Hálózati Alkalmazások

Karanjit S. Siyan: Inside TCP/IP Third Edition Chapter 13, 1997

Fordította: Kismődi Tamás, dr Lencse Gábor

A TCP/IP felett működő hálózati alkalmazások magukban foglalják az OSI 5., 6., és 7. rétegének funkcionalitását. Nem minden alkalmazás igényli az 5. és 6. réteg szolgáltatásait. A hálózati alkalmazások foglalkoznak az alkalmazási réteg adatainak formázásával, küldésével és fogadásával. Azonban nem tartalmazzák a felhasználó felé nyújtott kezelői felületet.

Ez a jegyzet a következő főbb hálózati alkalmazásokat tárgyalja, melyekhez szükséges némi előtanulmány a TCP/IP protokoll infrastruktúrájáról:

- Domain Name System (DNS)
- Levelező protokollok: SMTP, POP3, IMAP4
- Távoli elérési protokollok: Telnet, Berkeley r*
- Fájl átviteli protokollok: FTP, TFTP
- Fájl hozzáférési protokollok: NFS
- Web hozzáférési protokollok: HTTP/HTML, Gopher

1. Domain Name System

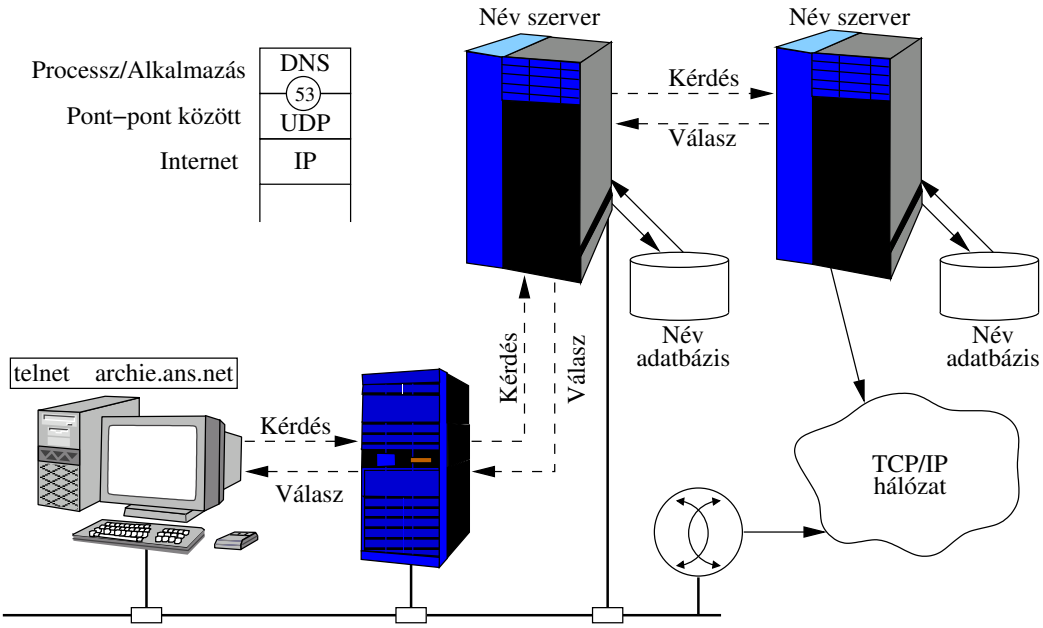
A modern Internetben a host-ok a Domain Name System (DNS) mechanizmusa alapján kapnak nevet. A DNS minden host mellé rendel egy szimbolikus nevet, melyen elérhető. A DNS-t közvetlenül használja majdnem minden hálózati alkalmazás, mert a felhasználók tipikusan úgy hivatkoznak a host nevekre, mint a DNS nevekre. Példa képpen, ha egy felhasználó egy Telnet kapcsolatot akar létrehozni, a következő parancsot adja:

```
%telnet archie.ans.net
Trying 147.255.1.2...
Connected to nis.ans.net.
Escape character is '^]'.
AIX telnet (nis.ans.net)
IBM AIX Version 3 for RISC System/6000
(c) Copyrights by IBM and by Others 1982, 1990
login:
```

A Telnet kapcsolat azonnali első válasza a következő üzenet:

```
Trying 147.255.1.2...
```

A TCP/IP szoftver lefordította a host nevet `archie.ans.net` egy 32-bites IP címre (147.255.1.2). Ezt a fordítást hajtotta végre a DNS (lásd 1. ábra).



1. ábra. DNS névfeloldási példa

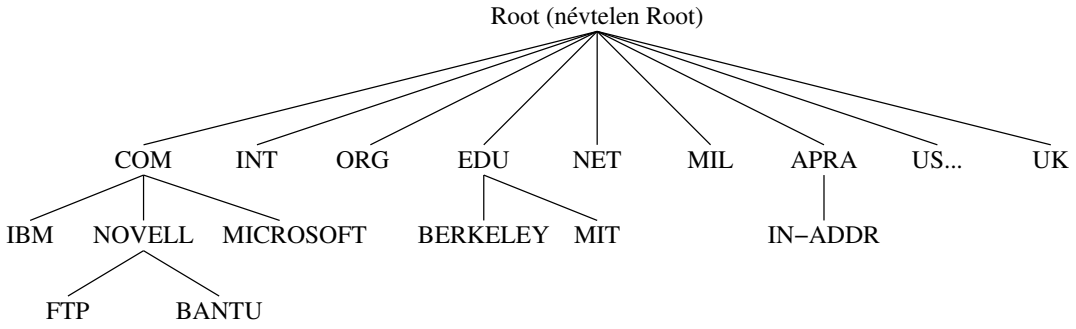
A TCP/IP alkalmazások beállíthatók oly módon, hogy használják a DNS névfeloldást. Amennyiben egy ilyen TCP/IP alkalmazás találkozik egy host névvel, akkor küld egy kérdést a *név feloldónak* (name resolver), hogy adja meg a névhez tartozó IP címet. Néhány rendszerben a névfeloldás történhet ugyanazon a munkaállomáson amelyen felmerült a kérdés. A névfeloldó általában tartalmaz egy könyvtárprogram készletet a rendszerben.

Ha a névfeloldó nem találja a választ, küld egy kérdést a *név szerver* felé. A név szerver általában nem a kérdést feltevő munkaállomás hálózatában tartózkodik. Ha a név szerver nem találja a választ, a kapcsolat nem jöhet létre a másik host-tal a TCP/IP hálózaton.

A DNS kérdés/válasz típusú kommunikációt követ és UDP-t használ, mint szállítási protokollt. Az UDP sokkal kényelmesebb a kérdés/válaszon alapuló alkalmazásoknak, mert nincs kapcsolat fenntartási vezérlés az adatátvitelben. TCP protokollt is alkalmazhatunk kérdés/válaszon alapuló alkalmazásoknál, azonban be kell építenünk egy kapcsolat felépítést, illetve egy kapcsolat bontást, ha a kérdés/válasz folyamat lezajlott. Ha előre láthatóan csak egy egyszerű kérdés/válasz vagy egy alkalmi kérdés/válasz folyamatra lesz szükség a vezérelt kapcsolat felépítés és bontás túlzásnak minősíthető.

A fejezetben leírt Telnet példában a hálózati alkalmazás DNS-t használt, hogy a nevet lefordítsa IP címmé. Igaz, elsődleges feladta a DNS-nek, hogy feloldja a DNS szimbolikus neveit IP címekké, azonban van egy másik fontos tevékenysége is, méghozzá a kapcsolat a levelező protokollokkal. A DNS-t speciálisan a levelező szerverek is alkalmazzák, hogy kezeljék az elektronikus levélcímeket.

A DNS legszélesebb körben használt megvalósítása a Berkeley Internet Domain Name (BIND) szerver, amely eredetileg a BSD Unix-ban volt elérhető. Most elérhető a legfontosabb Unix platformokon. A Unix rendszerekben a BIND-ot megvalósító program neve: *named* (name daemon).



2. ábra. Hierarchikus nevek a DNS-ben

Korábban a fejezetben már látható volt egy host, név mely több szakaszból állt. Ezen típusú névhasználat egy egyezményes megállapodásból született. A hierarchikus név rendszerben amelyet a DNS használ a név egy hierarchikus fában helyezkedik el. A fa legtetején áll a root domain, melynek a neve a pont szimbólum („.”). Mivel minden név ezen közös root-hoz tartozik ezt a pontot elrejtjük, ha hierarchikus nevet adunk meg a főbb TCP/IP alkalmazásokban. A root domain felső szintjének részlete látható a 2. ábrán. Példák láthatók a felső szintű domain-ekre a 1. táblázatban.

A CCITT szabvány (mai nevén International Telecommunications Union, vagy ITU szabvány) illetve az ISO 3166 szabvány szerint az országra utaló kétbetűs megnevezést használjuk államok esetében (kivétel a Great Britain, ahol inkább a UK-t használják a GB helyett). Egy országon belül találhatók meg a közép szintű domain-ek, melyek azonos felső

szintű domain-nel rendelkeznek. A nevet szakaszokra osztva különböztetjük meg egymástól. A domain nevek nem különböztetnek meg kis illetve nagy betűket. A maximális hossza egy komplett domain névnek nem haladhatja meg a 255 karaktert. Példa:

`rs1.szif.hu`

Az `rs1.szif.hu`-ben a host neve az `rs1`, amely az `szif.hu` domain-en van.

Ha egy másik host neve `fossil` és ugyancsak a `szif.hu` a domain-en helyezkedik el, akkor *fully qualified name* (FQN) a következő:

```
fossil.szif.hu.
```

A fenti címben a `fossil` a relatív név, míg a `fossil.szif.hu.` az abszolút név (a „.” elhagyható). Az FQN középső része utal a szervezetre. Egy szervezet szabadon definiálhat *subdomain*-eket a szervezeten belül, de ha megteszi akkor fel kell töltenie a hozzá tartozó domain név szerver adatbázisát, hogy az végre tudja hajtani a névfeloldásokat. Példaként adott egy szervezet, melynek a domain neve a következő:

```
bme.hu
```

Ha ennek a szervezetnek különálló hálózatai vannak a hiradástechnikai, távközlési és telematikai, folyamatszabályozási tanszékeken, így definiálhat subdomain-eket mint a `hit`, `ttt`, `fsz`, és ezt az információt továbbítja a DNS szerverének vagy több DNS szervernek, hogy el tudják végezni a névfeloldást, akkor a következő neveken lehet elérni a szervezet ágazatait:

```
hit.bme.hu

ttt.bme.hu

fsz.bme.hu
```

Nem szükséges minden domain-hez egy DNS szervert alkalmazni, egy közös szerver ellátháat több domain-t is. A 2. ábrán látható néhány DNS szerver a root domain alatt. Ezek a szerverek több domain-t is ismernek a felső szintű domain-ekből, mint `com`, `edu`, `mil`, `org`, `net`, stb.

Számos név szerver kezeli a domain neveket a root domain-en. A 2. táblázatban látható néhány a root domain szerverek közül.

Felső szintű Domain	Megnevezés
COM	Gazdálkodó/üzleti szervezet
EDU	Oktatási intézmény: egyetemek, iskolák, stb.
MIL	Katonaság
GOV	U.S. közigazgatás
NET	Hálózat ellátó
ORG	Non-profit szervezet
ARPA	ARPANET: történelmi, fordított címleképzésre használják
INT	Nemzetközi szervezet
HU	Ország: Magyarország
US	Ország: Egyesült Államok
CA	Ország: Kanada
UK	Ország: Egyesült Királyság
DE	Ország: Németország
SE	Ország: Svédország
FR	Ország: Franciaország
IN	Ország: India
CN	Ország: Kína
JP	Ország: Japán

1. táblázat. Példa néhány felső szintű domain-re

Host név	Hálózat cím	Szerver program
`ns.internic.net`	198.41.0.4	BIND (Unix)
`ns.nic.ddn.mil`	192.112.36.4	BIND (Unix)
`ns1.isi.edu`	128.9.0.107	BIND (Unix)
`aos.arl.army.mil`	128.63.4.82	BIND (Unix)
	192.5.25.82	
`c.nyser.net`	192.33.4.12	BIND (Unix)
`terp.umd.edu`	128.8.10.90	BIND (Unix)
`ns.nasa.gov`	192.52.195.10	BIND (Unix)
	128.102.16.10	
`nic.nordu.net`	192.36.148.17	BIND (Unix)

2. táblázat. Root domain szerverek

Mivel a DNS szerverek több domaint is ismerhetnek, képesek a terhelés kiegyenlítés végrehajtására, a felesleges hálózati forgalom elkerülésére, és a megbízhatóságot növelve átvenni az elsődleges DNS szerver feladatát, ha az valami okból nem elérhető. A `com` domain-nek lehet egy vagy több DNS szervere is amely ismeri a kereskedelmi szervezetek neveit a `com` domain-en. A `com` domain-en belül létezhet subdomain, mint `ibm.com`, melynek van saját DNS szervere arra a domain-re. A host-ok a domain-en belül a helyi DNS szervert fogják kérdezni, hogy feloldja a szimbolikus neveket. Például, egy host a `world.std.com` a saját domain-jének az `std.com`-nak a DNS szerverét kérdezi meg, hogy megtalálja az `ftp.novel.com`, vagy a `athena.scs.org` nevű host IP címét. Azután, hogy a kérdést sikerült feloldani, az eredmény egy helyi cache-be tárolódik, egy adott időtartamig, amely beállítható.

Egy domain-hez tartozó DNS szervernek fel kell tudni oldani a domain-en belüli host-ok neveit. A DNS szervernek mindig ismernie kell a szülő DNS szerverének IP címét. Létezik egy primary (elsődleges) DNS szerver, amelyet ha bármi okból nem elérhető egy secundary (másodlagos) DNS szerver helyettesít.

A DNS UDP szállítási protokollt használ, hogy kérdezzen és válaszokat fogadjon a DNS szervertől. A DNS szerver az 53-as UDP porton várja a kérdéseket.

A DNS leírása megtalálható az RFC 1034 (STD 13) "Domain Names: Concepts and Facilities"-ben és frissítve az RFC 1982, 1876 és 1101-ben.

Az RFC 2137-ben "Domain Name System Dynamic Update" és az RFC 2136-ban "Dynamic Updates in the Domain Name System (DNS UPDATE)" néven találhatók meg az újabb módosítások.

2. Levelező protokollok

Az elektronikus levelezés talán a legelterjedtebb a hálózati alkalmazás. Igen sokféle protokoll létezik az elektronikus levelezés megvalósítására, azonban a Simple Mail Transzfer Protocol (SMTP) az egyik legelterjedtebb. A mobil és személyi számítógépen dolgozó felhasználók nagy száma miatt kifejlesztettek több félét is, úgymint a POP3 (Post Office 3-as verzió) és az IMAP4 (Internet Message Access Protocol 4-es verzió).

2.1. Simple Mail Transfer Protocol (SMTP)

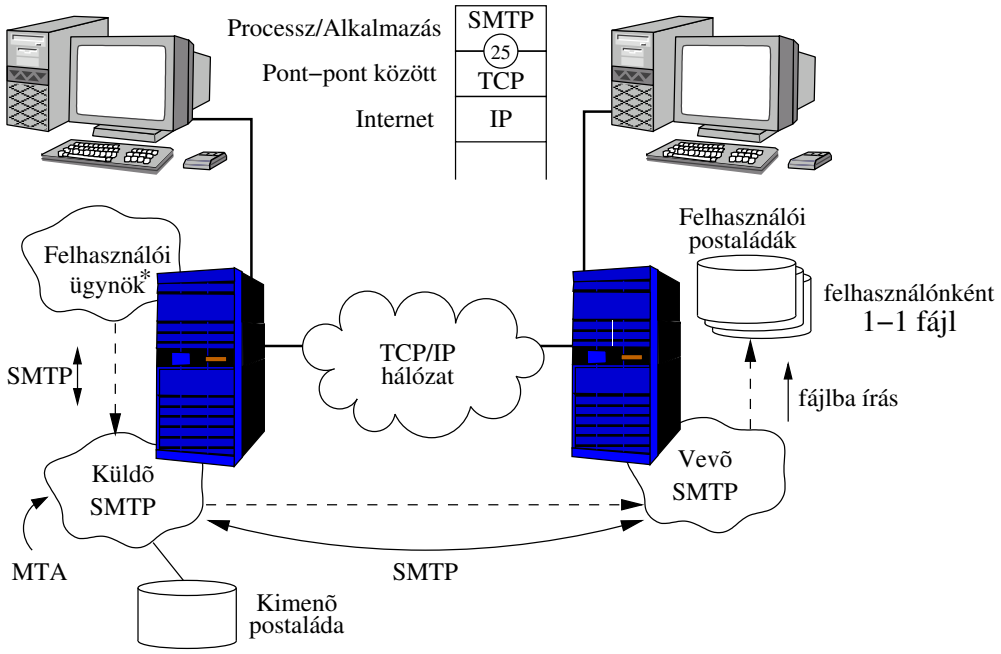
Az SMTP ASCII kódú szöveges üzenetek továbbítására képes TCP/IP protokollt használó host-ok között, ha azok levelezésre is konfigurálva vannak. A 3. ábrán egy SMTP-t használó levelezést mutatunk be. Mikor a felhasználó kezdeményezi a levél küldését, az User Agent (UA) kapcsolatba lép a kimenő SMTP szerverrel (Unix esetén pl. sendmail) és átadja neki a levelet. Az a levél átvitelét azonnal megkísérli, azaz TCP kapcsolatot épít ki a címzett leveleit kezelő SMTP szerverrel és megtörténik az átvitel. Ha ez nem sikerül, akkor egy kimenő postafiókban tárolja a levelet, és bizonyos időközönként újra megkísérli az átvitelt. A vevő oldalon az SMTP process fogadja a kapcsolódást és veszi az üzenetet, így bekerül a levél a címzett bejövő postaládába. Ha a cél host-on nem létezik az e-mail címben megadott postaláda, akkor a feladó erről a levélről értesítést kap A küldő és a vevő SMTP process-eket, melyek a levél átviteléért felelnek, üzenet átviteli ügynököknek nevezzük (Message Transfer Agents, MTAs).

Az SMTP-ben használt levél címek az RFC-822-es szabványt követik, ezért a levelek fejrészét gyakran hívják 822-es fejrésznek. Egy példa a 822-es levélcímre:

```
lencse@tilb.sze.hu
```

A @ jel előtti szöveg string megadja a postaláda nevét, a jel utáni szöveg string pedig a host nevét. Ha a postaláda neve speciális karaktert tartalmaz, mint például a %, akkor az a név egy speciális kódolást alkalmaz, melyet a levél átjárók (mail gateway) használnak. A `lencse@tilb.sze.hu` levélcímben a `lencse` a postaláda neve a `tilb.sze.hu` host-on.

