

# Reti di Telecomunicazione

## Lezione 7

---

---

---

Marco Benini  
Corso di Laurea in Informatica

[marco.benini@uninsubria.it](mailto:marco.benini@uninsubria.it)



# Programma della lezione

- Il protocollo FTP
  - file transfer protocol
  - descrizione architetturale
  - descrizione funzionale
- Realizzazione
  - protocollo di trasporto: TCP
  - trasmissione fuori dalla banda
  - protocollo con stato

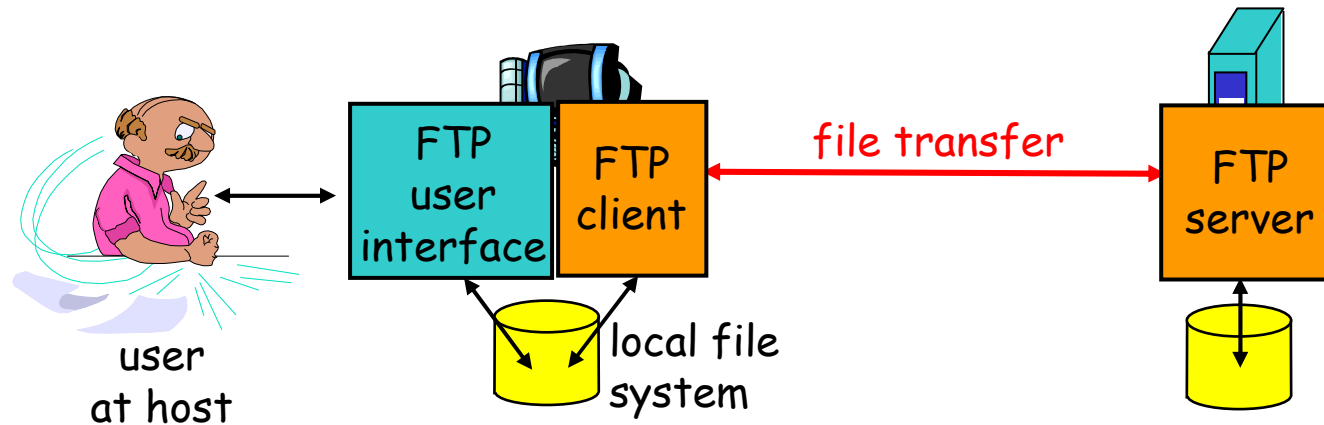


# File Transfer Protocol

- FTP = File Transfer Protocol
  - Un protocollo per muovere file da host a host
  - Definito nel 1984 nell'RFC 959
- Modello client/server
  - il client inizia la comunicazione
  - il client richiede di trasferire un file sul server
  - il client richiede di trasferire un file dal server
  - il client può richiedere di ottenere la lista dei file disponibili sul server
  - il client può muoversi nelle directory del server



# Descrizione architetturale



- trasferimento di file da/a host remoto
- modello client/server
- client: il lato che inizia il trasferimento
- server: l'host remoto, che fornisce il file
- FTP: descritto in RFC 959
- ftp server: porta TCP 21 (*ben nota*)



# Descrizione funzionale

- Un client è dotato di un programma
  - parte di comunicazione: implementa il protocollo FTP
  - interfaccia utente: permette di svolgere le funzioni che il protocollo offre
- Il programma svolge le seguenti funzioni:
  - connessione al server remoto
  - trasferimento di un file dal server al client (**get**)
  - trasferimento di un file dal client al server (**put**)
  - lettura dei files presenti nella directory corrente sul server (**dir**)
  - cambiamento di directory corrente sul server (**cd**)
  - disconnessione (**bye**)



# Descrizione funzionale

- Il server è dotato di un programma, detto ftp server (ftp daemon)
  - parte di comunicazione: implementa il protocollo FTP
  - parte user agent:
    - implementa l'accesso al file system
    - implementa l'autenticazione del client
    - permette la configurazione del servizio (chi può leggere che cosa, chi può scrivere nel file system e dove, politica di logging, ...)
- Il server svolge le seguenti funzioni
  - riceve le richieste di connessione e verifica se sono ammissibili in base alla propria configurazione
  - riceve i comandi relativi alle varie funzioni del client e le espleta, se sono autorizzate



# Descrizione funzionale

- Comandi principali
  - sono inviati come testo ASCII
  - **USER** <nome utente> (identifica l'utente)
  - **PASS** <password> (autentica l'utente)
  - **LIST** (lista i file nella directory corrente)
  - **RETR** <filename> (trasferisce il file dal server)
  - **STOR** <filename> (trasferisce il file al server)
  - **CWD** <directory> (cambia directory corrente)
  - **QUIT** (esegue la disconnessione)
- Esistono altri comandi, che noi non tratteremo...



# Descrizione funzionale

- Tipiche risposte dal server
  - in ASCII, un codice di stato ed una descrizione
  - **331** Username OK, password required
    - nome utente accettato, si attende la password
  - **125** data connection already open; transfer starting
    - connessione dati aperta, si inizia il trasferimento del file
  - **425** Can't open data connection
    - errore: non si riesce ad aprire una connessione dati (impossibile trasferire un file)
  - **452** Error writing file
    - errore: non è possibile scrivere il file
- Esistono numerose risposte, che noi non elenchiamo



# Descrizione funzionale

- Ciclo fondamentale d'utilizzo
  - il client esegue una connessione al server collegandosi alla porta TCP 21
  - il server risponde
  - Il client invia i dati necessari per identificare l'utente
    - USER login
    - PASS password
  - il server risponde che l'utente è correttamente autenticato, oppure nega l'accesso
  - se l'utente è autenticato, il client può dare i vari comandi previsti dal protocollo (\*)
  - l'utente si disconnette
    - QUIT



# Descrizione funzionale

- I comandi al punto (\*), che implicano un trasferimento di informazione seguono tutti lo schema
  - il client invia il comando
    - RETR filename
  - il server risponde con un codice di stato ed una descrizione
    - 125 Data connection already open; transfer starting
  - se il codice non è di errore ed il comando lo richiede, chi possiede il file richiesto lo invia alla controparte
    - se il comando è RETR filename, il server invia il file al client
    - se il comando è STOR filename, il client invia il file al server
    - se il comando è LIST, il server invia la lista della directory corrente al client
    - se il comando è CWD, non vi è scambio di dati
- Il trasferimento di file è uno scambio di dati, mentre il colloquio tra client e server è uno scambio di comandi



