RES - Découverte de HTTP

OLIVIER LIECHTI
15 mai 2018

1 Objectifs

Cette semaine, il n'y a pas de laboratoire à rendre. L'objectif est de découvrir le protocole HTTP :

— en utilisant des outils tels que telnet, curl ou postman
— en développant un client HTTP simple (pas besoin de lire toute la spécicication)
— en développant un serveur HTTP simple
— en faisant fonctionner le client et le serveur dans 2 containers Docker

Une personne peut travailler sur le client, pendant que l'autre travaille sur le serveur. Une fois la solution complète testée, il est important que les personnes aient bien compris le client et le serveur.

Pour réaliser le client et le serveur, vous pouvez utiliser Java, Javascript ou un autre language. Vous pouvez décider de travailler avec le niveau d'abstraction qui vous convient (Socket API ou librairie de plus haut niveau). Utiliser la Socket API est ce qui vous permettra comprendre les détails (syntaxe) du protocole HTTP.

2 Conseils

Vous pourriez travailler à deux et réaliser une application avec un client HTTP et un serveur HTTP. Le serveur ne donnerait pas accès à des documents HTML, mais il offrirait un service dynamique.

Par exemple, le serveur pourrait être une horloge et exposer ce service via une interface HTTP :

— Le client peut envoyer des requêtes au serveur pour obtenir l'heure courante (le client fait une lecture avec un GET : il demande au serveur de lui donner des informations). Le serveur doit parser la requête et renvoyer une réponse bien formée. Le client doit la parser et la présenter à l'utilisateur.
— le client peut aussi envoyer des requêtes au serveur pour changer l'heure (le client fait une écriture avec un POST). Le serveur doit lire les données envoyées par le client (dans le body et dans le query string). Il doit répondre au client.
— pour bien comprendre la notion de négociation, faites en sorte que le client puisse spécifier le format qu'il préfère (html, json ou xml). Faites en sorte que le serveur honore ce format quand il prépare la réponse. Dans la même idée, faites en sorte que le client puisse envoyer les données de mise à jour (avec le POST) en json ou en xml.