

Advanced
Network
Technologies
Laboratory



Infrastrutture e Protocolli per Internet Laboratorio 1

Stefano Napoli

Alberto Pollastro

Politecnico di Milano



Laboratori



Responsabili di Laboratorio:

Stefano Napoli

- Ufficio: Viale Rimembranze di Lambrate, 14 – 3 piano rialzato
- Tel: 9614
- www.elet.polimi.it/upload/napoli
- mail: napoli@elet.polimi.it

Alberto Pollastro

- Ufficio: Viale Rimembranze di Lambrate, 14 – 3 piano rialzato
- Tel: 9614
- www.elet.polimi.it/upload/pollastro
- mail: pollastro@elet.polimi.it

Homepage del Corso:

www.elet.polimi.it/upload/capone/Infrastrutture
www.elet.polimi.it/upload/cesana/TEACHING/IPI2007

Laboratorio: www.elet.polimi.it/napoli/IPI/

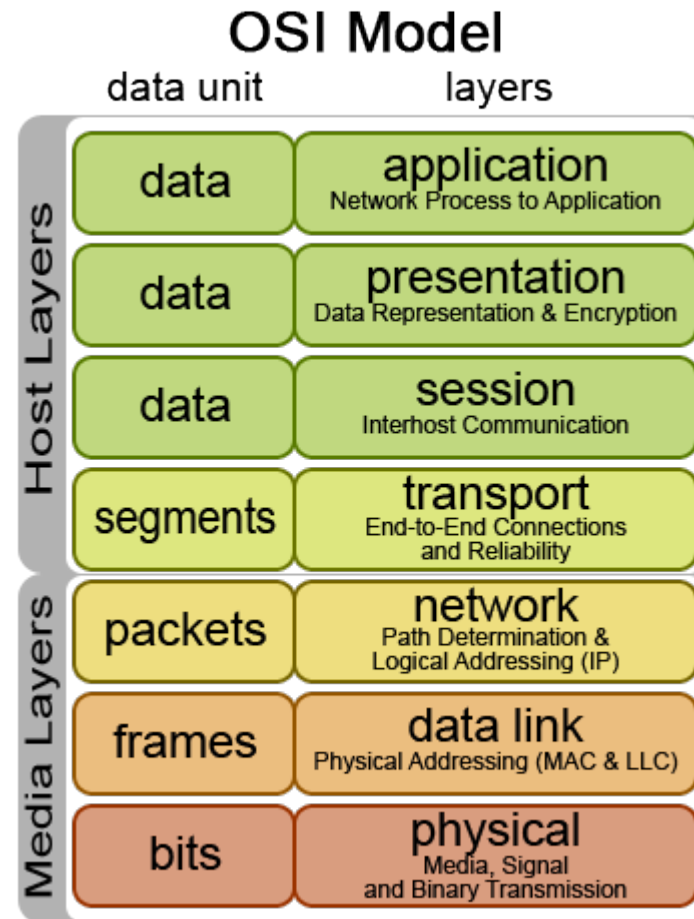
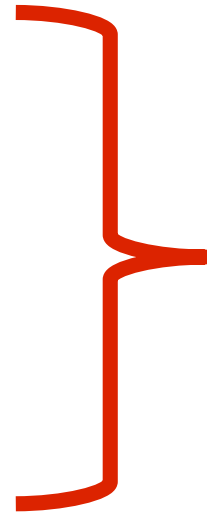


Agenda dei Laboratori



- **Lezione 1:** Protocolli Applicativi
- **Lezione 2:** Introduzione a Packet Tracer
- **Lezione 3:** Introduzione a IOS
- **Lezione 4:** Routing
- **Lezione 5:** Esercizi di ricapitolazione