# Realizzazione

- La connessione deve essere affidabile
  - il flusso dei comandi e delle risposte avviene su un canale TCP
- Il server è in ascolto, in attesa di client che vogliono connettersi
  - il server deve essere collegato ad una porta che sia nota al client
  - rendendo standard la porta per il servizio FTP posso realizzare l'applicazione di rete di trasferimento file
  - la porta su cui il server ascolta è TCP 21
- Quando un client si connette, il server inizia la procedura di autenticazione

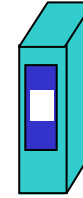


# Realizzazione

- Procedura di connessione e di autenticazione



FTP  
client



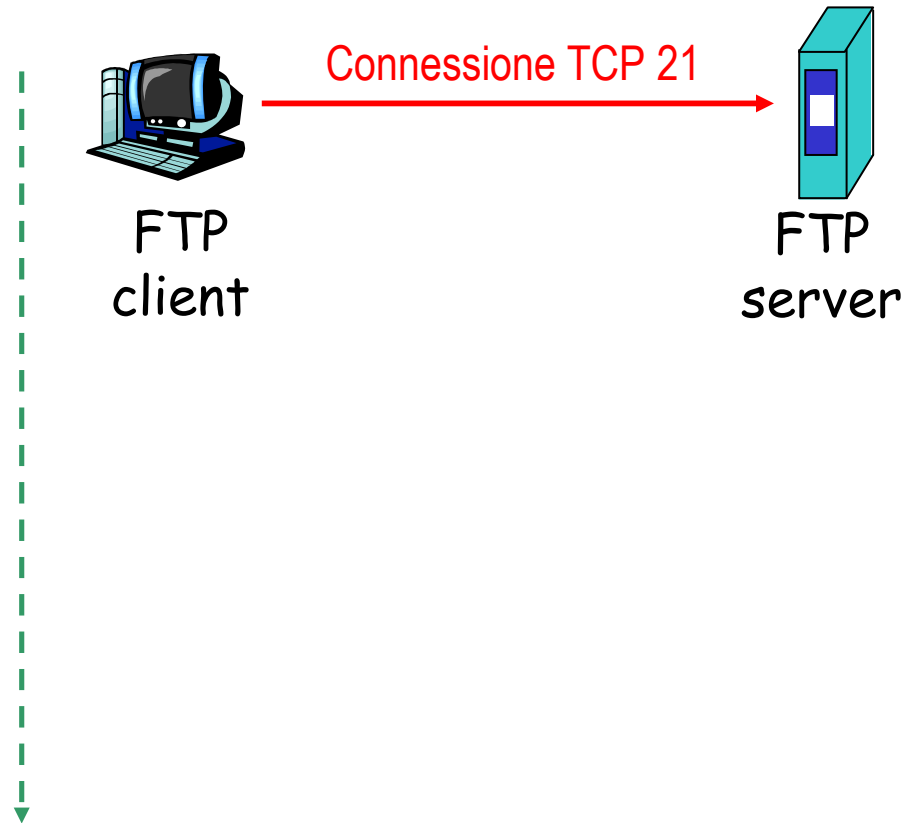
FTP  
server





# Realizzazione

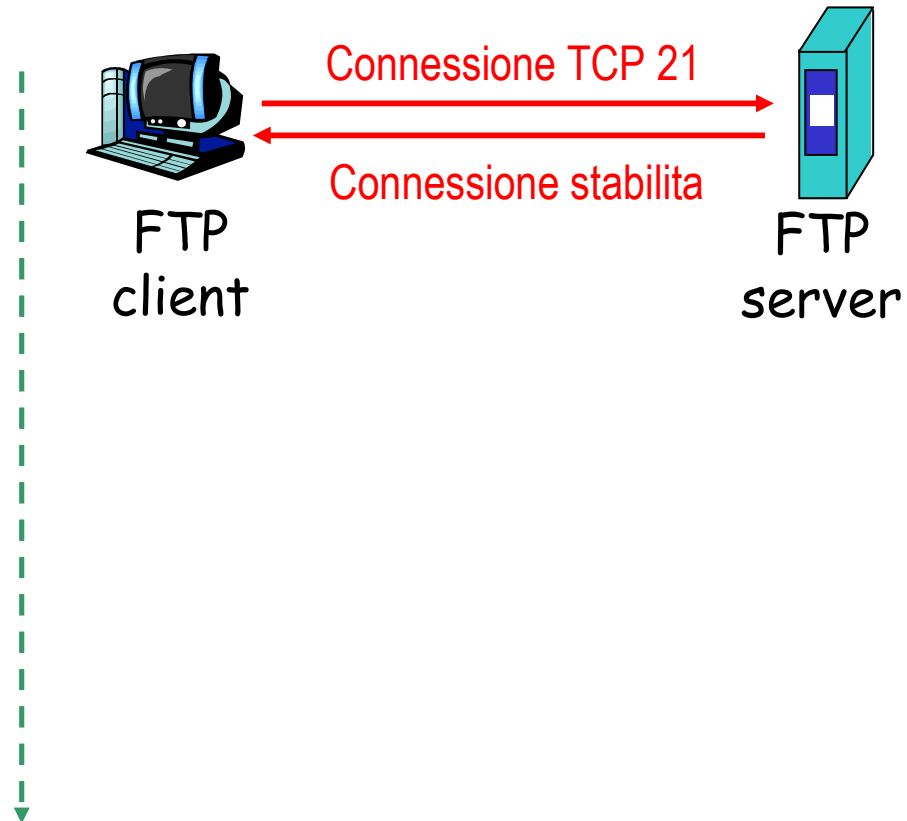
- Procedura di connessione e di autenticazione





