

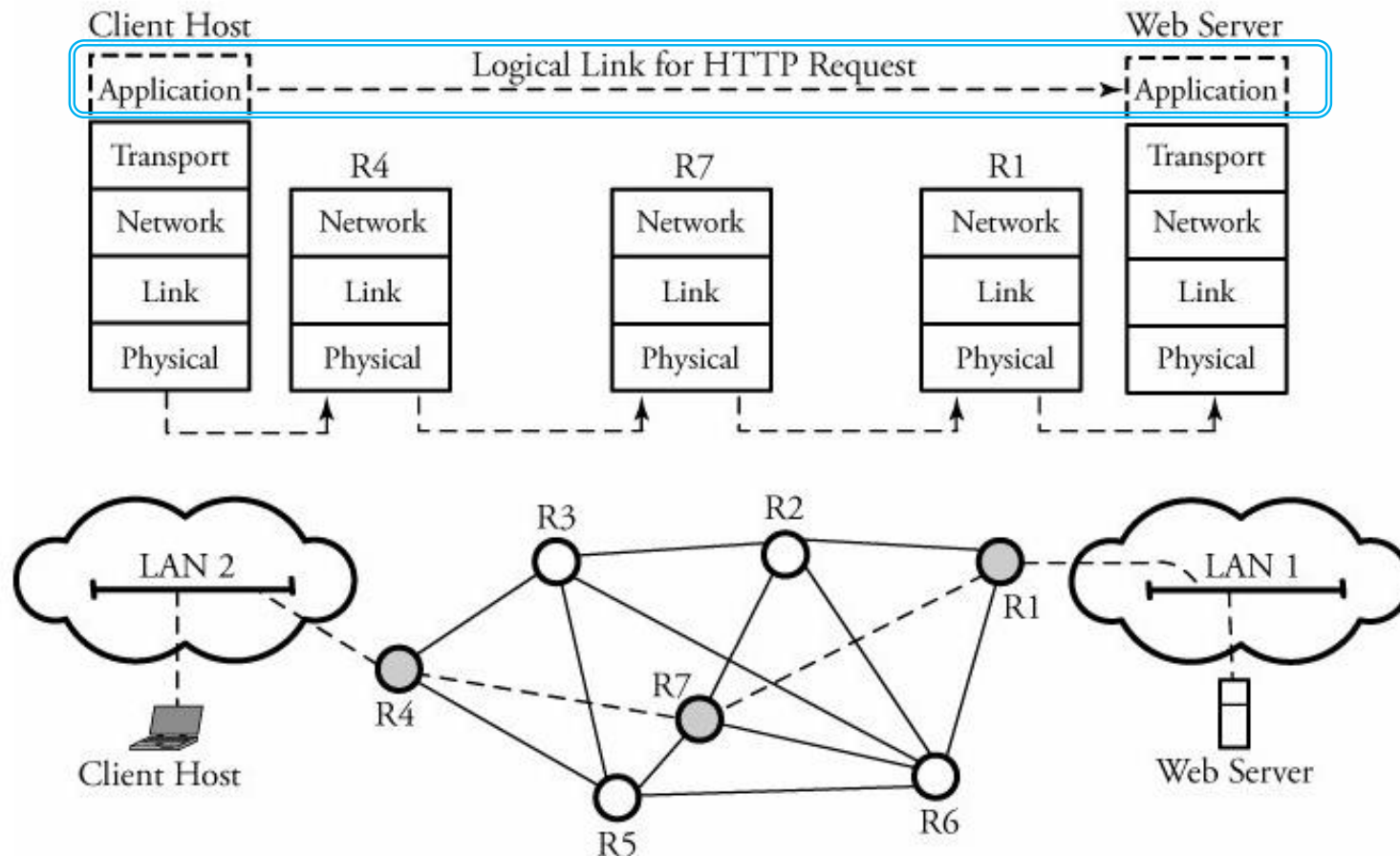
Nivelul Aplicatie

Lenuta Alboaie (adria@info.uaic.ro)
Andrei Panu (andrei.panu@info.uaic.ro)

Cuprins

- **Protocole la nivelul aplicatie**
 - Preliminarii
 - Caracteristici de proiectare
 - Accesul la terminal de la distanta
 - Posta Electronica
 - SMTP (Simple Mail Transfer Protocol)
 - POP (Post Office Protocol)
 - Transferul de fisiere
 - TFTP (Trivial File Transfer Protocol)
 - FTP (File Transfer Protocol)
 - World-Wide Web (HTTP)
 - Privire de ansamblu

Preliminarii



Comunicare intre doua *end-systems*

[Computer and Communication Networks , Nader F. Mir, 2006]

Preliminarii

Application	Application layer protocol	Underlying transport protocol
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP (eg Youtube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	typically UDP

Preliminarii

- **La nivelul aplicatie sunt puse la dispozitie o serie de servicii :**
 - Terminal la distanta (TELNET, SSH, ...)
 - Posta electronica (SMTP, IMAP, POP, ...)
 - Transferul de fisiere (TFTP, FTP si altele)
 - World-Wide Web (HTTP)
 - Conversatii instantanee (ICQ, XMPP (din Mai 2014 -> nu mai are suport in Google Voice), Hangouts IM, WhatsApp...)
- **Se ofera si protocoale pentru rezolvarea unor sarcini de sistem - /etc/services, /etc/protocols**
 - Sistemul de fisiere in retea (NFS)
 - Conectivitatea cu alte sisteme de fisiere (SMB)
 - Servicii de baze de date (MySQL, PostgreSQL, ..., Hive,...)

Caracteristici de proiectare

- Tipuri de protocoale in functie de natura datelor transferate
 - Fluxuri de caractere generate de utilizator
 - Folosite pentru aplicatii interactive la distanta (*telnet, rlogin, IRC, ...*)
 - Traficul este in mare masura compus din date neinterpretate
 - Se pot include secvente de control (i.e. controlul terminalului, coduri de culoare) – coduri ANSI
(Exemplu: **CSI *n* E** -> numit: CNL – Cursor Next Line
Moves cursor to beginning of the *n*-th (default 1) following line)

Caracteristici de proiectare

- **Tipuri de protocoale in functie de natura datelor transferate**
 - **Mesaje intrebare/raspuns ASCII**
 - Serverul si clientul vehiculeaza siruri de caractere care pot fi citite si de utilizatori umani (SMTP, FTP, HTTP, XMPP, SIP, ...)
 - Uzual, sunt compuse din linii de text
 - **Formate binare**
 - Utilizate pentru protocoale de nivel inferior (SNMP – Simple Network Management Protocol) sau de nivel inalt (NFS peste RPC)
 - Apar probleme la reprezentarea datelor (i.e. ordinea octetilor)
 - **Protocoale ad-hoc folosite de aplicatiile (nestandard) scrise de utilizatori**
 - Pot adopta unele dintre tipurile anterioare

Caracteristici de proiectare

- Cerinte referitoare la proiectarea unui protocol
 - Parametri critici: lungimea numelui comenzilor, marimea *buffer*-elor, modul de adresare
 - Definirea operatiilor permise (e.g., creare, citire, scriere, stergere, actualizare)
 - Raportarea erorilor: coduri de eroare, mesaje
 - Formatul mesajelor: sursa, destinatie, parametri, codificarea datelor, lungime fixa/variabila, ...

Caracteristici de proiectare

- Scenariul uzual
 - Serverul – citește coduri de operații (*opcode*-uri) și raportează starea folosind coduri de eroare
 - Clientul – construiește mesaje folosind *opcode*-urile permise
- Moduri de adresare
 - Proces executat pe o singură mașină
adresa (fizică/logică) a mașinii: thor.info.uaic.ro
 - Procese executate pe mașini diferite:
 - Adrese formate din 2 părți (proces, mașină)
thor.info.uaic.ro:80
 - Adrese ca nr. generate aleatoriu (*universal ID*)
 - fiecare ID trebuie difuzat tuturor

Caracteristici de proiectare

- Problema fiabilitatii (engl. *Reliability*) comunicarii
 - Reteaua poate pierde mesaje
 - Abordari:
 - Posta clasica (*post-office*)
 - Nu asteapta nici un fel de confirmari
 - *Handshaking* – toate mesajele sunt confirmate
 - Raspuns confirmat (*acknowledged reply*)
 - Se asteapta un raspuns, iar expeditorul raspunsului asteapta confirmarea primirii lui
 - Cerere/raspuns (*request/reply*) – expeditorul asteapta (un timp) venirea raspunsului (e.g. RPC, SOAP)

Accesul la terminal

- Serviciu “antic” standard Internet
- Folosit prin comenzi precum *rlogin*, *telnet*, *ssh* (varianta securizata a *telnet*)
- Utilizeaza modelul client/server:
 - Clientul interactioneaza cu utilizatorul
 - Serverul furnizeaza acces la un *shell* (e.g., *bash*)



