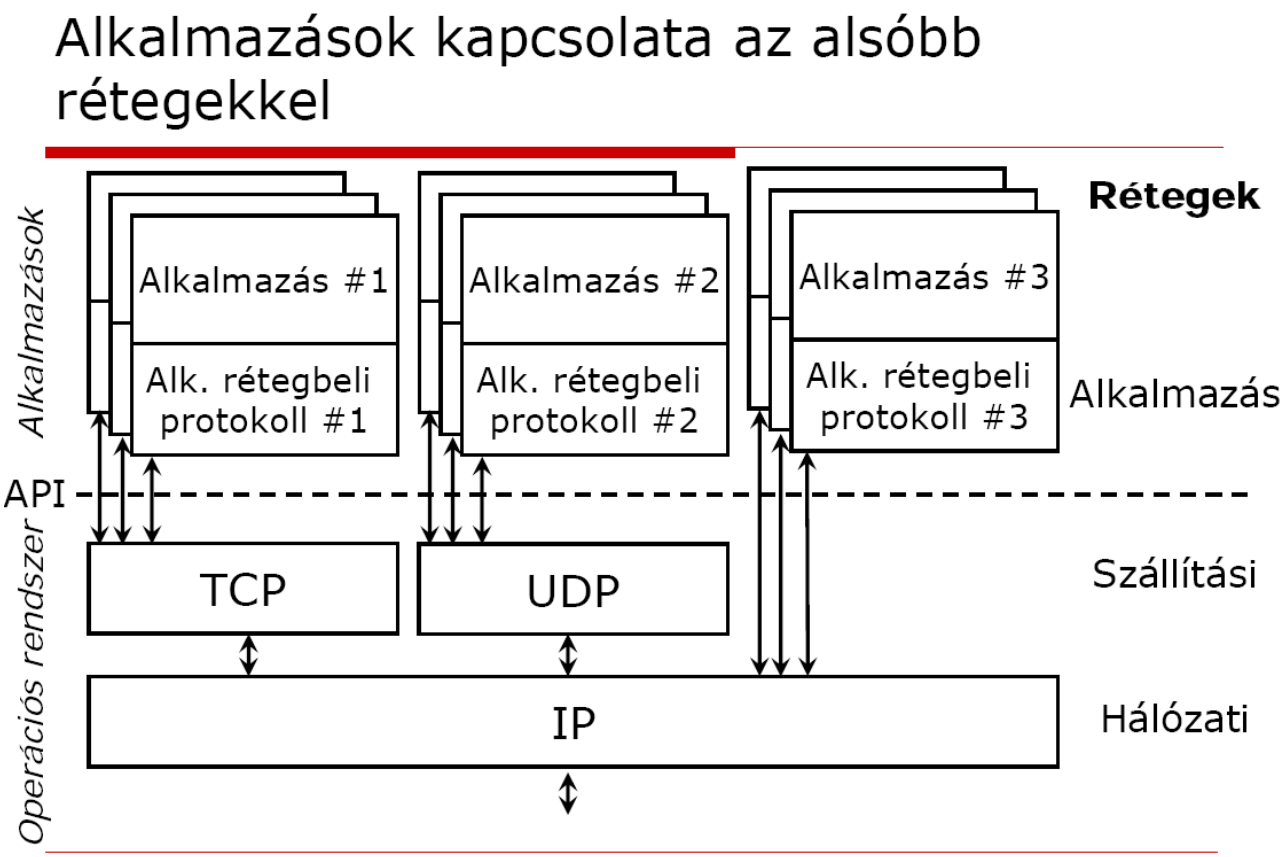
Alkalmazás-rétegbeli protokollok:

Általában az alkalmazásban implementálják, igazodnak az alkalmazás igényeihez és logikájához, ezért többé-kevésbé eltérnek egymástól. Bizonyos fokú szabványosítás viszont szükséges, mert gyakran ezen protokolloknak együtt kell működniük



A kommunikációs csatorna létrehozása nem tartozik az alkalmazás hatáskörbe, ezt a feladatot az oprendszer látja el, és egy szabványosított API-t biztosít az alkalmazások részére. Tehát az alsóbb rétegek kezelését nem a programozónak kell megírnia, ha pl. egy TCP socketen keresztül kell kommunikálnia a programnak, csupán használja az ennek megfelelő API-függvényt.

