Università di Roma Tor Vergata
Corso di Laurea triennale in Informatica

Sistemi operativi e reti

A.A. 2019-2020

Pietro Frasca

Parte II: Reti di calcolatori

Lezione 7 (31)

Giovedì 2-04-2020

GET condizionale

- Per aumentare la velocità di trasferimento dei documenti e diminuire la quantità di traffico Web, i browser utilizzano due tipi di cache, uno posto in memoria principale e l'altro residente su memoria secondaria.
- Quando un browser ottiene una pagina, la visualizza, ne mantiene il contenuto in memoria ram e salva tutti i file che la compongono nella cache su memoria secondaria, all'interno di una specifica cartella.
- Quando un browser richiede un oggetto, verifica prima se esso si trova nelle cache, prima in memoria ram poi su memoria secondaria. Se è presente lo carica dalla cache.
- Oltre alle suddette **cache,** interne al client, è possibile utilizzare anche un **server cache** esterno detto **server proxy**.
- L'uso delle cache riduce i tempi di risposta per ottenere una pagina web, ma ovviamente crea il problema di aggiornamento della pagina.

- In altre parole, se la pagina originale nel server Web viene modificata la pagina presente nella cache del client non è aggiornata.
- L'HTTP risolve questo problema con un meccanismo detto **GET condizionale**, basato sulla linea di intestazione **If-Modified-Since**.
- Per descrivere il funzionamento del GET condizionale, consideriamo il seguente esempio.

1. un browser richiede un oggetto, non presente nella cache, al server web www.pf.uniroma2.it:

```
GET /img/schema1.gif HTTP/1.1
Host: www.pf.uniroma2.it
…
…
```

2. il server web invia al client un messaggio di risposta con l'oggetto richiesto:

```
HTTP/1.1 200 OK
Date: Tue, 13 Mar 2012 10:25:26 GMT
Last-Modified: Sat, 25 Feb 2012 11:34:56
Server: Apache/2.2.2 (Unix) PHP/5.1.6
Content-Length: 10022
Content-Type: image/gif

(dati dati dati …)
```

Il browser visualizza l'oggetto (nell'esempio un immagine gif) e lo salva anche nella cache su disco, o su altro dispositivo di memoria secondaria. Il browser oltre al file salva anche il suo URL e l'ultima data di modifica del file stesso che recupera dal campo **Last-Modified**.

3. Successivamente, l'utente richiede lo stesso file e supponiamo che questo sia ancora presente nella cache. Dato che il file potrebbe essere stato modificato sul server web, il browser inserisce nel messaggio di richiesta la linea di intestazione **If-Modified-Since:**

```
GET /img/schema1.gif HTTP/1.1
Host: www.pf.uniroma2.it
If-modified-since: Sat, 25 Feb 2012 11:34:56
…
```

Il valore della linea di intestazione

`If-modified-since:`

è uguale al valore della linea di intestazione **`Last-Modified:`** che era stata inviata al server tempo prima.

- Questo messaggio di GET condizionale richiede al server di inviare il file solo se è stato modificato dopo la data specificata nella linea **If-modified-since**.
- Supponiamo che l'oggetto non abbia subito modifiche dalla data `Sat, 25 Feb 2012 11:34:56.` Allora:

4. il server Web invia un messaggio di risposta al client:

```
HTTP/1.1 304 Not Modified
Date: Tue, 20 Mar 2012 14:25:26 GMT
Server: Apache/2.2.2 (Unix) PHP/5.1.6

(corpo del messaggio vuoto)
```

Il server Web invia ancora un messaggio di risposta, ma non in inserisce nel corpo del messaggio il file richiesto.

- Il rinvio dell'oggetto richiesto è inutile, poiché nella cache del client è presente una copia aggiornata, e aumenterebbe il tempo di trasferimento dell'oggetto, soprattutto se questo è di grandi dimensioni.
- Il messaggio di risposta dell'esempio contiene nella linea di stato il codice **304** e la descrizione **`Not Modified`**, che indica al client che il file richiesto non è stato modificato e quindi può utilizzare la copia del file presente nella cache.

