ekkor egy ICMP időtúllépés üzenetet (time exceeded) küld az útvonalválasztó

IPv4 csomagformátum 5.

Verzió 4bit	Fejrész hossz 4bit	Szolgáltatás minőség 8bit	Teljes hossz 16bit	
Azonosítás 16 bit			Jelzőbitek 3bit	Töredék eltolás 13bit
TTL 8bit	Protokoll 8bit		Fejrész ellenőrző összeg 16bit	
Forrás IP cím 32bit				
Cél IP cím 32bit				
Opcionális IP fejrész elemek				
Adat				

- Protokoll: 8 bit, A szállítási réteg protokoll azonosítója, ami az IP csomagot kezeli
  - RFC 790-ben megtalálható a pontos lista
  - 1: Internet Control Message Protocol (ICMP)
  - 2: Internet Group Management Protocol (IGMP)
  - 6: Transmission Control Protocol (TCP)
  - 17: User Datagram Protocol (UDP)

IPv4 csomagformátum 6.



- Fejrész ellenőrző összeg (Header Checksum): 16 bit
  - Elég a fejrészre, a szállítási réteg védi a többit
  - A csomagtovábbítás során ellenőrizni kell, és újra kell számolni (TTL változik)
  - Speciális algoritmus az újraszámolás egyszerűsítésére
- Forrás és Cél IP cím (Destination and Source address): 2 db 32 bites mező (Részletesen beszélünk majd az IP címekről)
- Opcionális fejrész elemek:
  - Kiterjeszthetővé teszik a protokollt, sokféle van, pl. IPsec, stb.
- Adat : Ezt kapja meg az IP réteg felett réteg, a pontos protokoll a Protokoll mező alapján dől el

IPv4 címek

- 32 bites IP címek
  - Byte-onként decimális formában ponttal elválasztva írjuk
  - Pl. 152.66.252.1
- Alhálózatok (Az Internet alhálózatokból áll)
  - Korai megoldás:
    - 1. byte alhálózati cím, 2-4. byte alhálózaton belüli cím
    - Hamar elérte a korlátait
  - Osztály alapú (classful networking) alhálózat megadás
    - Különböző osztályok az 1. byte alapján
      - A, B, C, D, E osztályok, azokhoz rendelt alhálózat méret
    - 1980-as évek végére nyilvánvalóvá vált, hogy nem skálázódik az internettel

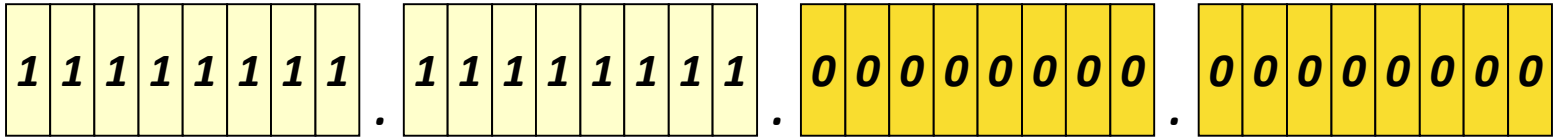
IPv4 címek, CIDR

- 1993-tól Osztály nélküli (Classless Inter-Domain Routing, CIDR) cím megadás (alhálózat lokálisan tovább osztható)
  - A cím mellé ismerni kell az alhálózati maszkot is
  - Az alhálózati maszk (subnet mask) egyeseket tartalmaz az alhálózatot azonosító részen
  - Pl. 255.255.254.0 alhálózati maszk esetén első 23 bit adja meg az alhálózatot, és az utolsó 9 bit az alhálózaton belül az állomást
  - Az alhálózatok tovább oszthatók
    - Ez nem látható a továbbosztott alhálózaton kívülről (BME, BME-MIT, tanszéki labor)
- A, B, C, D, E osztályok nem léteznek már!

Alhálózati maszk példa (BME IP címei)

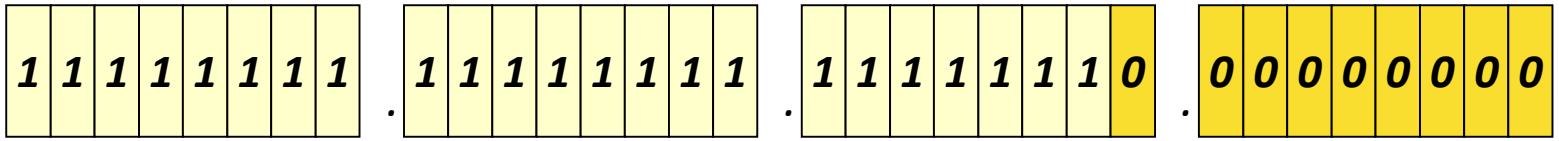
BME egyik IP címtartománya a BME-én kívülről, subnet maszk 255.255.0.0

152.66.0.0 – 152.66.255.255, 152.66.0.0/16



BME-MIT egyik IP címtartománya a BME-én belülről, subnet maszk 255.255.254.0

152.66.252.0-152.66.253.255, 152.66.252.0/23



Speciális IP címek 1.

CIDR cím tartomány	Magyarázat
0.0.0.0/8	Aktuális hálózat (csak forráscímként használható)
10.0.0.0/8	Privát címtartomány
127.0.0.0/8	Loopback cím (gépen belül)
128.0.0.0/16	Reserved (IANA)
169.254.0.0/16	Link Local cím, pl. IP autokonfig.
172.16.0.0/12	Privát cím tartomány
191.255.0.0/16	Reserved (IANA)
192.0.0.0/24	Reserved (IANA)
192.0.2.0/24	Documentation and example code
192.88.99.0/24	IPv6 és IPv4 közötti átjáró
192.168.0.0/16	Privát címtartomány
198.18.0.0/15	Hálózati teljesítmény vizsgálata
223.255.255.0/24	Reserved (IANA)
224.0.0.0/4	Multicast címek (D osztályú címek régen)
240.0.0.0/4	Reserved (former Class E network)
255.255.255.255	Broadcast

Speciális IP címek 2.

- 0.0.0.0/8 Forráscímként használható, pl. DHCP-nél a küldő gép még nem tudja az IP címét (éppen azt akarja lekérdezni)
  - Csak forráscím lehet…
- Privát címek (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16)
  - Szervezeten belül, nem kapcsolódhat az Internetre a gép direkt módon
    - Az Internet elérés megoldható
      - Network Address Translation (NAT)
      - Alkalmazási szintű proxy szerver
- 127.0.0.0/8 SW loopback interfész a gépen belüli kommunikációra
- 169.254.0.0/16 Automatikus konfiguráció, pl. Windows XP, ha nincs IP és nem kap DHCP-én keresztül
- 224.0.0.0/4 Multicast IP címek

Speciális IP címek 3.

- 240.0.0.0/4, nem osztották ki (SW + config problémák)
- Subnet broadcast cím
  - Minden lokális címbit 1-es
  - Pl. 152.66.253.255, 255.255.254.0 subnet maszk esetén
  - Csak a saját hálózatban használható
    - Más subnet-be küldve az útvonalválasztók nem továbbítják
- 255.255.255.255 Internetre kiterjedő broadcast cím
  - Az útvonalválasztók nem továbbítják
  - Súlyos túlterheléses támadást lehetne egyébként indítani
- Globális IP címek (összes többi)
  - A szervezetnek igényelnie kell egy Internet szolgáltatótól

Privát címtartományok

- Address Allocation for Private Internets (RFC1918)
  - Számos előny és hátrány származik az alkalmazásából
  - RFC1918 + network address translation (NAT) + IP masquerading = 10 évvel többet kibírt az IPv4
    - A végpontok közötti kommunikáció megnehezült
- 10.0.0.0/8
  - Tipikus felhasználás: Kisebb alhálózatokkal nagy szervezetek belső hálózatában
- 172.16.0.0/12
  - Ritkán használják
- 192.168.0.0/16
  - Tipikus felhasználás : Broadband/WI-FI router, Virtualizációs megoldások, stb.

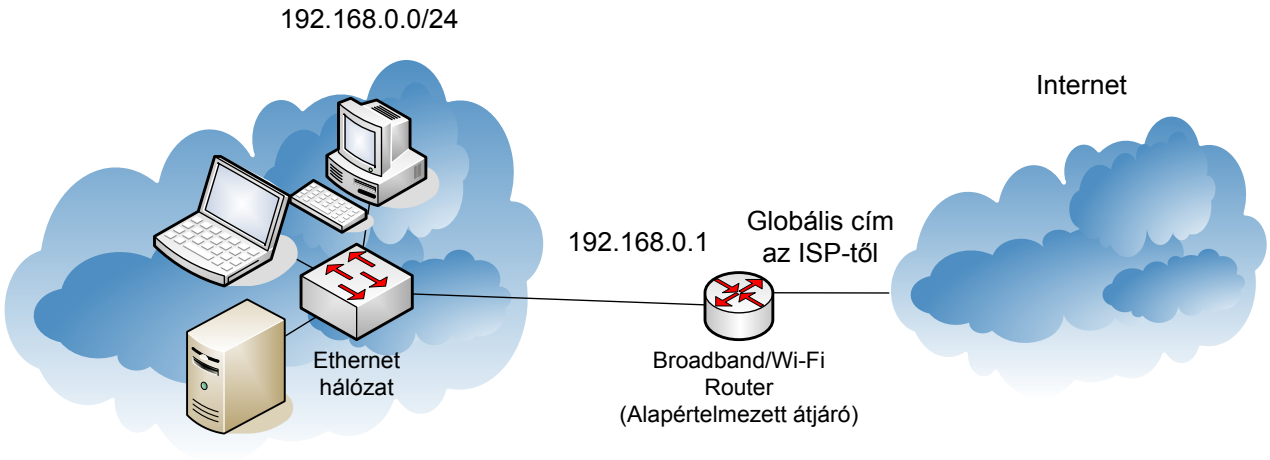
Tördelés az IPv4-ben

- Az IP csomag maximális mérete 65535 byte (min 576 byte)
- Különböző adatkapcsolati réteg technológiák maximális csomagmérete eltérő
  - Ethernet esetén 1500 byte, de pl. PPPoE esetén lehet csak 1492
  - A használt adatkapcsolati réteg technológiák csak a továbbítás során derülnek ki az útvonalválasztókban
- A csomag tördelése ezért sokszor elkerülhetetlen
  - A tördelés rossz, kerülendő
    - Az útvonalválasztó és a fogadó állomáson erőforrás igényes
    - Jelentősen növeli az overhead-et (sok fejrész)
  - Tördelés tiltott (Do not fragement) flag beállítása
    - ICMP hibaüzenettel (ICMP "can't fragment") eldobja az útvonalválasztó csomagot
  - Path MTU discovery
    - Megtalálja a maximális értéket egy távoli állomásra
- Az Ethernet egyeduralkodóvá válása miatt is ritkán szükséges
  - MTU = 1500 szinte mindig megfelelő beállítás

Network Address Translation (NAT)

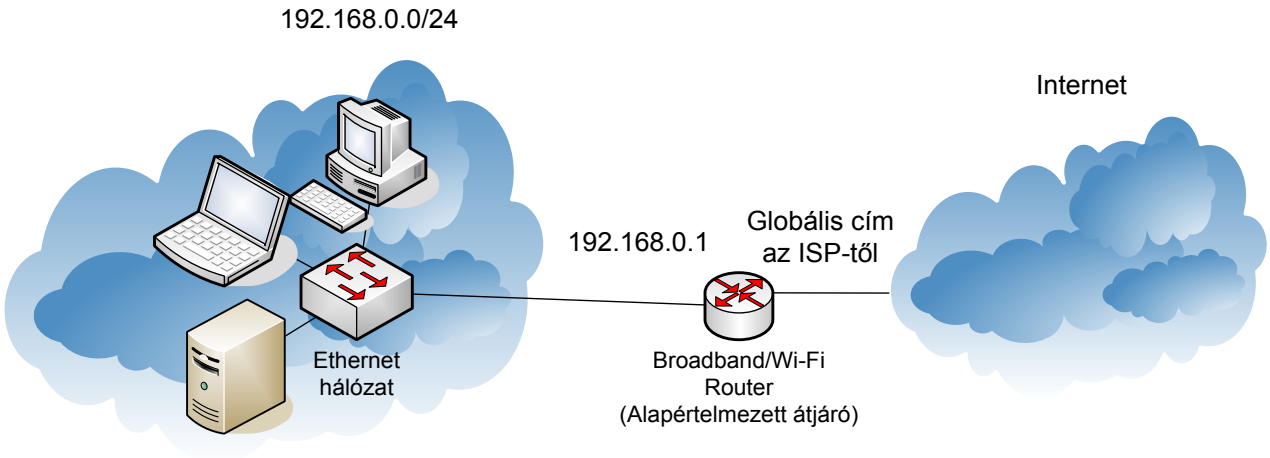
- Network address translation (NAT)
  - Az IP címek módosítása a csomagtovábbítás során
  - Sok okból lehet rá szükség
- IP masquerading
  - Egy (vagy több) nyilvános IP cím mögött elrejteni egy nagyobb privát címet használó hálózatot
  - Hibásan használják (és mi is használjuk majd) valójában a NAT fogalmat, mert nem csak hálózati hanem szállítási cím módosítás is történik…
- Lényeg TCP és UDP szállítási réteg esetén (ott van szállítási réteg cím, azaz port, majd beszélünk róla ott):
  - Belső fél azonosítása belül: iAddr:iPort
  - Belső fél azonosítása kívül: eAddr:ePort
  - Távoli fél azonosítása: rAddr:rPort
  - Egy táblázat megadja ennek az iAddr:iPort-eAddr:ePort-rAddr:rPort hármasnak a leképzését a NAT-ot végző eszközben
  - Az IPv4 és a szállítási réteg header-ben ellenőrző összegeket is újra kell számolni
- ICMP esetén az ICMP csomag típusától függ a működés

NAT példa 1.



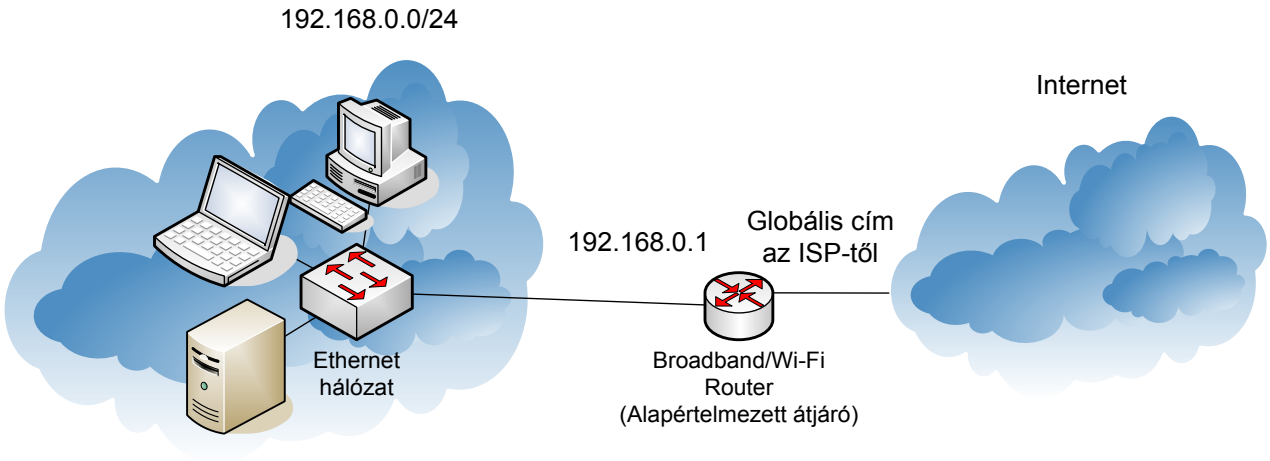
- Privát hálózat által indított kommunikáció (automatikus működés):
  - A privát hálózatban lévő állomás elküldi az információt az iAddr:iPort-ról a rAddr:rPort-ra
  - Az áthalad az alapértelmezett átjárón
  - Az hozzárendel egy eAddr:ePort-ot, ahol eAddr a globális ISP-tól kapott IP címe, ePort egy szabad eAddr-hoz tartozó TCP vagy UDP port
    - Ezt nyilvántartja egy táblázatban a későbbi kommunikáció leképzésére…
  - A csomagban minden szükséges helyen lecserélni iAddr:iPort fejrész elemeket a eAddr:ePort elemekkel
  - Úgy küldi tovább a csomagot a rAddr:rPort-ra

NAT példa 2.