# Realizzazione

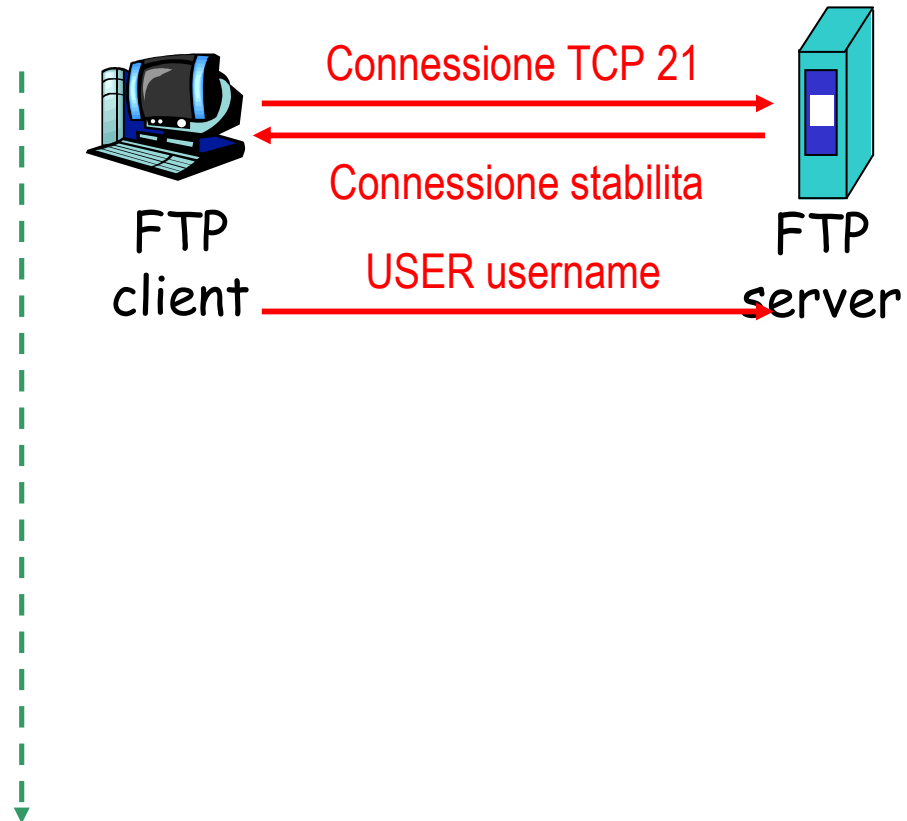
- Procedura di connessione e di autenticazione





# Realizzazione

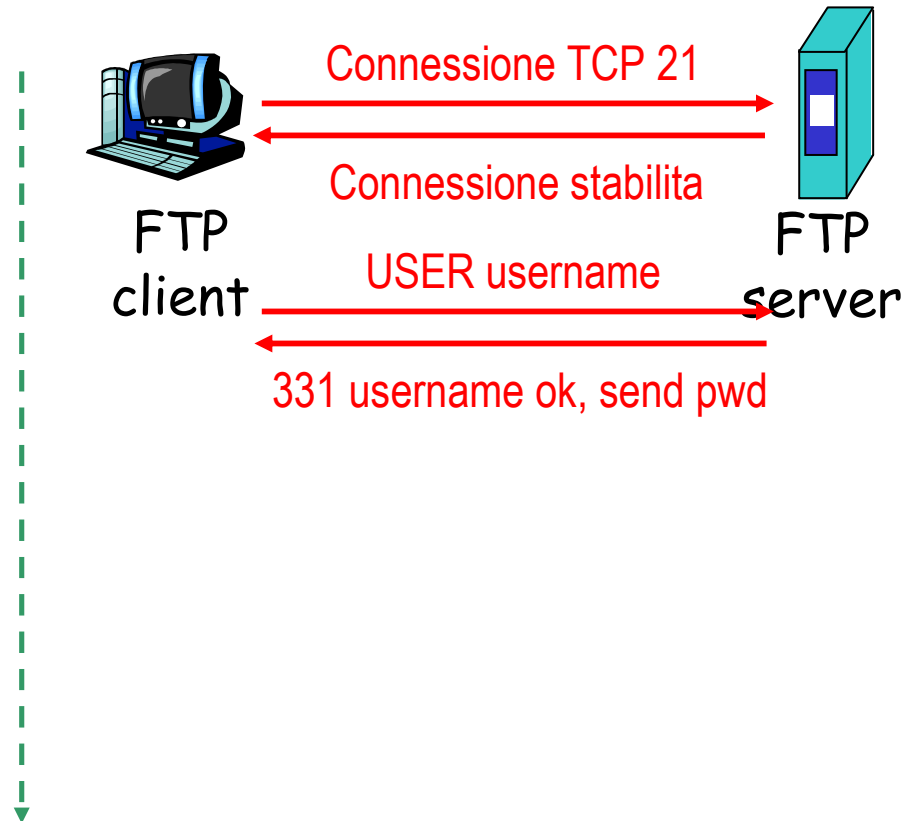
- Procedura di connessione e di autenticazione





# Realizzazione

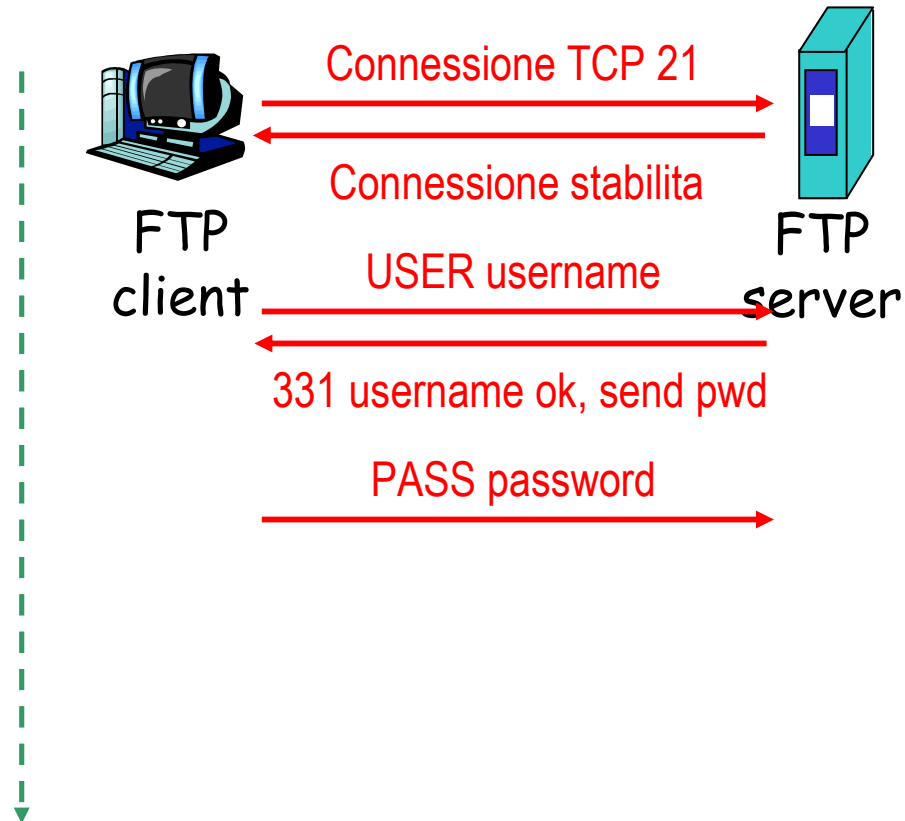
- Procedura di connessione e di autenticazione