Interazione user-server: autorizzazione e cookie

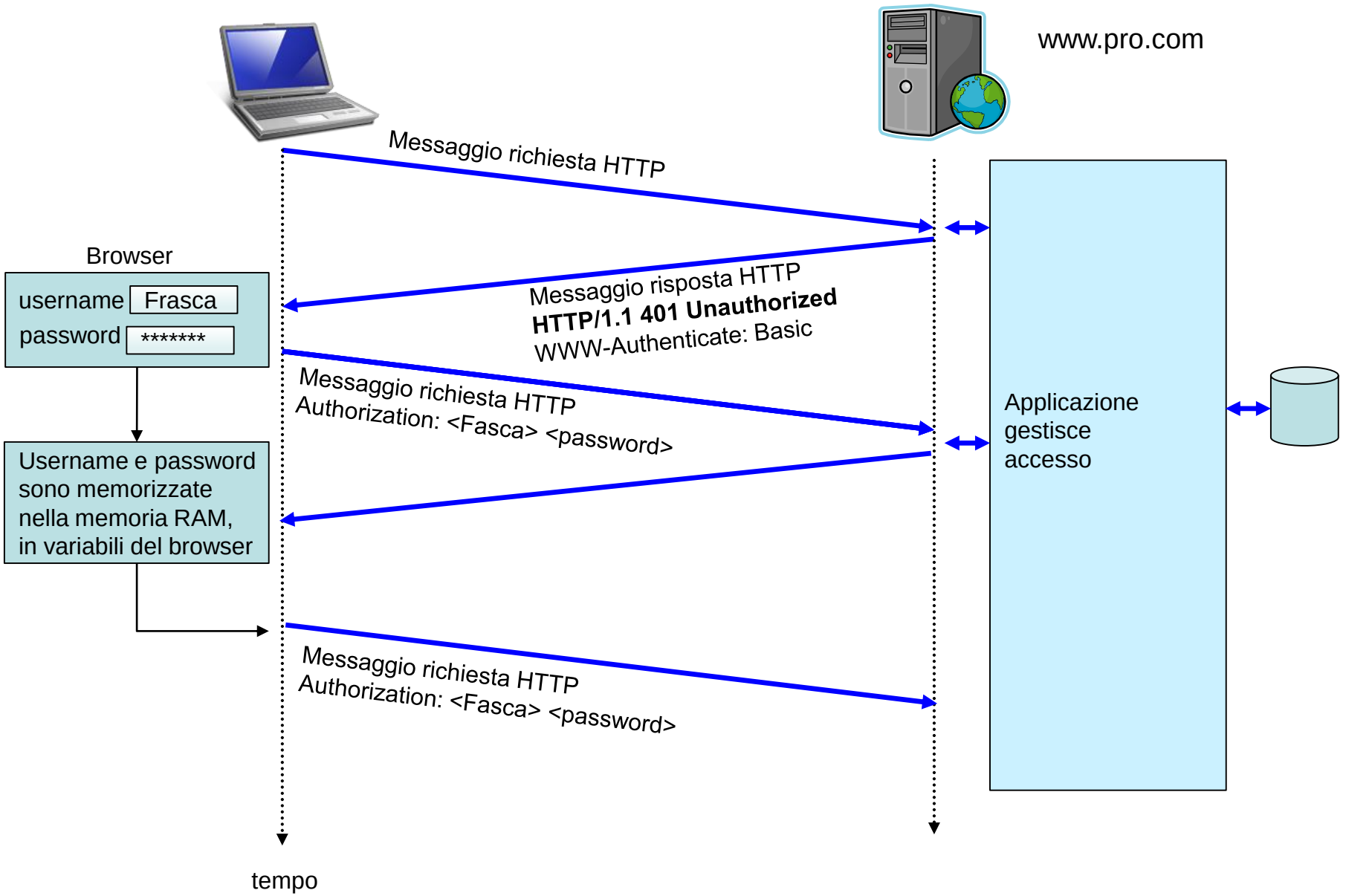
- Il protocollo HTTP è stato progettato ***senza stato*** per semplificare lo sviluppo dei server Web che in tal modo possono gestire migliaia di connessioni TCP contemporaneamente.
- Tuttavia in molte applicazioni web è necessario che un sito Web debba identificare gli utenti, e consentire sessioni di lavoro, come ad esempio nelle applicazioni di commercio elettronico.
- L'HTTP fornisce due meccanismi di identificazione degli utenti: **l'autorizzazione** e i **cookie**.

Autorizzazione

Molti siti richiedono agli utenti di digitare uno ***username*** e una ***password*** per poter accedere alle loro pagine. Questa procedura è chiamata **autorizzazione** *(authorization).*

- Ci sono varie modalità di autorizzazione, più o meno sicure. La più semplice (non sicura) è detta *basic authorization (autorizzazione di base).*
- La richiesta e la risposta, dell'autorizzazione avviene usando speciali intestazioni e codici dell'HTTP. La procedura di autorizzazione si svolge nelle seguenti fasi:

1. Il server risponde a un messaggio di richiesta con un messaggio avente:
   A. La linea di stato con codice di stato **401** e descrizione **Unauthorized;**
   B. l'intestazione **WWW-Authenticate**: che specifica che il client deve fornire uno username e una password;
2. Il client (browser) riceve il messaggio di risposta e vedendo la presenza dell'intestazione **WWW-Authenticate**: visualizza un finestra di dialogo per consentire all'utente di inserire username e password.
3. Il client allora rispedisce il messaggio di richiesta, includendo la linea di intestazione **Authorization**: contenente **username** e **password** inseriti dall'utente.


www.pro.com
Messaggio richiesta HTTP
Browser
username Frasca
password *******
Messaggio risposta HTTP
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic
Messaggio richiesta HTTP
Authorization: <Fasca> <password>
Applicazione
gestisce
accesso
Username e password
sono memorizzate
nella memoria RAM,
in variabili del browser
Messaggio richiesta HTTP
Authorization: <Fasca> <password>
tempo

- Il client continua a inviare username e password nei successivi messaggi di richiesta al server. Lo username e la password sono mantenute in variabili del browser (in memoria ram), in modo che l'utente non debba digitarli ogni volta che chiede un nuovo file.
- In questo modo il sito può identificare l'utente per ciascuna richiesta. Per cancellare lo username e la password è necessario che l'utente chiuda il browser.

Cookie

- I cookie, definiti nella RFC 2965 (obsoleto RFC 2109), sono un meccanismo alternativo che i siti web possono usare per tenere traccia delle attività svolte dagli utenti.
- L'uso dei cookie è molto diffuso, soprattutto nei siti di commercio elettronico.
- Il funzionamento dei cookie si basa su quattro componenti:

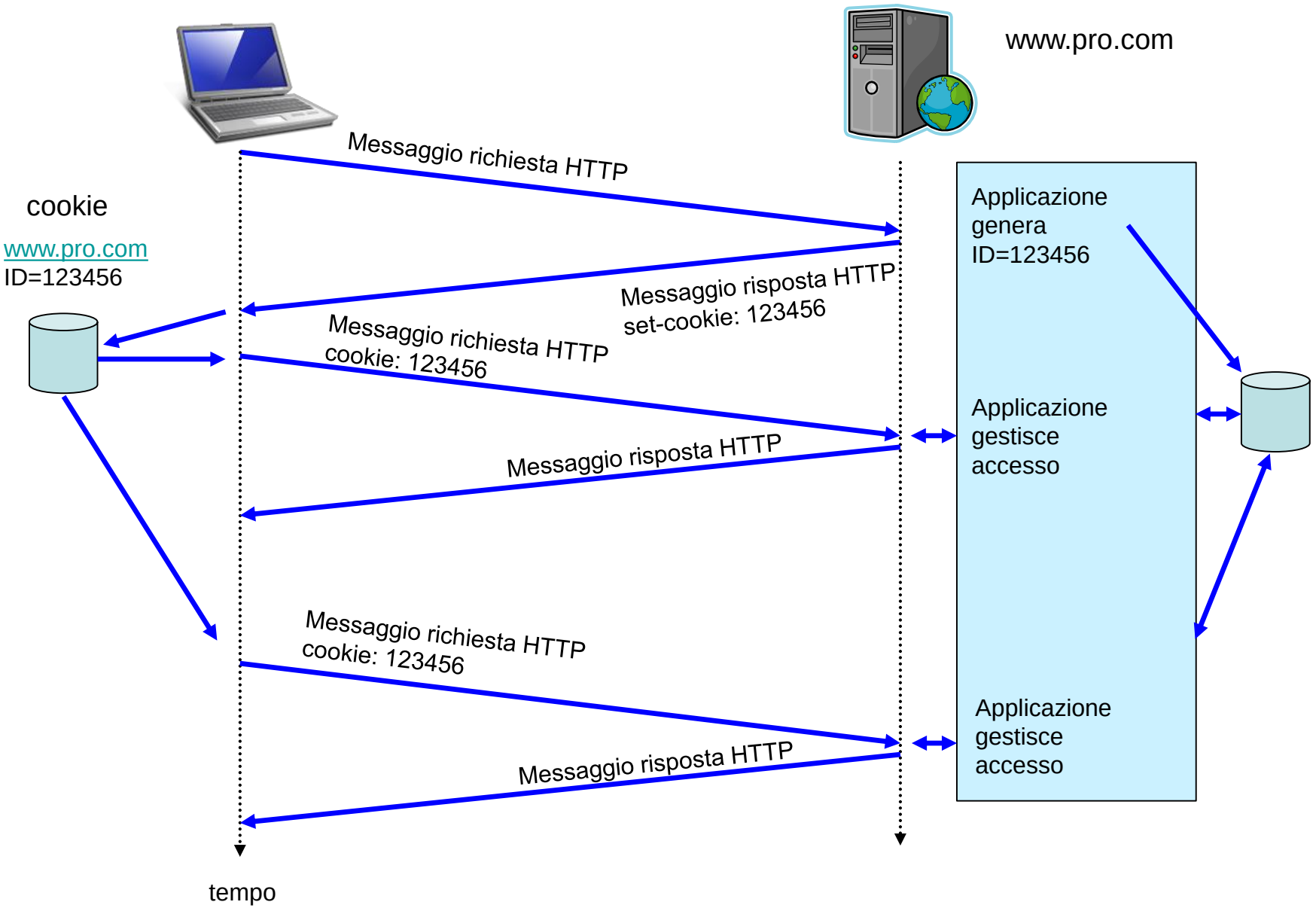
1. una linea di intestazione **Set-cookie**: inserita nel messaggio di risposta HTTP (lato server);
2. una linea di intestazione **Cookie:** presente nel messaggio di richiesta HTTP (lato client);
3. un *file cookie* gestito dal browser dell'utente;
4. un'applicazione di gestione dei cookie nel sito Web.

- Esaminiamo un tipico esempio di come sono utilizzati i cookie. Supponiamo che un utente, acceda per la prima volta ad una pagina di un sito web di commercio elettronico, ad esempio www.pro.com, e che questo sito usi i cookie.

1. Quando un messaggio di richiesta del browser arriva al server web, l'applicazione web **crea un numero di identificazione unico** e inserisce in una tabella di un database una riga la cui chiave è il numero di identificazione stesso.
2. Il server web risponderà al browser, inserendo nel messaggio di risposta HTTP l'intestazione **Set-cookie**: che contiene il numero di identificazione. Ad esempio, la linea di intestazione potrebbe essere:

```
Set-cookie: 123456
```

3. Quando il browser riceve il messaggio di risposta HTTP, vedendo la presenza della linea **`Set-cookie`**, crea e


www.pro.com
Messaggio richiesta HTTP
cookie
www.pro.com
ID=123456
Messaggio risposta HTTP
set-cookie: 123456
Messaggio richiesta HTTP
cookie: 123456
Applicazione
genera
ID=123456
Applicazione
gestisce
accesso
Messaggio risposta HTTP
Messaggio richiesta HTTP
cookie: 123456
Applicazione
gestisce
accesso
Messaggio risposta HTTP
tempo

memorizza in un'apposita cartella un file cookie associato al sito. Nel file cookie sono salvate varie informazioni, comprese **il nome del server** e **il numero di identificazione** contenuto nell'intestazione Set-cookie.

4. Quando l'utente, navigando in questo sito, richiede una pagina, il browser legge il file cookie, relativo a questo sito, estrae il suo numero di identificazione e inserisce nella richiesta HTTP la linea d'intestazione **Cookie:** con valore pari al numero di identificazione. In questo caso, ogni richiesta HTTP al server web contiene la linea:

   Cookie: 123456

   In questo modo, il sito è in grado di registrare le attività che l'utente svolge nel sito web stesso.

- Sebbene il sito web non conosca il nome dell'utente **123456**, sa comunque l'indirizzo IP del suo host, quali pagine ha visitato, in quale ordine, e a che ora.
- Il sito di commercio elettronico può quindi usare i cookie per realizzare *un servizio di carrello* per gli acquisti gestendo una

sessione di lavoro, mantenendo una lista di tutti gli acquisti dell'utente.

- Se tempo dopo, l'utente ritorna a visitare il sito, il suo browser continuerà a inserire la linea di intestazione **Cookie: 123456** nei messaggi di richiesta. Il sito può allora consigliare i prodotti a questo utente in base alle pagine che ha visitato in passato.
- Se l'utente si registra nel sito, fornendo i suoi dati anagrafici, fiscali etc., il sito può associare l'identità di questo utente al suo numero di identificazione.
- Pertanto, i cookie consentono di creare uno **strato di sessione di utente sovrapposto all'HTTP** che è senza stato.
- Questi cookie, ***cosiddetti tecnici***, sono spesso utili, perché usati per rendere più veloce la navigazione e la fruizione delle informazioni, rendere possibile, come sopra descritto, la realizzazione di procedure per l'autenticazione e per gli acquisti online.