- Válaszok rAddr-rPort-tól:
  - A távoli fél rAddr:rPort-ról az eAddr:ePort-ra küldi a csomagjait
  - Az áthalad az alapértelmezett átjárón, ahol a táblázatban ez az rAddr:rPort-eAddr:ePort-ra páros megtalálható (ismert felekről van szó)
  - A csomagban minden szükséges helyen lecserélni eAddr:ePort fejrész elemeket a iAddr:iPort elemekkel
  - Úgy küldi tovább a csomagot a iAddr:iPort-ra
- További viszontválaszok iAddr:iPort-ról
  - A leképzés ismert, iAddr:iPort csere eAddr:ePort-ra
- A leképzés lebontása a protokoll ismeretében (TCP FIN), vagy timeout-ra

NAT példa 3.



- Szerver az alapértelmezett átjáró mögött:
  - A szerver iAddr:iPort pároson várja a beérkező kéréseket…
  - A távoli fél rAddr:rPort-ról az eAddr:ePort-ra küldi a kéréseit
    - ePort ismert port (WWW esetén tipikusan 80, de a küldőnek mindenképpen tudnia kell)
  - A leképzés nem épülhet ki automatikusan
  - Az alapértelmezett átjáróban statikusan be kell állítani, hogy az eAddr:ePort-ra bejövő kéréseket az iAddr:iPort-ra kell küldenie
- ICMP csomagok (RFC 5508 - NAT Behavioral Requirements for ICMP)
  - Továbbítani kell, a legrosszabb esetben az ICMP üzenetből kibányászható a cél

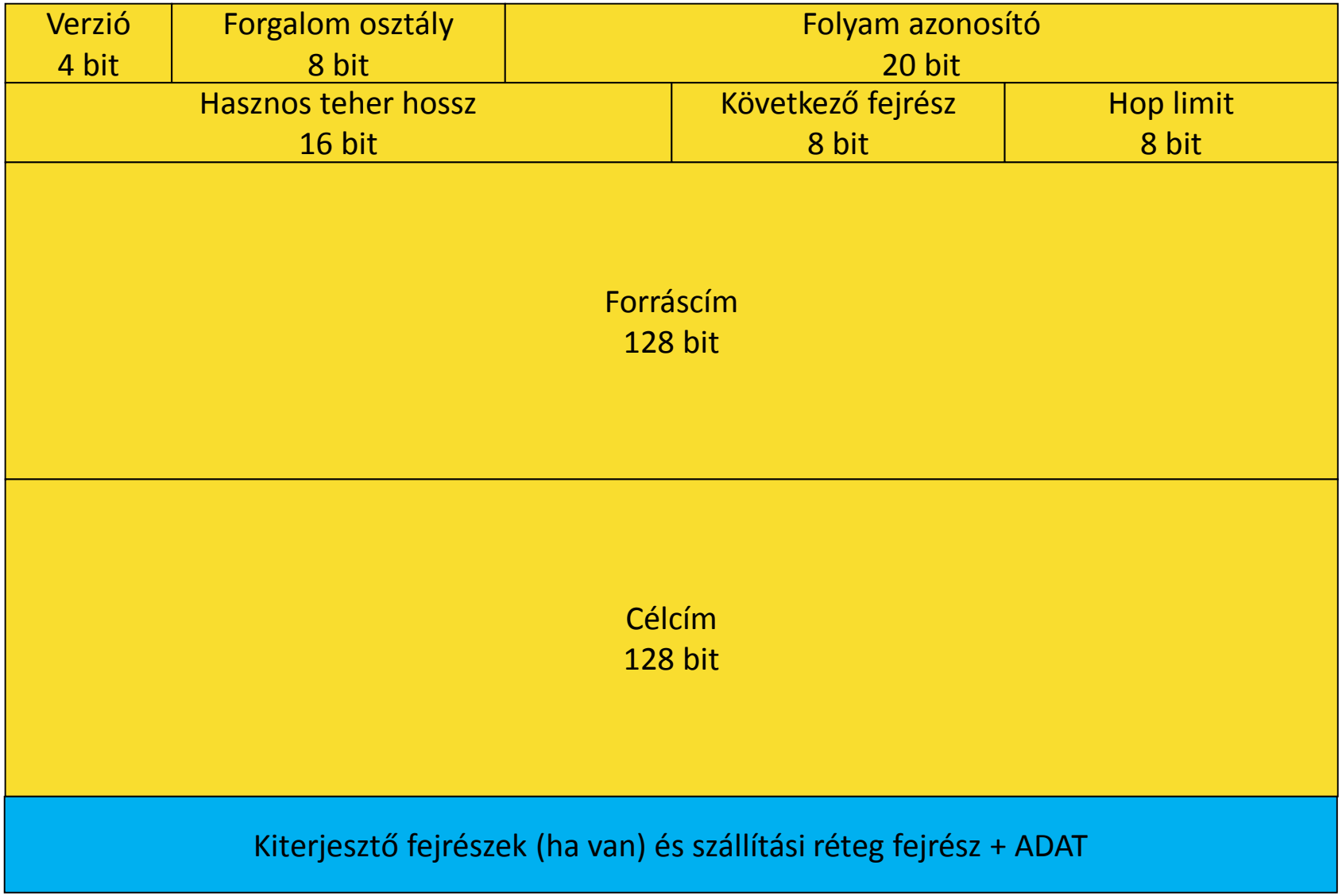
Network Address Translation (NAT)

- Problémák
  - Az Internet végpontok közötti transzparens működése (end-to-end nature) sérül
  - Bizonyos hálózati protokollok nem vagy nehezen használhatóak (pl. active mode FTP, P2P, stb.)
  - Kívülről elérhető szerverek üzemeltetése a rejtett hálózatban nehézkes
  - Lassabb, a csomagokat jelentősen módosítani kell a NAT-olás során
- Előnyök
  - Nagyobb biztonság
    - A rejtett gépek nem elérhetőek a nyilvános hálózatból speciális konfigurációs beállítások nélkül
  - Kisebb nyilvános címtartomány használat (+10év)
- Probléma: Carrier Grade NAT
  - A szolgáltatók NAT-olnak, a felhasználók már privát IPv4 címet kapnak
  - Nyilvános IPv6 címet kaphatnak, de IPv4-et nem
  - Többszörös NAT, akár háromszoros is (Virtuális gép, Broadband router, Szolgáltató)
  - A TCP/UDP portok száma korlátos, hány felhasználót lehet egyetlen IP címmel kiszolgálni?

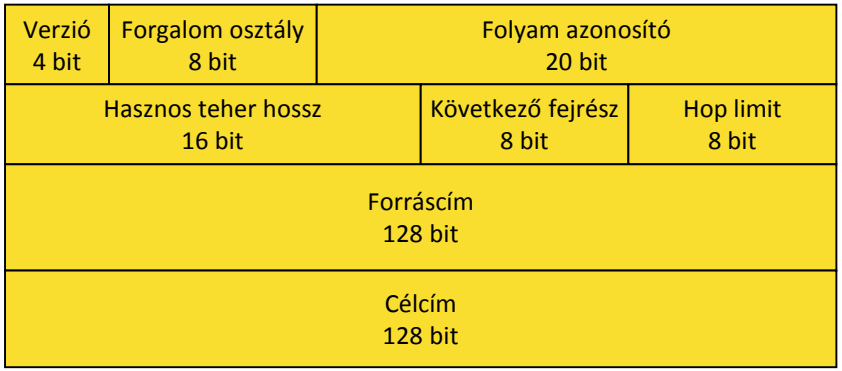
IPv6

- Az IPv4 sok szempontból elavult
  - Sokkal többet tudunk arról, hogy hogyan is kéne egy jó hálózati réteg protokollnak működnie
  - Címtér kimerült IANA szinten…
    - A NAT elterjedése adott kb. 10 év plusz időt az átállásra
    - De ezzel sérült az Internet teljes átláthatósága
  - A biztonság (küldő azonosítása és továbbítás során történő módosítás felismerése) nem megoldott IP szinten
- 1990-es évek elejétől folyt az új verzió kidolgozása
- IPv6 nevet kapta (az IPv5-öt korábban már lefoglalták másra)
  - Új, bővíthető keretformátum
  - 128 bit hosszú címek (fe80:0:0:0:200:5aff:fe9a:290c)
- Probléma: Elterjedése lassú
  - Erőforrás igényesebb (pl. hosszabb címek feldolgozása)
  - Az eszközök már támogatják, de a legtöbb alkalmazás nem
    - Eszközök: kivéve az otthoni Broadband (ADSL/kábel) routereket...

IPv6 csomagformátum



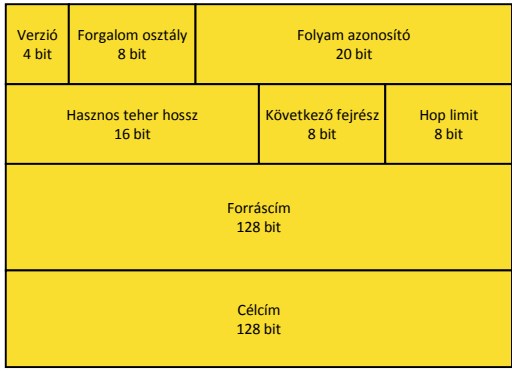
IPv6 csomagformátum részletei 1.



- Verzió (Version, 4 bit) : mindig b0110 (6) értékű
- Forgalom osztály (Traffic Class, 8 bit) : DSCP (Differentiated Services Code Point, 6 bit) és ECN (Explicit Congestion Notification, 2 bit)
  - DSCP: Forgalmi osztályok, az útvonalválasztók ez alapján lényegében máshogy kezelnék a csomagokat, többnyire nem használják napjainkban
  - ECN: Túlterhelés kezelésére, többnyire nem használják napjainkban
- Folyam azonosító (Flow Label, 20 bits) : Nem nagyon használják
  - Pl. valós idejű „kapcsolatok” azonosítására, pl. multimédia kommunikáció
  - Az útvonalválasztók speciálisan kezelhetnék ez alapján
- Hasznos teher (Payload Length, 16 bits) : Beleértve a kiegészítő fejrészt is
  - Azt a következő fejrésszel adhatjuk meg, lásd következő fólia

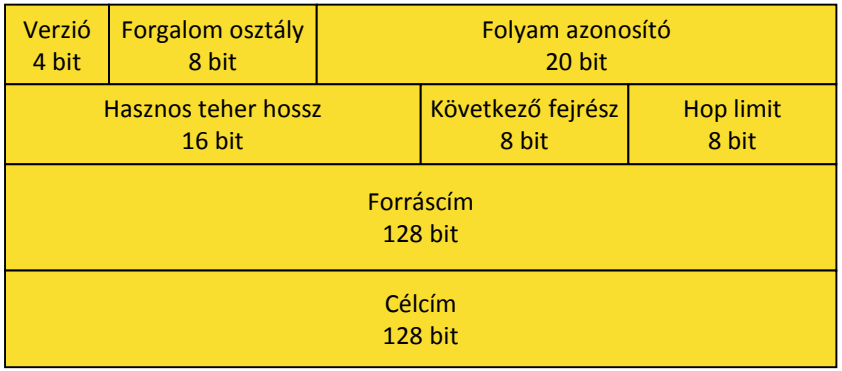
IPv6 csomagformátum részletei 2.

- Következő fejrész (Next Header, 8 bits) : A következő fejrész elem típusa, ha ilyen nincs, akkor a szállítási réteg típusa
  - Nem tetszőleges sorrendben fordulhatnak elő!



- Következő fejrész azonosítók:
  - 0, Hop-by-Hop Options , minden köztes állomásnak meg kell néznie
  - 6, TCP
  - 11, UDP
  - 43, Routing, útvonalat lehet megadni vele, pl. mobil IPv6 esetén
  - 44, Fragment, Tördelés
  - 50, Encapsulating Security Payload, végpontok közötti titkosítás (IPsec)
  - 51, Authentication Header, Hitelesítés (IPsec)
  - 60, Destination Options, A célállomás számára érdekes információk, mindig az „utolsó”
  - Lehet, hogy erről a mezőről kiderül, hogy túl rövid?
- Pl. Tördelés (Fragment : 44)
  - Tördelés „kiterjesztő fejrész” alkalmazásával lehetséges
  - Az útvonalválasztók nem tördelnek, ez a szállítási réteg feladata
  - Ha az nem lehetséges, akkor a küldő fél IP rétege is megteheti a tördelés kiegészítő fejrésszel
    - Csak a legvégső esetben történik tördelés!

IPv6 csomagformátum részletei 3.



- Hop limit
  - A „time to live” mező helyett, lényegében azonos működéssel
  - Minden útvonalválasztó eggyel csökkenti, ha 0 értékű akkor a csomag eldobásra kerül
- Célcím és forráscím : 128 bites IPv6 cím
  - Ez nagyon sok , de…
  - Nagyon pazarlóan bánnak vele szerintem
  - A felhasználók (a BME is) egy /48 címtartományt kapnak az RFC 3177 alapján
  - Aztán ezt osztják tovább, pl. a BME-MIT IPv6 címtartománya 2001:738:2001:4040::/64
  - Egy draft alapján lehetne kisebb címtartományokat is osztani a felhasználóknak, pl. /64-et
    - /64-nél kisebbet nem, mert ez a minimális alhálózat méret
- Az MTU (Maximum Transmission Unit, valójában minimum!) : 1280 byte (octet)
  - Ez IPv4 MTU 576 byte (octet)

IPv6 címek

- 2001:0db8:85a3:0000:0000:8a2e:0370:7334 (WIKI állatorvosi ló cím)
  - Hexadecimális formában a 128 bites cím 16 bites blokkjai
  - Kezdő 0-ák elhagyhatók, egy tisztán 0-ákból álló rész ::-re helyettesíthető
  - Legrövidebb forma : 2001:db8:85a3::8a2e:370:7334
- 3 fajta cím: Unicast, Multicast, Anycast
  - Nincs broadcast cím: link-local multicast group ff02::1
- Az alhálózatok CIDR szerint
  - 2001:738:2001:4040::/64

Unicast és anycast címek

- Egyelőre az IPv4 kezdeti időszakához hasonló statikus címstruktúra
  - Nem tudjuk, mit hoz a jövő, sokan bírálják ezt a sémát…
  - Egyes szolgáltatók már osztanak /56 tartományokat is végfelhasználóknak
- Az Interface azonosító az alhálózatban kerül megadásra
- Lehetőségek:
  - MAC címből automatikus generálva az EUI-64 formátum alapján
    - A Windows nem szabványos, registry módosítással tehető azzá
  - DHCPv6 szerverrel dinamikusan vagy statikusan kiadva
- Link local address (nem továbbítják a routerek)
  - fe80::/64 (ez a 169.254.0.0/16 IPv4 esetén)
  - Mindig lennie kell, még ha van globális IPv6 cím is
  - A 48 bites MAC címből hozzák létre (Modified EUI-64)

Az unicast és anycast IPv6 címek szerkezet

48 bit	16bit	64 bit
Routing prefix	Subnet address	Interface identifier

Stateless address autoconfiguration (SLAAC)

- Neighbor Discovery Protocol
  - Minden működéshez szükséges információ beállítható
  - Az ICMPv6 protokollt használja
    - Az ARP-ét, az ICMP-t, és sok más protokollt is helyettesít
  - Funkciók (rengeteg)
  - Automatikus konfiguráció
    - Address autoconfiguration, Next-hop determination, Router discovery, Prefix discovery, Parameter discovery (MTU), Recursive DNS Server (RDNSS)
  - Hibakezelés
    - Neighbor unreachability detection (NUD), Duplicate address detection (DAD), Redirect
  - Alapvető működés
    - Address resolution (IP cím és MAC cím megfeleltetés, ha szükséges)

Speciális IPv6 címek

- ::/128, nem specifikált cím, 0.0.0.0 az IPv4-ben, amikor a host még nem tudja a címét
- ::1/128, loopback cím, gépen belüli kommunikációra
- fe80::/10, link local address
- fc00::/7, privát IPv6 címtartomány (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 IPv4-ben)
- Multicast címek
  - ff01::1 and ff02::1, interface local (1) és node local (2) multicast cím
  - ff01::2, ff02::2, és ff05::2, all routers címek, (5) a site local cím
  - ff02::1:ff00:0/104, valódi multicast címek (Solicited-node multicast)
- IPv4 és IPv6 közötti átmenet során használt címek (fontosak)
  - ::ffff:0:0/96, transzparens IPv4 cím leképzés szerver oldalon a szállítási réteg felé
  - Teredo : 2001::/32 (NAT mögött is működik)
  - 6to4 : 2002::/16 (szélesebb körben használt, de nem megy NAT-tal)
  - 6to4 IPv4 – IPv6 címleképzés:
    152.66.253.251 gép IPv6 címe 2002:9842:fdfb::9842:fdfb, mivel 152=h98, 66=h42, 253=hfd, 251=fb

IPv6 beágyazott rendszerekben?