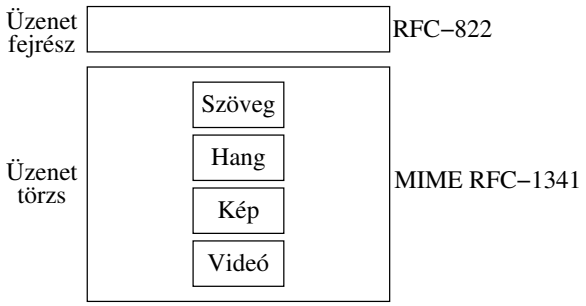
Ha nem szöveges üzenetet akarunk küldeni, hanem bináris fájlt, hangot, képet SMTP-n keresztül, akkor UUENCODE kódolási eljárást használhatunk, amely szöveges üzenetté alakítja az adatot. Ez a kódoló eljárás sok rendszerben elérhető. A vevő oldalán visszaalakíthatjuk az üzenetet a UUDECODE segítségével. Egy másik út, hogy nem szöveges üzenetet küldünk SMTP-n keresztül a MIME (Multipurpose Internet Mail Extensions) protokoll. A MIME megtalálható az RFC



* User Agent:"levelzõ program", amivel a felhasználó a levelet megírja, például pine, eudora, netscape messenger

3. ábra. SMTP felhasználók közötti kapcsolat

1896, RFC 2045, RFC 2046 és az RFC 2049-ben. A MIME képes különböző típusú adatokat is kódolni, mint egyszerű szöveg, gazdagabban formázott szöveg (rich text), kép, hang, videó, HTML dokumentum és így tovább.



4. ábra. MIME üzenet

A MIME üzenet törzs részekre osztott tartalommal rendelkezik, és a MIME felhasználói ügynök válogathat a megjelenítendő adatok közül. Például egy „dumb” (buta) terminál, amely nem képes sem hang lejátszásra, sem kép, sem videó megjelenítésre csak a szöveges részét írja ki az üzenetnek a képernyőre. Egy másik hasznos tulajdonsága, hogy használhat mutatót olyan adatra amely valahol máshol került tárolásra. Például ez a mutató megadhat egy FTP helyet, ahonnan letölthető az adott dokumentum. Ez a process megszünteti annak szükségességét, hogy egy körlevélben mindenkinek elküldjük az adott dokumentumot, csak annak kell megvárnia a letöltést, akit érdekel az információ.

Az alábbi részben látható, hogy a Unix-ban használható *mail* programmal, mint UA miként kommunikálhat a felhasználó.

```
% mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/jok": 1 message 1 unread
>U  1 jok@Portocom.tilb.sz  Sat Jul 20 02:39  15/506   "Minta levél"
```

```
& ?
Mail   Commands
t <message list>type messages
n goto and type next message
e <message list>edit messages
f <message list>give head lines of messages
d <message list>delete messages
s <message list> file append messages to file
u <message list>undelete messages
R <message list>reply to message senders
r <message list>reply to message senders and all recipients
pre <message list>make messages go back to /usr/spool/mail
m <user list>mail to specific users
q quit, saving unresolved messages in mbox
x quit, do not remove system mailbox
h print out active message headers
! shell escape
cd [directory] chdir to directory or home if none given

A <message list> consists of integers, ranges of same, or user names
separated  by spaces.  If omitted, Mail uses the last message typed.

A <user list> consists of user names or aliases separated by spaces.
Aliases are defined in .mailrc in your home directory.
& m jok
Subject: Bemutató levél üzenet
Ez a levél tartalma.
Ahhoz hogy befejezzük a levelet egy üres sor első karaktere ként
,,.'' jelet kell tenni, majd ,,enter''-t ütni.
.
& h
> 1 jok@portocom.tilb.sze.hu Sat Jul 20 02:50:07 2002
& p
Message 1:
From jok  Sat Jul 20 02:50:07 2002
Date: Sat, 20 Jul 2002 02:50:06 +0200
From: Kismodi Tamas <jok@Portocom.tilb.sze.hu>
To: jok@Portocom.tilb.sze.hu
Subject: Bemutató levél üzenet
Ez a levél tartalma.
Ahhoz hogy befejezzük a levelet egy üres sor első karaktere ként
,,.'' jelet kell tenni,  majd ,,enter''-t ütni.
& x
%
```

Az SMTP parancsok amelyekkel üzenetet küldhetünk a 3. és 4. táblázatban láthatók. A táblázatok tartalmazzák a minimális eszközkészletét mind a küldő, mind a vevő oldalnak. Minden SMTP parancs négy karakter hosszú. Az SMTP vevő tipikusan egy levelező szerver, amely az SMTP parancsok fogadása után egy három számjegyes állapot kóddal válaszol, ami a következőképen néz ki:

nnn Szöveges üzenet

Az *nnn* a három számjegyből álló állapot kód.

Parancs	Jelentés
HELO *küldő*	A küldő gép, amelyen a UA fut.
MAIL FROM: *feladó_címe*	Ez a parancs adja meg a feladó postaláda címét.
RCPT TO: *célcím*	Ez a parancs adja meg a cél postaláda nevét. Több címzett esetén többször kell alkalmazni a parancsot.
DATA	A parancs után lehet megadni a levél tartalmát. Az üzenet végén a következőnek kell állnia `<CRLF>.<CRLF>`.
QUIT	A parancs hatására a vevő `OK` választ küld és bezárja a kapcsolatot.
RSET	Ez a parancs törli az épp aktuális folyamatot.
NOOP	Ez a parancs nem hajt végre műveletet. A vevő egy OK választ ad. A parancs akkor hasznos, ha meg akarunk győződni róla, hogy a kapcsolat rendben van-e.

3. táblázat. SMTP küldő (kliens) parancsok

Parancs	Jelentés
250	Kért levél művelet `OK`, sikeresen befejeződött.
251	A felhasználó nem helyi, továbbításra kerül a megadott útvonalra.<forward-path>
450	Kért levél művelet nem történt meg, a postaláda nem elérhető. Például a postaláda foglalt.
550	Kért levél művelet nem történt meg, a postaláda nem elérhető.
451	Hiba a kért műveletben.
551	A felhasználó nem helyi, kérlek add meg az útvonalat.<forward-path>
452	Kért levél művelet nem történt meg. Nincs elég szabad hely.
553	Kért levél művelet nem történt meg. A postaláda neve nem elérhető. Például szintaktikailag hibás.
354	Kezdje a levelet. Befejezés a `<CRLF>.<CRLF>`.
554	Művelet nem sikerült

4. táblázat. SMTP vevő válaszai

A 5. táblázatban látható egy példa arra, ha SMTP parancsokkal akarunk levelet küldeni. (A K jelöli a küldőt, a V pedig a vevő felet.)

```
K:  HELO gép
V:  HELO gép, Pleased to meet you

K: MAIL FROM: jok@tilb.sze.hu
V: 250 OK

K: RCPT To: lencse@tilb.sze.hu
V: 250 OK

K: RCPT To: kazmer@tilb.sze.hu
V: 550 No such user here

K: RCPT To:kutya@pittypang.hu
V: 250 OK
```

```
K: DATA
V: 354 Start mail input; end with <CRLF>.<CRLF>
K: üzenet szövege
K: üzenet szövege
K: <CRLF>.<CRLF>
V: 250 OK
```

5. táblázat. Levél küldése SMTP parancsokkal

Ez a levél kézbesítve lett a `lencse@tilb.sze.hu` postaládacímre, mivel levél szerver válasza `OK` volt. Azonban a `kazmer@tilb.sze.hu` címre nem, mert hibaüzenet volt a válasz (550-es állapot kód) azaz a postaláda nem elérhető. A távoli gépen lévő
`kutya@pittypang.hu` cím helyeségét nem ellenőrzi.

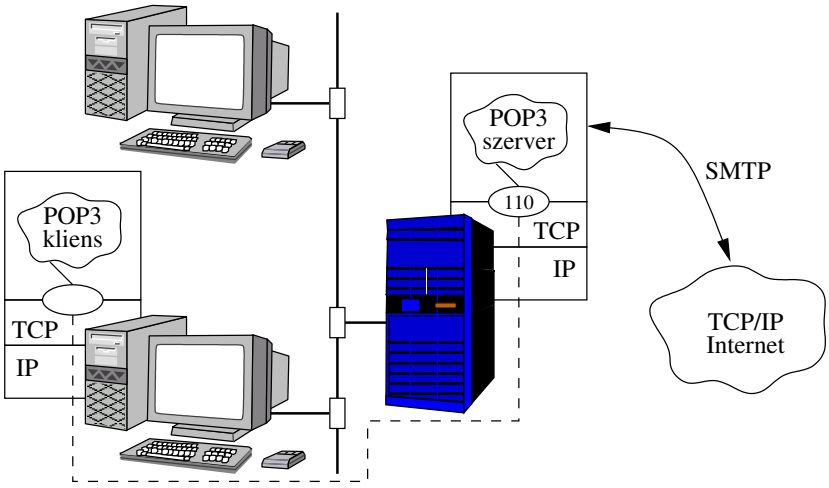
Az SMTP-ről szóló dokumentumok az 6 táblázatban láthatók.

Protokoll	Név	RFC#	STD#
SMTP	Simple Mail Transzfer Protocol	821	10
SMTP-SIZE	SMTP Service Ext for Message Size	1870	10
SMTP-EXT	SMTP Extensions	1869	10
MAIL	Format of Electronic Mail Messages	822	11

6. táblázat. SMTP dokumentumok

2.2. Post Office Protocol Version 3 (POP3)

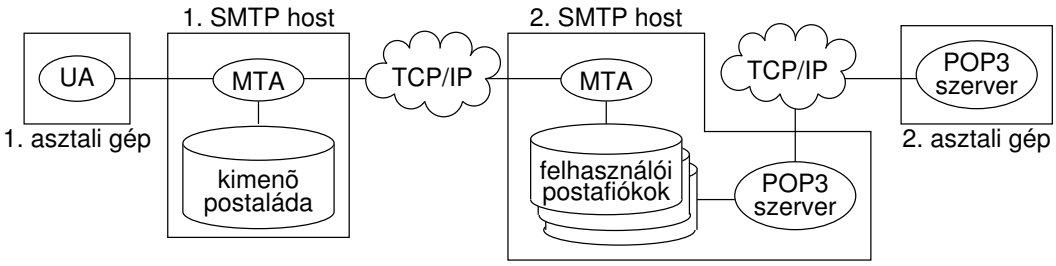
Az SMTP protokoll elvárja, hogy a levelet fogadó mail szerver on-line (bekapcsolt és a hálózaton elérhető) legyen, különben a TCP kapcsolat nem hozható létre. Éppen ezért nem praktikus asztali számítógépeknél kizárólag SMTP alapokra bízni a levelezést, mivel az asztali gépeket munka után ki szokták kapcsolni.



– Az üzenet átviteli ügynök (MTA) egy erõsebbszámítógépen fut.
Postafiók szolgálatot nyújt a kisebb gépeknek, munkaállomásoknak.
– APOP3 dinamikus hozzáférést biztosít a szerverhez.

5. ábra. POP3 kliens/szerver architektúra

Sok hálózati környezetben az SMTP levelek fogadását egy olyan SMTP host végzi, amely mindig be van kapcsolva (lásd 5. ábra). Ez az SMTP host látja el a postafiók (mail-drop) szolgálatot. A munkaállomások kapcsolatba lépnek az SMTP host-tal, majd letöltik az üzeneteket egy kliens/szerver levelező protokoll-lal, mint például a POP3 (Post Office Protocol version 3), melynek leírása megtalálható az RFC 1939-ben. A POP3 a TCP szállítási protokollt használja, és a POP3 szerver a szabványos 110-es TCP porton érhető el.



– Az ábra bemutatja, hogy két asztali gépet használó felhasználó hogyan tud egymással levelezni.
Természetesen általában egy felhasználói levelzõ program mind az UA, mind a POP3 kliens feladatát képes ellátni.

6. ábra. Felhasználók közti levelezés

Bár az üzenetek letöltéséhez a POP3-at használjuk, a munkaállomás felhasználója továbbra is az SMTP levél szervert használja az üzenetek elküldéséhez.

A 7., 8. és 9. táblázatban POP3 parancsok láthatók, melyek az RFC 1939-en alapulnak. Igaz a USER és a PASS parancsok opcionális parancsok az RFC 1939-ben, de általában támogatottak. A USER/PASS parancsok azért opcionálisak, mert van egy másik lehetőség: az APOP parancs, amely az MD5 (Message Digest version 5) hitelesítő eljárás használja.

Parancs	Jelentés
STAT	Erre a parancsra kapunk egy pozitív visszajelzést, amely áll egy +OK string-ből majd egy betűköz után az üzenetek száma és még egy szóköz után a levelek össz-mérete látható oktetekben.
LIST [üzenet_sorszám]	Ha megadunk üzenet sorszámot, akkor a parancs hatására a szerver válaszképpen kiírja a kért üzenet azonosító mellé a méretét oktetekben. Ha nem adunk meg sorszámot, akkor egy többsoros válasz kapunk, melynek első sora egy +OK string-ből majd egy betűköz után az üzenetek száma és még egy betűköz után a levelek mérete látható oktetekben. A következő sorok első karaktere az üzenet sorszám és a betűköz után hozzá tartozó méret látható.
RETR <üzenet_sorszám>	Ez a parancs arra szolgál, hogy letöltsük a POP3 szerveren lévő üzenetünket. Válaszként ad egy +OK string-et, majd a letöltött üzenet méretét oktetekben. Ezután következik maga az üzenet (az, amelyiknek a sorszámát megadtuk). Ha olyan azonosítót adunk meg amely nem létezik egy -ERR üzenetet küld a szerver.
DELE <üzenet_sorszám>	A parancs az adott üzenetet törölésre kijelöli.
NOOP	Ez a parancs nem hajt végre műveletet. A vevő egy +OK választ ad. A parancs akkor hasznos, ha meg akarunk győződni róla, hogy a kapcsolat rendben van-e és a POP3 szerver működik-e.
RSET	Ez a parancs törli az összes törlésre való kijelölést. A szerver visszaad egy +OK string-et.
QUIT	A parancs hatására a szerver törli az összes törlésre kijelölt levelet, és az elvégzett művelettől függően +OK vagy -ERR string-et ad vissza, majd lezárja a kizárólagos elérést a mail-drop-on és lebontja a TCP kapcsolatot.

7. táblázat. Szükséges POP3 parancsok

Parancs	Jelentés
USER <név>	Ezzel a paranccsal adható meg a kezelni kívánt postaláda.
PASS <string>	Ez a parancs adja meg a szerver/postaláda-specifikus jelszót a felhasználóhoz.
TOP <üzenet_sorszám> <n>	A parancs hatására a szerver válaszol egy +OK string-et, majd kiírja az üzenet fejrészét és egy üres sor után az üzenet törzsből *n* sort. Ha ez a szám nagyobb mint ahány sor van az üzenetben akkor a szerver elküldi az összes sort.
UIDL [üzenet_sorszám]	Minden üzenet kap egy egyedi azonosítót, amely alapján bárhol azonosítható lesz a levél. Ez a parancs az UID azonosító alapján listázza ki az üzeneteket.

Parancs	Jelentés
APOP <név> <digest>	A *név* azonosítja a felhasználót, a *digest* pedig egy MD5-ös (Message Digest version 5) digit string. Ezt a parancsot a normális egyszerű szöveg string-eket használó normális USER/PASS azonosítási metódus helyett szokták használni. A legfontosabb tulajdonsága, hogy a jelszót nem *nyílt* szövegként küldi el.

8. táblázat. Opcionális POP3 parancsok

Parancs	Jelentés
+OK	A parancs rendben lefutott
-ERR	Hiba történt a parancs teljesítése közben

9. táblázat. A POP3 szerver válaszai

A 10. táblázatban látható egy példa a POP3 szerver és a POP3 kliens közötti kapcsolatra. A kapcsolat használ néhány parancsot a 7. és 9. táblázatban felsoroltak közül. (A K jelöli a klienst, a SZ pedig a szervert.)

A példában látható, hogy a POP3 kapcsolat kezdeti részében belépünk egy *kapcsolódási állapotba.* A kapcsolódási állapotban pedig létrehozzuk a TCP kapcsolatot a POP3 szerverrel. A következő lépésben a POP3 kapcsolat belép a *hitelesítési állapotba*. Ebben az állapotban a felhasználónak meg kell adnia a felhasználói nevét illetve a jelszavát, hogy azt hitelesíthesse a szerver. A korábbi POP3 implementációkban a felhasználói név és jelszó-hitelesítési információk nyílt szövegként lettek továbbítva, ami azt jelenti, hogy ha valaki kibontotta a TCP csomagokat kiolvashatta belőlük a felhasználói név/jelszó kombinációkat. Biztonságosabb alternatívát nyújt az RFC 1939-ben leírt MD5-ös hitelesítési folyamat.

Miután a felhasználót hitelesítette a szerver a POP3 kapcsolat átlép a *tranzakciós állapotba*. Ebben az állapotban számos parancs kiadására van lehetőségünk - mint például a STAT, LIST, RETR, DELE, RSET és így tovább. A 10. táblázatban a POP3 kliens kiad egy STAT parancsot, amire válaszul megkapta, hogy 8 üzenete érkezett és hogy azoknak mekkora a méretük (153681 oktet). A POP3 kliens ezután a LIST parancsot használta, hogy lekérdezze az üzenetek listáját. A szerver válaszképpen megadta az üzenet azonosító száma mellé az adott üzenet méretét is. Ezután a kliens a RETR parancsot használta az üzenet azonosítóval, hogy letöltse a levelét. A POP3 kliens beállításaitól függően a kliens letörölheti a letöltött üzenetet a szerverről.

Miután az üzeneteket letöltöttük a POP3 kapcsolat átléphet a *frissítési állapotba*. Ebben az állapotban a felhasználó kiadhatja a QUIT parancsot, hogy bezárja a kapcsolatot. Mind a POP3 kliens, mind a POP3 szerver frissíti a belső állapotait, hogy rögzítse, mennyi üzenet van a postaládáikban. Végül a TCP kapcsolat bezárul.

SZ:	(kapcsolatra vár a 110-es TCP porton)	**Kapcsolódási állapot**
K:	(megnyitja a kapcsolatot)	
SZ:	+OK POP3 tilb.sze.hu v2000.70 server ready	
K:	USER <*felhasználó_név*>	**Hitelesítési állapot**
SZ:	+OK User name accepted, password please	
K:	PASS <*jelszó*>	
SZ:	+OK Mailbox open, 8 messages	
K:	STAT	**Tranzakciós állapot**
SZ:	+OK 8 153681	
K:	LIST	
SZ:	+OK Mailbox scan listing follows	
SZ:	1 29486	⋆ STAT
SZ:	2 512	⋆ LIST
SZ:	3 64	⋆ RETR <sorszám>
	...	⋆ DELE <sorszám>
SZ:	8 65677	⋆ RSET
SZ:	<CR>.<LF>	
K:	RETR 1	
SZ:	+OK 29486 octets	
	itt van a levél tartalma	
SZ:	<CR>.<LF>	
K:	QUIT	**Frissítési állapot**
SZ:	+OK Sayonara	
K:	(bezárja a kapcsolatot)	
SZ:	(vár a következő kapcsolatra)	

K=Kliens SZ=Szerver

10. táblázat. POP3 szerver és a POP3 kliens közötti kapcsolat

2.3. Internet Message Access Protocol Rev 4 (IMAP4)

A POP3 egy jó kliens/szerver protokoll a munkaállomásoknak a leveleik letöltéséhez, azonban néhány esetben ez kevés lehet. Például POP3-on keresztül nem vizsgálhatjuk meg a leveleinket mielőtt letöltöttük volna azokat, és a POP3 nem teszi lehetővé a levelekkel (azok részeivel) való közvetlen műveletvégzést a szerveren. Tehát, ha ilyen feladatokat szeretnénk megoldani az IMAP4-et ajánljuk a POP3 helyett.

