

Reti di Calcolatori  
AA 2011/2012



UNIVERSITÀ DEGLI STUDI DI TRENTO

<http://disi.unitn.it/locigno/index.php/teaching-duties/computer-networks>

Protocolli di applicazione  
(2)

Csaba Kiraly  
Renato Lo Cigno

---

---

---

---

---

---

---

---

Livello di applicazione (2)



**A note on the use of these slides:**  
These slides are an adaptation from the freely available version provided by the book authors to all (faculty, students, readers). The originals are in PowerPoint and English.

The Italian translation is originally from Gianluca Torta, Stefano Leonardi, Francesco Di Tria  
Adaptation is by Csaba Kiraly and Renato Lo Cigno

All material copyright 1996-2012  
J.F. Kurose and K.W. Ross, All Rights Reserved

*Computer Networking:  
A Top Down Approach,  
5th edition.  
Jim Kurose, Keith Ross  
Addison-Wesley, April 2009.*

*Reti di calcolatori e Internet:  
Un approccio top-down  
4ª edizione  
Pearson Paravia Bruno Mondadori Spa  
©2008*

{kiraly,locigno}@disi.unitn.it

2




---

---

---

---

---

---

---

---

Capitolo 2: Livello applicazione

- ▣ 2.1 Principi delle applicazioni di rete
- ▣ 2.2 Web e HTTP
- ▣ 2.3 FTP
- ▣ 2.4 Posta Elettronica
  - SMTP, POP3, IMAP
- ▣ 2.5 DNS

{kiraly,locigno}@disi.unitn.it

3




---

---

---

---

---

---

---

---

## GET condizionale

- Obiettivo: non inviare un oggetto se il client ha una copia aggiornata dell'oggetto
- Cache del browser: tiene una copia dell'oggetto già scaricato
- client: specifica la data della copia dell'oggetto nella richiesta HTTP
  - IF-modified-since: <data>
- server: la risposta non contiene l'oggetto se la copia nella cache è aggiornata:
  - HTTP/1.0 304 Not Modified

oggetto non modificato

client      server

Richiesta HTTP  
If-modified-since: <data>

Risposta HTTP  
HTTP/1.0 304 Not Modified

oggetto modificato

client      server

Richiesta HTTP  
If-modified-since: <data>

Risposta HTTP  
HTTP/1.0 200 OK  
<dati>

{kiraly,locigno}@disi.unitn.it

---

---

---

---

---

---

---

---

---

---

## GET condizionale (2)

- D: Qual'è la data utilizzata?
- R: la data nella risposta originale

oggetto non modificato

client      server

Richiesta HTTP  
If-modified-since: <data>

Risposta HTTP  
HTTP/1.0 304 Not Modified

oggetto modificato

client      server

Richiesta HTTP

Risposta HTTP  
HTTP/1.0 200 OK  
Last-Modified: <data>  
<dati>

{kiraly,locigno}@disi.unitn.it

---

---

---

---

---

---

---

---

---

---

The screenshot shows the CNN website in a browser window. Below it, a network traffic analysis tool displays a list of HTTP requests and responses. The table below summarizes the data shown in the tool:

URL	Status	Domain	Size	Remote IP	Timeline
GET edition.cnn	200 OK	edition.cnn.com	22 KB	157.166.255.32.80	48ms
GET iadthpb-mi	304 Not Modified	z.cdn.turner.com	33.1 KB	95.101.34.9.80	18ms
GET iadthpb-mi	304 Not Modified	z.cdn.turner.com	112.2 KB	95.101.34.9.80	18ms
GET iad-globe-c	304 Not Modified	i.cdn.turner.com	4.4 KB	192.231.106.126.80	18ms
GET btn_search	304 Not Modified	i.cdn.turner.com	858 B	196.78.208.254.80	36ms
GET site-css.js	200 OK	eds.cnn.com	2.3 KB	157.166.226.207.80	248ms
GET banner.html	200 OK	edition.cnn.com	255 B	157.166.255.32.80	120ms
GET 1203050831	200 OK	i2.cdn.turner.com	17.6 KB	196.78.208.254.80	77ms
GET video_icons	304 Not Modified	i.cdn.turner.com	159 B	192.231.106.126.80	27ms
GET 1px.gif	304 Not Modified	i.cdn.turner.com	43 B	192.231.106.126.80	9ms
GET 120305104	200 OK	i2.cdn.turner.com	43.5 KB	196.78.208.254.80	92ms
GET 1201250746	200 OK	i2.cdn.turner.com	9.5 KB	196.78.208.254.80	39ms
GET 1203050231	200 OK	i2.cdn.turner.com	5.9 KB	196.78.208.254.80	56ms
GET 120222055	200 OK	i2.cdn.turner.com	5.2 KB	196.78.208.254.80	57ms
GET 120304132	200 OK	i2.cdn.turner.com	5.8 KB	196.78.208.254.80	57ms
GET 120305022	200 OK	i2.cdn.turner.com	5.9 KB	196.78.208.254.80	57ms
GET 120220823	200 OK	i2.cdn.turner.com	4.9 KB	196.78.208.254.80	75ms
GET 120305030	200 OK	i2.cdn.turner.com	3.6 KB	196.78.208.254.80	76ms
GET 12024122	200 OK	i2.cdn.turner.com	6.3 KB	196.78.208.254.80	76ms
GET 120305103	200 OK	i2.cdn.turner.com	7.6 KB	196.78.208.254.80	75ms
GET site-css.js	200 OK	eds.cnn.com	6.4 KB	157.166.226.207.80	36ms
GET advertisement	304 Not Modified	i.cdn.turner.com	94 B	192.231.106.126.80	30ms
GET 35x35_gene	304 Not Modified	i.cdn.turner.com	229 B	192.231.106.126.80	37ms

---

---

---

---

---

---

---

---

---

---

## Server Proxy

**Proxy:**

- interpone tra un **client** ed un **server** facendo da tramite tra i due
- inoltra le richieste e le risposte dall'uno all'altro

**Obiettivo:**

- ▢ Caching proxy
- ▢ Connettività
- ▢ Controllo/filtraggio/modifiche
- ▢ Privacy

{kiraly,locigno}@disi.unitn.it 7

---

---

---

---

---

---

---

---

## Incapsulamento (richiamo)

{kiraly,locigno}@disi.unitn.it 8

---

---

---

---

---

---

---

---

## Server Proxy (2)

{kiraly,locigno}@disi.unitn.it 9

---

---

---

---

---

---

---


---

## Server Proxy (3)

- Richiesta senza Proxy  
GET /pub/WWW/TheProject.html HTTP/1.0
- Richiesta con Proxy  
GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.0

**absolute URL**

- Absolute URL: necessario per aprire un connessione TCP verso il server nel Proxy

{kiraly,locigno}@disi.unitn.it 10 

---

---

---

---

---

---

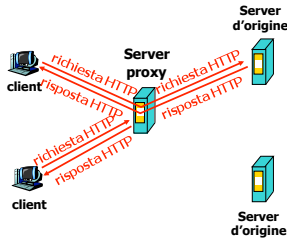
---


---

## Cache web (proxy)

**Obiettivo: soddisfare la richiesta del client senza coinvolgere il server d'origine**

- L'utente configura il browser: accesso al Web tramite la cache
- Il browser trasmette tutte le richieste HTTP alla cache
  - oggetto nella cache: la cache fornisce l'oggetto
  - altrimenti la cache richiede l'oggetto al server d'origine e poi lo inoltra al client



{kiraly,locigno}@disi.unitn.it 11 

---

---

---

---

---

---

---


---

## Cache web (continua)

- La cache opera come client e come server
- Tipicamente la cache è installata da un ISP (università, aziende o ISP residenziali)

**Perché il web caching?**

- Riduce i tempi di risposta alle richieste dei client
- Riduce il traffico sul collegamento di accesso (dell'ISP) a Internet

{kiraly,locigno}@disi.unitn.it 12 

---

---

---

---

---

---

---

---

## Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Posta Elettronica  
SMTP, POP3, IMAP
- 2.5 DNS

{kiraly,locigno}@disi.unitn.it 13

---

---

---

---

---

---

---

---

## FTP: file transfer protocol

- Trasferimento file a/da un host remoto
- Modello client/server
  - *client*: il lato che inizia il trasferimento (a/da un host remoto)
  - *server*: host remoto
- ftp: RFC 959
- server ftp: porta 21