- Internet of Things esetén jól jönne
  - Sok cím
  - Teljes hálózati transzparencia
- Hátrány
  - Akarom, hogy kivülről minden el lehessen érni?
  - Erőforrásigény?
- Konkrétumok
  - 6LoWPAN : IPv6 over Low power Wireless Personal Area Networks (RFC 4944)
    - IEEE 802.15.4 alapú alsóbb rétegeken (ZigBee versenytársa)
    - Encapsulation and header compression
    - Elfér és fut egy ATmega1281-ben
      - 128K Flash, 8K RAM, 4K EEPROM, n*10 Kbit/s sebesség

Vissza az IPv4 világába

IP Ethernet felett

- Hogyan kerül beágyazásra egy IP csomag egy Ethernet keretbe
  - IEEE802.3 szerinti beágyazás LLC fejrésszel
    - Opcionálisan megvalósítható
    - Ritkán használják (a „soha” jobb szó)
  - EtherType szerinti beágyazás (kötelezően megvalósítandó)
    - EtherType Field:
      - `0x0800` Internet Protocol, Version 4
      - `0x86DD` Internet Protocol, Version 6
      - Redundancia! Megvan a csomagban is a verzió információ
    - Az első EtherType-ot az IPv4 kapta
- Hogyan feleltetjük meg az IP címeket MAC címnek?
  - 1. megoldás: Mindig Ethernet broadcast-ot küldünk
    - Nagy erőforrás igény, nem túl okos ötlet
  - 2. Statikus megfeleltetés
    - Nem jól karbantartható, hibalehetőség nagy
  - Speciális protokoll a megfeleltetés kezelésére

MAC és IP cím megfeleltetés

- Az Address Resolution Protocol (ARP)
  - IPv4 segédprotokoll
    - Ethertype: `0x0806`
  - Más hálózati réteg protokollok is használják
  - Más adatkapcsolati rétegeket is támogat
- Az alhálózatba tartozó IP címre küldeni kíván egy gép
  - ARP kérést küld (Who has x.y.w.z?)
    - Broadcast Ethernet címre (minden állomás megkapja)
      - Megadja benne a keresett gép IP címét
    - Az ARP adatmező része minden címet tartalmaz
      - Ne kelljen alsóbb rétegekbe lelátni az ARP megvalósításban
  - Az adott IP címet használó gép ARP választ küld (unicast cél)
    - A válaszban megadja az Ethernet címét is
  - Az ARP kérések eredményeit cache-eli a kérést küldő
    - Ne kelljen minden IP csomaghoz ARP kérést küldeni
    - Adott idő után elfelejti a nem használt bejegyzéseket
- Alhálózaton kívülre: Alapértelmezett átjáró MAC címére küld

ARP specialitások

- ARP Proxy
  - Egy speciális szolgáltatás, amely más állomások helyett annak adataival válaszol az ARP kérésekre
  - Beágyazott rendszerekben felmerül
    - Nem kell minden állomásban ARP implementáció
    - Nem jó ötlet termék/rendszer szinten:
      - Ritka megoldás, nem ismerik
      - A beágyazott node-ok nem tudnak kommunikációt kezdeményezni, vagy azokat statikusan kell konfigurálni
- Gratuitous ARP vagy ARP probe
  - Speciális kérés, az a baj, ha érkezik rá válasz
  - A TCP/IP protokoll készlet új IP cím használata előtt ezzel teszteli, hogy azt használja-e egy másik állomás
    - Windows XP használja, Linux nem
- Régen volt egy Reverse ARP nevű protokoll is
  - Nem elosztott rendszer volt, hanem egy szerver szolgáltatás
  - Ma már nem használják, helyette a BOOTP/DHCP van

ARP vizsgálata OS-ben

- Az `arp` paranccsal érhető el az ARP cache tartalma
  - Linux-on és Windows-on is!
  - Az `arp -a` parancs listázza a cache tartalmát (OS független)
  - Bejegyzések vagy az egész cache törölhető (`arp -d`)
  - Statikus (permanens) bejegyzések is elhelyezhetők
  - Többnyire nincs rá szükség, csak hibakeresés során használjuk

IPv4 beállítások

- Milyen információk szükségesek, hogy egy IPv4-et használó állomás kommunikálni tudjon?
- Minimálisan szükséges:
  - IP cím
  - Alhálózati maszk (subnet mask)
- Alhálózaton kívüli kommunikációhoz kell
  - Alapértelmezett átjáró címe (defaul gateway)
- Névfeloldással kapcsolatos információk: (később tárgyaljuk)
  - DNS szerverek IP címe
    - IP cím kell...
  - DNS név helyettesítésnél használható rész stringek
    - Lásd majd DNS bemutatásánál
  - WINS szerver IP címe (Microsoft specifikus)
    - Működési problémák a file és printer megosztásnál, ha nincs megadva
- Statikusan megadva (config) vagy DHCP

IPv4 beállítások beágyazott rendszerekben

- Nincs vagy minimális a felhasználói interface (UI)
  - Statikus konfiguráció bevihető az UI-n
    - 4 gomb + 2x16 alfanumerikus LCD UI esetén egy rémálom felhasználó és programozói oldalról is
    - Soros porton egy parancssoros interfészen megadható
  - Automatikus konfiguráció
    - Nem szabványos megoldások
    - DHCP protokoll kliens megvalósítása és DHCP szerver üzemeltetése
  - A statikus-automatikus konfiguráció közötti választási lehetőséget is meg kell adni
    - Default : DHCP talán a legjobb megoldás
  - A statikus konfigurációt menteni kell nem felejtő memóriába

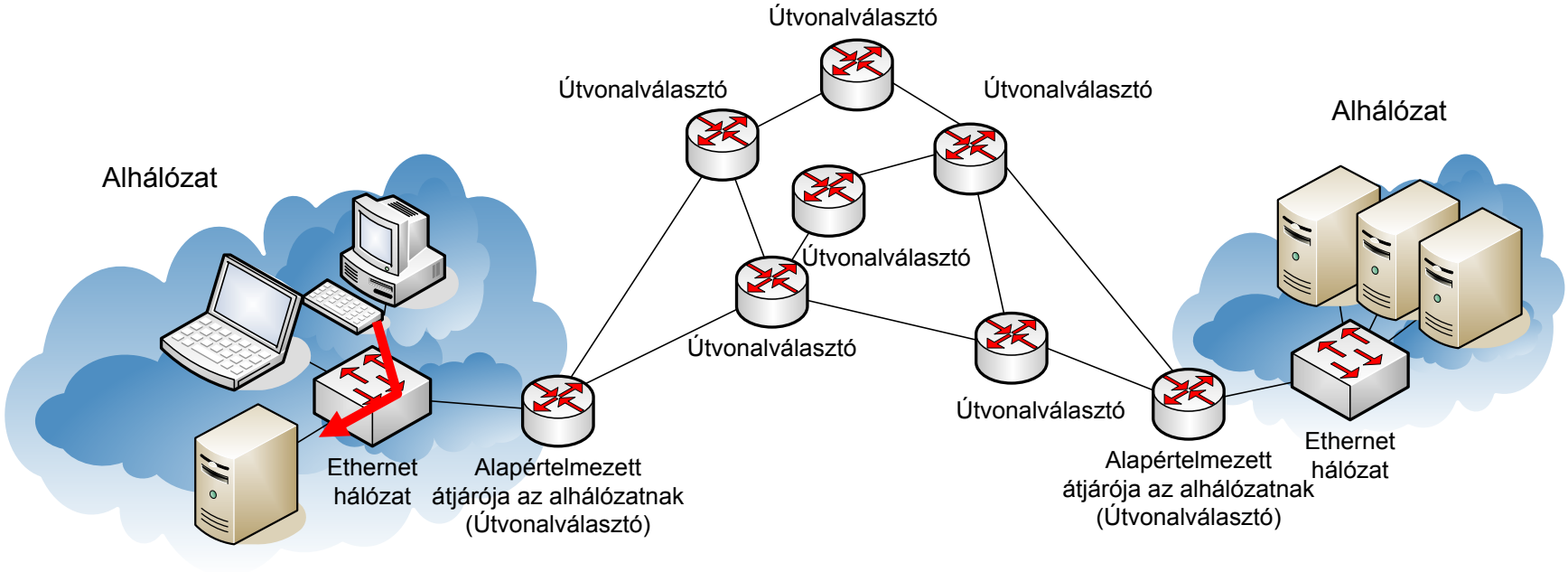
Kimeneti útvonalválasztó tábla

- Az állomásokon többnyire egynél több IP interfész van
  - Külső Ethernet interfész
  - Loopback interfész (beágy. rendszerben ez sem feltétlenül)
- Hogyan dől el, hogy melyik interfészt használja küldés során?
  - Kimeneti útvonalválasztó tábla
  - Az nagygépes rendszerekben automatikusan beállításra kerül
  - A `route` paranccsal kezelhető
  - Listázás „`route print`” Windows alatt
  - Listázás „`route`” Linux alatt (nem kell paraméter)

A route parancs kimenete

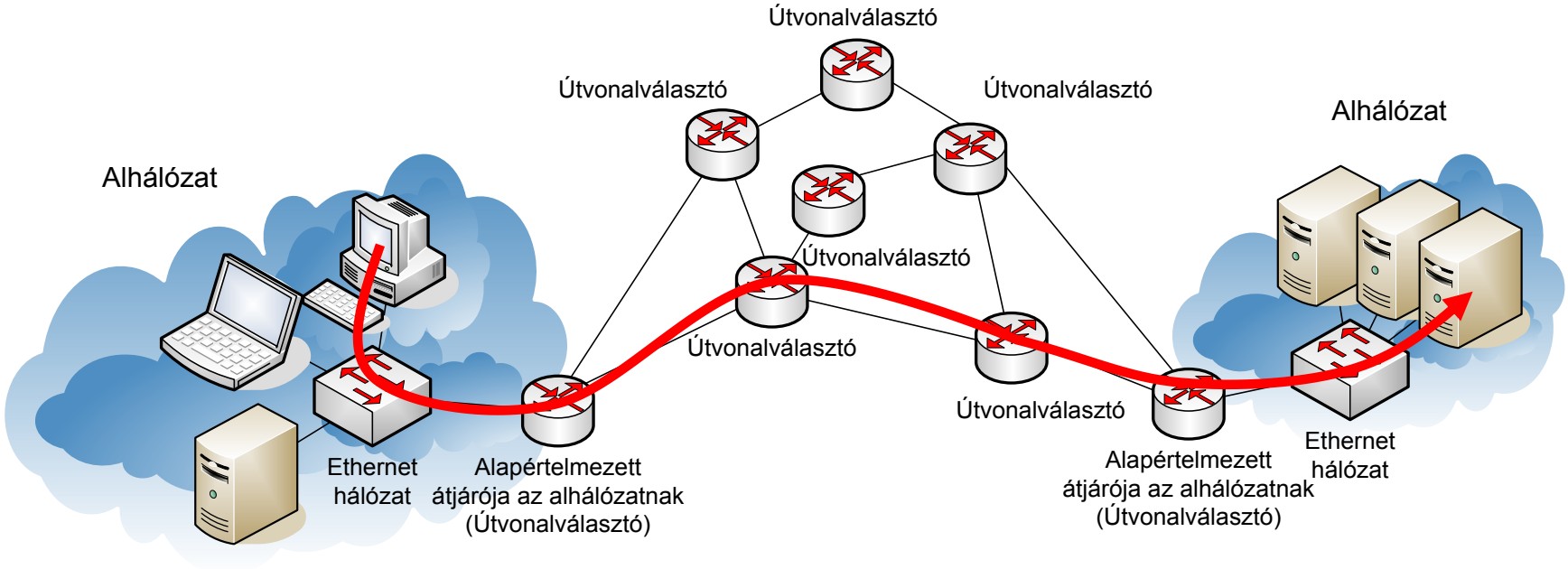
- A sorok mezőinek az értelmezés:
  - Network destination és Netmask: A címtartomány megadás az adott sorra
    - ha a küldött célcím illeszkedhet rá, akkor ez az útvonal használható/él
  - Gateway: Az alapértelmezett átjáró az adott tartományra
    - Windows Vista és 7 : On-link: Nincs szükség alapértelmezett átjáróra, a lokális hálózatba kell küldeni…
  - Interface: A gép adott IP című interfészén megy ki a csomag
  - Metric: Ha ugyan abba a címtartományba több útvonal is él, akkor ez alapján választ (kisebb a jobb)

IP kommunikáció alhálózaton belül



- Direkt módon megcímezhető a cél az adatkapcsolati rétegben
  - Ez a kimeneti útvonalválasztó táblából kiderül…
- IP és MAC szinten is a gépnek küldjük az információt
  - ARP kérés a célcímre, ha az még nincs bent az ARP cache-ben…

IP kommunikáció alhálózaton kívülre



- Csak IP szinten érhető el a célgép (kimeneti útvonalválasztó tábla)
  - Nincs az alhálózatban, IP szinten a cél IP címre küldjük a csomagot
  - Ethernet szinten az alapértelmezett átjárónak küldjük
- ARP az alapértelmezett átjáróra (cache)
- A köztes útvonalválasztók továbbítják

Útvonalválasztás

- Milyen információ alapján döntenek az útvonalválasztók?
  - „Routing table” alapján
    - Mi van benne?
      - A releváns alhálózatok melyik interfészen érhetőek el
      - Multicast információ külön kerül kezelésre
    - Tábla mérete erősen alkalmazásfüggő
  - Hogyan frissül a tábla
    - Statikus útvonalválasztás (admin megadja)
      - A hálózat szélén működik (alhálózat v. Internet)
    - Dinamikus: Routing protokollok
      - Alkalmazási réteg protokollok a „routing table” karbantartására
      - Internet szolgáltatókon, nagy szervezeteken belül és közöttük

Útvonal vizsgálata

- Hogyan tudhatjuk meg felhasználóként a gépünk és a célállomás közötti útvonalat:
  - Traceroute, tracert (gyakorlat során majd megnézzük)
    - IPv4/IPv6 TTL/HOP_limit mező innovatív felhasználása
    - Nem tökéletes, de ennél többet felhasználóként nehéz megtudni

ICMP protokoll

- Internet Control Message Protocol (ICMP), van ICMPv6 is
  - Vezérlő és hibajelző üzenetek
  - Az IP segédprotokollja (az IP felett, de nem a szállítási rétegben)
    - IP csomagokba van ágyazva
    - Nem megbízható (elveszhet)
    - 8 byte hosszú fejrész
- Üzenet típus, és azon belül hiba/üzenet kód
  - 42 típus (type), azon belül több altípus (code)
  - Fontos típusok
    - ICMP Echo request és replay (ping program)
    - Célállomás elérhetetlen (Destination Unreachable)
    - Időtúllépés (Time Exceeded)

ICMP echo (ping program)

- Kérés ICMP üzenet a célállomásnak:
  - Típus: 8 - Echo Request, Kód: 0
- A célállomás válaszol:
  - Típus: 0 - Echo Reply, Kód: 0
    - Ha elérkezik hozzá az üzenet
    - Ha a szükséges ICMP echo szolgáltatás fut
      - Többnyire kernel szintű
    - Ha a tűzfala nem tiltja
- A válasz megérkezik a kérést küldőhöz
- Mi tudunk meg belőle?
  - Elveszett üzenetek aránya tájékoztat a megbízhatóságról
  - Válaszidő számítható (round trip time)
    - Több üzenetből még válaszidő ingadozás is
  - ICMP üzeneteket használ, más típusú üzenetekre a rendszer máshogy válaszol (korlátozott információ)
    - Pl. Az ICMP letiltva egy működő WEB szerveren...

