Objectifs : Réaliser vos premières applications distribuées en utilisant le protocole UDP et comprendre le fonctionnement d'UDP en comparaison avec TCP.

Notations et rappel :
Le protocole de transport **UDP** permet de réaliser des communications en mode non connecté. Un message envoyé en **UDP** est transféré/acheminé en un seul paquet **UDP** et est indépendant de tout autre paquet. Enfin, **UDP** ne gère ni la duplication, ni la remise dans l'ordre des paquets à leur réception.

1 Echanges de messages et gestion de leurs limites

Etape 1

Écrire deux programmes :

- un programme émetteur qui envoie une chaîne de caractères à un programme récepteur dont l'adresse et le numéro de port sont des paramètres de l'émetteur. La chaine de caractères est à saisir au clavier.
- un programme récepteur qui reçoit une chaîne de caractères et l'affiche à l'écran.

Exécuter les deux programmes sur deux machines différentes (en utilisant la commande ssh) et assurez vous du bon fonctionnement de votre application avant de passer à la suite.

Etape 2 (Pendant le TP)

Modifier vos programmes pour que :

- le programme émetteur envoie successivement deux chaînes de caractères saisies au clavier (il fait donc 2 envois).
- le programme récepteur demande à recevoir (une seule fois) une suite d'octets dont la taille est passée en paramètre du programme, et affiche le nombre d'octets effectivement reçus.

Ensuite :

1. Exécuter les deux programmes en s'assurant que les deux chaines de caractères sont envoyées après que la socket du récepteur soit prête à recevoir des messages et avant de se mettre en réception. Qu'observez vous lorsque le nombre d'octets passés en paramètre du récepteur est inférieur, égale ou supérieur à la longueur de la première ou seconde chaine de caractères ? Que se passe-t-il lorsqu'il est supérieur à la somme des longueurs des deux chaines ?
2. Qu'aurez vous obtenu en résultat si le protocole utilisé était TCP (cf. exercice du TP précédent) ?

Etape 3 (Pendant le TP)

1. Modifier vos programmes pour pouvoir envoyer/recevoir des messages dont la taille peut varier jusqu'à dépassement de la taille du buffer d'envoi. A ce moment, que se passe-t-il du coté de l'émetteur ? Du récepteur ?
2. Quel aurait été le comportement en utilisant le protocole TCP ?

2 Perte de paquets (Pendant le TP)

Modifier l'application précédente (pensez à faire des sauvegardes avant chaque modification demandée) pour que :

- l'émetteur envoie en boucle une chaîne de caractères. Après chaque envoi, le programme doit afficher le nombre d'octets total effectivement envoyés depuis le début. Le nombre d'itérations à effectuer, soit N, est à passer en paramètre du programme.
- le récepteur boucle indéfiniment sur 1) la réception d'une chaîne de caractères (taille maximum doit être connue), 2) l'incrémentation d'une variable *sommeOctets* qui calcule le nombre d'octets total reçus depuis le début 3) l'affichage du résultat de cette incrémentation.

Ensuite :

1. Exécuter les deux programmes sur deux machines différentes, en testant avec des petites valeurs de N.
2. Faire des tests en augmentant progressivement N. Le récepteur reçoit-il tous les messages ?
3. Mettez en place un scénario permettant de mettre en évidence la perte de paquets.
4. Pour mieux mettre en évidence la perte de paquets sans avoir à mettre en oeuvre les scénarios de la question 3, ainsi qu'une arrivée dans le désordre, vous êtes vivement encouragés à exécuter vos programmes sur des machines reliées par Internet (cf. dernier exercice du TP précédent).