TP4

November 2020

1 Objectifs

Durant ce TP, vous allez créer un verrou d'enregistrement interactif. Les objectifs pédagogiques sont de:

- poser un verrou et l'enlever sur une partie d'un fichier;
- comprendre les différents types de verrous

2 Description

Dans le système Linux, il existe deux fonctions de verrouillage: flock() et fcntl (). Le premier ajoute principalement des verrous facultatif. Ce dernier peut ajouter un verrou facultatif (par défaut) ou obligatoire. En outre, il peut verrouiller un fichier ou une partie du fichier.

Lorsque fcntl () est utilisé pour verrouiller une partie d'un fichier, il est généralement appelé verrouillage d'enregistrement. Ceci peut être utilisé pour synchroniser deux processus visitant la même partie d'un fichier.

Dans le TP, vous allez verrouiller / déverrouiller une partie d'un fichier de manière interactive. Les utilisateurs peuvent verrouiller et déverrouiller une partie spécifique du fichier, obtenir les informations de verrouillage, choisir le type de verrouillage et définir le décalage pour le verrouillage.

Voici un exemple du programme:

```
$ ./your-programme bla.txt
Enter ? for help
PID=258> ?

    Format: cmd l_type start length [whence(optional)]
    'cmd' ---  'g' (F_GETLK), 's' (F_SETLK), or 'w' (F_SETLKW)
    'l_type' --- 'r' (F_RDLCK), 'w' (F_WRLCK), or 'u' (F_UNLCK)
    'start' --- lock starting offset
    'length' --- number of bytes to lock
    'whence' --- 's' (SEEK_SET, default), 'c' (SEEK_CUR), or 'e' (SEEK_END)

PID=258>
```

Vous pouvez utiliser un format différent pour votre programme. Mais tous les paramètres énumérés ci-dessus devraient pouvoir être manipulés. Voici un exemple de sortie du programme:

```
$ ./your-programme bla.txt
```

```
Enter ? for help
PID=258> s w 0 5
[PID=258] got lock

PID=270> g r 0 5
[PID=270] Denied by WRITE lock on 0:5 (held by PID 258)
```

3 flock

```
#include <sys/file.h>
int flock(int fd, int operation);
```

Flock est un verrou facultatif et n'est pas obligatoire. Lorsqu'un processus verrouille un fichier à l'aide de flock, un autre processus peut manipuler directement le fichier en cours de verrouillage et modifier les données qu'il contient, car flock est uniquement utilisé pour détecter si le fichier est verrouillé ou non. Dans le cas où un autre processus écrit des données, le noyau ne bloque pas l'opération d'écriture du processus, qui correspond à la stratégie de traitement du noyau du verrou proposé.

Flock() effectuera diverses actions de verrouillage ou de déverrouillage sur le fichier désigné par le paramètre fd de la manière spécifiée par l'opération paramètre. Cette fonction ne peut verrouiller que le fichier entier et ne peut pas verrouiller une zone du fichier.

Le verrou placé par flock () est limité de plusieurs manières. 1. Verrouille uniquement le fichier entier, ce verrouillage limite la simultanéité des processus de collaboration grossièrement. 2. Seuls les verrous facultatif peuvent être placés via flock (). 3. De nombreuses implémentations NFS ne reconnaissent pas les verrous placés par flock ().

4 fcntl

```
#include <unistd.h>
#include <fcntl.h>
int fcntl(int fd, int cmd, ... /*arg */);
```

Il existe deux types de fcntl: le verrous facultatif et le verrou obligatoire. fcntl est par défaut un verrou facultatif, le verrou obligatoire est non standard POSIX.

Des précautions doivent être prises lors de l'utilisation de verrous sur des fichiers avec fcntl pour les processus d'I/O, telles que la façon dont le processus gère le verrouillage avant de commencer des opérations d'I/O et quoi faire avant de déverrouiller le fichier. Si le fichier est ouvert avant que le verrou ne soit défini ou s'il est fermé après la lecture du verrou, un autre processus peut accéder au fichier en une fraction de seconde entre les opérations de verrouillage / déverrouillage et d'ouverture / fermeture. Lorsqu'un processus verrouille un fichier, qu'il libère ou non le verrou ajouté, le noyau libère automatiquement le verrou suggéré attaché au fichier tant que le fichier est fermé (c'est également la plus grande différence entre le facultatif et le verrou obligatoire).

4.1 Utilisation

La forme générale de l'appel fcntl() utilisé pour créer ou supprimer un verrou de fichier est la suivante:

```
struct flock lock;
fcntl(fd, cmd, &lock);
```

```
struct flock {
...
short l_type; /* Type of lock: F_RDLCK,F_WRLCK, F_UNLCK */
short l_whence; /* How to interpret l_start: SEEK_SET, SEEK_CUR, SEEK_END */
off_t l_start; /* Starting offset for lock */
off_t l_len; /* Number of bytes to lock */
pid_t l_pid; /* PID of process blocking our lock (set by F_GETLK and F_OFD_GETLK)
*/
...
};
```

Fcntl() a généralement trois paramètres, F_SETLK , F_UNLCK et F_GETLK pour verrouiller, libérer et tester les verrous d'enregistrement, respectivement. Le verrouillage des enregistrements est un verrou sur une partie du fichier plutôt que sur l'ensemble du fichier, ce qui permet au processus de mieux coopérer pour partager les ressources du fichier. Fcntl() peut être utilisé pour les verrous en lecture et en écriture. Les verrous en lecture sont également appelés verrous partagés, car plusieurs processus coopérants peuvent créer des verrous en lecture dans la même partie d'un fichier. Pour le verrou en écriture, un seul verrou peut être posé à la fois (verrou exclusif, exclusive).

5 Questions

- Que se passera-t-il si nous déverrouillons un fichier (ou une partie du fichier) qui n'est pas verrouillé?
- Que se passera-t-il si nous mettons un nouveau verrou sur une section déjà verrouillée? Le type de verrou changera-t-il le résultat? Expliquer dans la situation avec le même processus et avec 2 processus différents.