IGMP protokoll 1.

- Internet Group Management Protocol (IGMP)
  - Egy adott multicast IP címet használó állomások csoportja az u.n. Host Group
  - A Host Group-ba tartozó állomások minden ilyen multicast címre küldött információt megkapnak
- Multicast címtartományok kezelésére IPv4-ben
  - Útvonalválasztók küldik a kérdéseket (Membership Query Message)
    - Ezzel jelzik, hogy képesek mutlicast üzeneteket kezelni
  - Egy adott multicast cím iránti érdeklődését az állomás az útvonalválasztónak a Membership Report Message nevű IGMP üzenettel jelenti be
  - Az útvonalválasztón fut egy IGMP-t kezelő szolgáltatás

IGMP protokoll 2.

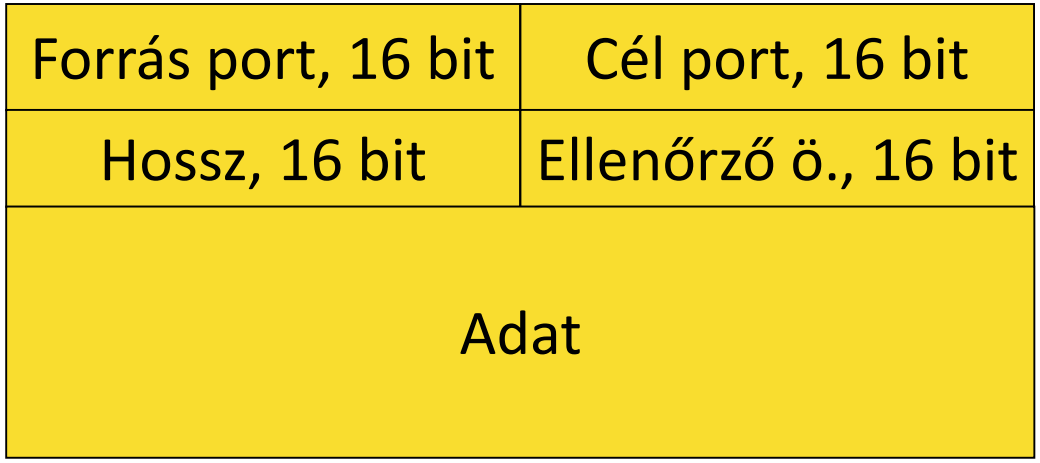
- A multicast IP címek le vannak képezve Ethernet címekre
  - A 01:00:5e gyártóazonosítót használják, a 23. bit 0, 23 bit marad a 28 bites IP címtartomány leképzésére (nem egyértelmű a leképzés)
  - IGMP snooping az Ethernet kapcsolókban, hogy ne minden host kapja meg a multicast tartalmat (erőforrás gazdálkodás)
- Beágyazott rendszerekre nem jellemző
  - Pedig a Publish-Subscribe pattern alapú elosztott beágyazott rendszerek esetén ideális lenne
  - Túl bonyolult…

Szállítási réteg protokollok

- Az IPv4 által nyújtott szolgáltatás
  - Csomagkapcsolt
    - A csomagok sorrendje felcserélődhet
    - Egy csomag több példányban is megérkezhet
  - Nem megbízható
    - Csomagok elveszhetnek, vagy megsérülhetnek
- A szállítási réteg feladata
  - Az alkalmazási réteg számára különböző szolgáltatások nyújtása a hálózati réteg felhasználásával
    - Kapcsolat orientált és csomag alapú
    - Megbízható és nem megbízható
    - A szolgáltatásokhoz egy szolgáltatás specifikus API is tartozik
    - A kernel-alkalmazás határ is itt található a modern implementációkban
  - Alkalmazások címzésének megoldása a gépen belül
    - Egy gépen belül nagyszámú alkalmazás akarja a hálózati szolgáltatásokat használni párhuzamosan
      - Melyik alkalmazástól érkezik az információ
      - Melyik alkalmazásnak küldjük az információt
    - Egy kommunikáló alkalmazás párt két IP cím (két fél címe) és két port (az alkalmazások gépspecifikus portjának címe) azonosít

User Datagram Protocol, UDP

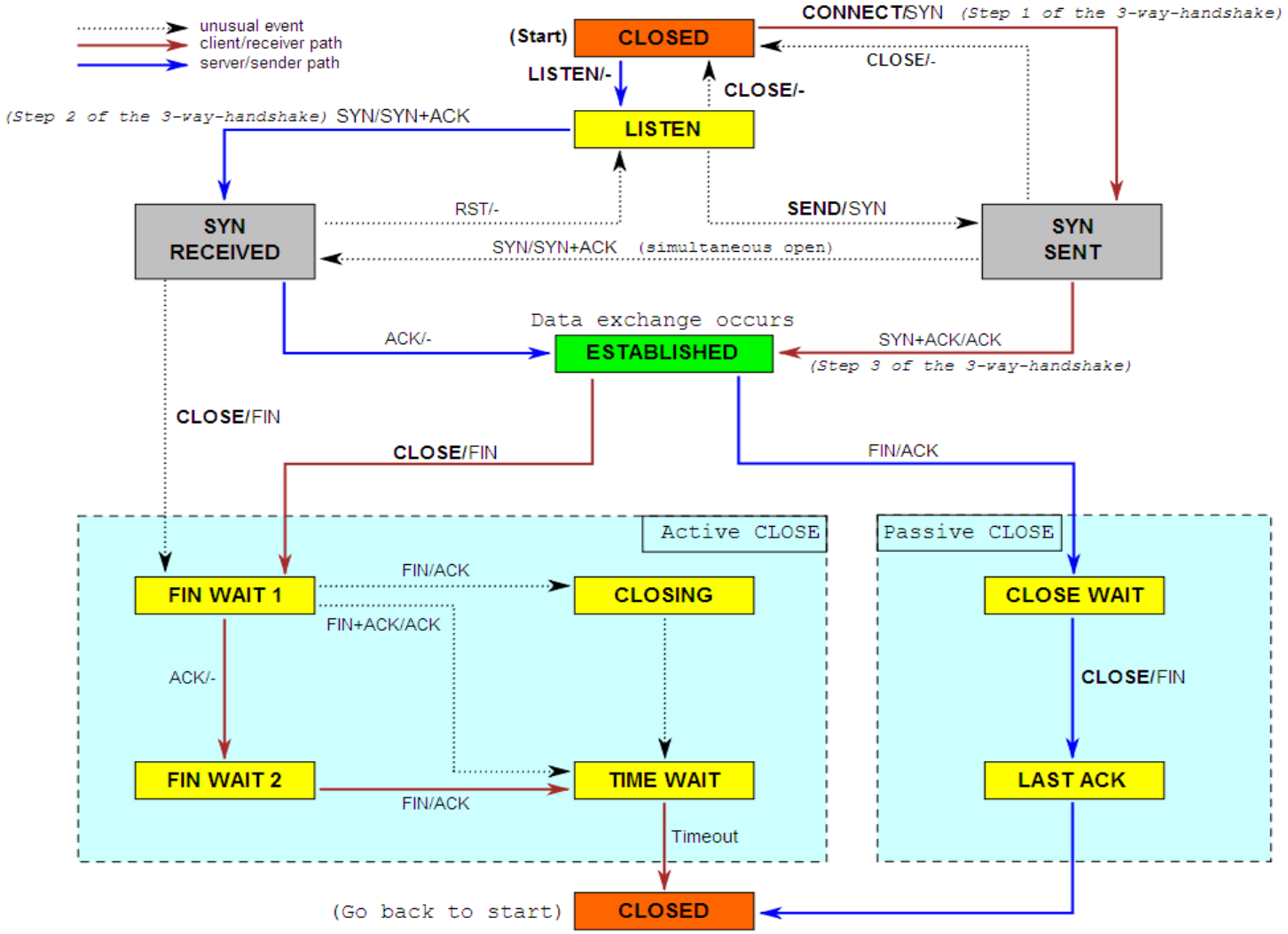
- Egyszerű, nem megbízható, csomag alapú szolgáltatás
  - Lényegében az IP szolgáltatás direkt módon becsomagolva az alkalmazásoknak (minimum service)
- 16 bites alkalmazás portok
  - 2 IP cím és 2 port azonosítja a létrejövő folyamot
  - UDP socket programozási interfész a programozó számára
  - Adott lokális portra csatlakozó állomás csomagokat küld és fogad
- 16 bites ellenőrző összeg
  - IPv4: opcionális, de erősen ajánlott, IPv6: kötelező
  - Részben az IP fejrészre (IP címekig) is kiszámolják (pszedudó header)



Transmission Control Protocol ,TCP

- Megbízható, kapcsolat alapú, adatfolyam szolgáltatás két állomás között
  - Kapcsolatfelvétel és lebontás
  - Hibakezelés (timeout és újraadás)
  - Forgalomszabályozás, torlódásvezérlés
  - Nagysebességű és interaktív adatátvitel támogatása
  - Socket interfész, de itt folyam jellegű (gyűjt az adó oldalon)
    - RS232 szerű full-duplex byte folyam a programozó számára
- Nagyon összetett protokoll
  - IP csomagokban küld TCP szegmenseket (40 byte-os fejrész + opciók)
  - A fejrésszel és működés részleteivel nem fogunk foglalkozni
  - 16 bites forrás és cél port itt is az alkalmazások azonosítására
    - A szerverek csatlakoznak majd hallgatóznak (listen) egy lokális porton, a kliensek csatlakoznak hozzá távolról (connect)
    - Két IP cím és két port azonosít egy TCP folyamot
  - Bonyolult állapotgép a kapcsolat menedzselésére

TCP állapotgép



http://en.wikipedia.org/wiki/File:Tcp_state_diagram_fixed.svg

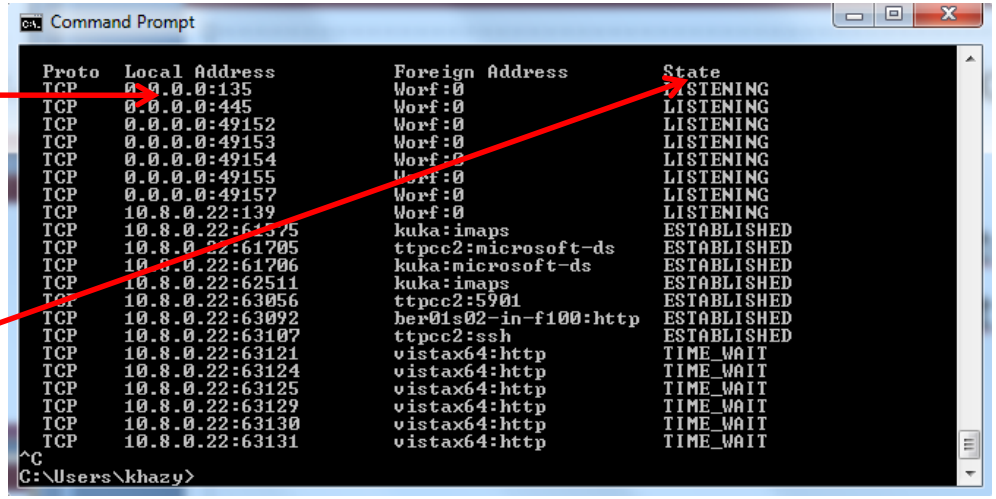
Netstat

- A meglévő TCP kapcsolatok és azok állapotának kiírása
  - Alapesetben a teljes
  - Netstat –a esetén még a Listening állapotban lévő TCP, és az összes UDP nyitott socket-et is kiírja
  - -b : kiírja a socket-et használó programot
  - -n : numerikus formában ír ki mindent

IP:port

Ha tudja, DNS névvel és ismert port névvel (pl. HTTP) írja ki

Kapcsolat állapota



TCP kapcsolatfelvétel és lebontás

- Kapcsolatfelvétel: 3 utas kézfogás
  - Kapcsolatfelvételt kérő küld egy SYN csomagot (kapcsolatfelvétel kérése)
  - A másik fél küld egy SYN-ACK csomagot (kapcsolatfelvétel elfogadása)
  - Az első fél ismét küld egy ACK-t (a kapcsolatfelvétel elfogadásának elfogadása)
- Adatküldés és fogadás
- Kapcsolatlebontás: 3 utas kézfogás többnyire
  - A lebontást kezdeményező küld egy FIN csomagot
  - A másik fél küld egy FIN-ACK csomagot
  - A lebontást kérő fél küld egy ACK csomagot
  - Max. 4 utas lehet (keresztbe küldenek FIN-t)
  - Egy kapcsolat lehet félig nyitva (nem sok értelme van)

TCP hibakezelés

- Sorszám (sequence number) azonosítja a byte-okat
  - A kapcsolatfelvételkor megállapodnak felenként egy kezdeti sorszámban (ISN)
  - Az ISN-eknek véletlennek kell lennie
- Nyugtázás (ráültethető az adatküldést végző csomagra)
  - Összesített nyugta (cumulative acknowledgment) az első byte sorszáma, amit nem kapott meg
    - A kérdéses byte előtt mindent megkapott a vevő
  - Szelektív nyugta (selective acknowledgment) opcionális
    - Jobb teljesítmény
    - Összetett, nem foglalkozunk vele
- Az adott időn belül nem nyugtázott részeket az adó újraadja
- Hibamodell:
  - Elsődleges hiba ok: csomagvesztés a rendszer túlterhelése miatt
    - A gyakorlatban igaz vezetékes vonalakon
    - Vezeték nélküli esetben sajnos nem igaz
  - Az ellenőrző összeg csak 16 bites, de mivel az adatkapcsolati réteget is védik, ezért többnyire a hibák detektálhatóak