*Aplicatie de tip
remote login*

Accesul la terminal

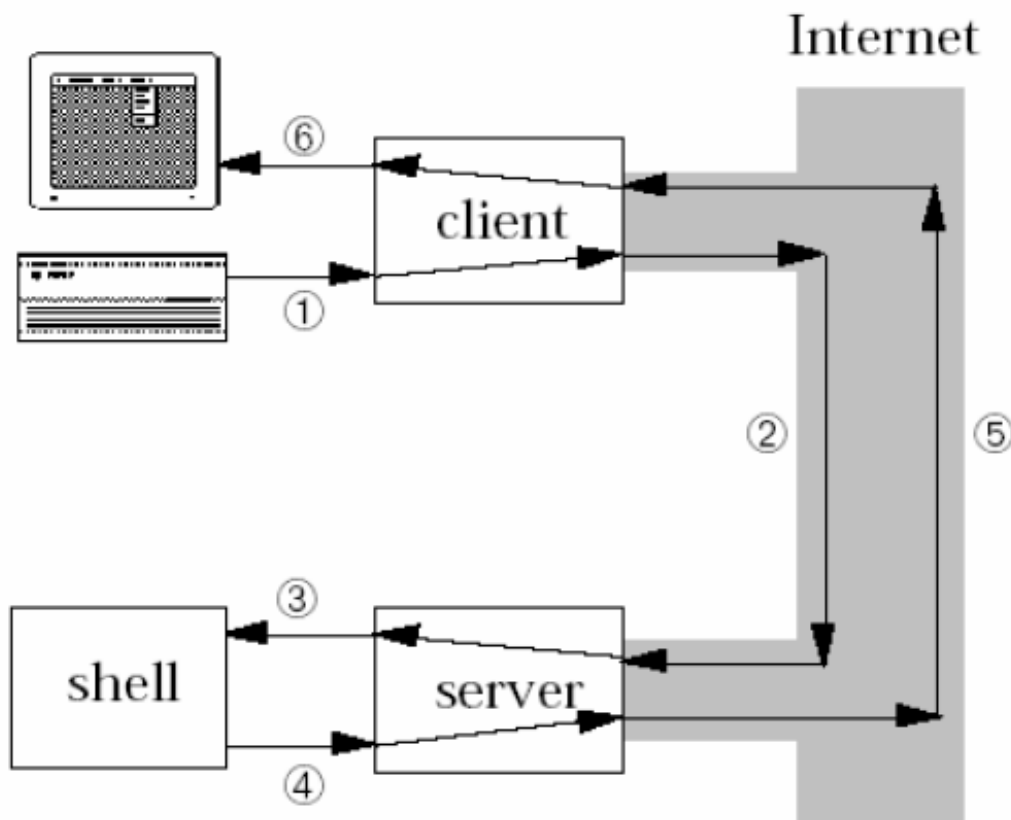


Figura: Mecanismul de functionare a unei aplicatii de tip *remote login*

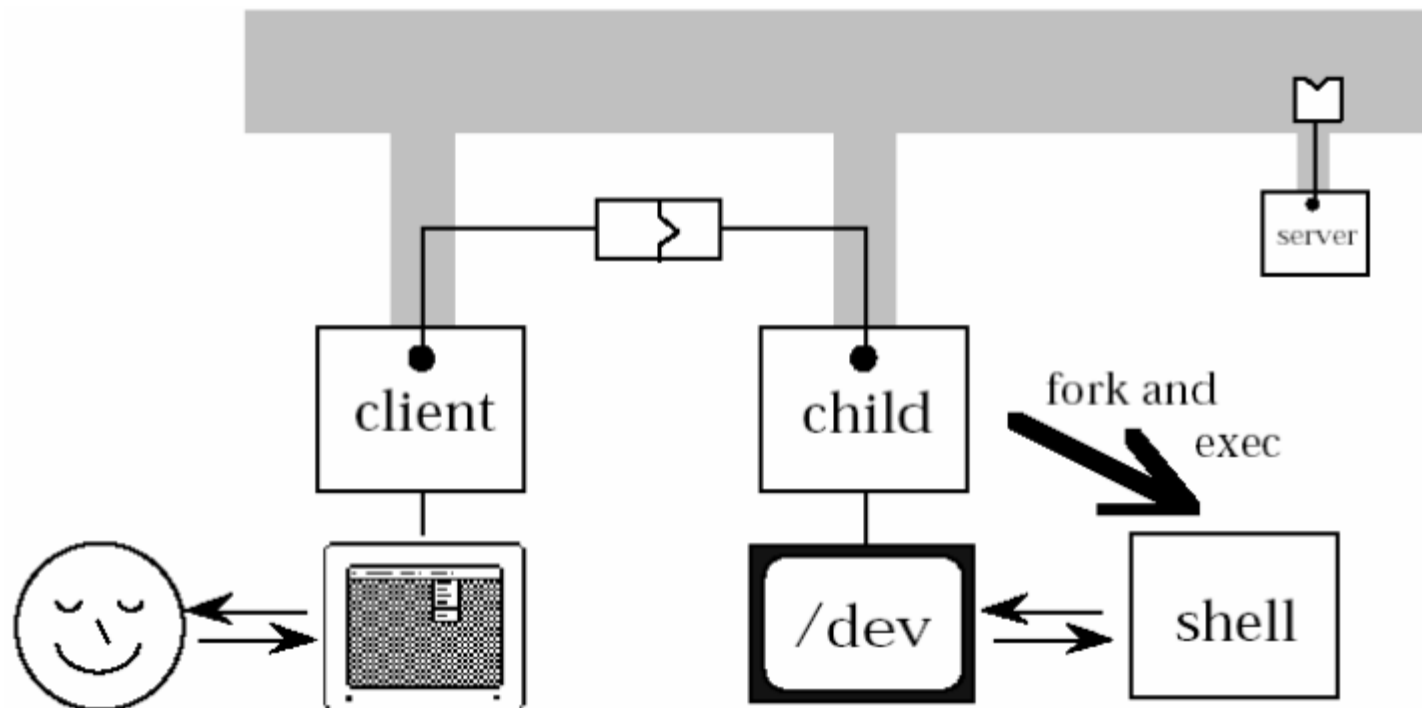
[Rețele de calculatoare –
curs 2007-2008, Sabin Buraga]

Accesul la terminal

- Implementare – mecanism general

Pentru conectarea la distanta, fiecare client va fi deservit de un proces copil al serverului

Procesul copil va crea un alt proces care va conecta clientul la un pseudo-terminal si va executa *shell*-ul



[Rețele de calculatoare –
curs 2007-2008, Sabin Buraga]

Accesul la terminal

- **Probleme**

- Initializarea si autentificarea
 - Cum identificam clientul?
 - Cum stim ca serverul este unul oficial?
- Procesarea caracterelor speciale (inclusiv sfarsitul de linie – EOL)
- Cine proceseaza actiuni precum editarea liniei, afisarea caracterelor testate (*echoing*), suspendarea terminalului (CTRL + S) etc.?
- Modul de comunicare intre client si server
 - Intreruperi din partea utilizatorului
 - Controlul dimensiunii ferestrei de afisare

Accesul la terminal

- **rlogin**

- protocol simplu de acces la distanta
- utilizat exclusiv pentru conectarea de masini UNIX
- RFC 1258: *“The rlogin facility provides a remote-echoed, locally flow-controlled virtual terminal with proper flushing of output”*

Functionare:

- **rlogin** comunica cu un *daemon* **rlogind** de pe gazda *remote*
- autentificarea se face prin apelarea la gazde “de incredere” (*“trusted” hosts*)
 - rlogind permite logarea fara parola daca gazda *remote* apare in fisierul `/etc/hosts.equiv` sau daca utilizatorul are un fisier `.rlogin` in directorul *home*

Accesul la terminal

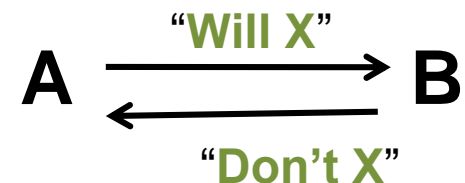
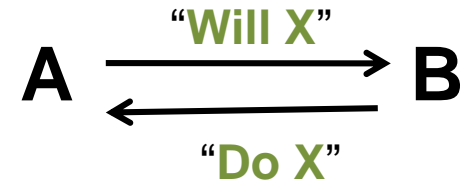
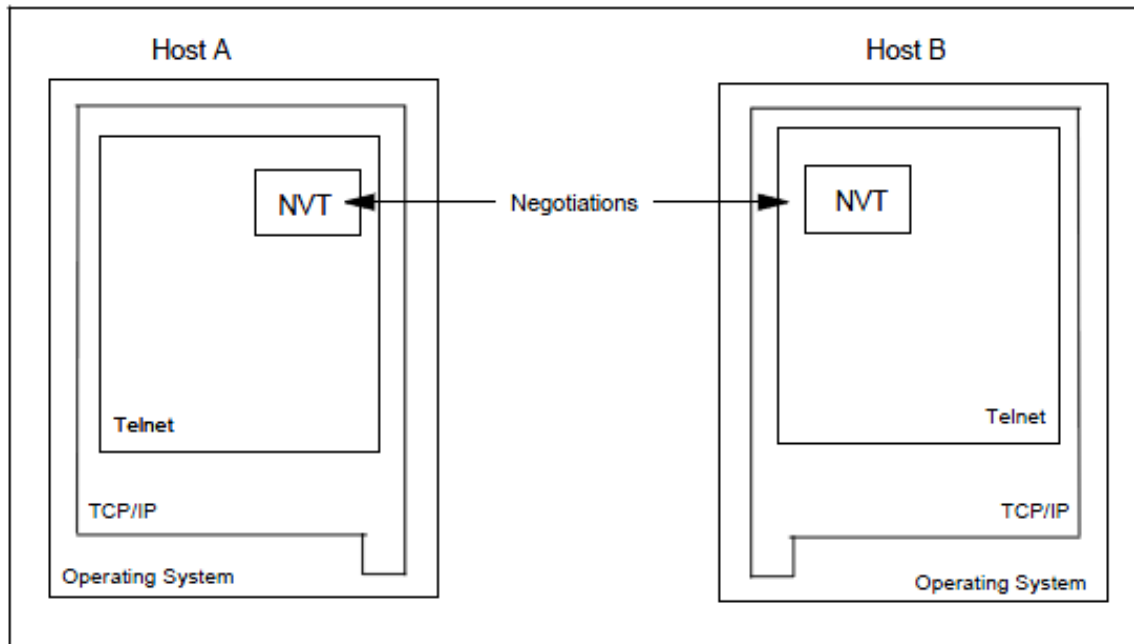
- **telnet** (*terminal network*)
 - Protocol standard TCP/IP de acces la distanta (RFC 854,855)
 - Utilizat indiferent de platforma
 - Poate fi utilizat drept client generic fara a sti detalii despre server
 - Autentificarea clientilor nu se face de catre protocol, ci de catre aplicatie
 - Protocolul se bazeaza pe:
 - Conceptul **NVT** (**Network Virtual Terminal**): un dispozitiv virtual cu o structura generala comuna cu o gama larga de terminale; fiecare host face maparea caracteristicilor propriului terminal cu cele ale NVT
 - Odata ce a fost stabilita o conexiune prin TELNET, ambele capete ale comunicarii sunt tratate simetric

Accesul la terminal

- **telnet** (*terminal network*)

- Protocolul se bazeaza pe:

- Ambele parti ale comunicarii pot sa negocieze utilizarea de optiuni aditionale care sa reflecte partea hardware utilizata
 - Optiuni pentru : editarea liniei, dimensiunea ferestrei de afisare etc.



telnet ofera compatibilitate cu terminale vechi (vt52, vt100,...)

[TCP/IP Tutorial and Technical Overview, IBM, 2006]

Accesul la terminal

- **telnet** (*terminal network*)
 - Comunicare dintre client si server se realizeaza prin comenzi de tipul:
 - **IP** (**Interrupt Process**; 244) -> terminarea programului care ruleaza
 - **AO** (**Abort output**; 245) -> elibereaza orice buffer de iesire
 - **AYT** (**Are you there**; 246) -> permite clientului trimiterea unei interogari OOB pentru verificarea faptului ca partea *remote* este activa
 - **EC** (**Erase character**; 247) -> stergerea caracterului anterior
 - **EL** (**Erase Line**; 248) -> stergerea intregii linii curente
 - ... (RFC 854)
 - **Trimiterea unei comenzi:** comanda (1 octet) precedata de un octet cu valoarea 255 - IAC (*Interpret As Command*)

SSH

- **SSH** (*secure shell*)
 - Fata de *telnet*, furnizeaza o comunicare sigura (bazata pe TCP) prin mesaje criptate si mesaje de autentificare
 - SSH foloseste modelul client/server
 - Un program client SSH este utilizat pentru stabilirea unei conexiuni cu un *daemon* SSH
 - Utilizari:
 - logarea pe o masina la distanta si executarea de comenzi
 - suport pentru *tunneling* (Curs viitor)
 - permite si transfer de fisiere in asociere cu protocoalele SFTP sau SCP
 - Are suport in majoritatea sistemelor de operare moderne

E-mail

- **Protocoloale bazate pe TCP:**
 - **SMTP** (*Simple Mail Transfer Protocol*)
 - RFC 821 (specifica modul de schimb a mail-ului intre doua host-uri)
 - **POP** (*Post Office Protocol*)
 - RFC 1939
 - **POP3S** – varianta securizata a **POP3**
 - A se vedea si: RFC 822 (specificatii privind antetul unui mail), 2049 (specificatii privind documente diferite *de plain text ASCII* ce pot fi continute intr-un email), RFC 974 (standard privind rutarea mailurilor folosind DNS)
 - RFC 822 si 974 -> consolidate in RFC 2821, 2822

