Corso di Web Programming

2. Elementi di Reti e Internet

Paolo Milazzo

Dipartimento di Informatica, Università di Pisa
http://www.di.unipi.it/∼milazzo
milazzo@di.unipi.it

Corso di Laurea in Informatica Applicata
A.A. 2010/2011

Sommario

Che cos'è Internet?

Una possibile definizione di Internet è la seguente:

> *Internet è l'insieme di tutti i computer collegati in rete tra loro (usando varie tecnologie di collegamento) e che impiegano i protocolli di rete di Internet al di sopra dei propri protocolli di rete. I protocolli di rete di Internet realizzano una rete a pacchetto che è capace di interconnettere reti basate su protocolli e tecnologie di collegamento differenti.*

Quindi:

- Internet è una rete di reti di computer
- Internet è caratterizzata dall'uso di propri protocolli di rete (TCP/IP)
- A basso livello le reti connesse possono usare tecnologie di collegamento differenti (Ethernet, Wireless, ATM, PPP, ecc...)

Dove è descritta Internet?

- Descrizioni ufficiali dei protocolli e delle tecnologie usate in Internet sono raccolte dalla Internet Engineering Task Force (IETF) e pubblicate sotto forma di Request for Comments (RFC).
- Gli RFC sono liberamente disponibili all'indirizzo `http://www.ietf.org`
- Alcuni RFC che contengono informazioni generali su Internet sono i seguenti:
  - RFC 2235 - Hobbes' Internet Timeline – Storia di Internet dal '57 al '97
  - RFC 2264 - Answers to Commonly Asked "New Internet User" Questions
- Mentre tutte i protocolli e le applicazioni che i computer collegati a internet devono utilizzare sono descritti (in dettaglio) nei seguenti RFC:
  - RFC 1122 - Requirements for Internet Hosts – Communication Layers
  - RFC 1123 - Requirements for Internet Hosts – Application and Support

Lo stack dei protocolli di Internet (1)

- Uno dei principi fondamentali delle reti di computer è la strutturazione gerarchica (a livelli) delle architetture di comunicazione.
  - A livelli più bassi corrispondono aspetti di base dell'architettura di comunicazione (cavi, connettori, livelli di voltaggio utilizzati, ecc...)
  - A livelli più alti corrispondono aspetti via via più astratti e vicini alle applicazioni (destinatario di un messaggio, creazione di sessioni di comunicazione, ecc...)
- La strutturazione a livelli
  - rende più semplice la progettazione delle tecnologie e dei protocolli di comunicazione (un protocollo di un certo livello può assumere che certe funzionalità di base siano già fornite dai protocolli dei livelli sottostanti)
  - favorisce l'intercambiabilità dei livelli stessi e la portabilità delle applicazioni

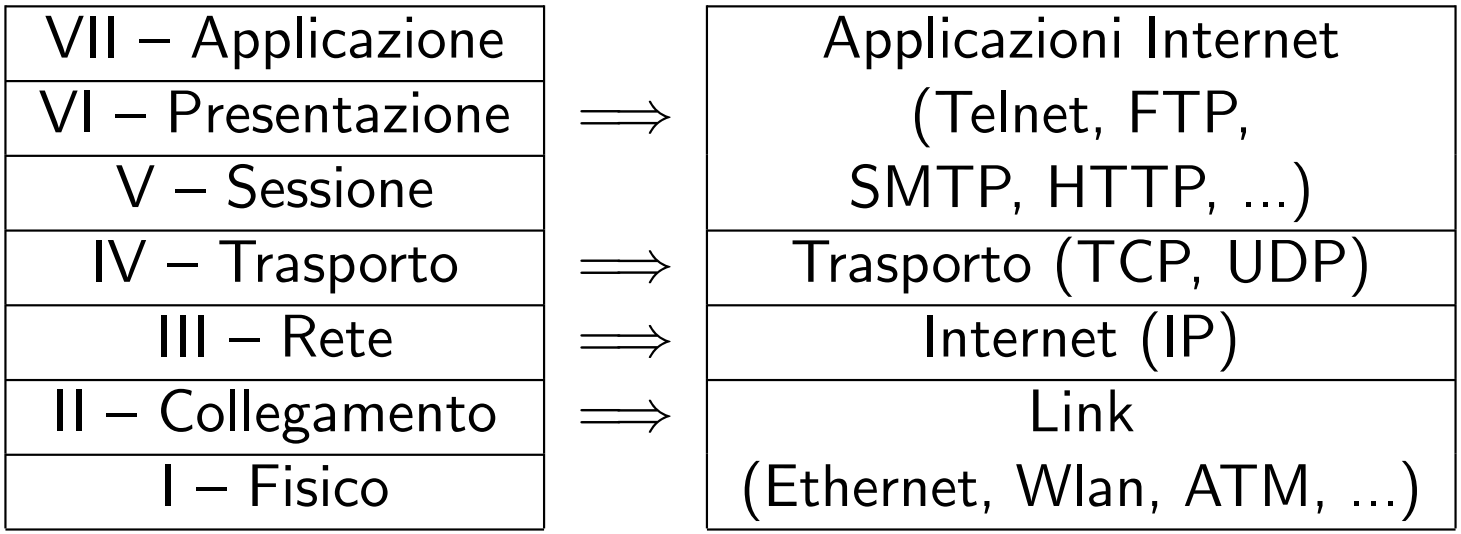
Lo stack dei protocolli di Internet (2)

Un noto modello di strutturazione a livelli è il modello OSI:

VII – Applicazione	Applicazioni che usano la rete
VI – Presentazione	Conversioni di dati, crittografia, ecc...
V – Sessione	Creazione di sessioni tra host
IV – Trasporto	Connessioni e reliability
III – Rete	Determinazione dei cammini
II – Collegamento	Indirizzamento fisico (MAC)
I – Fisico	Media, segnale e trasmissione binaria

Lo stack dei protocolli di Internet (3)

L'architettura di Internet si basa sulla struttura a livelli detta stack TCP/IP:



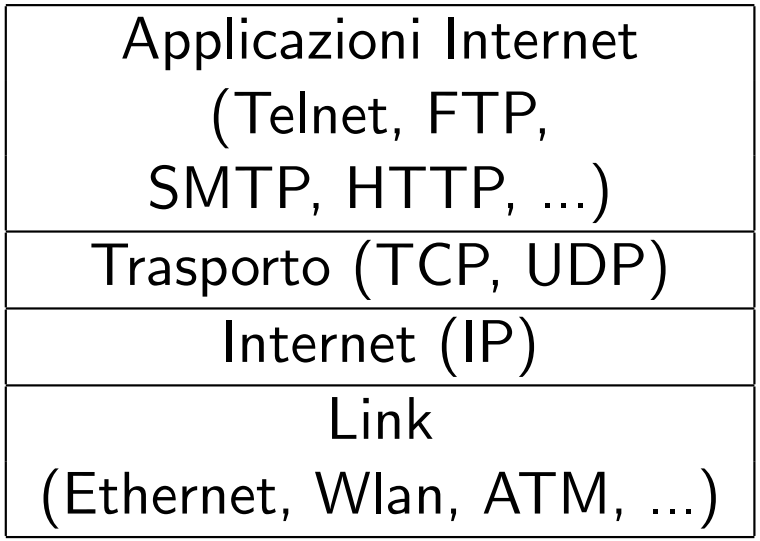
Lo stack dei protocolli di Internet (4)

Uno degli aspetti chiave che ha portato al successo dello stack dei protocolli di Internet è la sua robustezza.