Az IMAP4 egy kliens/szerver protokoll, mellyel elérhetjük és szerkeszthetjük elektronikus leveleinket a szerveren. A protokoll lehetővé teszi, hogy a távoli postafiókon elvégezhessük ugyanazokat a műveleteket, amellyeket a helyi postaládánkon el tudunk végezni. Az IMAP4 továbbá támogatja a kapcsolat nélküli munkát, és lehetőséget biztosít az újraszinkronizálásra a szerverrel kapcsolatfelvétel esetén.

Egy IMAP4 kliens szolgáltatásai lehetővé teszik:

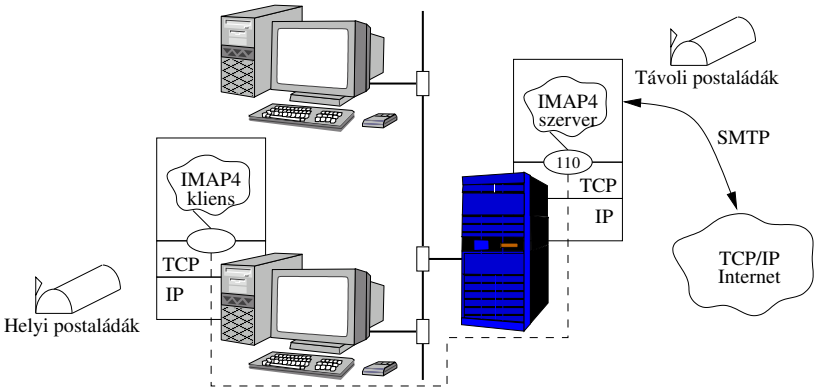
- Az e-mail-ek elérése és szerkesztési lehetősége a szerveren anélkül, hogy letöltenénk őket
- Levelek és csatolt fájlok áttekintése anélkül, hogy letöltenénk őket
- Levelek letöltése kapcsolat nélküli munkához
- Szinkronizálás a helyi és a szerveren lévő postaládák között

Az IMAP4 műveletei között szerepelnek:

- Postaládák létrehozása, átnevezése, törlése
- Új levelek érkezésének ellenőrzése
- Levelek eltávolítása a postaládából
- Levelek állapotjelzőinek beállítása, és törlése
- RFC-822 fejrész felismerése és MIME kódolású levelek elemzése („érti” a MIME kódolást)
- Keresés és szelektív letöltés az üzenet tulajdonságaiból, szövegéből illetve annak részeiből

Az üzenetek eléréshez az IMAP4-ben számokat használunk. Ezek a számok vagy az üzenetek sorszámai vagy az egyedi azonosítói. Az IMAP4 csak egy szerver elérését támogatja. A több IMAP4 szerver elérését lehetővé tevő konfigurációt az IETF-ben fontolgatják.

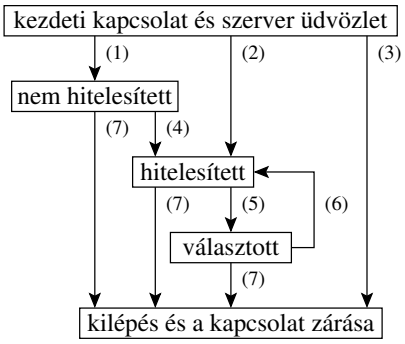
Ugyanúgy mint a POP3 az IMAP4 sem képes a levelek feladására. Ezt a funkciót általában egy levél átviteli protokoll látja el, ilyen például az SMTP. Az 7. ábra egy IMAP4-es kliens szerver kapcsolatot mutat.



– Az üzenet átviteli ügynök (MTA) egy erősebb számítógépen fut.
– Az IMAP4 dinamikus hozzáférést biztosít a postafiók szerverhez.

7. ábra. IMAP4 kliens/szerver architektúra

Az IMAP4 viselkedését írja le a 8. ábra egy állapot diagrammban.



8. ábra. IMAP4 állapotdiagram

1. kapcsolat előzetes azonosítás nélkül (OK üdvözlés)
2. előzetesen azonosított kapcsolat (PREAUTH üdvözlés)
3. visszautasított kapcsolat (BYE üdvözlés)
4. sikeres LOGIN vagy AUTHENTICATE (azonosítás) parancs
5. sikeres SELECT (választás) vagy EXAMINE (vizsgálat) parancs
6. CLOSE (bezárás) parancs, vagy sikertelen SELECT illetve EXAMINE parancs
7. LOGOUT parancs, szerver kikapcsolása vagy kapcsolat bontása.

Az IMAP4 az 8. ábrán négy állapot valamelyikében található. A legtöbb IMAP4 parancs csak bizonyos állapotban adható ki. Ha a kliens nem megfelelő állapotban adja ki a parancsot, akkor protokoll hiba generálódik.

A következő IMAP4 állapotokat definiáljuk ebben a részben:

- Azonosítás előtti állapot
- Azonosított állapot
- Választott állapot
- Kijelentkezett állapot

Az *azonosítás előtti állapot*ban az IMAP4 kliens azonosítási „okmányokat” nyújt be. A legtöbb parancs nem használható, míg a felhasználó nem azonosította magát. A kapcsolat kezdetén ebbe az állapotba kerülünk, hacsak nem történt előzetes azonosítás (preauthentication).

Az *azonosított állapot*ban a felhasználó már hitelesített, de mielőtt kiadná a parancsokat ki kell választania egy postaládát. Ebbe az állapotba lépünk, ha az előzetesen azonosított kapcsolat kezdődik vagy ha a kliens elfogadható azonosítási okmányokat nyújtott be. Amennyiben hiba történik a postaláda kiválasztásánál újra bekerülünk ebbe az állapotba, hogy egy másik postaládát választhassunk ki.

A *választott állapot*ban vagyunk, ha sikeresen kiválasztottuk a kezelni kívánt postaládát.

*Kijelentkezett állapot*ban, a kapcsolat befejeződött és az IMAP4 szerver is bezárja a kapcsolatot. Ebbe az állapotba a kliens kérése, vagy a szerver döntése nyomán kerülhetünk.

A 11. táblázatban láthatók az IMAP4-gyel kapcsolatos RFC-k.

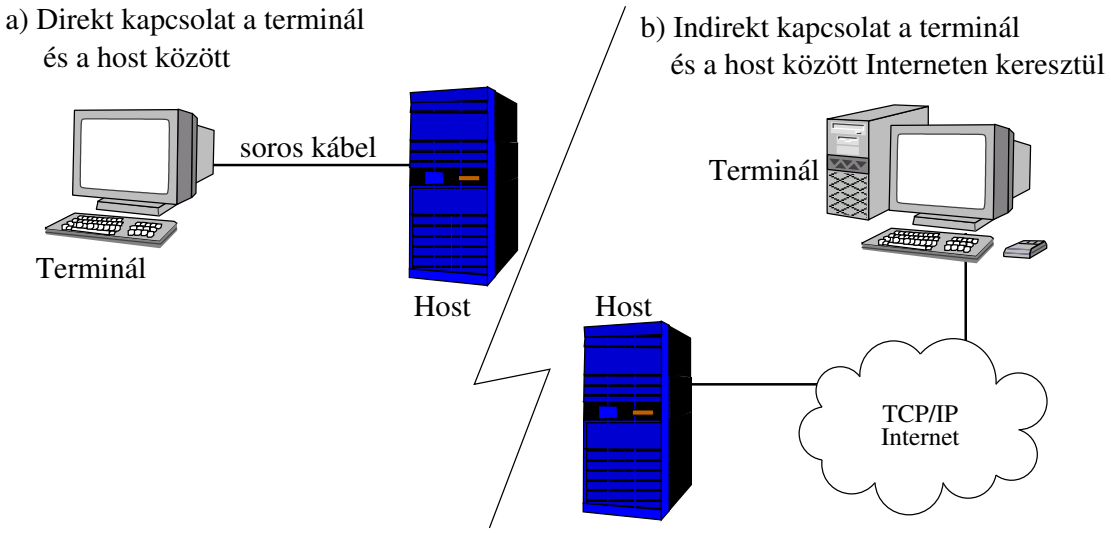
RFC#	Állapot	RFC cím
2095	A	IMAP/POP AUTHorize Extension for Simple Challange/Response
2088	A	IMAP4 nonsynchronizing literals
2087	A	IMAP4 QUOTA extension
2086	A	IMAP4 ACL extension
2061	I	IMAP4 COMPTIBILITY WITH IMAP2BIS
2060	A	INTERNET MESSAGE ACCESS PROTOCOL VERSION 4rev1
1733	I	DISTRIBUTED ELECTRONIC MAIL MODELS IN IMAP4
1732	I	IMAP4 COMPATIBILITY WITH IMAP2 AND IMAP2BIS
1731	A	IMAP4 authentication mechanisms

I=Információ A=Ajánlás (Proposed Standard)

11. táblázat. IMAP4 RFC-k

3. Távoli elérési protokollok

A nagy számítógépekre (main-frame) a felhasználók terminálról szoktak bejelentkezni. A terminál jellemzői annak hardverétől és a host-on futóoperációs rendszer által definiált terminálmodelltől függnek. Miután a kapcsolat létrejött a terminál és a host között, a felhasználó karaktersorozatokban parancsokat küldhet a host felé. A host-on futó terminál illesztő veszi és puffer-eli a karaktersorozatokat, összeállítja belőlük a felhasználói parancsokat. Ezeket átadja a hostnak, amely végrehajtja a user által kért műveleteket és visszaküldi az eredményt.



9. ábra. Terminál/Host architektúra

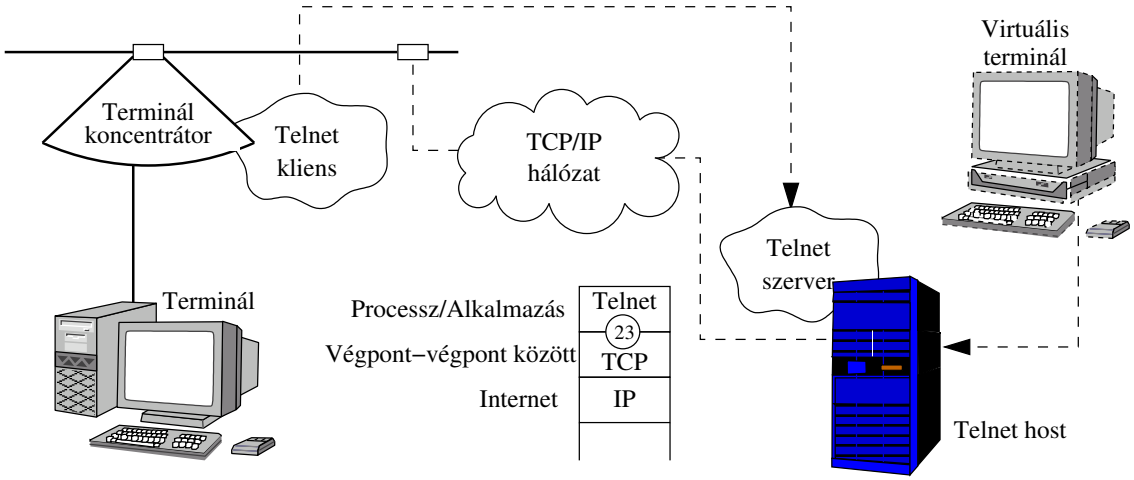
Ha a terminál hálózaton keresztül kapcsolódik a host-hoz, akkor szükség van egy távoli elérési protokollra, amely biztosítja azokat a szolgáltatásokat melyek közvetlen kapcsolat esetén elérhetőek (9. ábra). Az elsődleges távoli elérési protokollok a TCP/IP protokoll családban a következők:

- Telnet
- Berkeley r* segédprogramok

3.1. Telnet

A Telnet protokollt arra használjuk, hogy emulálja egy terminál kapcsolódását a host-hoz. TCP protokollt használ az információátvitelre a terminál billentyűzete és a host között, válaszirányban a terminál kijelzőjén jelennek meg a host üzenetei.

A 10. ábra egy Telnet kapcsolatot mutat. Ahhoz, hogy működhessen egy Telnet kapcsolat a felhasználó munkaállomásán egy Telnet kliensnek, a távoli host-on egy Telnet szervernek kell futnia. A Telnet kliens és a szerver között TCP kapcsolat van. A Telnet szerver a 23-as TCP porton várja kapcsolat felépülését. Unix rendszerekben a Telnet szervert *telnetd-nek* hívják (Telnet daemon, ejtsd: telnet-dí).



10. ábra. Példa egy Telnet kapcsolatra

A felhasználó által begépelt parancsokat veszi a szerver Telnet komponense, és elküldi a szerveren futó operációs rendszer felé, ami úgy hajtja azokat végre, mintha egy helyileg kapcsolt terminál adta volna ki. A végrehajtott parancs végeredményét a Telnet szerver visszaküldi a Telnet kliensnek. A Telnet kliens a szervertől kapott eredményeket kiírja a felhasználó munkaállomásának képernyőjére. Csak bejegyzett felhasználók jelentkezhetnek be a szerverre. A bejelentkezés után az, hogy milyen parancsokat adhatunk ki, kizárólag a szervergépen futó operációs rendszertől függ.

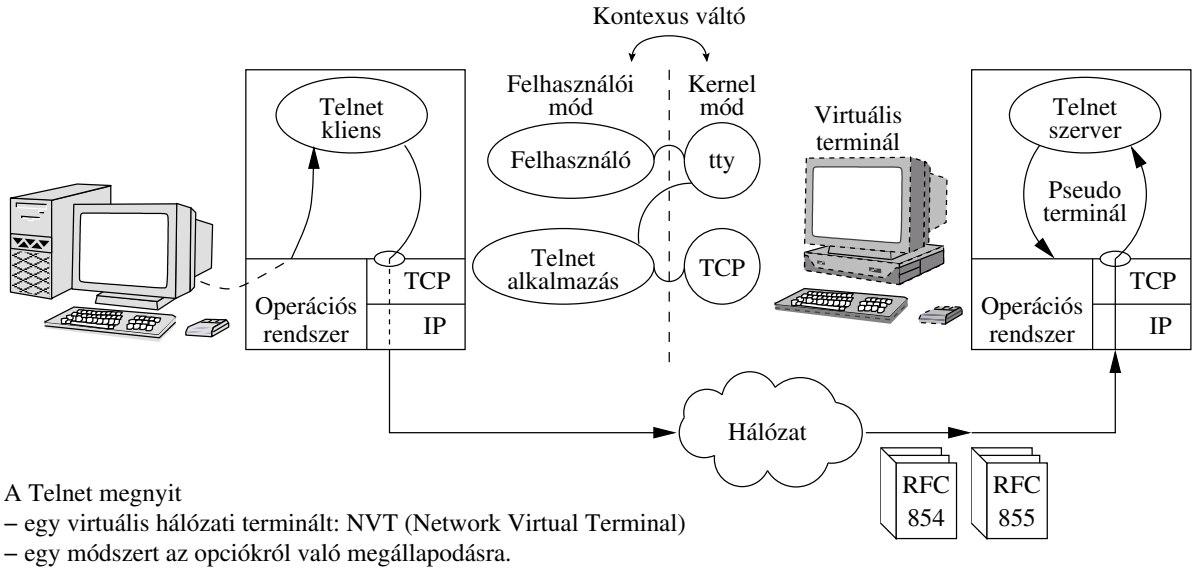
A következőkben látható egy példa egy Telnet kapcsolatra egy Unix host-tal:

```
$ telnet tai2.szif.hu
Trying 193.224.130.140...
Connected to tai2.szif.hu.
Escape character is '^]'.

FreeBSD/i386 (tai2.szif.hu) (ttyp0)

login: kismodit
Password:
Last login: Thu Oct 31 12:17:09 from ede.tilb.sze.hu
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
        The Regents of the University of California.  All rights reserved.

FreeBSD 4.1-RELEASE (GENERIC) #0: Fri Jul 28 14:30:31 GMT 2000
You have mail.
$ ls -la
drwxr-xr-x    4 jok      users        4096 Sep 27 09:45 .xmms
drwxr-xr-x    2 jok      users        4096 Sep 26 13:59 .xvpics
drwx------    3 jok      users        4096 Oct 31 13:33 Desktop
```

```
-rw-r--r--    1 jok      users       43607 Oct 31 13:31 Erich.pdf
drwxr-xr-x    2 jok      users        4096 Sep 27 09:58 mp3
drwxr-xr-x    3 jok      users        4096 Oct 30 18:02 psz
-rw-r--r--    1 jok      users      224168 Oct  2 08:02 wallpaper.jpg
...
$ ping tilb.sze.hu
PING tilb.sze.hu (193.224.128.28): 56 data bytes
64 bytes from 193.224.128.28: icmp_seq=0 ttl=253 time=1.064 ms
64 bytes from 193.224.128.28: icmp_seq=1 ttl=253 time=0.883 ms
64 bytes from 193.224.128.28: icmp_seq=2 ttl=253 time=0.808 ms
^C
--- tilb.sze.hu ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.808/0.918/1.064/0.107 ms
$ logout
Connection closed by foreign host.
```

3.1.1. Telnet architektúra

A 11. ábra egy Telnet kliens/szerver modellt mutat. Amikor a felhasználó elindít egy Telnet alkalmazást és megadja a célállomást egy TCP kapcsolat épül fel a célállomás 23-as portján keresztül. Az adatok a Telnet kliens és a Telnet host között (korábban szerverként említve) TCP kapcsolaton keresztül áramolnak. Rendes körülmények között a TCP kapcsolat akkor bomlik le, amikor a felhasználó kiadja a kijelentkezés parancsot, hogy bezárja a Telnet kapcsolatot.



11. ábra. Telnet kliens/szerver modell

Miután a TCP kapcsolat létrejött a Telnet kliens és a Telnet szerver megállapodnak a paraméterekben, melyek meghatározzák a terminál típusát és a működés módját. Példaképpen egy terminálnál beállítható, hogy csak akkor küldjön el egy begépelt sort, ha azt befejeztük, pedig az alapbeállításban a karakterenkénti küldés van megadva.

Ha a felhasználó gépelni kezd, akkor azt átveszi a terminál meghajtóprogramja, és átadja a Telnet kliensnek. A Telnet kliens általában minden billentyűleütést külön TCP szegmensben küld el. Ez elég nagy pazarlás, de egy gyors hálózat számára nem okoz problémát, ha csak néhány Telnet felhasználó van. A Telnet szerver válaszai több karakterből álló üzenetek lehetnek, így sokkal jobban használja ki a TCP protokoll adta lehetőségeket, mint a Telnet kliens.

A Telnet szerver sok implementációban együtt indul az Internet daemon (szuper daemon) szerverrel, amely egy konfigurációs fájlban megadott listán szereplő portokat figyel. Ha kapcsolat-felépítési szándékot észlel a TCP 23-as portján (amely általában a Telnet szerveré), akkor Telnet szerver rákapcsolódik arra. Gyakran szokott egymás mellett több Telnet szerver is futni, hogy nagy számú Telnet klienst tudjon kezelni egy host. Egy másik megoldás, hogy egy Telnet szerver külön szálat használ az egyes Telnet kapcsolatokhoz.

