

---

Subject: Linux-VServer example results for sharing vs. separate mappings ...

Posted by [Herbert Poetzl](#) on Fri, 23 Mar 2007 19:30:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Hi Eric!

Hi Folks!

here is a real world example result from one of my tests regarding the benefit of sharing over separate memory

the setup is quite simple, a typical machine used by providers all over the world, a dual Pentium D 3.2GHz with 4GB of memory and a single 160GB SATA disk running a Linux-VServer kernel (2.6.19.7-vs2.2.0-rc18)

the Guest systems used are Mandriva 2007 guests with syslog, crond, sshd, apache, postfix and postgresql installed and running (all in all 17 processes per guest)

the disk space used by one guests is roughly 148MB

in addition to that, a normal host system is running with a few daemons (like sshd, httpd, postfix ...)

the first test setup is starting 200 of those guests one after the other and measuring the memory usage before and after the guest did start, as well as recording the time used to start them ...

this is done right after the machine was rebooted, in one test with 200 separate guests (i.e. 200 x 148MB) and in a second run with 200 unified guests (which means roughly 138MB of shared files)

separate guests:

GUEST	TIME	ACTIVE	BUFFERS	CACHE	ANON	MAPPED	SLAB	RECLAIM	URECL
001	0	16364	2600	20716	4748	3460	8164	2456	5708
002	7	30700	3816	42112	9052	8200	11056	3884	7172
003	13	44640	4872	62112	13364	12872	13248	5268	7980
004	20	58504	5972	82028	17684	17504	15348	6616	8732
005	28	72352	7056	102052	21948	22172	17640	8020	9620
....									
196	1567	2072172	156404	2409368	841168	915484	414056	246952	167104
197	1576	2080836	154680	2402344	845544	920268	414432	246784	167648

```
198 1585 2093424 153400 2399560 849696 924760 414892 246572 168320
199 1593 2103368 151540 2394048 854020 929660 415300 246324 168976
200 1599 2113004 149272 2382964 858344 934336 415528 245896 169632
```

unified guests:

```
GUEST TIME ACTIVE BUFFERS CACHE ANON MAPPED SLAB RECLAIM URECL
-----
001 0 16576 2620 20948 4760 3444 8232 2520 5712
002 10 31368 4672 74956 9068 8140 12976 5760 7216
003 14 38888 5364 110508 13368 9696 16516 8360 8156
004 18 44068 6104 146044 17696 11236 19868 10972 8896
005 22 49324 6824 181540 21964 12764 23264 13580 9684
....
196 1289 1159780 88856 2503448 841864 304544 383196 232944 150252
197 1294 1166528 88524 2500616 846168 306068 384056 233096 150960
198 1304 1172124 88468 2492268 850452 307596 384560 232988 151572
199 1313 1178876 88896 2488476 854840 309092 385384 233064 152320
200 1322 1184368 88568 2483208 858988 310640 386256 233388 152868
```

the second test was quite interesting too, as it showed nicely what the effect on the overall performance can be:

in this test, all guests are started at the same time, and the script waits until the last guest has successfully started ...

the 200 separate guests (as you probably can imagine) caused quite a load when started at once (there are a number of userspace tools preparing the guest on startup and setting up the context) and obviously they also pushed the memory limits somewhat ...

the startup for 200 separate guests (at once) did take this system 1h 11m 27s (compared to the 26m 39s in sequence)

the startup for 200 unified guests (at once) OTOH, did take 45s (yes, below a minute! compared to 22m 2s in sequential order)

HTH,  
Herbert

PS: if you need details for the setup, and/or want to recreate that on your system, just let me know, I can provide all the required data (including the guests)

---

Containers mailing list  
Containers@lists.linux-foundation.org  
<https://lists.linux-foundation.org/mailman/listinfo/containers>

---