- HTTP
- FTP
- SMTP
- POP3



Fonte: <http://it.wikipedia.org/wiki/ISO/OSI>



HTTP

Generalità



- Protocollo fondamentale del World Wide Web (**WWW**)
- RFC 1945 (**HTTP/1.0**) – RFC 2068 e RFC 2616 (**HTTP/1.1**)
- Protocollo **Client-Server**
- Completamente **testuale**
- Protocollo **Stateless** (ma il server può implementare tecniche per mantenere lo stato quali i *cookies*, le *sessioni*, *campi nascosti nei form*, *URL redirection*,...)
- Utilizza la Porta **80**



HTTP



Scambio di Messaggi HTTP

- Il **client** invia una richiesta contenente un metodo al server
- Il **server** risponde alla richiesta
- Il protocollo è interamente testuale!

```
Stream Content
GET / HTTP/1.1
Host: www.corriere.it
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.2) Gecko/20070208 Iceweasel/2.0.0.2 (Debian-2.0.0.2
+dfsg-3)
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive

HTTP/1.1 200 OK
Date: Tue, 13 Mar 2007 11:58:48 GMT
Server: Apache/2.0.46
Accept-Ranges: bytes
Keep-Alive: timeout=5, max=1000
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso8859-1
```

Salva come | Stampa | Entire conversation (178792 bytes) | ASCII EBCDIC Hex Dump C Arrays Raw

Filter Out This Stream



HTTP

Richieste HTTP



- Le **richieste** sono composte da tre parti:
 - Riga di richiesta (*request line*)
 - Intestazione (*header*)
 - Corpo del messaggio (*body*)
- Nella riga di **richiesta** si trovano i metodi HTTP:
 - GET
 - POST
 - HEAD
 - PUT
 - DELETE
 - TRACE
 - OPTIONS
 - CONNECT



HTTP



Header delle Richieste HTTP

- Gli **header** forniscono informazioni sul contenuto della richiesta
- Seguono il formato:

Header: valore



```
From: pippo@disney.it  
Host: www.google.it  
User-Agent: Mozilla/5.0  
Accept-Language: en-us, it-it  
Keepalive: 300  
Accept: text/xml
```

- I messaggi HTTP possono contenere un **body** che segue gli header
- Quando c'è il body, ne vengono specificate le proprietà negli header



HTTP

Risposte HTTP



- La struttura delle **risposte** è analoga a quella delle domande.
 - La riga di richiesta prende il nome di *Status Line*.
- I messaggi contenuti nella **status line** sono identificati da un codice:
 - **1xx**: informazione
 - **2xx**: successo
 - **3xx**: redirectione (richiesta è corretta, è stata rediretta ad un altro server)
 - **4xx**: errore lato client (richiesta errata)
 - **5xx**: errore lato server (c'è un problema interno del server)
 - I messaggi sono accompagnati da un testo *“human readable”*
- La status line è seguita da un insieme di **header**
 - Nel **body** della risposta si trovano tipicamente i contenuti richiesti al web server (pagina HTML, immagini, filmati...)



Risposte HTTP: Status Codes

- Riportiamo alcuni esempi tipici di status code di HTTP
- L'elenco completo è definito nell'RFC 2616
(<http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>)

- **301 Moved Permanently:** la risorsa è stata spostata in un'altra locazione in maniera permanente
- **304 Not Modified:** la risorsa non è stata modificata dall'ultima volta, non serve reinviarla
- **400 Bad Request** il server non è in grado di capire la richiesta
- **404 Not Found:** la risorsa richiesta non è disponibile (ad es., è stata richiesta una pagina inesistente)
- **500 Internal Server Error:** c'è stato un errore interno nel server (ad es., un errore di connessione al db)



HTTP



Download di una pagina HTTP (1)

Scarichiamo una pagina col protocollo HTTP usando *telnet*.