- Sebbene i cookie consentano agli utenti di svolgere operazioni di acquisto on-line, essi possono anche essere usati per raccogliere informazioni sul comportamento degli utenti attraverso un grande numero di siti web.
- Una particolare tipologia di cookie, detti ***cookie analytics***, sono utilizzati dai gestori dei siti web per raccogliere informazioni, e quindi elaborare statistiche sul servizio e sul suo utilizzo.
- Inoltre, i cookie possono anche essere usati per monitorare e classificare gli utenti studiando i loro movimenti e le loro preferenze di consultazione e consumo sul web (ad esempio, registrando quali prodotti comprano, quali fonti d'informazione leggono, ecc.).
- Tale operazione di marketing, detta spesso di ***profilazione***, ha lo scopo di inviare pubblicità di prodotti e servizi mirate e personalizzate agli utenti in accordo alle preferenze che hanno manifestato durante la navigazione sul web.
- In questo caso i cookie sono definiti con il termine ***cookie di profilazione***.

- Può accadere anche che una pagina web contenga cookie provenienti da altri siti e contenuti in vari elementi ospitati sulla pagina stessa, come ad esempio banner pubblicitari, immagini, video, ecc.. In questo caso, i cookie sono etichettati come ***cookie di terze parti***, che di solito sono utilizzati a fini di profilazione.
- Così, i cookie scaricati sui dispositivi degli utenti possono anche essere letti da altri soggetti, diversi dai gestori delle pagine web al momento visitate.
- Considerata la sostanziale invasività che i cookie di profilazione possono avere per la privacy degli utenti, la normativa europea e italiana prevede che i gestori di siti web forniscano adeguate informazione all'utente richiedendogli un esplicito consenso per l'accettazione dell'uso dei cookie.
- In particolare, con il provvedimento *Individuazione delle modalità semplificate per l´informativa e l´acquisizione del consenso per l´uso dei cookie* dell' 8 maggio 2014 il Garante per la protezione dei dati personali

ha stabilito che quando si visitano le pagine di un sito web che usa cookie per finalità di profilazione e marketing deve immediatamente comparire un banner ben visibile, in cui sia indicato chiaramente:

- che il sito utilizza cookie di profilazione per inviare messaggi pubblicitari mirati;
- che il sito consente anche l´invio di *cookie di terze parti*;
- un link a una informativa dettagliata, con le indicazioni sull´uso dei cookie inviati dal sito;
- l´indicazione che proseguendo nella navigazione del sito si accetta l´uso dei cookie.

Connessione non persistente e persistente

- Il TCP offre alle applicazioni un servizio di trasferimento di dati affidabile. Il client richiede la connessione al server, dopo la fase di *handshake* il TCP crea una connessione full-duplex, punto-punto tra le due estremità. Le due parti possono scambiarsi i dati contemporaneamente nelle due direzioni, fino a quando una delle due estremità chiude la connessione.
- L'HTTP usa la connessione TCP in due modalità dette connessione **persistente** e **non persistente**.
- La versione HTTP/1.1 e HTTP/2 usano le **connessioni persistenti** nella modalità di default. Le connessioni persistenti, consentono di trasferire più file con un'unica connessione TCP.
- La vecchia versione HTTP/1.0 invece usa la connessione non persistente che consente di trasferire un singolo file per ogni connessione TCP. Tuttavia, è possibile configurare sia i client che i server HTTP/1.1 o HTTP/2 per utilizzare le connessioni non persistenti.

-

Connessione non persistente

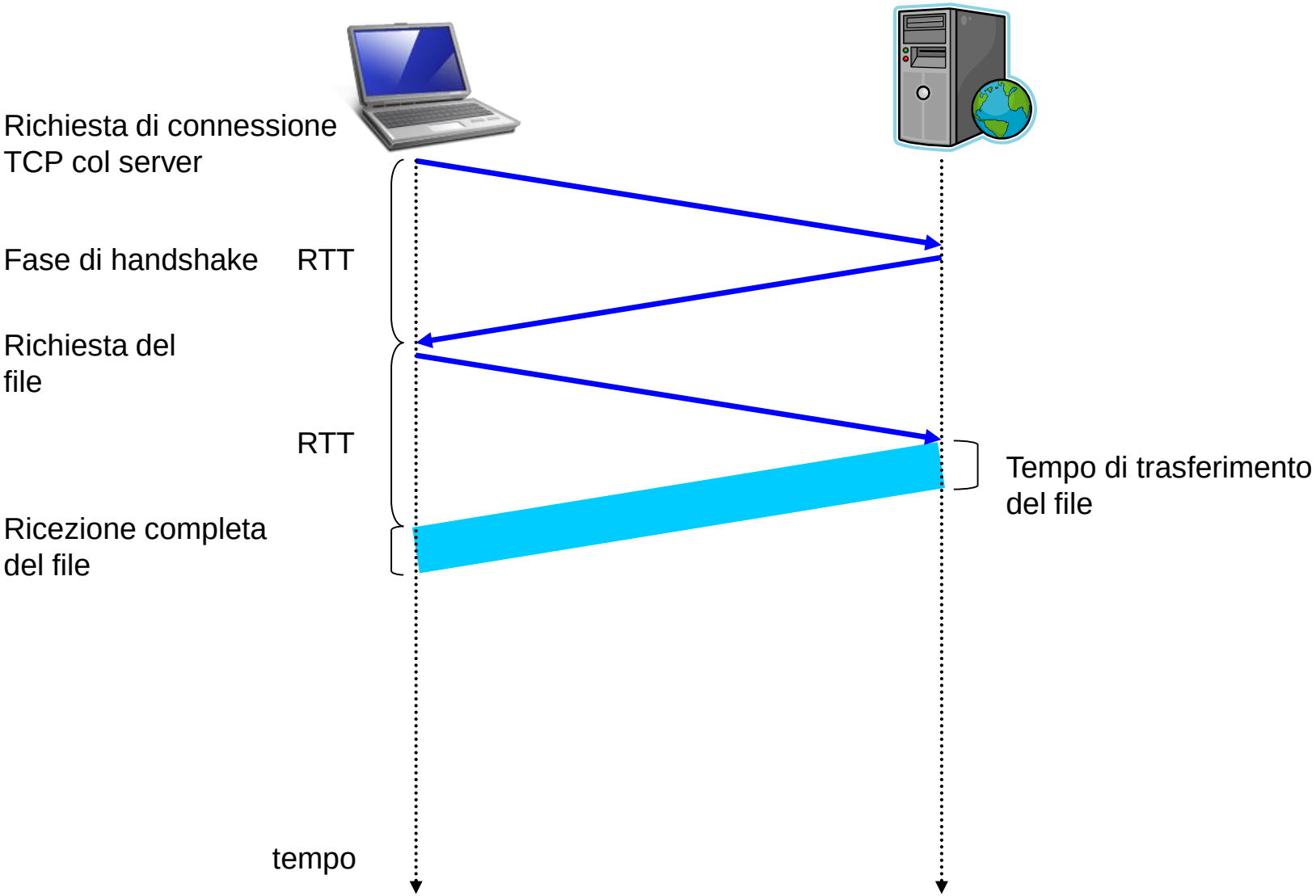
Descriviamo con un esempio come avviene il trasferimento di file con la connessione **non persistente**. Supponiamo che un client richieda una pagina formata da un file base HTLM e da 10 immagini JPEG, per un totale di 11 file. La transazione HTTP avviene nel seguente modo:

1. Il browser instaura una connessione TCP col server (ad esempio **www.pf.uniroma2.it)** alla porta **80**.
2. Il browser invia un messaggio di richiesta HTTP.
3. Il server HTTP riceve il messaggio di richiesta, cerca il file richiesto e se lo trova, lo inserisce in un messaggio di risposta HTTP che invia al client.
4. Il server web **chiude la connessione TCP** instaurata con il client.
5. Il client riceve il messaggio di risposta e riconosce che il file ricevuto è un file HTML, quindi lo interpreta e trova i riferimenti ai 10 oggetti JPEG. La connessione TCP si chiude.
6. La sequenza relativa ai passi 1-4 viene ripetuta per ciascuno dei file JPEG.

Listato del file html dell'esempio:

```
<h1> Esempio di connessione <b>non persistente</b></h1>
<img src="figura1.jpg"> <img src="figura2.jpg">
<img src="figura3.jpg"> <img src="figura4.jpg">
<img src="figura5.jpg"> <img src="figura6.jpg">
<img src="figura7.jpg"> <img src="figura8.jpg">
<img src="figura9.jpg"> <img src="figura10.jpg">
```

- Notiamo che ciascuna connessione TCP trasporta un messaggio di richiesta e un messaggio di risposta. Quindi, in questo esempio, quando un utente richiede una pagina, si generano **11 connessioni TCP**.
- Il browser (client) può ottenere le 10 immagini JPEG su 10 connessioni TCP in serie, o su più connessioni TCP in parallelo (fino ad un numero massimo prestabilito). Generalmente i browser possono essere configurati con un numero massimo di connessioni parallele.
- L'uso di connessioni in parallelo diminuisce i tempi di risposta.


Richiesta di connessione TCP col server
Fase di handshake
RTT
Richiesta del file
RTT
Tempo di trasferimento del file
Ricezione completa del file
tempo

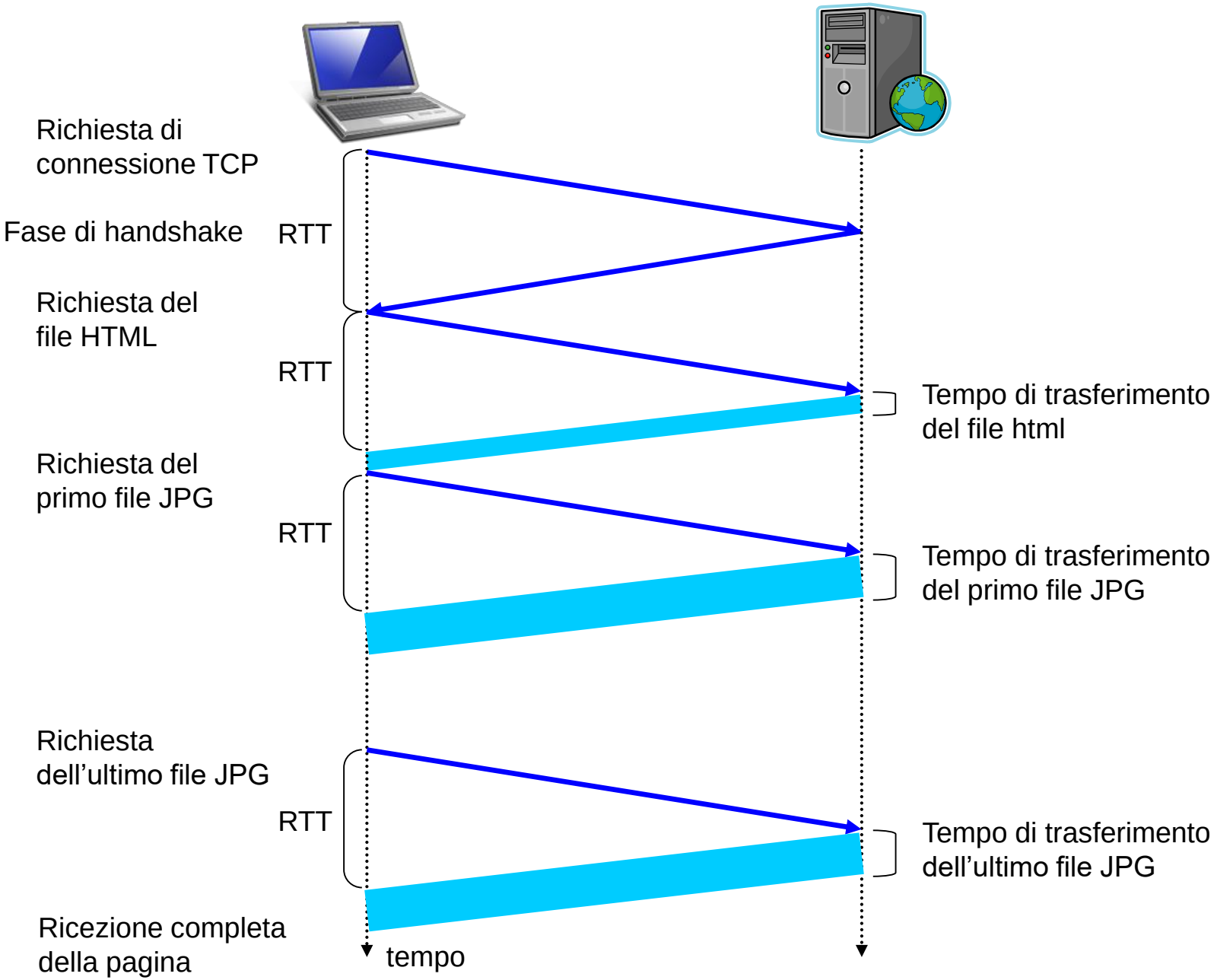
- Il server, quando riceve il messaggio di richiesta, invia il file HTML mediante il TCP. Il tempo che passa dall'inizio della richiesta all'ottenimento della risposta HTTP è di circa due **RTT** più il tempo di trasferimento del file HTML.
- Con **RTT** (**Round Trip Time, Tempo di andata e ritorno)** si indica l'intervallo di tempo che passa da quando un piccolo messaggio (come ad esempio il messaggio di richiesta di connessione), parte dal client, arriva al server e ritorna al client.
- Le connessioni non persistenti hanno alcuni svantaggi.
  - per ogni file richiesto è necessario instaurare una nuova connessione TCP che richiede buffer e variabili sia sul client che sul server. Se il server web deve soddisfare richieste di centinaia di client è evidente che risulterà fortemente sovraccarico;
  - prima che inizi il trasferimento di ogni file devono trascorrere due RTT.