Kliens – server architektúra:

Adott egy kiszolgáló (server), ami kérésekre várakozik, ezeket igyekszik teljesíteni. Az ezt megvalósító programot egy porttal azonosítjuk az adott gépen, ezt 1-65536 port tartományból lehet kiválasztani, kiszolgálóknál ezt általában az 1-1023 tartományból teszik meg, ez a port statikusan hozzárendelődik a kiszolgáló programhoz

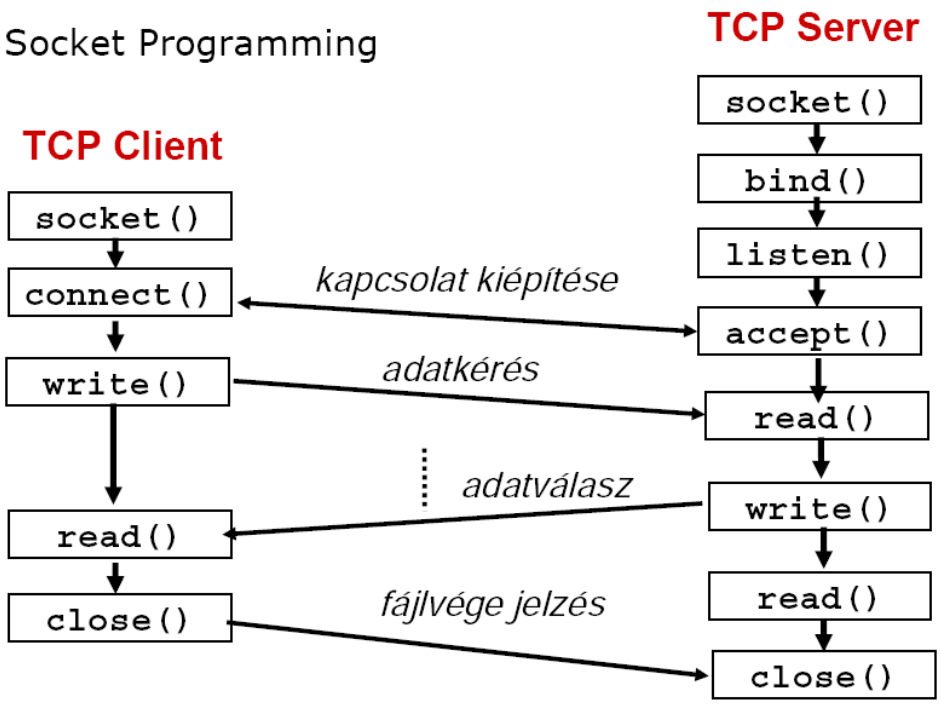
A klienshez viszont dinamikusan szokás portot rendelni, mikor a kérés megtörténik, az 1024-65536 tartományból.

Egy portot egy végponton nem rendelhetünk hozzá több kiszolgálóhóz.

Példák protokollokra és portokra

- IP protokollok (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)
- Ugyanígy TCP és UDP portokra
  - Az alkalmazás igényei (megbízhatóság) szerint
- Az IANA felügyeli ezeket az azonosítókat

TCP kommunikáció socket-hívásokkal



socket() - socket létrehozása
bind() - kapcsoloódunk az előbb létrehozott sockethez, hozzárendelünk egy portszámot a kiszolgáló programhoz

listen() - pl. blokkolva kérésre vár a kiszolgáló (lehet polling módszerrel is ellenőrizni, hogy jött-e kérés)

Telnet:

Kezdetleges távoli paranccsort valósított meg, a legnagyobb hibája az, hogy teljesen mellőzi a csomagok titokosítását, így minden nehézség nélkül le lehet pl. sniffelni az általa generált forgalmat, jelszavastul mindenestül. Manapság még beágyazott rendszerek konfigurálásánal szokás még használni (pl. switchek, routerek, stb.), ahol ez nem jelent nagy problémát, minden más esetben SSH-t (Secure Shellt) ajánlott alkalmazni.