1) Ci connettiamo al server web sulla porta 80

```
myhost$ telnet www.elet.polimi.it 80
Trying 131.175.120.33...
Connected to web0.elet.polimi.it.
Escape character is '^]'.
```

2) Richiediamo una pagina web (è necessaria una riga vuota alla fine della richiesta: dare 2 [INVIO])

```
GET /upload/napoli/IPI/testpage.html HTTP/1.0
```

3) La pagina viene scaricata e la connessione viene chiusa



HTTP



Analisi della Risposta

HTTP/1.1 200 OK

Date: Fri, 16 Mar 2007 10:41:36 GMT

Server: Apache/2.0.52 (Red Hat)

Last-Modified: Fri, 16 Mar 2007 09:40:43 GMT

ETag: "110014-417-8817cc0"

Accept-Ranges: bytes

Content-Length: 1047

Connection: close

Content-Type: text/html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

```
<html>
```

```
[...] </html>
```



HTTP



Download di una pagina HTTP (2)

- È stata scaricata la pagina, ma non gli elementi *non HTML* che la compongono: in questo caso, mancano le immagini!
- Dal codice HTML leggiamo che si chiamano `img/logo_poli_small.png` e `img/logo-antlab.png`

4) Ci connettiamo ancora al server web

```
myhost$ telnet www.elet.polimi.it 80 [...]
```

5) scarichiamo la prima immagine

```
GET /upload/napoli/IPI/img/logo_poli_small.png HTTP/1.0
```

(ovviamente l'immagine è un file binario, per cui compariranno a monitor un insieme di caratteri)



HTTP

Esercizi



- Provare ad ottenere dal web server un messaggio di risposta con la Status Line **400 Bad Request**
- Provate a richiedere al web server il file `index.php` invece che `index.html`: che risposta da il server?

WEB SERVER: *www.elet.polimi.it*

- Le risposte 3xx prevedono che il browser compia ulteriori azioni.
- In particolare, 302 FOUND indica al browser che la risorsa è presente in un'altra locazione:
The URL has moved here
- Gli amministratori del sito reindirizzano gli utenti che cercano pagine inesistenti ad una pagina di errore di default



HTTP

Keep Alive



È possibile rendere persistente una connessione HTTP, in modo che tutti i messaggi passino sulla stessa connessione!

HTTP/1.0

- Non c'è una specifica indicazione. Si usa l'header:
Connection: Keep-Alive
sia nella richiesta che nella risposta.
- La connessione viene chiusa esplicitamente dal client oppure dal server dopo un timeout

HTTP/1.1

- La connessione viene mantenuta aperta **di default**, si può renderla non permanente con un apposito header (vedi sotto).
- L'header **Connection: Keep-Alive** non ha significato.
- La connessione viene chiusa con l'header:
Connection: Closed



HTTP



Download di una pagina HTTP/1.1

Scarichiamo una pagina usando HTTP/1.1

1) Ci connettiamo al server web sulla porta 80

```
myhost$ telnet www.elet.polimi.it 80
```

2) Richiediamo una pagina web

```
GET /upload/napoli/IPI/testpage.html HTTP/1.1
```

3) Usando HTTP/1.1, c'è un header obbligatorio (Host), che contiene il nome del dominio

```
Host: elet.polimi.it
```



HTTP



Download di una pagina HTTP/1.1 (2)

La pagina viene scaricata, ma la connessione rimane aperta!

4) Scarichiamo allora la prima immagine:

```
GET /upload/napoli/IPI/img/logo_poli_small.png HTTP/1.1  
Host: elet.polimi.it
```

5) Scarichiamo la seconda immagine, chiudendo la connessione:

```
GET /upload/napoli/IPI/img/logo_antlab.png HTTP/1.1  
Host: elet.polimi.it  
Connection: Closed
```

La connessione ha un timeout, scaduto il quale viene chiusa!



