Communication entre client et serveur

Table des matières

1) Modèle Client/Serveur

Deux ordinateurs en réseau peuvent s'échanger des données. Dans la plupart des cas ces échanges ne sont pas "symétriques" : en effet un ordinateur A va souvent se contenter de demander des ressources (fichiers contenant du texte, photos, vidéos, sons...) à un ordinateur B. L'ordinateur B va lui se contenter de fournir des ressources à tous les ordinateurs qui lui en feront la demande. On dira alors que l'ordinateur A (celui qui demande des ressources) est un client alors que l'ordinateur B (celui qui fournit les ressources) sera qualifié de serveur.

En tapant «http ://www.google.fr», votre machine va chercher à entrer en communication avec le serveur portant le nom «www.google.fr» (en faite c'est plus compliqué, pour les puristes nous dirons donc que la communication va être établie avec le serveur www du domaine google.fr, mais bon, pour la suite nous pourrons nous contenter de l'explication « simplifiée »).

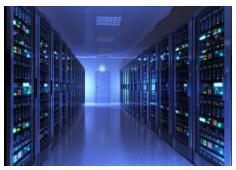
Une fois la liaison établie, le client et le serveur vont échanger des informations en dialoguant :

↦ client : bonjour www.google.fr (ou bonjour www se trouvant dans le domaine google.fr), pourrais-tu m'envoyer le fichier index.html
↦ serveur : OK client, voici le fichier index.html
↦ client : je constate que des images, du code css sont utilisés, peux-tu me les envoyer
↦ serveur : OK, les voici

Évidemment ce dialogue est très imagé, mais il porte tout de même une part de « vérité ». Sur internet, ce modèle client/serveur domine assez largement, même s'il existe des cas où un ordinateur pourra jouer tour à tour le rôle de client et le rôle de serveur, très souvent, des ordinateurs (les clients) passeront leur temps à demander des ressources à d'autres ordinateurs (les serveurs) . Par exemple, comme expliqué dans l'exemple ci-dessus on retrouve cet échange client/serveur à chaque fois que l'on visite une page web. Il y a de fortes chances pour que votre ordinateur personnel joue quasi exclusivement le rôle de client (sauf si vous êtes un adepte du "peer to peer").

N'importe quel type d'ordinateur peut jouer le rôle de serveur, mais dans le monde professionnel les serveurs sont des machines spécialisées conçues pour fonctionner 24h sur 24h. Ils peuvent aussi avoir une grosse capacité de stockage afin de stocker un grand nombre de ressources (vidéos, sons,...).

Afin assurer une continuité de service, dans les sociétés, plusieurs serveurs assurent exactement le même rôle (on parle de redondance). Vous vous doutez bien que Google ne possède pas qu'un seul serveur, en effet, en moyenne, chaque seconde, c'est environ 65000 clients qui se connectent aux serveurs du moteur de recherche de Google. Aucun serveur, même extrêmement performant, ne serait capable de répondre à toutes ces requêtes. Google, Amazon ou encore Facebook possèdent un très grand nombre de serveurs afin de pouvoir satisfaire les demandes des utilisateurs en permanence. Ces entreprises possèdent d'immenses salles contenant chacune des centaines ou des milliers de serveurs (ces serveurs sont rangés dans des armoires appelées "baie serveur").



Souvent les serveurs sont spécialisés dans certaines tâches, par exemple, les serveurs qui envoient aux clients des pages au format HTML sont appelés "serveur web".

Il y a quelques années, le web était dit « statique » : le concepteur de site web écrivait son code HTML et ce code était simplement envoyé par le serveur web au client. Les personnes qui consultaient le site avaient toutes le droit à la même page, le web était purement « consultatif ».

Les choses ont ensuite évolué : les serveurs sont aujourd'hui capables de générer eux-mêmes du code HTML. Les résultats qui s'afficheront à l'écran dépendront donc des demandes effectuées par l'utilisateur du site : le web est devenu dynamique.

Différents langages de programmation peuvent être utilisés « côté serveur » afin de permettre au serveur de générer lui-même le code HTML à envoyer. Le plus utilisé encore aujourd'hui se nomme PHP. D'autres langages sont utilisables côté serveur (pour permettre la génération dynamique de code HTML) : Java, Python...

2) Les URL

La première étape lorsque l'on souhaite accéder à un site Web est la saisie de l'adresse de ce site dans la barre d'adresse du navigateur. Une telle adresse s'appelle une *URL* (pour l'anglais *Uniform Resource Locator* ou " localiseur uniforme de ressource".). La syntaxe (simmplifiée) d'une URL est la suivante :

Protocole ://nom-ou-adresse/document

Avec :

- *Protocole* : définit le protocole utilisé (http, https, ftp) ;
- *nom-ou-adresse* : peut être le nom de domaine ou encore l'adresse IP de la machine faisant office de serveur ;
- *document* : permet de localiser une ressource stockée sur le serveur et que le programme client (un navigateur) souhaite récupérer ou afficher.

3) Le protocole de communication entre client et serveur : le protocole HTTP

3.1) Définition

Un protocole est ensemble de règles qui permet à 2 ordinateurs de communiquer ensemble

Le protocole HTTP (HyperText Transfert Protocol) est un des protocoles de communication entre client et serveur. Ce protocole définit les messages envoyés entre le navigateur et le serveur Web. Les messages envoyés par le client sont appelés des *requêtes*. Ceux envoyés par le serveur sont appelés *réponses*.

3.2) Requête HTTP

Voici une version simplifiée de la composition d'une requête HTTP :

- la méthode employée pour effectuer la requête
- l'URL de la ressource
- la version du protocole utilisé par le client (souvent HTTP 1.1)
- le navigateur employé (Firefox, Chrome) et sa version
- le type du document demandé (par exemple HTML)
- ...

