

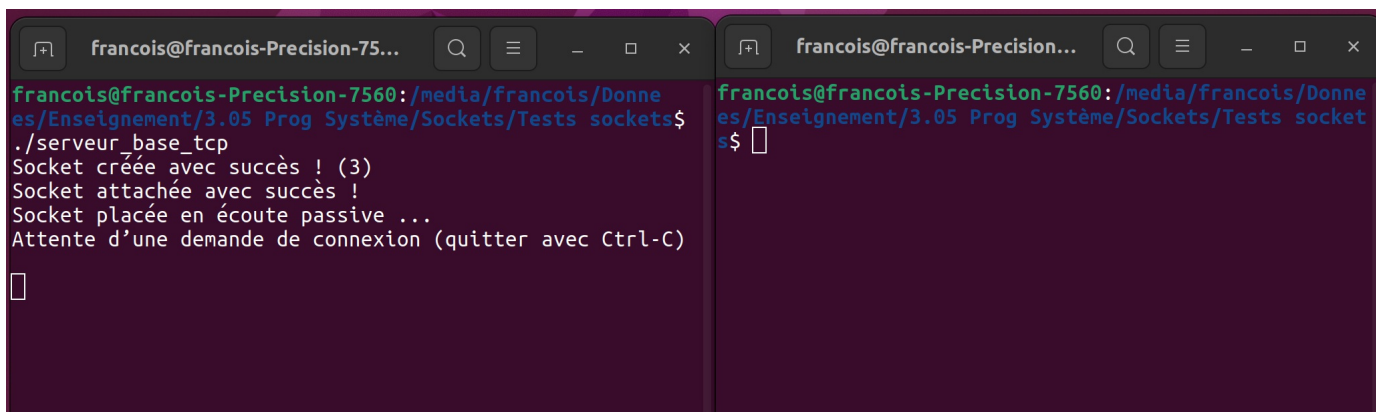
TP Socket  
Année 2022- 2023

Le but de ce TP est d'écrire une application qui permet à deux processus d'échanger des données en utilisant le mécanisme des sockets en mode TCP/IP. L'un des processus sera donc le serveur, l'autre le client. L'heure ou la date seront les données échangées entre ces deux processus.

Tout comme les TP de R3.05, vous ferez ce TP sous Linux et en langage C.

Avant de commencer :

1. Récupérez les 2 fichiers client\_base\_tcp.c et serveur\_base\_tcp.c
2. Récupérez les 3 fiches sur le langage C
3. Ouvrir votre dossier contenant les 2 fichiers client et serveur avec VSC et compilez uniquement sans exécuter ces 2 fichiers (si problème, revoir les fiches installation et utilisation du langage C avec VSC)
4. Lancer 2 terminaux côte à côte et allez pour chacun dans le dossier de vos fichiers de programme

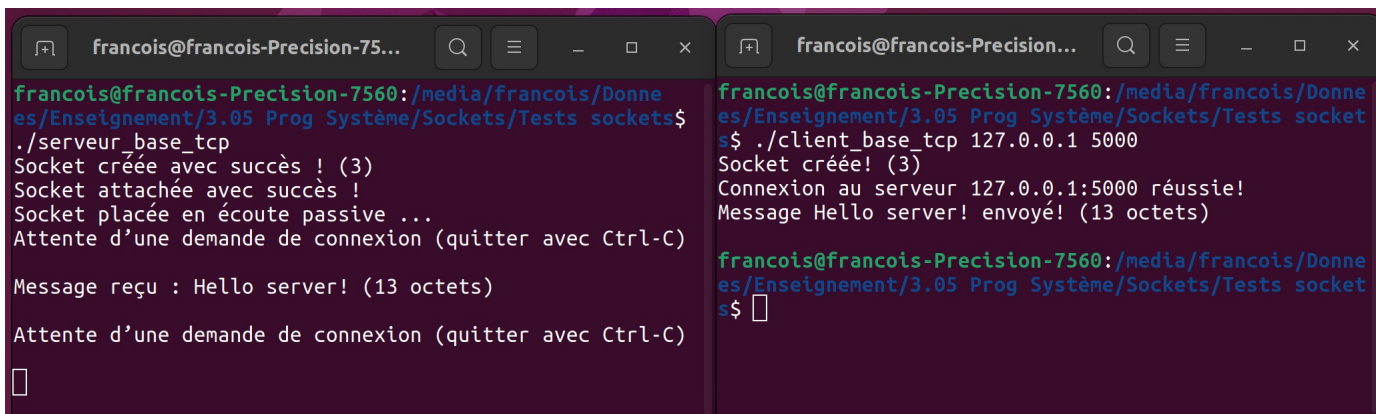


```
francois@francois-Precision-7560:/media/francois/Donnees/Enseignement/3.05 Prog Système/Sockets/Tests sockets$ ./serveur_base_tcp
Socket créée avec succès ! (3)
Socket attachée avec succès !
Socket placée en écoute passive ...
Attente d'une demande de connexion (quitter avec Ctrl-C)

francois@francois-Precision-7560:/media/francois/Donnees/Enseignement/3.05 Prog Système/Sockets/Tests sockets$
```