Differentemente da molte altre tecnologie di rete, che assumono di lavorare in condizioni perfette o vicine alla perfezione (no crash dei nodi della rete, pacchetti sempre consegnati entro un tempo stabilito, ecc...), Internet e progettato per tollerare una quantità anche molto alta di errori.

Nella specifica di un livello dei protocolli si assume sempre che ai livelli inferiori qualcosa possa non aver fuzionato. Questo obbliga i software che realizzano tali protocolli a prevedere sempre di dover gestire le situazioni di errore.

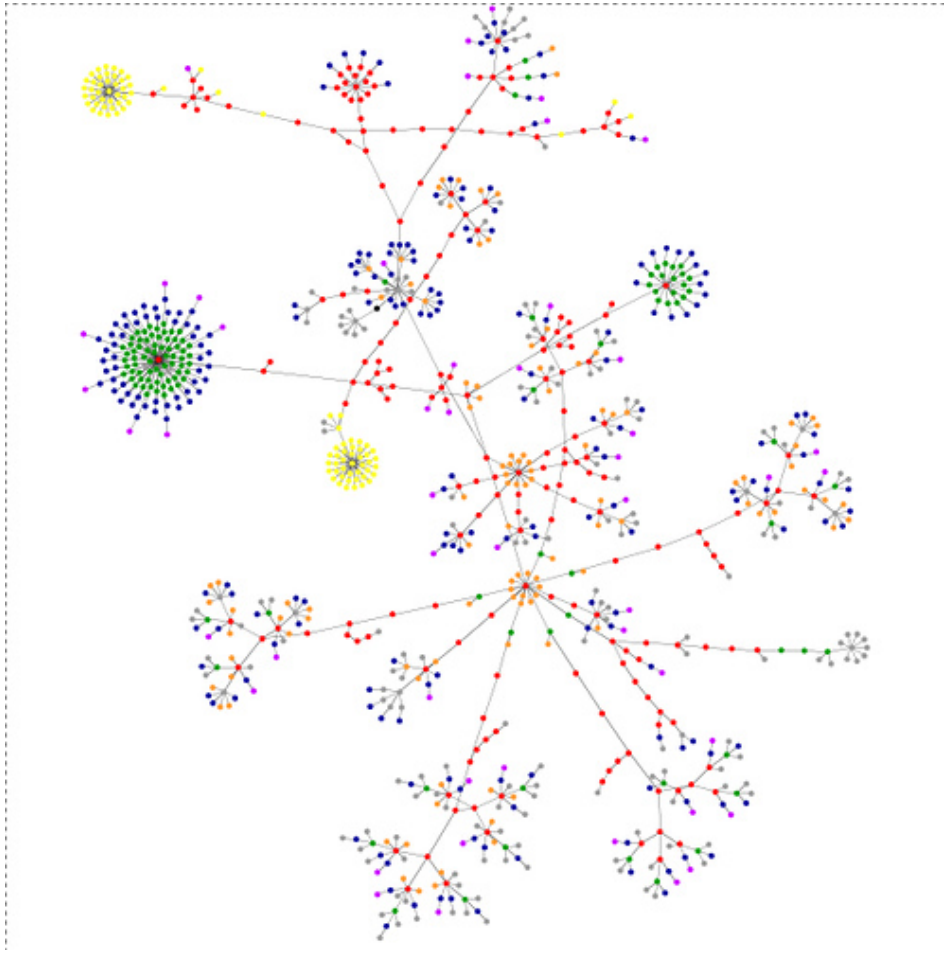
Il livello Link (1)



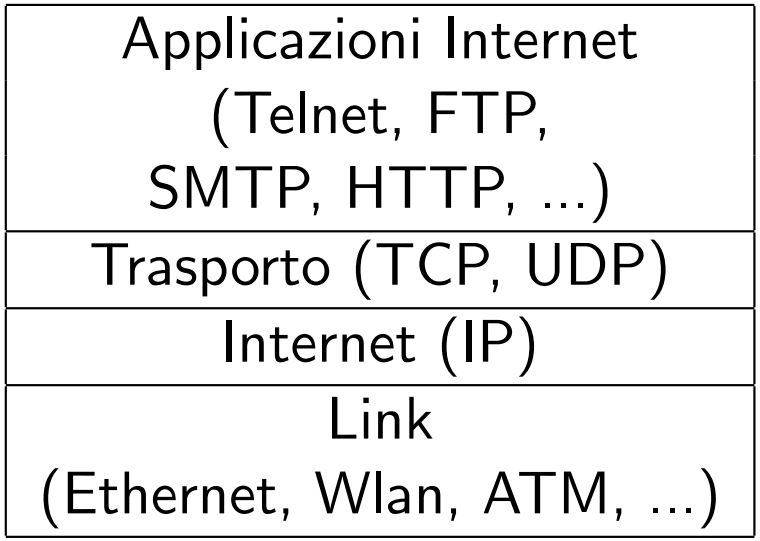
- Intenet può basarsi su qualunque tecnologia di rete.
- Esempio comune: Ethernet (reti locali, collegamento con modem adsl, ecc...)
- Inotre consente la comunicazione tra nodi di reti diverse collegate attraverso gateways (nodi condivisi da più reti anche eterogenee)

Il livello Link (2)

Un esempio grafico di Internet costituita da reti eterogenee colleagata tramite gateways:



Il protocollo IP (1)



- Il protocollo IP (Internet Protocol) è uno dei protocolli più importanti dello stack di Internet (assieme forse a TCP).
- E' un protocollo per la distribuzione di pacchetti su una rete definito nell RFC 791.

Il protocollo IP (2)

Il protocollo IP ha le seguenti caratteristiche:

- connection-less:
  - Ogni pacchetto viene spedito individualmente (non si creano connessioni permanenti tra nodi della rete per la spedizione di sequenze di pacchetti).
  - Due pacchetti successivi per la stessa destinazione possono seguire rotte diverse (favorisce robustezza).
- best-effort: non offre garanzie sull'arrivo dei pacchetti e sulla loro integrità (ci dovranno pensare i protocolli di livello superiore)

Nella sua versione più comune (IPv4) usa indirizzi (in 32 bit) della forma:

$$x.y.z.w \text{ dove } x, y, z, w \in [0, 255]$$

ad esempio 127.0.0.1, 192.168.10.20 e 131.114.8.153 sono indirizzi IP validi.

