UNIVERSITÉ DE ROUEN

Langages Web 1 – TP 02 2016–2017

Étude du protocole HTTP
Seconde partie

1 Introduction

— Pour ce TP, nous utilisera les mêmes techniques que pour le TP précédent. Les informations et ressources données en introduction de ce dernier restent donc nécessaires pour le présent sujet (utilisation de `netcat`, références `HTTP`, ...).

2 Exercices – Le point de vue du serveur

Nous allons maintenant nous placer du point de vue du serveur. Pour cela, nous allons cette fois utiliser `netcat` pour simuler un serveur HTTP et faire une réponse à un client web. Nous utiliserons des commandes de la forme :

```
cat header body | nc -l 8080
```

où `header` sera un fichier contenant la partie en-tête de la réponse `HTTP` et `body`sera un fichier représentant le corps de la réponse, c'est-à-dire la partie ressource renvoyée. On notera que, selon ce schéma, la réponse faite est indépendante de la requête reçue puisque celle-ci est purement ignorée.

Exercice 1 – *Premier essai - réponse simple*
Placez-vous dans le dossier `reponse-simple` de l'archive associée au TP.
Question 1.1 : Préparez une réponse `HTTP` dans laquelle :
— vous renverrez un code de retour indiquant que la requête a été exécutée sans erreur
— vous utiliserez le mode où la connexion est fermée après la requête.
— vous utiliserez le contenu de `simple.html` pour le flux de la ressource.
et que vous enverrez au navigateur web à l'aide de `netcat` :

```
cat rep_http simple.html | nc -l 8080
```

Faites pointer votre navigateur web sur `http://locahost:8080` pour observer le résultat. À quoi correspond le texte produit par `netcat` sur le terminal ?
Question 1.2 : Recommencez l'opération en indiquant cette fois que la connexion va rester ouverte. Il faudra ajouter `-q -1` à la commande `netcat` :

```
cat rep_http2 simple.html | nc -l 8080 -q -1
```

Qu'observez vous au niveau du comportement du navigateur et de la sortie de `netcat` sur le terminal ? Expliquez.
Question 1.3 : Préparez maintenant une réponse renvoyer cette fois le contenu de `pipe.jpg` et faire en sorte que le navigateur l'affiche :

```
cat rep_http3 pipe.jpg | nc -l 8080 -q -1
```

Pour simplifier, vous pouvez utilisez une connexion fermante.
Question 1.4 : Combinez les réponses de deux dernières questions afin de satisfaire les deux requêtes émises par le navigateur. Attention, le navigateur accepte mal de recevoir une réponse avant d'avoir émis la requête. Nous allons ajouter une temporisation (5 secondes) entre les deux, la commande devrait ressembler à

```
(cat rep_http2 simple.html; sleep 5; cat rep_http3 pipe.jpg)| nc -l 8080 -q -1
```

Vous pouvez supprimer la partie après le pipe (`|`) inclus si vous voulez observer dans le terminal ce que recevra exactement le navigateur.
Question 1.5 : Reprenez la première question (envoi de `simple.html`) en demandant au navigateur d'afficher le code source du flux `HTML` du fichier sans l'interpréter.
Question 1.6 : Formez une réponse dans laquelle vous enverrez le contenu de `simple.html` en demandant au navigateur de l'enregistrer sur le disque (ouverture d'une boite de dialogue *ouvrir avec/Enregistrer sous).*

Exercice 2 – `HTTP` all mighty

Placez vous dans le dossier `ressources` *de l'archive associée au TP.*
Question 2.1 : *À l'aide de la commande* `file --mime`*, identifiez le contenu des chacun des fichiers présents.*
Question 2.2 : *Formulez des réponses dans lesquelles vous enverrez chacun des fichiers au navigateur (en connexion close pour faire simple) en faisant varier le type que vous indiquez dans l'en-tête de la réponse. Vous remplirez la matrice suivante avec le comportement du navigateur (vous compléterez la ligne* type réel *avec le résultat de la première question) :*

	res1	*res2*	*res3*
type réel	•	•	•
`image/jpeg`	•	•	•
`text/html; charset=utf-8`	•	•	•
`text/html; charset=iso-8859-1`	•	•	•

IMPORTANT *: utilisez Firefox, n'utilisez pas Chromium.*
Question 2.3 : *Quelle conclusion peut-on tirer de ceci ?*
Question 2.4 : *Copiez ces fichiers en leur donnant une extension correspondant à leur type et essayez de les ouvrir directement à l'aide du navigateur (sans passer par le réseau). Que constatez vous ?*