FTP:

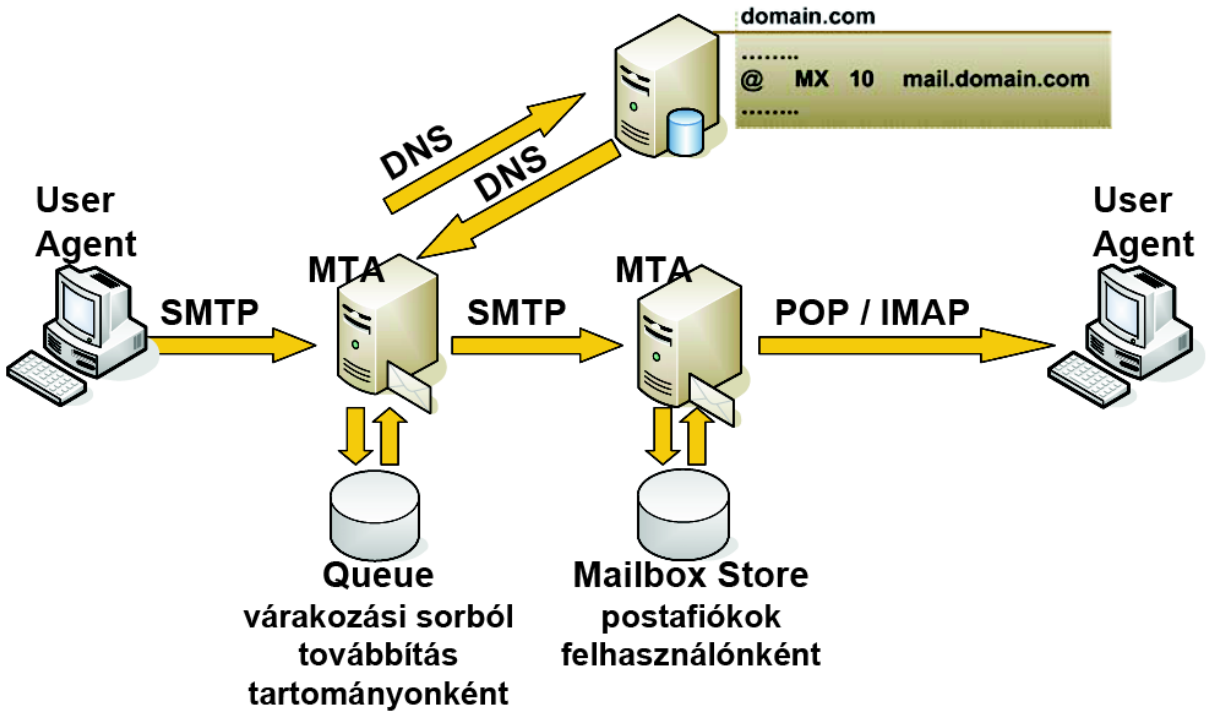
A legelső fájlátviteli protokoll, talán a legismertebb is. Szintén nagy hiányossága a titkosítás teljes mellőzése, ezért ajánlott legalább az SFTP használata, ami ezt a problémát kiküszöböli.
A szabvány szerint a 21-es portot használja a protokoll a vezérlés lebonyolítására, míg a 20-as porot adatcsatornaként használja. Aktív ill. passzív módokban is képes működni, ez a server oldaláról nézve értendő, aktív módban a server nyit adatkapcsolatot a kliens felé, passzívban fordítva.
A vezérlés állapotkódokkal (parancsokkal) történik:

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
  - ...
- Adattípus figyelembevétele (megjelenítési réteg)
  - ASCII
  - Binary

Levelezés:

Levelező rendszerek



Egy e-mail küldésének a folyamata: a levél megírását és megcímzését követően a felhasználónál futó **User Agent** az SMTP protokollt alkalmazva elküldni a levelet a felhasználó által megadott **Mail Transfer Agent** -nek, amely berakja azt a küldésre váró levelek várakozási sorába. Küldéskor a DNS szolgáltatás segítségével lekérdezi, hogy az adott domainhez milyen MX rekord tartozik. Ez a rekord adja meg a domain levelezést lebonyolító kiszolgálójának az IP címét. Amennyiben több MX rekord is tartozik a domainhez, akkor lehetséges ezeket priorizálni pl. a terhelés elosztása, rendundancia miatt. A felhasználó MTA-ja elküldi a levelet a cél-MTA-nak, amely berakja azt a címzett postafiókjába. Amikor a címzett ellenőrzi a postafiúkját, látja, hogy új levele jött, és letölti azt, teszi mindezt a **POP3**, vagy az **IMAP** protokollokkal.

POP3:
-Post Office Protokoll version 3
-parancsorientált
-a 110-es TCP portot használja
-a POP3S változata TLS titkosítást is használ
-hátrányként írható fel, hogy miután a felhasználó letöltötte az új leveleit, azok rögtön törlődnek is a mail-serverről, tehát a leveleknek ekkor az egyetlen példánya a felhasználónál van, ami veszélyes/kényelmetlen lehet

IMAP4:
-Internet Message Protocol version 4
-parancsorientált
-a 143-as TCP portot használja
-az IMAP4S változata a 993-as TCP portot használja, TLS titkosítással
-jobb a POP3-nál az alábbi területeken:
-támogatja a könyvtárstruktúrát, jobban lehet rendszerezni a leveleket

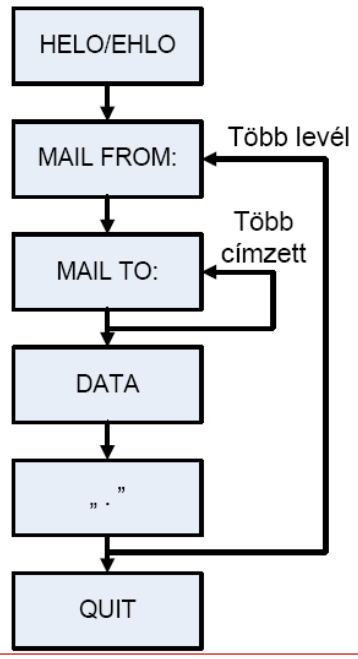
-lehet vele keresni a levelek közt
-az új levelek letöltésekor a serverről nem törlődnek azok automatikusan, így több helyről is könnyen le lehet kérdezni a leveleket pl.

SMTP:
-feladata a levelek továbbításának megvalósítása
-a TCP 25 portot használja
-parancsorientált állapotkódokkal
-lehetséges a relay-ezés, vagyis a nem közvetlen levéltovábbítás, egy, vagy több smtp-relay server közbeiktatásával
-SMTPS változata a TCP 465 porton, TLS titkosítással továbbítja a leveleket
-ESMTP változata kibővített parancskészlettel rendelkezik