TCP folyamszabályzás

- A rendelkezésre álló sávszélesség és állomás erőforrások optimális kihasználása (Established állapotban)
  - Torlódás miatti hibák elkerülése
    - A hálózat túlterhelődik
    - Az állomásoknak feleslegesen kell kezelniük a torlódás miatt előálló hibákat
- Folyamszabályzás a vevő túlterhelésének elkerülésére
  - Csúszóablakos (sliding window) folyamszabályzást alkalmaz:
    - A nyugtában a vevő megadja, hogy még hány byte fogadására képes tárolóterülettel rendelkezik
    - Ha ez 0, akkor az adó leáll az adással
    - Ha közel van 0-hoz akkor nem érdemes küldeni (nagy overhead), Nagle algoritmust használja ilyenkor
  - Az ablakméret erősen befolyásolja a teljesítményt
    - Sávszélesség és késleltetés függő
    - Lesz még szó róla (Bulk data transfer)

TCP torlódás menedzsment

- Torlódásvezérlés (hibamodell: a csomagvesztés oka torlódás)
  - TCP slow start (valójában exponenciális)
    - Hamar csomagvesztést okoz valamilyen rendszerrész túlterhelésével
  - Ettől kezdve implementáció függő az algoritmus működése
    - Az algortimusok kompatibilisek egymással (TCP opciókkal pl.)
    - Többnyire mérik a késleltetést (round trip time)
  - Gondok vezeték nélküli csatornán jelentkeznek
    - A hiba oka nem torlódás, hanem a csatorna nagy hibaaránya
    - A TCP visszavesz a sebességből
    - A jó stratégia a tovább kísérletezés lenne

TCP bulk és interaktív kommunikáció

- A TCP képes nagysebességgel nagymennyiségű adatot továbbítani (bulk data transfer)
  - Pl. nagy file letöltése (HTTP, FTP)
  - Ehhez sok puffer (window) kell
    - Egyébként az adó nem tud folyamatosan küldeni, mert nem kap ACK-t
    - Sávszélesség*késleltetés byte tároló terület
    - Windows scaling (65,535 byte helyett max. 1 Gbyte)
    - Windows XP és Windows 7 esetén ez nagyon szépen látható
      - XP: nagy késleltetésű és sávszélességű vonalakon kb. 20-30 Mbps érhető el
      - Win7: igazoltan 100 Mbps feletti sebesség érhető el
- A TCP képes interaktív kommunikációra
  - Pl. Telnet, SSH, Remote GUI (X11, RDP)
  - Ez nagyon eltérő működést igényel, mint a bulk data transfer
    - Késleltetésre kell optimalizálni
    - Azonnali információ továbbítás
  - PSH flag a TCP stack ürítésére (ne gyűjtse az infót)
  - URG flag, Out of Band Data, nem használják

TCP és UDP összehasonlítása

TCP

- Kapcsolat orientált
- Megbízható
- Pont-pont
  - Unicast
- Folyamszabályozás és torlódásvezérlés
- Összetett
- Erősen optimalizált
- Erőforrás igényes (CPU és memória is)
- Késleltetés ingadozik hiba esetén (újraadás)
- Kétirányú byte stream interface

UDP

- Üzenet orientált
- Nem megbízható
- Multicast és broadcast is használható
- Nincs korlátozás, az adótól függ
- Egyszerű
- Nincs mit optimalizálni
- Nem igényel sok erőforrást
- Nincs újraadás, az adat hiányzik (multimédia...)
- Csomag alapú interface

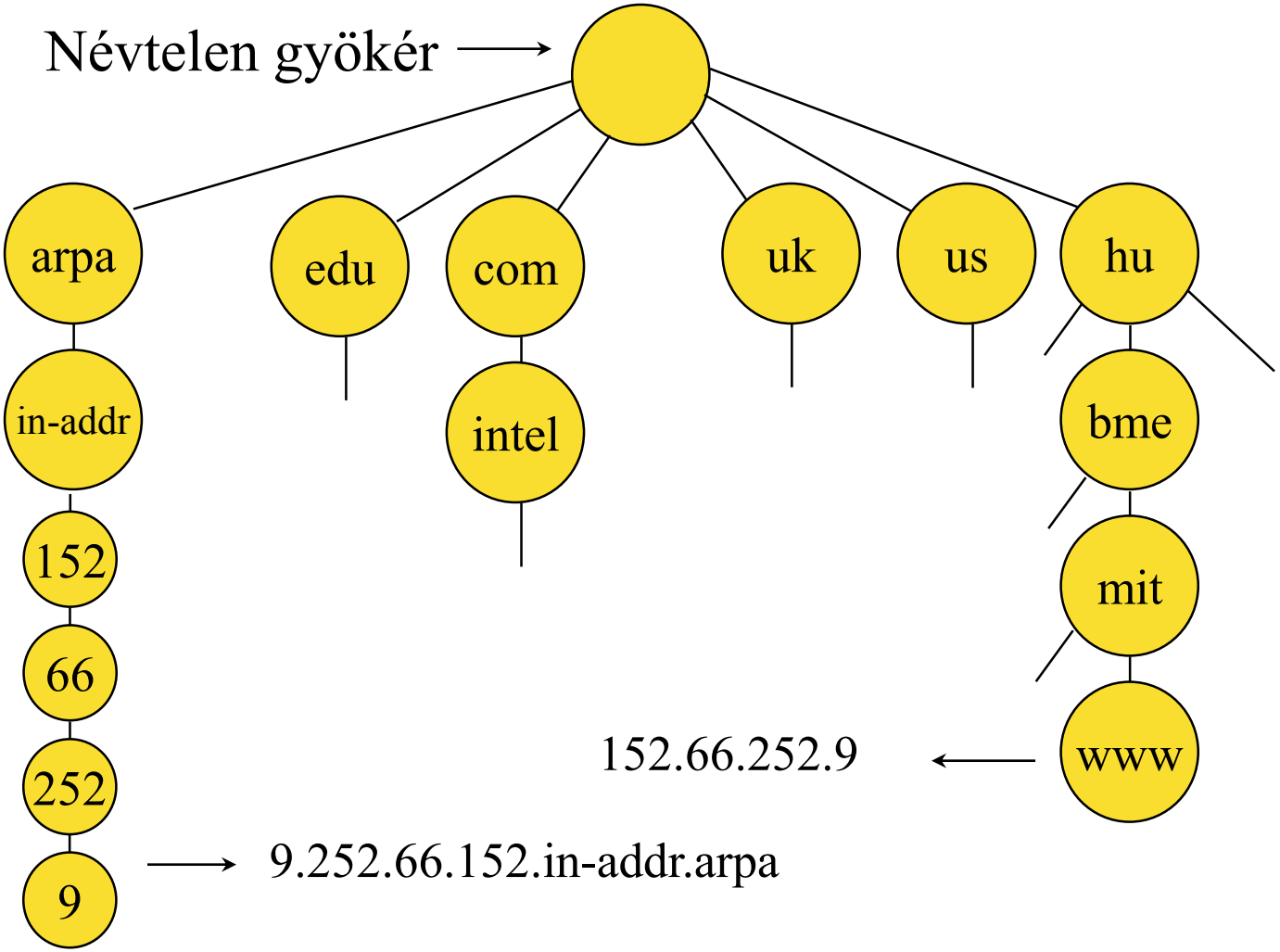
Alkalmazási réteg

- Segédprotokollok
  - DNS
  - DHCP
  - TFTP
- Elsődleges alkalmazási réteg protokollok
  - HTTP
  - FTP
  - POP/IMAP, SMTP
  - SNTP, NTP, PTP
  - SNMP

Domain Name System, DNS

- Feladata az IP címek (IPv6 is) leképzése emberi formában megjegyezhető, hierarchikus szöveges nevekre (és vissza)
- Egyéb név/IP cím specifikus információk tárolása
  - Pl. Email szerver
- A domén név részei:
  - Csúcs (top-level) domén nevek (com, hu, stb.)
  - Subdomén nevek (több szinten)
  - Állomás neve (host name)
- Minden egyes részhez (kivéve az állomást) tartoznak DNS szerverek (min. kettő a redundancia miatt szintenként)
  - A DNS szerverek információt cserélnek (zone transfer)
- A kliensben van egy u.n. resolver (CACHE-el)
  - Rekurzív vagy nem rekurzív lekérdezések
    - A szerver képességeitől függ melyik
    - A rekurzív a tipikus (a szerver is CACHE-el)
- Domain nevek fája (leképezve az IP címeket is)

A DNS fa szerkezet IPv4 esetén



A DNS protokoll

- A kliensek UDP szállítási réteget használnak a szerver elérésére
- A szerverek között TCP-ét
- Mindkét esetben az 53-as porton található a szerver
- A kliensek konfigurációjában meg kell adni a DNS szerverek IP címét
  - Redundancia miatt legalább 2 szervert kell megadni
  - Minden subdomén-ben kell két szerver, egyébként nem jegyzik be a subdomént
  - Teljesen megadott domén név (Fully qualified domain name, FQDN)
    - Pl. www.mit.bme.hu. (a végén a pont jelzi, hogy FQDN-ről van szó)
  - Egyébként domén név kiegészítés (Domain Suffix)

A DNS cache

- Fontos mező a protokollban a TTL
  - Time to Live, de ez itt más jelent, mint az IPv4 protokollnál
  - Meddig tartható lokális CACHE-ben a bejegyzés
  - Ha a CACHE-ben régi bejegyzés van valamilyen szerverben, akkor a hierarchiában alatta levő szerverek a gépet nem tudják elérni (más IP címet használnak)
  - Több napba is tellhet a cache-ek szinkronizálása...

Dynamic Host Configuration Protocol

- DHCP, van DHCPv6 is
- Bootstrap Protocol (BOOTP) utódja
- Elsődleges cél: IPv4 címek allokációja (IPv6-re kiterjesztve)
  - dinamikus, automatikus vagy statikus allokáció
- UDP protokollt használ (u.a. mint BOOTP)
  - Szerver: 67
  - Kliens: 68
- Más információt is tud szolgáltatni:
  - Alhálózati maszk, alapértelmezett átjáró
  - DNS és WINS szerver specifikus információk
  - Hálózati boot image helye (szerver és file megadása)
  - Egyéb, akár gyártó specifikus opciók

DHCP működése

- A kérés küldésekor az állomás nem tudja az IP címét
  - Mit állít be forráscímnek: 0.0.0.0
  - Mit állít be célcímnek: 255.255.255.255
  - Bizonyos implementációk először a legutolsó IP címmel próbálkoznak (elárulják honnan jöttek ☺)
- A válasz a szervertől már az új IP címre megy!
  - Mivel a MAC ismert, a gép megkapja
- Discovery, Offer, Request, Acknowledgment, Release, Information
  - Lease time (az időintervallum, amelyre a gép az IP címet megkapja)
- DHCP proxy vagy relay
  - Ha az alhálózatban nincs DHCP szerver, a routernek kell tudnia

Secure Socket Layer (SSL)

- SSL-t eredetileg a Netscape dolgozta ki
- Tényleges Internet szabvány: Transport Layer Security (TLS) Protocol Version 1.2, RFC 5246
- TCP felett, kapcsolat orientált
  - De van UDP feletti módosítás is Datagram Transport Layer Security (DTLS) néven (ritkán használják tudomásom szerint)
- Opció egyeztetés: A felek által ismert biztonsági protokollok (cipher and hash) listájából a legerősebbet választják
  - Nyilvános kulcsú megoldásokat használ ebben a szakaszban
- Tanúsítványokat cserélnek
  - Azokat ellenőrizhetik a felek a CA-nál (Certificate Authority)
- Utána titkos kulcsot cserélnek a nyilvános kulccsal titkosított csatornán
  - Pl. DES, 3DES, AES
- Azzal kommunikálnak, azt időnként cserélik, stb.
- Összetett, bonyolult, erőforrás igényes protokoll
  - Rengeteg buktató az implementációban (pl. véletlen szám genrálás)

Elsődleges alkalmazási réteg protokollok

- File átvitel
  - TFTP
  - FTP, SCP, SFTP
  - HTTP, HTTPS
- Levelzés:
  - SMTP
  - IMAP
  - POP
- Óraszinkronizáció
  - Daytime
  - Time
  - NTP
  - IEEE 1588 (PTP)
- Menedzsment és adatgyűjtés
  - SNMP

Trivial File Transfer Protocol, TFTP

- Egyszerű file átvitel
  - 69-es UDP portot használja
  - Nincs dir/ls jellegű parancs (tudni kell a file nevét)
  - Nincs felhasználó azonosítás
  - Nem gyors (hálózati késleltetés, hibák erősen befolyásolják)
    - Egyszerű kérés-válasz-nyugta protokoll a file darabjaira
- Nagyon kevés erőforrásra van szüksége
- Lokális hálózatokon belül image/config file csere
  - Remote BOOT (BOOTP/DHCP adja meg a file-t)
  - Remote install (Windows XP is tudja)
  - Remote firmware upgrade
  - Config file mentése és visszatöltése
  - Adatgyűjtés (nem a legjobb megoldás)

TFTP megvalósítás

- A beágyazott rendszerben kliensre van szükségünk
  - Szinte minden TCP/IP portolás során egy új platformra az első lépések egyike egy TFTP kliens elindítása
- Még a Windows XP-ben is van kliens
  - Windows 7-ben engedélyezni kell (Control Panel > Programs and Features > click Turn Windows features > enable Client Telnet and Client TFTP)
- Windows alá letölthető szerver (Win7 is)
  - Tftpd32: http://tftpd32.jounin.net/
  - Mást is tud (DHCP, TFTP, DNS, SNTP és Syslog)
- Linux-ban van kliens és szerver is
  - Installálni kell…

File Transfer Protocol, FTP

- File-ok teljes értékű átvitele
- Összetett protokoll
  - Eredeti formájában (Aktív FTP) tűzfal és NAT működésképtelenné tette
    - FTP proxy-ra volt szükség
  - A passzív FTP ezt a problémát részben megoldja
- Aktív FTP
  - Kliens megnyitja a vezérlő (control) csatornát a szerver 21-es TCP portján
  - A vezérlő csatornán megállapodnak az adatcsatorna beállításaiban
  - A szerver nyitja meg a kliens felé (NAT vagy tűzfal rémálom)
- Passzív FTP
  - Az adatcsatornát a kliens nyitja meg a szerver felé
  - Ekkor is gond, hogy az adatátvitel alatt a control csatorna nem használható (nincs benne forgalom), vagyis a NAT vagy tűzfal azt letilthatja (timeout)
    - Miért volt szükség akkor 2 csatornára, azt nem tudom?
- A jelszavakat nem védi, azok eredeti formájukban kerülnek átvitelre a hálózaton
  - FTP over SSL problémás a két TCP folyam miatt
  - SSH File Transfer Protocol (SFTP) veszi át a helyét remélhetőleg (WINSCP)
- Véleményem szerint egy alapvetően „elszúrt” protokoll, szerencsére eltűnőben van

Telnet

- Soros terminál távoli gépről egy TCP kapcsolatban
- Az egyik legelső alkalmazási réteg protokoll (RFC 15 egy korai verziót ír le)
- 23-as TCP portot használja PSH-val
  - A leütött karaktert küldi a kliens (terminál)
  - A választ (echo) küldi vissza a szerver (gép)
  - De a jelszó esetén pl. nem küld választ, vagy „*”-kat küld
- Command Line Interface (CLI)
  - Soros porton lokálisan, Telnettel hálózaton keresztül
  - Ugyan az a parancsértelmező (shell) fut mögöttük
- A jelszavat karakterenként, nyíltan küldi át
  - SSH a megoldás (Secure Shell), 22-es TCP portot használja
  - Az SSH SSL-t használ a titkosításra
  - Az SSH sokkal erőforrásigényesebb

