Université Paris Diderot – Paris 7 **Programmation réseau**
L3 Informatique **Anne 2010–2011, 2ème semestre**

TP n°3

Protocoles : découverte

1 Expérimentations sur les services réseau

1. Déterminez le port associé au service `discard`.
2. En utilisant successivement les commandes `host`, `nslookup` et `dig`, déterminez l'adresse IP de `www.free.fr`, puis de `www.informatique.univ-paris-diderot.fr`. Quels sont les noms d'hôtes associés aux adresses obtenues ?
3. À l'aide de la commande `dig`, déterminer les serveurs de courrier électronique du réseau `informatique.univ-paris-diderot.fr` et `free.fr`. (voir la page de manuel de `dig`).
4. À l'aide de la commande `telnet`, déterminez l'heure (service `daytime`, RFC 867) qu'il est sur la machine `nivose` et `lucien`.
5. À l'aide de la commande `finger`, affichez les informations de quelques utilisateurs (RFC 1288).

2 Le courrier *à la main*

Le protocole SMTP (*Simple Mail Transfer Protocol*, RFC 2821, port tcp 25) sert à envoyer du courrier électronique (*e-mail*) à des utilisateurs locaux ou distants. Il s'agit d'un protocole dit *requête-réponse*, dans lequel le dialogue consiste pour le client à envoyer une commande au serveur puis attendre la réponse de ce dernier, et de recommencer.

> **Remarque :** les techniques utilisées dans cette partie permettent, en théorie, d'envoyer des mails en se faisant passer pour quelqu'un d'autre. Une telle activité est totalement illégale, et nous vous déconseillons fortement de mettre à l'épreuve les capacités d'investigation de nos administrateurs système et réseau.

Les commandes SMTP les plus utiles sont les suivantes :

- « `HELO` *machine* » : à envoyer au début d'une connexion SMTP. La chaîne *machine* est le nom de l'hôte à partir duquel vous vous connectez.
- « `MAIL FROM:` *utilisateur* » : commence une transaction SMTP visant à envoyer un message. La chaîne *utilisateur* est l'adresse de l'auteur du message (de la forme `<user@domain>`).

– « `RCPT TO:` *utilisateur* » : déclare un destinataire de la transaction courante ; peut être répétée plusieurs fois.
– « `DATA` » : déclare le début de l'envoi du message, lequel commence à partir de la ligne suivante et se termine par une ligne ne contenant que le caractère point « `.` »tout seul sur une ligne.
– « `QUIT` » : termine une connection SMTP.

Le message passé à la commande « `DATA` »doit avoir le format défini par le document de normalisation RFC 2822. Il doit commencer par une série d'en-têtes (« `From:` », « `To:` », « `Subject:` », etc.) suivis d'une ligne vide, suivie du corps du message lui-même.

1. En vous connectant directement au port SMTP d'une machine (par exemple `nivose`) à l'aide de la commande « `telnet` », envoyez un mail à l'un de vos collègues.
2. Répétez l'expérience précédente en donnant des adresses différentes dans l'enveloppe (commande « `RCPT TO:` »de SMTP) et dans le message (entête « `To:` »de RFC 822). Que se passe-t-il ?

3 Clients Java

En Java, il est possible d'ouvrir une connexion à un service de *port* donné sur une *machine* donnée en utilisant la classe `java.net.Socket`. Par exemple :

```
Socket s = new Socket(machine,port);
```

Les méthodes `getOutputStream()` et `getInputStream()` permettent de retrouver des flux d'entrées/sorties (respectivement de type `OutputStream` et `InputStream`) pour écrire et lire sur cet objet de communication. De façon à manipuler ces objets, regardez comment les utiliser avec les classes `PrintStream`, `InputStreamReader` et `BufferedReader`.

Écrivez un programme Java qui se connecte au service `daytime` d'une *machine* donnée en paramètre et récupère la date pour l'afficher à l'écran.

4 Des applications de « pro » : *mailer* et *wget*

1. Écrivez un programme Java *jmailer* qui se connecte au service `smtp` du réseau et envoie un courrier électronique et qui lit et affiche les réponses du serveur (une réponse d'une ligne par « ordre »). N'oubliez pas de traiter les codes de retour attendus et d'afficher un message d'erreur si ils ne correspondent pas au bon comportement.
2. Écrivez un programme Java *jwget* qui prend en argument une URL (comme `http://www.example.com/index.html`), et récupère la page WEB correspondante pour l'écrire dans le fichier correspondant (ici le fichier de nom `index.html`).

Pour récupérer le document d'URL `http://machine/chemin` il faut se connecter au service `http` sur la *machine* et envoyer la requête suivante :
`GET /chemin`
Attention : Il est possible (comme à l'UFR) que la connexion directe à la machine soit interdite (pour des raisons de sécurité par exemple), dans ce cas votre fournisseur d'accès vous fournit généralement un intermédiaire (serveur dit *proxy*). Si c'est le cas, il faut vous connecter sur le serveur intermédiaire (à l'UFR il s'appelle `cache`) et sur le port (à l'UFR c'est le port `3128`) qui lui correspond et envoyer la requête :
`GET http://machine/chemin`
Ceci pour faire en sorte que l'intermédiaire effectue la requête à votre place et vous renvoie la réponse... Pour décomposer une URL, nous vous conseillons de consulter la documentation de la classe `java.net.URL`.

Indication : La classe `InetAddress` vous permet d'avoir des informations sur la machine où vous exécutez votre programme, par exemple :
`InetAddress.InetAddress.getLocalHost().getHostName();`.