Nancy-Université — Master IST-IE — 2007–08 — UE 103b —

Les serveurs — TD n° 1

Cette première séance a un double objectif : d'une part revoir quelques notions d'adressage vues en cours, et par cet intermédiaire, tester quelques commandes utiles, sous Windows ; d'autre part, réaliser par vous- même (avec un peu d'aide, bien sûr), une communication basée sur le modèle client-serveur.

Pour chaque question, une démonstration vidéo-projetée sera effectuée et commentée, et vous serez ensuite invités à tester chaque commande. Toutefois, la démonstration sera effectuée sous environnement Unix, ce qui entraînera de petites différences. N'essayez pas d'aller trop vite, sinon cela n'aura aucun intérêt.

1 Un peu d'adressage...

1.1 Connaître l'adresse IP de sa machine

Quand vous tentez de réaliser un jeu en réseau, ou seulement un petit réseau avec plusieurs ordinateurs, vous avez besoin de connaître les adresses IP (pour être sûr que toutes les machines sont sur le même sous-réseau).

- Lancez l'icône Invite de commandes sur le bureau. Si elle n'apparaît pas, allez dans le menu Démarrer ▶ Programmes ▶ Accessoires ▶ Invite de commandes .
- Saisissez la commande `ipconfig` puis appuyez sur Entrée.
- Notez l'adresse IP de votre machine pour la suite.

1.2 Connaître l'adresse MAC de sa carte

Cela peut être utile lorsque vous désirez vous connecter à un réseau sécurisé : l'administrateur réseau vous demande alors votre adresse MAC (carte ethernet ou wifi) pour enregistrer votre ordinateur dans la base et ainsi vous autoriser l'accès.

- Toujours dans l'invite de commandes, saisissez la commande `ipconfig /all` puis appuyez sur Entrée.
- L'adresse MAC apparaît à la ligne `Adresse physique`.

1.3 Quelle est l'adresse IP de...

La commande `nslookup` permet d'interroger un serveur de noms pour obtenir des informations concernant un domaine ou un hôte.

- Saisissez `nslookup www.google.fr`.

Cette commande permet également de trouver un nom de domaine à partir d'un adresse IP.

- Saisissez `nslookup` suivi de l'adresse IP de votre machine que vous avez notée à la question 1.1.

Vous obtenez ainsi le nom de domaine de votre machine, qui devrait être de la forme `plj102pc`*nn*. Le numéro correspond à celui du poste sur lequel vous êtes, indiqué en haut de votre écran.

1.4 Suis-je seul au monde ?

Vous venez de finir le câblage de votre petit réseau, vous avez établi les adresses IP et le masque de sous-réseau : tout est prêt, mais un des ordinateur reste « invisible ». Il existe une commande pour vérifier la connectivité d'un autre ordinateur : `ping`.

- Demander le nom de la machine de votre voisin. Saisissez ensuite `ping` suivi du nom de machine de votre voisin.
- On peut également procéder avec des adresses IP. Saisissez `ping /a` suivi de l'adresse IP de la machine de votre voisin.

1.5 Quel est mon chemin ?

Il est possible de suivre l'itinéraire liant son ordinateur à n'importe quel autre ordinateur, c'est-à-dire voir l'ensemble des routeurs empruntés. Cet itinéraire est asymétrique (le chemin de retour n'est pas forcément identique), il dépend du point de départ (l'itinéraire peut-être différent, même pour une destination identique).

- Saisissez la commande `tracert www.google.fr`
- Saisissez à nouveau la même commande et comparer les itinéraires.

2 Le modèle client-serveur

Après cette introduction aux commandes « réseaux » sous Windows (elles sont équivalentes sous Unix), nous allons entrer dans le cœur du sujet, à savoir le modèle client-serveur. Pour illustrer cela, vous allez tester le fonctionnement d'une petite application (en Java) permettant à deux machines distantes (le client et le serveur) de communiquer. Nous allons procéder en plusieurs étapes, pas à pas.

2.1 Tout en local

Bien que le client et le serveur soient rarement sur la même machine, nous allons d'abord procéder de la sorte, car c'est le plus simple, et vous n'avez alors besoin d'aucune autre ressource matérielle que votre ordinateur. C'est d'ailleurs souvent la première étape, pour tester une application de ce type. Nous verrons par la suite d'autres raisons qui nous poussent à agir ainsi.

1. Récupération des fichiers nécessaires

Avec votre navigateur web préféré, tapez cette adresse : `http://www.loria.fr/~burel/cours/Serveur.jar` et enregistrez le fichier sur le bureau. Faites de même avec les fichiers situés aux adresses

– `http://www.loria.fr/~burel/cours/Client.jar`
– `http://www.loria.fr/~burel/cours/client.conf`
– `http://www.loria.fr/~burel/cours/serveur.conf`

Les fichiers en `.jar` sont des archives Java ; ils contiennent des programmes Java compilés.

2. Quelques explications sur le code

Les différents fichiers vont vous être commentés (du moins les parties intéressantes).

