Hyper Text Transfer Protocol (Rappel)

Youssef Saadi

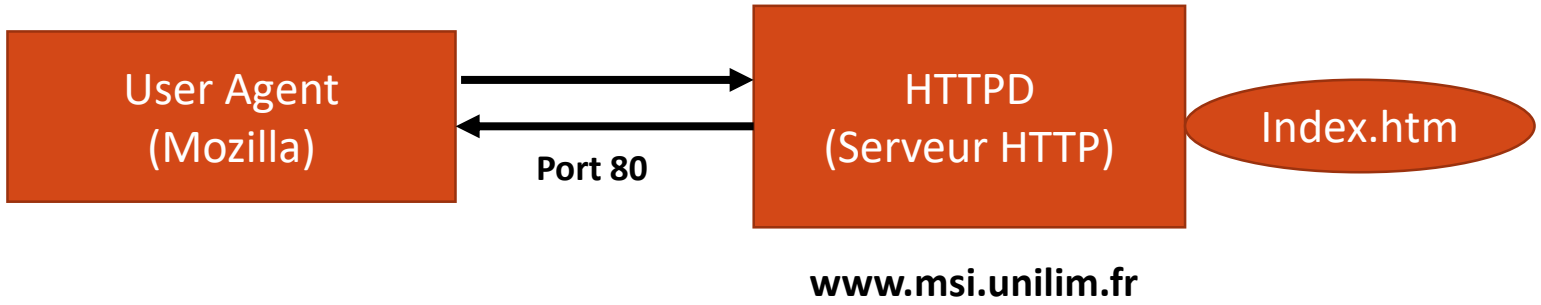
Master Informatique Décisionnelle

Faculté Des Sciences Et Techniques
Université Sultan Moulay Slimane Béni-Mellal

AU: 2019/2020

HTTP : HyperText Transfert Protocol *(RFC 1945 et 2068)*

- Il s'agit d'une norme régissant les échanges (demande d'un document, envoi d'informations, envoi d'un document) entre un client Web et un serveur Web.
- Il s'agit d'un protocole texte (protocole en mode de lignes de caractères) qui utilise la liaison TCP pour la communication entre un client et un serveur.



Introduction

- Fonctionnement *(très simple en HTTP/1.0)*
  - *Connexion*
  - *Demande (GET) d ’un document*
  - *Renvoi du document ou d ’une erreur*
  - *Déconnexion*
- HTTP est un protocole sans états « ***Stateless Protocol*** »
  - *Problème d’identification du client lors d’une session.*
  - *Cookies, HTTP v1.1 (RFC 2068)*

Format des requêtes HTTP

- Dans un navigateur web (Mozilla, Chrome, Internet Explorer, …), on demande le document référencé par l'URL suivante:

```
http://www.msi.unilim.fr/~poulingeas/index.htm
```

- La requête HTTP qui sera générée par le navigateur est la suivante :

```
GET /~poulingeas/index.htm HTTP/1.1
Accept: image/gif, image/jpeg, */*
Accept-Language: fr
Host: www.msi.unilim.fr
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
```

Format des requêtes HTTP

- Une requête HTTP est composé de deux parties principales :
  - Un **en-tête** contenant des lignes dont chacune se termine par les caractères CRLF.
  - Un **corps** éventuel qui contient les informations envoyées au serveur dans le cas d'une commande POST.
- L'en-tête et le corps éventuel sont séparés par une ligne vide c-à-d CRLF.