Connessione persistente

- La connessione persistente, consente di trasferire più file con un'unica connessione TCP. Ad esempio, la pagina Web del caso precedente, costituita da un file html e da 10 immagini jpeg, può essere inviata su un'unica connessione TCP persistente.
- La chiusura della connessione avviene quando il server HTTP rileva che la connessione stessa non è usata da un certo tempo, detto **intervallo di timeout**.
- Esistono due modalità di funzionamento della connessione persistente: **senza parallelismo** e **con parallelismo**.

Connessione persistente senza parallelismo

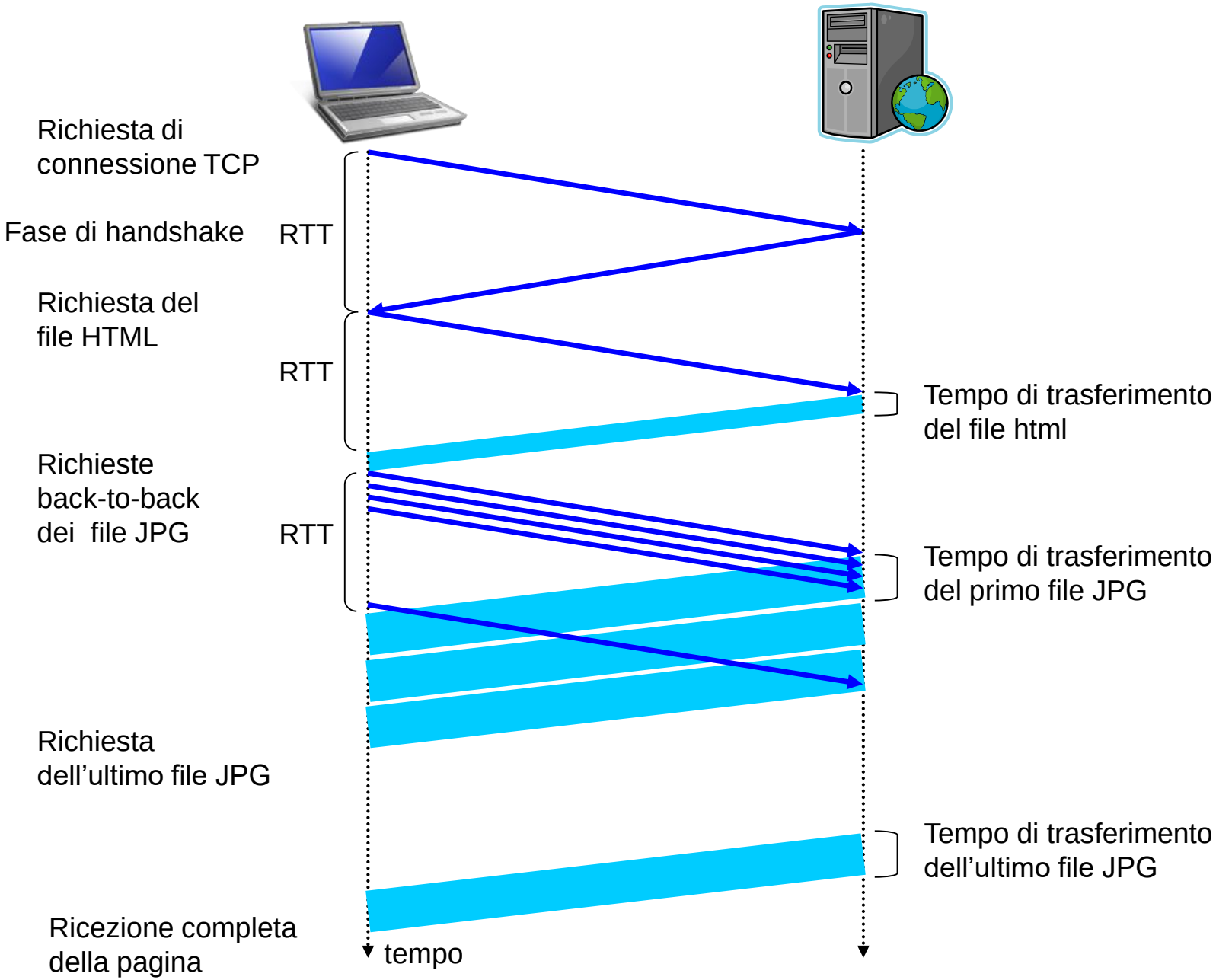
- Nella modalità persistente senza parallelismo, il client può inviare una nuova richiesta solo quando la risposta precedente è stata ricevuta.


