

Re: Syslets, Threadlets, generic AIO support, v6

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-05/msg13685.html>

- *From:* Eric Dumazet <dada1@xxxxxxxxxxxxxx>
 - *Date:* Thu, 31 May 2007 12:41:29 +0200
-

On Thu, 31 May 2007 11:02:52 +0200
Ingo Molnar <mingo@xxxxxxx> wrote:

* Ingo Molnar <mingo@xxxxxxx> wrote:

it's both a flexibility and a speedup thing as well:

flexibility: for libraries to be able to open files and keep them open comes up regularly. For example currently glibc is quite wasteful in a number of common networking related functions (Ulrich, please correct me if i'm wrong), which could be optimized if glibc could just keep a netlink channel fd open and could poll() it for changes and cache the results if there are no changes (or something like that).

speedup: i suggested O_ANY 6 years ago as a speedup to Apache – non-linear fds are cheaper to allocate/map:

<http://www.mail-archive.com/linux-kernel@xxxxxxxxxxxxxxxxx/msg23820.html>

(i definitely remember having written code for that too, but i cannot find that in the archives. hm.) In theory we could avoid _all_ fd-bitmap overhead as well and use a per-process list/pool of struct file buffers plus a maximum-fd field as the 'non-linear fd allocator' (at the price of only deallocating them at process exit time).

to measure this i've written fd-scale-bench.c:

<http://redhat.com/~mingo/fd-scale-patches/fd-scale-bench.c>

which tests the (cache-hot or cache-cold) cost of open()-ing of two fds while there are N other fds already open: one is from the 'middle' of the range, one is from the end of it.

Lets check our current 'extreme high end' performance with 1 million fds. (which is not realistic right now b