

Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

# Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/performance/2006-05/msg00004.html>

- *From:* Kris Kennaway <[kris@xxxxxxxxxxxxxxxx](mailto:kris@xxxxxxxxxxxxxxxx)>
- *Date:* Sat, 6 May 2006 18:19:08 -0400

On Sat, May 06, 2006 at 03:16:48PM +0100, Robert Watson wrote:

Dear all,

Attached, please find a patch implementing more fine-grained locking for the POSIX local socket subsystem (UNIX domain socket subsystem).

Dear Sir,

Per your request, please find attached the results of my measurements using super-smack on a 12-cpu E4500.

supersmack queries/second with n worker threads:

norwatson = without your patch (but with some other local locking patches)

rwatson = also with your patch

x norwatson-4  
+ rwatson-4

```

+-----+
| x xx ++ |
|x xx xx x +++++ +++|
| | _AM_ | | _A_ | |
+-----+

```

N	Min	Max	Median	Avg	Stddev	
x	10	3067.92	3098.05	3086.945	3084.402	8.8815574
+	10	3245.06	3287.8	3270.52	3270.475	13.241953

Difference at 95.0% confidence  
186.073 +/- 10.5935  
6.03271% +/- 0.343455%  
(Student's t, pooled s = 11.2746)

x norwatson-6  
+ rwatson-6

+-----+

Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

```
| xx x + |
|x *xxxx + + + + + |
| |__A__| |____A__M____| |
+-----+
N Min Max Median Avg Stddev
x 10 3641.11 3693.89 3679.735 3677.083 14.648967
+ 10 3672.23 3896.32 3869.415 3845.071 66.826543
Difference at 95.0% confidence
167.988 +/- 45.4534
4.56851% +/- 1.23613%
(Student's t, pooled s = 48.3755)
```

i.e. in both cases there is a clear net gain in throughput with your patch.

Without your patch, 6 clients is the optimum client load on this 12-cpu machine. At higher loads performance drops, even though formally all CPUs are not saturated. This is due to rapidly diverging lock contention (see below).

```
x norwatson-8
+ rwatson-8
+-----+
| + |
| + + + x x |
| + + + + + x xxxxx x x |
| |____A__M____| |__A__| |
+-----+
N Min Max Median Avg Stddev
x 10 2601.46 2700.26 2650.52 2653.441 30.758034
+ 10 2240.86 2516.87 2496.085 2468.468 81.868576
Difference at 95.0% confidence
-184.973 +/- 58.1052
-6.97106% +/- 2.1898%
(Student's t, pooled s = 61.8406)
```

We see the drop in performance in both cases indicating that we are in the "overloaded" regime. The fact that your patch seems to give worse performance is puzzling at first sight.

Running mutex profiling (and only keeping the unp mutex entries and the 10 most contended for clarity) shows the following:

norwatson, 8 clients:

```
debug.mutex.prof.stats:
max total count avg cnt_hold cnt_lock name
5 40 9 4 0 3 kern/uipc_usrreq.c:170 (unp)
8 8 1 8 0 0 vm/uma_core.c:2101 (unpcb)
13 283 52 5 0 0 vm/uma_core.c:890 (unpcb)
14 1075 200 5 0 0 vm/uma_core.c:1885 (unpcb)
```

Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

```
4 52 18 2 4 6 kern/uipc_usrreq.c:577 (unp)
5 39 9 4 4 2 kern/uipc_usrreq.c:534 (unp)
5 35 11 3 6 6 kern/uipc_usrreq.c:974 (unp)
5 45 11 4 7 4 kern/uipc_usrreq.c:210 (unp)
171 1164 9 129 7 2 kern/uipc_usrreq.c:917 (unp)
14 78 20 3 11 2872481 kern/uipc_usrreq.c:709 (unp)
70 156 11 14 13 4 kern/uipc_usrreq.c:895 (unp)
43 581 20 29 24 6 kern/uipc_usrreq.c:239 (unp)
44 429 18 23 26 8 kern/uipc_usrreq.c:518 (unp)
55 491 12 40 30 10 kern/uipc_usrreq.c:251 (unp)
...
449 20000519 320038 62 15158 0 kern/uipc_usrreq.c:431 (so_rcv)
459 86616085 2880079 30 15699 4944 kern/uipc_usrreq.c:319 (so_snd)
146 2273360 640315 3 27918 29789 kern/kern_sig.c:1002 (process lock)
387 3325481 640099 5 38143 47670 kern/kern_descrip.c:420 (filedesc structure)
150 1881990 640155 2 64111 49033 kern/kern_descrip.c:368 (filedesc structure)
496 13792853 3685885 3 101692 132480 kern/kern_descrip.c:1988 (filedesc structure)
207 4061793 551604 7 115427 118242 kern/kern_synch.c:220 (process lock)
391 10332282 3685885 2 194387 129547 kern/kern_descrip.c:1967 (filedesc structure)
465 25504709 320042 79 1632192 294498 kern/uipc_usrreq.c:364 (unp)
470 124263922 2880084 43 13222757 2685853 kern/uipc_usrreq.c:309 (unp)
```

i.e. there is indeed heavy contention on the unp lock (column 5 counts the number of times we tried to acquire it and failed because someone else had the lock) – in fact about 5 times as many contentions as successful acquisitions.