En tête----------
Ligne de commande
Ligne 2
…
Ligne n

Corps----------

Format des requêtes HTTP

- L'en-tête débute par une ligne de commande de la forme :
  - **Méthode URL Version** du protocole HTTP
    - Exemple : **GET ~poulingeras/index.htm HTTP/1.1**
- Les principales méthodes
  - Récupération d 'un document
    - méthode **GET**
  - Soumission d 'un formulaire
    - méthodes **GET** ou **POST**
  - Envoi de Document et Gestion de Site
    - méthodes **PUT**, **DELETE**
  - *Demande de renseignements sur un document*
    - *méthode **HEAD** (date de création, taille du document, etc.).*

URL/URI

- Une **URI** (Uniform Ressource Identifier) est une chaîne de caractère identifiant une ressource physique ou abstraite.
- Une **URL** est une URI qui identifie une ressource à travers un mécanisme d'accès (par exemple : un protocole réseau + chemin d'accès).

Exemples d'URL :

http://www.library.cornell.edu/nr/cbookcpdf.html

ftp://ftp.lip6.fr/pub/gnat/README

Exemples d'URI :

http://www.library.cornell.edu/nr/cbookcpdf.html	URI absolue
/nr/cbookcpdf.html	URI relative
ftp://ftp.lip6.fr/pub/gnat/README	URI absolue

Format des réponses HTTP

- Les réponses du serveur sont constituées de deux parties :
  - Un en-tête contenant notamment **une ligne de statut** indiquant si la demande a pu être satisfaite.
  - Un **corps** contenant le document demandé
- L'en-tête et le corps éventuel sont séparés par une ligne vide c-à-d **CRLF**.
- Suite à la requête suivante :

  http://www.msi.unilim.fr/~poulingeas/index.htm

  le serveur peut répondre par :

Format des réponses HTTP

```
HTTP/1.1 200 OK
Date: Tue, 18 Jan 2005 15:25:00 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2
Last-Modified: Thu, 04 Nov 2004 11:20:11 GMT
Content-Length: 361
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>
Site de Patrick Poulingeas
</title>
</head>
<frameset cols="18%,82%">
<frame src="choix.htm" name="index" frameborder="0" scrolling="yes"
noresize>
<frame src="present.htm" name="Debut" frameborder="0">
</frameset>
</html>
```

Format des réponses HTTP

- La **ligne de statut de l'en-tête de la réponse** se présente sous la forme suivante :
  - **Version** HTTP **CODE_ÉTAT DESCRIPTION**
  - Exemple : HTTP/1.1 200 OK ou HTTP/1.0 404 Not Found
- Le **premier chiffre** du code d'état retourné par le serveur correspond à une catégorie de réponse. On a les catégories suivantes :

  1: réservé

  2: action exécuté avec succès.

  3: Accès impossible, redirection nécessaire.

  4: Requête invalide.

  5: Erreur du serveur

Quelques codes d'état retournés par le serveur

Code	Signification
200	Opération effectuée avec succès.
201	Envoyé à la suite d'un POST lorsque le document concerné a été établi.
202	Requête enregistrée ; elle sera exécutée plus tard.
204	Le serveur a traité la demande, mais il n'existe aucun document à renvoyer.
301	Le document réclamé a été déplacé et se trouve maintenant à une autre adresse mentionnée dans la réponse.
304	Réponse à une requête GET lorsque le serveur ne renvoie pas de document car il n'y a pas de nouvelle version de celui-ci.
305	Le document doit être accédé via un mandataire (*proxy*).

Code	Signification
400	Erreur de syntaxe : le serveur n'a pas compris la requête.
401	Le client doit s'authentifier avant d'avoir accès au document demandé.
403	Accès au document interdit, même avec une authentification éventuelle.
404	Le serveur n'a pu trouver le document demandé.
405	Méthode non autorisée pour le document spécifié.
406	Le document n'existe pas dans le format exigé par le client.
500	Erreur interne du serveur.
501	Le serveur ne prend pas en charge l'action de la requête.
503	Le serveur est actuellement indisponible.

Fonctionnalités de HTTP

Codification du type

Un client peut indiquer au serveur le type de document qu'il accepte en réponse.