Il protocollo IP (3)

Il formato dei pacchetti IP include i seguenti tra i suoi campi più importanti:

Version La versione del protocollo (IPv4 o IPv6);

Total Length La dimensione totale del pacchetto in bytes;

Time-to-live (TTL) Tempo di vita residua di un pacchetto, decrementato ogni volta che attraversa un nodo e utilizzato per rimuovere pacchetti "persi" quando TTL=0;

Protocol Indica quale protocollo di livello superiore è stato utilizzato per creare le informazioni immagazzinate nel pacchetto (serve al destinatario per interpretare correttamente i dati ricevuti);

Source IP Indirizzo IP del mittente;

Destination IP Indirizzo IP del destinatario;

Data I dati trasmessi.

I campi dei un pacchetto escluso il campo data sono detti "header" (testata).

Il protocollo IP (4)

Il protocollo IP prevede:

- Diverse tecniche di routing (instradamento) dei pacchetti:
  - Algoritmi utilizzati dai nodi della rete quando ricevono un pacchetto non destinato a loro (spesso basati su tabelle di routing)
- Una tecnica di frammentazione di pacchetti da usare:
  - per suddividere un dato di grandi dimensioni in più pacchetti e
  - quando un pacchetto passa (attraverso un gateway) da una rete che consente di trasmettere pacchetti di una certa dimensione massima a una rete la cui dimensione massima dei pacchetti è inferiore.
- Una tecnica di ri-assemblamento di pacchetti frammentati (per ricostruire il dato precedentemente frammentato).

Il protocollo IP (5)

Attualmente la versione più usata del protocollo IP è la 4 (IPv4). Tale versione soffre di alcune limitazioni:

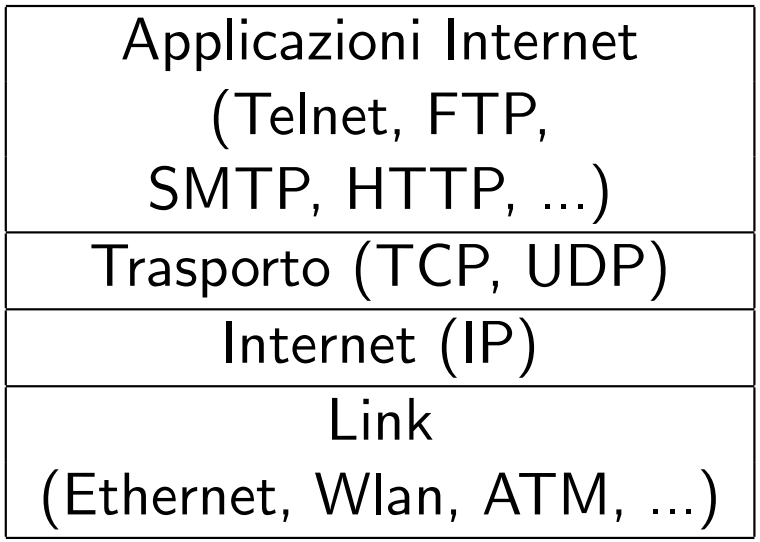
Scarsità di indirizzi La struttura degli indirizzi IP consente di creare circa 4 miliardi di indirizzi diversi. Il modo grossolano in cui gli indirizzi vengono assegnati causa un forte spreco di indirizzi che stanno quindi per terminare;

No prenotazione delle risorse Alcuni contenuti trasmessi su Internet (e.g. audio/video) soffrono dell'assenza di garanzie in IPv4 sul tempo di consegna dei pacchetti. Per queste applicazioni sarebbe meglio prevedere la possibilità di riservare risorse (banda di trasmissione lungo le rotte di interesse) al fine di garantire una migliore qualità del servizio;

Scarso supporto per dispositivi mobili Sempre più frequentemente i dispositivi collegati a Internet sono mobili (laptop, telefoni, ecc...). IPv4 non prevede nessun supporto per la mobilità.

Alcune di queste limitazioni sono superate dalla nuova versione IPv6 (RFC 1883) che diventerà nei prossimi anni il nuovo standard.

I protocolli TCP e UDP (1)



- Nonostante il protocollo IP consenta di far comunicare nodi di reti diverse ed eterogenee, non consente di indicare il processo in esecuzione in un nodo a cui i dati devono essere consegnati.
- Inoltre IP non offre garanzie: i pacchetti possono andare persi, possono arrivare in ordine sparso o essere danneggiati.
- Il protocollo TCP (Transmission Control Protocol – RFC 793) per fornire una tecnologia di comunicazione affidabile tra processi.

I protocolli TCP e UDP (2)

- Il protocollo TCP è progettato per porsi al di sopra di un protocollo di comunicazione simile a IP:
  - Assume che il livello sottostante non offra garanzie sulla distribuzione dei pacchetti;
- TCP fornisce la possibilità di creare connessioni tra processi eseguiti sui nodi della rete tramite l'uso di numeri di porta.
  - L'indirizzo 131.114.8.149:1044 rappresenta la porta 1044 usata in una connessione TCP da un processo eseguito all'indirizzo IP 131.114.8.149
- TCP offre garanzie sulla comunicazione tramite:
  - controllo degli errori (tramite checksum)
  - riordino dei pacchetti
  - ritrasmissione dei pacchetti persi (non arrivati entro un timeout)

I protocolli TCP e UDP (3)

Il formato dei pacchetti TCP include i seguenti tra i suoi campi più importanti:

Source port Identifica il numero di porta del mittente;

Destination port Identifica il numero di porta del destinatario;

Sequence number Numero di sequenza usato per riordinare i pacchetti;

Ack number Usato in fase di risposta per indicare il sequence number del prossimo pacchetto che ci si aspetta di ricevere (può causare ritrasmissione di pacchetti persi);

Checksum Campo di controllo usato per la verifica della validità del pacchetto;

Flags SYN/FIN Usati per aprire e chiudere connessioni;

Data I dati effettivamente trasmessi.

I protocolli TCP e UDP (4)

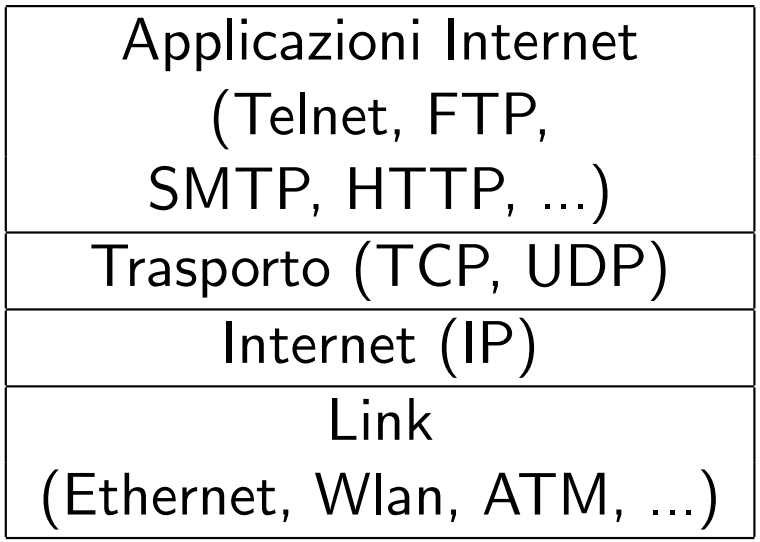
Nella pratica:

- Un pacchetto TCP viene inserito nel campo data di un pacchetto IP.
- Quando il pacchetto IP raggiunge il nodo di destinazione, l'header IP viene rimosso e il contenuto (il campo data, quindi il pacchetto TCP) viene processato.