# Realizzazione

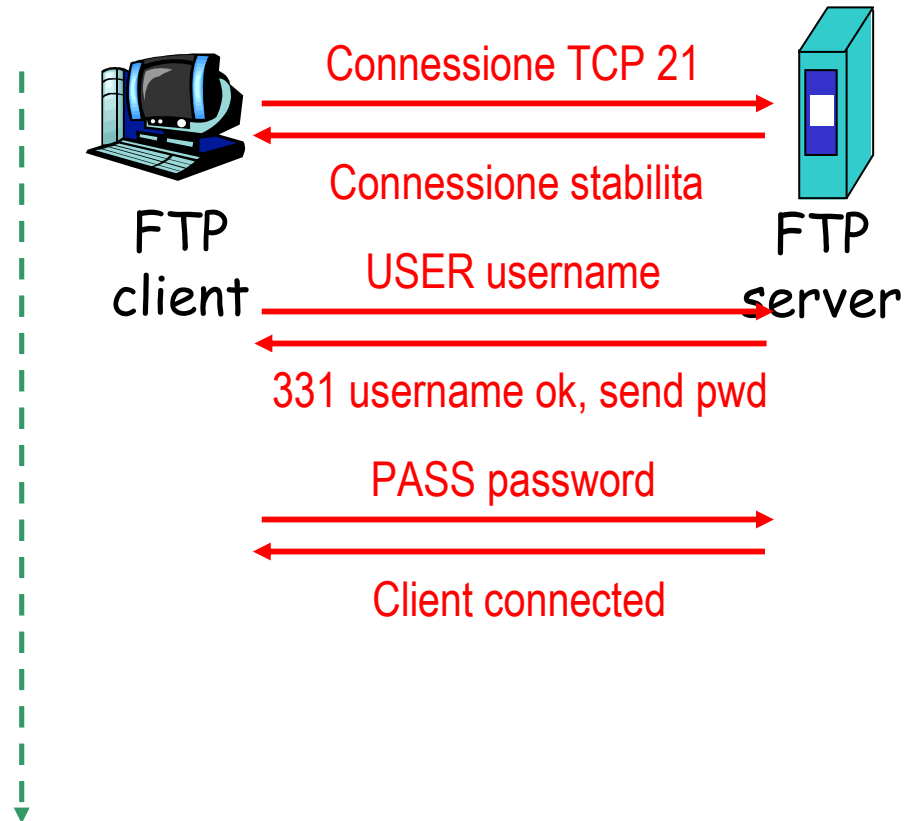
- Procedura di connessione e di autenticazione





# Realizzazione

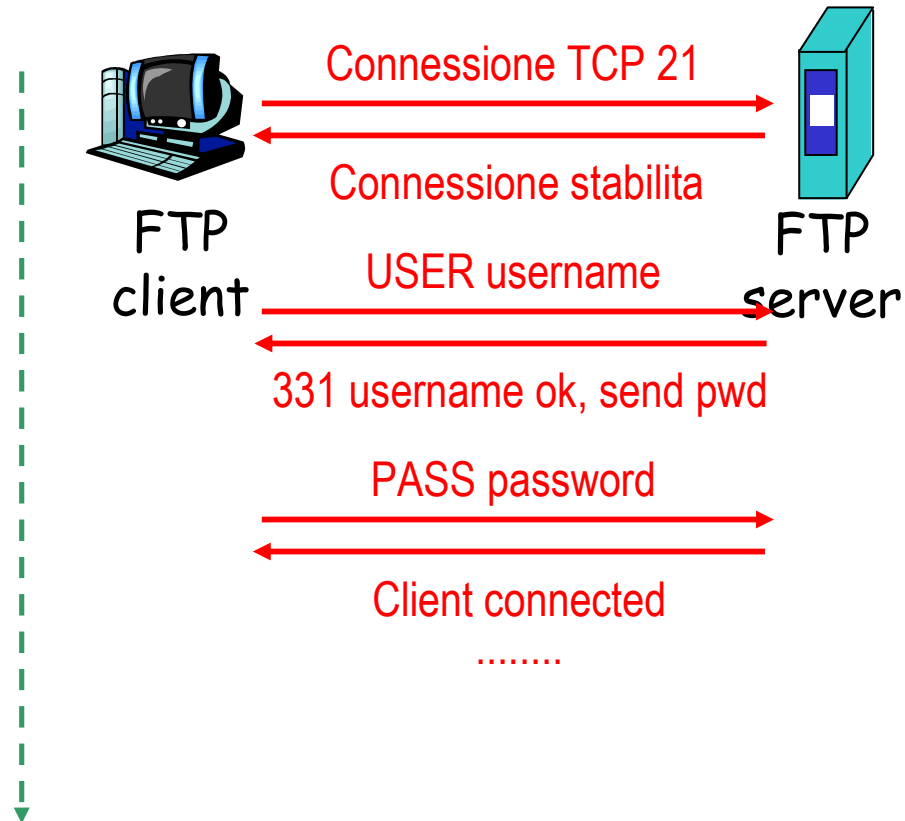
- Procedura di connessione e di autenticazione





# Realizzazione

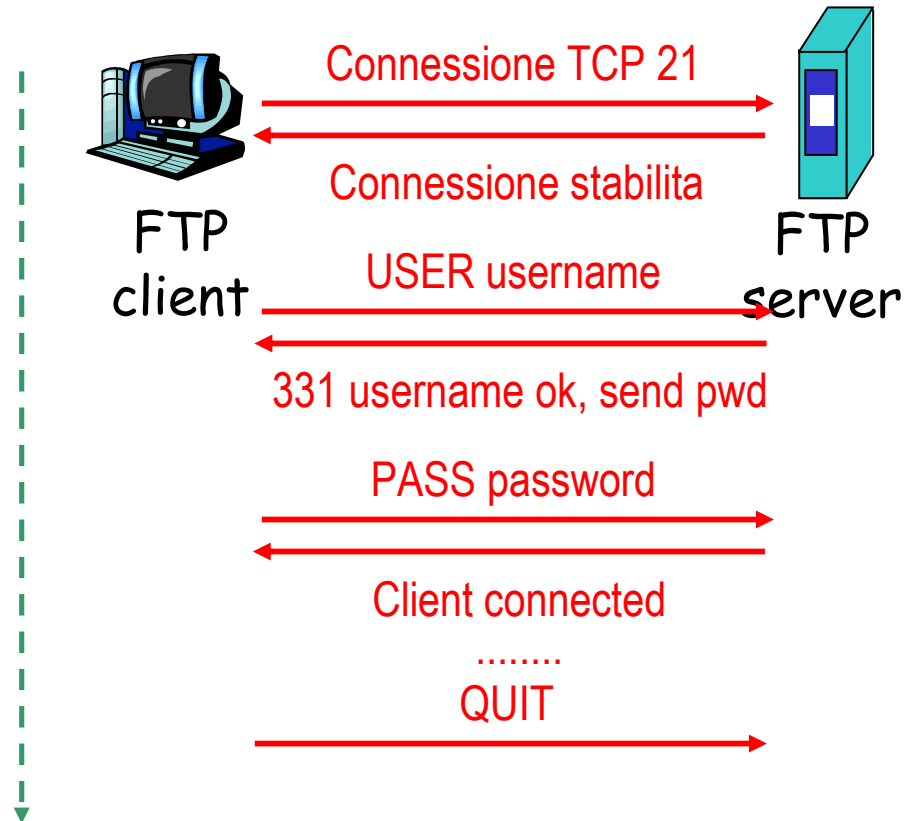
- Procedura di connessione e di autenticazione