- Le type est défini avec la norme **MIME** (Multipurpose Internet Mail Extension).
- Du coté du client on dispose du champs (optionnel) **ACCEPT**, pour spécifier le type document attendu.
- Plusieurs types peuvent être mentionnés sur la même ligne **ACCEPT** séparés par des virgules.
- Si le serveur ne trouve aucun document de type tel précisé dans **ACCEPT**, alors il renvoi le code d'erreur **406**.

Exemples de types de document

Exemple d'emploi de ce champ dans l'en-tête d'une requête :

Accept: text/*, text/html, */*

Le serveur essaie alors de renvoyer un document en suivant l'ordre de préférence suivant pour le type :

1. text/html
2. text/* → N'importe quel type de document texte
3. */* → N'importe quel type de document

Codification du type

- Du coté du **serveur**, un champ **content-type** est obligatoirement envoyé en réponse au client.
- Le navigateur se base sur ce champ pour décider du traitement en adéquation sur le document renvoyé.

Le serveur détermine le type MIME d'un fichier en se basant sur son extension. Par exemple, Apache dispose à cette fin d'un fichier de correspondance entre extension et type MIME : /etc/apache2/conf/mime.types

Exemple d'emploi du champ Content-Type :

Content-Type: text/html

Fonctionnalités de HTTP
Sélection du jeu de caractères

Le champ **Accpet-Charset** dans une requête indique le jeu de caractères préféré pour la réponse.

- Si le serveur ne trouve pas un document avec le jeu de caractère spécifié renvoie le code d'erreur **406**.
- Le jeu de caractère par défaut associé aux fichier texte est **l'ISO-8859-1.**
- La spécification **1.1** indique qu'un client doit respecter une indication du jeu de caractères quand elle apparaît dans le champ **Content-Type** retourné par le serveur comme par exemple :
  - Content-Type : text/html; charset à iso-8859-1

Fonctionnalités de HTTP

Sélection de la langue

Le champ **Accept-Language** permet au client d'indiquer une préférence quant à la langue du document demandé.

- Le champ **Content-Language**, retourné par le serveur, signale au client quelle est la langue du document qu'il fournit en réponse à sa requête.
- Ces options n'ont d'intérêt que si le serveur dispose de plusieurs version dans différentes langues du document demandé.
- Apache permet de distinguer ces versions en prenant en compte l'extension du fichier correspondant à la langue (utilisation de l'option **Multiviews** dans le fichier de configuration d'Apache).
- On peut donc rencontrer dans un même répertoire du serveur des fichiers du genre : **index.html.de index.html.fr ou index.de.htm index.fr.htm**

Fonctionnalités de HTTP
Longueur du document envoyé

- Le champ **Content-Length** indique la taille en octets du document présent dans le corps du message.
- Ce champ peut apparaître dans une requête (envoi de données d'un formulaire) ou dans une réponse.
- Dans une requête contenant un corps à l'attention du serveur, ce champ est obligatoire.
  - Exemple : Content-Length : **361.**

Fonctionnalités de HTTP
Demande d'une nouvelle version d'un document

- Un client peut demander à recevoir un document en réponse à un **GET** uniquement si ce document a été modifié depuis une certaine date.
- Cette option a un intérêt quand le client gère un cache de pages Web (comme le fait, par exemple un mandataire « proxy ».
- Cette possibilité est obtenue en rajoutant le champ : IF-Modified-Since dans la requête :
  - Exemple : If-Modified-Since : Sat, 29 Oct 1994 19:43:31 GMT
- Si le serveur ne dispose pas d'une version du document plus récente que la date et l'heure indiquées, il renvoie le code **304**.