Alcune applicazioni non hanno necessità di tutti i servizi offerti dal protocollo TCP e/o richiedono maggiori performance (e.g. audio/video).

- Alternativa: UDP (User Datagram Protocol – RFC 768) è una versione semplificata (e più efficiente) di TCP che offre solo indirizzamento di processi (numeri di porta) e checksum.

I protocolli a livello applicazione



I protocolli di livello applicazione si dividono in due categorie:

- Protocolli di applicazione vera e propria (e.g. SMTP, HTTP, telnet, FTP, etc...) che forniscono il servizio agli utenti finali
- Protocolli di servizio (e.g. DNS) forniscono servizi alle applicazioni usate dagli utenti

Questi protocolli utilizzano TCP/IP come strumento di comunicazione affidabile tra nodi della rete.

I protocolli a livello applicazione

Vediamo alcuni esempi di protocolli a livello applicazione:

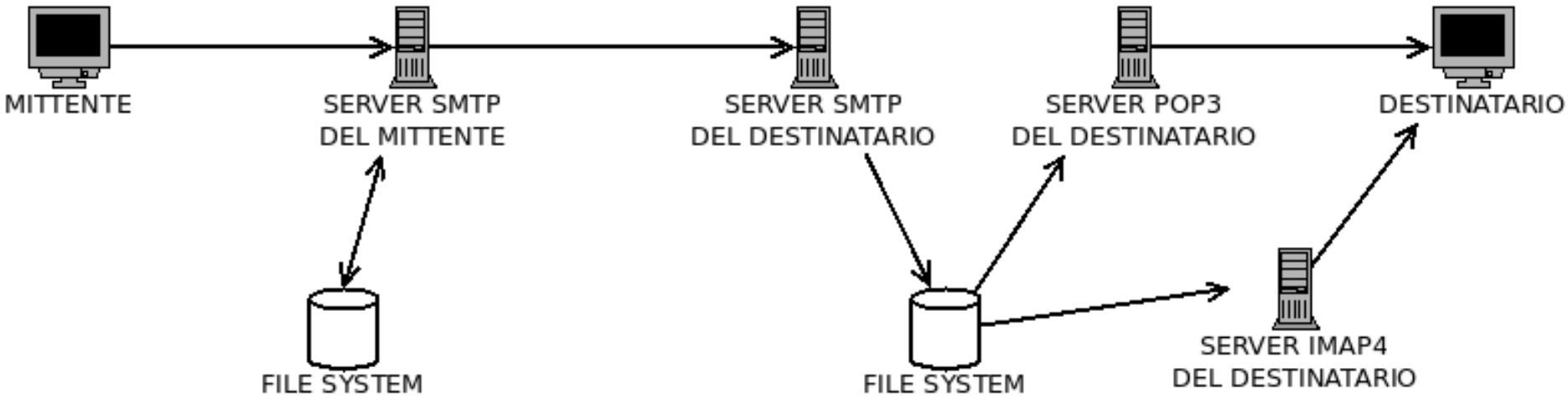
- I protocolli per la posta elettronica (SMTP, POP3 e IMAP)
- Il protocollo per il trasferimento di file FTP

Protocolli per la posta elettronica

La posta elettronica è basata sull'applicazione di 3 protocolli di VII livello:

- Simple Mail Transport Protocol (SMTP, RFC 821 e 822 aggiornati in 2821 e 2822): client-to-host e host-to-host;
- Post Office Protocol ver. 3 (POP3, RFC 1939): host-to-client;
- Internet Message Access Protocol ver. 4rev1 (IMAP4, RFC 2060): host-to-client.

Il funzionamento della posta elettronica è schematizzato nel seguente diagramma:



Il protocollo SMTP (1)

- SMTP è un protocollo text-based per lo scambio di messaggi di posta elettronica.
- Una connessione SMTP è composta da una apertura, una sequenza di comandi, e una chiusura.
- Ad ogni comando corrisponde una risposta composta da un codice numerico (per le applicazioni) e una stringa leggibile (per gli umani).
  - MAIL FROM:⟨Smith@alpha.com⟩
  - 250 OK
  - RCPT TO:⟨Green@alpha.com⟩
  - 550 No such user here

- La spedizione di un messaggio SMTP avviene attraverso l'identificazione del mittente (MAIL FROM), del/dei destinatari (RCPT TO), e del messaggio da trasmettere (DATA):
  - 220 alpha.com Simple Mail Transfer Service Ready
  - HELO beta.com
  - 250 alpha.com says: Nice to meet you beta.com
  - MAIL FROM:⟨Smith@alpha.com⟩
  - 250 OK
  - RCPT TO:⟨Green@alpha.com⟩
  - 550 No such user here
  - RCPT TO:⟨Brown@alpha.com⟩
  - 250 OK
  - DATA
  - 354 Start mail input; end with ⟨CRLF⟩.⟨CRLF⟩
  - Blah blah blah....
  - etc. etc. etc.
  - .
  - 250 OK
  - QUIT
  - 250 alpha.com Service closing transmission channel

Il protocollo SMTP (3)

Il protocollo SMTP ha alcune forti limitazioni:

- La lunghezza massima di un messaggio è di 1MB;
- I caratteri accettati sono solo ASCII a 7 bit;
- Ogni messaggio deve avere un CRFL ogni 1000 caratteri o meno (alcune antiche implementazioni lo aggiungevano automaticamente se non lo trovavano).

Questi limiti impediscono la trasmissione di documenti binari:

- Un file binario usa tutti i 256 tipi di byte;
- Un file binario può facilmente essere più lungo di 1MB;
- In un file binario la sequenza CRFL è una sequenza come tutte le altre, e può esserci o mancare senza vincoli. Introdurla artificialmente può corrompere il file.

Estensioni del protocollo SMTP: MIME (1)

I limiti di SMTP sono comunemente superati attraverso l'uso delle estensioni MIME (Multipurpose Internet Mail Extensions, RFC 2045-2049)

MIME ridefinisce il formato del corpo di un messaggio SMTP per permettere (tra le altre cose):

- Messaggi di testo in altri set di caratteri al posto di ASCII;
- Un insieme estensibile di formati (tipi) per messaggi non testuali;

Il problema di determinare il "tipo" di un contenuto non testuale è comune a molti protocolli a livello applicazione.

