locks.txt
File Locking Release Notes

Andy Walker <andy@lysaker.kvaerner.no>

12 May 1997

1. What's New?

1.1 Broken Flock Emulation

The old flock(2) emulation in the kernel was swapped for proper BSD compatible flock(2) support in the 1.3.x series of kernels. With the release of the 2.1.x kernel series, support for the old emulation has been totally removed, so that we don't need to carry this baggage forever.

This should not cause problems for anybody, since everybody using a 2.1.x kernel should have updated their C library to a suitable version anyway (see the file "Documentation/Changes".)

1.2 Allow Mixed Locks Again

1.2.1 Typical Problems - Sendmail

Because sendmail was unable to use the old flock() emulation, many sendmail installations use fcntl() instead of flock(). This is true of Slackware 3.0 for example. This gave rise to some other subtle problems if sendmail was configured to rebuild the alias file. Sendmail tried to lock the aliases.dir file with fcntl() at the same time as the GDBM routines tried to lock this file with flock(). With pre 1.3.96 kernels this could result in deadlocks that, over time, or under a very heavy mail load, would eventually cause the kernel to lock solid with deadlocked processes.

1.2.2 The Solution

The solution I have chosen, after much experimentation and discussion, is to make flock() and fcntl() locks oblivious to each other. Both can exists, and neither will have any effect on the other.

I wanted the two lock styles to be cooperative, but there were so many race and deadlock conditions that the current solution was the only practical one. It puts us in the same position as, for example, SunOS 4.1.x and several other commercial Unices. The only OS's that support cooperative flock()/fcntl() are those that emulate flock() using fcntl(), with all the problems that implies.

1.3 Mandatory Locking As A Mount Option

Mandatory locking, as described in 'Documentation/filesystems/mandatory.txt'

was prior to this release a general configuration option that was valid for all mounted filesystems. This had a number of inherent dangers, not the least of which was the ability to freeze an NFS server by asking it to read a file for which a mandatory lock existed.

From this release of the kernel, mandatory locking can be turned on and off on a per-filesystem basis, using the mount options 'mand' and 'nomand'. The default is to disallow mandatory locking. The intention is that mandatory locking only be enabled on a local filesystem as the specific need arises.