Fonctionnalités de HTTP
Identification des correspondances

- Le client peut s'identifier auprès du serveur en employant le champ **User-Agent** dans sa requête.
- Ce champ permet au serveur d 'établir des statistiques sur les navigateurs avec lesquels les internautes visitent ses sites.
  - Exemple : User-Agent: links(0,38;)
- Le serveur peut aussi s'identifier auprès du client lors d'une réponse. Ceci s'effectue à l'aide du champ **Server**.
- Pour des raisons de sécurité ce champ n'est parfois pas retourné au client.
  - Exemple : Server : Apache/1.3.26 (Unix) Debian Gnu/Linux PHP

Fonctionnalités de HTTP

Date et délai d'expiration

- Il est possible d'avoir une indication sur la date et l'heure où une opération s'est effectuée (pour une requête ou une réponse).
- C'est le rôle du champ **Date**:

Exemple d'emploi du champ Date :
Date: Tue, 18 Jan 2005 15:25:00 GMT

- Le champ **Last-Modified** renseigne le client sur la date et l'heure de la dernière modification du document renvoyé :

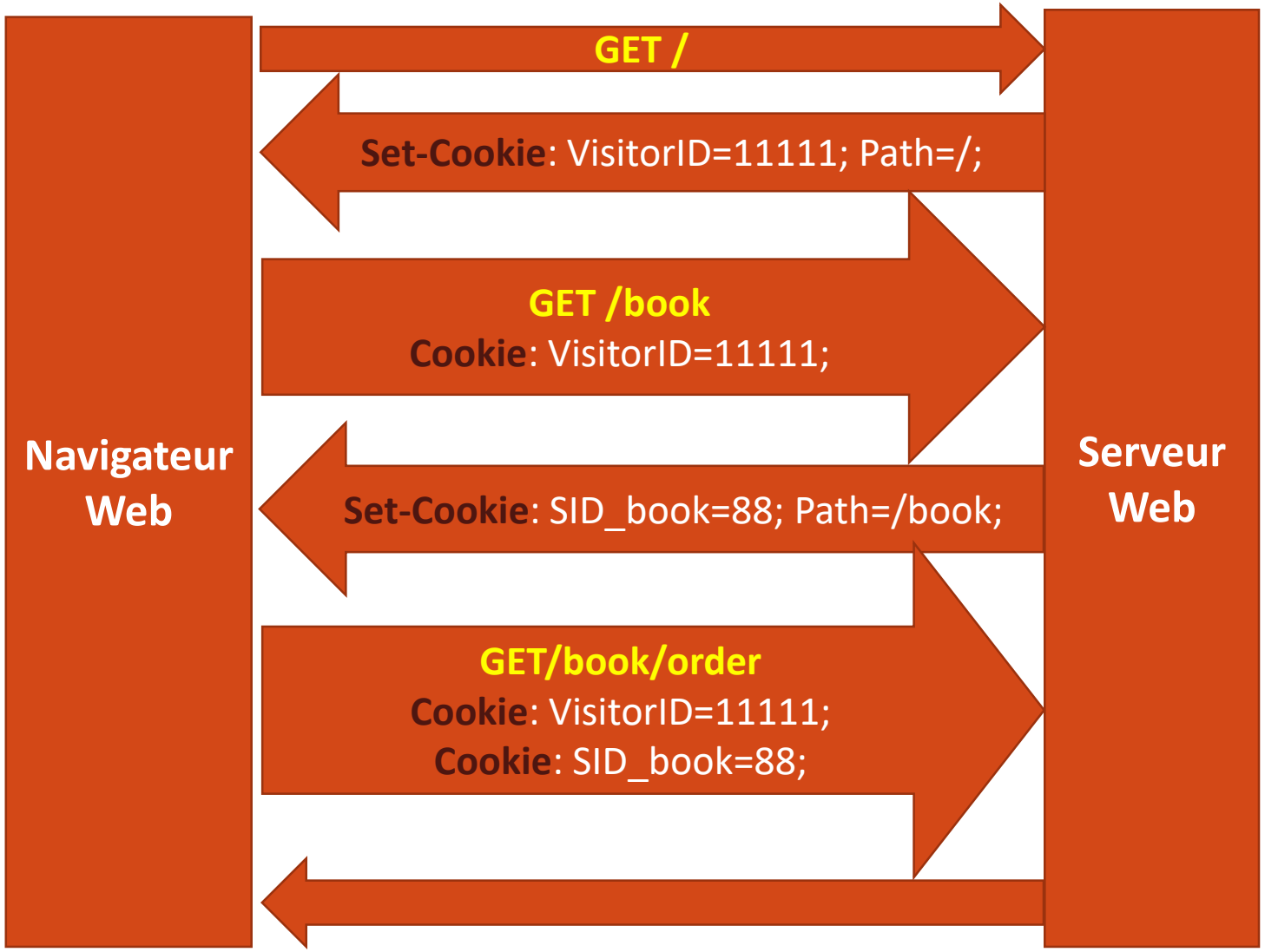
Exemple d'utilisation du champ Last-Modified dans une réponse :
Last-Modified: Tue, 15 Nov 1994 12:45:26 GMT