A 11. ábrán egy felhasználói módban futó Telnet kliens és szerver látható. A modern operációs rendszerek két módban képesek programokat futtatni: *felhasználói* és *kernel* módban. A legtöbb alkalmazás felhasználói módban fut, amely korlátozott elérést nyújt a számítógép hardvere felé. Ez a mód korlátozza az olyan alkalmazások használatát melyek a rendszer összeomlását okozhatják. Az operációs rendszer szolgáltatásai, a protokollok, az eszközök kezelői általában csak kernel módban futnak. Továbbá kernel módban a programok korlátlan hozzáférést kapnak a számítógép hardveréhez.

Ha a Telnet kliens és a szerver is felhasználói módban fut és el szeretnénk érni valamilyen operációs rendszer szolgáltatást, mint például a TCP/IP protokollt, egy *kontextus váltás*t kell végrehajtanunk, hogy átkapcsoljunk a kernel módba, ahol elérhető a TCP/IP protokoll.

Amikor egy karakter megérkezett a termináltól, a kontextus váltás történik felhasználói módból kernel módba, hogy elérje az operációs rendszer teletype (tty) meghajtóprogramját, amely kernel módban fut. A tty meghajtóprogram átadja ezt a karaktert a Telnet kliensnek, amely felhasználói módban fut. Így tehát a kontextus váltás történik kernel módból a felhasználói módba. Amikor a Telnet kliens elküld egy karakter adatot a TCP-nek, egy másik kontextus váltás történik, mivel a TCP kernel módban fut. Ebben a példában minden a Telnet kliens gépről küldött karakter esetén három kontextus váltás történik.

A szerver oldalon a kontextus váltás történik, amikor a TCP átadja a beérkezett karaktert a Telnet szerver programnak. A Telnet szerver elküldi az összes karakter adatot az operációs rendszernek egy pseudoteletype-on (ptty) keresztül, ami több kontexus váltást is tartalmazhat. Amennyiben nagy számú karakter érkezik sok kontextus váltásra van szükség ami a gép lelassulását okozhatja. Ha más alkalmazások futnak a Telnet host-on, a Telnet által okozott terhelés azok futására is hatással van

A Telnet szabvány (más néven STD 8) megtalálható a következő RFC-kben:

- RFC 854 on „Telnet Protocol Specification”
- RFC 855 on „Telnet Option Specification”

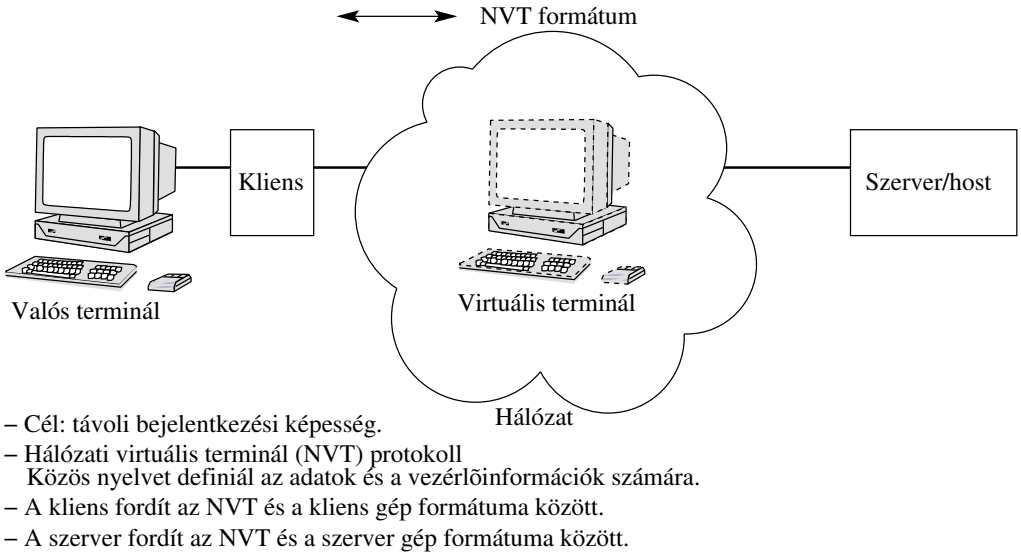
3.1.2. Az NVT Terminál és formátuma

A terminál hardverek és a Telnet szerver képességek széles skálájához való alkalmazkodás érdekében lehetőséget nyújt a paraméterekről való megállapodásra. Ezeket a paramétereket definiálja a hálózati virtuális terminál (Network Virtula Terminal, NVT) eszköz, amely a 12. ábrán látható.

Az NVT eszköz és az NVT protokoll meghatároz egy eljárást a vezérlő és adatinformációk átviteléhez. A különbségeket a terminál hardverek, a vezérléseik és az adat információk között az NVT eszköz modell és a NVT protokoll kezeli.

A Telnet kliens lefordítja a felhasználó billentyűleütéseit egy karaktersorozattá NVT formátumban. Az NVT formátum kezeli a felhasználói adat és a vezérlő adat kódolását. A *felhasználói adat* tartalmazhat betűket és számjegyeket melyeket a felhasználó begépel. A *vezérlő adat* tartalmazhat olyan karaktereket, amelyek megállítanak egy futó pocess-t (CTRL-C sok rendszerben), törlés karaktert (backspace), sor törlést és így tovább. A Telnet szerver elfogadja a NVT adat formátumot és lefordítja a helyi operációs rendszer karakterformátumára. A választ a Telnet szerver NVT adatsorozatként küldi el. A választ fogadva a Telnet kliens olyan formátumba konvertálja, amilyent használ. Összegzés képpen:

- A Telnet kliens fordít az NVT formátum és a kliens gép formátuma között.
- A Telnet szerver fordít az NVT formátum és a szerver gép formátuma között.



12. ábra. Hálózati virtuális terminál

Az NVT formátum hét bites US ASCII kódot használ és fenntartja a legmagasabb helyértékű bitet a karakterben a parancs szekvenciának. A US ASCII karakterkészlet 95 megjeleníthető karaktert tartalmaz: betűket, számokat, központozási jeleket, és 33 vezérlő kódot. A 12. táblázatban látható néhány szabványos vezérlő karakter.

Név	Kód	Jelentés
NUL	0	nincs művelet
BEL	7	hang/kép jelzés
BS	8	mozgás egy pozícióval balra
HT	9	mozgás a következő vízszintes tabulátorra
LF	10	mozgás (függőlegesen) a következő sorra
V	T11	mozgás a következő függőleges tabulátorra
FF	12	mozgás a következő oldal tetejére
CR	13	mozgás az adott sor elejére

12. táblázat. Szabványos NVT vezérlő karakterek

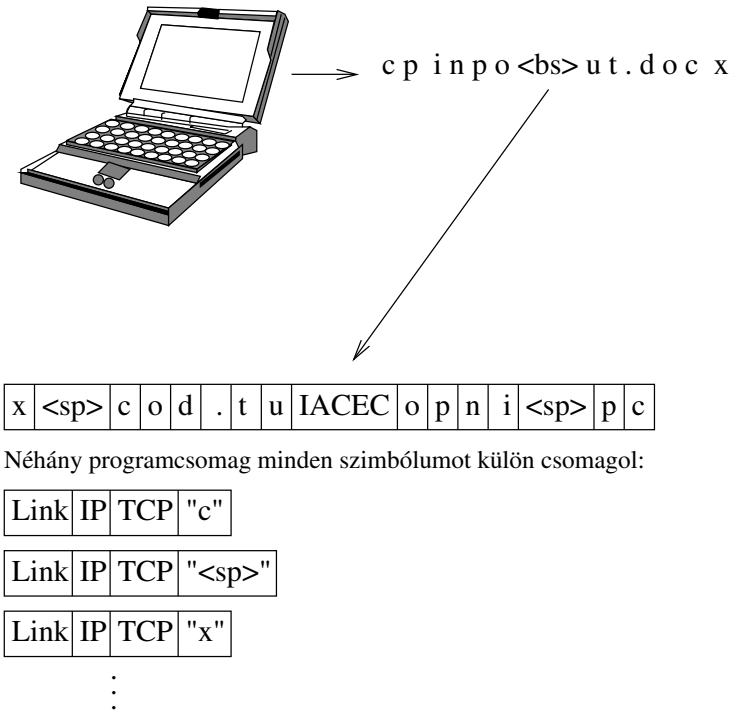
A 13. ábra példa az NVT formátumra. A felhasználónak a parancshoz a következő karakter szekvenciát kell begépelnie:

```
cp inpo<bs>ut.doc x
```

A <*bs*> jelenti a backspace karaktert ami ki fogja törölni az előzőleg hibásan begépelt *o* betűt. A Telnet kliens elküldte a NVT karaktereket US ASCII megjelenítést használva, minden karakter külön TCP szegmensben. Ha megfigyeljük a <*bs*> karakter a következő képpen nézett ki az 13. ábrában:

```
IACEC
```

Az IAC jelentése interpret-as-command (értelmezd mint parancsot) karakter, aminek a kódértéke 255. Az EC karakter a törlés, aminek a kódértéke 247.



13. ábra. Példa az NVT formátumra

3.1.3. Telnet beállítások meghatározása

Miután a Telnet kapcsolat létrejött TCP protokollon keresztül mindkét oldalon - Telnet kliens és szerver - meghatározzák azokat a beállításokat, melyeket használni fognak. Ezek az opciók például a következők lehetnek: terminál típus, fél-duplex vagy duplex mód, sor mód, karakter mód, stb..

A Telnet kliens speciális Telnet kódokat küld a beállítás megadására, mint DO, WILL, DON'T, WON'T. A szerver szintén hasonló kódokkal válaszol, mint WON'T, DON'T, WILL és a DO. A kódok, mint karakter lesznek elküldve és mindegyik előtt ott kell legyen a 255 (IAC) parancsot jelző kód. Például a 253-as kód le van foglalva a DO parancs számára, és 252-es pedig WON'T parancs számára, így tehát:

255 253 = IAC DO = DO beállítás indikátor

255 252 = IAC WON'T = WON'T beállítás indikátor

A 13. táblázatban látható a Telnet parancsok hivatkozáslistája, és a 14. táblázatban kódértékek néhány Telnet beállításhoz.

Kód		Jelentés
IAC(255)		a következő oktet parancsot tartalamaz
IAC(255)	DON'T(254)	a küldő a vevő kéri, hogy a vevő törölje a beállítást
IAC(255)	DO(253)	a küldő a vevő kéri, hogy a vevő engedélyezze a beállítást
IAC(255)	WON'T(252)	a küldő törölni akarja a beállítást
IAC(255)	WILL(251)	a küldő engedélyezni akarja a beállítást
IAC(255)	GA(249)	„kezdje el/folytassa adását" jelzés ("Go Ahead")
IAC(255)	EL(248)	„sor törlés" jelzés ("Erase Line")
IAC(255)	EC(247)	„karakter törlés" jelzés ("Erase Character")
IAC(255)	AYT(246)	„ott vagy?" jelzés ("Are You There?")
IAC(255)	AO(245)	„kimenet megszakítása" jelzés ("Abort Output")
IAC(255)	IP(244)	„process megszakítása" jelzés ("Interrupt Process")
IAC(255)	BRK(243)	„Break" jelzés
IAC(255)	NOP(241)	„nincs művelet" jelzés ("No Operation")
IAC(255)	SB(250)	Adott opciókban való megegyezés kezdete
IAC(255)	SE(241)	Adott opciókban való megegyezés vége

13. táblázat. Telnet parancsok hivatkozáslistája

Név	Kód	RFC#	Jelentés
Echo	1	857	A vevő engedélyezi a visszhang adatot
SGA	3	858	Megszünteti a „kezdje el/folytassa adását" jelzés küldését az adat végén
Status	5	859	TELNET beállítás állapotának lekérdezése a távoli oldalról
Timing Mark	6	860	Időzítő jelzés elhelyezésének kérése a visszatérő adatfolyamba (szinkronizálás céljából)
Terminal Type	24	884	Információcsere a terminál típusairól
NAWS	31	1073	Ablakméret beállítása
Tspeed	32	1079	Információ elküldése a Terminál sebességéről
TFC	33	1080	Terminál (távoli) Flow Control (adatáramlás vezérlése)
Linemode	34	1116	Teljes sorok küldése karakterek helyett
Xdisploc	35	1096	X képernyő helye

14. táblázat. Kódértékek néhány Telnet beállításhoz

3.2. Berkeley r* segédprogramok

A BSD Unix disztribúció elérhetővé tett számos parancsot, melyeket r parancsoknak vagy r* segédprogramoknak hívnak. Ezek a parancsok végrehajthatnak egy távoli gépen bejelentkezést, futtatást. A parancsokat arra tervezték, hogy olyan hálózatokban ahol a felhasználókban megbízhatunk a távoli gépekhez átlátszó (transzparens - ez alatt azt értjük, hogy nem kell magát felhasználói névvel és jelszóval azonosítani, a távoli gépen enélkül is ugyanúgy dolgozhat, mint a helyigépen) Ez a szituáció különlegesen igaz az egyetemek területén, mint például a Univesity of California, Berkeley, ahol a diákoknak és a tanároknak több számítógépre is van bejelentkezési jogosultságuk, és transzparens hozzáférésre van szükségük. Sok más platform - mint a Unix implementációk, VMS, MVS, DOS, NT és így tovább - átvette az r* segédprogramokat az operációs rendszerbe, így ezek a programok széles körben elérhetők. A 15. táblázatban látható néhány általában elterjedt r* parancs.

Parancs	Jelentés
rlogin	Hasonló mint a Telnet. Lehetővé teszi a felhasználónak, hogy bejelentkezzen egy távoli gépre.
rexec	Engedélyezi, hogy egy parancsot futtassunk egy távoli gépen
rsh	Engedélyezi, hogy elindítsunk egy shell-t (parancs feldolgozó) a távoli számítógépen, és futtassuk a megadott parancsot (amit a parancssorba beírunk).
rcp	Engedélyezi a másolást távoli rendszerek között, illetve a távoli és a helyi számítógép között.
rwho	Kiírja azoknak a felhasználóknak a listáját akik bejelentkeztek a hálózatba.
rwall	Üzenetet küld az összes felhasználónak a meghatározott gépen
ruptime	Kiírja a hálózatban lévő gépek terhelését, a felhasználók számát és hogy mióta vannak bekapcsolva a gépek.

15. táblázat. Néhány r* parancs

3.2.1. Az rlogin parancs

Az rlogin parancs nem használ olyan szintű beállítás meghatározást, mint a Telnet, így távoli gépnek vagy ismernie kell a terminál típusát vagy úgy kell bekonfigurálni, hogy elfogadja a felhasználó bejelentkezésekor kért termináltípust. Néhány rlogin implementáció továbbítja a helyi felhasználó környezeti beállításait a távoli gépre a bejelentkezéskor.

Az r* parancsok átlátszó hozzáférésre a *bizalom* segítségével valósul meg. Ahhoz, hogy a felhasználónak átlátszó hozzáférése legyen a host-hoz, a felhasználó számítógépének az IP címe meg kell legyen adva a távoli gép következő fájljai valamelyikében:

- **hosts.equiv fájl.** Unix rendszerekben ez a fájl a /etc könyvtárban található. Ez a fájl tartalmazza azokat a gépeket, amelyekben *megbízunk* (trusted host), ami azt jelenti, hogy a host-nak átlátszó hozzáférése lehet ahhoz a távoli géphez, amelyikben a rendszergazda bejegyezte őket a hosts.equiv fájlba.
- **.rhosts fájl.** A felhasználó alkalmazhatja ezt a fájlt arra, hogy engedélyezze más gépek felhasználóinak, hogy az adott gépen (amelyikben az engedélyt adja) az ő jogaival (a login neve alatt) dolgozhatnak. Ezt a fájlt a felhasználó saját home könyvtárában kell elhelyezni. Csak saját magának és csak olvasási joga lehet a fájlnak (Unix alatt).

Amellett, hogy megadjuk a host nevet vagy az IP címet a hosts.equiv és .rhosts fájlokban, azon felhasználókat is felsorol*hat*juk akik átlátszó hozzáférést kaphatnak a gépen. Ha a felhasználókat nem soroljuk fel, akkor az összes felhasználó átlátszó hozzáférést kap az előzőekben megadott gépről. Ezen fájlok használatának és tartalmának részletei különbözőek lehetnek más-más rendszernél, ezért ajánlott megnézni a host dokumentációját, hogy az adott gépen pontosan hogyan kell ezeket a parancsokat használni.

Unix rendszerekben az /etc/passwd fájlt (amely a regisztrált felhasználók listáját tartalmazza) is felhasználják annak ellenőrzésére, hogy a felhasználóknak tényleg van-e felhasználói neve az adott gépre.

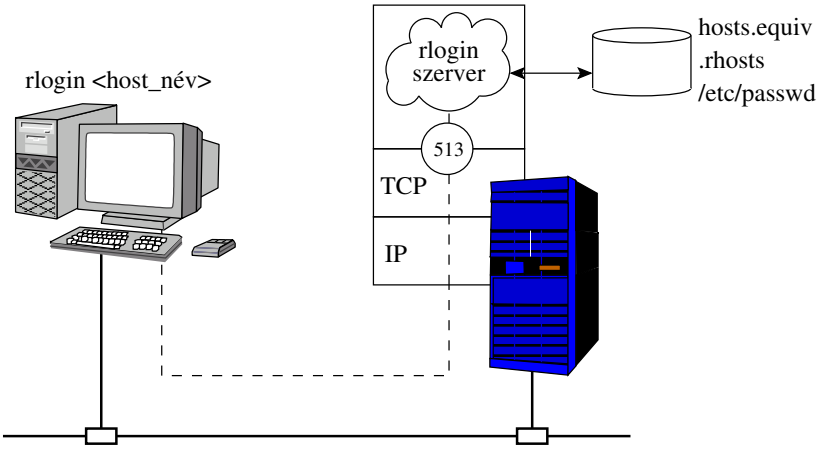
Általában amikor a felhasználó kiad egy parancsot, mint az rlogin, a következőképpen adja meg a távoli gép nevét:

```
rlogin <host_név>
```

Például:

```
$rlogin tilb.sze.hu
```

Itt nincs szükség a felhasználó névre és a jelszóra, mert a felhasználót hitelesíti az rlogin szerver (aminek a neve *rlogind*, ejtsd *rlogin-dí* a Unix rendszerekben). Az rlogin szerver a távoli gépen fut és és ellenőrzi a host.equiv, a .rhosts és (Unix rendszereknél) a /etc/passwd fájlokat (látható az 14. ábrán).