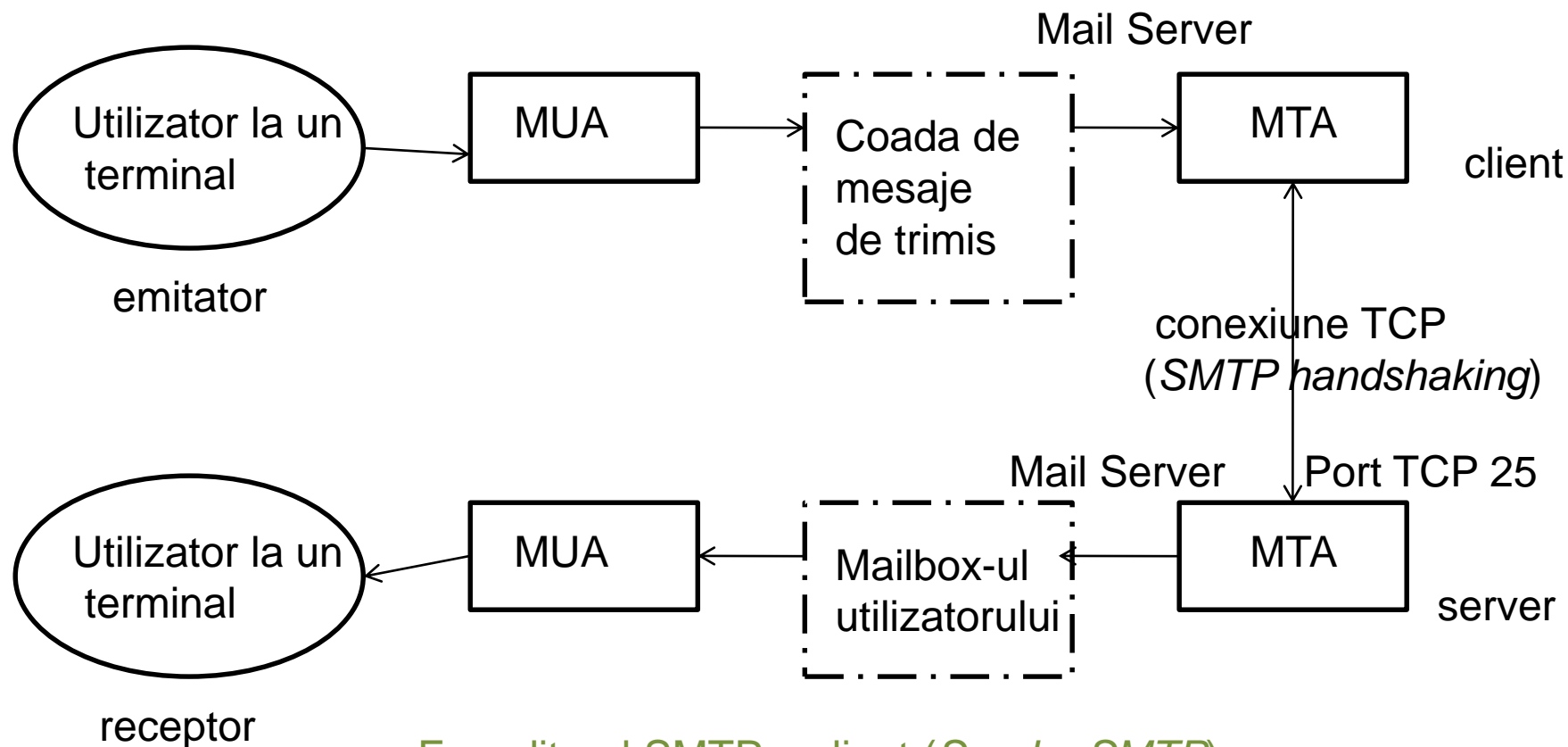
E-mail (*Electronic Mail*)

- Terminologie

- **Agent utilizator (MUA – Mail User Agent)**: client (local) pentru posta electronica
Ex: alpine, mutt, Mozilla Thunderbird, Kmail, Outlook etc.
- **Agent de transfer (MTA – Mail Transport Agent)**
responsabil cu comunicarea cu gazdele la distanta si cu trimiterea/receptionarea de posta
(client & server) - sendmail, qmail
- **Agent de distributie (MDA - Mail Distribution Agent sau LDA – Local Delivery Agent)** - directioneaza mesajele primite catre casuta postala a utilizatorului; Ex: maildrop, Sieve, procmail
- **Mail exchanger (MX)** – gazda responsabila cu e-mail-urile unui domeniu (masina intermediara)

E-mail | SMTP

- Utilizat in schimbul de mesaje de posta intre serverele de mail (MTA-uri)



Expeditorul SMTP = client (*Sender SMTP*)

Destinatarul SMTP = server (*Receiver SMTP*)

E-mail

- **Caracteristici**

- Distinctia dintre **plic** si **continut**

- **Plicul** incapsuleaza mesajul, contine date necesare pentru transportul mesajului: destinatar, adresa, prioritate, securitate, ...

- Plicul este folosit pentru dirijarea mesajului la destinatar

- **Mesajul** din plic contine un **antet** (date de control pentru MUA) si un **corp** (date pentru utilizator)

- Fiecare utilizator este identificat printr-o adresa de e-mail:
cutie_postala@locatie (cont@adresaInternet)

E-mail | SMTP

- **Componente:**

- **Plic** (*envelope*) – folosit de MTA pentru livrare

Exemplu:

MAIL From: <adria@info.uaic.ro>

RCPT to: <adria@info.uaic.ro>

- **Anteturi** (*headers*) – folositi de MUA

Exemplu: Received, Message-ID, From, Date, Reply-To,
Subject,...

- **Continut** –ul mesajului (*body*) -

- Mecanism: MUA preia continutul, adauga anteturi si il transmite la MTA; MTA adauga anteturi, adauga plicul si il trimite la un alt MTA

E-mail | SMTP

- Campuri de antet utilizate in transportul de e-mail-uri:

Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

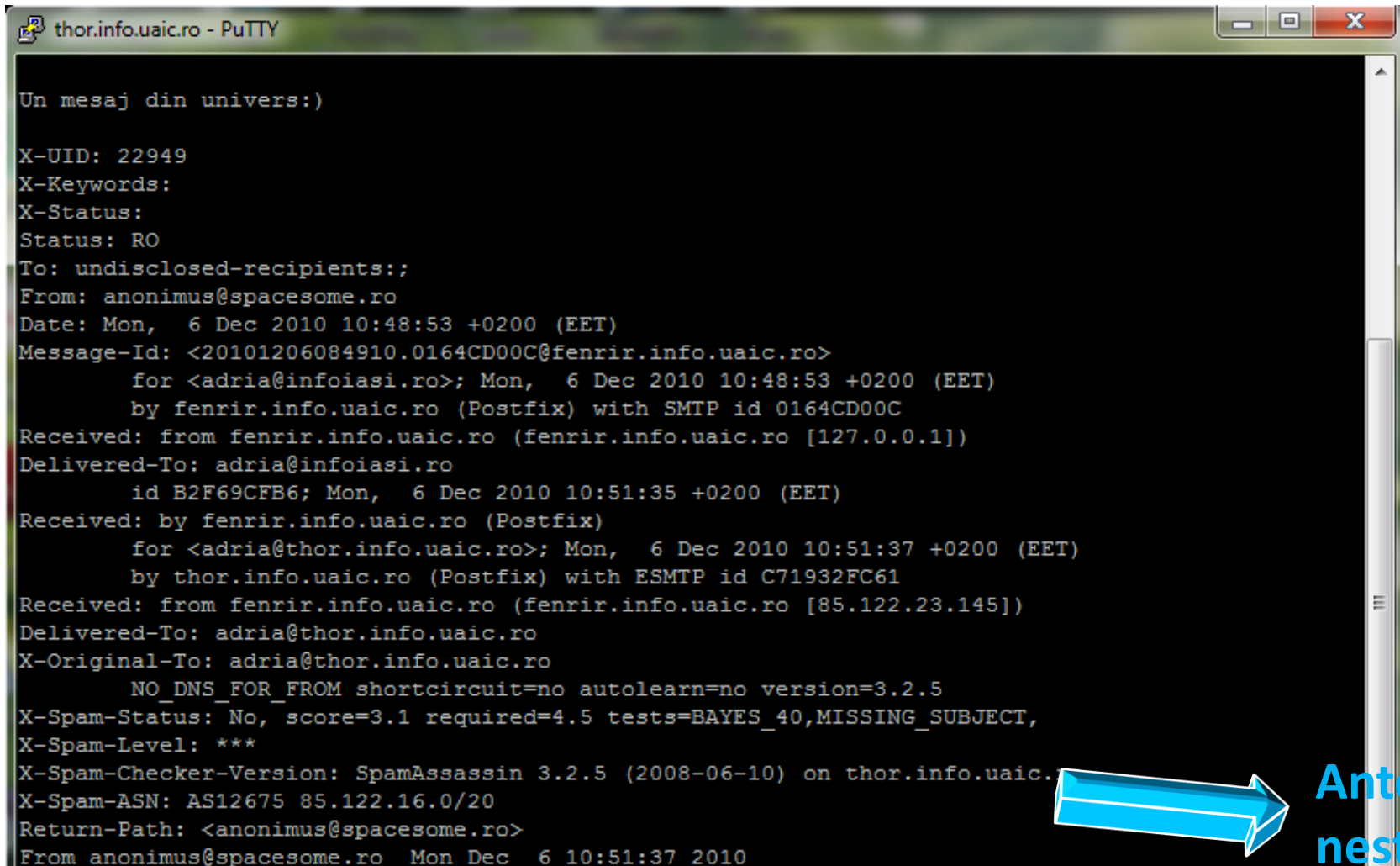
E-mail | SMTP

- Campuri de antet utilizate in transportul de e-mail-uri:

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	Email address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User chosen keywords
Subject:	Short summary of the message for the one-line display

E-mail | SMTP

Exemplu



The image shows a PuTTY terminal window titled 'thor.info.uaic.ro - PuTTY'. The terminal displays the header of an email message. The text is as follows:

```
Un mesaj din univers:)
X-UID: 22949
X-Keywords:
X-Status:
Status: RO
To: undisclosed-recipients;;
From: anonimus@spacesome.ro
Date: Mon, 6 Dec 2010 10:48:53 +0200 (EET)
Message-Id: <20101206084910.0164CD00C@fenrir.info.uaic.ro>
    for <adria@infoiasi.ro>; Mon, 6 Dec 2010 10:48:53 +0200 (EET)
    by fenrir.info.uaic.ro (Postfix) with SMTP id 0164CD00C
Received: from fenrir.info.uaic.ro (fenrir.info.uaic.ro [127.0.0.1])
Delivered-To: adria@infoiasi.ro
    id B2F69CFB6; Mon, 6 Dec 2010 10:51:35 +0200 (EET)
Received: by fenrir.info.uaic.ro (Postfix)
    for <adria@thor.info.uaic.ro>; Mon, 6 Dec 2010 10:51:37 +0200 (EET)
    by thor.info.uaic.ro (Postfix) with ESMTP id C71932FC61
Received: from fenrir.info.uaic.ro (fenrir.info.uaic.ro [85.122.23.145])
Delivered-To: adria@thor.info.uaic.ro
X-Original-To: adria@thor.info.uaic.ro
    NO_DNS_FOR_FROM shortcircuit=no autolearn=no version=3.2.5
X-Spam-Status: No, score=3.1 required=4.5 tests=BAYES_40,MISSING_SUBJECT,
X-Spam-Level: ***
X-Spam-Checker-Version: SpamAssassin 3.2.5 (2008-06-10) on thor.info.uaic.
X-Spam-ASN: AS12675 85.122.16.0/20
Return-Path: <anonimus@spacesome.ro>
From anonimus@spacesome.ro Mon Dec 6 10:51:37 2010
```