Richiesta di connessione TCP
Fase di handshake
RTT
Richiesta del file HTML
RTT
Tempo di trasferimento del file html
Richiesta del primo file JPG
RTT
Tempo di trasferimento del primo file JPG
Richiesta dell'ultimo file JPG
RTT
Tempo di trasferimento dell'ultimo file JPG
Ricezione completa della pagina
tempo

- In questo caso prima che inizi il trasferimento di ciascuna delle 10 immagini, dell'esempio precedente, deve passare **un RTT**.

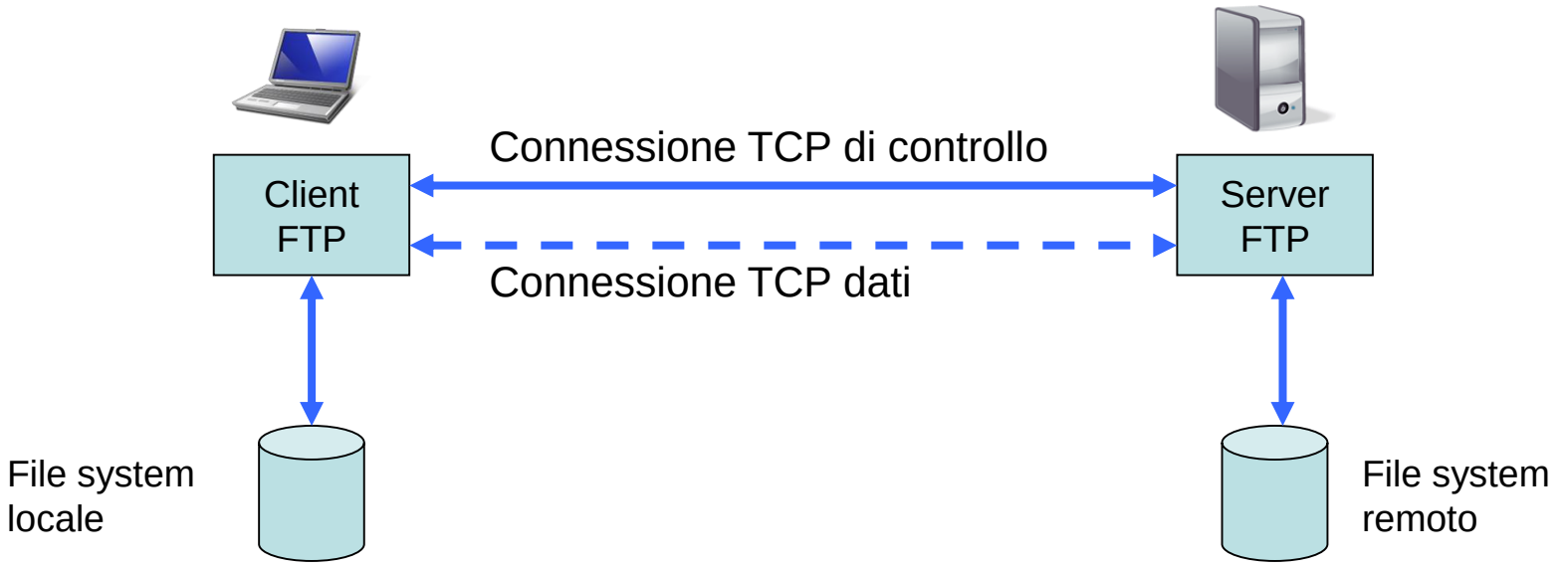
Connessione persistente con parallelismo

- Con la **connessione persistente con parallelismo,** il client può inviare una richiesta appena trova nella pagine HTML base un riferimento di file, senza attendere la risposta alla precedente richiesta.
- Quando il server riceve le richieste, invia i file uno dopo l'altro.
- Il parallelismo, quindi, utilizza **un solo RTT per tutti i file richiesti** invece di un RTT per ciascun file come avviene quando non si usa il parallelismo.
- Inoltre la connessione TCP con parallelismo resta attiva per tempo più piccolo rispetto a quella senza parallelismo.
- Per default**, l'HTTP/1.1** usa la connessione persistente con parallelismo.


Richiesta di connessione TCP
Fase di handshake
RTT
Richiesta del file HTML
RTT
Tempo di trasferimento del file html
Richieste back-to-back dei file JPG
RTT
Tempo di trasferimento del primo file JPG
Richiesta dell'ultimo file JPG
Tempo di trasferimento dell'ultimo file JPG
Ricezione completa della pagina
tempo

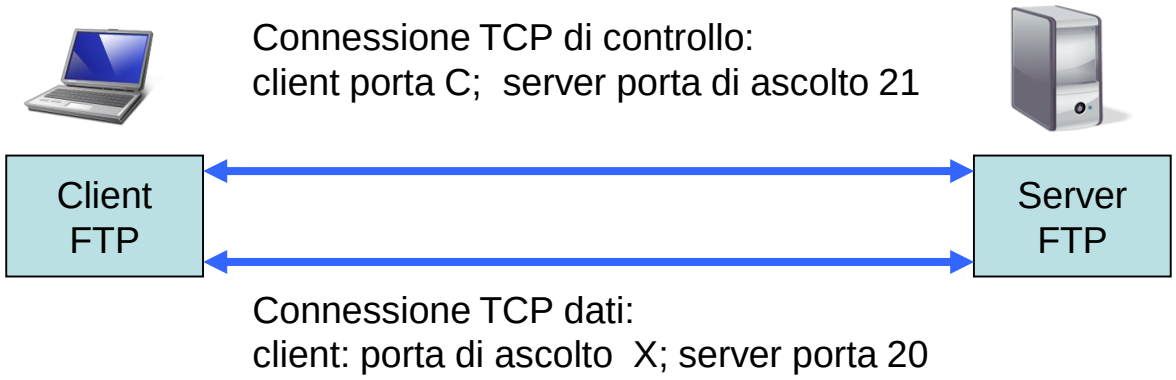
Il protocollo FTP

- Il protocollo FTP (**F**ile **T**ransfer **P**rotocol, *protocollo di trasferimento file*), consente di trasferire file da un host (locale) ad un altro host (remoto) e viceversa.
- FTP si è evoluto negli anni. La prima specifica (RFC 114), sviluppata presso il **MIT** (Massachusetts Institute of Technology), è del 1971. L'attuale specifica (RFC 959) è del 1985.

