【Linux Programming】Day22

Tags	
Date	@June 20, 2022
Summary	Create Lock Files and Regional Lock Use flock

【Ch7】Data Management

7.1 File Locking

Linux has several features for file locking. The simplest method is a technique to create lock files in an atomic way, so that nothing else can happen while the lock is being created.

The second method is more advanced: it enables programs to lock parts of a file for exclusive access.

7.1.1 Creating Lock Files

To create a file to use as a lock indicator, we can use the `open` system call defined in `fcntl.h` with the `O_CREAT` and `O_EXCL` flags set. This guarantees that `open` fails if the specified file already exists.

On Linux, error 17 refers to `EEXIST`, an error used to indicate that a file already exists.

```
#define EEXIST 17
```

7.1.2 Locking Regions

We can lock only part of the file by using the `fcntl` system call.

```
#include <fcntl.h>
```

```
int fcntl(int filedes, int command, ...)
```

`fcntl` operates on open file descriptors and, depending on the `command` parameter, can perform different tasks. The three command options are:

- F_GETLK
- F_SETLK
- F_SETLKW

When we use these the third argument must be a pointer to a `struct flock`.

```
int fcntl(int fildes, int command, struct flock *flock_structure);
```

It contains at least the following members:

```
struct flock {
  short l_type;
  short l_whence;
  off_t l_start;
  off_t l_len;
  pid_t l_pid;
};
```

The `l_type` member takes one of several values:

Value	Description
F_RDLCK	A shared (or "read") lock. Many different processes can have a shared lock on the same (or overlapping) regions of the file. If any process has a shared lock, then no process will be able to get an exclusive lock on that region. In order to obtain a shared lock, the file must have been opened with read or read/write access.
F_UNLCK	Unlock; used for clearing locks
F_WRLCK	An exclusive (or "write") lock. Only a single process may have an exclusive lock on any particular region of a file. Once a process has such a lock, no other process will be able to get any sort of lock on the region. To obtain an exclusive lock, the file must have been opened with write or read/write access.

The `l_whence`, `l_start`, and `l_len` members define a region-a contiguous set of bytes-in a file.

The `l_whence` must be one of `SEEK_SET`, `SEEK_CUR`, `SEEK_END`.(from `unistd.h`) `l_whence` defines the offset to which `l_start`, the first byte in the region, is relative. Normally, this would be `SEEK_SET`, so `l_start` is counted from the beginning of the file.

The `l_len` parameter defines the number of bytes in the region.

The `l_pid` parameter is used for reporting the process holding a lock.

7.1.3 The F_GETLK Command

The `F_GETLK` command gets locking information about the file.

Value	Description
F_RDLCK	A shared (or "read") lock. Many different processes can have a shared lock on the same (or overlapping) regions of the file. If any process has a shared lock, then no process will be able to get an exclusive lock on that region. In order to obtain a shared lock, the file must have been opened with read or read/write access.
F_UNLCK	Unlock; used for clearing locks
F_WRLCK	An exclusive (or "write") lock. Only a single process may have an exclusive lock on any particular region of a file. Once a process has such a lock, no other process will be able to get any sort of lock on the region. To obtain an exclusive lock, the file must have been opened with write or read/write access.

A process may use the `F_GETLK` call to determine the current state of locks on a region of file.

It should set up the flock structure to indicate the type of lock it may require and define the region it's interested in.

The `fcntl` call returns a value other than -1 if successful. It the file already has locks that would prevent a lock request from succeeding, it overwrites the flock structure with the relevant information. If the lock will succeed, the flock structure is unchanged.

If the call to `F_GETLK` is successful, the caller must check if the contents of the flock structure is modified. Since the `l_pid` value is set to the locking process(if one was found), this is a convenient field to check to determine whether the flock structure has been changed.

7.1.4 The F_SETLK Command

This command attempts to lock or unlock part of the file referenced by `filedes`.

Value	Description
`l_type`	One of the following: `F_RDLCK` for a read-only, or shared, lock `F_WRLCK` for an exclusive or write lock `F_UNLCK` to unlock a region
`l_pid`	Unused

`F_SETLK` returns -1 when it fails and a value other than -1 when the setting is successful.

7.1.5 The F_SETLKW Command

The `F_SETLKW` command is the same as `F_SETLK` command except that if it cannot obtain the lock,