Anteturi
nestandard

E-mail | SMTP

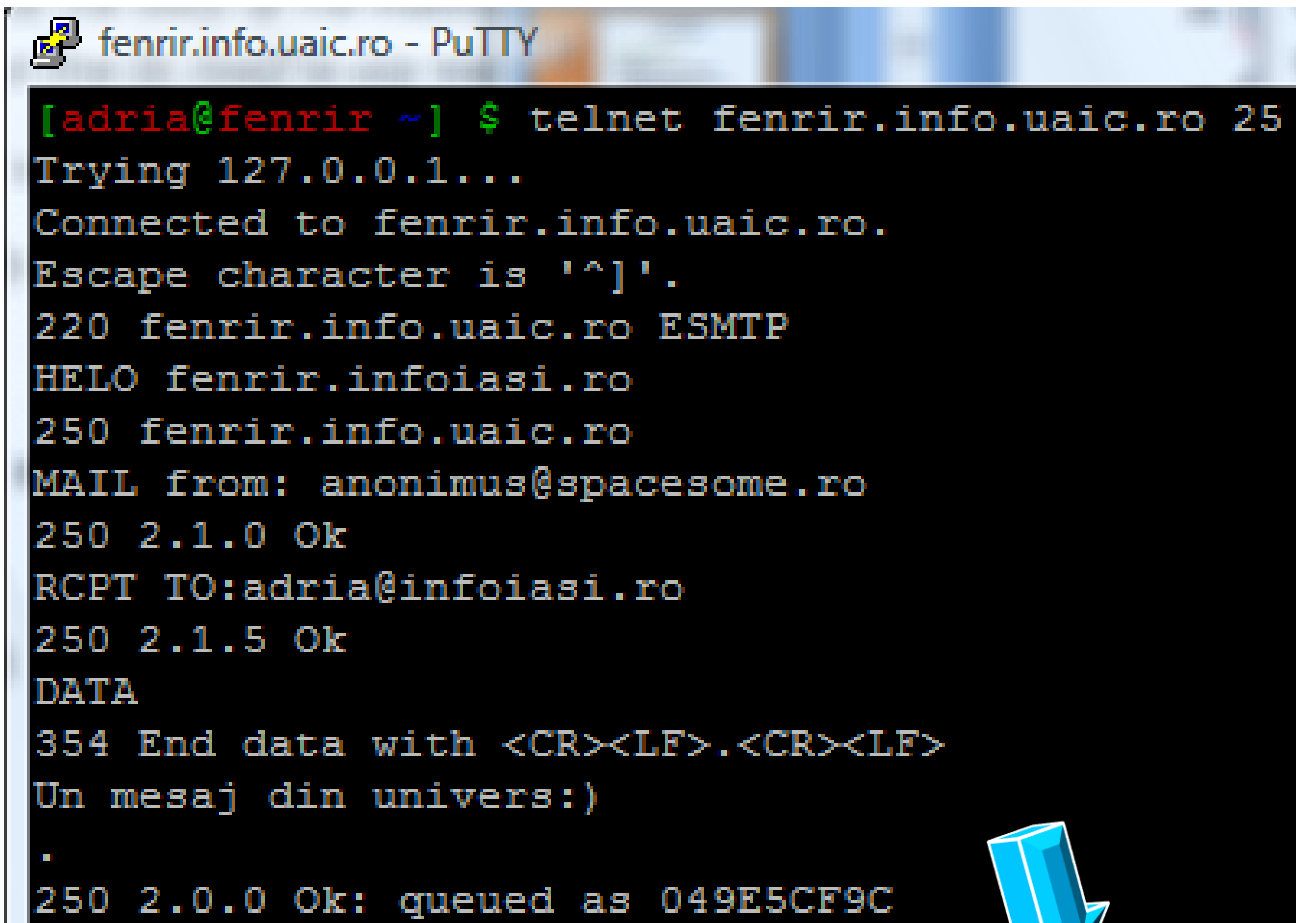
- Comunicarea:
 - Se realizeaza o conexiune TCP intre *Sender SMTP* si *Receiver SMTP* (intre MTA-uri). Obs. Receiver SMTP poate fi destinatia finala sau un intermediar (*mail gateway*)
 - Clientul trimite comenzi SMTP, iar serverul raspunde cu coduri de stare
 - Mesajele de stare include coduri numerice NNN si texte explicative
 - Ordinea comenzilor este importanta
 - Se utilizeaza portul 25

E-mail | SMTP

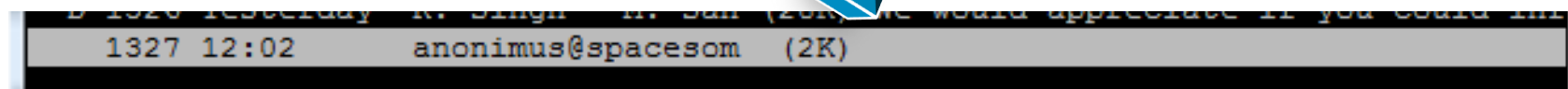
- Comenzi uzuale:
 - **HELO**: identifica gazda expeditoare
 - **MAIL FROM**: porneste o tranzactie si identifica originea e-mail-ului
 - **RCPT TO**: identifica recipientii individuali ai mesajului (adrese de e-mail); pot exista comenzi **RCPT TO**: multiple
 - **DATA** desemneaza o serie de linii text terminate cu \r\n, ultima linie continind doar “.”
 - **QUIT**

E-mail | SMTP

- Exemplu:



```
fenrir.info.uaic.ro - PuTTY
[adria@fenrir ~] $ telnet fenrir.info.uaic.ro 25
Trying 127.0.0.1...
Connected to fenrir.info.uaic.ro.
Escape character is '^]'.
220 fenrir.info.uaic.ro ESMTP
HELO fenrir.infoiasi.ro
250 fenrir.info.uaic.ro
MAIL from: anonimus@spacesome.ro
250 2.1.0 Ok
RCPT TO:adria@infoiasi.ro
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Un mesaj din univers:)
.
250 2.0.0 Ok: queued as 049E5CF9C
```



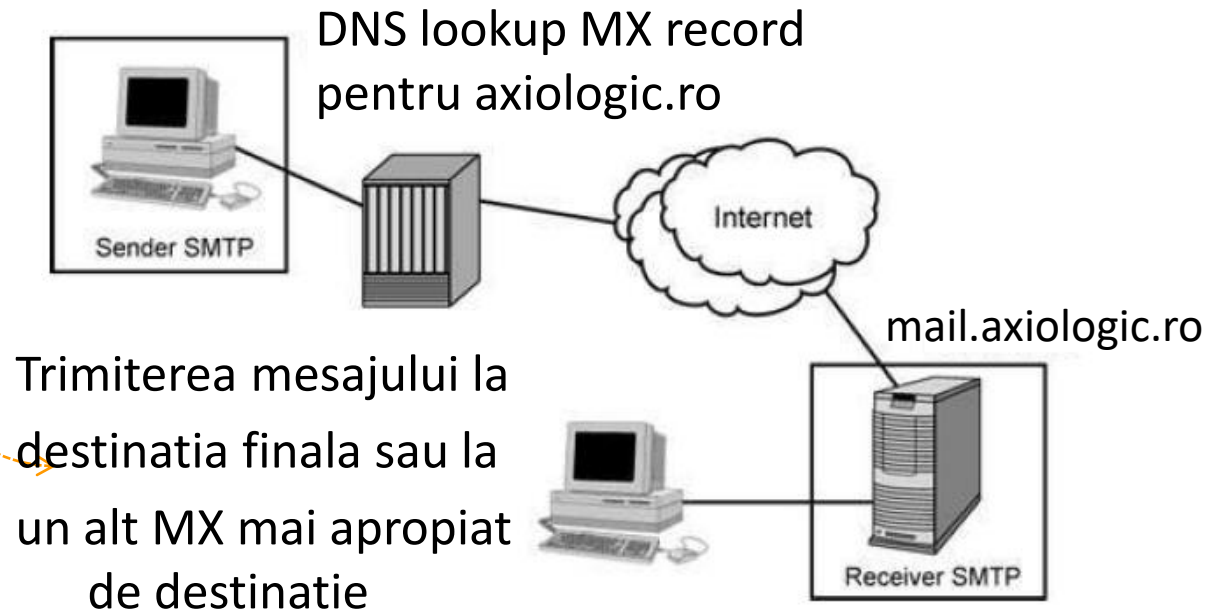
E-mail | SMTP

- Alte comenzi:
 - **VRIFY**: permite verificarea validitatii unui *recipient*
 - **NOOP**: forteaza serverul sa raspunda cu un cod de OK (200)
 - **EXPN**: expandeaza un grup de adrese (*alias*)
 - **TURN**: interschimba destinatarul cu expeditorul fara a fi necesara crearea unei noi conexiuni TCP
(*sendmail* nu suporta aceasta comanda)
 - **RSET** abandoneaza tranzactia curenta

E-mail | SMTP

- DNS si e-mail-ul

- Inregistrarea de tip MX din DNS identifica gazda (MX) cu rol de procesare si *forward*-are a mailurilor pentru respectivul domeniu



- Mecanism general:

- Serverul SMTP verifica inregistrarea MX a domeniului specificat in adresa de email (e.g. axiologic.ro pentru adresa b@axiologic.ro) si sa zicem ca aceasta inregistrare este mail.axiologic.ro.
- Se va trimite acest mail pe serverul SMTP de pe mail.axiologic.ro .

E-mail | SMTP

- RFC 822: SMTP este limitat la text ASCII pe 7 biti
- RFC 1521: defineste un standard care sa rezolve limitarile anterioare -> MIME (*Multipurpose Internet Mail Extensions*)
 - Standard de codificare a continutului mesajelor non-ASCII
 - Limbi cu accente, cu alfabet non-latine, fara alfabet, mesaje non-textuale
 - Permite atasarea la e-mail a fisierelor de orice tip
 - Se foloseste campul:

Content-Type: tip/subtip

Exemplu: Mime-Version: 1.0

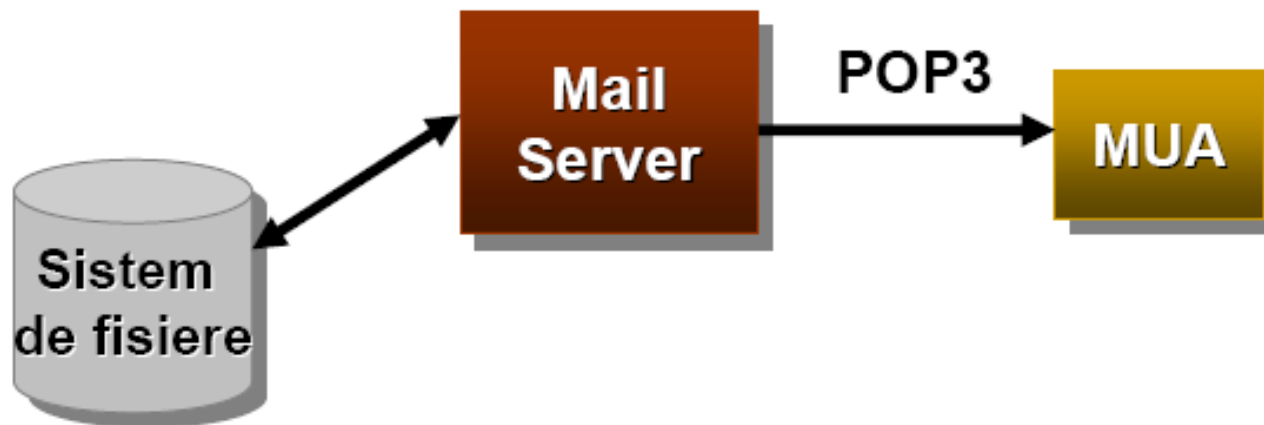
Content-Type: TEXT/PLAIN

E-mail | SMTP

- Tipuri MIME principale:
 - application** defineste aplicatiile client
(application/executable)
 - text** defineste formatele text
(text/plain, text/html)
 - image** specifica formatele grafice
(image/gif, image/jpeg)
 - audio** specifica formatele audio (audio/basic)
 - video** specifica formatele video (video/mpeg)
 - multipart** utilizat pentru transportul datelor compuse
(multipart/mixed, multipart/alternative)

E-mail | POP

- POP (Post Office Protocol) – RFC 1939
- Utilizat la transferul de mesaje de pe un server de posta la un MUA – portul 110
- Comenzile si raspunsurile sunt mesaje ASCII
- Raspunsurile incep cu +OK sau -ERR



[Rețele de calculatoare –
curs 2007-2008, Sabin Buraga]