A leggyakoribb SMTP parancsok

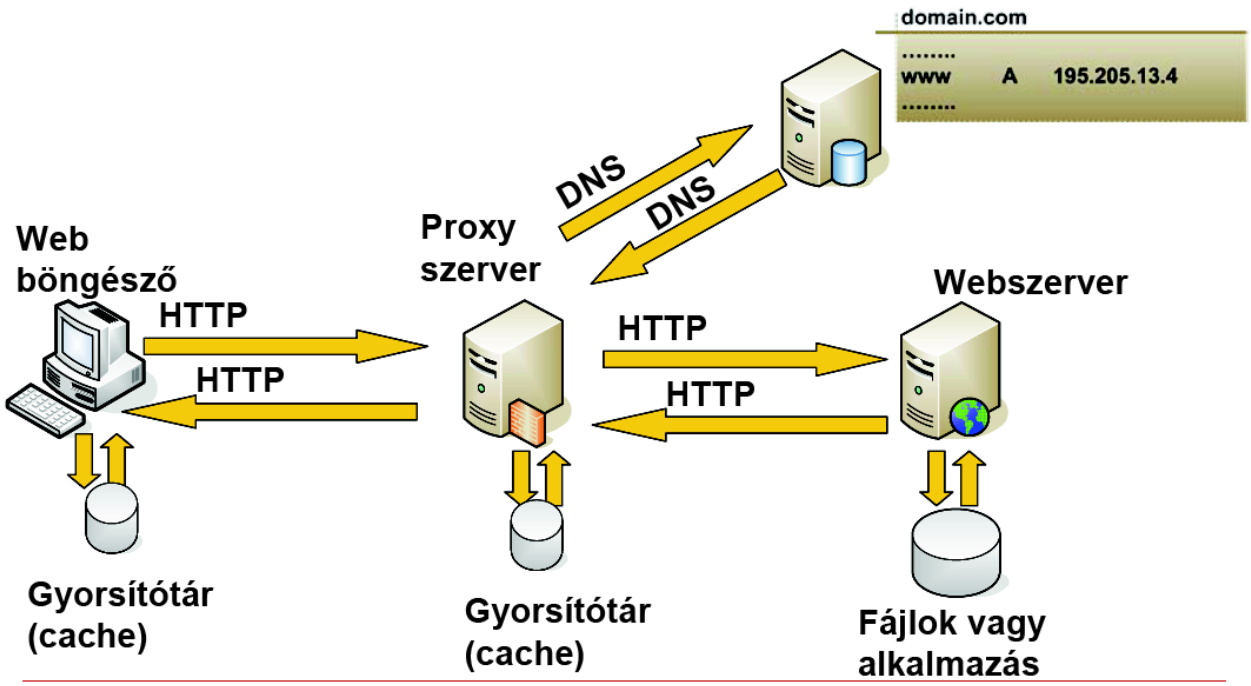
- HELO
  - Üdvözlés
  - ESMTP esetén EHLO
- MAIL FROM:<feladó e-mail címe>
- RCPT TO:<címzett e-mail címe>
- DATA
  - Adat következik
- <CR><LF>.<CR><LF>
  - Adat vége
- QUIT
  - SMTP kapcsolat bontása
- VRFY <e-mail cím>
  - Létezik-e az adott e-mail cím
- HELP
- NOOP
  - Kapcsolat ellenőrzése, fenntartása

Tipikus kapcsolat folyamatábrája



Webes rendszerek:

Webes rendszerek proxyval



Egy oldal lekérése HTTP-vel:
Miután a felhasználó megadta az oldal címét, a böngésző ellenőrzi, hogy be lehet-e tölteni az oldalt a gyorsítótárból. Amennyiben nem, akkor az esetlegesen beállított proxy-tól kéri az oldalt. Ha a proxy el tudja küldeni azt a gyorsítótárjából, akkor megteszi, de ha nem (mert pl. még senki sem volt kíváncsi arra az oldalra, vagy már lejárt az ahhoz tartozó chache-bejegyzés), akkor DNS segítségével lekérdezi a keresett domainhez tartozó kiszolgáló IP címét, majd http kérést küld a Webservernek. Egy webserver persze több domainért is felelős, így gyakran megnézi, hogy a http kérésben milyen domain szerepel, és innen tudja, melyik oldalt kell visszaküldenie.

HTTP:

-TCP 80-t használja a szabvány szerint
-parancsorientált állapotkódokkal
-speciális fejléceket alkalmazhat
-a proxy a kliens nevében jár el, a 8080 TCP portot használja, leginkább a gyorsítótár miatt szokás alkalmazni.

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 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

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)

HTTP alkalmazási területei

- HTML
  - statikus oldalak
    - Beágyazott tartalommal (pl. képek)
  - dinamikus oldalak
    - PHP, ASP, JSP, CGI, DHTML, ASPX, (ActiveX Control)
- Fájl le- és feltöltés
  - Kiegészítés: WebDAV
    - HTTP parancs- és fejléc-bővítmény
    - FTP-hez hasonló fájlkezelés
    - Hozzáféréskezelés (meg van nyitva írásra, ennek megújítása)
- Webszolgáltatások (WebService)
  - RPC-hez hasonló távoli eljáráshívás
  - SOAP
    - HTTP-n XML alapú kérés/válasz
- Protokollalagút
  - Más protokollokat HTTP-be csomagolva visznek át
  - Tűzfalak kijátszása (HTTP 80-as port mindenhol engedélyezve)