{kiraly,locigno}@disi.unitn.it 14

---

---

---

---

---

---

---

---

## FTP: connessione di controllo, connessione dati

- Il client FTP contatta il server FTP alla porta 21, specificando TCP come protocollo di trasporto
- Il client ottiene l'autorizzazione sulla connessione di controllo
- Il client cambia la directory remota inviando i comandi sulla connessione di controllo
- Quando il server riceve un comando per trasferire un file, apre una connessione dati TCP con il client
- Dopo il trasferimento di un file, il server chiude la connessione

- Il server apre una seconda connessione dati TCP per trasferire un altro file.
- Connessione di controllo: "fuori banda" (*out of band*)
- Il server FTP mantiene lo "stato": associare la connessione di controllo ad un utente e tenere traccia della directory corrente

{kiraly,locigno}@disi.unitn.it 15

---

---

---

---

---

---

---

---

## Comandi e risposte FTP

**Comandi comuni:**  
 Inviati come testo ASCII sulla connessione di controllo

- USER *username*
- PASS *password*
- LIST elenca i file della directory corrente
- RETR *filename* recupera (*get*) un file dalla directory corrente
- STOR *filename* memorizza (*put*) un file nell'host remoto

**Codici di ritorno comuni:**  
 Codice di stato ed espressione (come in HTTP)

- 331 Username OK, password required
- 125 data connection already open; transfer starting
- 425 Can't open data connection
- 452 Error writing file

{kiraly,locigno}@disi.unitn.it 16

---

---

---

---

---

---

---

---

## Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Posta Elettronica  
 SMTP, POP3, IMAP
- 2.5 DNS

{kiraly,locigno}@disi.unitn.it 17

---

---

---

---

---

---

---

---

## Posta elettronica

**Componenti principali:**

- agente utente
- server di posta

**Protocolli principali:**

- SMTP: Simple Mail Transfer Protocol
- POP3: Post Office Protocol
- IMAP: Internet Mail Access Protocol

**Agente utente**

- detto anche "mail reader"
- composizione, editing, lettura dei messaggi di posta elettronica
- esempi:
  - Eudora, Outlook, Mozilla Thunderbird
  - pine, elm
  - Web browser!
- i messaggi in uscita o in arrivo sono memorizzati sul server

Legend:  
 [Green bars] Coda di messaggi in uscita  
 [Yellow box] casella di posta dell'utente

{kiraly,locigno}@disi.unitn.it 18

---

---

---

---

---

---

---

---

## Posta elettronica: server di posta

- Server di posta
  - Casella di posta (*mailbox*) contiene i messaggi in arrivo per l'utente
  - Coda di messaggi da trasmettere
  - Protocollo SMTP tra server di posta per inviare messaggi di posta elettronica
    - client: server di posta trasmittente
    - "server": server di posta ricevente

{kiraly,locigno}@disi.unitn.it 19 UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---

## Posta elettronica: SMTP [RFC 2821]

- usa TCP per trasferire in modo affidabile i messaggi di posta elettronica dal client al server, porta 25
- trasferimento diretto: il server trasmittente al server ricevente (di solito)
- tre fasi per il trasferimento
  - handshaking (saluto)
  - trasferimento di messaggi
  - chiusura
- interazione comando/risposta
  - comandi: testo ASCII
  - risposta: codice di stato ed espressione
- i messaggi devono essere nel formato ASCII a 7 bit

{kiraly,locigno}@disi.unitn.it 20 UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---

## Scenario: Alice invia un messaggio a Roberto

- Alice usa il suo agente utente per comporre il messaggio da inviare "a" `rob@someschool.edu`
- L'agente utente di Alice invia un messaggio al server di posta di Alice; il messaggio è posto nella coda di messaggi
- Il lato client di SMTP apre una connessione TCP con il server di posta di Roberto
- Il client SMTP invia il messaggio di Alice sulla connessione TCP
- Il server di posta di Roberto pone il messaggio nella casella di posta di Roberto
- Roberto invoca il suo agente utente per leggere il messaggio

{kiraly,locigno}@disi.unitn.it 21 UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---

## Esempio di interazione SMTP

```

S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <rob@hamburger.edu>
S: 250 rob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
  
```

{kiraly,locigno}@disi.unitn.it 22

---

---

---

---

---

---

---

---

---

---

## SMTP: note finali

- SMTP usa connessioni persistenti
- SMTP richiede che il messaggio (intestazione e corpo) sia nel formato ASCII a 7 bit
- Il server SMTP usa CRLF, CRLF per determinare la fine del messaggio

