1 Lezione del 25-09-24

1.1 Domain Name System

Il Domain Name System (DNS) è il sistema che usiamo per tradurre gli indirizzi IP in nomi simbolici, più familiari ad utenti umani. L'idea è quella di ricavare un indirizzo IP a partire dal DNS, stabilire una comunicazione col protocollo IP, e trasmettere una pagina web HTML.

La risoluzione di un DNS viene effettuata da un Domain Name Server, solitamente gestito dall'ISP.

I nomi DNS sono formati da più livelli (**domini**) separati da punti, ad esempio `server1.www.pippo.com`. L'ultimo dominio si chiama Top Level Domain (TLD), e da lì in poi, da destra verso sinistra, si assegnano numeri progressivi da 2 (Second Level Domain, Third Level Domain, ecc...). Il TLD è più generico, l'ultimo dominio il più specifico.

Esistono più tipo di TLD:

- Generic top-level domain (gTLD)
  - Unrestricted (.com, .net,. org, ...)
  - Sponsored (.gov, .mil., edu, ...)
- Country code top-level domain (.it, .us, ...)

1.1.1 Registrazione di nomi

I nomi di dominio vengono assegnati e gestiti da particolari organi detti **registri**. Per registrare un dominio ci si rivolge ai registri o agenzie intermedie, e si forniscono alcune informazioni particolari di natura amministrativa.

1.1.2 Risoluzione di nomi DNS

La traduzione dal nome simbolico a quello numerico (cioè l'IP) viene effettuato dai DNS resolver. Solitamente, in verità, i DNS noti sono memorizzati nella cache del nostro browser. Nel caso un DNS non sia trovato nella cache, si cerca in una componente apposita del sistema operativo. Se nemmeno qui si trova il DNS desiderato, si fa una richiesta al server DNS dell'ISP.

A questo punto pure il server DNS controlla nella sua cache. Nel caso nemmeno il server DNS trovi il DNS desiderato, esso si rivolge a un Root name server, ovvero uno dei 13 server delegati a quest'operazione, che restituirà l'indirizzo IP per il dominio di livello più alto del DNS. Questa operazione si ripete su ogni livello del dominio per risolvere il DNS fino al livello più profondo.

1.2 Uniform Resource Locator

L'Uniform Resource Locator (URL) è un sistema per dare nomi a ogni file contenuto all'interno di uno web server. L'URL ha forma:
`http://www.pippo.com/index.php?page=17#article`
Prima si specifica il protocollo (`http://`), poi il dominio (`www.pippo.com`), il percorso o *path* (`index.php`), la stringa di query (?page=17) e il frammento (`#article`).

1.2.1 Protocollo e dominio

La prima parte dell'URL indica il protocollo usato, e la seconda il dominio, che può essere un DNS o un'indirizzo IP, e su cui si puo specificare dopo "::" il numero di porta. Entrambe le parti sono case insensitive.

I numeri di porta di default sono ad esempio 21 per il protocollo ftp, e 80 per il protocollo http.

1.2.2 Stringa di Query

Una stringa di query serve a passare informazioni dall'utente al server. Sono codificati come coppie chiave-valore delimitate dal carattere **&** e precedute dal carattere **?**.

1.2.3 Uniform Resource Identifier

Un Uniform Resource Locator, come un Uniform Resource Name (URN), fa parte di una categoria più ampia detta Uniform Resource Identifier (URI). In particolare, si può dire che:

- Un **URI** identifica una risorsa senza necessariamente contenere particolari informazioni su come trovarla;
- Un **URL** identifica una risorsa e specifica come trovarla;
- Un **URN** fa il lavoro di un URI ma con regole molto più stringenti.

1.3 Richieste HTTP

L'HTTP è il protocollo usato per ottenere pagine web da web server. Quando si accede ad un sito con il browser, questo invia al web server una richiesta HTTP, dove specifica il DNS cercato (più server possono gestire più siti web), richiede una certa risorsa, e trasmette informazioni su di sé (tipo di browser, encoding e lingue accettate, ecc...), ad esempio:

```
GET /index.html HTTP/1.1
Host: pippo.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:15.0) Gecko/20100101
    Firefox/15.0.1
Accept: text/html,application/xhtml+xml
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Connection: keep-alive
Cache-Control: max-age=0
```

dove si nota la richiesta `keep-alive` di mantenere aperta la connessione, e la richiesta `max-age=0` sulla cache che chiede al server di fornire risorse aggiornate.

A questo punto il server risponde alla domanda fornendo informazioni sul tipo di server, sul formato della risorsa inviata, e la risorsa stessa:

```
HTTP/1.1 200 OK
Date: Mon, 25 Sep 2024 02:08:49 GMT
Server: Apache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 4538
Connection: close
Content-Type: text/html; charset=UTF-8
```

```
<html>
...
```

segue la pagina web vera e propria.

1.3.1 Parsing e richieste successive

Solitamente un file HTML contiene ulteriori riferimenti ad altre risorse (stylesheet, altri file html, immagini, ecc..). Per ogni nuova risorsa che si rende necessaria, si fa una nuova richiesta al web server.

1.3.2 Metodi di richiesta

I tipi di richiesta vengono anche detti **metodi**. Esistono più metodi, fra cui:

- **GET:** richiede una risorsa dal server;
- **POST:** invia informazioni al server, ad esempio per trasferire un form.
- **HEAD:** richiede solo l'intestazione o *header* della risorsa, ad esempio per controllare se ha già la versione più recente in cache;
- **PUT:** aggiorna o rimpiazza una risorsa ad un dato URL. Se non esiste, la crea;
- **DELETE:** Rimuove una risorsa a un dato URL.
- **CONNECT:** Stabilisce una connessione col server. Spesso è utilizzato per connessioni SSL (HTTPS);
- **TRACE:** Risponde con la stessa richiesta. Usata per motivi di debug;
- **OPTIONS:** Descrive le opzioni di comunicazione per la risorsa interessata. Utile per trovare quali metodi HTTP sono supportati dal server.