14. ábra. Példa az rlogin használatára

3.2.2. Az rsh segédprogram

Az rsh parancs segítségével egy távoli gépen lefuttathatunk egy parancsot, melynek eredménye visszatér hozzánk. Például, ha a `tilb.sze.hu` távoli gépen akarjuk a *ps -a* parancsot futtatni, és az eredményt visszakapni a következő parancsot kell kiadnunk:

```
rsh tilb.sze.hu ps -a
```

Az r* segédprogramok képesek kezelni mindkét gép környezetét, és értik a fájlrendszer fogalmakat, mint a *standard input* (alapértelmezett bemenet), *standard output* (alapértelmezett kimenet) és a *standard error* (alapértelmezett hiba), melyeket sok operációs rendszer támogat a Unix-on kívül is. Amennyiben az előbbi rsh parancs kimenetét egy helyi fájlban szeretnénk eltárolni a következő parancsot kell kiadnunk:

```
rsh tilb.sze.hu ps -a  > helyi_file
```

3.2.3. Az rcp segédprogram

Az rcp segédprogram lehetővé teszi a másolást távoli rendszerek között, illetve a távoli és a helyi számítógép között. A távoli fájlnevekre a következőképen hivatkozhatunk:

```
hostname:pathname
```

Ahol a *hostname* adja meg a host nevét (vagy IP címét), a fájl elérési útját pedig a *pathname*.
A következő parancs egy fájlt másol a távoli rama.kinetics.com rendszerről a helyi gépre:

```
rcp tilb.sze.hu:/etc/passwd passwd.helyi
```

A következő parancs egy fájlt másol a távoli `pc2.tilb.sze.hu` rendszerről a távoli `pc3.tilb.sze.hu` rendszerre:

```
rcp pc2.tilb.sze.hu:/etc/ftpusers  pc3.tilb.sze.hu:/ftpusers
```

Természetesen a parancs sikeres végrehajtásához korábban mindkét távoli gépen meg kellet tenni a szükséges bejegyzéseket a helyi gépre (ahol a parancsot kiadjuk) vonatkozólag!

3.2.4. A Berkeley r* segédprogramok a biztonság szempontjából

Az a probléma az átlátszó hozzáféréssel, hogy ha egy behatoló hozzáférést nyer egy kritikus fájlhoz, mint például ahhoz amely tartalmazza a megbízható gépek (trusted hosts) listáját, annak megváltoztatásával hozzáférést nyerhet a rendszerhez.

Néhány r* implementáció nem működik IP címek megadásával kizárólag DNS szimbolikus neveket fogad el. Ez a biztonsági megoldás szükségessé teszi, hogy minden gép regisztrálva legyen a DNS-ben. A behatoló ilyenkor csak úgy tud hozzáférést szerezni, ha regisztrálja magát a DNS-ben ami viszont lehetővé teszi a behatoló azonosítását.

Sok nem Unix r* implementációban az r parancs használata előtt a felhasználónak meg kell adnia egy helyi jeszót. Ezeket a jelszavakat azonban egy helyi fájlban tárolják, és ha a behatoló ellopja a fájlt kinyerheti belőle a távoli gép hozzáférésének jogát.

4. Fájl hozzáférési protokollok

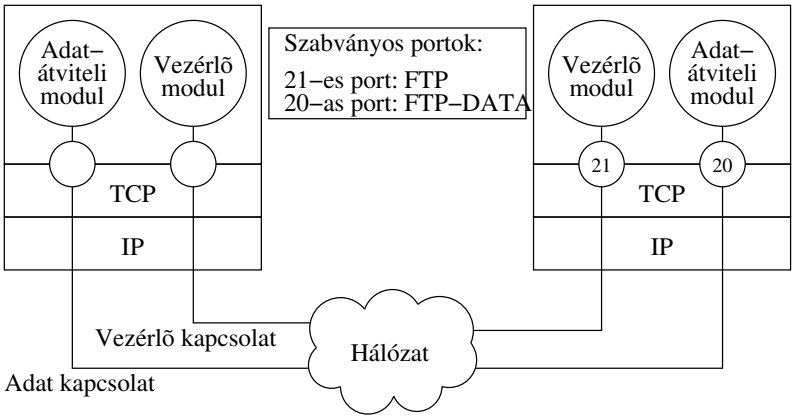
Az adatokat a főbb rendszerekben fájlnak (álomány) nevezett egységekben tárolják, melyeket fájlnak hívnak. A fájlnak egy rendszeren a következő jellemzői lehetnek: fájl név, létrehozás, az utolsó módosítás, a utolsó hozzáférés időbélyege, a fájl mérete és még további részletek az operációs rendszertől függően.

Az adatátvitel két rendszer között általában a fájlok vagy azok részeinek átvitelét jelenti. A TCP/IP feletti alkalmazások között a következő két protokoll szolgálja számítógépek között teljes fájlok átvitelét: a File Transfer Protocol (FTP) és a Trivial File Transfer Protocol (TFTP). (Ezt a feladatot végzi az `scp` is, csak az nem fért bele a tananyagba!)

A fájlokhoz való közvetlen hozzáférésre fájloknál, ahol egy távoli fájlrendszer a helyi fájlrendszerből részeként látszik, egy másik szabványos protokollt használnak, melynek a neve Network File System (NFS). Ezekről a protokollokról szólnak a következő fejezetek.

4.1. File Transfer Protocol (FTP)

Az FTP két rendszer között TCP-t használ a fájlok átvitelére IP hálózaton keresztül. Az FTP implementáció két komponenst tartalmaz: FTP kliens és az FTP szerver. Az 15. ábrán látható az FTP működési modellje.



Vezérlõ kapcsolat (control connection)
– akkor jön létre, amikor az FTP bejelentkezik az FTP szerverre és mindvégig fennáll
– csak a kliens és a szerver közti FTP parancsok és válaszok átvitelére használják
Adatátviteli kapcsolat
–minden egyes adatátvitelkor létrehozzák
–az adatátvitel végén lebontják
(Az adatátvitel nemcsak egy fájl átvitele lehet, hanem például egy könyvtár listázása is!)

15. ábra. FTP művelet modell

Az FTP kliens a felhasználó gépén fut és a kapcsolatot kezdeményez a szerverrel. Az FTP szerver a szabványos 21-es TCP porton várja a kapcsolat felépítési elvet. Amikor a kapcsolat kérés megérkezett, lejátszódik a háromutas kézfogás a TCP kapcsolat felépítéséhez. Ezt a TCP összeköttetést *vezérlő kapcsolatnak* hívják, az FTP parancsok és válaszok küldésére és a visszaigazolások fogadására használják. Amikor az FTP kliens kiad egy parancsot az adatátvitel megkezdésére, a kliens elindít egy adatátviteli folyamatot (process) egy helyi TCP porton. Ezt a helyi TCP port számot elküldi a szervernek a vezérlőkapcsolaton keresztül. Mikor az TFP szerver megkapja az FTP kliens adatátviteli process-ének port számát a szerver elindít egy adatátviteli process-t a szabványos 20-as TCP porton, amely kiad egy TCP kapcsolat felépítési kérést, hogy létrejöhessen az összekötetés az FTP kliens adatátviteli process-ével a kliens által korábban elküldött sorszámú porton keresztűl. Ezt az összeköttetést *adat kapcsolatnak* hívják, és kizárólag a kért fájl (könyvtár, lista) adatainak átvitelére használják. Az FTP engedélyezi, hogy le és feltölthessenek fájlokat. Ezek a fájlok mind az adatkapcsolaton keresztül továbbítódnak. Összegzés képpen a következő két összeköttetés jön létre egy FTP kapcsolat alkalmával:

- Vezérlő kapcsolat
  (TCP, FTP_KLIENS_IP, KLIENS_PORT1, FTP_SZERVER_IP, 21)
- Adat kapcsolat
  (TCP, FTP_KLIENS_IP, KLIENS_PORT2, FTP_SZERVER_IP, 20)

Az adatkapcsolat csak a fájl átvitelének idejére jön létre és bezáródik amint az átvitel befejeződött. Új kapcsolat jön létre minden alkalommal mikor egy fájlt átviszünk. Az FTP kapcsolat fenn marad amíg azt a kliens vagy a szerver be nem zárja. Általában a kliens szakítja meg a kapcsolatot, azonban túlterhelés vagy más probléma esetén a szerver is bezárhatja. Például ha adott ideig az FTP kliens nem fordul hozzá, akkor be szokta zárni.

Az FTP működési modellben látható, hogy két külön process fut egyszerre. Bizonyos esetekben nem jöhet létre két különálló process. Például, ha egy olyan korlátozott operációs rendszert használunk, amely nem képes több process-t egymás mellett futtatni vagy az FTP egy beágyazott rendszer amely operációs rendszer nélkül fut. Ezekben az esetekben egy egyszerű egy darabból álló program vagy process kezeli mindkét FTP komponenst. Tekintet nélkül arra, hogy az FTP kliens vagy szerver milyen megoldást használ az FTP protokoll mindenképp két kapcsolatot igényel. Megjegyzés: a fentiekkel ellentétben az elterjedt rendszerekben az FTP protokollban meghatározott két funkciónális komponenst a valóságban egyetlenprogram valósítja meg!

Az FTP szabvány megtalálható az RFC 959 (STD 9) „File Transfer Protocol” című részében. A következőkben az FTP kliens-szerver kommunikációs parancsaival ismerkedünk meg.

4.1.1. FTP parancsok

A parancsokat az FTP vezérlő kapcsolatán keresztül továbbítják a Telnet által használt adatformátumban. (Emlékeztetőül a Telnet NVT formátumot használ adattovábbításra.) Az FTP parancsok nem használják az NVT formátum minden lehetőségét, csak annak egy részhalmazát. (Például nem használja az option neogation-re használt NVT parancsokat)

Az FTP parancsok maximum négy karakteres string-ek, opcionálisan paraméterekkel kiegészítve. A parancsok és jelentésük a 16. táblázatban láthatók.

Vezérlő parancs	Jelentés
USER <felhasználói_név>	Megadja az FTP kapcsolatot kezdő felhasználó nevét.
PASS <jelszó>	Megadja a USER parancsban megadott felhasználóhoz tartozó jelszót.
ACCT <account>	Megadja a PASS parancs után a további account információkat. (A parancs opciónális.)
CWD <könyvtár>	Megváltoztatja az aktuális könyvtárat a szerveren.
CDUP	Átvált az aktuális könyvtárról annak szülő könyvtárára.
SMNT <elérési_út>	Mount-ol egy külső fájlrendszer adatstruktúrájából anélkül, hogy megváltoztatná a felhasználói nevet vagy jeszót.
REIN	Visszaállítja a kezdeti értékeket a felhasználó azonosítása előtti állapotba. A USER parancsnak kell követnie.
QUIT	Bezárja az FTP kapcsolatot.
PORT h1,h2,h3,h4,p1,p2	Megadja a TCP végpont információkat. Segítségével megadhatjuk az adat-process FTP kliens portjának számát az FTP szervernek. A h1-h4-ig az IP cím oktetjei decimálisan pontokkal elválasztva, p1 és a p2 az adat-process portjának száma decimálisan pontokkal elválasztva.

Vezérlő parancs	Jelentés
PASV	Megadja, hogy az FTP szerver várakozzon az adatkapcsolatra, amit a kliens fog létrehozni. Ha erre a parancsra adott válasz tartalmazza a TCP végpontot, ahol az FTP szerver figyel.
TYPE <kód>	Megadja az adatmegjelenítés típusát.
STRU <kód>	Megadja a fájl struktúráját. (Például: fájl, rekord, oldal.)
MODE <kód>	Megadja az átvitel módját. (Például: byte folyam (straem), blokk (block), tömörített (compressed).)
RETR <elérési_út>	Megadott fájl letöltése.
STOR <elérési_út>	Megadott fájl tárolása a szerveren (feltöltés).
STOU <elérési_út>	Hasonló mint a STOR, azonban az eredményfájlnak, melyet létrehoz egyedi neve kell hogy legyen a könyvtárban.
APPE <elérési_út>	Hasonló mint a STOR, azonban ha a megadott fájlnév már létezik az új adatot folyamatosan hozzáírja a régihez.
ALLO <argumentumok>	Helyet foglal a szerveren a fájlnak.
REST <argumentumok>	Meghatároz a szerveren egy jelző pontot, ahonnan a fájl átvitele folytatódhat.
RNFR <elérési_út>	Megadja a régi elérési utat ahhoz a fájlhoz amely át lesz nevezve. Az RNTO parancsnak kell követnie.
RNTO <elérési_út>	Megadja az elérést az új fájlhoz. Az RNFR parancs után kell használni.
ABOR	A szerver megszakítja az aktuális parancs végrehajtását.
DELE <elérési_út>	Megadott fájl törlése.
RMD <könyvtár>	Megadott könyvtár törlése.
MKD <könyvtár>	Megadott könyvtár létrehozása.
PWD	Kiírja az aktuális könyvtár elérési útját a szerveren.
LIST [könyvtár]	Kiírja a megadott könyvtárban lévő fájlok neveit, ha nincs megadva könyvtár, akkor az aktuális könyvtár fájljainak neveit írja ki.
NLST [könyvtár]	Kiírja a megadott könyvtárban lévő fájlok neveit attribútumok nélkül, ha nincs megadva könyvtár, akkor az aktuális könyvtár fájljainak neveit írja ki hasonlóképpen. (Az MGET parancs használja.)
SITE <paraméterek>	Megadja a hely-specifikus parancsokat a szervernek.
SYST	Megkapható a paranccsal , hogy milyen operációs rendszer fut a távoli gépen.
HELP [parancs]	Segítséget nyújt a parancs használatához, ha nem adunk meg parancsot kiírja az összes a szerveren használható parancsot.
NOOP	Nincs művelet, a szerver küld egy visszaigazolást.

16. táblázat. FTP parancsok

Az FTP szerver válasza egy három számjegyű kódból, illetve szóköz után opcionálisan szöveges üzenetből áll:

```
nnn [szöveges üzenet]
```

Az nnn a három-számjegyes kódot jelenti.

Példa képpen az FTP szerver válaszolhatja a következőt:

```
200 Command okay.
```

A 17. táblázatban található az FTP szerver néhány lehetséges válasza és azok jelentése.

Kód	Jelentés
200	Parancs rendben
500	Szintaktikus hiba, nem értelmezett parancs
501	Szintaktikus hiba a paraméterben
202	Parancsot nem implementálták
502	Parancsot nem implementálták
503	Rossz sorrendben megadott parancsok
504	Parancsot nem implementálták ezzel a paraméterrel
110	Restart marker reply
211	Rendszer állapot
212	Könyvtár állapot
213	Fájl állapot
214	Segítség üzenet
215	NAME rendszer típus
120	Üzemkész nnn perc múlva
220	Üzemkész egy új felhasználónak
221	Szolgáltatás bezárja a vezérlő kapcsolatot
421	Szolgáltatás nem elérhető, vezérlő kapcsolat zárása
125	Adat kapcsolat már nyitva, átvitel kezdődik
225	Adat kapcsolat megnyitva, nincs átvitel
425	Adat kapcsolatot nem tudja megnyitni
226	Adat kapcsolat bezárása
426	Kapcsolat bezárva, átvitel megszakadt
227	Passzív módba lépés (h1,h2,h3,h4,p1,p2)
230	Felhasználó bejelentkezve
530	Nincs bejelentkezve
331	Felhasználó neve rendben, jelszóra vár
332	Felhasználói név szükséges a bejelentkezéshez
532	Felhasználói név szükséges a fájl mentéséhez
150	Fájl állapot rendben adatkapcsolat megnyitása következik
250	Kért fájl művelet rendben lezajlott
257	„PATHNAME” Elérési út létrehozva
350	Kért fájl művelet további információra vár
450	Kért fájl művelet nem történt meg
451	Kért fájl művelet megszakítva, helyi hiba
452	Kért fájl művelet nem történt meg
550	Kért fájl művelet nem történt meg
551	Kért fájl művelet megszakítva, ismeretlen

Kód	Jelentés
552	Kért fájl művelet megszakítva
553	Kért fájl művelet nem történt meg

17. táblázat. Visszaigazolási kódok

4.1.2. FTP a felhasználó szemszögéből

Az FTP lehetővé teszi a felhasználó számára a távoli gépen történő interaktív fájl és könyvtár hozzáférést és a következő műveletek végrehajtását:

- A helyi vagy távoli gépen lévő könyvtárak listázását
- Fájlok átnevezése és törlése (jogosultság szükséges)
- Fájlok átvitele a távoli gépről a helyi gépre (letöltés)
- Fájlok átvitele a helyi gépről a távoli gépre (feltöltés)

Ahhoz, hogy FTP-t vagy bármi más TCP/IP feletti hálózati alkalmazást használjunk szükséges egy kliens verziójú alkalmazás. Ilyen TCP/IP kliens alkalmazások sok platformra elérhetők. Bizonyos platformok rendelkeznek grafikus felhasználói interfésszel (Graphical User Interface, GUI), míg mások parancssort használnak. Az oktatás szempontjából érdemesebb a parancssoros interfészt tárgyalni, mert könnyebb észrevenni az összefüggést az FTP felhasználói parancsai és az FTP belő parancsai között (16. táblázat).

Ahhoz, hogy megnyissunk egy FTP kapcsolatot a következő parancsot kell kiadnunk:

```
ftp [host_név]
```

A *host_név* szimbolikus neve vagy IP címe annak a host-nak amelyre be szeretnénk jelentkezni. Ha a host nevét nem adjuk meg, akkor csak néhány FTP parancs adható ki. Ha kiadjuk a HELP parancsot vagy parancsként a kérdőjelet (?) a gép kiírja az alkalmazható parancsok listáját. A következőkben látható egy parancslista melyet a `tilb.sze.hu` gépre bejelentkezve használhatunk:

!	debug	mdir	sendport	site
$	dir	mget	put	size
account	disconnect	mkdir	pwd	status
append	exit	mls	quit	struct
ascii	form	mode	quote	system
bell	get	modtime	recv	sunique
binary	glob	mput	reget	tenex
bye	hash	newer	rstatus	tick
case	help	nmap	rhelp	trace
cd	idle	nlist	rename	type
cdup	image	ntrans	reset	user
chmod	lcd	open	restart	umask
close	ls	prompt	rmdir	verbose
cr	macdef	passive	runique	?
delete	mdelete	proxy	send	

Ha tudjuk a távoli host IP címét, akkor azt alkalmazva így kell kiadnunk a parancsot:

```
ftp [IP_cím]
```

A parancs kiadása után a következő lépés az azonosítás, melyben meg kell adni a felhasználói nevet és a jelszót. Az FTP szerver az FTP host hitelesítő mechanizmusát veszi alapul a felhasználó jogainak meghatározásához. Tehát ha van egy `kajla` felhasználónév bejegyzésünk a távoli host-on, akkor bejelentkezhetünk az FTP szerverre mint `kajla` és használhatjuk a host-on megadott jelszavunkat.

Számos FTP host támogatja az *anonymous* (névtelen) bejelentkezést. Amennyiben anonymous-t adunk meg felhasználói névként a jelszavunk „guest" (vendég, de sok rendszerben az e-mail címünket kell megadni - néhány esetben csak a „@" karakter meglétét figyelik) és be tudunk lépni a host-ra a helyi adminisztrátor által korlátozott jogokkal.

A következő példa egy rövid FTP kapcsolatot kísér végig.

1. Kiadjuk az FTP parancsot és mellé a host nevét.

   ```
   % ftp ftp tilb.sze.hu
   ```

   A % jel a parancs előtt az alapbeállított Unix prompt.

2. Ha a host elérhető a következőhöz hasonló üzenetet kapunk. A bejelentkezés részletei különbözőek lehetnek.

   ```
   Connected to tilb.sze.hu.
   220 ProFTPD 1.2.5rc1 Server (Debian) [tilb.sze.hu]
   Name (tilb.sze.hu:root):
   ```

   A kapott üzenet után bejelentkezhetünk mint *anonymous* felhasználó.