**Confronto con HTTP:**

- HTTP: pull
- SMTP: push
- Entrambi hanno un'interazione comando/risposta in ASCII, codici di stato
- HTTP: ciascun oggetto è incapsulato nel suo messaggio di risposta
- SMTP: più oggetti vengono trasmessi in un unico messaggio

{kiraly,locigno}@disi.unitn.it 23

---

---

---

---

---

---

---

---

---

---

## Formato dei messaggi di posta elettronica

SMTP: protocollo per scambiare messaggi di posta elettronica  
RFC 822: standard per il formato dei messaggi di testo:

- Righe di intestazione, per esempio
  - To/A:
  - From/Da:
  - Subject/Oggetto:*differenti dai comandi SMTP!*
- corpo
  - il "messaggio", soltanto caratteri ASCII

{kiraly,locigno}@disi.unitn.it 24

---

---

---

---

---

---

---

---

---

---



## Formato del messaggio: estensioni di messaggi multimediali

- MIME: estensioni di messaggi di posta multimediali, RFC 2045, 2056
- Alcune righe aggiuntive nell'intestazione dei messaggi dichiarano il tipo di contenuto MIME

Versione MIME

metodo usato per codificare i dati

Tipo di dati multimediali, sottotipo, dichiarazione dei parametri

Dati codificati

```

From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.....base64 encoded data

```

{kiraly,locigno}@disi.unitn.it 25

---

---

---

---

---

---

---

---

---

---

## Messaggio ricevuto

Chi ha inviato

Chi ha ricevuto

Quando

```

Received: from crepes.fr by hamburger.edu;
12 Oct 98 15:27:39 GMT

From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.....base64 encoded data

```

{kiraly,locigno}@disi.unitn.it 26

---

---

---

---

---

---

---

---

---

---

## SMTP demo

- telnet servername 25
- Server risponde con codice 220
- Utilizzando I comandi
  - HELO, MAIL FROM, RCPT TO, DATA, QUIT
- potete mandare e-mail "a mano"

{kiraly,locigno}@disi.unitn.it 27

---

---

---

---

---

---

---

---

---

---

## Protocolli di accesso alla posta

Server di posta del mittente      Server di posta del destinatario

- SMTP: consegna/memorizzazione sul server del destinatario
- Protocollo di accesso alla posta: ottenere i messaggi dal server
  - (agente utente direttamente sul server di posta)
    - accesso diretto tramite il file system
  - POP: Post Office Protocol [RFC 1939]
    - Porta 110
    - autorizzazione (agente <-> server) e download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - più funzioni (più complesse)
    - manipolazione di messaggi memorizzati sul server
  - HTTP: gmail, Hotmail, Yahoo! Mail, webmail UNITN, ecc.

{kiraly,locigno}@disi.unitn.it      28      UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---

## Protocollo POP3

**Fase di autorizzazione**

- Comandi del client:
  - user**: dichiara il nome dell'utente
  - pass**: password
- Risposte del server:
  - +OK
  - ERR

**Fase di transazione**

- Comandi del client:
  - list**: elenca i numeri dei messaggi
  - retr**: ottiene i messaggi in base al numero
  - dele**: cancella
  - quit**

```

S: +OK POP3 server ready
C: user rob
S: +OK
C: pass hungry
S: +OK user successfully logged on

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
          
```

{kiraly,locigno}@disi.unitn.it      29      UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---

## POP3 (altro) e IMAP

**Ancora su POP3**

- Il precedente esempio usa la modalità "scarica e cancella"
- Roberto non può rileggere le e-mail se cambia client
- Modalità "scarica e mantieni": copia i messaggi su più client
- POP3 è un protocollo senza stato tra le varie sessioni

**IMAP**

- Mantiene tutti i messaggi in un unico posto: il server
- Consente all'utente di organizzare i messaggi in cartelle
- IMAP conserva lo stato dell'utente tra le varie sessioni:
  - I nomi delle cartelle e l'associazione tra identificatori dei messaggi e nomi delle cartelle

{kiraly,locigno}@disi.unitn.it      30      UNIVERSITÀ DEGLI STUDI DI TRENTO

---

---

---

---

---

---

---

---