TP SDE
BONNIN-AOUIBATE

Présentation du TP :

Le but de ce TP est de coder une phrase donnée par l'utilisateur, de la transmettre à un ordinateur destinataire puis enfin la décodée. Pour ce faire, nous allons utiliser plusieurs processus que nous allons détaillés dans la suite de ce compte rendu.

-Saisie prend en compte les caractères rentrés par l'utilisateur, les transmets à Code via un tube nommé. Chaque phrase est identifiée par un nombre choisis aléatoirement.

-Choix choisit aléatoirement une lettre qu'il transmet à Code et envoi le numéro de la phrase ainsi que la lettre à la boite aux lettres.

-Code récupère la lettre qui va servir à décaler toutes les autres afin de coder la phrase initiale. Il va ensuite envoyer la phrase codée dans la boite aux lettres avec le numéro de la phrase.

-Réception récupère la phrase codée, créer une mémoire partagée avec Décode

-Décode prends la lettre stockée dans la boite aux lettres et inverse le décalage afin de retrouver la phrase originale. Enfin Décode stock cette phrase dans un fichier "Secret Defense".

Le processus Saisie :

- Le processus saisie récupère les phrases saisies par l'utilisateur et les inscrit dans un tube nommé qui le relie avec code,on utilise le signal SIGINT qui sert à détruire le processus.

- Les processus choix et code se transférent les données par signaux tandis que les processus Saisie et Code communiquent par le biais d'un tube nommé.

les processus code et choix s'attachent a la même file de message ; code écrit les phrases codés tandis que choix choisit une lettre aléatoirement donc aura besoin de deux structures groupe_donnée et groupe_message ,de ce fait on pourra donc envoyer des messages ayant des types différents dans une unique boite aux lettres et réception saura tout de même les distinguer et faire les opérations souhaitées .

Le processus Code et Choix :

Le processus code récupère la lettre du tube nommé , donne le numéro de la phrase à l'aide de la sémaphore , envoie SIGUSR2 au processus choix et transmet la lettre a la boite aux lettres. Le processus Code récupère les lettres via le tube et code avec la valeur en ASCII de la lettre aléatoire transmis par le processus Choix.

On décale ensuite chacune des lettres avec la fonction coder() avant de transférer la phrase codée dans la boite aux lettres avec la valeur du sémaphore Sentence.

- le processus choix choisit une lettre aléatoirement , envoie le signal SIGUSR2 vers la fonction writefile() et s'attache a la semaphore
-

Le processus Réception :

On utilise le signal SIGINT afin de détruire proprement tous les IPC et stoppé correctement l'application.

On se connecte ensuite à la boite aux lettres ce qui va permettre de récupérer la phrase précédemment codée. Ensuite, on créer la mémoire partagée avec Décode puis on s'y attache.
On récupère le numéro de la phrase ainsi que les lettres afin de s'assurer que les lettres et le code correspondent, et enfin on met ces lettres et le numéro de la phrase dans un buffer.

Le processus Décode :

On se connecte à la boite aux lettre ainsi qu'a la mémoire partagée, afin de récupérer de la première le numéro de la phrase ainsi que la lettre de décalage.
On vérifie que la lettre provenant de la boite aux lettres et de Réception concordent et ensuite, on décale dans le sens inverse les lettre afin d'obtenir la phrase décodée.
Enfin on écrit cette phrase dans un fichier nommé "Secret Defense"

Structures :

```
typedef struc groupe_donnee{
        char sentence[64];
        int sentenceid;
        int lettre_code
} DATA;

typdef struct groupe_message{
        int cat;
        DATA données;
}MESSAGE;

typedef struct memorie_partagee{
        char sentence[64]
        int processeddata;
        int sentenceid;
}SHARE_MEM;
```

Clés d'identification des IPC :

```
#define KEY_SEMAPHORE_ID 20
#define KEY_SEMAPHORE_LETTRE_CODE 30
#define KEY_MESSAGE 40
#define KEY_SEMAPHORE_SHARE_MEM 50
#define KEY_SHARE_MEM 60
```

Saisie :

Rediriger SIGINT vers la fonction quittertube()
Créer tube
TANT QUE : pas fin de phrase
    Scanner caractères
    Ecrire les lettres à Code par tube
FIN TANT QUE
Fermer tube

Code :

Créer letter box
Rediriger le signal SIGINT vers quitter()
Rediriger le signal SIGUSR2 vers coder()
Sémaphore Sentence
S'attacher à letter box
S'attacher au Sémaphore
GetPID(choix)
Se détacher du Sémaphore
TANT QUE : pas fin de phrase
    Récupérer lettre_phrase de tube
    S'attacher au Sémaphore
    Sentence=numéro phrase+1
    Récupérer lettre
    coder()
    Se détacher du Sémaphore
    Envoyer le signal SIGUSR2 au processus Choix
    sleep(1)
    Transmettre lettre_code à letter box
FIN TANT QUE

Choix :

Rediriger SIGINT vers la fonction quitter ()
Rediriger SIGUSR2 vers writefile()
S'attacher à letter box
S'attacher au Sémaphore
Choisir une lettre aléatoirement
Down(Sentence)
numéro_phrase=Sentence
Transférer lettre à Code
Transférer numéro_phrase && lettre à letter box

Fonction quittertube()
    Fermer tube
    End Saisie

Fonction quitter()
    Essayer de détruire la file de message
    End choix
End quitter

Fonction tirerLettre()
    Tirer un nombre au hasard entre “97” et “122”
    Retourner le caractère ASCII correspondant
End tirerLettre

Récéption :

Rediriger le signal SIGINT vers quitterreception()
S'attacher à letter box
Créer mémoire partagée
S'attacher à la mémoire partagée en écriture
Récupérer numéro_phrase
verif=numéro_phrase
TANT QUE : pas fin de phrase
    Récupérer lettre_code
    Mettre lettre_code dans buffer

FIN TANT QUE

Fonction quitterreception()

Se détacher des IPC
Fermer mémoire partagée
End Récéption

Décode :

S'attacher à letter box
S'attacher à la mémoire partagée en lecture
Récupérer numéro phrase et lettre
TANT QUE : pas fin de phrase && numéro de phrase==vérif
    Accéder à la mémoire partagée
    lancer decoder()
    Mettre lettre_décode dans fichier "SecretDefense"
FIN TANT QUE
Se détacher de tous les IPC

Fonction coder() :

TANT QUE : pas fin de phrase
    lettre_code=ASCII(lettre)+97
    Renvoyer lettre_code
FIN TANT QUE
End coder()

Fonction Décoder :

TANT QUE : pas fin de phrase
    lettre_decode=lettre_code-97
    Renvoyer lettre_decode
FIN TANT QUE
End Décoder()

Lâcatre :