Les cookies

- Une cookie est une **association** d'un nom à une valeur que **le navigateur fournit automatiquement** à un site après que celui-ci lui est demandé d'enregistrer cette association.
- C'est le **serveur** web qui décide de **demander** au client **d'enregistrer** certaines informations qui lui permettront de reconnaitre l'internaute lors d'une prochaine visite.
- La gestion des cookies se fait par l'ajout de lignes dans l'en-tête HTTP.

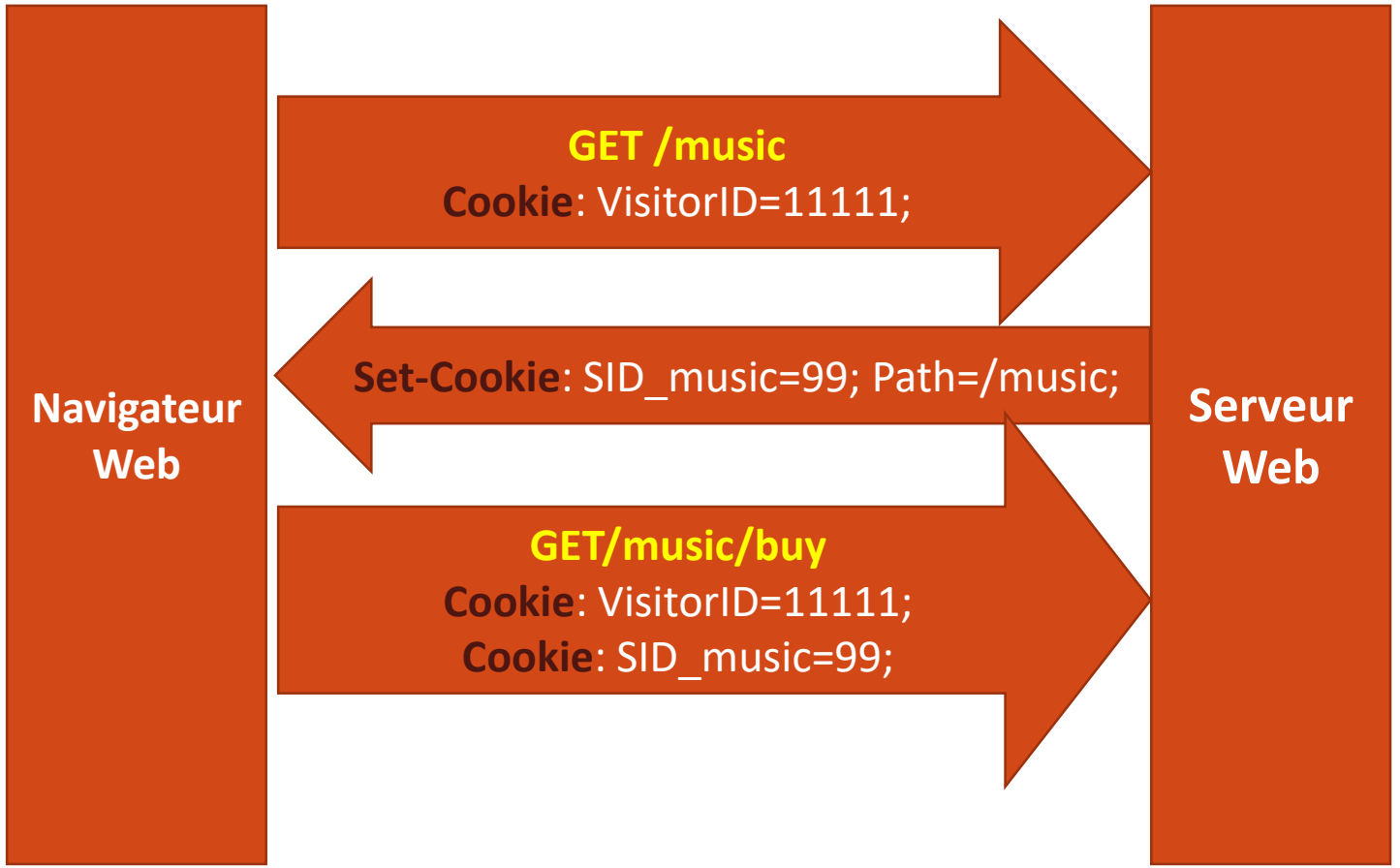
Principe des cookies

Table des cookies du navigateur
VisitorID=11111; Path=/;
SID_book=88; Path=/book;



Principe des cookies

Table des cookies du navigateur
VisitorID=11111; Path=/;
SID_book=88; Path=/book;
SID_music=99; Path=/music;



Remarques

- Plusieurs directives **Set-Cookie** peuvent être retournée par un même serveur.
- Un navigateur devrait satisfaire les exigences nécessaires en termes de gestion de cookies:
  - Pouvoir mémoriser au minimum 300 cookies;
  - Avoir une taille minimale de 4Ko pour chaque cookie;
  - Conserver au minimum 20 cookies par serveur.
- Un cookie peut être supprimer à partir de la machine du client selon la valeur de l'attribut « Expires »:

```
Set-Cookie2:  client="dupond"; Version="1"; Path="/";
              Expires Fri, 11-Feb-2005 14:00:00 GMT
```

- Un cookie sans date d'expiration est retiré lorsque l'internaute quitte son navigateur.