# Realizzazione

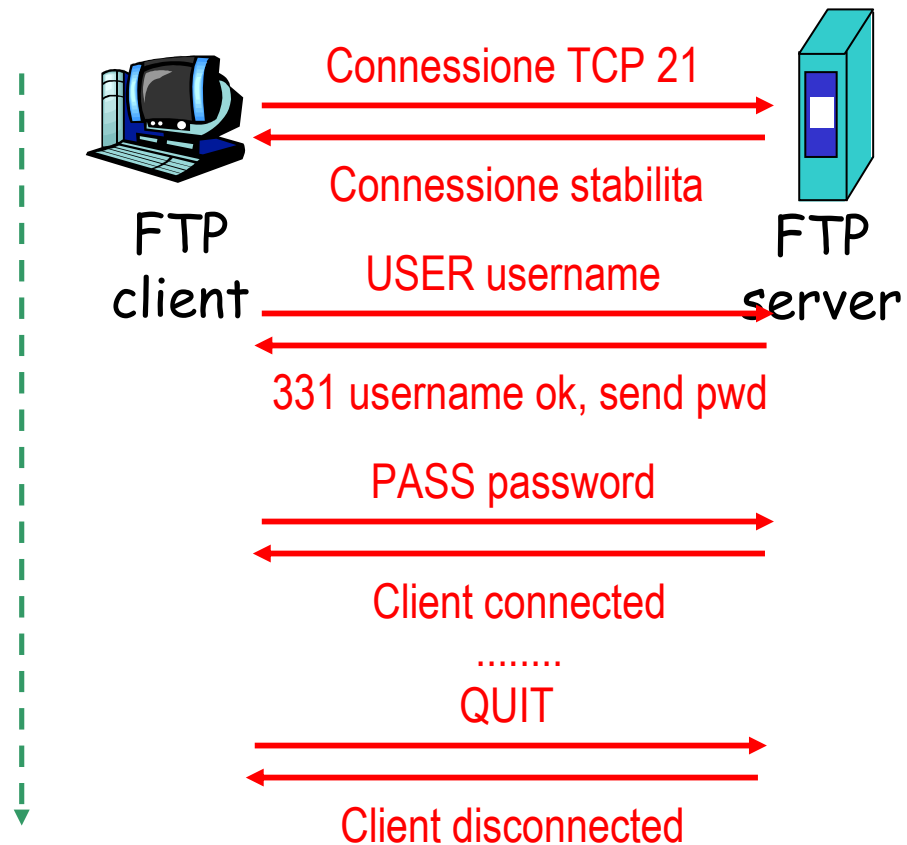
- Procedura di connessione e di autenticazione





# Realizzazione

- Procedura di connessione e di autenticazione





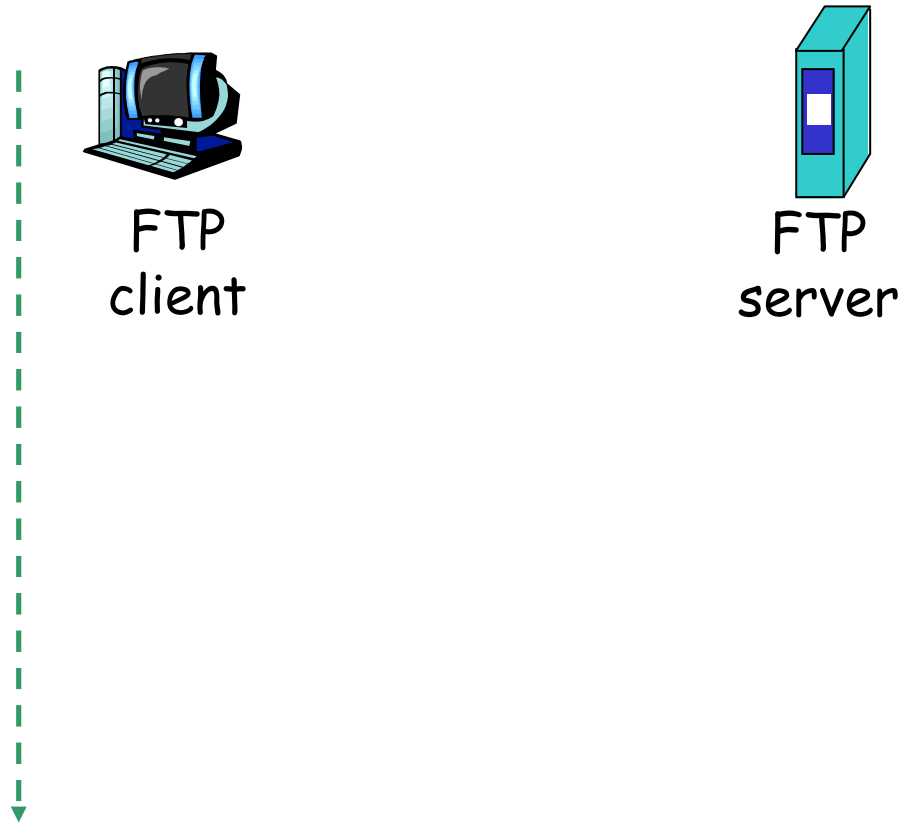
# Realizzazione

- Quando il client chiede di trasferire un file
  - il server riceve la richiesta sul canale di controllo (TCP 21)
  - il server verifica la richiesta
    - il file richiesto esiste?
    - l'utente ha il permesso di leggere/scrivere il file?
  - quindi il server apre un canale dati con il client
    - dovendo essere una connessione affidabile, si userà il protocollo TCP
    - non sapendo il server come indirizzare il client (conosce l'indirizzo dell'host, ma non ha modo di decidere quale sia l'applicazione sull'host), deve usare una porta standard
    - il client deve aspettarsi di ricevere una connessione sulla porta TCP 20
  - quindi il server (comando RETR), o il client (comando STOR), trasferisce il file lungo il canale dati
  - infine il server chiude la connessione dati