WWW fejlődése

- Elődje a Gopher
- World Wide Web
  - Tim Berners-Lee CERN-ben dolgozó fizikus találta ki
  - 1989-1991 fejlesztés
  - 1991 augusztus 6-án publikáció + első elérhető szerver (saját gépe)
- Részei:
  - HyperText dokumentum
    - Hivatkozásokat tartalmaz más tartalmakra
  - Kliens-szerver architektúra a dokumentumok cseréjére
- Kiegészítések:
  - Szerver oldali programozás (1993-tól, NCSA Web Server)
    - Common Gateway Interface, CGI (szerver oldali program indítás)
    - Server plug-ins (a szerverbe beépülő programok, pl. Perl/PHP/Python értelmező), hatékonyabb mint a CGI
  - Kliens oldali programozás (1995 decemberétől, Netscape)
    - Javascript
  - Extensible Markup Language (XML) 1998-tól
    - Egyértelmű adat reprezentáció (szemantikus WEB)

WWW technológiák

- Technológiák az előző oldalról
  - Bármelyikből több tárgy lenne összeállítható
  - Pl. XML…
- Ebből most fontos:
  - Kliens-szerver architektúra hálózati szempontból
  - Dokumentumok kezelése hálózati szempontból

HTTP kliens szerver modell

- Szerver
  - Valamilyen filerendszerben tárolja a dokumentumokat és a konfigurációját
    - Tényleges file, adatbázis, C konstans kódmemóriában, stb.
  - Beérkező kérés esetén:
    - A dokumentumot elküldi/fogadja vagy hibajelzést ad
      - A dokumentum létrehozható futási időben egy program eredményeképpen (részben vagy egészben)
      - Jelzi a kliensnek a dokumentum típusát
    - Log-olja a kéréseket és azok eredményét
- Kliens
  - Letölti a dokumentumot (a letöltés paraméterezhető)
    - Értelmezi azt
    - Letölti a megjelenítéséhez szükséges beágyazott dokumentumokat
    - Megjeleníti
  - Feltölt/töröl dokumentumot
  - CACHE-el

Az URI/URL

- A dokumentum általánosítható erőforrásra
- Uniform Resource Identifier (URI) a hálózaton található erőforrások azonosítására szolgál
- Uniform Resource Locator (URL) megadja, hogy az erőforrás hol érhető el
- Abszolút vagy relatív lehet (a hivatkozó erőforráshoz képest)
- Szintakszis:

```
resource_type://username:password@domain:
   port/filepathname?query_string#anchor
```

- resource_type: megadja az erőforrás kezelésére alkalmas protokollt, mint pl. http:, mailto:, ftp:, stb.
- username:password: Nyílt szöveg, HTTP authentikáció esetén
- domain:port: Megadja az erőforrást tartalmazó szerver domén nevét és a portot (default portot nem kell megadni)
- filepathname: A szerveren a file helye
- query_string: field1=value1&field2=value2&field3=value3 formában az erőforrás elérésének paraméterei
  - Speciális karaktereket encode-olni kell (URL encoding)
- Horgony (anchor): A dokumentumon belüli hivatkozás egy előre megjelölt részhez

HTTP protokoll

- Karakteres protokoll, olvasható (telnet www.mit.bme.hu 80)
- A WWW esetén használt alapértelmezett port:
  - 80-as TCP, 8080-as TCP (kísérleti szerver)
  - 443-as TCP port https protokoll esetén (HTTP over SSL)
  - Állapotmentes
    - Cookie-val (süti) megkerülhető, kliens tárolja és elküldi a következő kérésekkel is
  - Opció egyeztetés a kliens és a szerver között
- Metódusok
  - GET, POST, HEAD, PUT, OPTIONS, stb.
- HTTP 1.0:
  - Minden egyes kéréshez új TCP kapcsolat épül fel
- HTTP 1.1:
  - A TCP kapcsolatokban több kérés is küldhető egymás után
  - Keep-alive, kisebb késleltetés, kevesebb erőforrás
- Felhasználó megadása
  - Basic Authentication
    - Base64 kódolt felhasználó és jelszó küldése, dekódolható, újra lejátszható
  - Digest Authentication
    - MD5 kriptográfiai hash függvény alapú, összetettebb, jelszófüggő erősségű

HTTP metódusok: GET

- Erőforrás letöltésére
- `GET /filepathname?query_string`
- Felfogható egy függvény hívásnak
  - `filepathname`: függvény neve
  - `query_string`: paraméterek
- CACHE-elhető
- A visszatérési értéket kapja meg a kliens
  - A visszatérési érték
    - Státusz kód: 1xx (info), 2xx (ok), 3xx (redirect), 4xx (kliens hiba), 5xx (szerver hiba)
    - A visszatérési érték típusa, hossza, legutolsó módosítás időpontja, stb. kerül megadásra
    - Visszatérési érték (dokumentum, html file, kép, stb.)
  - Szerver oldali programozás esetén lehet mellékhatása
    - A protokoll definíció szerint nem lehetne, arra a POST metódus szolgál
  - Mellékhatás:
    - Változás a rendszer (szerver) állapotában a hozzáférés log-oláson kívül
    - Pl. file vagy adatbázis írás, stb.

HTTP metódusok: POST

- Erőforrás elérésére, amely valamilyen akció sorozat végrehajtását is igényli a szerveren (mellékhatással)
  - URL megadása (az üzenet testet kezelő program azonosítása)
  - Az akció sorozat paraméterei a POST üzenet testében kerülnek megadásra
  - A paraméterek azonos formátumúak az URL query_string részével
  - HTML űrlap kitöltésénél adható meg például a POST használata
  - Nem CACHE-elhető
- GET és POST összehasonlítása
  - GET CACHE-elhető, a POST nem
  - GET méretét korlátozza az URL maximális mérete (kliens és szerver függő)
  - POST esetén nincs ilyen korlát, az üzenet test tetszőleges méretű lehet
  - GET beágyazható HTML lapba
  - POST csak HTML űrlapból vagy programozottan küldhető (wget, curl)
  - GET látható a kliensek URL mezőjében (biztonság érzése)
  - POST nem látható

További HTTP metódusok

- HEAD
  - CACHE-elés támogatására
  - Ugyan az mint a GET
    - Nem adja vissza a dokumentumot
    - Csak a header-t
  - Kliens dönthet
    - A CACHE-ben lévőt használja
    - Újra lekéri az információt GET-tel
- PUT
  - Dokumentum feltöltése
  - Engedélyezni kell a feltöltést szerver szinten
  - Jogosultság kell hozzá (felhasználó azonosítás)
- Options
  - GET-hez hasonló, de csak az adott erőforráshoz tartozó szerver opciókat kapja meg a kliens

Általános célú HTTP szerver működése

- Várja a beérkező kéréseket a 80-as (konfigurálható) porton
- A beérkező kéréseket átadja a dolgozó szálaknak (worker thread)
  - Azok kiszolgálják a beérkező kéréseket
  - A szerver annyi dolgozó szálat indít, amennyi a beérkező kérések kiszolgálásához szükséges, vagy amíg a rendszer erőforrásai el nem fogynak
  - A szerver oldal CGI vagy más módon tetszőleges programot tud futtatni (PHP, Perl, Python, stb.)
- Kisteljesítményű beágyazott rendszerben ez nem járható út…
  - GET metódus többnyire elég
  - Véges számú dolgozó szál
    - 2 minimum szükséges, mivel a legtöbb kliens 2 szálon tölt le
    - Első szálon letölti a HTML file-t, azonnal elkezdi feldolgozni
    - Talál benne erőforrást (pl. kép), azt megpróbálja a 2. szálon letölteni
    - Ha nincs második szerver szál, akkor az erőforrást elérhetetlennek fogja látni
  - Hogyan képezzük le a beágyazott rendszer funkcióit HTTP kérésekre?

Adatgyűjtés HTTP-vel

- A szenzor kliens vagy szerver legyen? Mindenki szervert használ, de…
- Szerver:
  - Humán interfész szempontjából jobb ha szerver
    - A szenzort egy böngésző segítségével lehet konfigurálni
    - Az adatokat a böngészőben lehet nézegetni
    - Élő adat pl. DHTML, AJAX, JAVA Applet, stb. segítségével
  - Ekkor egy adatgyűjtő pooling jelleggel kérdezheti le
    - Az adatgyűjtő HTTP kliens ekkor
  - A HTTP szerver biztonsági kockázat
    - HTTPS és kliens azonosítás (erőforrás igényes)
  - A HTTP szerver erőforrás igényes
    - Denial of Service (DoS) attack lehetséges
- Kliens
  - Humán interfész szempontjából nem jó
    - Konfiguráció letöltése jelentheti a megoldást
    - A szerveren a konfiguráció összeállítható
  - A szenzor feltölti az adatokat a szerverre
    - A szerver feltöltő URL-t meg kell adni a konfigurációban
  - Alacsonyabb erőforrás igény, kisebb biztonsági kockázat, stb.

Konfiguráció kezelés HTTP-vel

- Konfiguráció megjelenítése
  - A konfiguráció alapján dinamikusan generált WEB lapok
  - Többnyire egyből felajánlható a beállítás is
- Beállítás
  - A beállítandó paraméterek űrlapon
- Konfiguráció le- és feltöltése
  - Le- és feltöltés más formátumban
    - CLI batch command file (CISCO és egyéb hálózati eszköz gyártók)
    - Kulcs-érték párok, XML, JSON, vagy más strukturált leírás
- Felhasználó azonosítás és hálózati biztonság nagyon fontos
  - A konfiguráció nagyon érzékeny információ…

Firmware kezelés HTTP-vel

- Feltöltés: PUT
- Letöltés (mentés): GET
- Felhasználó azonosítás és hálózati biztonság nagyon fontos
  - A firmware a legkritikusabb rendszerkomponens hálózati biztonsági szempontból
- Hibakezelés nagyon fontos
  - Firmware formátumot jól kell kialakítani (ne egy HEX file legyen)
  - Firmware és HW kompatibilitása (ellenőrzés)
  - Firmware hibamentessége (ellenőrző összeg, CRC, digitális aláírás, stb.)

Simple Mail Transfer Protocol, SMTP

- Elektronikus levelek továbbítására
  - 25-ös TCP portot használja a szerver
  - Csak levél küldésre használják, fogadásra más protokollok szolgálnak
  - Erős authentikációval és titkosítással védhető
  - Szöveges protokoll (érthető üzenetváltások)
    - Egyszerű üzenetformátum eredetileg
    - Napjainkra a biztonsági követelmények miatt egyre összetettebb lesz (felhasználó azonosítás, hozzáférési korlátozások)
  - A kliensnek tudnia kell
    - A helyi MTA címét (domén név vagy IP cím), DNS-ből megtudható
    - A felhasználó levélcímét
    - A felhasználói nevet és jelszót authentikáció esetén
    - SMTP-vel továbbítja a levelet ezek alapján a helyi MTA-nak
  - Az MTA átveszi az üzenetet és továbbítja
    - DNS Mail Exchanger megadja a címzett MTA-jának a domén nevét
    - A címzett MTA tárolja, amíg azokat a címzett meg nem nézi
    - Hibaüzenetek levél formában
- A helyesen konfigurált szerverek csak a belső IP cím tartományokból fogadnak el kimenő leveleket (SPAM küldés elleni védelem)
- Üzenet alapesetben ASCII text
  - Multipurpose Internet Mail Extensions (MIME) bináris tartalmat is lehetővé tesz

POP/IMAP protokoll

- Levél fogadást lehetővé tevő protokollok
- Authentikáció szükséges
- Erős kriptográfiai algoritmusokkal védhető a kommunikáció
- Post Office Protocol (POP)
  - 110-es TCP port (SSL opció egyeztetés), de a 995-ös TCP portot (POP3S) is használják néha
  - A levelek letöltését teszi lehetővé
  - Opcionálisan a levelek megőrizhetőek a szerveren is
  - Elavultnak tekinthető, de kis erőforrás igénye miatt az Internet szolgáltatók ezt szokták támogatni elsősorban
- Internet Message Access Protocol (IMAP)
  - 143-as TCP port, vagy 993-as TCP port (IMAP over SSL)
  - Off-line és on-line működés
  - A levelek tárolása a szerveren és/vagy a kliensen (szinkronizáció)
  - A levelek folder-ekbe rendezhetők
  - Az üzenetek részei külön tölthetőek le (erőforrás gazdálkodás)
    - Csak azt tölti le, amire szükség van (pl. levéllista elkészítéséhez nem kell az egész levelet letölteni
  - Összetett, erőforrás igényes protokoll
  - Jellegzetesen vállalti informatikai rendszerekben használják
- Ritkán szükségesek beágyazott rendszerekben

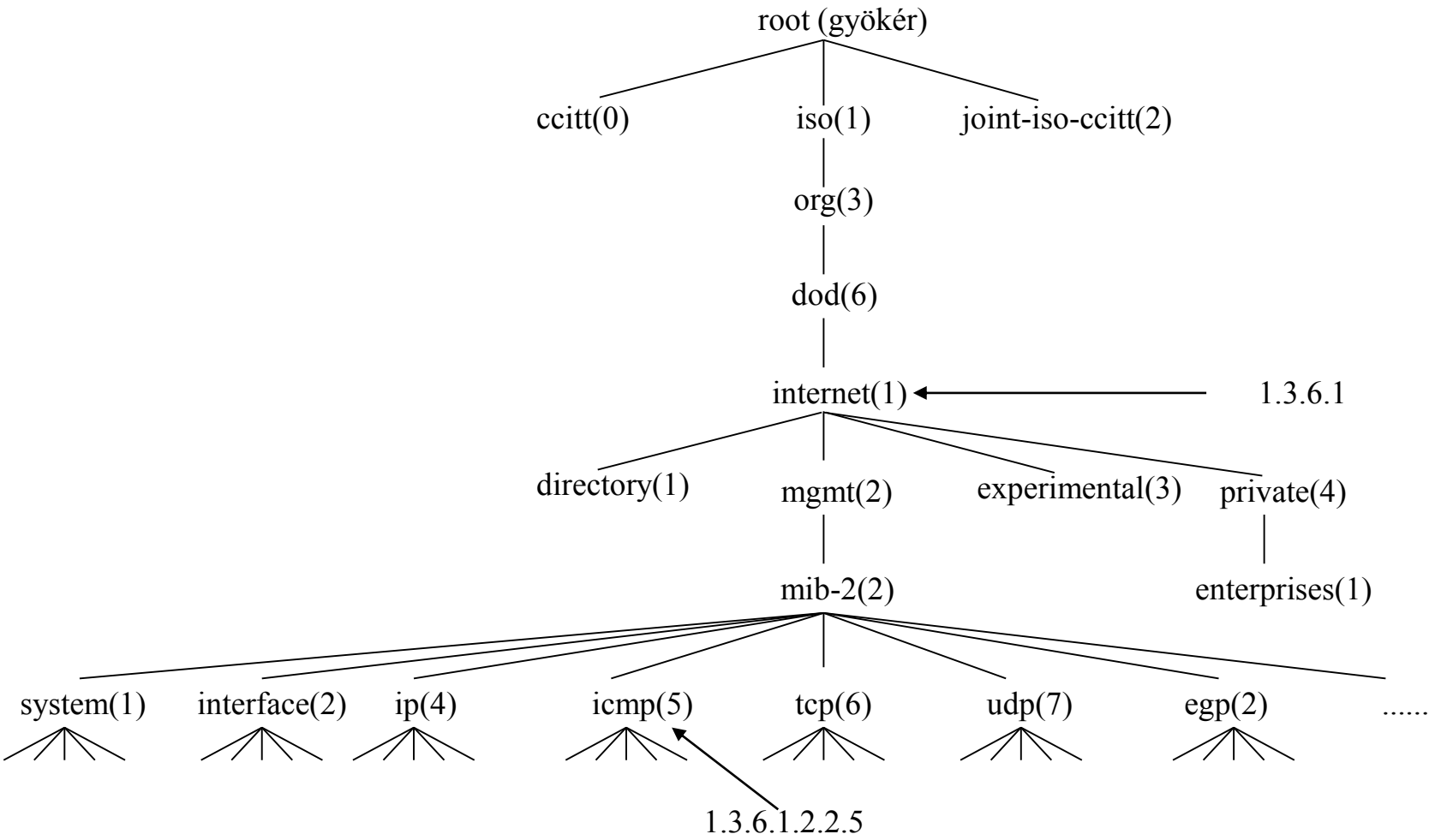
Simple Network Management Protocol (SNMP)