HTTP biztonság

- Hitelesítés
  - Névtelen hozzáférés (Anonymous)
  - Alapvető (Basic)
    - Base64 kódolás
  - Digest
    - Challenge/response alapú
  - Integrált (Integrated)
    - LANMan
    - NTLM
    - Kerberos
  - Tanúsítvány (Certificate)
  - Egyéb
- HTTPS
  - HTTP SSL-en (PKI:RSA; DES, 3DES)
  - TCP 443

HTTP áttekintés

- Webes rendszerek
- HTTP-ről általában
- HTTP parancsok
- HTTP fejlécek
- HTTP kódok
- HTTP proxy és cache
- Gyakori hitelesítési protokollok
- HTTPS: biztonságos HTTP
- WebDAV
- Gyakori alkalmazásai:
  - HTML: statikus és dinamikus oldalak
  - Fájl le- és feltöltés
  - Webszolgáltatások (SOAP)
  - Protokollalagút

Hálózatbiztonság:

-Authentication: Hitelesítés
- -Azonosítás és hitelesség
- -Kerberos:
  - -adott tartományon belül működő rendszer
  - -idbőbélyeget használ
- -PKI (Public Key Insfrastructure):
  - -nyilvános kulcsú titkosítás alkalmaz
  - -Digitálisan aláírt tanúsítványokat kiállító szervezetek, CA-k (Certificate Authority) alkalmazása
  - -a legfelsőbb szintű CA-ban mindenképpeb meg kell bízni, mindegyik CA a felette levőben bízik meg, ezek a szervezetek adhatnak ki tanúsítványokat

-Authorization: jogosultság- hozzárendelés

-Accounting: számlázás

Megoldások titkosításra

- Szimmetrikus kulcsú
  - DES
  - 3DES
  - AES
- Nyilvános kulcsú
  - RSA
  - Elliptikus görbék

- Részletesebben: *Kódolástechnika* c. tárgy

Összetett megoldások

- IPSec
  - Hitelesítés, kulcscsere- és titkosítás-egyeztető és –megvalósító keretrendszer
  - Két fő típus
    - Authentication Header (AH): digitálisan aláírt IP-csomag beleértve az IP-fejlécet is
    - Encapsulated Security Payload (ESP): Titkosított tartalmú csomag
  - IPv6-ban kötelezően támogatott
- SSL
  - SSH, FTPS, HTTPS, SMTPS, POP3S, IMAP4S, …
- TLS

- Részletesebben: *Kódolástechnika* c. tárgy

A hálózatbiztonság eszközei

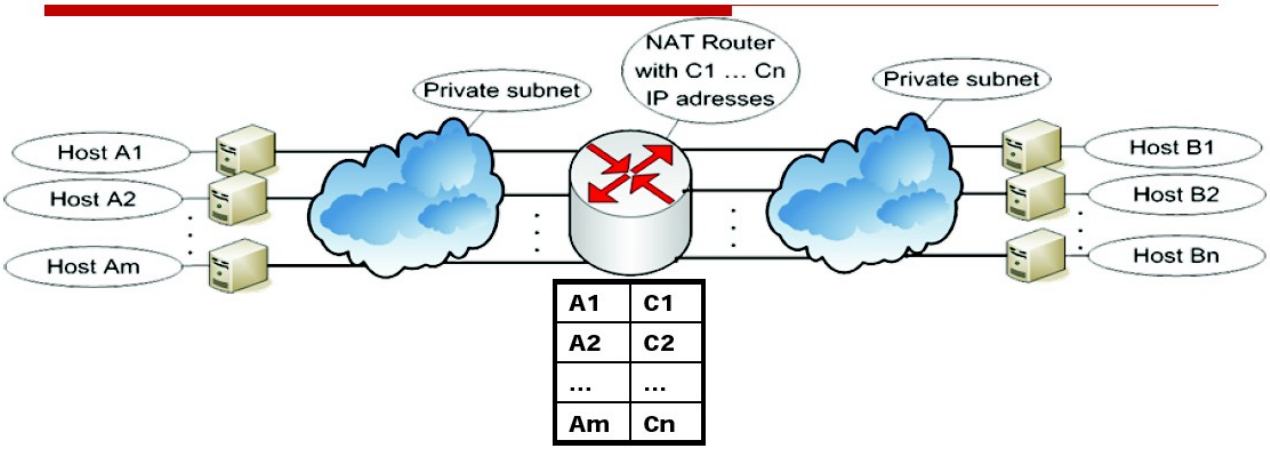
- Hálózati eszközök
  - Tűzfal (FW: FireWall)
  - Behatolás-észlelő rendszer (IDS: Intrusion Detection System)
  - Behatolás-megelőző rendszer (IPS: Intrusion Prevention System)
- Biztonságos alkalmazás (tervezés, implementálás, terjesztés,…)
- Biztonságosan üzemeltetett/használt alkalmazás (Social Engineering)