3. Configuration

Les fichiers `.conf` contiennent des informations de configuration utilisées dans les programmes. Le fichier `client.conf` contient les informations pour le client.

– Cliquez avec le bouton droit dessus, et choisissez Éditer avec Notepad++.
– Laissez la valeur `port` à 9000, mais indiquez le nom de votre machine (ou votre adresse IP) à la ligne `serveur`. Ceci permet d'indiquer que le serveur (la machine) sur lequel le client se connectera.
– Vous pouvez également ouvrir le fichier `serveur.conf`, mais il n'y a rien à modifier.

4. Exécution

On peut maintenant lancer les programmes.

– Dans l'invite de commande, tapez `cd Bureau` pour vous placer dans le répertoire de votre bureau.
– Tapez maintenant `java -jar Serveur.jar` pour lancer le serveur.

Une fenêtre devrait s'ouvrir, vous informant que le pare-feu de Windows est en train de bloquer le programme Java que vous venez d'exécuter. Ceci signifie que les machines extérieures ne pourront pas se connecter au serveur que vous avez lancé. Si vous étiez administrateurs de la machine, il vous serait possible de modifier le pare-feu pour autoriser le programme en question à recevoir des requêtes de l'extérieur.

On peut maintenant lancer le client :

– Ouvrez un autre invite de commande, et placez vous dans le répertoire du bureau.
– Tapez la commande `java -jar Client.jar` pour lancer le client.
– Observez les résultats à la fois dans la fenêtre du client et dans celle du serveur.

Remarque : le client cherche à se connecter à un serveur, il faut donc que celui-ci soit lancé avant le client (il vaut mieux que le Quick soit ouvert si vous voulez être servis)

2.2 S'ouvrir au monde

Comme cette application fonctionne bien en local (tout sur la même machine), il faudrait maintenant la tester sur des machines distinctes. Toutefois, comme indiqué plus haut, le pare-feu nous en empêche.

Pour contourner cela, nous allons utiliser un port qui n'est pas bloqué par le pare-feu. S'il n'est pas bloqué, c'est qu'il est utilisé par d'autre programmes, qui du coup vont interférer avec notre serveur.

– Ouvrez le fichier `serveur.conf` avec Notepad++.
– Mettez la valeur 138 pour le port.
– Lancez le serveur avec `java -jar Serveur.jar` Remarquez que la fenêtre du pare-feu ne s'ouvre pas cette fois-ci.
– Attendez un peu. Au bout d'un certain temps, votre serveur devrait recevoir un message bizarre. Il s'agit d'une requête envoyée par un client qui ne pensait pas se connecter à votre serveur.
– Remarquez l'adresse du client qui a tenté de se connecter, il ne s'agit a priori pas de celle de votre machine. Votre serveur est donc bien visible de l'extérieur.

2.3 Serveur et client*s*

Jusqu'à maintenant, nous avons testé de manière simple la communication client-serveur sur une seule machine. Comme nous avons vu en cours, le modèle client-serveur est bien souvent un modèle client*s*-serveur : le serveur traite simultanément les requêtes de plusieurs clients. Par exemple, un serveur de mail est capable de répondre à plusieurs utilisateurs. En effet, pendant la durée d'exécution d'une requête d'un utilisateur, il est plus que probable qu'un autre utilisateur va formuler lui aussi une requête. Ce dernier exercice va montrer qu'il est possible d'avoir un serveur et plusieurs clients en simultané. Pour ce faire, nous allons raffiner un peu le code de l'exercice précédent.

1. Avec votre navigateur web, récupérez les fichiers
   – `http://www.loria.fr/~burel/cours/Client_2.jar`
   – `http://www.loria.fr/~burel/cours/Serveur_2.jar`
2. Modifiez `serveur.conf` pour remettre la valeur 9000 pour le port.
3. Ouvrez trois invites de commande et placez vous dans tous dans le répertoire du bureau.
4. Dans un des invites, lancez le serveur avec la commande `java -jar Serveur_2.jar`
5. Dans chacun des autres invites, lancez un client avec la commande `java -jar Client_2.jar`
6. Entrez un nombre dans un des clients et tout de suite après un nombre dans l'autre.

Vous pouvez voir que cela se fait, du côté serveur, client par client. Dans la pratique, ce n'est pas la meilleure solution, car le traitement d'un client par le serveur empêche aux autres clients d'être servis. Si le traitement est long, cela peut-être embêtant (d'autant plus si la tâche du client 2 est plus urgente que la tâche du client 1). La solution est donc de disposer de plusieurs « copies » du serveur.

Dans la pratique :

1. Le client lance sa requête un port n du serveur.
2. Un processus du serveur accepte cette communication, et renvoie au client un numéro de port q sur lequel la communication va vraiment s'effectuer.
3. Le client relance alors la communication sur ce port q.

Cela permet de gérer autant de connexions simultanées que nécessaire (enfin, selon les capacités du serveur), sans bloquer les ressources. C'est sur ce mode que fonctionne le serveur web Apache. Il existe également d'autres solutions, mais nous ne les traiterons pas ici.