HTTP

GET Condizionato



Per risparmiare banda, HTTP/1.1 prevede l'uso di GET condizionato:

```
GET /upload/napoli/IPI/testpage.html HTTP/1.1
Host: elet.polimi.it
If-Modified-Since: Fri, 16 Mar 2007 13:59:59 GMT
```

Il server risponde col messaggio **200 OK** e invia la pagina se essa è stata modificata dopo il 16/03/2007, altrimenti risponde con un **304 Not-Modified**:

```
HTTP/1.1 304 Not Modified
Date: Fri, 16 Mar 2007 13:59:59 GMT
```



FTP

Generalità



- Protocollo di condivisione file tra host
- RFC 959
- Protocollo **Client-Server**
- Completamente **testuale**
- Opera direttamente sul filesystem del sistema
- Connessione di controllo: porta **21**



FTP



Comandi FTP (1)

Comandi necessari a comunicare via FTP:

Connessione al server sulla porta 21 (connessione di controllo):

```
myhost$ telnet ftp.elet.polimi.it 21
```

Autenticazione:

```
USER <nomeutente>  
PASS <password>  
QUIT
```

Informazioni:

```
HELP  
STAT < NULL | nome_file>  
SYST
```

Lista più completa di comandi: <http://www.nsftools.com/tips/RawFTP.htm>



FTP

Comandi FTP (2)



Movimenti:

```
PWD  
CWD  
LIST  
QUIT
```

Setup:

```
MODE <Stream | Block | Compressed>  
PASV  
PORT a1, a2, a3, a4, p1, p2*
```

Manipolazione:

```
RETR  
STOR  
DELE
```

Modalità di impiego:

- Protocollo con 2 connessioni -> 2 telnet
- I comandi vengono dati su connessione di controllo
- I trasferimenti avvengono sulla connessione dati

* IP del server a cui connettersi: a1.a2.a3.a4 - PORT a cui connettersi: p1*256 + p2

Sessione d'esempio

Shell 1

```
$ telnet ftp.elet.polimi.it 21
USER Anonymous
PASS
PASV
227 Entering Passive Mode
(131,175,120,33,46,204)
CWD /outgoing/Stefano.Napoli/
```

Server

Shell 2



$$46 * 256 + 204 = 11980$$

```
$telnet ftp.elet.polimi.it 11980
```

LIST

```
-rw-r--r-- 1 1463 1463 63763 Mar 19 08:13 Debian.png
-rw-r--r-- 1 1463 1463 22493 Mar 19 08:23 GPL.txt
-rw-r--r-- 1 1463 1463 758965 Mar 19 08:13 tux.bigger.png
Connection closed by foreign host.
```

PASV

```
227 Entering Passive Mode
(131,175,120,33,420,21)
```

```
$telnet ftp.elet.polimi.it 107541
```

RETR GPL.txt

```
Questa è una traduzione italiana non ufficiale della Licenza
Pubblica Generica GNU. Non è pubblicata dalla Free Software
[...]
```



FTP

Esercizi



- Loggarsi sul server FTP messo a disposizione in laboratorio
*(l'indirizzo di tale server verrà fornito durante il laboratorio stesso
– USER anonymous, PASS <anything>)*
- Scoprire il sistema operativo del server
- Scoprire informazioni generali sul server
- Provare a scaricare un'immagine
- Provare l'altro metodo di connessione, **PORT**: funziona?