NAT:

Network Address Translation

Feladata a belső, vagy magánhálózatok és az Internet összekötése, címfordítással. A belső hálózatok alkalmazott címek nem használhatóak közvetlen módon a neten, tehát pl. a 192.168.0.1 címmel a belső hálózaton levő gépet nyilván nem tudja egy külső szemlélő ebben a formában elérni. Ehhez címcserére van szükség, vagyis hálózati rétegbeli átalakításra.Ezt a műveletet routernek kell elvégeznie, több lehetőség fordulhat elő:

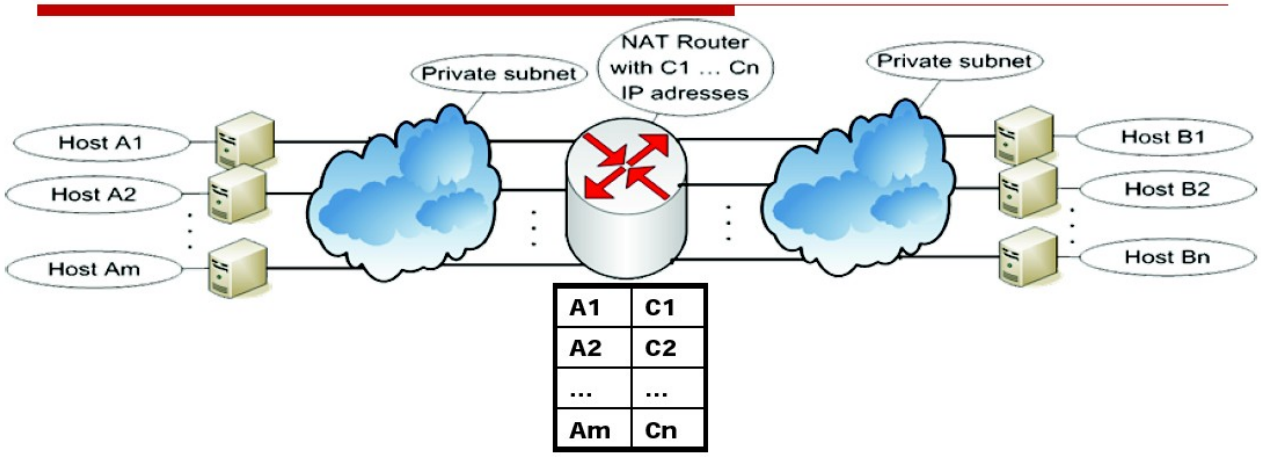
NAT router több IP-címmel (m=n)



- A router rendelkezik megfelelő számú nyilvános címmel, hogy minden belső címhez hozzárendeljen egyet
- Külső-belső címösszerendelés
  - Statikus
  - Dinamikus – biztonsági szempontból előnyös

Ebben az esetben a router megfelelő mennyiségű nyilvános címmel rendelkezik, hogy minden belső hálózaton levő eszközhöz rendeljen külön egyet-egyet, így a címfordítási feladat egyértelművé válik. Meg lehet oldani statikus és dinamikus hozzárendeléssel is. A router egyszerűen csak kicseréli a megfelelő ki- és bemenő csomagokban szereplő címeket a megfelelő belső / nyilvános címekre, iránytól függően.