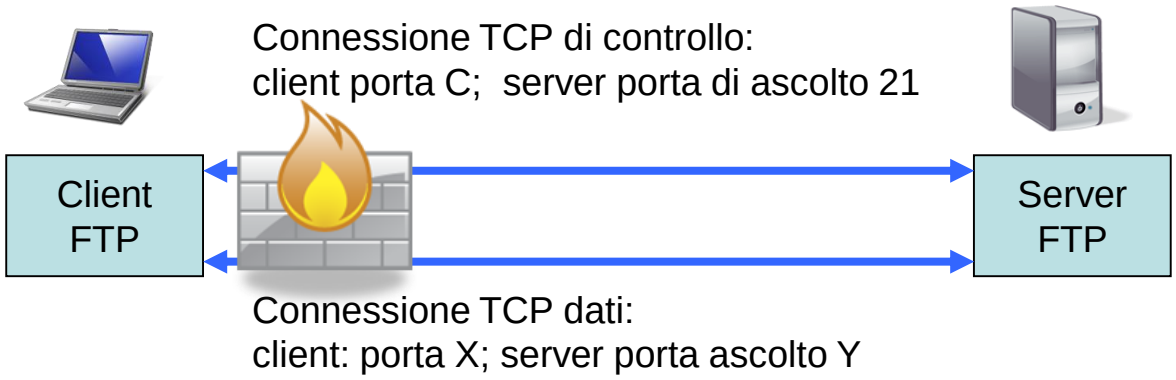


- Anche l'FTP, come l'HTTP, utilizza il TCP. l'FTP usa due connessioni TCP in parallelo: una **connessione di controllo** *per inviare i comandi* e una **connessione dati** per il trasferimento dei file.
- Il client prima instaura una connessione TCP di controllo con il server sulla **porta 21**. Il numero di porta di connessione del client è scelto automaticamente dal TCP che assegna il primo numero non usato superiore a 1024.
- Il client invia l'identificazione dell'utente e la password.
- Quando il client invia un comando per un trasferimento di file al server, viene creata una connessione dati TCP che sarà usata per il trasferimento di file. L'FTP invia un file sulla connessione dati e poi la chiude. Se, durante la stessa sessione, l'utente vuole trasferire un altro file, l'FTP apre un'altra connessione dati. Quindi, con FTP, la connessione di controllo rimane aperta per tutta la durata della sessione, ma viene stabilita una nuova connessione dati per ciascun file trasferito all'interno della sessione cioè, **la connessione dati non è persistente**.

- FTP può essere eseguito in **modalità attiva o passiva**, che determina il modo in cui viene stabilita la connessione dati.
- In **modalità attiva**, il client attraverso la connessione di controllo, invia al server mediante il comando PORT il numero di porta (listen port), ad esempio **X** su cui esso ascolterà e il server avvia la connessione dati.



- In situazioni in cui il client è protetto da un firewall e non può accettare le connessioni TCP in ingresso, può essere utilizzata la **modalità passiva**. In questa modalità, il client invia il comando PASV al server e riceve da questo, mediante il comando PORT, un numero di **porta Y** di ascolto che il client utilizza per aprire la connessione dati al server.



- Il trasferimento dei dati può avvenire in quattro modalità, in base al formato dei dati:
  - **Modalità ASCII**: utilizzato per il testo. I dati se necessario vengono convertiti al formato ASCII a 8-bit prima della trasmissione e (ancora una volta, se necessario) nella rappresentazione di caratteri dell'host ricevente. Di conseguenza, questa modalità non è adatta per i file che contengono dati diversi da testo normale.
  - **Modalità immagine** (detta anche modalità **binaria**): l'host mittente invia il file byte per byte e il destinatario salva il flusso di byte appena lo riceve.
  - **Modalità EBCDIC:** utilizzata per il testo normale tra gli host che utilizzano il set di caratteri EBCDIC. Questa modalità è altrimenti come la modalità ASCII.
  - **Modalità locale:** permette a due computer con configurazioni identiche di inviare dati in un formato proprietario senza la necessità di convertirlo in ASCII

- Durante una sessione, l'FTP deve mantenere lo stato della sessione dell'utente. In particolare, deve tenere traccia delle directory, locale e remota, mentre l'utente naviga all'interno del file system locale e remoto. Dovendo tenere traccia di queste informazioni di stato e mantenere aperta sempre la connessione di controllo, i SO limitano il numero totale di sessioni che l'FTP può aprire simultaneamente.

Comandi FTP

- Il client e il server si scambiano messaggi di richiesta e risposta, attraverso la connessione di controllo (porta 21), in formato ASCII a sette bit. Quindi, come per i messaggi HTTP, i comandi FTP sono leggibili.
- Ogni messaggio di richiesta (comando) ha un formato costituito da un massimo di quattro caratteri maiuscoli ASCII.
- Alcuni dei comandi più usati sono mostrati nella tabella seguente:

Comando	descrizione
CWD path	Cambia directory corrente
DELE path	Cancella il file specificato in path.
HELP	Ritorna la lista dei comandi accettati dal server. Con argomento fornisce spiegazioni riguardo il comando specificato.
LIST [path]	Visualizza un elenco di tutti i file contenuti nella directory remota. L'elenco dei file viene spedito su una nuova e non persistente connessione dati TCP.
PASS password	Invia la password al server
PASV	Inizializza una connessione passiva.
PWD	Visualizza la directory corrente remota.
QUIT	Chiude la sessione.
RETR path	trasferisce *(get)* un file dalla directory remota corrente alla directory corrente locale.
STOR path	trasferire *(put)* un file nella directory corrente dell'host remoto.
USER username	Invia l'identificativo dell'utente al server.