# Realizzazione

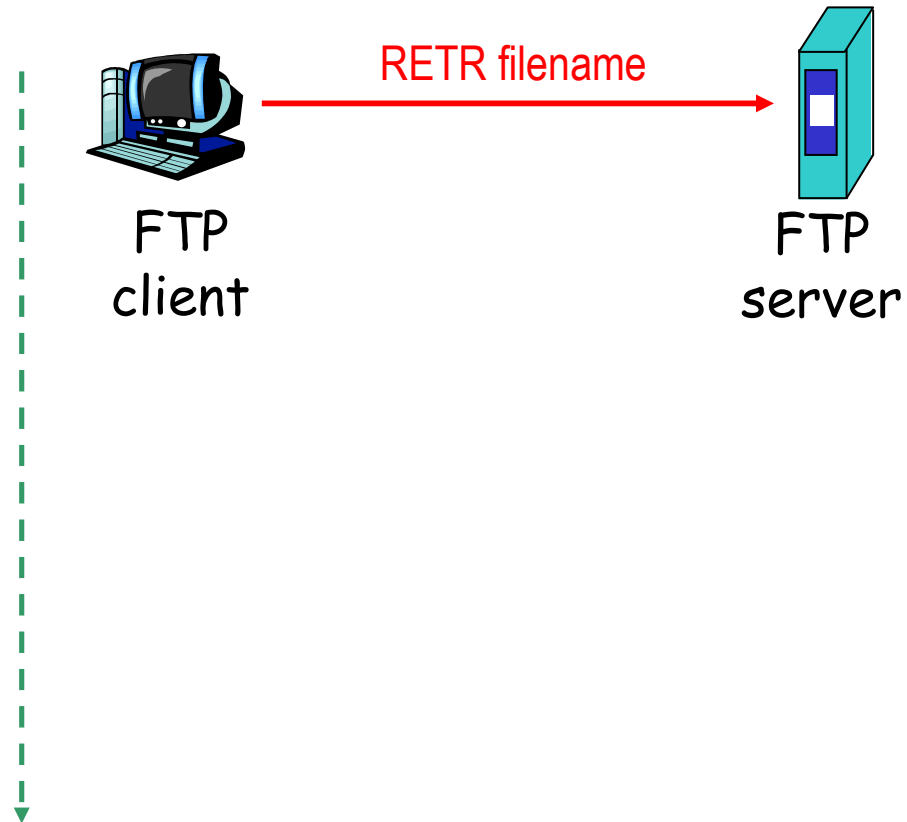
- Esempio: trasferimento di un file dal server al client





# Realizzazione

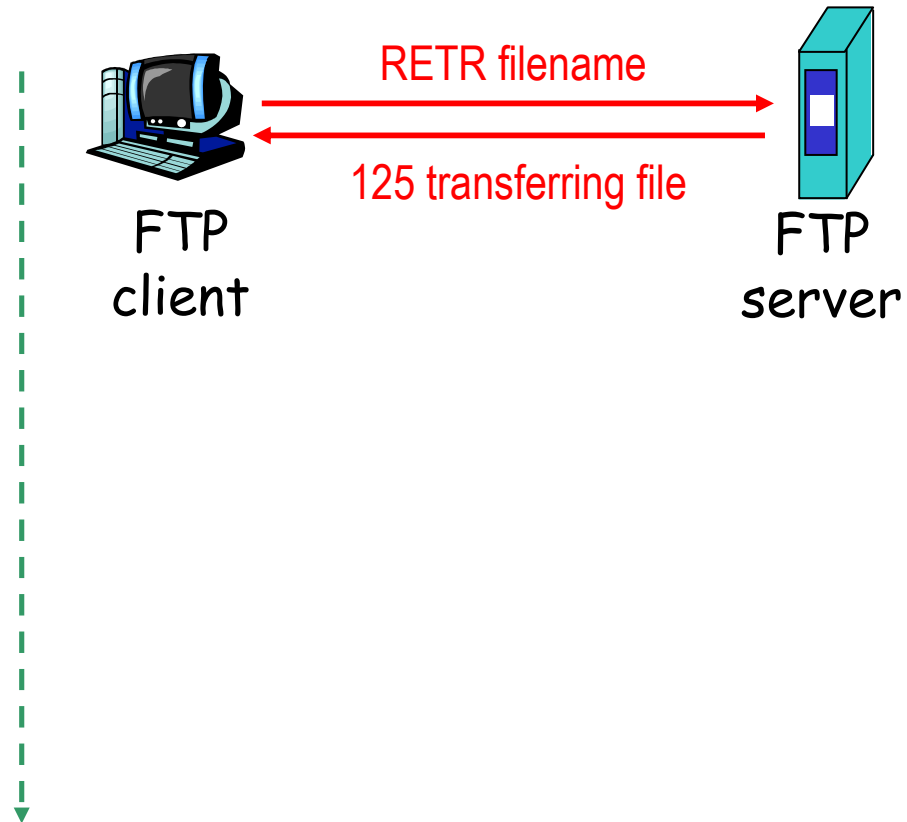
- Esempio: trasferimento di un file dal server al client





# Realizzazione

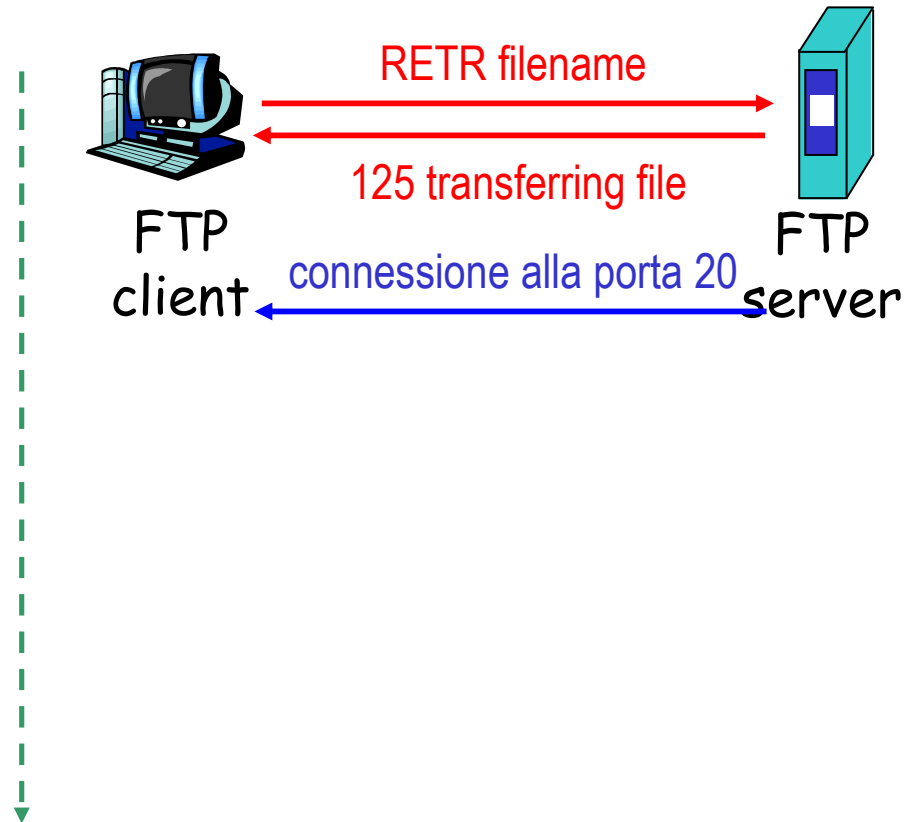
- Esempio: trasferimento di un file dal server al client