1.3.3 Codici di risposta

HTTP prevede dei codici di risposta alle richieste:

- **1##:** risposte informative, che assicurano il proseguimento dell'operazione;
- **2##:** codici di successo operazione, ad esempio:
  - 200 "OK"
- **3##:** codici di ridirezione (risorse spostate), ad esempio:
  - 301 "Risorsa spostata permanentemente"
  - 304 "Ridirezione temporanea"
- **4##:** errori lato client, ad esempio:
  - 400 "Richiesta malformata"
  - 401 "Non autorizzato"
  - 404 "Non trovato"

  - 414 "URI richiesto troppo lungo"

- **5##:** errori lato server, ad esempio:
  - 500 "Errore server interno"

1.4 Web server

Un web server è un computer che risponde a richieste HTTP. Sul web server gira il cosiddetto **stack**, che comprende il software del server:

- Il sistema operativo;
- Il software web server;
- Un database;
- Un linguaggio di scripting;
- ...

Solitamente ci si riferisce agli stack comuni:

- **LAMP:** Linux, Apache web server, MySQL database, PHP;
- **WISA:** Windows, IIS web server, SQL Server database, ASP.NET.
- **XAMP:** un pacchetto fornito da Apache. XAMPP Apache, MariaDB, PHP, Perl-
- **WAMP:** Windows, Apache web server, MySQL databse, PHP.

1.5 HTML

L'HTML non è un linguaggio di programmazione, ma un linguaggio di **markup**, ovvero usato per dare una struttura a dei documenti.

L'HTML è gestito dal W3C, che produce raccomandazioni (chiamate anche **specifiche**). Nel 1998 il W3C ha proposto uno standard diverso, detto XHTML, che cercava di risolvere alcuni dei problemi dell'HTML, adottando regole di sintassi più severe e basate sull'XML.

Le regole principali dell'XHTML sono:

- I nomi dei tag sono in lower case;
- Gli attributi sono sempre fra virgolette;
- Tutti gli elementi devono avere un elemento di chiusura (o chiudersi da soli).

Per aiutare gli sviluppatori, due versioni di XHTML furono create: XHTML 1.0 Strict e XHTML 1.0 Transitional.

- La versione **strict** doveva essere renderizzato da un browser usando le regole di sintassi più severe;
- La versione **transitional** aveva delle regole più rilassate, ed era pensato come strumento per la transizione temporanea da HTML a XHTML.

1.5.1 Validatori

Parte degli sforzi di questi anni hanno dato luogo allo sviluppo di **validatori**, ovvero strumenti atti a validare che un dato documento HTML rispetti determinati standard.

Nella metà degli anni 2000, XHTML 2.0 propose un cambiamento sostanziale all'HTML, che abbandonava la compatibilità con HTML e XHTML 1.0.

In risposta, sì formo un comitato detto Web Hypertext Application Technology Working Group (WHATWG) all'interno del W3C. Il lavoro di questo comitato ha portato all'ultima versione, l'HTML5, caratterizzato da:

- Specifica non ambigua di come i browser dovrebbero gestire il markup invalido;
- Un framework aperto e non prioritario (JavaScript) per lo scripting;
- Compatibilità con il web già esistente.

1.6 Sintassi dell'HTML

I documenti HTML sono composti da contenuti testuali ed elementi HTML.

1.6.1 Elementi

Gli elementi HTML sono formati da:

- Il nome dell'elemento o **tag** racchiuso fra freccette (< e >);
- Eventuali **attributi**;
- Il contenuto dentro il tag.

Ad esempio:

```
<a href="http://www.pippo.com"> Pippo </a>
```

1.6.2 Elementi vuoti

Un elemento può essere vuoto, ovvero non contenere contenuti. In questo caso si può adottare un *trailing slash* opzionale:

```
<img src="pluto.gif" alt="pippo" />
```

1.6.3 Annidamento di elementi

Gli elementi HTML sono effettivamente annidati, ovvero possono contenere altri elementi HTML. In questo caso si stabiliscono le solite relazioni padre-figlio.

1.6.4 Markup semantico

L'HTML ha il compito di definire la struttura semantica del documento, e non come questo viene mostrato, ad esempio su più dispositivi. A occuparsi di questo sono i Cascading Style Sheets (CSS).

Questa separazione è utile a più scopi:

- **Mantenibilità:** il markup semantico rende più semplice la modifica di pagine graficamente complesse;

- **Prestazioni:** le pagine semantiche sono più veloci da scrivere e da scaricare, il CSS può essere messo in cache;
- **Accessibilità:** strumenti come le lettura dello schermo sono più semplici da implementare su documenti semantici;
- **Ottimizzazione di motori di ricerca:** il markup semantico rende il sito più semplice da vedere per i motori di ricerca.

1.6.5 Struttura di un documento HTML

Un documento HTML semplice si presenta come:

```
<!DOCTYPE html>
<html>
<head lang="en">
<meta charset="utf-8">
<title>Share Your Travels -- New York - Central Park</title>
<link rel="stylesheet" href="css/main.css">
<script src="js/html5shiv.js"></script>
</head>
<body>
<h1>Main heading goes here</h1>
...
</body>
</html>
```

Vediamo le sue componenti:

- `DOCTYPE` specifica il tipo di documento, in questo caso HTML;
- `html` è un nodo radice, opzionale, da cui partono:
  - `head`, che è la testata della pagina (banalmente il titolo). Si noti come qui è specificato il character set (qui utf-8);
  - `body`, che contiene i contenuti veri e propri del sito.