TP Réseaux

4 avril 2003

Ce document est disponible à l'URL http://www.liafa.jussieu.fr/~sighirea/cours/stic/

Cette feuille de TP est une liste exhaustive des travaux pratiques à réaliser pour le cours "Réseaux".

1 Expérimentation avec les applications réseaux

La liste des services plus ou moins standardisés et des ports qui leur sont associés est donnée, sur toute machine Unix, dans le fichier `/etc/services`. Les descriptions précises de beaucoup de ces services sont données par les documents dits *Request For Comments* (RFC), disponibles en particulier par le protocole FTP sur (par exemple) `ftp.jussieu.fr` dans le répertoire `/pub/rfc/rfc/`.

0. A l'aide de la commande `ftp`, récupérez le texte du RFC 821.

Dans la suite, vous êtes invités à vous connecter directement aux services mentionnés à l'aide de la commande `telnet`. Une connection telnet permet :

- de travailler sur un ordinateur distant en mode caractère : `telnet -l user host`
- de communiquer avec un autre service réseau : `telnet host port-service`

Une connection `telnet` peut être abandonnée à tout moment en tapant `^]` suivi de `quit`.

1. A l'aide du service `daytime` (port 13), déterminez l'idée que se font de l'heure courante quelques unes des machines du réseau local.

2. Le protocole SMTP (Simple Mail Transfer Protocol, RFC 821, port 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 client envoie une commande au serveur puis attend la réponse de ce dernier.

Les commandes SMTP les plus utiles sont les suivantes :

- `HELO machine` : à envoyer au début d'une connection SMTP. La chaîne `machine` est le nom de la machine à partir de laquelle 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 repétée plusieurs fois.
- `DATA` : déclare le début du message, qui commence sur la ligne suivante et se termine par un point tout seul sur une ligne. Cette commande a pour effet de terminer la transaction courante.

- `QUIT` : termine une connection SMTP.

Le message passé à la commande `DATA` doit avoir le format défini par le RFC 822. Il doit commencer par une série d'entêtes (`From:`, `To:`, `Subject:`, etc.) suivis d'une ligne vide suivie du corps du message lui-même.

1. En vous connectant directement au port 25 d'une machine à l'aide de la commande `telnet`, envoyez un mail à l'un de vos collègues.
2. Répétez l'expérience précédente en donnant des addresses différentes dans l'enveloppe (commande `RCPT TO:` de SMTP) et dans le message (entête `To:` de RFC 822). Que se passe-t-il ?
3. Regarder le répertoire `/var/mail`. A quoi correspondent les fichiers de ce répertoire ? Regarder le contenu du fichier ayant votre nom de login.

3. Le protocole HTTP (RFC 2616, port 80) sert à transférer des documents multi-média. Comme vu en cours, il s'agit également, d'un protocole *requête-réponse*.

- Connectez-vous à un serveur HTTP en utilisant `telnet`.
- Déterminez quelle est la version de HTTP supportée par le serveur.
- Essayez d'obtenir les différentes classes de réponse : succès, page non trouvée, inchangé.

4. Le protocole DNS (RFC 921, port 53) sert à résoudre les correspondances entre les noms des machines et leurs noms canoniques ou leurs adresses IP.

1. A l'aide de la commande `host -v` déterminez l'adresse IP de `www.lsv.ens-cachan.fr`, celle de `ftp.jussieu.fr`, celle de `ftp.lip6.fr`. Ces machines sont-elles toutes distinctes ?
2. A l'aide de la commande `host -v -t ns` déterminez un serveur de noms du domaine `dptmaths.ens-cachan.fr`.
3. A l'aide de la commande `host -v` déterminez de façon authoritative l'adresse de la machine `www.lsv.ens-cachan.fr`.
4. Quelles machines desservent le courrier destiné aux domaines `liafa.jussieu.fr`, `lsv.ens-cachan.fr` ?

2 Monitoring du réseau

1. A l'aide de la commande `ping` déterminez le RTT entre votre machine et quelques machines connues. Observez le comportement de la commande quand vous essayez de contacter une machine hors du réseau local.

2. A l'aide de la commande `traceroute` déterminez la route entre votre machine et les machines `www.liafa.jussieu.fr` et `www.lsv.ens-cachan.fr`.

3. La commande `netstat` permet de connaître l'état des connexions réseau en cours, les tables de routage, les statistiques des interfaces réseau, etc.

1. A l'aide de la commande `netstat -inet`, déterminez l'adresse IP, l'adresse de diffusion (broadcast) et le masque de votre machine.
2. A l'aide de la commande `netstat -t`, déterminez les connexions TCP ouvertes sur votre machine.

3. A l'aide de la commande `netstat -r`, indiquez quel sera le routeur contacté pour sortir du domaine `lsv.ens-cachan.fr`.

4. A l'aide de la commande `netstat -s` déterminez quel est le nombre de paquets IP reçus par votre machine et combien d'entre eux ont été corrompus.

3 Programmation réseau

Un client en Java Ecrivez un programme Java qui se connecte au service `daytime` d'une machine courante et affiche l'heure donnée par celle-ci. Votre programme commencera par obtenir l'addresse du serveur au moyen de la méthode statique `getByName` de la classe `java.net.InetAddress`. Il construira ensuite une socket et le connectera au serveur en une seule étape au moyen du constructeur

```
java.net.Socket(java.net.InetAddress address, int port)
```

Il construira ensuite un `java.io.InputStream` au moyen de la méthode `Socket.getInputStream`, et construira enfin un `java.io.Reader` au moyen du constructeur

```
InputStreamReader(InputStream)
```

On pourra lire à partir de la connection au moyen de la méthode `Reader.read()`.

Un client en C Ecrivez maintenant le même programme en C. Ce programme aura la même structure que le programme Java, mais devra gérer quelques détails en plus. Il faudra tout d'abord inclure les fichiers suivants :

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <unistd.h>
```

L'addresse du serveur peut être déterminée au moyen de la fonction `gethostbyname(3)`, qui retourne un pointeur sur une `struct hostent`. Il faudra d'abord vérifier que le champ `h_addrtype` de cette dernière vaut bien `AF_INET`, puis construire une `struct sockaddr`. On peut le faire à l'aide du code suivant :

```
struct hostent *host;
struct in_addr inaddr;
struct sockaddr_in sin;
struct sockaddr *sa;
...
inaddr = *(struct in_addr*)host->h_addr;
sin.sin_family = AF_INET;
sin.sin_port = htons(13);
sin.sin_addr = inaddr;
sa = (struct sockaddr*)&sin;
```

Il faudra ensuite construire une socket comme suit :

```
fd = socket(PF_INET, SOCK_STREAM, 0);
```

qu'on pourra connecter à l'aide de la fonction `connect(2)`. Enfin, on pourra lire les données retournées par le serveur à l'aide de la fonction `read(2)`.