# Realizzazione

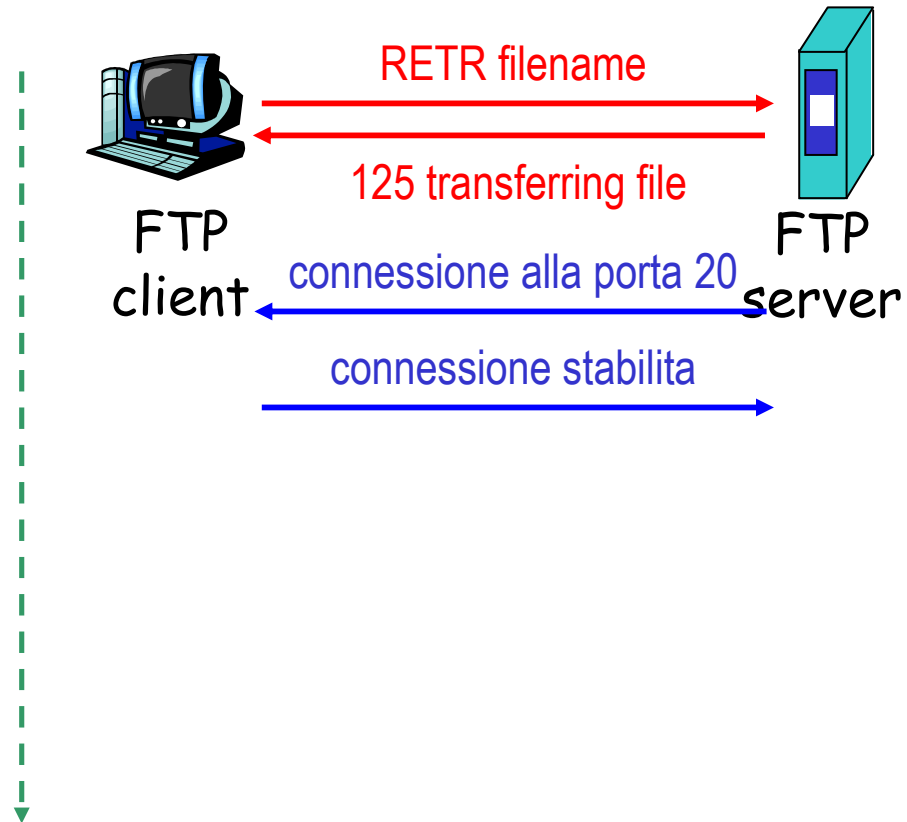
- Esempio: trasferimento di un file dal server al client





# Realizzazione

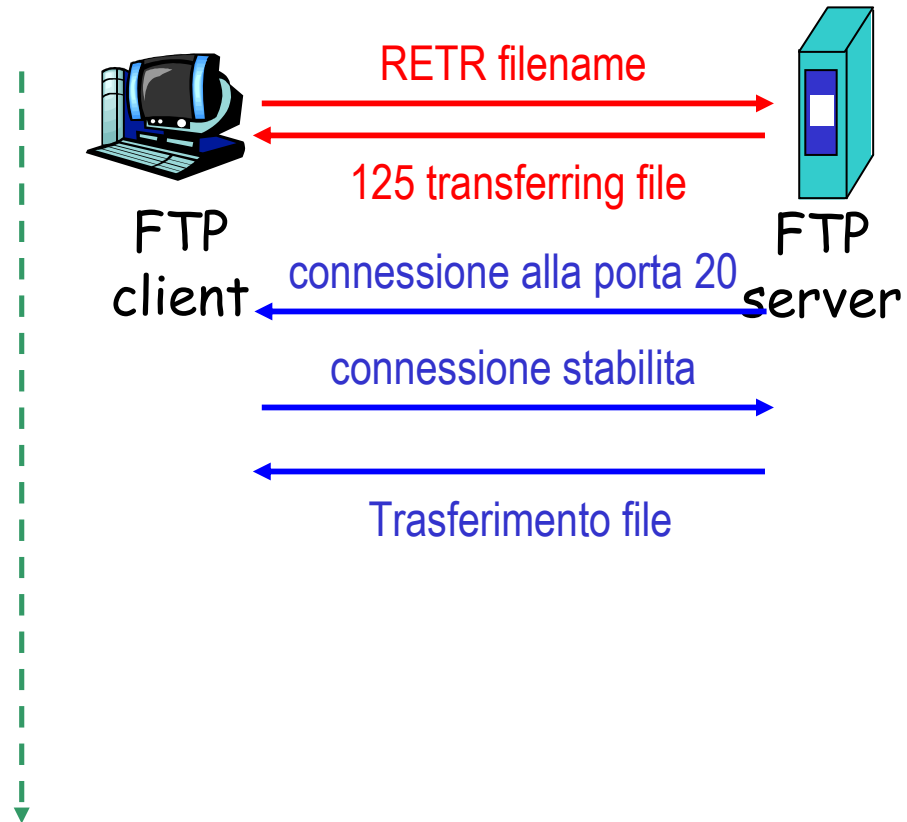
- Esempio: trasferimento di un file dal server al client