- Kliens-szerver architektúra hálózat menedzsmentre
  - Konfiguráció lekérdezése és beállítása
  - Működéssel kapcsolatos információk lekérdezése (adatgyűjtés)
  - Események kezelése
- SNMP ágens a menedzselt eszközökön
- SNMP menedzser komponensek a menedzselő munkaállomásokon
- SNMP protokoll (információ csere az ágens és a mendzser között)
- Az elérhető információ leírása (Management Information Base, MIB)
  - Leírás a menedzser komponenseken
    - SMI, Structure of Management Information (SMI), ASN.1 nyelven leírva
  - Implementáció a menedzselt eszközökön
- SNMPv1, SNMPv2c/SNMPv2u, SNMPv3
  - Az SNMPv1 és az SNMPv3 terjedt el
  - SNMPv1 nem biztonságos
- Nem csak hálózat, tetszőleges eszköz menedzselésére használható

Management Information Base, MIB

- Fa struktúrájú adatbázis
  - Fa levelein tároljuk az típusos adatokat
    - Egyszerű típusos adatok
    - Táblázatok egyszerű típusos adatokból
  - Elemek azonosítására az objektum azonosító (object identifier):
    - iso.identified-organization.dod.internet.mgmt.mib-2
    - 1.3.6.1.2.1
- Leírás ASN.1 nyelven
  - A MIB-ből kód (C/C++/Java) generálható az ágens-en történő implementációhoz
  - A MIB alapján a menedzser komponens lekérdezi az ágenst, és értelmezi az abból kapott adatokat
- A MIB felépítése, az alábbi részekből áll össze
  - IETF által definiált MIB-ek
  - IEEE által definiált MIB-ek
  - Gyártó specifikus MIB-ek
- Speciális típus: Counter32/Counter64
  - Túlcsordulás jelzés nincs: Esemény számlálók kezelése elosztott rendszerben

SNMP MIB



MIB példa : System MIB specifikáció

- rfc1213-mib2.asn1
- Abstract Syntax Notation One (ASN.1) nyelven
- Lásd csatolt file
- Példa: sysUpTime
  - típusa TimeTicks (counter-ből származtatva)
  - OID: 1.3.6.1.2.1.1.3

```
sysUpTime                    OBJECT-TYPE
SYNTAX                       TimeTicks
ACCESS                       read-only
STATUS                       mandatory
DESCRIPTION                  "The time (in hundredths of a
second) since the network management portion of the
system was last re-initialized."
::= { system 3 }
```

MIB példa : System MIB implementáció

SNMPWALK

```
khazy@spock:~$ snmpwalk -c community_str -v1 spitfire system
SNMPv2-MIB::sysDescr.0 = STRING: 3Com SuperStack 3 Switch 3848
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.43.1.8.45
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (28106246) 3 days,
6:04:22.46
SNMPv2-MIB::sysContact.0 = STRING: Kovacshazy Tamas (khazy@mit.bme.hu)
SNMPv2-MIB::sysName.0 = STRING: spitfire
...
```

SNMPGET

```
khazy@spock:~$ snmpget -v1 -c  community_str spitfire 1.3.6.1.2.1.1.3.0
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (28188821) 3 days,
6:18:08.21
```

SNMP PDU-k

- SNMPv1 esetén COMMUNITY STRING alapú felhasználó azonosítás
  - Nyíltan küldi a menedzser, READ vagy WRITE jog van hozzá csatolva az ágensen
- GetRequest PDU (kérés)
  - OID vagy OID lista lekérés, a válasz GetResponse
- GetResponse PDU (válasz)
  - OID lista értékkel vagy hibaüzenet
- GetNextRequest PDU (kérés)
  - A paraméterként megadott OID utáni (fában) OID lekérdezése (fa bejárása), a válasz GetResponse
- SetRequest PDU (kérés)
  - OID beállítása, a válasz GetResponse
  - Az új érték
- Trap PDU: Az ágens küldő esemény jelzésként
  - Előzetesen konfigurálni kell (alkalmas SetRequest sorozat)
- SNMPv2- : GetBulk PDU
  - Táblázat gyors letöltésére
- SNMP ágens 161-es UDP porton vár a kérésekre
- A menedzser a 162 UDP porton vár a TRAP PDU-ra

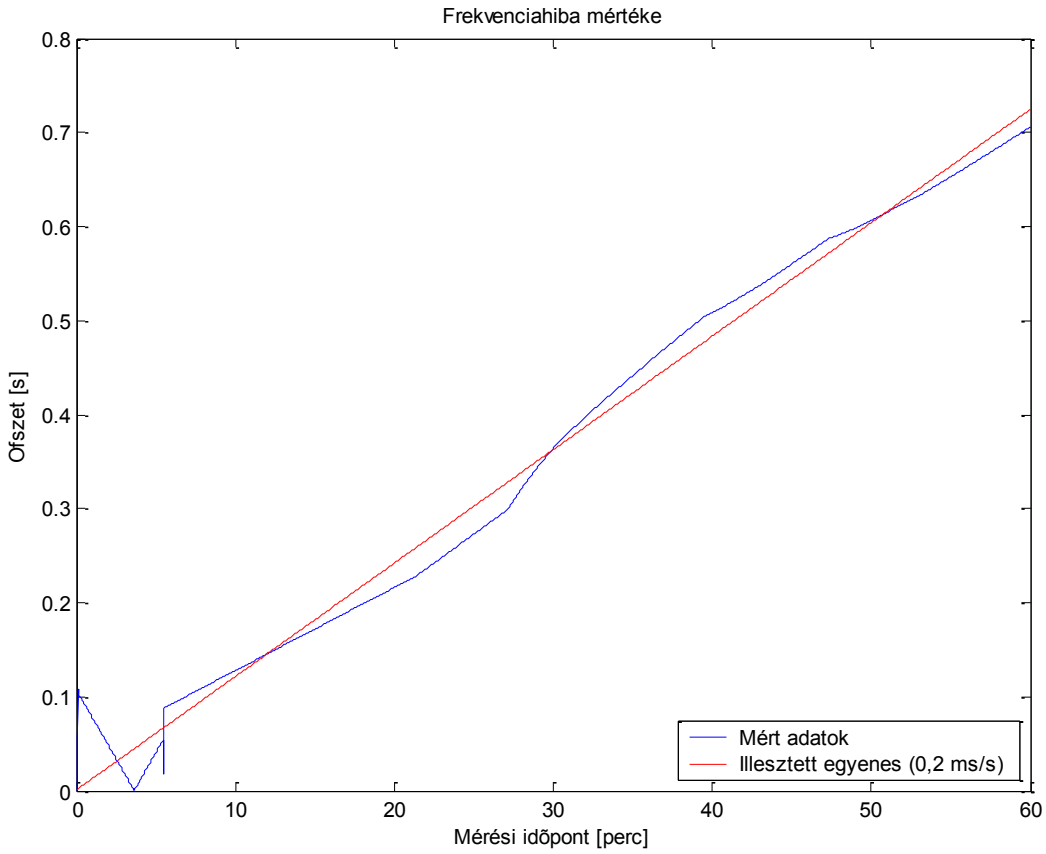
SNMP Counter típus 1.

- 32 vagy 64 bites esemény számláló
  - Túlcsordulást nem jelez
  - A felhasználónak kell észrevennie a túlcsordulást
    - A számláló bitszáma ismert (bits)
    - Eseménygyakoriság felső korlátja ismert (F_{event})
    - Ez alapján a maximális lekérdezési periódusidő meghatározható (T_{max})
    - Ha ennél gyakrabban kérdezzük le, akkor a két lekérdezés között történt események száma számítható (ha a rendszer nem indult újra)
- Példa:
  - 100 Mbps sebességű Ethernet interface 32 bites in vagy out byte számláló
  - Maximális esemény gyakoriság: $12.5*10^6$ byte/s
  - Maximális (hibamentes) lekérdezési periódusidő:

$$T_{max} = \frac{2^{bits} - 1}{F_{event}} = \frac{2^{32} - 1}{12{,}5 * 10^6} = 343.59s$$

Óraszinkronizáció, de miért?

Szinkronizálatlan órák (két PC)



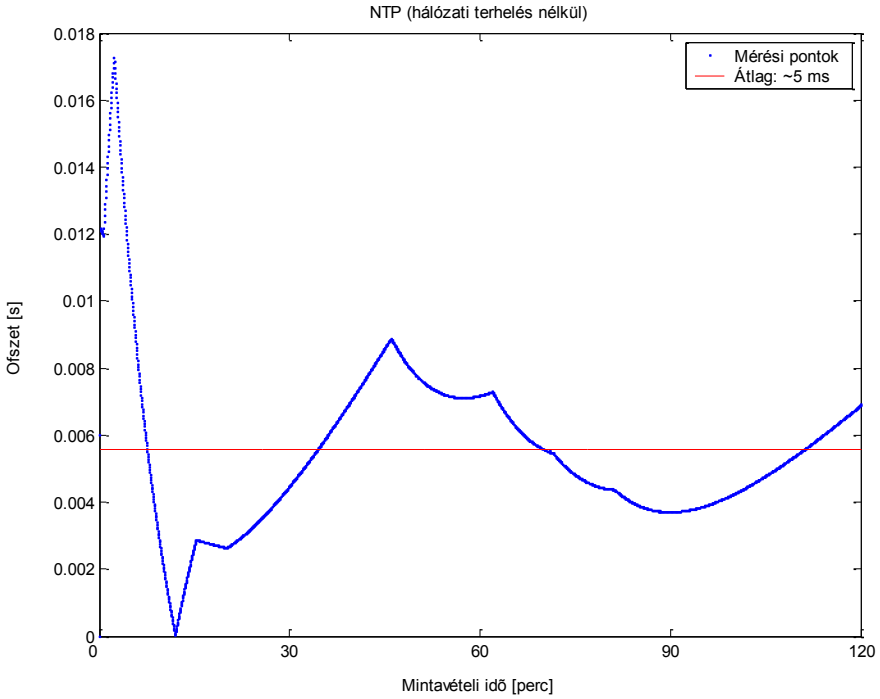
Forrás: Ferencz Bálint, Hardver támogatott IEEE 1588 alapú óraszinkronizáció, TDK dolgozat, 2010

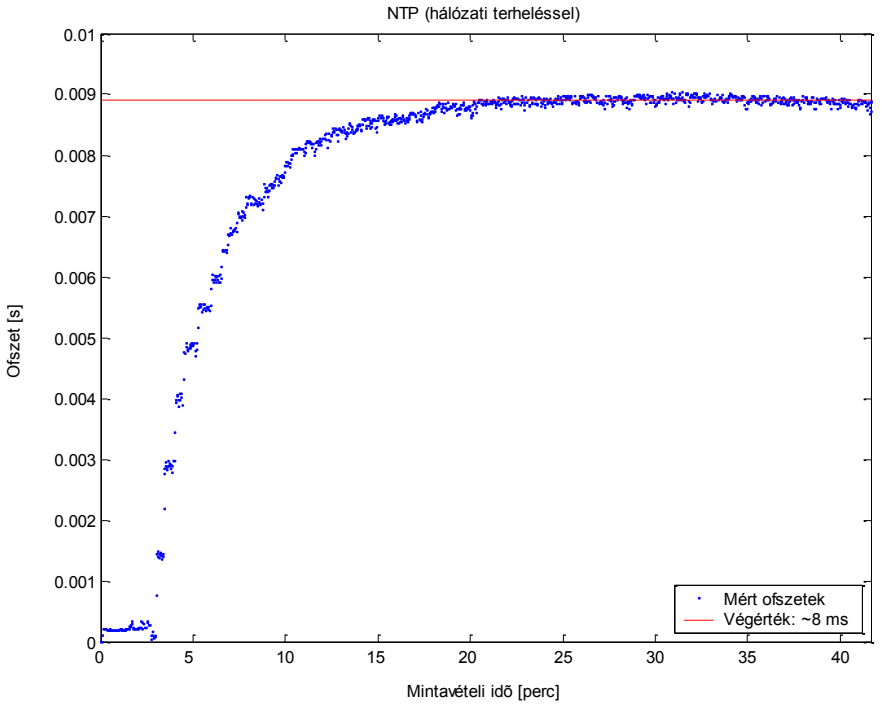
Óraszinkronizáció TCP/IP hálózatokban

- DAYTIME, TIME egyszerű órabeállítás
  - Késleltetést nem veszi figyelembe, nem redundáns
- Network Time Protocol, NTP
  - Célja: Óraszinkronizáció IP alapú hálózatokban
  - Csomagvesztés, ingadozó késleltetés
  - NTP szerverek megbízhatósága megkérdőjelezhető, több szerver redundáns használata (clock ensamble)
  - Unicast vagy multicast üzenetküldés (UDP, 123-as port)
  - UTC alapú (nem foglalkozik az időzónával, helyesen, az csak a humán interfészen érdekes)
- Hierarchikus rendszer
  - Stratum 0: GPS, atomóra, stb.
  - Stratum 1: Stratum 0-hoz szinkronizálódik, és így tovább
  - Alapesetben csak szerverek léteznek:
    - Simple Network Time Protocol (SNTP) klienseknek (egyszerűbb)
- Összetett algoritmus
  - Méri a szerverek közötti átlagos késleltetést, és kompenzál vele (offszet kompenzáció)
  - Méri a lokális óra paramétereit, és arra is kompenzál (frekvencia kompenzáció)
  - Kb. 100 ms pontosság az Interneten, <10 ms LAN-on belül elérhető
  - Több szerverhez szinkronizálódik, a hibás szerverek kiszűrhetők elégséges számú jó szerverrel fenntartva a kapcsolatot

NTP tulajdonságai 2.