HTTP 2

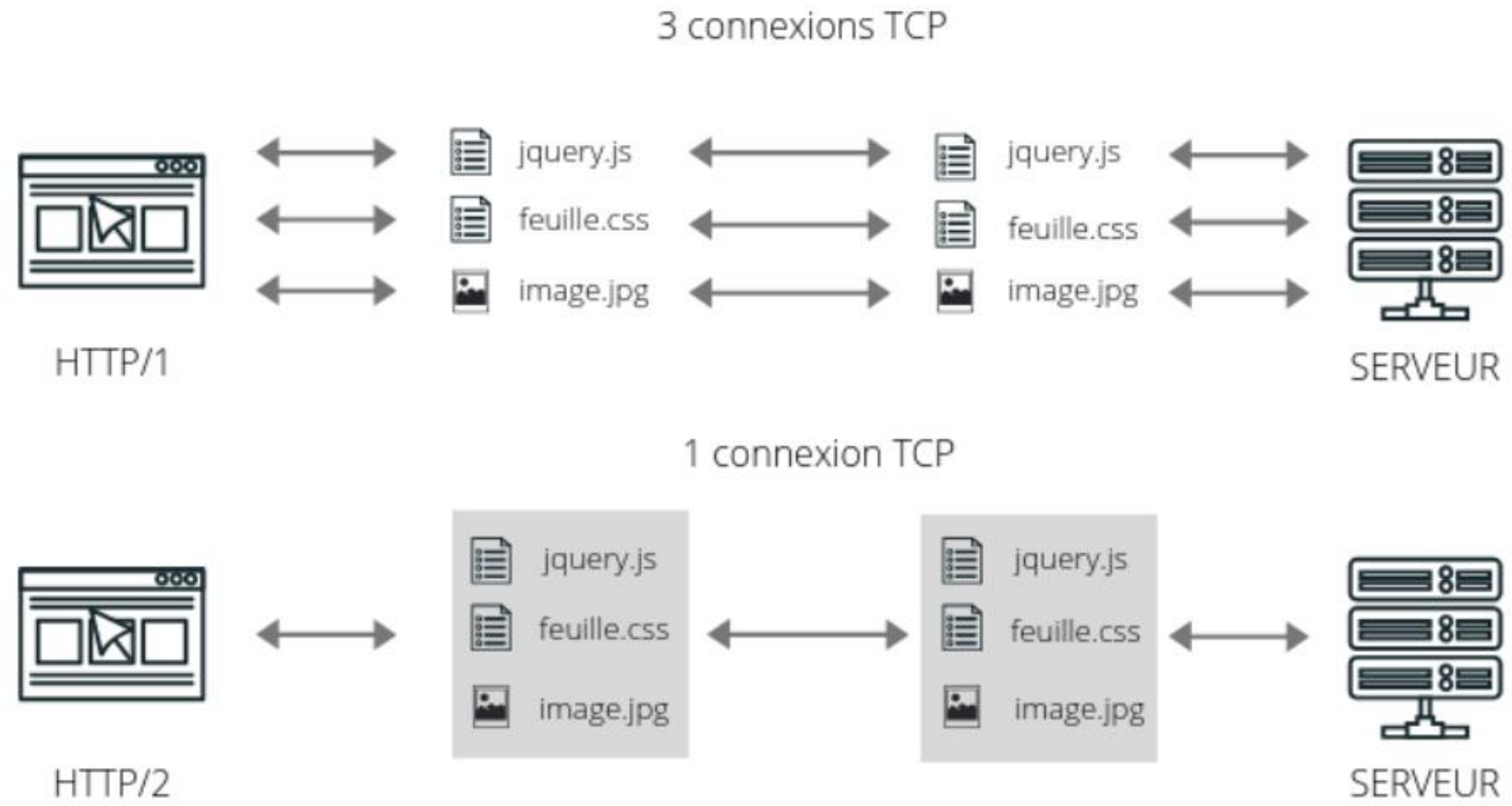
- HTTP/1 est actuellement en train d'être remplacé par une nouvelle version plus modernisée, plus rapide et sécurisée : HTTP/2.
- L'IETF, une communauté internationale de spécialistes du web a commencé le développement de HTTP/2 en 2012. Après avoir été adopté par la plupart des grands navigateurs (Chrome, Firefox, Safari, Opera) en 2015, ce nouveau protocole est devenu accessible pour le grand public. Voyons donc quels sont ses avantages principaux !

Source :https://www.hosteur.com/ressources/articles/adopter-http-2

Avantages du HTTP 2: rapidité

Lorsqu'on utilise HTTP/1, les navigateurs effectuent une requête par ressource. Ceci augmente considérablement le temps de chargement d'une page. Quant à HTTP/2, il utilise ce qu'on appelle le **multiplexage** : le navigateur télécharge avec une seule requête toutes les ressources nécessaires à l'affichage de la page, comme montré sur le schéma ci-dessous.

Avantages du HTTP 2: rapidité



Avantages du HTTP 2: sécurité

Le protocole HTTP/2 peut être utilisé avec ou sans chiffrement TLS. Néanmoins, les navigateurs web les plus populaires (Firefox, Chrome, Safari, etc.) ont décidé d'implémenter uniquement la version d'HTTP/2 avec chiffrement. Les créateurs de sites sont donc obligés d'installer un certificat SSL leur permettant de mieux sécuriser leurs données et celles de leurs clients.