File Locking

Release 0.0.1

Gene C

Nov 05, 2023

CONTENTS

NOTES ON LOCKING

1.1 Linux: C Implementations

File Locking has linux kernel support via standard C library using fcntl(). The mechanism uses 'struct flock' as the communication mechanism with (posix) fcntl().

The standard locking mechanism uses F_SETLK. This lock lives at the process level and is the original locking in linux. Around 2015 or so 'open file desription' locks came to be via F_OFD_SETLK (See[1] and[2]). These are at the open file level. So while F_SETLK is not passed to child processes, OFD locks are. And OFD locks remain attached to the open file handle. This can be enormously useful and also surprising.

Posix locks are attached to (inode, pid) pair which means they work at the process level. If thread opens same file, even tho has different file handle, when that thread closes the fd, the lock is released for all threads in same process.

OFD locking was introduced to deal with this. Locking is attached to the "open file" and not the PID.

linux also provides "lockf" which is a wrapper around fcntl() locks - should only be used in simple cases and will interact with "normal" fcntl() locks - caveat emptor.

Summary:

- Posix Lock: fcntl() - F_SETLK

- OFD Lock: fcntl() - F_OFD_SETLK

- lockf - bah

NB. The flock struct contains l_pid - this MUST be 0 for OFD locks and PID for posix locks.

Its also worth noting that locking can be file system dependent. In particular NFS should probably be avoided. Since my dominant use case is single machine, multiple process, I use /tmp which is a TMPFS file system and works well.

1.2 Python

Python provides for same locking mechanisms - I recommend only 1 way for file locks in python. Python library provides for:

- fcntl.fcntl => do not use

As with C there is support for F_SETLK and F_OFD_SETLK. While these work fine, they require using 'struct' module to 'pack' and 'unpack' the C flock struct. To make this work the caller must provide the sizes (coded with letters as per the python struct module) of each element being packed.

---

[1] File private locks https://lwn.net/Articles/586904/

[2] Open File Description https://lwn.net/Articles/640404/

The python 3.12 docs have examples[3], and while they may well work for a Sun workstation or similar, if you have one, the struct element sizes dont seem correct for X86_64.

I provide a little C-program to print out the correct byte sizes which you can then map to the python struct letter codes[4]

This approach is brittle - its one thing when you are coding with your own C structures, its another entirely when using system ones - these sizes should be compiled into python - while these routines work I strongly recommend not using them for this reason.

- fcntl.lockf => do not use

Wrapper around fcntl() - in spite of name this is NOT C lockf() function.

- fcntl.flock => *use this one*

Wrapper around fcntl with OFD support. i.e. this lock is associated with open file descriptor. This is what I use and recommend.

---

[3] Python fcntl docs: https://docs.python.org/3/library/fcntl.html
[4] Python struct module: https://docs.python.org/3/library/struct.html

TWO

EXAMPLES

2.1 C-code

Sample code for F_SETLK and F_OFD_SETLK

- build

  make

Builds 2 programs - *flock_sizes* and *c_lock_test*.

To print size of struct flock elements which are Used to enssure we use the correct sizes in python fcntl.fcntl approach.

```
* ./flock_sizes
```

The test program demonstrates locking with and without OFD. To run the test progrm see the *Tests: c_lock_test* section below.

2.2 Tests: c_lock_test

To run locking tests, use 2 terminals. Run c_lock_test in both. The first will acquire lock while second will fail until first exits or is interrupted.

2.2.1 Test 1 : Using F_SETLK

```
./c_lock_test
```

2.2.2 Test 2 : Using F_OFD_SETLK

Repeat test but with argument to turn on OFD

```
./c_lock_test ofd
```

Test (1) and (2) both work.

2.3 Python : lock_fcntl

F_SETLK and F_OFD_SETLK tests in python. Run test in 2 terminals as above:

2.3.1 Test 3 : Using F_SETLK

```
./lock_fcntl.py
```

2.3.2 Test 4 : Using F_OFD_SETLK

```
./lock_fcntl.py ofd
```

Test (3) and (4) both work.

2.4 Python : lock_flock

This is what I am using. As above, run test in 2 terminals.

2.4.1 Test 5 :

```
./lock_fcntl.py
```

Test (5) works.

INDICES AND TABLES

- genindex
- modindex
- search