# Realizzazione

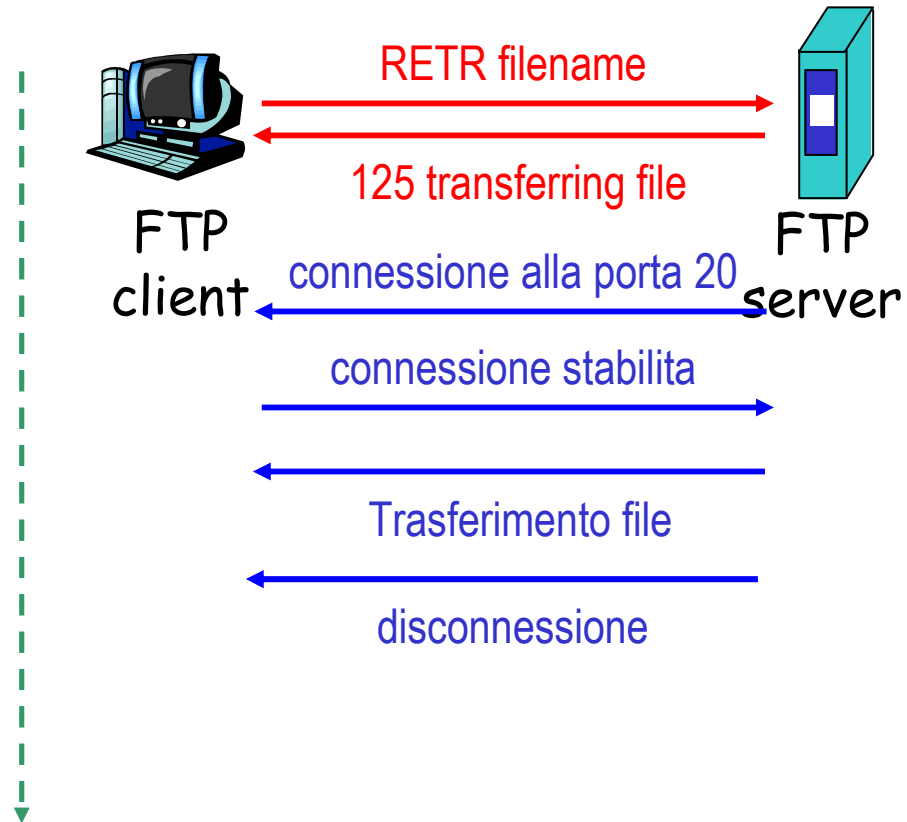
- Esempio: trasferimento di un file dal server al client





# Realizzazione

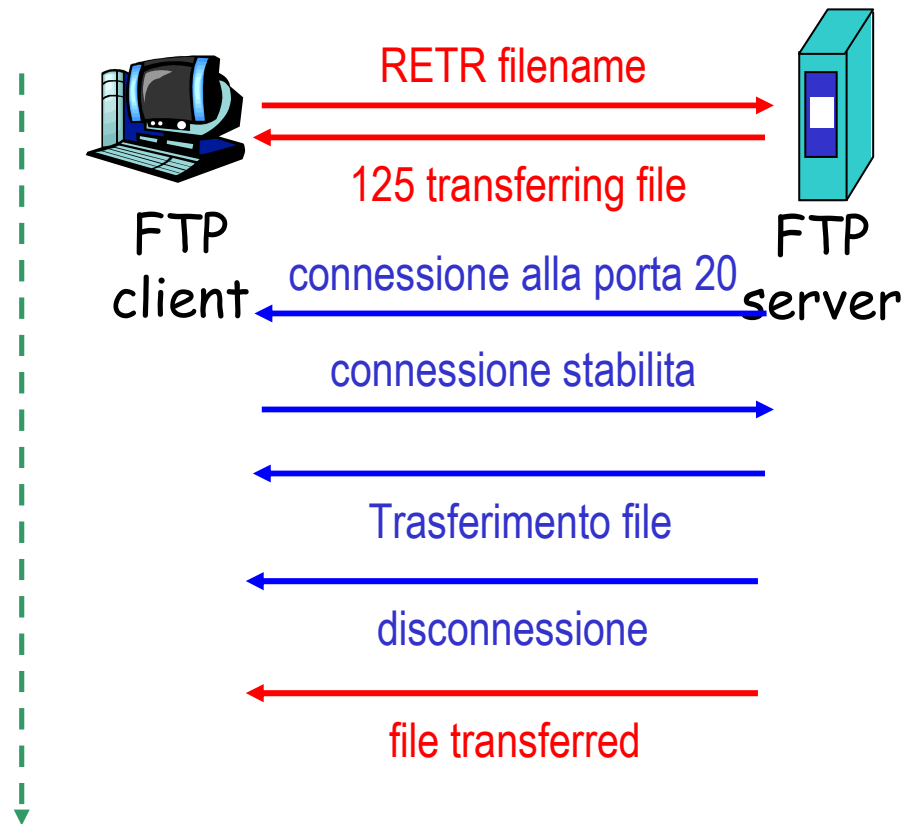
- Esempio: trasferimento di un file dal server al client





# Realizzazione

- Esempio: trasferimento di un file dal server al client





# Trasmissione fuori banda

- Abbiamo visto che il protocollo FTP trasferisce i dati (file) su una connessione TCP differente da quella usata per trasmettere i comandi
  - questo modo di operare è detto trasmettere i comandi **fuori dalla banda**
  - questa definizione è giustificata poiché
    - è presumibile che il maggior consumo di banda sia dovuto al trasferimento dei file
    - effettivamente i comandi non consumano la banda destinata al trasferimento dei dati
    - i comandi hanno un consumo di banda trascurabile



# Connessione persistente

- La connessione di controllo (porta 21) è **persistente**
  - essa dura tutto il tempo della sessione di lavoro tra client e server
- La connessione dati (porta 20) è **transiente** (non persistente)
  - essa dura solo il tempo necessario a trasmettere un file
- In generale, un protocollo che utilizza una **sessione** di lavoro per far comunicare un client ed un server può decidere di realizzare la sessione per mezzo di una connessione persistente



# Protocollo con stato

- Il protocollo FTP deve mantenere lo stato della connessione
  - lo stato descrive chi è connesso
  - quale è la directory corrente di ogni utente connesso
  - altre informazioni (ad esempio il modo e la codifica di trasferimento)
- Lo stato determina il modo in cui il server risponde ai comandi del client
  - se nella directory corrente non vi è il file richiesto, ottengo un errore
  - se tale file è presente, il server lo invia
  - quindi l'ordine dei comandi nella sequenza inviata al server è importante e modifica il comportamento del server
    - se il file X non è presente nella directory corrente, ma nella sottodirectory `subdir`, allora la sequenza `CD subdir, RETR X` ha un esito ben diverso da `RETR X, CD subdir`



# Conclusione

- Questa lezione è tratta da:
  - Kurose, cap. 2.3
  - RFC 959
- Esercizi:
  - Provate a descrivere il protocollo FTP con UML
  - Per chi sa cos'è un firewall: perché esiste il modo FTP passivo? Come funziona?
  - Il protocollo HTTP usa connessioni non persistenti ed è privo di stato: quali sono le conseguenze? Esistono anche le connessioni persistenti in HTTP: perché sono state introdotte?