Exercice 3 – Cachons ces ressources que nous ne saurions voir

Choisissez un fichier de ressource parmi ceux des exercices précédents.
Question 3.1 : *Faites une réponse pour envoyer cette ressource au navigateur en faisant de telle sorte qu'il ne vous la demande plus durant l'heure suivante. Il est préférable d'utiliser une URL avec un chemin autre que* `/`*.*
Question 3.2 : *Faites une réponse en faisant en sorte que le navigateur redemande la ressource avec une requête conditionnelle.*

Exercice 4 – Les cookies (petit cours de pâtisserie)

Question 4.1 : *Rappelez ce qu'est un* `cookie`*.*

La liste des cookies enregistrés sur votre ordinateur par firefox peut être consultée dans `Édition>Préférences>Vie Privée>Supprimer des cookies spécifiques`*. (note : il est nécessaire pour cela d'avoir la règle de conservation, dans le même menu, positionnée sur* `conserver l'historique`*)*
Question 4.2 : *Vérifiez l'absence de cookies associés à* `localhost` *(utiliser le filtrage).*
Question 4.3 : *En préparant une réponse envoyée dans* `netcat` *comme nous l'avons fait à l'exercice* Le point de vue du serveur*, forcez Firefox à enregistrer un cookie nommé* ingredient *avec pour valeur* `"farine,sucre,chocolat"` *et constatez son apparition.*
Question 4.4 : *Sans préparer de réponse cette fois, faites à nouveau une requête sur netcat (sans faire de cat sur l'entrée standard de netcat). Constatez-vous une différence dans la requête formulée par Firefox ?*

Exercice 5 – Les redirections

Dans cet exercice il ne s'agit pas de redirection d'entrées/sorties standards mais de redirection du navigateur. Nous avons vu en cours qu'un serveur Web peut donner comme instruction au navigateur de se rediriger vers une autre URL.

Question 5.1 : *Rappelez comment s'effectue une redirection avec le protocole HTTP.*
Question 5.2 : *Lancez un* `netcat` *simulant un serveur qui redirigera tout client qui s'y connecte vers l'adresse* `http://www.google.fr`.
Question 5.3 : *Est-il possible de lancer un téléchargement tout en affichant une nouvelle page à partir d'une unique URL ?*

Exercice 6 – La couche sécurité

Ce que nous avons avec la commande netcat durant les exercices précédents, nous pourrions le faire avec un renifleur de paquets (comme wireshark [1]*) sur un routeur situé sur le chemin entre l'ordinateur d'un internaute et un serveur (sa banque par exemple).*

Pour sécuriser la communication, on nous enseigne qu'il faut veiller à ce que `HTTPS` *soit le protocole utilisé.*
Question 6.1 : *En quoi le protocole* `HTTPS` *diffère-t-il du protocole* `HTTP` *?*
Question 6.2 : *À l'aide de la commande* `netcat`*, mettez vous en écoute sur un port de votre machine (numéro de votre choix mais supérieur à 1024), et demandez à votre navigateur de se connecter en utilisant le protocole* `HTTPS` *sur ce port. Qu'obtenez-vous comme résultat ? En quoi diffère-t-il de ce que vous obteniez avec le protocole HTTP ?*
Question 6.3 : *Répétez exactement la même expérience. Obtenez-vous exactement le même résultat ? Expliquez pourquoi (alors que le résultat final devrait être exactement le même) et comment cela se fait ?*
Question 6.4 : *Répétez l'expérience jusqu'à ce que vous soyez capable de lire un morceau de texte en clair dans le flux chiffré. De quelle information s'agit-il ? Pourquoi est-elle transmise en clair ?*

Exercice 7 – Serveur mandataire

Question 7.1 : *Rappelez le rôle d'un serveur mandataire.*
Question 7.2 : *Démarrez un* `netcat` *en écoute sur un port* `Y` *de votre choix. Réglez votre navigateur afin qu'il utilise* `localhost:Y` *comme serveur mandataire et demander lui de charger une page. Consignez et expliquez vos observations.*

Exercice 8 – Exercice bonus

En utilisant le langage de votre choix, écrivez un script qui produit automatiquement en boucle des réponses HTTP à un client (en utilisant netcat ou bien une connexion directe en fonction du langage choisi) :
Question 8.1 : *Les ressources seront prises*

1. Anciennement Ethereal.