NAT router kevés IP-címmel (m>n)

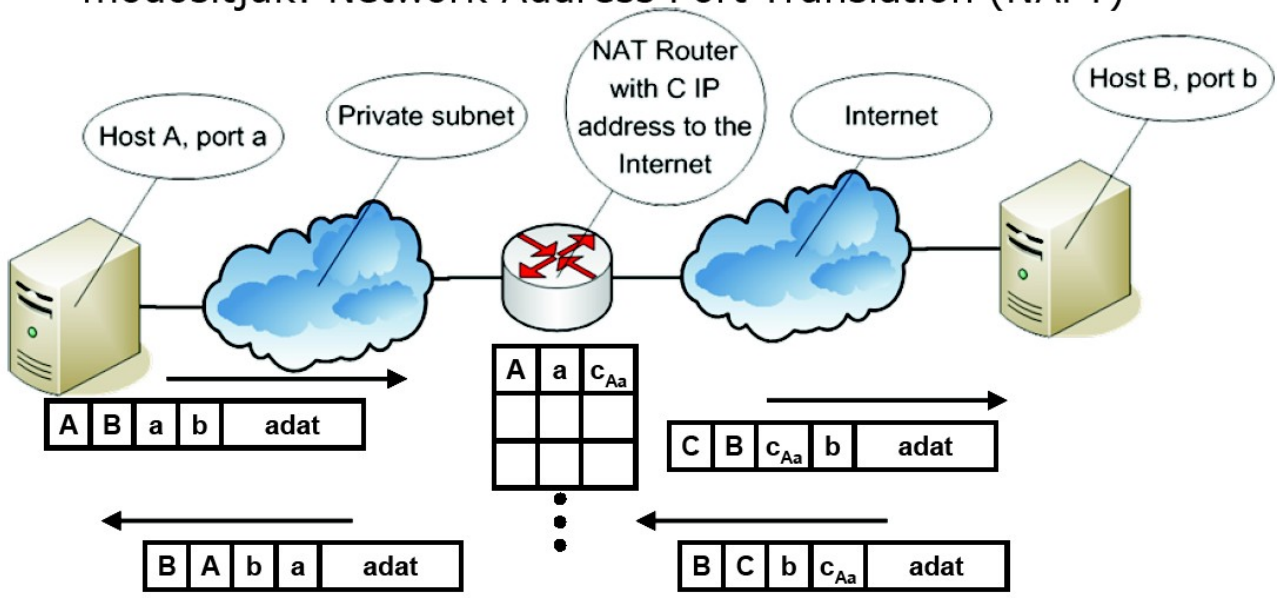


- A router nem rendelkezik megfelelő számú nyilvános címmel, hogy minden belső címhez hozzárendeljen egyet
- Külső-belső címösszerendelési stratégiára van szükség

Ebben az esetben nem lehet 1-1 leképezést megvalósítani a belső és külső címekre. Probléma: hogyan talál vissza a megfelelő belső állomásra a vissza-irányú kommunikáció?
A megoldás, hogy router az egyes belső állomásokból érkező adatfolyamot nem ugyanazon a porton továbbítja, amin az eredetileg elhagyta az állomást, hanem egyedi, attól eltérő portot ad ennek az adatfolyamnak. Így amikor érkezik a válasz, akkor erre az egyedi portra címezve érkezik meg, és a router ekkor tudni fogja, hogy melyik végpontnak továbbítsa a csomagokat, visszafordítja a portot a csomagokban.

NAT – Hogyan működik valójában?

- Nem csak az IP-címeket, hanem a portokat is módosítjuk: Network Address Port Translation (NAPT)



NAT – Biztonsági megoldás

- A NAT táblát bővíteni kell a külső fél IP és port címével
- A NAT router eldob minden más állomástól jövő csomagot
- Ezt a módszert *maszkolás*nak (*masquerade)* hívják
- Egy egyszerű tűzfalmegoldás