E-mail | POP

- Comenzi uzuale:
 - **USER** specifica numele de cont
 - **PASS** specifica parola
 - **STAT** furnizeaza numarul de mesaje din cutia postala (*mailbox*)
 - **LIST** afiseaza lista de mesaje si lungimea, cate 1 pe linie
 - **RETR** preia un mesaj
 - **DELE** reseteaza tranzactia, iar orice marcaj de stergere este eliminat
 - **QUIT** sterge mesajele marcate si inchide conexiunea

```
adria@thor:~$ telnet thor 110
Trying 85.122.23.1...
Connected to thor.info.uaic.ro.
Escape character is '^]'.
+OK POP3 thor.info.uaic.ro 2007b.104 server ready
user adria
+OK User name accepted, password please
pass [REDACTED]
+OK Mailbox open, 1108 messages
stat
+OK 1108 194519602
retr 1108
+OK 1115 octets
Return-Path: <anonymus@spacesome.ro>
X-Spam-ASN: AS12675 85.122.16.0/20
X-Spam-Checker-Version: SpamAssassin 3.2.5 (2008-06-10) on thor.info.uaic.ro
X-Spam-Level: ***
X-Spam-Status: No, score=3.1 required=4.5 tests=BAYES_40,MISSING_SUBJECT,
NO_DNS_FOR_FROM shortcircuit=no autolearn=no version=3.2.5
X-Original-To: adria@thor.info.uaic.ro
Delivered-To: adria@thor.info.uaic.ro
Received: from fenrir.info.uaic.ro (fenrir.info.uaic.ro [85.122.23.145])
    by thor.info.uaic.ro (Postfix) with ESMTP id C71932FC61
    for <adria@thor.info.uaic.ro>; Mon,  6 Dec 2010 10:51:37 +0200 (EET)
Received: by fenrir.info.uaic.ro (Postfix)
    id B2F69CFB6; Mon,  6 Dec 2010 10:51:35 +0200 (EET)
Delivered-To: adria@infoiasi.ro
Received: from fenrir.info.uaic.ro (fenrir.info.uaic.ro [127.0.0.1])
    by fenrir.info.uaic.ro (Postfix) with SMTP id 0164CD00C
    for <adria@infoiasi.ro>; Mon,  6 Dec 2010 10:48:53 +0200 (EET)
Message-Id: <20101206084910.0164CD00C@fenrir.info.uaic.ro>
Date: Mon,  6 Dec 2010 10:48:53 +0200 (EET)
From: anonymus@spacesome.ro
To: undisclosed-recipients:;
Status: O

Un mesaj din univers:)
```

E-mail | POP

Exemplu

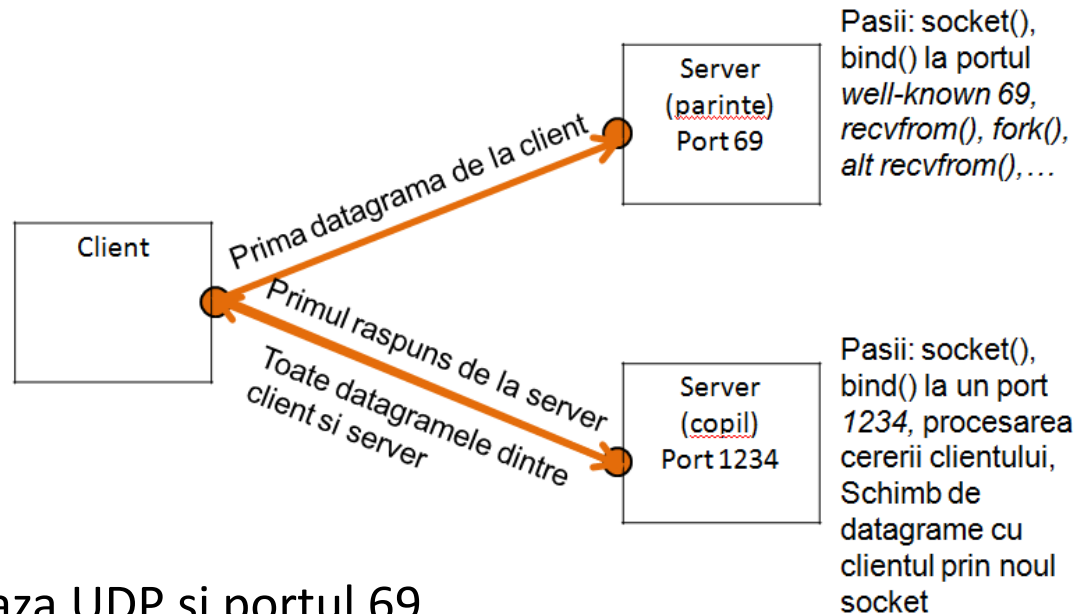
Parola necriptata

E-mail | POP

- POP 3 - caracteristici:
 - In general, daca utilizatorul schimba clientul el nu-si mai poate reciti mailurile; Obs: Clienti cu optiunea: *'keep a copy of the email on the server'*
 - Foloseste mecanismul *"download-and-keep"*: copierea mesajelor pe clienti diferiti
 - POP3 este fara stare intre sesiuni
- Alte solutii:
 - IMAP (Interactive Mail Access Protocol) – RFC 1730
 - Pastreaza toate mesajele intr-un singur loc: pe server
 - Permite utilizatorului sa organizeze mesajele in directoare
 - Pastreaza starea "utilizatorului" intre sesiuni
 - Numele directoarelor si maparea dintre ID-urile mesajelor si numele folderului

Transferul de fisiere | TFTP

- TFTP (Trivial File Transfer Protocol) -> ...Cursul 6 & RFC 1350



- utilizeaza UDP si portul 69
- utilizat deseori la initializarea statiilor de lucru fara disc sau a altor dispozitive
- nu are mecanisme de autentificare si criptare => este utilizat in retele locale
- RFC 1785, 2347, 2348, 2349

Transferul de fisiere | TFTP

- TFTP (Trivial File Transfer Protocol)

Implementarile TFTP utilizeaza comenzi de tipul:

<code>Connect <host></code>	Specifies the destination host ID.
<code>Mode <ascii binary></code>	Specifies the type of transfer mode.
<code>Get <remote filename> [<local filename>]</code>	Retrieves a file.
<code>Put <remote filename> [<local filename>]</code>	Stores a file.
<code>Verbose</code>	Toggles verbose mode, which displays additional information during file transfer, on or off.
<code>Quit</code>	Exits TFTP.

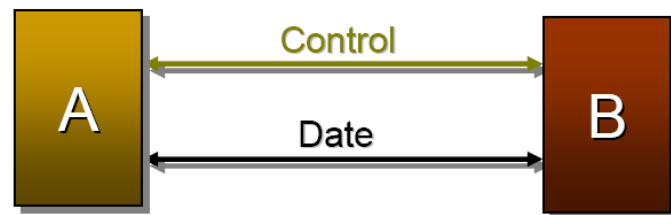
– RFC 1785, 2347, 2348, 2349

[TCP/IP Tutorial and Technical
Overview, IBM, 2006]

Transferul de fisiere | FTP

FTP – caracterizare

- Folosit atat interactiv, cat si de programe
- Asigura transferul sigur si eficient al fisierelor
- Se bazeaza pe modelul client/server
- FTP utilizeaza doua conexiuni TCP pentru transferul fisierelor:
 - **Conexiune de control**
 - folosita pentru trimiterea comenzilor si receptionarea codurilor de stare
 - Conexiunea de control utilizeaza portul 21
 - **Conexiunea de date**
 - folosita pentru transferul efectiv
 - conexiunea de date foloseste portul 20 sau unul aleator ($P > 1023$)
 - nu este obligatorie intr-o sesiune FTP



Transferul de fisiere | FTP

FTP – caracterizare

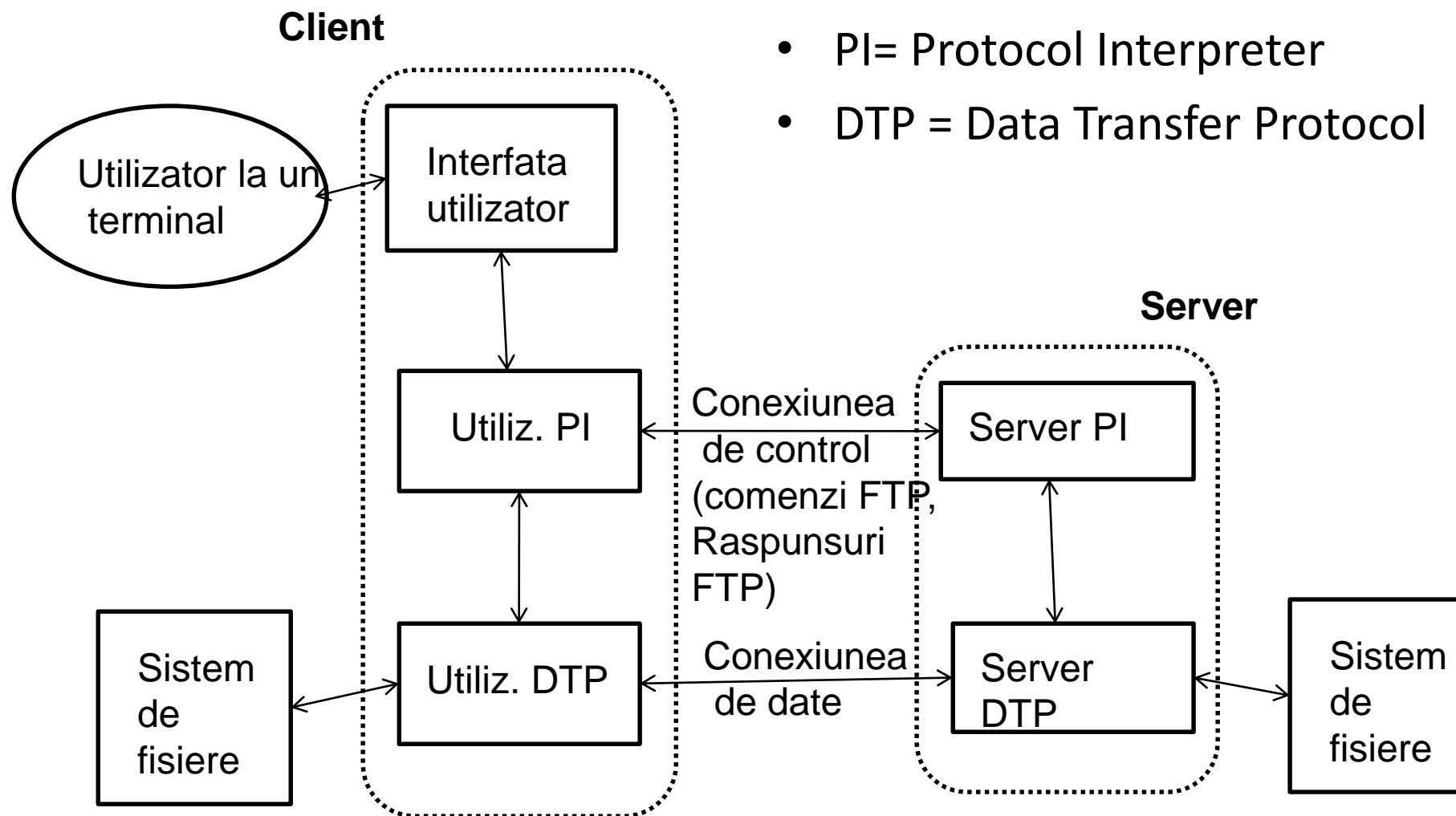
- Comenzile si raspunsurile sunt linii de text
- Obs. (FTP->)file transfer != file access (->NFS)
- Vezi si RFC 956, 1068, 2228, (FTP Security Extensions), 2428 (FTP Extensions for IPv6 and NATs)
- Pentru interactivitate se foloseste protocolul TELNET

Tipuri de acces:

- **Anonim (FTP anonymous)** – RFC 1635
 - Autentificare cu numele *anonymous* si drept parola o adresa de e-mail
 - Acces public la o serie de resurse (aplicatii, date, multimedia etc.)
- **Autentificat**
 - Necesita un nume de utilizator existent, insotit de o parola valida
 - Pentru transferul de date in/din contul personal