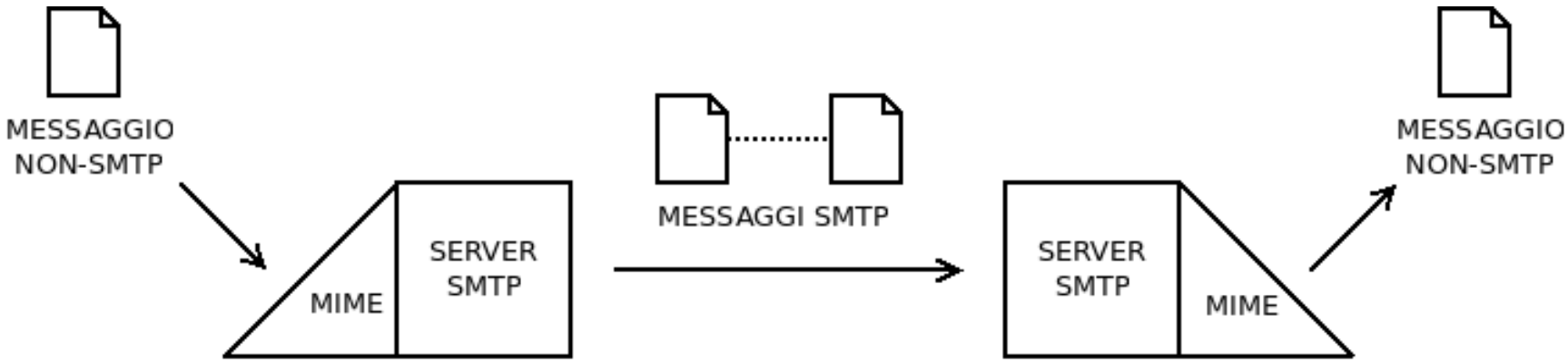
- MIME viene utilizzato anche in altri contesti (e.g. WWW)

Estensioni del protocollo SMTP: MIME (2)

Come funziona MIME:

1. Il messaggio non compatibile con SMTP viene trasformato in uno o più messaggi SMTP da un preprocessore al server SMTP;
2. All'arrivo, il (o i) messaggi SMTP vengono decodificati e riaccorpati a formare il messaggio originale.

Ancora oggi, quelli che viaggiano su canali SMTP sono puri messaggi SMTP con gli stessi limiti di allora.



Estensioni del protocollo SMTP: MIME (3)

- La specifica di MIME include aspetti riguardanti la codifica dei caratteri non in formato ASCII, la frammentazione di messaggi non SMTP in SMTP e il successivo riaccorpamento, ecc...
- L'aspetto di MIME che trova applicazione in altri contesti (e.g. WWW) è il concetto di `Content-type` (o tipo MIME)

Un tipo MIME definisce uno specifico tipo per un contenuto associandogli un identificatore di "media type" e un identificatore di "subtype" (rappresentati così: `mediatype/subtype`.

- Il media type dichara il tipo generale del dato trasmesso
- Il subtype dichiara il formato specifico del dato trasmesso

Ad esempio:

- il tipo MIME `text/plain` identifica testo semplice;
- il tipo MIME `text/html` identifica testo html;
- il tipo MIME `image/gif` identifica immagini in formato GIF;
- il tipo MIME `video/mpeg` identifica video in formato MPEG.

Estensioni del protocollo SMTP: MIME (4)

La definizione compelta di `Content-type` MIME può contenere anche parametri, ad esempio:

```
Content-type:  text/plain; charset=ISO-8859-1
```

I client di posta elettronica (e similmente le altre applicazioni – e.g. i browser – che usano tipi MIME) possono essere configurati per inoltrare i contenuti di un certo tipo (o sottotipo) ad altre applicazioni

- Ad esempio, si può decidere di inoltrare tutti i contenuti di tipo `image` ad un visualizzatore di immagini, indipendentemente dal sottotipo

E' possibile strutturare i messaggi di posta elettronica in parti (messaggi multipart) aventi tipi MIME diversi.

I tipi MIME sono in continua evoluzione:

- La lista dei tipi MIME è mantenuta dall'Internet Assigned Numbers Authority (IANA)

Il protocollo POP3

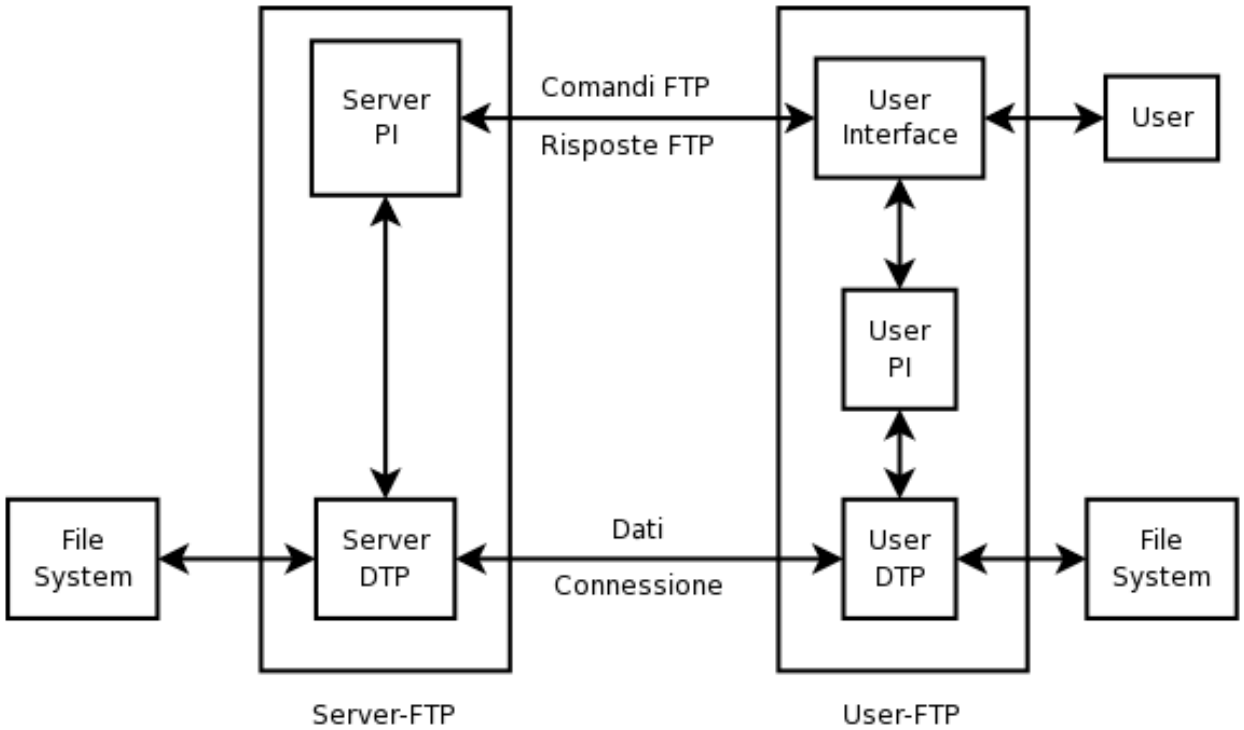
- SMTP si disinteressa di come il ricevente di un messaggio acceda alla sua mailbox. Si supponeva all'epoca che tutti avessero accesso via file system alla directory con le mailbox;
- POP3 permette ad una applicazione utente di accedere alla mailbox posta su un altro sistema;
- POP3 non permette manipolazioni complesse sulla mailbox, ma soltanto la possibilità di scaricare e cancellare mail. Per operazioni più complesse si usa IMAP4.