NTP alapú óraszinkronizáció terhelés nélkül és terheléssel





Forrás: Ferencz Bálint, Hardver támogatott IEEE 1588 alapú óraszinkronizáció, TDK dolgozat, 2010

Simple NTP

- A komplett NTP implementáció szerver és kliens is (P2P lényegében)
- Összetett protokoll, felesleges a kliens gépeken
  - Akikhez nem fognak további gépek szinkronizálódni
  - Nem érhetők el folyamatosan
  - Kevés erőforrással rendelkeznek
  - Csak a saját órájukat szeretnék beállítani
- Simple NTP, SNTP
  - Csak kliens
  - Az NTP egy része kerül megvalósításra
  - Legmagasabb Stratum-on használható csak (levelek a NTP fában)
  - Cél a lokális óra beállítása a Daytime/Time protokolloknál pontosabban, és NTP kompatibilisen (NTP szervereket használva)
- A Windows beépített NTP kliense is ezt valósítja meg
  - Nem teljesen kompatibilis (természetesen)
- Beágyazott rendszerekben speciális esetektől eltekintve ez ajánlható
  - Alacsony erőforrás igény, elégséges funkciók

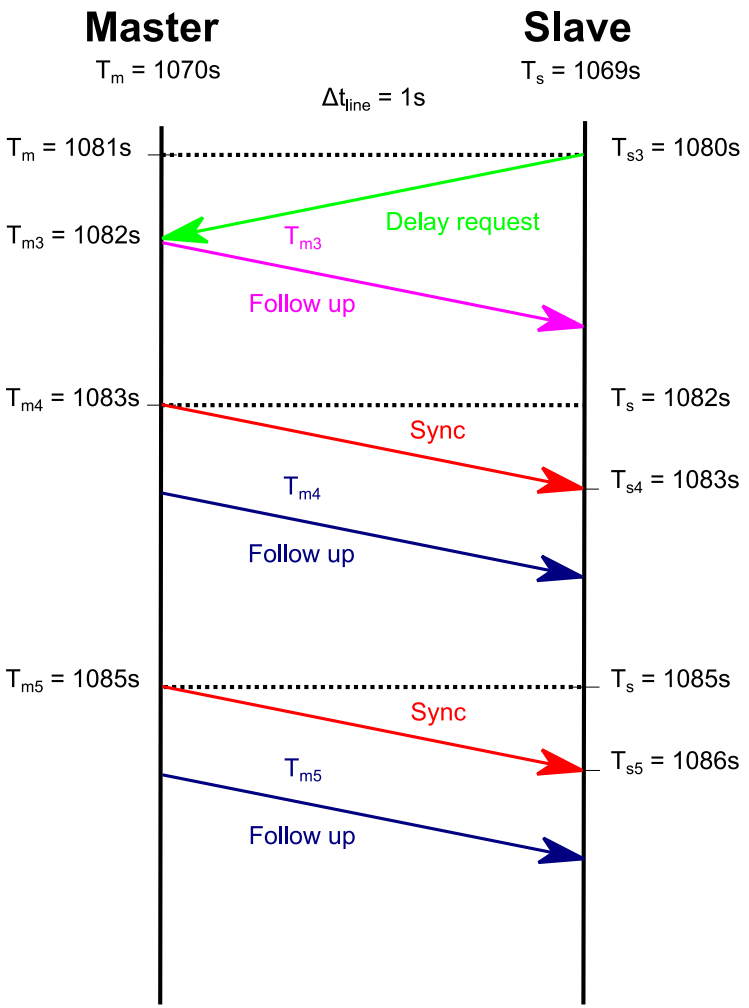
NTP és az SNTP értékelése

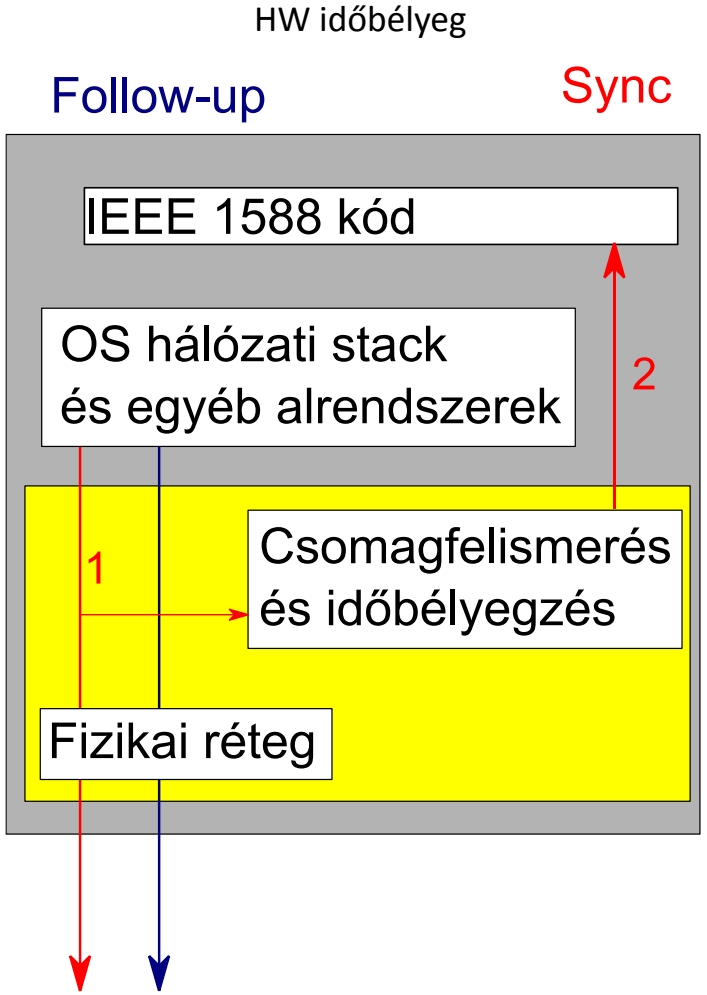
- Humán felhasználók számára elégséges pontosság
  - „A rendszerórák együtt járnak kb. 1 s-nél jobban…”
  - A rendszeróra rezgéskeltőjének (kvarc) a frekvenciahibáját meg tudjuk mérni, és azzal is tudunk kompenzálni
  - Folyamatosan, vagy legalábbis rendszeresen kell futnia (drift)
  - Titkosítás, authentikáció is támogatott (secure NTP)
- Internetre lett tervezve
  - Túl általános feltételezések alapján működik
  - Erősen függ a működése a hálózati terheléstől (késleltetés)
- Szoftver alapú időbélyegeket használ
  - SW időbélyeg a szerver és a kliens gépben is nagyon pontatlan, terhelésfüggő
- Nem alkalmas méréstechnikai és szabályozástechnikai igényességgel működni
  - Globálisan használható lokális időbélyegek lokális eseményekhez rendelése elosztott rendszerekben
  - Együttes működés (órajel, idő vezérelt rendszer)
  - Kis méretű, dedikált rendszer (pl. beágyazott rendszer)

IEEE 1588, Precision Time Protocol

- IEEE 1588-2002, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems
  - PTPd v1
- IEEE 1588-2008, PTPd v2 (módosítások a gyakorlat alapján)
- Elsősorban lokális hálózatban történő szinkronizációra
  - Master clock: GPS vagy atomóra
  - Slave clock Ethernet interfésszel
- Hardware támogatás lehetséges
  - A beérkező és a küldött IEEE 1588 üzeneteket tartalmazó keretek HW-ből láthatók el időbélyeggel
  - SW időbélyeg pontatlan (OS ütemezés)
- Hálózati támogatás
  - IEEE 1588 kompatibilis switch-ek
  - A switch által okozott késleltetés okozta hiba is csökkenthető

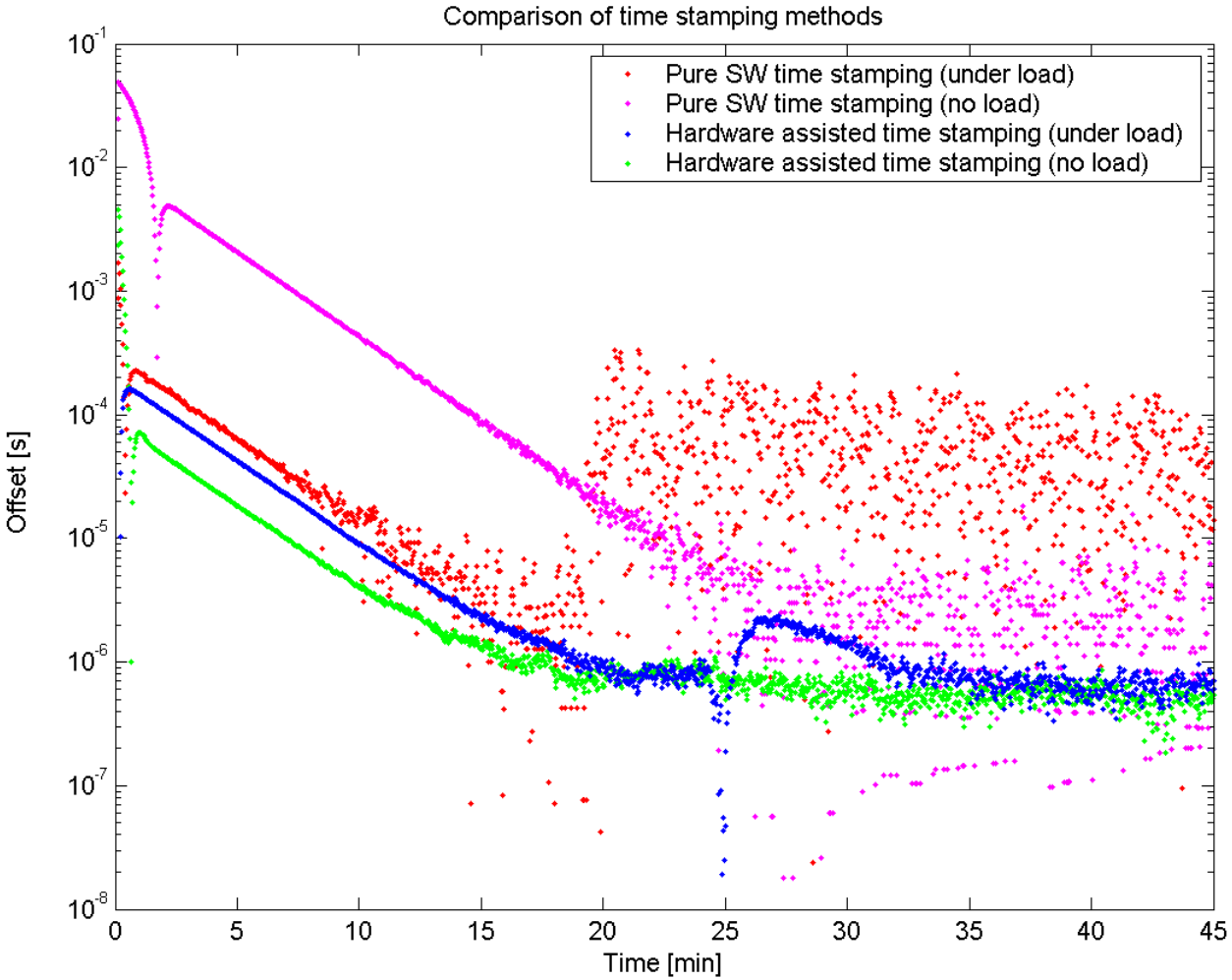
IEEE 1588 működése és HW időbélyeg





Forrás: Ferencz Bálint, Hardver támogatott IEEE 1588 alapú óraszinkronizáció, TDK dolgozat, 2010

PTPd tulajdonságai

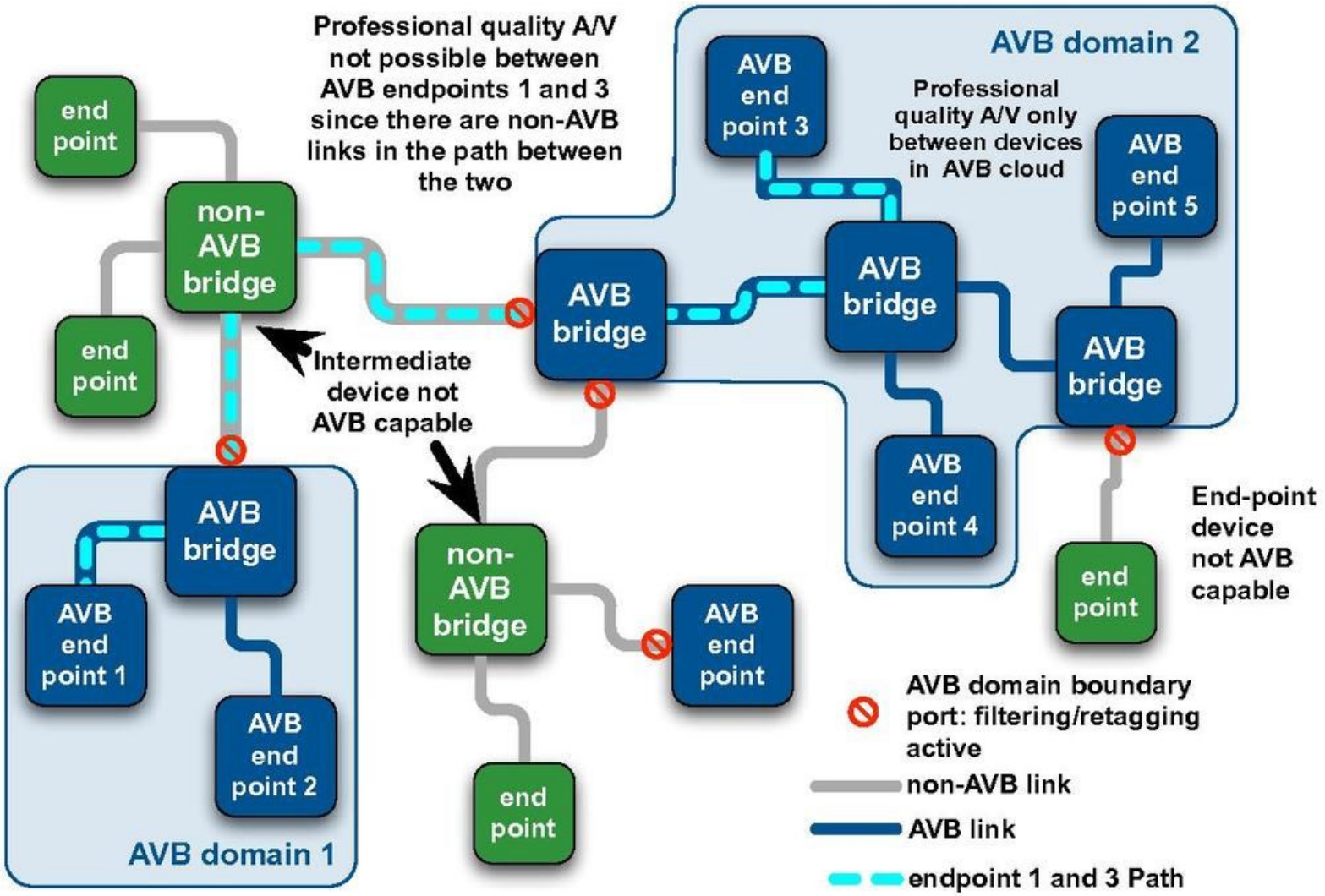


Forrás: Ferencz Bálint, Hardver támogatott IEEE 1588 alapú óraszinkronizáció, TDK dolgozat, 2010

Audio Video Bridging

- EEE 802.1BA:[4] Audio Video Bridging (AVB) Systems
  - IEEE 802.1AS: Timing and Synchronization for Time-Sensitive Applications (gPTP)
    - IEEE 1588 profile
    - Digitális „GENLOC” megvalósítás
  - IEEE 802.1Qat: Stream Reservation Protocol (SRP) és IEEE 802.1Qav: Forwarding and Queuing for Time-Sensitive Streams (FQTSS)
    - Erőforrás foglalás a switch-ekben
    - Lényegében csomagvesztés és késleltetés optimalizálás
- AVB kompatibilis eszközök
  - AVB végeszköz (forrás és/vagy nyelő)
  - AVB switch
- Nem TCP/IP, hanem tisztán Ethernet felett működik
  - De inkább alkalmazási réteg protokoll
  - Támogatja a WI-FI és az EPON hálózatokat is

AVB rendszer példa…



Time-Sensitive Networking

- „Időérzékeny” hálózatok
  - A beágyazott rendszerekben fontos a válaszidő
  - Valós idejű elosztott rendszerek terjednének, de…
  - Az Ethernet és TCP/IP felett a valós idejű működés nem garantált
    - Az AVB valami ilyet akar, de nagyon speciális alkalmazási körben...
- Ethernet + TCP/IP módosítása lehetséges:
  - Nem szabványos megoldás (cég specifikus)
  - IEEE 802 Time-Sensitive Networking szabványosítási folyamat
    - AVB alapján, általánosabb alkalmazási körre
    - Redundancia biztosítása is előírás
  - Eredmények:
    - 2-3 év szükséges a szabványosításra (még)
    - 4-5 év múlva várható az elterjedés…