3. Megadjuk a felhasználónevet és jelszót.

   ```
   Name (tilb.sze.hu:root): anonymous
   331 Anonymous login ok, send your complete email address as your password.
   Password:
   230-Udvozlet! Ez itt a Tavkozles-informatika laboratorium anonymous ftp
   szerver szolgaltatasa...
   230 Anonymous access granted, restrictions apply.
   Remote system type is UNIX.
   Using binary mode to transfer files.
   ftp>
   ```

4. A bejelenkezés után használhatjuk a `?` vagy a `help` parancsot:

   ```
   ftp>?
   Commands may be abbreviated.  Commands are:
   ```

   A parancsra kapott listát az előzőekben már láthattuk.

5. A `pwd` paranccsal megkaphatjuk, hogy a távoli host-on mi az aktuális könyvtár.

   ```
   ftp>pwd
   ftp> 257 "/" is current directory.
   ```

   Minden FTP parancs állapotát egy számkóddal jelzi vissza a host, mint az 257, melyet egy szöveges üzenet egészít ki.

6. Az aktuális könyvtár fájljainak listáját az `ls` vagy `dir` parancsokkal kaphatjuk meg.

   ```
   ftp> ls
   200 PORT command successful.
   150 Opening ASCII mode data connection for file list.
   drwxr-xr-x   2 1012     vip             8 May  3 06:37 1
   drwxr-xr-x   2 1012     vip             8 May  3 06:37 10
   drwxr-xr-x   2 1012     vip             8 May  3 06:37 2
   drwxr-xr-x   2 1012     vip             8 May  3 06:37 3
   ```

```
drwxr-xr-x   2 1012     vip             8 May  3 06:37 4
drwxr-xr-x   2 1012     vip             8 May  3 06:37 5
drwxr-xr-x   2 1012     vip             8 May  3 06:37 6
drwxr-xr-x   2 1012     vip             8 May  3 06:37 7
drwxr-xr-x   2 1012     vip             8 May  3 06:37 8
drwxr-xr-x   2 1012     vip             8 May  3 06:37 9
-rw-r--r--   1 1012     vip        342856 Sep 30  2003 letolt.gif
drwxr-xr-x   2 0        root           16 Nov  5  2003 protokollok
drwxr-xr-x   4 0        root           96 Sep 17  2003 pub
-rw-r--r--   1 0        root           92 Aug 28  2003 welcome.msg
226-Transfer complete.
226 Quotas off
```

Az információ megjelenítése Unix stítusú, mivel egy Unix rendszerre jelentkeztünk be.

7. Ha tudjuk, hogy egy Unix rendszer FTP szerevére csatlakoztunk, akkor a következő hasznos parancsot használhatjuk.

   Ha `ls` parancsot használunk a Unix `ls` parancsa fog lefutni. Kiegészithető az `ls` parancs bármely Unix opcióval, mint például a `-lR` opció, amely egy rekurzív hosszú formátumú listát ad a fájlokról és alkönyvtárakról. Az `-lR` opció nem szabványos FTP parancs, de alkalmazható Unix szervereken illetve ahol emulálják azt. A következő példa bemutatja az `ls -lR` parancs kimenetét. A parancs segítségével egy gyors áttekintét kaphatunk az FTP host-on elérhető fájlokról.

```
ftp> ls -lR
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
drwxr-xr-x   2 1012     vip             8 May  3 06:37 1
drwxr-xr-x   2 1012     vip             8 May  3 06:37 10
drwxr-xr-x   2 1012     vip             8 May  3 06:37 2
drwxr-xr-x   2 1012     vip             8 May  3 06:37 3
drwxr-xr-x   2 1012     vip             8 May  3 06:37 4
drwxr-xr-x   2 1012     vip             8 May  3 06:37 5
drwxr-xr-x   2 1012     vip             8 May  3 06:37 6
drwxr-xr-x   2 1012     vip             8 May  3 06:37 7
drwxr-xr-x   2 1012     vip             8 May  3 06:37 8
drwxr-xr-x   2 1012     vip             8 May  3 06:37 9
-rw-r--r--   1 1012     vip        342856 Sep 30  2003 letolt.gif
drwxr-xr-x   2 0        root           16 Nov  5  2003 protokollok
drwxr-xr-x   4 0        root           96 Sep 17  2003 pub
-rw-r--r--   1 0        root           92 Aug 28  2003 welcome.msg

1:
-rwxr--r--   1 1012     vip           372 May  3 07:24 szgh_besz_2004_01.txt

protokollok:
-rw-r--r--   1 0        root       342856 Nov  5  2003 beszamolo.gif
-rw-r--r--   1 0        root          718 Nov  5  2003 meres.html

pub:
drwxr-xr-x   2 0        root            8 Sep 17  2003 debian
drwxr-xr-x   4 0        root           24 Sep 17  2003 netbsd

pub/debian:
-rw-r--r--   1 1008     vip      195125404 Sep  3  2003 debian3_minicd.nrg

pub/netbsd:
drwxr-xr-x   2 0        root            8 Sep 17  2003 iso
drwxr-xr-x   2 0        root           80 Sep 17  2003 packages
```

```
-rw-r--r--   1 0        root          166 Sep 17  2003 url

pub/netbsd/iso:
-rw-r--r--   1 1008     vip      124387328 Sep 17  2003 netbsd_i386cd.iso

pub/netbsd/packages:
-rw-r--r--   1 1008     vip        786173 Sep 17  2003 bash-2.05nb1.tgz
-rw-r--r--   1 1008     vip         18631 Sep 17  2003 gettext-lib-0.10.35nb1.tgz
-rw-r--r--   1 1008     vip        176074 Sep 17  2003 glib-1.2.10nb3.tgz
-rw-r--r--   1 1008     vip        554810 Sep 17  2003 libslang-1.4.5.tgz
                ...
        (további kimenet)
226-Transfer complete.
226 Quotas off
```

8. Könyvtár váltásra a `cd` parancsot használjuk, az alábbi példában a `/pub/debian/packages` alkönyvtárba lépünkbe.

```
ftp> cd pub
250 CWD command successful.
ftp> cd netbsd
250 CWD command successful.
ftp> cd packages
250 CWD command successful.
```

9. A fájlok megtekintéséhez az `/pub/debian/packages` alkönyvtárban az `ls` és `dir` parancsokat használhatjuk.

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
-rw-r--r--   1 1008     vip        786173 Sep 17  2003 bash-2.05nb1.tgz
-rw-r--r--   1 1008     vip         18631 Sep 17  2003 gettext-lib-0.10.35nb1.tgz
-rw-r--r--   1 1008     vip        176074 Sep 17  2003 glib-1.2.10nb3.tgz
-rw-r--r--   1 1008     vip        554810 Sep 17  2003 libslang-1.4.5.tgz
-rw-r--r--   1 1008     vip       1527071 Sep 17  2003 mc-4.5.51-i386nb.tgz
-rw-r--r--   1 1008     vip       4330056 Sep 17  2003 mc-4.5.51.tgz
-rw-r--r--   1 1008     vip      10121200 Sep 17  2003 perl-5.8.0.tgz
-rw-r--r--   1 1008     vip        508305 Sep 17  2003 pico-4.2.tgz
-rw-r--r--   1 1008     vip        253596 Sep 17  2003 proftpd-1.2.5.tgz
-rw-r--r--   1 1008     vip        294001 Sep 17  2003 pth-1.4.1nb2.tgz
226-Transfer complete.
226 Quotas off
```

10. Ha egy olyan fájlt szeretnénk letölteni, amely nem elérhető, mint például az `rfc1365.txt`, az FTP a következőt fogja válaszolni:

```
ftp> get nano-4.5.51.tgz
local: nano-4.5.51.tgz remote: nano-4.5.51.tgz
200 PORT command successful.
550 nano-4.5.51.tgz: No such file or directory
```

11. Ahhoz, hogy egy fájlt másoljunk a távoli gépről a helyi host-ra a következő FTP parancsot kell kiadnunk:

```
get <távoli_fájl> [helyi_fájl]
```

Ebben a parancsban a *távoli_fájl* a távoli gépen lévő fájl nevét jeleni, és a *helyi_fájl* a helyi gépen lévő fájlt jelenti. Ha a *helyi_fájl-t* nem adjuk meg akkor ugyan az lesz a fájl neve a helyi gépen mint a távolin. Szövegfájloknál az FTP támogatja a jellegzetes carriage-return-to-carriage-return/linefeed konverziót a különböző operációs rendszereknél ha az átviteli mód ASCII. A bináris fájlok továbbításához használjuk az `image` vagy a `bin` parancsot hogy

kikapcsoljuk a carriage-return/linefeed konverziót.

Az alábbiakban látható egy FTP *GET* parancs:

```
tp> get pico-4.2.tgz
local: pico-4.2.tgz remote: pico-4.2.tgz
200 PORT command successful.
150 Opening BINARY mode data connection for pico-4.2.tgz (508305 bytes).
226 Transfer complete.
508305 bytes received in 0.0626 secs (7.9e+03 Kbytes/sec)
```

12. Az FTP kapcsolat bezárásához a *close* parancsot kell kiadnunk:

```
ftp> close
221 Goodbye.
```

13. Ahhoz, hogy teljesen kilépjünk az FTP-ből a `bye` parancsot használhatjuk. Ez a parancs bezárja az esetleg nyitva hagyott FTP kapcsolatot mielőtt kilép az FTP-ből:

```
ftp> bye
%
```

4.2. Trivial File Transfer Protocol (TFTP)

A TFTP az UDP protokollt használja mint átviteli protokoll. Mivel az UDP nem garantálja az adat megérkezését a TFTP-nek kell megoldania a hibajavítást. A TFTP egyszerű PAR (Positiv Acknowledgment Retransmission) sémát használ a hibajavításra. A TFTP definiál egy nyugtázó üzenetet, amellyel visszaigazolást kap a megérkezett adatról. Amennyiben egy meghatározott időn belül nem érkezik meg a nyugta, újra küldi az adatot.

A TFTP-t úgy tervezték, hogy egyszerű és kis kódmérettel alkalmazható legyen akár önbetöltő ROM-ban egy munkaállomáson. A TFTP-t gyakran használják BOOTP-vel együtt. A BOOTP-t a konfigurációs információk kinyerésére, a TFTP-t pedig az operációs rendszer image-ének letöltésére használják.

A TFTP szabványának leírása megtalálható az RFC 1350 „The TFTP Protocol (Revision 2)”-ban és kiegészítések az RFC 1782, 1783, 1784 és 1785-ben.

4.2.1. TFTP üzenetformátum

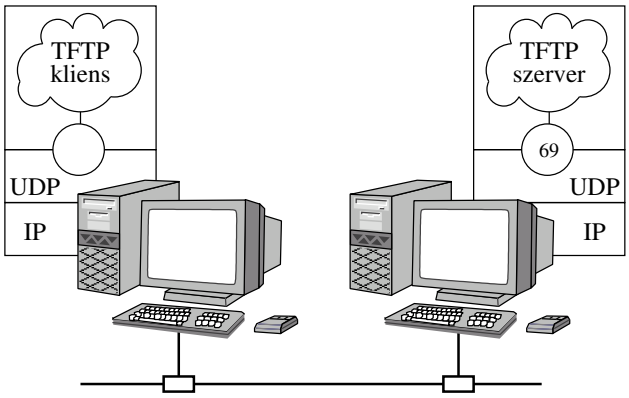
A TFTP öt féle típusú üzenetet definiál. Ezen üzenettípusok a 18. táblázatban láthatók.

Művelet kód	Leírás
1	Olvasás kérés (Read request, RRQ)
2	Írás kérés (Write request, WRQ)
3	Adat (DATA)
4	Nyugta (Acknowledgment, ACK)
5	Hiba (ERROR)

18. táblázat. TFTP üzenettípusok

Az TFTP kliens küld egy kezdő olvasás (RRQ) vagy írás (WRQ) kérés üzenetet (16. ábra) a 69-es TFTP szerver UDP portjára. A TFTP szerver ezen a porton várja az olvasás/írás kérést.

Az írás és olvasás kérés formátuma a 17. ábrán látható. Minden TFTP üzenetnek van egy *műveleti kód* mezője melynek lehetséges értékei és jelentései a 18. táblázaban megtalálhatók. Az írás és olvasás kérés üzenetekben megtalálható



Trivial File Transfer Protocol (TFTP)
–UDP felett müködik és egyszerûbb mint az FTP
–a TFTP–nek saját hibakezelése van
–nem használ autentikációt
–könnyen implementálható ROM–ban
–ideális merevlemerevlemez nélküli munkaállomások számára

16. ábra. TFTP kliens/szerver kapcsolat

egy fájlnév és egy mód mező. A fájlnév egy karakterekből álló string, amely egy nulla oktetre végződik. A mód mező meghatározza az átviendő fájl típusát. Az értéke lehet NETASCII, BINARY vagy MAIL és egy nulla oktetre végződik. Mivel a fájlnév és a mód is nulla oktetre végződik változó méretűek lehetnek, mert a null oktet fogja jelezni a végüket.

2 oktet	string	1 oktet	string	1 oktet
Mûveletkód	Fájlnév	0	Mód	0

(a) RRQ csomag

2 oktet	string	1 oktet	string	1 oktet
Mûveletkód	Fájlnév	0	Mód	0

(b) WRQ csomag

2 oktet	2 oktet	n(<=512) oktet
Mûveletkód	Blokk sorszám	Adat

(c) DATA csomag

2 oktet	2 oktet
Mûveletkód	Blokk sorszám

(d) ACK csomag

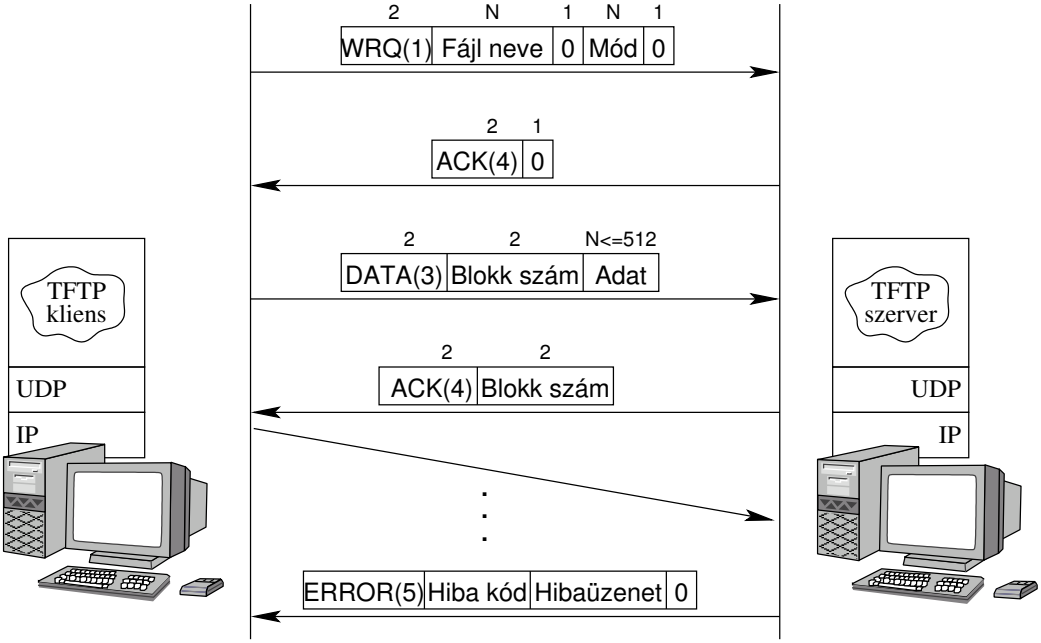
2 oktet	2 oktet	string	1 oktet
Mûveletkód	Hibakód	Hibaüzenet	0

(e) ERROR csomag

17. ábra. TFTP üzenet formátum

4.2.2. TFTP művelet

Az írás/olvasás kérés megérkezése után a TFTP szerver rögzít a kliens IP címét és portszámát. A későbbiekben a válaszokat a TFTP szerver ezen IP címre és portszámra továbbítja. Az írás/olvasás kérés nyugtázása után az adatátviteli folyamatban adat tartalmú üzenetek és azok nyugtázásai következnek. A TFTP 512 oktet méretű üzenteket küld. Ha az adat tartalmú üzenet mérete kisebb mint 512 az az átvitel végét jelenti. Amenyiben az áviendő fájl mérete (oktetben) az 512 egész számú többszöröse, akkor küld egy olyan plusz nulla adat tartalmú üzenetet, amely jelzi az adatátvitel végét. Az adatblokkok sorszáma egyesével nő. Adat és nyugta tartalmú üzenetek tartalmaznak egy blokk számot amint elkezdődött az adatátvitel. A 18. ábrán látható egy adatátvitel, melyben a TFTP kliens adta ki az írás kérés üzenetet.



18. ábra. Példa egy TFTP adatátvitelre

A TFTP felhasználó azonosítás nélkül képes az adatátvitel lebonyolítására. A fájlok átvitelekor csak a fájl nevét kell megadni. Mivel felhasználói account és jelszó nélkül használható a rendszergazdák általában letiltják ezt a funkciót, vagy korlátozzák az átvihető fájlok körét.

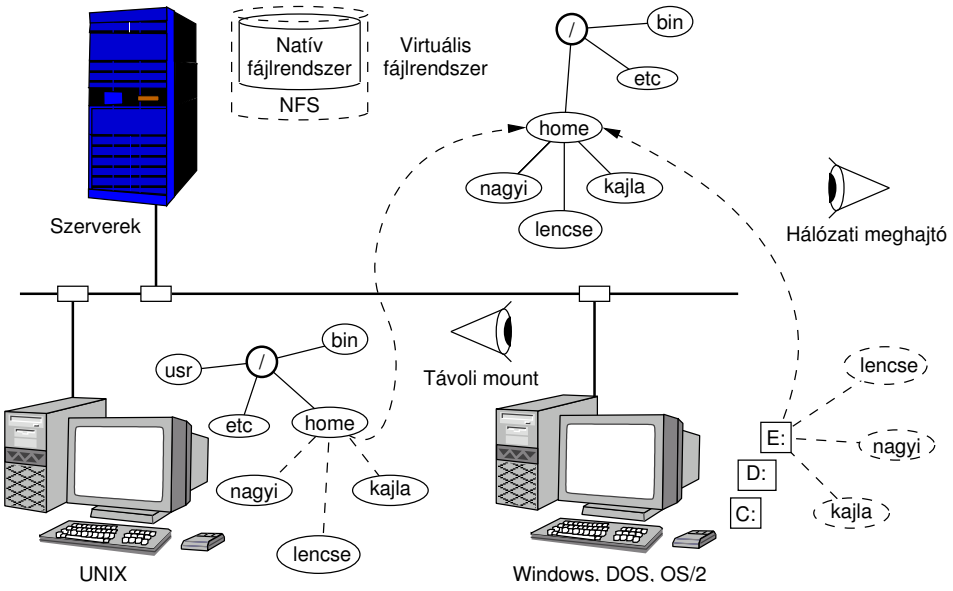
4.3. Network File System Protocol (NFS)

A Network File System (NFS) egy fájlkezelő protokoll, melyet eredetileg a SUN Microsystems fejlesztett, majd sok gyártó megvásárolta a licenszét. Az NFS lehetővé teszi, hogy a számítógép, amelyen az NFS szerver program fut *exportálja* a fájlrendszerét egy kliensnek. A fájlrendszer *kiadása* (*exportálása*) azt jelenti, hogy a kliens számára elérhető lesz az, még ha a szoftvertől eltérő operációs rendszert használ is, feltéve ha fut az NFS kliens program.

A 19. ábrán látható amint az NFS szerver exportálja a `/home` könyvtárat. Ez az exportált könyvtár egyszerre több kliens által is elérhető különböző operációs rendszert futtató gépekről. Minden NFS kliens úgy látja az exportált fájlrendszert, mint a saját fájlrendszerének egy részét. Például egy PC DOS NFS kliens egy hálózati meghajtóként fogja elérni a fájlrendszert, egy Unix NFS kliens saját fájlrendszerébe linkelve érheti el.

4.3.1. NFS protokollok

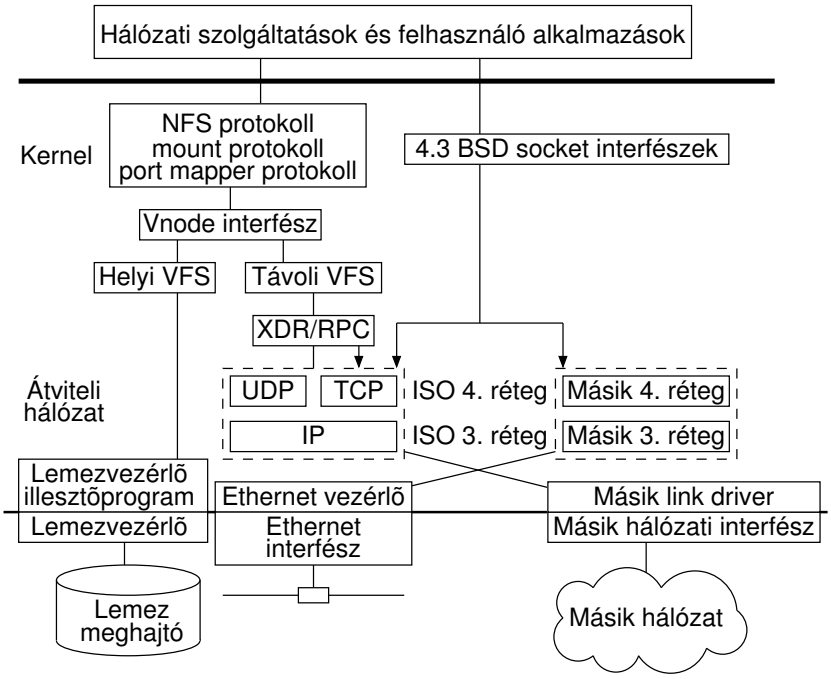
Az NFS hálózati alkalmazás tartalmaz egy magas szintű NFS protokollkészlet segítségével valósították meg. A 20. ábrán különböző NFS protokollok láthatók. Az OSI adatkapcsolati és fizikai szintjein számos technológiát támogat az NFS. A hálózati rétegben az NFS IP-t használ. A szállítási rétegben az NFS UDP-t vagy TCP-t használ. Ha az NFS-t UDP



19. ábra. Az NFS hasznlata

felett használjuk, akkor nagyon ajánlott az opcionális UDP ellenőrző összeget használni.

A viszony rétegben az NFS a Remote Procedure Call (RPC, távoli eljáráshívás) protokollt használja. Több egymástól teljesen eltérő RCP protokoll létezik, ezért megkülönböztetés képpen az NFS RPC-jét gyakran hívják Sun-RPC-nek. Az RPC lehetővé teszi, hogy az NFS szolgáltatásokat a szerveren az eljáráshívás módszerével érjék el, amit a programozók jól ismernek. Az RPC magas szintű hozzáférést biztosít az NFS szerverhez anélkül, hogy a kommunikációs protokollok bonyolult részleteibe merülne. Az programozóknak nem kell elmélyedniük a kommunikáció rejtelmeiben, hogy RPC-vel elérhető hálózati szolgáltatásokat használjanak. A Sun-RPC protokoll dokumentációja megtalálható az RFC 1057-ben „RPC: Remote Procedure Call Protocol Specification version 2.” cím alatt.



20. ábra. Az NFS protokollok

Az OSI megjelenítési rétegében az NFS az External Data Representation (XDR) protokollt használja. Az XDR biztosítja az egységes adatmegjelenítést. Például a számok megjelenítésére a kettes komplemens jelölést használják. Ha a rendszer más megjelenítést használ a konverziót az XDR végzi el. Az XDR protokoll dokumentációja megtalálható az RFC 1832-ben „XDR: External Data Representation Standard” cím alatt.

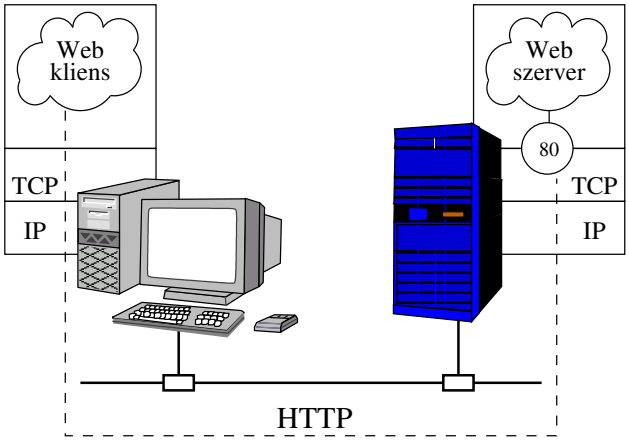
Az OSI alkalmazási rétegében az NFS az Network File System (NFS) protokollt használja. Az NFS olyan műveletek végrehajtását teszi lehetővé, mint például fájl írás, létrehozása, olvasása az NFS szerveren. Az alkalmazási réteg néhány protokollal támogatja az NFS-t, úgy mint például a *Mount* és a *Portmapper* protokollok. A Mount protokoll végzi az NFS mount-olási folyamatát, és a Portmapper protokoll adja meg a szolgáltatás eléréshez a port számát a kliensek. A Portmapper protokoll a szabványos 111-es UDP port-ot használja. Ha egy NFS kliens hozzáférést szeretne valamilyen NFS szolgáltatáshoz, egy kérést küld a 111-es UDP port-ra. A szerver programok, melyek valamilyen szolgáltatást szeretnének nyújtani beregisztrálják magukat a Portmapper-nél. Ha beérkezik egy kérés egy regisztrált szolgáltatásra, a Portmapper választ küld, amely tartalmazza a port számát, melyen keresztül elérhető a szolgáltatás.

5. Internet hozzáférési protokollok

Számos protokollt hoztak létre, hogy dokumentumokat tegyenek elérhetővé az Interneten. A két kiemelkedő protokoll a HTTP (HyperText Transfer Protocol) és a Gopher. Ezen protokollok közül a legszélesebb körben a HTTP-t használják, melyet népszerűbb nevén World Wide Web (WWW) protokollnak hívnak.

5.1. World Wide Web

A 21. ábra a World Wide Web használatát illusztrálja. A World Wide Web-et gyakran csak web-nek hívják. A web sok web szerverből áll. A web szerverek dokumentumokat tárolnak, amelyek tartalmazhatnak például szöveget, képet, hangot és videót, melyeket *weblapokba* (web pages) szerveztek. Minden weblap tartalmazhat *link*eket (hyperlink), amelyek mutatók web dokumentumokra. A hyperlink-ek mutathatnak azonos, illetve különböző web szerveren lévő dokumentumokra egyaránt. A linkekkel keresztül-kasul behálózott dokumentumok szövevénye átnyúlik ország és kontinens határokon, ezt fejezi ki a World Wide Web (világ méretű pókháló), magyarítása: világszőttes.



21. ábra. A World Wide Web

A web dokumentumokat a web klienseken, azaz web böngészőkön keresztül érheti el a felhasználó, melyet saját gépén futtat. A weblapokat, mint dokumentumot egy speciális nyelven kódolják, amelyet HTML-nek (HyperText Markup Language) nevezünk. Miután a böngésző letöltötte a HTML dokumentumot, megjeleníti azt a képernyőn grafikusan, esetleg szöveges formátumban. A HTML tartalmazza a HTML dokumentum különböző elemeinek leírását, melyeket felhasználva a böngésző grafikusan megjelenítheti a weblapot. A linkek általában aláhúzott szövegként jelennek meg. (Vigyázat kép is lehet link!) A linkekre kattintva letölti azt a dokumentumot, amire a link mutat. A következőkben a HTML nyelvről és a HTTP protokollról lesz szó.

5.1.1. HyperText Markup Language (HTML)

A HTML a Standard Generalized Markup Language (SGML) egy implementációja. Az SMGL szabvány megad egy általános módszert, hogy miképpen készítsünk olyan dokumentumokat, amelyek hyperlink-eket tartalmaznak. A *hyperlink* egy kiemelve látható kifejezés illetve egy mondat a szövegben, melyet kiválasztva egy újabb dokumentumot kapunk. Egy hyperlink kiválasztása többféle cselekményt is kiválthat, mint például egy kép megjelenítése, levél küldése, felhasználótól adatok kérése form-on keresztül, távoli bejelentkezés vagy fájl átvitel kezdeményezése, adatbázis lekérdezése, program futtatása és így tovább. Az olyan dokumentumot, amely hyperlink-et tartalmaz hyper-dokumentumnak is hívjuk.

Minden HTML dokumentum tag-eket tartalmaz (magyarul talán *cimkének* lehetne nevezni), a következő struktúrában:

```
<tag>
</tag>
```

A *tag* egy hivatkozás egy speciális kulcsszóra, melyet a HTML dokumentum különböző komponenseinek megadására használunk. A lezáró tag `</tag>` megadja a komponens végét. Majdnem minden HTML tag-nek megvan a hozzá tartozó lezáró tag-je. Például minden HTML documentum eleje és vége követi a következő megadást:

```
<HTML>
A HTML különböző elemeit e két tag közé kell elhelyezni.
</HTML>
```

A <HTML> és </HTML> tag-ek között meg kell adni a HTML dokumentum fejrészét és törzsét. A fejrészt a <HEAD> és </HEAD> tag-ek között, és a törzset pedig a <BODY> és </BODY> tag-ek között kell megadni:

```
<HTML>
<HEAD> Ez itt a fejrész helye. </HEAD>
<BODY> Ez itt a törzs helye. </BODY>
</HTML>
```

Az üres sorokat és a sortörés karaktereket a tag-ek között nem vesszük figyelembe. Például a következő HTML kód jelentésben teljesen megegyezik az előzővel:

```
<HTML><HEAD> Ez itt a fejrész helye. </HEAD>

<BODY> Ez itt a törzs helye. </BODY>
</HTML>
```

Az üres sorok és a sortörések általában javítják a HTML dokumentumok forrásának olvashatóságát, ezért ajánlatos alkalmazni. A tag-ek nem érzékenyek a kis illetve nagy betűkre és az egyéb szövegformázásra. Ha azt szeretnénk, hogy a böngésző által megjelenített weblapon sortörést, újsort, vagy egyéb formázást hajtsunk vége, akkor speciális HTML tag-eket kell a forrásban elhelyeznük. A weblap címét a <TITLE> és </TITLE> tag-ek között kell megadni. A következő szöveg a böngésző címrészében (az ablak címsorában) fog megjeleni:

```
<HTML>
<HEAD>
<TITLE>
Ez az első egyszerű weblapom!
</TITLE>
</HEAD>
<BODY> Ez itt a törzs helye. </BODY>
</HTML>
```

Hyperlink megadásához használjuk a következő tag-et:

```
<A HREF=URLcím és egyéb paraméterek> hyperlink szöveg </A>
```

A tag-nek számos paramétert lehet megadni az *egyéb paraméterek* résznél, melyek modósítják a tag viselkedését. Általában a HREF paramétert használjuk az URL cím megadására. Az URL cím egy szabványos út, hogy meghatározzunk egy forrás helyét az Interneten. Az URL általános szintakszisa:

```
protokoll://gépnév/elérési_út
```

A *protokoll* bármelyik lehet az Interneten használt protokollok közül, mint például a HTTP, FTP, Telnet, Gopher, File. Ha megadunk egy HTML dokumentumot egy másik gépen, akkor a HTTP (HyperText Transfer Protocol) protokollt szoktuk használni a letöltésére. Ha egy dokumentumot fájl transzferrel szeretnénk letölteni az FTP protokollt kell használnunk, és így tovább. Ha egy helyi fájlt szeretnénk megnyitni a saját gépünkről a web böngészőben, akkor a File protokollt kell megadnunk.

A *gépnév* egy IP cím vagy egy DNS (Domain Name System) szimbolikus név, amely az erőforrást birtokló host-ot adja meg. Az *elérési_út* a könyvtárat és fájl nevét adja meg, ahol a könyvtár/fájl elérhető. Az *elérési_út* érzékeny lehet a kis és nagy betűkre, ha a web szerver Unix-ot használó host-on fut. Az *elérési_út* opcionális. Ha nem adjuk meg akkor a HTML dokumentum alapértelmezett neve index.html vagy index.htm. Itt látható néhány példa az URL címekre:

```
http://www.sze.hu

http://tilb.sze.hu

http://tilb.sze.hu/ta13

ftp://tilb.sze.hu

telnet://tai2.szif.hu

file://home/lencse/public_html/index.html
```

Nézzük a következő HTML dokumentumforrást, amely hyperlink-eket tartalmaz a különböző erőforrások elérésére. Megjegyzéseket a következőképpen írhatunk a forrásba: `<!--Megjegyzés>`.

```
<HTML>
<HEAD><TITLE> Hyperlink </TITLE></HEAD>
<BODY>
<!-- Ez egy megjegyzés, a használatához jegyezze meg a szintaktikát.>

<H1> Bemutató a hyperlink-ek használatáról</H1>
<P> Ezt a bekezdést egy bekezdés tag-gel kezdtük. A bekezdés tag
automatikusan formázza a szöveget megjelenítéskor. A bekezdést
egy sortöréssel befejezhetjük, így az azt következő szöveget nem
formázza.
<BR><!-- Az előző tag sortörést idéz elő a szövegben.>
A hyperlink-eknek sokféle paraméterük lehet, de itt csak a
legegyszerűbb megadást használjuk.
<HR><!-- Az előző tag egy vízszintes vonalat húz.>
<P> Itt vannak a hivatkozások:
<P><A HREF=http://tilb.sze.hu> A TILB web szerver </A>
<P><A HREF=ftp://tilb.sze.hu> A TILB ftp szerver </A>
<P><A HREF=telnet://tai2.szif.hu>Bejelentkezés a TAI2-re</A>
<P><A HREF=mailto:lencse@sze.hu> Levél küldés </A>
</BODY>
</HTML>
```

A HTML kód tanulása közben az aktuális megjelenést figyelve a web böngészőn könnyebben megérhetjük a különböző HTML elemeket.

Az előző HTML dokumentumban a **<P>** tag-ek külön sorba tettek minden egyes hyperlink-et. A HTML lehetőséget ad számozott és számozatlan listák létrehozására, melynek tagjai ugyancsak külön-külön sorban lesznek. A számozatlan lista minden tagja elé egy jelet tesz. A lista szintaktikája a következő:

```
<!-- Számozott lista>
<OL>
```

```
<LI> Lista elem 1
<LI> Lista elem 2

<LI> Lista elem N
</OL>
<!-- Számozatlan lista>
<UL>
<LI> Lista elem 1
<LI> Lista elem 2

<LI> Lista elem N
</UL>
```

A következő HTML kód megmutatja, hogy miképp tegyünk hyperlink-eket számozott listába. Definiálunk két új hyperlink tag paramétert. Az egyik a már ismert HREF, azonban egy speciális # karaktert tartalmazó string-re mutat. Ez a hyperlink az éppen megjelenített dokumentum egy bizonyos helyére mutat. Azt a bizonyos helyet, pedig a NAME paraméter adja meg, méghozzá úgy, hogy ugyanolyan nevű string-re mutat, mint a HREF, csak a # karaktert nem tartalmazza:

```
<HTML>
<HEAD><TITLE> Hyperlink </TITLE></HEAD>
<BODY>
<!-- Ez egy megjegyzés, a használatához jegyezze meg a szintaktikát.>

<H1> Bemutató a hyperlink-ek használatáról </H1>
<P> Ezt a bekezdést egy bekezdés tag-gel kezdtük. A bekezdés tag
automatikusan formázza a szöveget megjelenítéskor. A bekezdést
egy sortörések befejezhetjük, így az azt következő szöveget nem
formázza.
<BR><!-- Az előző tag sortörést idéz elő a szövegben.>
A hyperlink-eknek sokféle paraméterük lehet, de itt csak a
legegyszerűbb megadást használjuk.
<HR><!-- Az előző tag egy vízszintes vonalat húz.>
<P> Itt vannak a hivatkozások:
<B><!--Vastag betűs szedés>
<OL><!-- Számozott lista kezdete>
<LI><A HREF=http://tilb.sze.hu> A TILB web szerver </A>
<LI><A HREF=ftp://tilb.sze.hu> A TILB ftp szerver </A>
<LI><A HREF=telnet://tai2.szif.hu> Bejelentkezés a TAI2-re</A>
<!-- A következő hypelink-ek ebben a dokumentumban ugranak. >
<LI><A HREF="#Heading1">Heading1 fejrész </A>
<LI><A HREF="#Heading2">Heading2 fejrész </A>
</OL><!-- Számozott lista vége>
</B><!-- Vastag betűs szedés vége>
<HR>
<!-- A fejrész szintek a H1-től H6-ig terjednek.>
<A NAME="Heading1"><H1>Heading1 fejrész</H1></A>
<A NAME="Heading2"><H2>Heading2 fejrész</H2></A>
</BODY>
</HTML>
```

A **??** ábrán látható az előző HTML kód web böngésző által megjelenített képe. Érdemes még megjegyezni, hogy a <B> és </B> tag-ek közé írt szöveg vastag betűs, míg a <I> és </I> tag-ek közé írt szöveg dőltbetűs lesz. Továbbá használhatunk fejrészeket, melyek <H1>-től <H6>-ig különböző megjelenítésűek.

5.1.2. HyperText Transfer Protocol (HTTP)

A web kliens parancsokat és a HTML dokumentum válaszait a web kliens parancsaira mind egy speciális protokollt használva továbbítjuk, melynek neve HyperText Transfer Protocol (HTTP). A web szervert más néven HTTP szervernek hívják, esetleg HTTP daemon-nak Unix rendszereken, mivel ezek küldik a válaszokat a HTTP kérésekre.

Ha a böngésző keres egy HTML dokumentumot, létrejön egy kapcsolat a böngésző és a web szerver között. A TCP kapcsolat alapértelmezetten a 80-as port-on jön létre. Megadhatunk más postokat is mint például a 8080-as port, de alapértelmezetten a 80-as port-ot foglalták le a HTTP számára.

Ha azt szeretnénk, hogy a web böngészőnk ne az alapértelmezett port-ot használja, akkor az URL címben meg kell adnunk a kívánt portot. Az alábbi példában egy URL-t láthatunk, amely a 8080-as portra jelentkezik be:

```
http://tilb.sze.hu:8080
```

A kapcsolat létrejötte után a web böngésző küld egy kérést a dokumentumért a HTTP protokollt használva. A kérés tartalmazza a letölteni kívánt objektum nevét és az általunk használt web böngésző verziószámát. A HTTP protokoll szöveg string-eket használ, így az emberek számára is könnyen érhetőek a kiadott parancsok. Például a következő HTTP kérést küldi el a web böngésző a web szervernek egy dokumentum letöltéséhez:

```
GET http://tilb.sze.hu/index.htm HTTP/1.0
```

Ebben a kérésben a *GET* HTTP kérést használjuk a dokumentum *http://tilb.sze.hu/index.htm* letöltéséhez. A HTTP argumentummal pedig megadtuk, hogy milyen verziószámmal rendelkezik a web böngészőnk (1.0).

A szerver a HTML dokumentum letöltésére irányuló kérés megérkezése után küld egy HTTP választ. A HTTP válasz három részből áll:

- státusz
- fejléc
- adat

A válasz státusz egy sor szöveg amely tartalmazza a szerver HTTP verziójának számát, a státusz kód megmondja a szerver eredményét, az utána következő szöveg segít a státusz kód jelentését értelmezni. A következő példában látható egy válasz státusz:

```
HTTP/1.1 200 OK
```

A válasz státusz után következik a válasz fejléc. A válasz fejléc a következő információkat tartalmazza: a szervez típusa, MIME verziószám, a tartalom típusa (megadja, hogy az adat részben milyen típusú tartalom van), és végül egy üres sor. A fejlécet és az adatot mindig egy üres sor választja el. Az alábbiakban látható egy példa a válasz fejlécre:

```
Date: Tue, 20 Jul 2004 14:24:29 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2
Last-Modified: Tue, 07 Oct 2003 07:33:45 GMT
Tag: "1025-b2-3f826c59"
Accept-Ranges: bytes
Content-Length: 178
Connection: close
Content-Type: text/html; charset=iso-8859-2
```

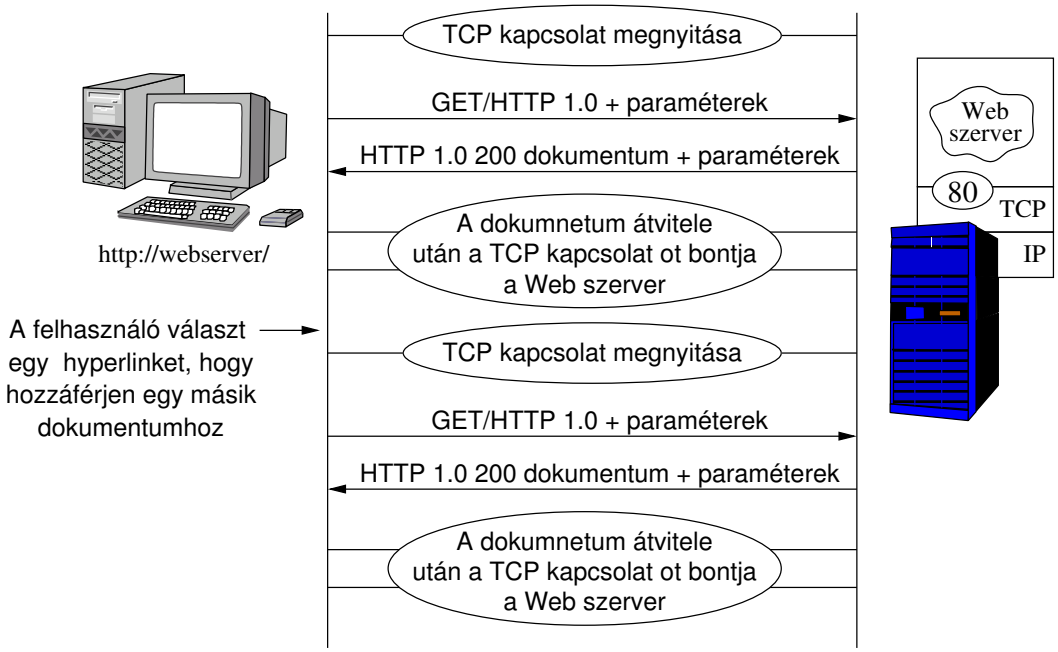
Ha a dokumentum amit lekérdezünk egy HTML dokumentum, akkor a válaszadat egyszerű szöveg lesz, amely a HTML dokumentumot tartalmazza. Egy példa a válasz adatra:

```
<HTML>
<FRAMESET COLS=198,* border=0>
<FRAME SRC="menu.html" name="menu" target="menu" scrolling=no NORESIZE>
<FRAME SRC="folap.html" name="main" target="main" NORESIZE>
</HTML>

JOK
```

A fenti HTML dokumentum frame-ekre osztja a az oldalunkat, tehát ezt az oldalt nem látjuk, csak megmondja hova mit kell betölteni.

Miután a HTTP kérés/válasz lebonyolódott a TCP/IP kapcsolat bezáródik. Külön TCP/IP kapcsolat jön létre minden egyes HTTP kérés/válasz folyamathoz.

A 22. ábrán látható egy tipikus HTTP kapcsolat. Megjegyzem az első lépés a TCP kapcsolat megnyitása a 80-as porton. A következő lépésben a kliens küld egy GET/HTTP 1.0 parancsot. A dokumentum átvitele után a kapcsolatot bezárja a web szerver. Egy másik dokumentum letöltéséhez egy új kapcsolatot kell létrehozni.



22. ábra. Egy tipikus HTTP kapcsolat

Ahhoz hogy megértsük, hogyan működik a HTTP protokoll végezzünk egy kísérletet. Egy Telnet alkalmazás segítségével jelentkezzünk be egy web szerver 80-as portjára a következő módon:

```
telnet <webhost> 80
```

A *webhost*-ot helyettesítsük valamilyen web szerver IP címével vagy szimbolikus nevével. A 80-as szám megadja, hogy a távoli gép web szerveréhez akarunk csatlakozni. Amennyiben nem adnánk meg a port számát, a Telnet alkalmazás az alapértékként megadott 23-as Telnet portra csatlakozna. A Telnet-et bármilyen Unix illetve MS Windows alapú operációs rendszerben megtaláljuk.

A kapcsolat létrehozása után a következőhöz hasonló üzenetet küld a web szerver:

```
telnet debian.org 80
Trying 192.25.206.10...
Connected to debian.org.
Escape character is '^]'.
```

A következő lépésben úgy teszünk mintha mi lennénk a web böngésző kliens és kiadjuk a HTTP GET parancsot a következő képpen:

```
GET /index.html HTTP/1.0
```

Miután lenyomtuk az enter billentyűt nem történik semmi, mivel a HTTP kérést két <CR><LF> karakternek kell követnie. Miután másodszor is lenyomjuk az enter billentyűt megjelenik a HTTP válasz a következő képpen:

```
GET /index.html HTTP/1.0
```

```
HTTP/1.1 200 OK
Date: Tue, 20 Jul 2004 15:40:37 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2 DAV/1.0.3
Last-Modified: Sat, 26 Oct 2002 09:12:14 GMT
ETag: "32c003-100e-3dba5c6e"
Accept-Ranges: bytes
Content-Length: 4110
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
<META NAME="GENERATOR" CONTENT="Mozilla/4.05 [en] (X11; I; Linux 2.3.99-pre3 i686)
[Netscape]">
<META NAME="Author" CONTENT="johnie@debian.org (Johnie Ingram)">
<META NAME="Description" CONTENT="The initial installation of Debian/GNU Apache.">
<TITLE>Welcome to Your New Home Page!</TITLE>
</HEAD>
<BODY TEXT="#000000" BGCOLOR="#FFFFFF" LINK="#0000EF" VLINK="#55188A"
ALINK="#FF0000">
<BR>
<H1>Welcome to Your New Home in Cyberspace!</H1>
<HR NOSHADE>
<BR>
<IMG ALIGN="right" ALT="" HEIGHT="247" WIDTH="278" SRC="icons/jhe061.gif">
<P>This is a placeholder page installed by the <A HREF="http://www.debian.org/">
Debian</A>
release of the <A HREF="http://www.apache.org/">Apache</A> Web server package,
because no home page was installed on this host. You may want to replace
this as soon as possible with your own web pages, of course....
<BLOCKQUOTE>
This computer has installed the Debian GNU/Linux operating system but has
nothing to do with the Debian GNU/Linux project.  If you want to
report something about this host's behavior or domain, please contact the
ISPs involved directly, <strong>not</strong> the Debian Project.
<P>See the <A HREF="http://www.abuse.net/">Network Abuse Clearinghouse</A>
for how to do this.
</BLOCKQUOTE>
...
</BODY>
</HTML>
Connection closed by foreign host.
```

A HTTP válaszban megtalálható a státusz a fejléc illetve egy üres sor után az adat rész is. A HTML dokumentum átvitele után a kapcsolat automatikusan megszakad. Ha egy másik dokumentumot szeretnénk letölteni a HTTP/1.0-val egy új kapcsolatot kell nyitnunk. Mivel az, hogy minden egyes HTML dokumentum letöltéséhez külön kapcsolatot kell létrehozni nem túl effektív, ha több fájlt is le szeretnénk tölteni, az újabb HTTP protokollokban ezt a tulajdonságot előreláthatólag megváltoztatják.

Az előbb látott példában sikeresen letöltöttünk egy HTML dokumentumot. Próbáljuk ki, hogy mi történik, ha egy olyan dokumentumot szeretnénk letölteni, ami nem létezik. Hozzunk létre egy újabb Telnet kapcsolatot a web szerverrel és adjuk ki például a következő HTTP parancsot:

```
GET /asdfasdf.htm HTTP/1.0
```

Nyomjuk le kétszer az enter-t és a következőhöz hasonló válasz kapunk:

```
HTTP/1.1 404 Not Found
Date: Tue, 20 Jul 2004 15:46:09 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2 DAV/1.0.3
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>404 Not Found</TITLE>
</HEAD><BODY>
<H1>Not Found</H1>
The requested URL /asdfasdf.htm was not found on this server.<P>
<HR>
<ADDRESS>Apache/1.3.26 Server at gluck.debian.org Port 80</ADDRESS>
</BODY></HTML>
Connection closed by foreign host.
```

A HTTP válasz ilyenkor is tartalmaz egy HTML dokumentumot, amelynek tartalma elárulja, hogy a kért dokumentumot nem találta. A hibakód melyet visszaadott a 404-es. A HTML dokumentum, amelyet a web szerver visszaadott egy *virtuális dokumentum*. Ilyen dokumentumok nem találhatók meg a web szerveren, ezeket dinamikusan generálja a szerver, hasonló természetű hibák esetén.

Próbáljuk ki milyen válasz küld a web szerver egy olyan hibásan kiadott HTTP parancsra, mint például a következő:

```
GET ezt egy hibásan kiadott parancs
```

A web szerver 400-as hibakóddal visszaküld egy virtuális HTML dokumentumot, amely azt jelenti, hogy hibás HTTP kérést kapott:

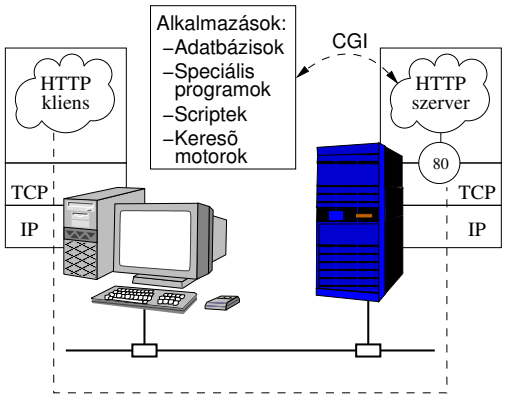
```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>400 Bad Request</TITLE>
</HEAD><BODY>
<H1>Bad Request</H1>
Your browser sent a request that this server could not understand.<P>
Invalid URI in request GET hibásasfsbgs<P>
<HR>
<ADDRESS>Apache/1.3.26 Server at gluck.debian.org Port 80</ADDRESS>
</BODY></HTML>
Connection closed by foreign host.
```

A HTTP version 1.1 megtalálható az RFC 2068-ban „HyperText Transfer Protocol HTTP version 1.1” cim alatt.

5.1.3. Web indexelés és CGI átjárók

Ahhoz, hogy megtaláljunk egy adott kulcsszót a web-en, számos kereső motort használhatunk. A kereső motorok indexelnek web dokumentumokat, követnek minden linket a web dokumentumban és indexelnek minden megtalált dokumnetumot. Néhány ezekből a kereső motorokból csak különálló szervezetekben futnak. Sok web böngésző tartalmaz elérési módozatokat ezekhez a kereső motorokhoz. Például kereső motort találhatunk a www.google.com-on, a www.hudir.hu-n és még számos helyen.

Számos kereső motor használ Common Gateway Interface-t (CGI), amely engedélyezi, hogy más alkalmazásokat is futtasunk (lásd 23. ábra). A CGI-t arra használják, hogy kéréseket továbbítson más alkalmazásoknak, melyek ugyanazon vagy akár egy másik szerveren futnak. A futtatás eredményét visszaküldi az alkalmazás a CGI-n keresztül a HTTP szervernek, majd az továbbküldi a web kliensnek.



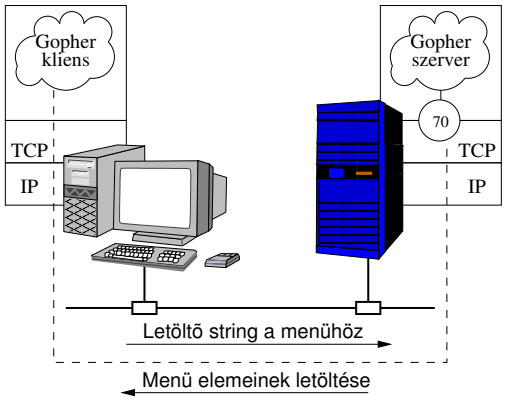
23. ábra. Common Gateway Interface (CGI)

5.2. Gopher

Igaz a fontosabb új információkat nyilvános használatra a web szervereken kersztül teszik ki az Internetre, de vannak régebbi hasznos információk melyeket régebbi eszközökkel lehet csak elérni, mint az FTP vagy a Gopher.

A Gopher protokollt és eszközöket minnesotai egyetem fejlesztette és így elérhetővé tett rendszereket menükön keresztül, ahol a címsorok között navigálhatunk. A Gopher-ben a menü címek a hyperlink-ek. Ha kiválasztunk egy menüpontot egy újabb menüt vagy egy dokumentumot nyithatunk meg ugyanazon vagy akár egy másik szerveren. A menük készletét, melyek egymásba vannak link-elve *gopher space*-nek hívjuk.

A 24. ábrán láthatunk egy kapcsolatot a Gopher kliens és a Gopher szerver között. A Gopher szerver a 70-es TCP port-on várja a szöveg letöltő string-et, melyet a Gopher kliens küld. A Gopher szerver visszaküld egy menüt, amely a szöveg letöltő string-en alapul.

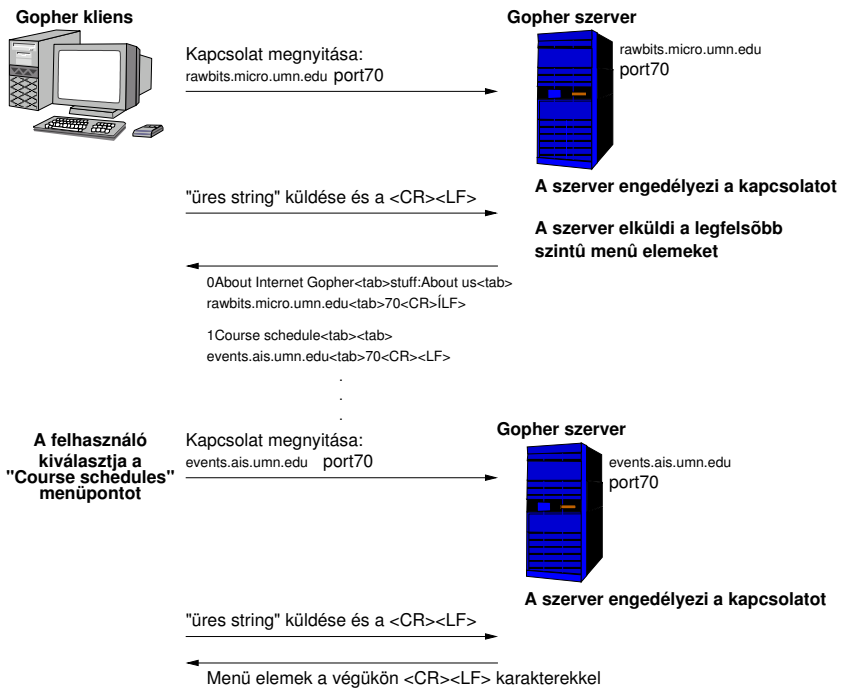


24. ábra. Gopher kliens/szerver

Egy speciális eszköz amelynek a neve Veronica (very easy rodent-oriented networkwide index to computerized archives) amit azonosítanak a Gopher-rel. A Veronica általában megtalálható a Gopher menük legfelső szintjeiben.

A Gopher kapcsolat azzal kezdődik, hogy a Gopher kliens küld egy TCP kapcsolatfelépítési kérést a Gopher szerver 70-es portjára (lásd 25. ábra). A Gopher engedélyezi a kapcsolatot. A Gopher kliens küld egy üres string-et („”), ami egy kocsi vissza és egy új sor karakterrel végződik. A Gopher szerver válaszában ott van ez a string és még a menü elemek a menü legfelsőbb szintjéről. A menü elemek küldésének formátuma a 26. ábrán látható.

A válaszó mező (lásd 26. ábra) lehetséges értékei a 19. táblázatban láthatók. A display string az a szöveg amelyet Gopher kliens menüelemként meg fog jeleníteni. A fájl hivatkozás mező egy speciális string amelyet a Gopher kliens küld



25. ábra. Gopher művelet

Választó	<tab>	Display string	<tab>	Fájl hivatkozás	<tab>	Host név	<tab>	Host portszám

26. ábra. Gopher menüelem formátum

a szervernek, hogy letöltse a mögötte lévő információt, ha kiválasztjuk az adott menüpontot. A host név és a host port mezők definiálják a Gopher szerver TCP végpontját, ahol a megadott fájlhivatkozást megtalálhatjuk.

Választó	A tartalom típusa
0	Fájl
1	Könyvtár
2	CSO telefonkönyv szolgáltatás
3	Hiba
4	Macintosh BinHex fájl
5	DOS bináris archív
6	Unix uuencode-olt fájl
7	Index kereső szerver
8	Pontok szöveg alapú Telnet kapcsolathoz
9	Bináris fájl
+	Redundáns szerver
T	Pontok szöveg alapú tn3270 kapcsolathoz
g	GIF formátumú grafikus fájl
I	Kép fájl

19. táblázat. A választó mező lehetséges értékei

A Gopher protokoll megtalálható az RFC 1436-ban a „The Internet Gopher Protocol (a distributed document search and retrieval protocol).” címen.

Tartalomjegyzék