SMTP

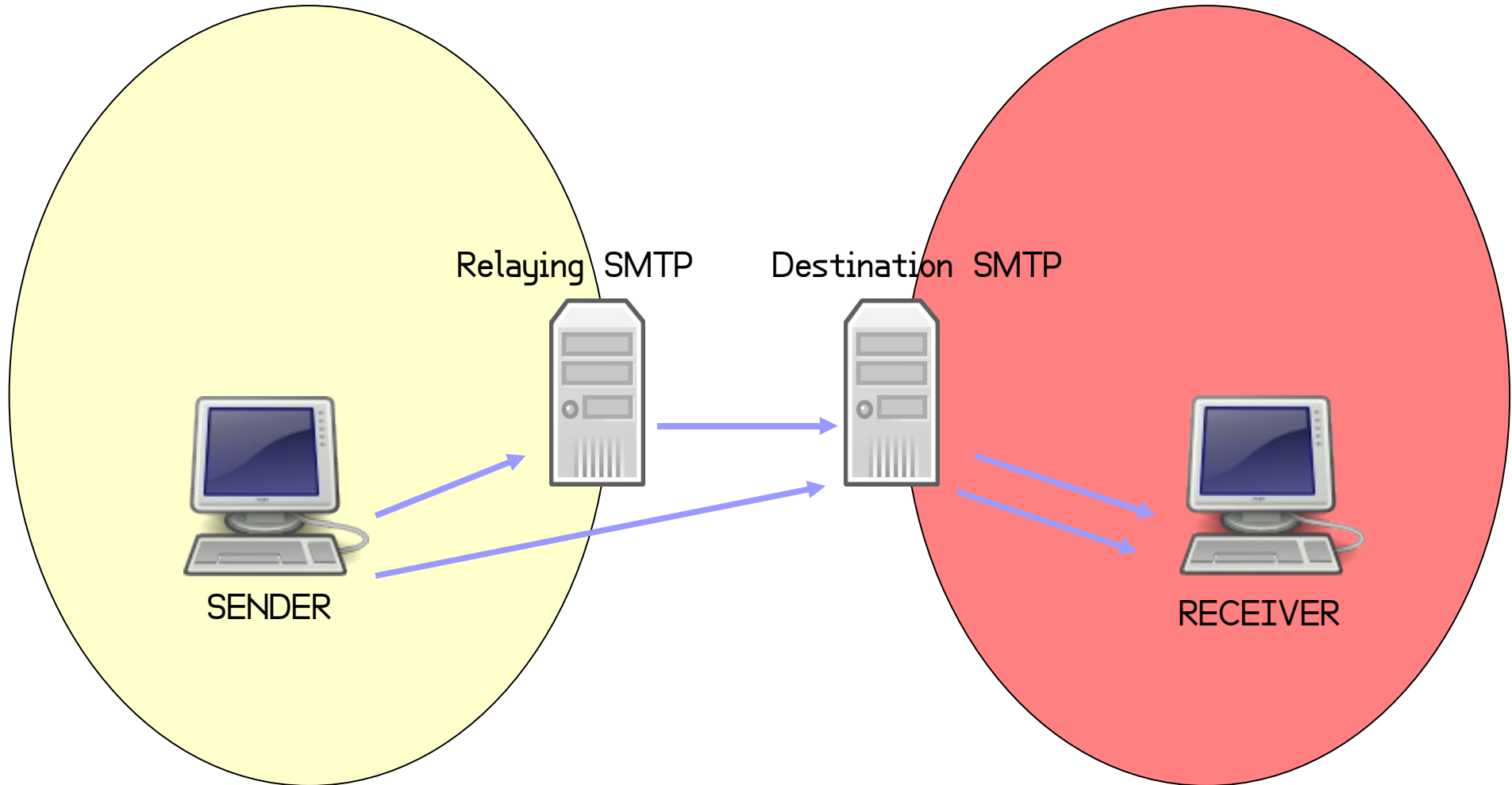
Generalità



- Protocollo di trasmissione di email
- RFC 821 – Specifica base
- Protocollo **testuale**
- Inizialmente permetteva solo il trasferimento ASCII, è stato successivamente esteso per trasmettere informazioni binarie (estensione MIME, 8BITMIME,...)
- Utilizza la porta **25**
- SMTP originariamente non prevede autenticazione:
 - SMTP-AUTH / Internet Mail 2000
 - IETF Anti-Spam Research Group

Sender Domain

Receiver Domain





SMTP

Generalità (3)



- Voglio inviare una mail ad un amico del dominio **receivers.com**
- Mi connetto al server SMTP del dominio *receivers.com*
- Dichiaro (nel mittente della mail) il dominio da cui scrivo:
senders.com
- Scrivo la mail e la deposito nella “cartella” di *friend@receivers.com*
- Non mi è stato chiesto alcun USER o PASS!!! (no autenticazione)
- (I gestori del dominio *receivers.com* tuttavia possono operare alcuni controlli, ad esempio bannando degli indirizzi IP o verificando che l'IP del mittente sia effettivamente parte del dominio dichiarato)
- I provider tipicamente offrono dei server SMTP che si occupano di consegnare alla mailbox del destinatario al posto nostro (gestendo ad es. il caso di server occupato, momenti di down,...)