Transferul de fisiere | FTP

FTP- model



Transferul de fisiere | FTP

FTP – comenzi (client)

```
adria@thor:~$ ftp
ftp> help
Commands may be abbreviated.  Commands are:

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

Transferul de fisiere | FTP

FTP – comenzi uzuale (client)

Command	Meaning
Open	Create an FTP connection between the two hosts.
Close	Close an FTP connection between two hosts.
Bye	End the FTP session.
Get	Retrieve a remote file from the remote host.
Put	Store a file on the remote host.
Mget	Get multiple files using wildcards (for example, mget a* fetches all files that begin with the letter "a" in the current directory).
Mput	Put multiple files on the remote host using wildcards.
Glob	Enable wildcard interpretation. This is usually on by default.
Ascii	The file transferred is in ASCII representation (a common default).
Binary	The file is in image (binary) format (sometimes the default), and is useful for programs and formatted word processing files.
Cd	Change the directory on the remote host.
Dir	Get a directory listing from the remote host.
Ldir	Get a directory listing from the local host.
Hash	Display hash marks (dots) to show file transfer progress.

→ **RETR** (*retrive*)

→ **STOR** (*store*)

Transferul de fisiere | FTP

FTP – comenzi (protocol)

- Comenzi de control al accesului
 - **USER** *username*, **PASS** *password*, **QUIT**, **ChangeWorkingDir**,...
- Comenzi de transfer a parametrilor
 - **PORT**, **TYPE**, **MODE**
- Comenzi de realizarea a serviciilor FTP
 - **RETR** *filename*, **ABOR**, **STOR** *filename*, **LIST**, **PrintWorkingDir**

Raspunsul de stare

Linie de text continind:

XYZ un cod de stare (utilizat de software) + un mesaj explicativ (destinat oamenilor)

Transferul de fisiere | FTP

FTP – codul de stare (xyz)

Prima cifra semnifica:

- 1** replica pozitiva preliminara (“am indeplinit, dar asteapta”)
- 2** replica pozitiva finala (“succes”)
- 3** replica pozitiva intermediara (“am nevoie si de alte informatii”)
- 4** replica negativa tranzitorie (“eroare, incerc iar”)
- 5** replica negativa finala (“eroare fatala”)

Transferul de fisiere | FTP

FTP – codul de stare (xyz)

A doua cifra specifica grupuri de functii:

- 0 erori de sintaxa
- 1 informare (ajutor, informatii de stare)
- 2 referitor la conexiuni
- 3 privitor la autentificarea utilizatorului
- 4 nespecificat
- 5 referitor la sistemul de fisiere

Transferul de fisiere | FTP

FTP – codul de stare (xyz)

A treia cifra da informatii suplimentare asupra mesajelor de eroare

Exemple:

125 Conexiune deschisa; transfer pornit

200 Comanda OK

226 Transfer complet

331 Nume utilizator OK, se cere parola

452 Eroare la scrierea fisierului

500 Eroare de sintaxa (comanda necunoscuta)

501 Eroare sintaxa (argumente invalide)

221 Goodbye /*rezultat al comenzii QUIT */

Transferul de fisiere | FTP

FTP – Moduri de transfer

- **STREAM**

- Fisierul este trimis ca un flux de octeti; sfirsitul transmisiei este indicat de inchiderea normala a conexiunii;

- **BLOCK**

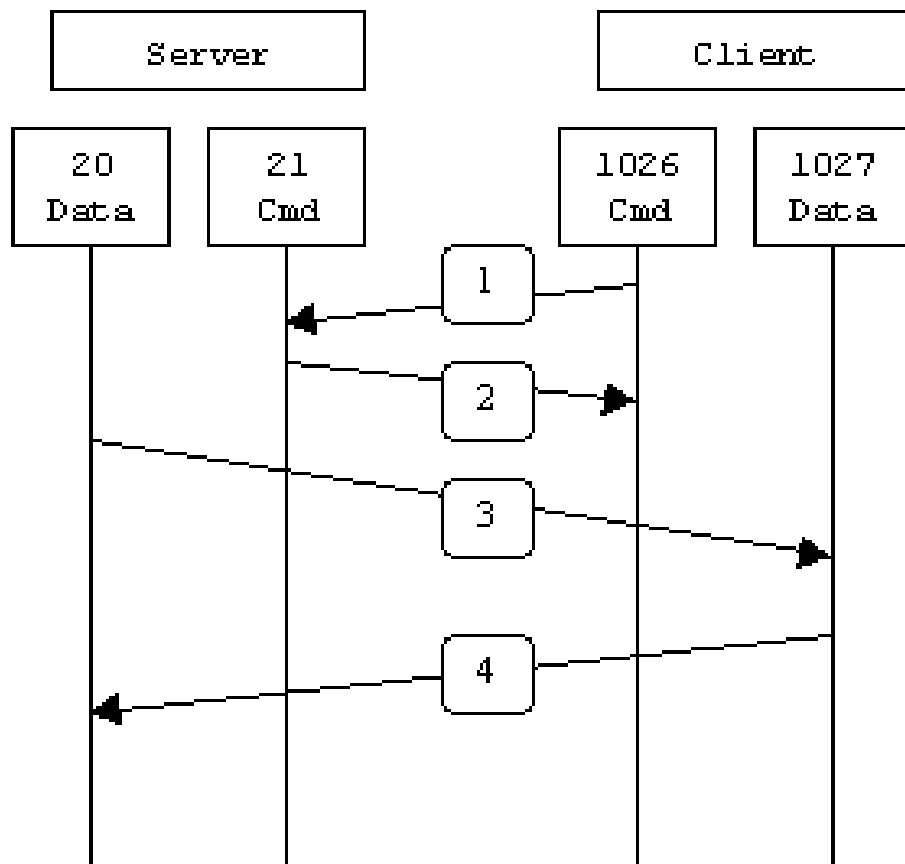
- Fisierul este transmis ca o serie de blocuri de date precedate de antete continand contoare si descriptori de bloc (e.g. End of data block)

- **COMPRESSED**

- Fisierul sunt compresate, conform unui algoritm de compresare (e.g., gzip) si sunt trimise ca date binare

Transferul de fisiere | FTP

Active FTP – exemplu

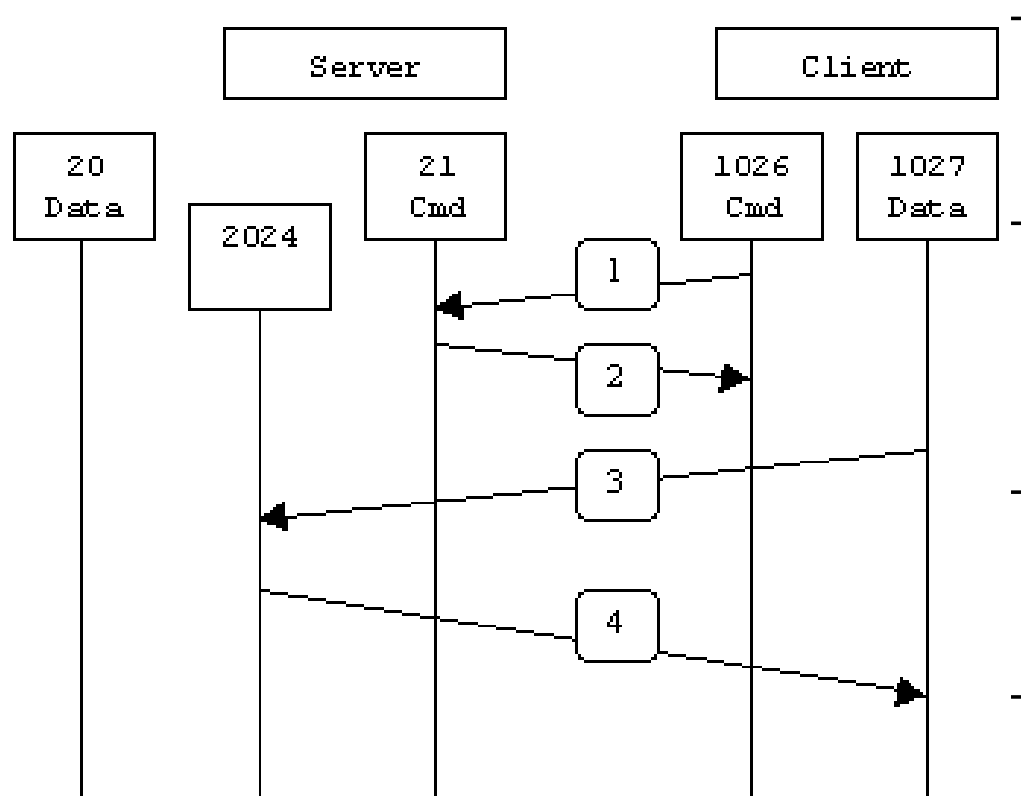


- Clientul se conectează la un server (85.122.23.145:**21**) de la un port $P > 1023$
- Clientul trimite comanda **PORT 85.122.23.1.4.2** ($4 \cdot 256 + 2 = \mathbf{1026}$) ce indica Server-ului să inițieze o conexiune cu clientul la portul $P+1$
- Clientul ascultă la $P+1$ și primește datele trimise de server prin portul **20**

Obs. Conexiunea inițiată de server poate fi interpretată ca un potențial atac de firewall-ul clientului

Transferul de fisiere | FTP

Pasive FTP – exemplu



- La initierea unei conexiuni FTP clientul foloseste doua porturi ($P > 1023$ si $P+1$)
- Clientul se conecteaza la un server (85.122.23.145:21) de la portul P si trimite comanda PASV
- Serverul deschide un port $P_s > 1023$ si trimite comanda PORT P_s clientului
- Clientul va initia o conexiune (de la portul $P+1$) cu serverul folosind portul primit (P_s)

HTTP

- **Hyper Text Transfer Protocol**

- Protocol utilizat in Internet, bazat pe stiva TCP/IP
- Sta la baza comunicarii dintre serverele si clientii Web
 - Client: in mod uzual poate fi un browser
 - Server: server Web care trimite raspunsuri la cererile primite
- HTTP 1.0 - RFC 1945
- HTTP 1.1 - RFC 2616
- HTTP 1.1 revised - RFC 723X (<https://www.w3.org/Protocols/>)

Protocolul HTTPS – asigura comunicatii “sigure” HTTP via TLS (Transport Layer Security):