Il protocollo IMAP4

- Il protocollo IMAP4 consente un controllo più sofisticato della propria mailbox, anche se posta su un server remoto;
- Le caratteristiche di IMAP4 più rilevanti per l'utilizzo quotidiano sono:
  - La possibilità di consultare la posta elettronica senza scaricarla dal server (favorisce l'accesso da diverse postazioni);
  - La possibilità di creare directory di messaggi sul server (da usare anche per filtrare i messaggi sul server – e.g. procmail)
- IMAP consente operazioni di creazione, cancellazione e cambio di nome a directory all'interno della mailbox; verifica di nuovi messaggi; cancellazione di messaggi; ricerca per contenuto e attributi; scaricamento selettivo di attributi, parti e messaggi.

Protocolli per il trasferimento di file (1)

- Il File Transfer Protocol (FTP, RFC 959) è un protocollo per la trasmissione di file;
- Utilizza due connessioni TCP separate: una per i comandi (detto "canale comandi") e una per i file trasmessi (detto "canale dati").



dove PI (protocol interpreter) è l'interprete dei comandi e DTP (data transfer process) è il processo di trasferimento dati.

Protocolli per il trasferimento di file (2)

- Il canale comandi rimane aperto per l'intera sessione utente
- Un nuovo canale dati (connessione TCP) viene utilizzato per ogni file trasmesso (favorisce la trasmissione contemporanea di più file)

Tra le funzioni offerte da un server FTP abbiamo:

- Download/upload di file;
- Resume di trasferimenti interroti;
- Rimozione e rinomina di un file;
- Creazione di directory;
- Navigazione tra directory.

Protocolli per il trasferimento di file (3)

- FTP offre un meccanismo di autenticazione in chiaro degli accessi;
- E' possibile accedere ai contenuti di un server FTP in modalità "anonima" (senza password). In questo caso tipicamente il client può accedere ai dati in modalità di "sola lettura";
- Una versione "sicura" di FTP basato su crittografia SSL/TLS è FTPS (da non confondere con SFTP che è una variante di SSH per il trasferimento di file). Questa è ormai la versione di FTP più utilizzata.

Esempi di client FTP (per Windows) sono:

- FileZilla (include anche un server)
- WinSCP (protocolli FTP, SFTP e SCP)

FTP(S) è usato (spesso in maniera trasparente) anche all'interno di browser web e download managers (e.g. JDownloader)

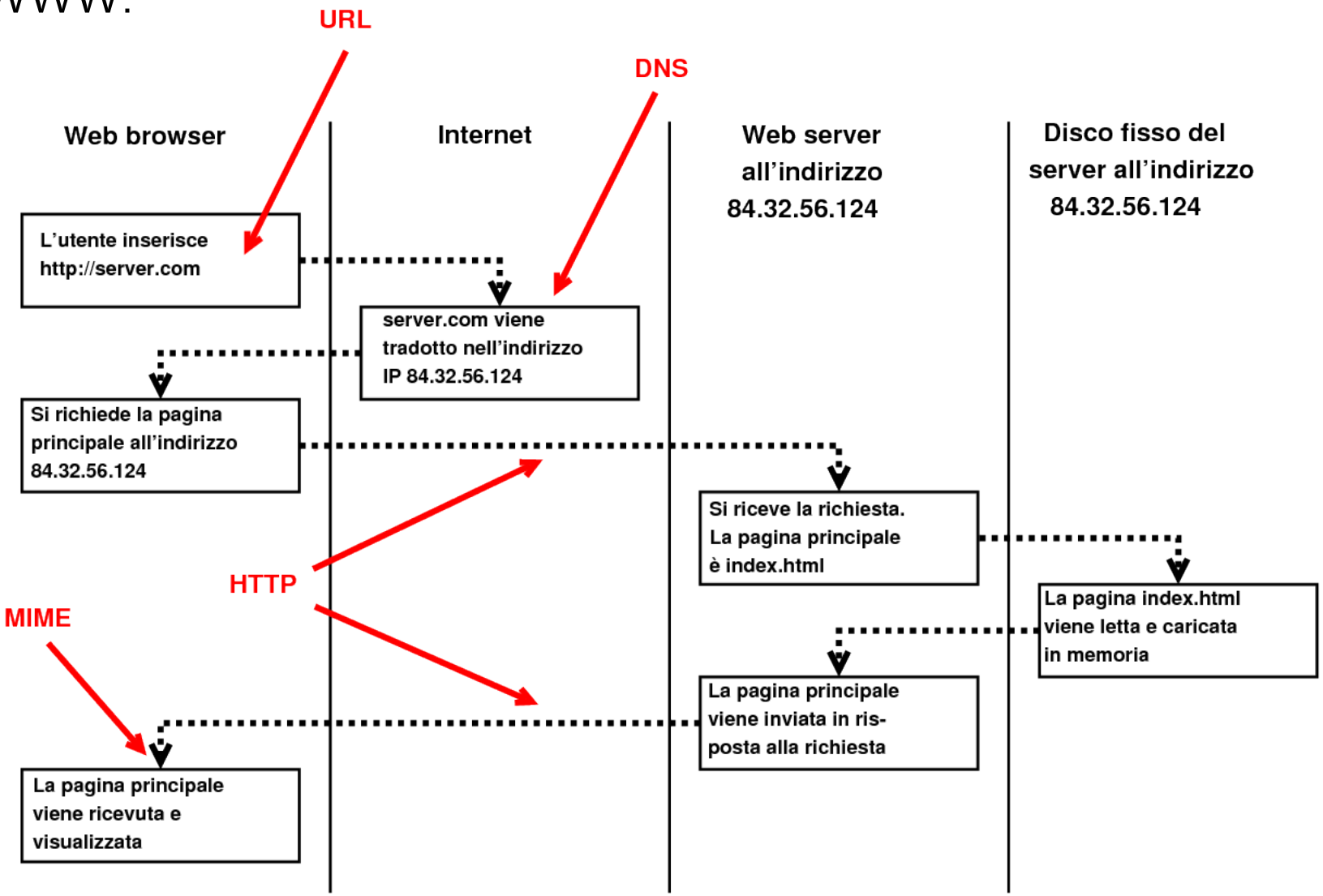
Internet e il WWW (1)

Il WWW usa solo alcuni dei servizi che l'infrastruttura di Internet mette a disposizione. I principali sono:

- L'indirizzamento delle risorse: tramite URL (e.g. `http://www.di.unipi.it`)
- La risoluzione degli indirizzi: traduzione di indirizzi testuali tipo `www.di.unipi.it` nei corrispondenti indirizzi IP tramite DNS;
- La gestione dei tipi di contenuti: tramite MIME;
- Il trasferimento dei contenuti: tramite il protocollo di comunicazione HTTP.

