Subject: ploop, simfs, and disk space
Posted by kir on Fri, 06 Apr 2012 21:33:15 GMT
View Forum Message <> Reply to Message

I want to summarise some facts about simfs, ploop, and disk quota.

== simfs case ==
If container is on simfs, it is using host file system, usually /vz.
Because many containers share one file system, per-container limits are needed for both disk space used and disk i-nodes (roughly number of files/directories) used. These two limits are called vzquota, and are controlled by --diskspace and --diskinodes parameters for vzctl set command.

For both diskspace and diskinodes, there are two values -- soft limit and hard limit, you can specify those using sss:hhh syntax. For example, "vzctl set 333 --diskspace 10G:11G --save" command sets the soft disk space limit to 10 GB and the hard disk space limit to 11G. The difference between soft and hard limit is that the soft limit can be temporary exceeded, while soft limit can not be exceeded. Here "temporary" is defined by the third parameter, --quotatime, which sets the time (in seconds) during which soft limit can be exceeded. This value is otherwise known as the grace period. Once the grace period has expired, the soft limit is enforced as a hard limit.

Example: admin sets disk limits in the following way:
vzctl set 333 --diskspace 10G:11G --diskinodes 1M:1.1M --quotatime 3600 --save

Now, a container root can use 10G of disk space, and have about 1 million files inside his CT. He can have 11G of disk space and about 1.1 million files, but for no longer than 1 hour. If he uses more than 10G of disk space, during the first hour (and only during the first hour) he will still be able to use 11th gigabyte.

This dimensional system of space, inodes, soft limits, hard limits and grace period is nothing new, it's the same as traditional UNIX per-user and per-group disk quotas. The only major difference is in this case quotas are per-CT (per simfs mount point).

There is a --diskquota parameter (and DISK_QUOTA config file parameter) which is used to enable/disable per-CT disk quotas. If you set DISK_QUOTA=no in /etc/vz/vz.conf, no per-CT disk quotas will be initialized. If you set DISK_QUOTA=no in CT configuration file (e.g. /etc/vz/conf/333.conf), no disk quotas for this CT will be initialized.

NOTE that as with any other disk quota, if you will write to the file system bypassing the quota (such as directly to VE_PRIVATE, e.g. /vz/private/333), current quota usage values will be incorrect. In that

case, you need to stop the CT and run vzctl quotainit, to recalculate quota usage. In some cases (such as after incorrect system shutdown caused by power outage) quota files are marked dirty, and such recalculation is happening automatically during CT start.

For the sake of completeness, there is vzctl quotaon and vzctl quotaoff commands, but usually you don't have to use those two, because quotaon is performed during vzctl mount (and vzctl start), and quotaoff is performed during vzctl stop (and vzctl umount).

From inside the CT, utilities such as df are showing those quota limits instead of actual available disk space and inodes (this is implemented in the kernel by having a special version of statfs() syscall for simfs which looks into vzquota). Sometimes it gets complicated, so if you see something strange in df output, it is either incorrect quota values (and you need to recalculate quota usage, see above), or perhaps the filesystem disk space available is less than quota limits. For lots of gory details on this stuff, please see
http://wiki.openvz.org/Disk_quota,_df_and_stat_weird_behavio ur

Also, you can check /proc/vz/vzquota to see for which containers quota is on, as well as its current limits and usage values.

I am leaving more advanced topics such as using vzquota utility directly as a (highly optional) exercise for (highly) advanced users.

== ploop case ==

In ploop case, there is an image file and the underlying file system, so there is no shared file system and vzquota is naturally not required. Therefore, options --diskquota (and DISK_QUOTA parameter), --diskinodes and --quotatime are silently ignored.

Option --diskspace is not ignored, but instead of changing vzquota disk space limit, it initiates the resize of the CT ploop image file and the filesystem which resides on top of that image.

NOTE that image and file system resize, especially in case when the CT is running (so-called online resize) is quite tricky, and in worst case scenario can lead to image or filesystem damage that is beyond repair. So exercise it a lot in testing environment, but do not abuse it in production*.

NOTE that specifying two values for --diskspace in case of ploop makes no sense. Only one value (hard limit) is used (and as with other parameters, if you only specify one value, second one becomes equal to the first one). So using --diskspace 1G:1.1G is the same as --diskspace 1.1G (or --diskspace 0:1.1G). Easy rule: do not use two numbers for

diskspace, just one.

* NOTE ploop is not yet ready for production, and will not be for at
least a few more months.

Subject: Re: ploop, simfs, and disk space
Posted by Corin Langosch on Sun, 08 Apr 2012 12:04:07 GMT
View Forum Message <> Reply to Message

Hi Kir,

Am 06.04.2012 23:33, schrieb Kir Kolyshkin:
>
> * NOTE ploop is not yet ready for production, and will not be for at
> least a few more months.
>

Now this really confuses/ worries me as the latest *stable* release has
ploop support. If it's not stable (so not ready for production), why is
it included in a *stable* release? And it's not even marked as
experimental - not on the website nor in the man pages.

How can I identify which kernel, tools and features are (supposed to be)
stable and ready for production if not by the "Status" (stable,
maintained) on the website/ man pages?

Corin

Subject: Re: ploop, simfs, and disk space
Posted by kir on Sun, 08 Apr 2012 12:46:45 GMT
View Forum Message <> Reply to Message

On 04/08/2012 04:04 PM, Corin Langosch wrote:
> Hi Kir,
>
> Am 06.04.2012 23:33, schrieb Kir Kolyshkin:
>> * NOTE ploop is not yet ready for production, and will not be for at
>> least a few more months.
>>
> Now this really confuses/ worries me as the latest *stable* release has
> ploop support. If it's not stable (so not ready for production), why is
> it included in a *stable* release? And it's not even marked as
> experimental - not on the website nor in the man pages.

Sorry that I haven't stated it before, should've done it. Let me explain.

The RHEL6-based kernel is stable (or supposed to be stable).

The ploop feature is not, and it not supposed to be as of now. Call it "beta" or "technology preview" or any other name suggesting it is not ready for production systems yet.

This feature is orthogonal to other kernel functionality, so there is no paradox here.

> How can I identify which kernel, tools and features are (supposed to be)
> stable and ready for production if not by the "Status" (stable,
> maintained) on the website/ man pages?

(1) Man pages are not supposed to mark feature status.

(2) I have put a big red warning on http://wiki.openvz.org/Ploop saying this is not yet ready for production.