SMTP

Comandi SMTP



Comandi:

```
HELO <SP> <domain> <CRLF>
MAIL <SP> FROM:<reverse-path> <CRLF>
RCPT <SP> TO:<forward-path> <CRLF>
DATA <CRLF>
RSET <CRLF>
SEND <SP> FROM:<reverse-path> <CRLF>
VRFY <SP> <string> <CRLF>
EXPN <SP> <string> <CRLF>
HELP [<SP> <string>] <CRLF>
NOOP <CRLF>
QUIT <CRLF>
```

I comandi sono Case Insensitive, tranne nel *reverse* e *forward path*, che sono Sensitive



SMTP

Sessione SMTP



```
[MyLinuxBox]$ telnet mail.provider.it 25
220 ipi2007.it ESMTP Postfix (Debian/GNU)
HELO studente.it
250 Hello...
MAIL FROM:<matr666999@studente.it>
250 Ok
RCPT TO:<professore@ipi2007.it>
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Subject: Verifica SMTP
From: matr666999@studente.it
To: professore@ipi2007.it
Buongiorno
Questa mail serve a dimostrare che ho imparato SMTP.
.
250 Ok: queued as 42597
QUIT
221 Bye
```



SMTP

Esercizi



- Inviare una mail da un mittente falso (*pippo@disney.it*) all'indirizzo email *labipi<numeroPC>@ipi07.lan*
(l'IP del server SMTP verrà comunicato a lezione)
(numeroPC è di tre cifre, ad es: 001)
- Inviare, sempre allo stesso indirizzo mail, altre 2 mail usando mittenti e subject diversi (serve per un esercizio successivo)
- *[A CASA]* provare a mandare delle mail al proprio indirizzo email da *pippo@disney.it*

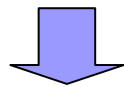


POP

Generalità



- Protocollo di accesso alla mailbox
- RFC 1939
- Protocollo **testuale**
- Protocollo con Autenticazione
- Utilizza la porta **110**
- Non è cifrato, per cui user e password passano in chiaro



Comando **APOP**



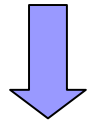
POP

Comandi POP



Login:

```
USER <username>  
PASS <password>  
APOP <username> <digest>
```



MD5 (Timestamp, Pass)

Risposte del server:

```
-ERR  
+OK
```

Operazioni:

```
STAT : info sullo stato mbox  
LIST : elenca il # messaggi  
RETR n : leggi messaggio n  
DELE n : cancella messaggio n  
RSET : annulla cancellazioni  
QUIT : esce  
NOOP : non fa operazioni  
CAPA : mostra le capabilities  
del server
```



POP

Sessione POP



```
[MyLinuxBox]$ telnet pop.provider.it 110
+OK CommuniGate Pro POP3 Server 5.1.6 ready <540714.1174383679@provider.it>
USER homer
+OK
PASS marge
+OK
LIST
+Ok
 1 125
 2 586
.
RETR 1
+OK
Return-Path: <moe@moesbar.com>
Delivered-To: homer@provider.it
Date: Sat, 10 Mar 2007 13:24:54 +0200
From: Moe <moe@moesbar.com>
Subject: Hey Homer
Content-Type: text/plain; charset=ISO-8859-1
<testo del messaggio>
DELE 1
+OK
QUIT
+OK
```



POP

Esercizi



- Collegarsi al server POP della lan dell'aula (l'IP è lo stesso dell'SMTP server, comunicato a lezione)
- Scaricare la mail precedentemente inviata dall'indirizzo pippo@disney.it
- Con il comando TOP verificare gli header della mail
- Cancellare le 3 mail inviate in precedenza
- Annullare la cancellazione con il comando RSET
- Cancellarle di nuovo, definitivamente