Certaines de ces lignes sont optionnelles.

Voici un exemple de requête HTTP :

```
GET /mondossier/monFichier.html HTTP/1.1
User-Agent : Mozilla/5.0
Accept : text/html
```

Nous avons ici plusieurs informations :

- "GET" est la méthode employée (voir ci-dessous)
- "/mondossier/monFichier.html" correspond à l'URL de la ressource demandée
- "HTTP/1.1" : la version du protocole est la 1.1
- "Mozilla/5.0" : le navigateur web employé est Firefox de la société Mozilla
- "text/html" : le client s'attend à recevoir du HTML

Revenons sur la méthode employée :

Une requête HTTP utilise une méthode (c'est une commande qui demande au serveur d'effectuer une certaine action). Voici la liste des méthodes disponibles :

GET, HEAD, POST, OPTIONS, CONNECT, TRACE, PUT, PATCH, DELETE

Détaillons 4 de ces méthodes :

- GET : C'est la méthode la plus courante pour demander une ressource. Elle est sans effet sur la ressource.
- POST : Cette méthode est utilisée pour soumettre des données en vue d'un traitement (côté serveur). Typiquement c'est la méthode employée lorsque l'on envoie au serveur les données issues d'un formulaire.
- DELETE : Cette méthode permet de supprimer une ressource sur le serveur.
- PUT : Cette méthode permet de modifier une ressource sur le serveur

3.3) Réponse du serveur à une requête HTTP

Une fois la requête reçue, le serveur va renvoyer une réponse, voici un exemple de réponse du serveur :

```
HTTP/1.1 200 OK
Date: Thu, 15 feb 2019 12:02:32 GMT
Server: Apache/2.0.54 (Debian GNU/Linux) DAV/2 SVN/1.1.4
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=ISO-8859-1
<!doctype html>
<html lang="fr">
<head>
<meta charset="utf-8">
<title>Voici mon site</title>
</head>
<body>
 <h1>Hello World! Ceci est un titre</h1>
<p>Ceci est un <strong>paragraphe</strong>. Avez-vous bien compris ?</p>
</body>
</html>
```

Nous n'allons pas détailler cette réponse, voici quelques explications sur les éléments qui nous seront indispensables par la suite :

Commençons par la fin : le serveur renvoie du code HTML, une fois ce code reçu par le client, il est interprété par le navigateur qui affiche le résultat à l'écran. Cette partie correspond au corps de la réponse.

La 1re ligne se nomme la ligne de statut :

- HTTP/1.1 : version de HTTP utilisé par le serveur ;
- 200 : code indiquant que le document recherché par le client a bien été trouvé par le serveur. Il existe d'autres codes dont un que vous connaissez peut-être déjà : le code 404 (qui signifie «Le document recherché n'a pu être trouvé»).

Les 5 lignes suivantes constituent l'en-tête de la réponse, une ligne nous intéresse plus particulièrement :

```
Server: Apache/2.0.54 (Debian GNU/Linux) DAV/2 SVN/1.1.4
```

Le serveur web qui a fourni la réponse http ci-dessus a comme système d'exploitation une distribution GNU/Linux nommée "Debian" (pour en savoir plus sur GNU/Linux, n'hésitez pas à faire vos propres recherches). "Apache" est le coeur du serveur web puisque c'est ce logiciel qui va gérer les requêtes http (recevoir les requêtes http en provenance des clients et renvoyer les réponses http). Il existe d'autres logiciels capables de gérer les requêtes http (nginx, lighttpd...) mais, aux dernières nouvelles, Apache est toujours le plus populaire puisqu'il est installé sur environ la moitié des serveurs web mondiaux !

3.4) HTTPS

Le "HTTPS" est la version "sécurisée" du protocole HTTP. Par "sécurisé" en entend que les données sont chiffrées avant d'être transmises sur le réseau.

Voici les différentes étapes d'une communication client-serveur utilisant le protocole HTTPS :

- le client demande au serveur une connexion sécurisée (en utilisant "https" à la place de "http" dans la barre d'adresse du navigateur web) ;
- le serveur répond au client qu'il est OK pour l'établissement d'une connexion sécurisée. Afin de prouver au client qu'il est bien celui qu'il prétend être, le serveur fournit au client un certificat prouvant son "identité". En effet, il existe des attaques dites "man in the middle", où un serveur "pirate" essaye de se faire passer, par exemple, pour le serveur d'une banque : le client, pensant être en communication avec le serveur de sa banque, va saisir son identifiant et son mot de passe, identifiant et mot de passe

qui seront récupérés par le serveur pirate. Afin d'éviter ce genre d'attaque, des organismes délivrent donc des certificats prouvant l'identité des sites qui proposent des connexions "https".

- à partir de ce moment-là, les échanges entre le client et le serveur seront chiffrés grâce à un système de clé de chiffrement (nous aborderons cette notion de clé de chiffrement l'année prochaine, en terminale). Même si un pirate arrivait à intercepter les données circulant entre le client et le serveur, ces dernières ne lui seraient d'aucune utilité, car totalement incompréhensible à cause du chiffrement (seuls le client et le serveur sont aptes à déchiffrer ces données)

3.5) Les cookies

Un cookie est un fichier qui est déposé par le navigateur sur votre ordinateur lorsque vous surfez sur Internet.

Il s'agit d'un fichier texte généré par le serveur du site web que vous visitez ou par le serveur d'une application tierce (régie publicitaire, logiciel d'analyse du trafic internet, etc.). Il ne pourra par la suite être réutilisé que par le serveur qui l'a déposé en premier lieu.