Internet e il WWW (2)

Vediamo dove DNS, MIME e HTTP vengono utilizzati in un tipico utilizzo del WWW:



Indirizzare le risorse: URL (1)

- L'indirizzo di una risorsa su Internet può essere rappresentata da un Uniform Resource Locator (URL, RFC 1738 e 1808);
- URL è una sintassi standard per la definizione di indirizzi sulla rete;
- Una definizione più generale di indirizzi è data da un Uniform Resource Identifier (URI) che, per semplicità, non trattiamo.

Un URL consiste di due parti:

- uno schema (in inglese scheme)
- e uno parte schema-specifica (in inglese scheme-specific part)

Indirizzare le risorse: URL (2)

L'insieme degli schemi validi è mantenuto dall'Internet Assigned Numbers Authority (IANA), e include i seguenti:

- `ftp` per gli indirizzi di risorse accessibili tramite FTP
- `http` per gli indirizzi di risorse accessibili tramite HTTP
- `https` per gli indirizzi di risorse accessibili tramite HTTP sicuro
- `mailto` per gli indirizzi di posta elettronica

Le parti schema-specifiche possono avere la seguente sintassi (in cui le parentesi quadre rappresentano parti opzionali):

```
// [ user [ :  password ] @ ] host [ :  port ] / url-path
```

Esempi:

- `http://www.di.unipi.it/~milazzo/`
- `mailto:milazzo@di.unipi.it` (in realtà non si usa `//` con `mailto`)
- `ftp://milazzo@ftp.di.unipi.it:2045/public/`

Nomi e indirizzi: il DNS (1)

- I computer su Internet sono identificati dal loro indirizzo IP;
- Nel contesto del WWW è molto più comune utilizzare nomi di dominio (e.g. `www.di.unipi.it`) anzichè indirizzi IP;
- Il Domain Name System (DNS, RFC 1034 e 1035) è un servizio di naming globale che mappa nomi di dominio in indirizzi IP.

L'utilizzo di nomi di dominio anzichè indirizzi IP offre due vantaggi:

- I nomi di dominio sono più facilmente leggibili e utilizzabili dagli utenti degli indirizzi IP;
- I nomi di dominio offrono un livello di astrazione: se si riorganizza una rete cambiando alcuni indirizzi IP è sufficiente aggiornare il DNS per rendere il cambiamento invisibile agli utenti.

Nomi e indirizzi: il DNS (2)

Il DNS può essere visto come un elenco telefonico globale in cui i nomi anziché essere in ordine alfabetico hanno una struttura gerarchica

- I domini di primo livello sono sigle di nazioni (e.g. `it`,`uk`,`fr`,`...`) o domini generici (e.g. `com`,`edu`,`org`,`...`);
- I domini di livello inferiore sono indicati da destra a sinistra separati da punti: e.g. in `www.di.unipi.it`
  - `it` è il dominio di primo livello;
  - `unipi` è il dominio di secondo livello attribuito all'Università di Pisa
  - `di` è il dominio di terzo livello attribuito al Dipartimento di Informatica;
  - `www` è il nome convenzionale del computer che ospita il web server in un dominio.

Nomi e indirizzi: il DNS (3)

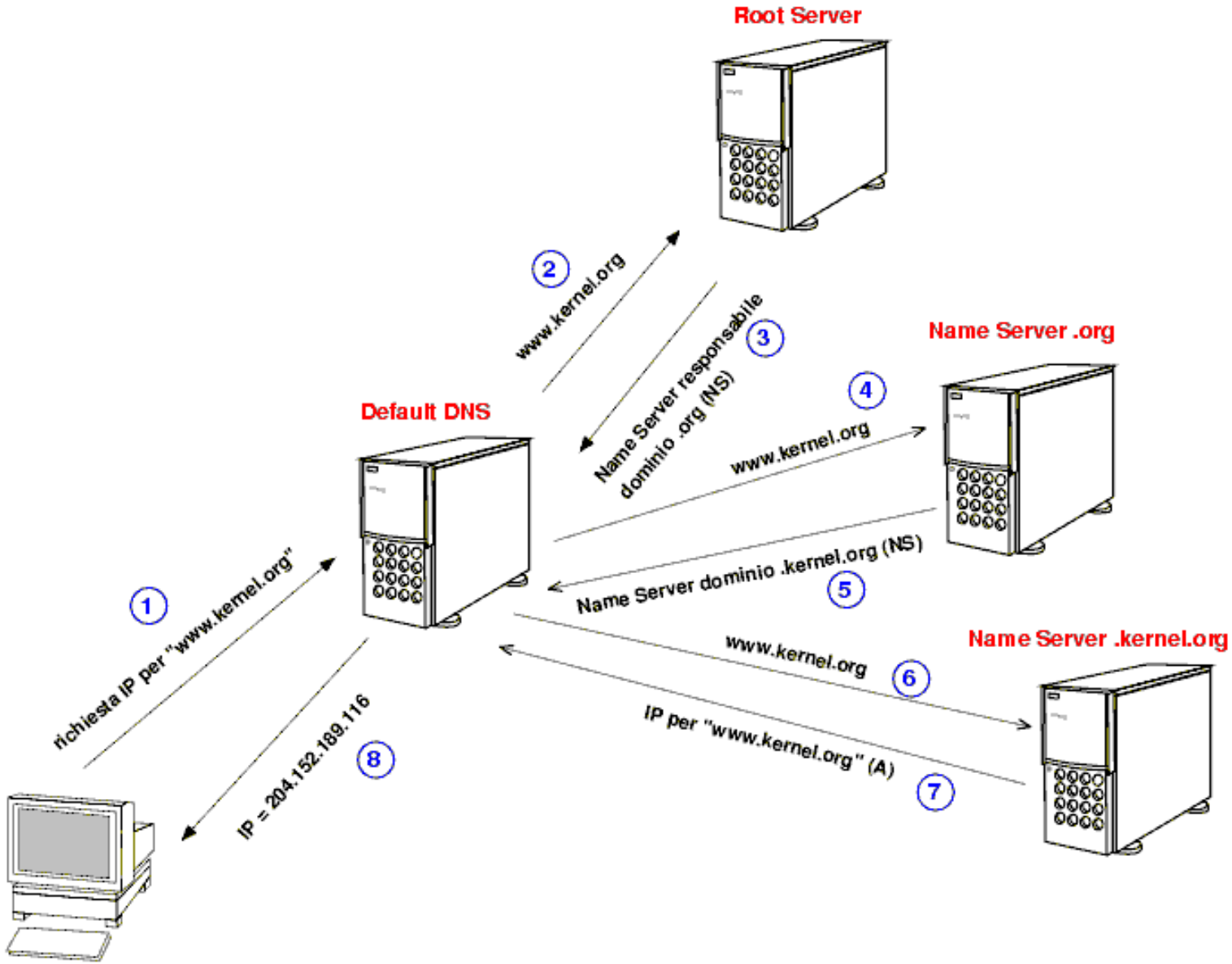
La risoluzione di un nome di dominio tramite DNS è basato sull'iter seguente:

- Ogni computer collegato a Internet deve avere un server DNS di riferimento;
- Quando un computer deve risolvere un nome lo inoltra al proprio server DNS di riferimento;
- Se il server DNS non conosce l'indirizzo IP corrispondente al nome richiesto inoltra la richiesta a un root server (un server DNS che conosce gli indirizzi dei server DNS responsabili dei domini di primo livello;
- Il server DNS invia la richiesta al server respondabile del dominio di primo livello dell'indirizzo richiesto;
- Tale server fornisce la risposta richiesta, o fornisce l'indirizzo del server DNS del dominio di secondo livello di interesse;
- Il procedimento continua finchè la richiesta non riceve risposta.

I server DNS possono fare uso di tecniche di caching delle informazioni scambiate.

Nomi e indirizzi: il DNS (4)

Un esempio:



Descrivere il tipo di un contenuto: MIME

- Come nel caso della posta elettronica, anche per il WWW si usa MIME per distinguere tra diversi tipi di contenuto;
- In questo caso il browser si occuperà di redirigere documenti che non sa visualizzare ad applicazioni specifiche selezionate a seconda del tipo MIME ricevuto (tramite HTTP, come vedremo).

Il protocollo HTTP (1)

L'HyperText Transfer Protocol (HTTP 1.1, RFC 2068 e 2616) è un protocollo di comunicazione client-server:

- per la trasmissione di contenuti a livello applicazione
- basato su un servizio di trasporto affidabile e orientato alle connessioni (come TCP)

Il protocollo HTTP, oltre alle figure del client e del server, prevede la presenza di intermediari, tra cui i proxy.

Un proxy:

- è un programma che riceve richieste HTTP dai client (opportunamente configurati) e le inoltra ai server;
- può applicare tecniche di caching dei contenuti ricevuti dai server per soddisfare autonomamente successive richieste dei client;
- può applicare filtri ai contenuti ricevuti (e.g. parental control).

Il protocollo HTTP (2)

Il protocollo HTTP:

- Normalmente usa una nuova connessione TCP per ogni richiesta (e.g. ogni volta che si segue su un link su una pagina web), anche se esiste la possibliltà di stabilire connessioni permanenti (per gestire in maniera più efficiente una sequenza di richieste);
- E' stateless (senza stato): il risultato di una richiesta non puù dipendere da richieste precedenti;
- Costringe lo sviluppatore di contenuti web a trovare metodi alternativi per memorizzare una forma di stato (e.g. cookies).

Un messaggio di richiesta di un contenuto è costituito da:

- Una riga di richiesta che contiene un "metodo", un URI (generalizzazione di URL) e l'indicazione della versione di HTTP utilizzata;
- Una sezione di header
- Il corpo del messaggio

Il protocollo HTTP (3)

Il metodo può essere uno dei seguenti:

- GET: per richiedere al server l'invio della risorsa indicata nell'URI;
- POST: per inviare informazioni al server;
- HEAD: simile a GET, richiede al server l'invio di informazioni sulla risorsa indicata nell'URI (non la risorsa stessa);
- PUT: per richiedere al server di sostituire una risorsa specificata con un'altra;
- DELETE: per richiedere al server di cancellare una risorsa;
- TRACE: per ricostruire la sequenza di intermediari tra il client e il server;
- OPTIONS: per ottenere informazioni sulle opzioni di comunicazione.

I metodi più comuni (e i soli che approfondiremo) sono GET e POST.

Il protocollo HTTP (4)

Il metodo GET:

- è il metodo più comune, che viene utilizzato ogni volta, ad esempio, che si segue un link in una pagina web o che si inserisce un indirizzo in un browser;
- serve per richiedere al server l'invio di una risorsa indicata nell'URI
- nell'header del messaggio:
  - può porre condizioni sul formato (e.g. tipo MIME) o le caratteristiche della risorsa (dimensioni, data ultimo aggiornamento, ecc..)
  - può includere informazioni addizionali (e.g. browser utilizzato)

Il protocollo HTTP (5)

Esempio di richiesta che usa il metodo GET:

```
GET /beta.html HTTP/1.1
Referer:  http://www.alpha.com/alpha.html
User-Agent:  Mozilla/4.61 (Macintosh; I; PPC)
Host:  www.alpha.com:80
Accept:  image/gif, image/jpeg, image/png, */*
Accept-encoding:  gzip
Accept-language:  en
Accept-charset:  iso-8859-1, utf-8
```

L'URL (o URI) della risorsa può essere ottenuto combinando `Host` con il seguito di `GET`.

Il protocollo HTTP (6)

Il metodo POST:

- Viene usato per trasmettere delle informazioni dal client al server;
- Esempio tipico di informazione trasmessa: i dati inseriti dall'utente in un form in una pagina web;
- I dati trasmessi vengono inseriti nel corpo del messaggio;
- I dati vengono passati alla risorsa (e.g. applicazione) specificata dall'URL.

Il protocollo HTTP (7)

Un messaggio di risposta a una richiesta di un contenuto è costituito da:

- Una riga di stato (status-line) che contiene un codice e una breve descrizione dell'esito della richiesta;
- Una sezione di header
- Il corpo del messaggio di risposta

Le righe di stato più comuni includono:

200 Ok	metodo eseguito con successo
201 Created	metodo eseguito con successo con relativa creazione di nuova risorsa (e.g. PUT)
400 Bad request	errore sintattico nella richiesta
403 Forbidden	richiesta non autorizzabile
404 Not found	URL errato (molto comune)
500 Internal server error	Errore interno (comune con siti web dinamici)

L'header di una risposta a GET include anche il tipo MIME del dato trasmesso

Il protocollo HTTP (8)

Un esempio di richiesta (semplificata):

```
GET /beta.html HTTP/1.1
Host:  www.alpha.com:80
```

E relativa risposta:

```
HTTP/1.1 200 OK
Date:  Mon, 28 Jun 2004 10:47:31 GMT
Server:  Apache/1.3.29 (Unix) PHP/4.3.4
Last-Modified:  Mon, 12 Jun 2004 11:32:12 GMT
Content-Language:  it
Content-Type:  text/html; charset=utf-8
```

Il protocollo HTTP (9)

- HTTP trasmette tutti i dati in chiaro;
- Esiste una versione di HTTP per comunicazioni che richiedono crittografia: HTTPS
- HTTPS:
  - utilizza TCP + SSL per trasmettere i soliti messaggi HTTP;
  - usa lo schema `https:` al posto di `http:` negli indirizzi.