- autentificare pe baza certificatelor digitale + criptare bidirectionala
- RFC 2818 – <https://tools.ietf.org/html/rfc2818>

HTTP

- Protocolul SPDY – un experiment Google, disponibil ca Internet Draft la care Google a renunat in 2016
 - Reducerea latentei incarcarii si cresterea securitatii
 - <https://www.chromium.org/spdy>
 - Implementari SPDY existau in : Chrome, Mozilla Firefox, Opera, Amazon Silk, Internet Explorer
- **Protocolul HTTP/2.0**
 - RFC 7540
 - Extinde ideile SPDY, focalizat asupra performantei
 - www.slideshare.net/mnot/what-http20-will-do-for-you

HTTP

- **Hyper Text Transfer Protocol**

Mecanism general:

Clientul initiaza o conexiune TCP cu serverul folosind portul 80

Serverul accepta conexiunea TCP

Are loc schimbul de mesaje HTTP intre clientul HTTP (browser) si server-ul Web

Se inchide conexiunea TCP

HTTP

- HTTP – nu se ocupa de partea de rutare sau verificarea cererilor
 - ? Cine: TCP&IP
 - HTTP lucreaza cu cereri la nivel inalt: *Fetch IndexPage* al <https://www.google.com>
 - *Live HTTP Headers* (Firefox) ->

http://127.0.0.1:8080/Curs_ProgramareJS/test.html

GET /Curs_ProgramareJS/test.html HTTP/1.1

Host: 127.0.0.1:8080

User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:23.0) Gecko/20100101 Firefox/23.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive

HTTP/1.1 200 OK

Date: Thu, 05 Sep 2013 15:57:00 GMT

Server: Apache/2.4.3 (Win32) OpenSSL/1.0.1c PHP/5.4.7

Last-Modified: Thu, 05 Sep 2013 15:51:41 GMT

Etag: "95-4e5a4e5efb9d1"

Accept-Ranges: bytes

Content-Length: 149

Keep-Alive: timeout=5, max=100

Connection: Keep-Alive

Content-Type: text/html

Detalii asupra portului: https://www.grc.com/port_8080.htm

http://127.0.0.1:8080/Curs_ProgramareJS/imgsrc2.jpg

GET /Curs_ProgramareJS/imgsrc2.jpg HTTP/1.1

Host: 127.0.0.1:8080

User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:23.0) Gecko/20100101 Firefox/23.0

Accept: image/png,image/*;q=0.8,*/*;q=0.5

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Referer: http://127.0.0.1:8080/Curs_ProgramareJS/test.html

Connection: keep-alive

HTTP/1.1 200 OK

Date: Thu, 05 Sep 2013 15:57:00 GMT

Server: Apache/2.4.3 (Win32) OpenSSL/1.0.1c PHP/5.4.7

Last-Modified: Wed, 28 Aug 2013 06:23:31 GMT

Etag: "3c9e-4e4fc0742da40"

Accept-Ranges: bytes

Content-Length: 15518

Keep-Alive: timeout=5, max=99

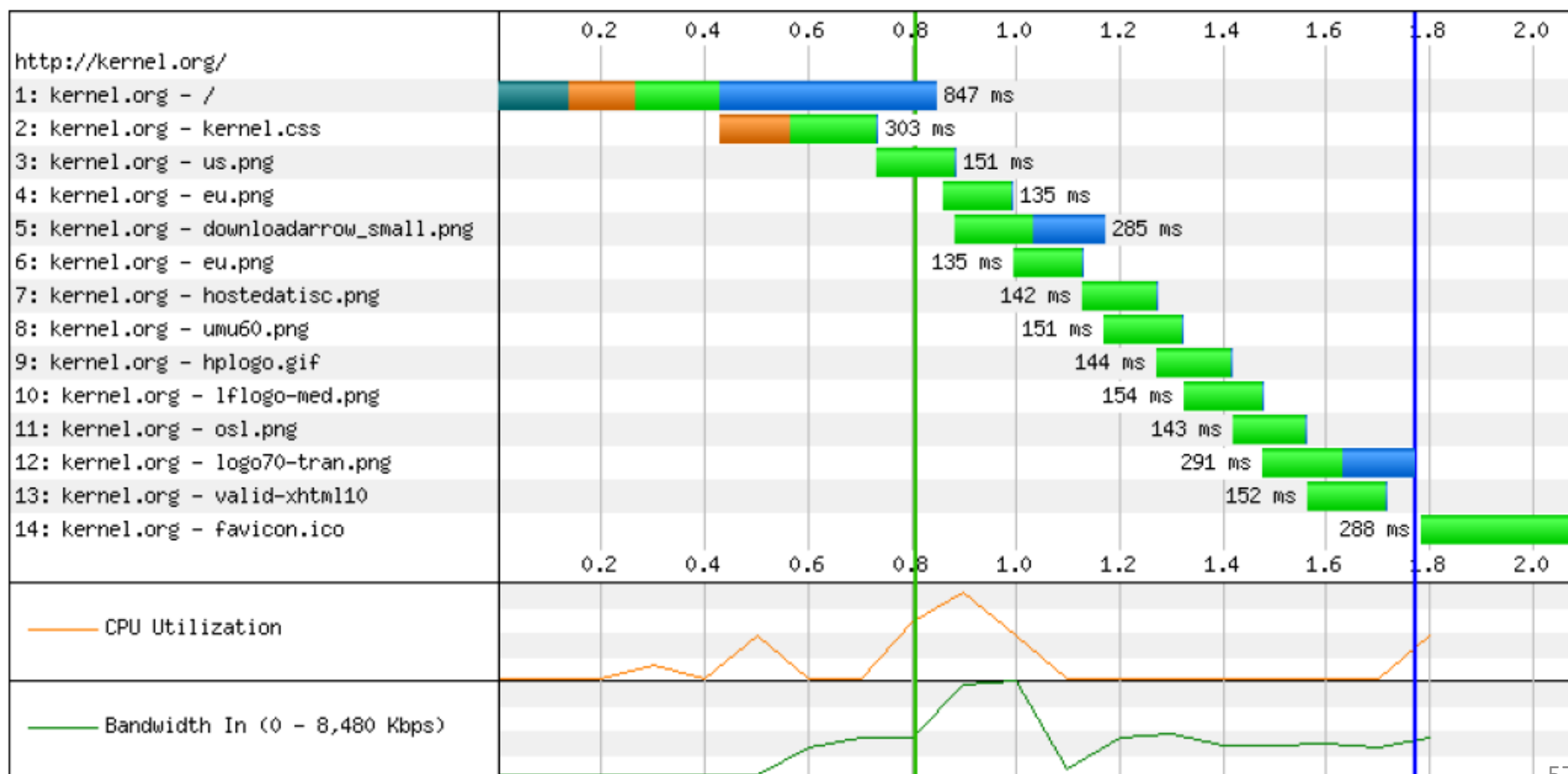
Connection: Keep-Alive

Content-Type: image/jpeg

Web Browser | Download

- Modul cum browserul reda un website este esential in procesul de optimizare

Exemplu: download resursa <http://kernel.org>



Web Browser | Download

- Pasii realizati de browser:
 - Rezolvarea *kernel.org* folosind DNS pentru aflarea IP (primul segment)
 - Al doilea segment indica incercarea de a crea o conexiune HTTP catre *kernel.org*
 - La inceputul celui de-al treilea segment, conexiunea TCP a fost creata si browserul isi poate primi raspunsul; in cazul nostru poate datorita latentei serverului, abia la inceputul celui de-al patrulea segment serverul web trimite continutul
 - Total: 847 milisecunde (ms) – si documentul HTML a fost obtinut
- Obs. In general paginile web constau din legaturi catre foi de stiluri, imagini, JavaScript etc.
 - Imediat ce documentul HTML incepe sa vina, browserul incepe operatia de fetch pe alta resursa (*kernel.css* in cazul nostru)
 - Obs. De data aceasta nu mai sunt intarzieri datorate DNS lookup, deoarece raspunsul anterior a fost plasat in cache-ul browserului
 - Sunt intarzieri datorate initierii conexunii TCP catre server

Web Browser | Download

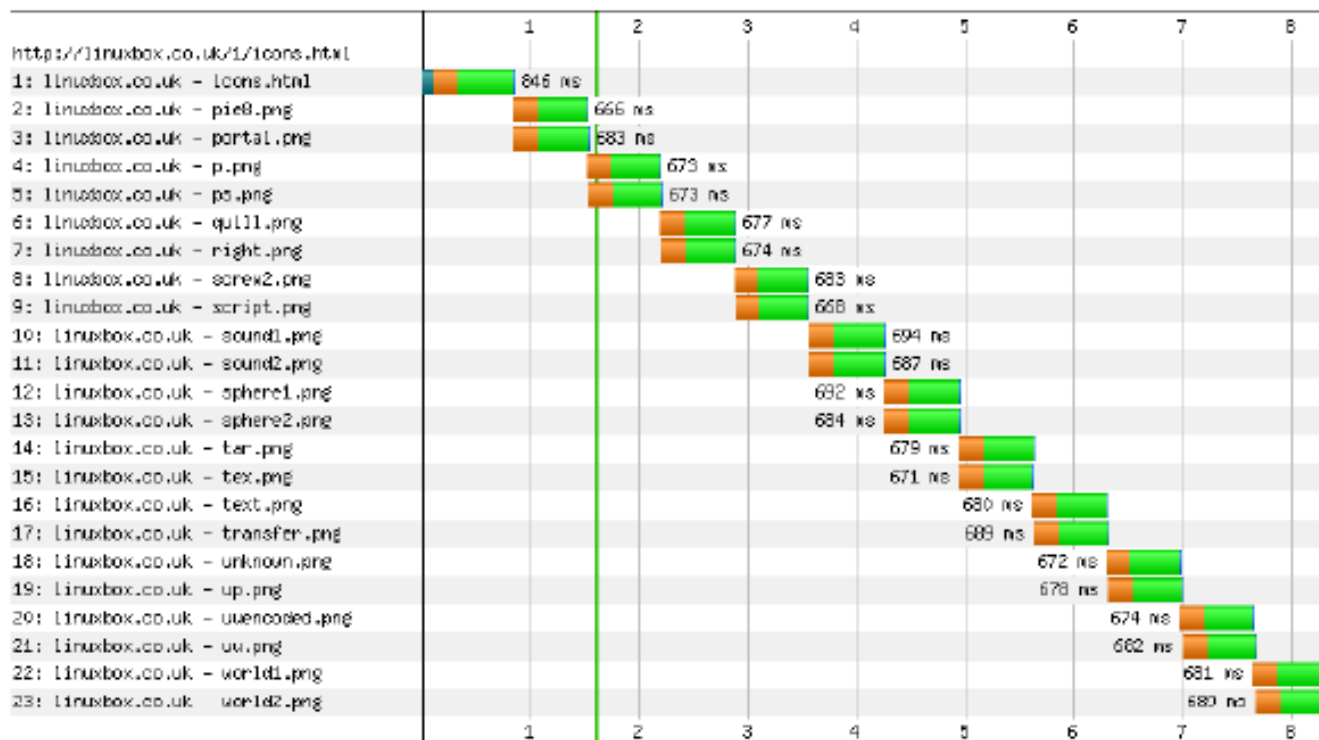
- ? De ce exista o asa mare intarziere pentru cererea ***us.png***?
 - “Motiv istoric”: in exemplul nostru browserul downloadeaza doar doua resurse in paralel de la acelasi host (cum s-a terminat kernel.css, incepe us.png)
 - Pentru resursele us.png si cele care urmeaza nu mai exista nici un segment de conexiune TCP ⇔ browserul reutilizeaza conexiunea TCP existenta cu serverul => o optimizare (salveaza 0.1 secunde per cerere)
 - Obs. Pentru resursele 1,5,12 timpul pentru descarcarea resursei e jumătate din intregul timp de *fetching* a resursei; la restul timpul de descarcare este insignifiant

Web Browser | Download

Conexiuni persistente si *Keep-Alive*

- In HTTP 1.0 – comportamentul implicit era ca dupa fiecare obtinere de resursa sa se inchida conexiunea

Efect => latenta in primirea raspunsului, utilizarea resurselor (CPU, RAM) la nivel de client si server



Web Browser | Download

Conexiuni persistente si *Keep-Alive*

- Problema a fost partial rezolvata prin introducerea lui *Keep-Alive*
 - Clientul include in antetul cererii campul: **Content: *Keep-Alive***
 - Daca serverul suporta acest aspect, trimite inapoi un *header* cu aceeasi valoare
