Thème: Realisation d'un moniteur système

L'objectif du tp

Le but de ce tp consiste à realiser un moniteur système qui ressemble à htop, permettant de récupérer quelques métriques de la machine (les pids des processus en cour d'exécution, occupation mémoire des processus, utilisation de la cpu, etc),tout on exploitant les notion vu en cour et en tp telles que l'utilisation des bibliothèque statiques et dynamiques et les I/O pour sauvegarder les informations colléctées par le capteur (joue le rôle d'un serveur) dans un fichier et les envoyer à l'interface (joue le rôle d'un client) qui sert à afficher ses informations selon la demande. Cet échange peut se faire grâce aux sockets. Notre travail est organisé en trois partie:

- La première partie: un serveur gère des demande d'un seule client
- La deuxième partie: un serveur gère des demandes de plusieurs clients situés sur des machines distinctes, ce concept est dit "multi-client". Ce qui nous permet d'ajouter un nouveau contexte d'exécution qui est le multithreading (parallélisme).
- La troisième partie : un chat entre un serveur et plusieurs clients situés dans différentes machines, ce concept est dit "client-avancé".

Les étapes du tp

1. l'architecture

l'architecture qu'on a adapté pour le tp ressemble à l'architecture Client/Serveur:

Capteur (Serveur):

permet de collecter les métriques de la machine.

Interface (Client):

permet d'afficher les métriques de la machine captéés par le capteur.

Réseau (socket):

permet au capteur et à l'interface de se connecter et de s'echanger des données.

Protocole:

le protocole utilisé est TCP qui est un protocole en mode connecté

2. Organisation du projet

le tp est organisé en deux partie:

2.1 partie1: un seul (capteur/client)

- Makefile: pour compiler le projet
  - build
    - include: détermine les fichiers header(.h)
    - lib: détermine les bibliothèques
  - src: détermine les fichiers source (.c)

2.2 partie2:(capteur/multi-clients)

- Makefile: pour compiler le projet
  - build-partie1
    - include: détermine les fichiers header(.h)
    - lib: détermine les bibliothèques
  - src-partie1: détermine les fichiers source (.c)
  - build-partie2
    - include: détermine les fichiers header(.h)
    - lib: détermine les bibliothèques
  - src-partie2: détermine les fichiers source (.c)

2.3 partie3:(chat serveur/clients)

- Makefile: pour compiler le projet
  - build
    - include: détermine les fichiers header(.h)
    - lib: détermine les bibliothèques
  - src: détermine les fichiers source (.c)

3. Compilation et Éxécution:

Pour compiler et exécuter le projet, on suit les étapes suivantes:

3.1 la partie1:

- D’abord on doit se situer dans le répértoire AISE/partie1:
- Ensuite on compile et on exécute,en tapant les commandes suivantes:

> on ouvre deux terminals,un pour executer le capteur et l’autre pour l’interface.
> make
> Capteur --> ./build/program_capteur_static
> Interface --> ./build/program_interface_static

- Enfin, après la compilation deux fichiers exécutable seront générés:

> program_capteur_static
> program_interface_static

3.2 la partie2:

- D’abord on doit se situer dans le répértoire AISE/partie2.
- Ensuite on compile et on exécute,en tapant les commandes suivantes:

> make
> Capteur --> LD_LIBRARY_PATH=./build-partie2/lib ./build-partie2/program_capteur2_dyn
> Interface --> LD_LIBRARY_PATH=./build-partie2/lib ./build-partie2/program_interface2_dyn

- Enfin, après la compilation deux fichiers exécutable seront générés:

> program_capteur2_dyn
> program_interface2_dyn

3.3 la partie3:

- D’abord on doit se situer dans le répértoire AISE/partie3.
- Ensuite on compile et on exécute,en tapant les commandes suivantes:

> make
> Serveur --> ./build/program_serveur
> Client --> LD_LIBRARY_PATH=./build/lib ./build/program_client

- Enfin, après la compilation deux fichiers exécutable seront générés:

> program_serveur
> program_client

4-Scenarios de l’exécution

4.1 partie1:

Dans cette partie,un capteur qui joue le role d'un serveur qui a pour objectif de collecter l'ensemble des informations concernant la machine tels que l'occupation mémoire de chaque processus, l'utilisation de La cpu, etc.Une interface qui joue le role d'un client, son objectif est d'afficher les informations captées par le capteur.Globalement, on établit la connexion entre le capteur et l'interface, ensuite l'interface demande au capteur les informations en tapant htop et ce dernier lui renvoie les informations voulues.

4.2 partie2:

On suit le même Scenarios que la partie1 sauf qu'ici plusieurs interfaces(multi-client) sur des machines distantes demandent des informations à un seul serveur.

4.3 partie3:

Dans cette troisième partie, on a essayé d'implémenter un serveur et des clients de messagerie (client avancé), Le serveur doit répondre à chaque message envoyé par le client de façon continue.