With your patch and the same load:

```
3 20 9 2 0 0 kern/uipc_usrreq.c:1028 (unp_mtx)
3 22 9 2 0 0 kern/uipc_usrreq.c:1161 (unp_mtx)
5 29 9 3 0 2 kern/uipc_usrreq.c:1065 (unp_global_mtx)
5 53 18 2 0 76488 kern/uipc_usrreq.c:287 (unp_global_mtx)
6 33 9 3 0 0 kern/uipc_usrreq.c:236 (unp_mtx)
6 37 9 4 0 0 kern/uipc_usrreq.c:819 (unp_mtx)
7 7 1 7 0 0 vm/uma_core.c:2101 (unpcb)
8 49 9 5 0 0 kern/uipc_usrreq.c:1101 (unp_mtx)
11 136 18 7 0 1 kern/uipc_usrreq.c:458 (unp_global_mtx)
32 143 9 15 0 1 kern/uipc_usrreq.c:1160 (unp_global_mtx)
44 472 18 26 0 0 kern/uipc_usrreq.c:801 (unp_mtx)
123 310 9 34 0 0 kern/uipc_usrreq.c:1100 (unp_mtx)
147 452 9 50 0 0 kern/uipc_usrreq.c:1099 (unp_mtx)
172 748 9 83 0 0 kern/uipc_usrreq.c:473 (unp_mtx)
337 1592 9 176 0 0 kern/uipc_usrreq.c:1147 (unp_mtx)
350 1790 9 198 0 0 kern/uipc_usrreq.c:1146 (unp_mtx)
780 39405928 320038 123 0 0 kern/uipc_usrreq.c:618 (unp_mtx)
18 140 9 15 1 0 kern/uipc_usrreq.c:235 (unp_global_mtx)
70 717 18 39 1 3 kern/uipc_usrreq.c:800 (unp_global_mtx)
528 2444 9 271 1 1 kern/uipc_usrreq.c:1089 (unp_global_mtx)
158 616 9 68 2 2 kern/uipc_usrreq.c:476 (unp_mtx)
794 175382857 2880084 60 2 7686 kern/uipc_usrreq.c:574 (unp_mtx)
```

Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

```
4 25 9 2 3 2 kern/uipc_usrreq.c:422 (unp_global_mtx)
186 874 9 97 3 3 kern/uipc_usrreq.c:472 (unp_global_mtx)
768 33783759 320038 105 7442 0 kern/uipc_usrreq.c:696 (unp_mtx)
...
465 913127 320045 2 43130 35046 kern/uipc_socket.c:1101 (so_snd)
483 2453927 628737 3 44768 46177 kern/kern_sig.c:1002 (process lock)
767 124298544 2880082 43 70037 59994 kern/uipc_usrreq.c:581 (so_snd)
794 45176699 320038 141 83252 72140 kern/uipc_usrreq.c:617 (unp_global_mtx)
549 9858281 3200210 3 579269 712643 kern/kern_resource.c:1172 (sleep mtxpool)
554 17122245 631715 27 641888 268243 kern/kern_descrip.c:420 (filedesc structure)
388 3009912 631753 4 653540 260590 kern/kern_descrip.c:368 (filedesc structure)
642 49626755 3681446 13 1642954 682669 kern/kern_descrip.c:1988 (filedesc structure)
530 13802687 3681446 3 1663244 616899 kern/kern_descrip.c:1967 (filedesc structure)
477 23472709 2810986 8 5671248 1900047 kern/kern_synch.c:220 (process lock)
```

The top 10 heavily contended mutexes are very different (but note the number of mutex acquisitions, column 3, is about the same).

There is not much contention on `unp_global_mtx` any longer, but there is a lot more on some of the other mutexes, especially the process lock via `msleep()`. Off-hand I don't know what is the cause of this bottleneck (note: `libthr` is used as threading library and `libpthread` is not ported to `sparc64`).

Also, a lot of the contention that used to be on the `unp` lock seems to have fallen through onto contending `*two*` of the `filedesc` locks (all about 1.6 million contentions). This may also help to explain the performance drop.

With only 6 clients, the contention is about an order of magnitude less on most of the top 10, even though the number of mutex calls is only about 25% fewer than with 8 clients:

```
195 715786 240037 2 47462 48821 kern/uipc_socket.c:1101 (so_snd)
524 3456427 480079 7 50257 53368 kern/kern_descrip.c:420 (filedesc structure)
647 21650810 240030 90 50609 2 kern/uipc_usrreq.c:705 (so_rcv)
710 37962453 240031 158 63743 57814 kern/uipc_usrreq.c:617 (unp_global_mtx)
345 1624193 488866 3 80349 62950 kern/kern_descrip.c:368 (filedesc structure)
595 108074003 2160067 50 83327 63451 kern/uipc_usrreq.c:581 (so_snd)
453 3706420 519735 7 119947 181434 kern/kern_synch.c:220 (process lock)
469 13085667 2800771 4 122344 132478 kern/kern_descrip.c:1988 (filedesc structure)
320 8814736 2800771 3 200492 148967 kern/kern_descrip.c:1967 (filedesc structure)
440 7591194 2400171 3 544692 507583 kern/kern_resource.c:1172 (sleep mtxpool)
```

In summary, this is a good test case since it shows both the benefits of your patch and the areas of remaining concern.

Yours sincerely,  
Kristian D. Kennaway

Re: Fine-grained locking for POSIX local sockets (UNIX domain sockets)

*Attachment:* [pgp7i7k3RbekP.pgp](#)

*Description:* PGP signature