 - => conexiunea ramine deschisa pana cand una din parti decide inchiderea ei
 - ? Dar daca un client nu inchide conexiunea?
 - Serverul este *idle* si consuma memorie
 - Majoritatea serverelor web implementeaza un ***Keep-Alive timeout***
 - De asemenea serverele pot limita numarul de resurse care se cer per o conexiune
- Obs. *Keep-Alive* nu a fost oficial recunoscuta si nu era suportata de toti clientii

Keep-Alive: timeout = 5, max 100

Web Browser | Download

Conexiuni persistente si *Keep-Alive*

- HTTP/1.1 a formalizat Keep-Alive => conexiuni persistente in mod implicit
- Daca un client | server nu doreste atunci poate utiliza un camp in antet:
Connection: close
- ? Cand comportamentul implicit al lui Keep-Alive nu este de dorit?

Download Paralel

- RFC 2616: *"Clients that use persistent connections should limit the number of simultaneous connections that they maintain to a given server. A single-user client should not maintain more than 2 connections with any server or proxy.... These guidelines are intended to improve HTTP response times and avoid congestion."*
- Scopul furnizorilor de clienti browser: cresterea gradului de interactiune cu utilizatorul
 - E problema serverelor web

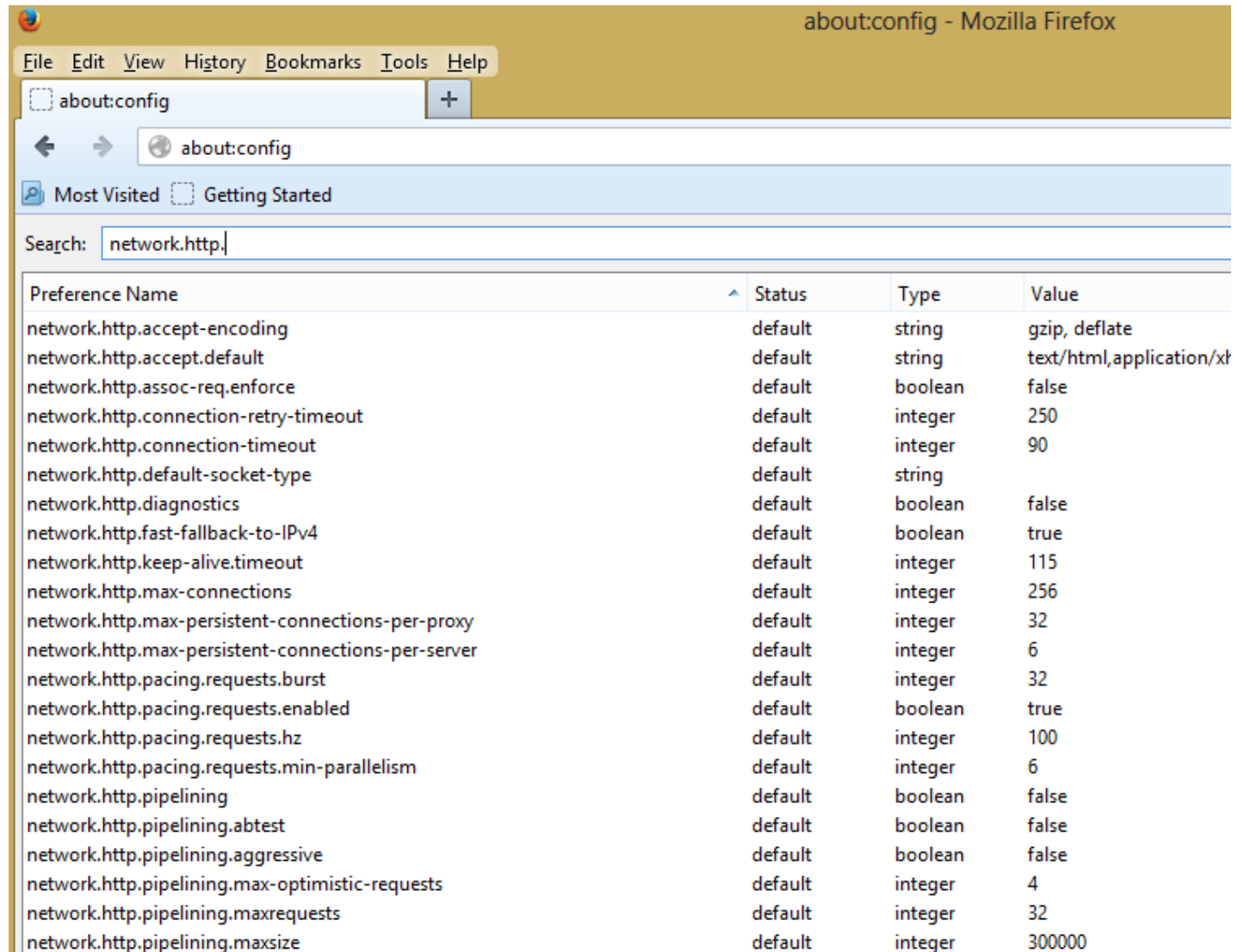
Web Browser | Download

Maximul de conexiuni paralele per host:

BROWSER	MAX PARALLEL CONNECTIONS PER HOST
IE 6 and 7	2
IE 8	6
IE 9	6
IE 10	8
Firefox 2	2
Firefox 3	6
Firefox 4 to 17	6
Opera 9.63	4
Opera 10	8
Opera 11 and 12	6
Chrome 1 and 2	6
Chrome 3	4
Chrome 4 to 23	6
Safari 3 and 4	4

Web Browser | Download

Firefox:
ajustarea
parametrilor
vizand
conexiunile
HTTP
via schema
URI
about:config

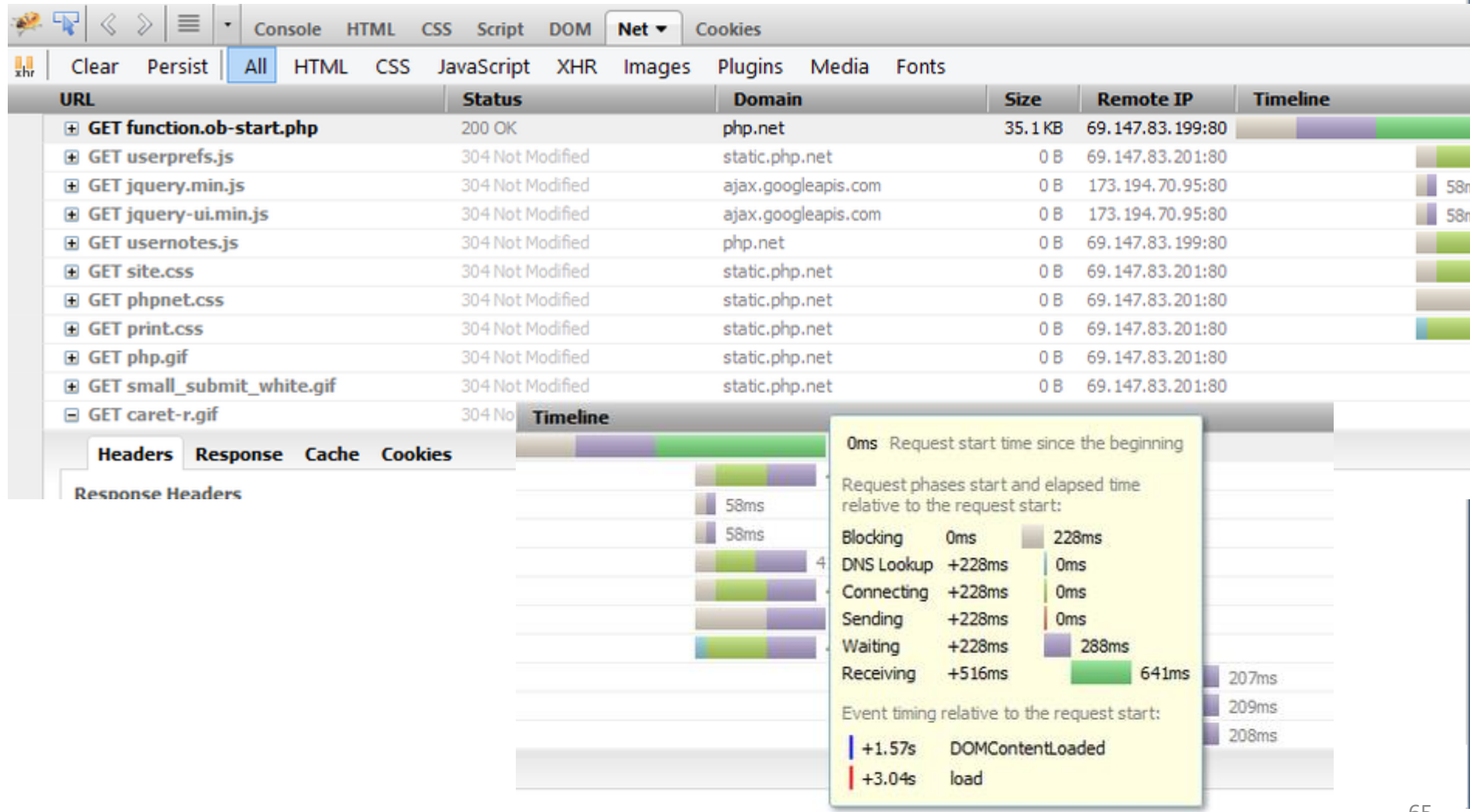


The screenshot shows the Mozilla Firefox 'about:config' page. The search bar contains 'network.http'. A table lists various network-related preferences, their status, type, and value.

Preference Name	Status	Type	Value
network.http.accept-encoding	default	string	gzip, deflate
network.http.accept.default	default	string	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
network.http.assoc-req.enforce	default	boolean	false
network.http.connection-retry-timeout	default	integer	250
network.http.connection-timeout	default	integer	90
network.http.default-socket-type	default	string	SOCK_STREAM
network.http.diagnostics	default	boolean	false
network.http.fast-fallback-to-IPv4	default	boolean	true
network.http.keep-alive.timeout	default	integer	115
network.http.max-connections	default	integer	256
network.http.max-persistent-connections-per-proxy	default	integer	32
network.http.max-persistent-connections-per-server	default	integer	6
network.http.pacing.requests.burst	default	integer	32
network.http.pacing.requests.enabled	default	boolean	true
network.http.pacing.requests.hz	default	integer	100
network.http.pacing.requests.min-parallelism	default	integer	6
network.http.pipelining	default	boolean	false
network.http.pipelining.abtest	default	boolean	false
network.http.pipelining.aggressive	default	boolean	false
network.http.pipelining.max-optimistic-requests	default	integer	4
network.http.pipelining.maxrequests	default	integer	32
network.http.pipelining.maxsize	default	integer	300000

Web Browser | Download

Firefox → Firebug



HTTP

- **Hyper Text Transfer Protocol**

Conexiunile HTTP sunt persistente

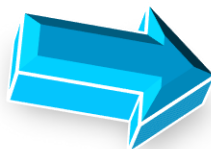
⇒ Includerea și deschiderea a mai puține conexiuni TCP ⇒ timp de CPU este salvat în rutare și host-uri (clicți, servere, proxy, ...) și se economisește memoria folosită

⇒ Clientii pot face cereri multiple în cadrul aceleiași conexiuni, fără a aștepta răspunsul pentru fiecare

⇒ Congestia în rețea este redusă datorită numărului de pachete mai mic

⇒ Cererile se desfășoară mai rapid deoarece nu mai este nevoie de un *handshake* la fiecare cerere

**Mai multe
detalii?**



Cursul de Tehnologii Web!

Rezumat

- **Protocoale la nivelul aplicatie**
 - Preliminarii
 - Caracteristici de proiectare
 - Accesul la terminal de la distanta
 - Posta Electronica
 - SMTP (Simple Mail Transfer Protocol)
 - POP (Post Office Protocol)
 - Transferul de fisiere
 - TFTP (Trivial File Transfer Protocol)
 - FTP (File Transfer Protocol)
 - World-Wide Web (HTTP)
 - Privire de ansamblu

Bibliografie

Content Networking Fundamentals, Silvano Da Ros, Publisher: Cisco Press Pub
Date: March 30, 2006 Print ISBN-10: 1-58705-240-7 Print ISBN-13: 978-1-58705-240-8 Pages: 576

Computer and Communication Networks, Nader F. Mir, Publisher: Prentice Hall Pub
Date: November 02, 2006 Print ISBN-10: 0-13-174799-1 Print ISBN-13: 978-0-13-174799-9 Pages: 656

TCP/IP Tutorial and Technical Overview, IBM, 2006

Network + Guide to Networks, Tamara Dean, 2009



Intrebari?

Intrebari?