Subject: Re: Re: [Vserver] VServer vs OpenVZ
Posted by dev on Mon, 12 Dec 2005 20:22:15 GMT
View Forum Message <> Reply to Message

>>> 1) "Fair scheduling" - as far as I can tell the VZ "fair scheduler"
>>> does nothing the VServer QoS/Limit system does. If anything, the VZ
>>> fair scheduler is not yet O(1) which is a big negative. VServer is
>>> built on standard kernel and therefore uses the O(1) scheduler (an
>>> absolute must when you have so many processes running on a single
>>> kernel)
 >>
>> this is not true! Fairscheduler in current implementation on 2.6
>> kernel is O(1) and doesn't depend on number of processes anyhow!
>> And we are working on improving it much more and implement some
>> additional features in it.
>
> Great, why not provide packages against latest Virtuozzo (with modules,
> vzfs, etc) for better real world testing? What numbers are you seeing
> in regards to load, based on my estimates with 3000 procs and an avg of
> 3 running on a server with a load average of 3 should drop to much
> lower. What kind of numbers do you see in your tests?
1. Virtuozzo 3.0 with O(1) scheduler will be released very soon. OpenVZ
already has it, so you can test it right now.
2. Can't understand your statement about 3000 procs/3 avg etc. What
estimations do you mean? Can you describe it in more details?

>>> 3) Disk/memory sharing - OpenVZ has nothing. Virtuozzo uses an
>>> overlay fs "vzfs". The templates are good for an enterprise
>>> environment, but really prove useless in a hosting environment. vzfs
>>> is overlay and therefore suffers from double caching (it caches both
>>> files in /vz/private (backing) and /vz/root (mount)).
>>
>> not sure what you mean... memory caching?! it is not true again then...

> The kernel caches based on inode number. If you modified the caching
> part of the module then I may be incorrect in my thinking. Take example:
>
> # ls -ai /vz/private/1/root/bin/ls
> 41361462 /vz/private/1/root/bin/ls
> # ls -ai /vz/template/redhat-as3-minimal/coreutils-4.5.3-26/bin/ls
> 1998864 /vz/template/redhat-as3-minimal/coreutils-4.5.3-26/bin/ls
> # ls -ai /vz/root/1/bin/ls
> 41361462 /vz/root/1/bin/ls
1. Kernel doesn't cache anything based on inode numbers. Inode numbers
are just a magic IDs. Nothing more. Internally everything is much more
complex.
2. inode numbers can be different with vzfs, but the data cached under
these inodes is the same, i.e. no double caching with vzfs happens. This

is the main purpose of VZFS and this is required for scalability: to
have only one instance of data in memory.

> The kernel will cache both inodes 41361462 and 1998864. Knowing that,
> when I look at my host servers with 8GB of RAM and see 4GB being used
> for cache/buffers I get angry.
Looks like you are misinterpreting /proc/meminfo output. Let me explain.
/proc/meminfo shows you amount of memory used for caching of files and
buffers, both of which are reclaimable. This means that:
- cached memory is not a wastage of you HW memory, it's a temporarely
cache of disk files. The bigger the better.
- since it's reclaimable it's not a memory which is pinned downed and is
freed on demand when some applications or kernel really need memory for
its own use. i.e. it means that you have 4GB of _FREE_ RAM which kernel
temporarily used for caches. Why are you angry with it?! I would be
happy in this situation :)
- as I wrote before the data is cached in memory only once, so in your
example with /bin/ls it takes only ~68k of data memory in caches +
dentry/inode caches (internal kernel structures). And raw figures in
/proc/meminfo doesn't allow to understand whether it was cached once or
twice or more times in memory. It's just 4GB of RAM which are used for
caches, no any other information here.
- on practice vzfs saves you 40-70% of VPS memory compared to OpenVZ
when VPSs are based on the same template. Sure, the more hungry VPS is
the less relative gain vzfs provide for such VPS.
So your comment about better scalability of OpenVZ than Virtuozzo looks
wrong to me...

> vzfs appears to be a standard unionfs
> with support for CoW to those who do not see the source. You ignored
> responding to how VServer does it which results in using a patched
> kernel to have special CoW links without a union mount. The links are
> based on a hard link architecture resulting in 1 inode. Also commenting
> on was ignored vunify and vzcache speeds.
It was not ignored actually, sorry that I didn't replied to it before
and made you think so.

We immediately started investigating your report. I believe vunify tool
is more like vzpkglink which is also fast enough. But it will be checked
more thoroughly.

More likely the whole vzcache will be reworked. I really appreciate such
reports and probably will return to you with more questions on it.

>> RSS is good yeah, but there are lot's of DoS possible if you limit RSS
>> only. No lowmem, no TCP bufs, etc... I personally know many ways of
>> DoSing of other resources, but if you don't care security this is
>> probably ok.

>>
> It does RSS and VM limiting with no guarantees. It also does locked
> pages, sockets, etc. The argument of who has more structures to limit
> is actually rather pointless now as VServer could take the OpenVZ limits
> to see what they can limit and decide which they want to implement. That
> is only a matter of time. I'm sure Herbert has seen output of
> /proc/user_beancounters before OpenVZ was even released and didn't see a
> reason for some of the limits. What I was pointing out was differences
> currently. A very minor advantage of VServer if they virtualize the
> meminfo structure to reflect memory/swap total/usage based on the RSS/VM
> limits.
meminfo will be fixed soon, I suppose.

Kirill