5. Exécutez d'abord le serveur dans le premier terminal : `./serveur_base_tcp`

6. Exécutez ensuite le client avec les paramètres suivants : `./client_base_tcp 127.0.0.1 5000`



```
francois@francois-Precision-7560:/media/francois/Donnees/Enseignement/3.05 Prog Système/Sockets/Tests sockets$ ./serveur_base_tcp
Socket créée avec succès ! (3)
Socket attachée avec succès !
Socket placée en écoute passive ...
Attente d'une demande de connexion (quitter avec Ctrl-C)

Message reçu : Hello server! (13 octets)
Attente d'une demande de connexion (quitter avec Ctrl-C)

francois@francois-Precision-7560:/media/francois/Donnees/Enseignement/3.05 Prog Système/Sockets/Tests sockets$ ./client_base_tcp 127.0.0.1 5000
Socket créée! (3)
Connexion au serveur 127.0.0.1:5000 réussie!
Message Hello server! envoyé! (13 octets)

francois@francois-Precision-7560:/media/francois/Donnees/Enseignement/3.05 Prog Système/Sockets/Tests sockets$
```

Pour démarrer ce tp, dupliquez les 2 fichiers exemples dans 2 nouveaux fichiers client\_date\_tcp.c et serveur\_date\_tcp.c. Vous pouvez compiler et tester vos 2 nouveaux fichiers.

En vous appuyant sur ce code existant, écrire les deux programmes respectant les étapes suivantes et l'organigramme de la page suivante :

Client	Serveur
1. Création de la socket de dialogue par le client	1. Création de la socket d'écoute par le serveur
2. Attachement de l'adresse locale à la socket	2. Attachement de l'adresse locale à la socket
3. Demande effectuée par le client au serveur d'une connexion	3. Déclaration du nombre maximum de connexions autorisées
4. Envoi de la demande : heure ou date	4. Attente du serveur d'une demande de connexion
5. Réception de la réponse du serveur	5. Réception de la demande du client
6. Fermeture de la socket de dialogue par le client	6. Traitement de la demande du client
	7. Envoi de la réponse au client
	8. Fermeture de la socket de dialogue par le serveur
	9. Fermeture de la socket d'écoute par le serveur

Pour récupérer l'heure et la date, on utilisera les deux fonctions suivantes, dont le code est donné en annexe :

1. void lire\_heure(char\* heure) : cette fonction permet de récupérer l'heure
2. void lire\_date(char\* date) : cette fonction permet de récupérer la date

Remarque : la fonction *popen* utilisée dans ces 2 fonctions permet d'ouvrir un pipe et d'exécuter une commande shell dont la sortie pourra être lue au travers du pipe si l'option « r » est indiquée. Au contraire le pipe pourrait servir à envoyer des données en entrée de la commande si « w » est spécifié. Voir les manpages de *popen* pour plus d'information.

#### ANNEXE :

<pre>void lire_heure(char* heure){     FILE *fpipe;      fpipe = popen("date '+%X'", "r");     if (fpipe == NULL){         perror("popen" );         exit(-1);     }     fgets(heure, LG_MESSAGE, fpipe);     pclose(fpipe); }</pre>	<pre>void lire_date(char* date){     FILE *fpipe;      fpipe= popen("date '+%A %d %B %Y'", "r");     if (fpipe == NULL){         perror("popen" );         exit(-1);     }     fgets(date, LG_MESSAGE, fpipe);     pclose(fpipe); }</pre>
--	---

