Subject: Re: [PATCH v4 14/14] Add documentation about the kmem controller
Posted by Michal Hocko on Thu, 11 Oct 2012 14:35:59 GMT
View Forum Message <> Reply to Message

On Mon 08-10-12 14:06:20, Glauber Costa wrote:
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> ---
> Documentation/cgroups/memory.txt | 55 ++++++++++++++++++++++++++++++++++++++-
> 1 file changed, 54 insertions(+), 1 deletion(-)
>
> diff --git a/Documentation/cgroups/memory.txt b/Documentation/cgroups/memory.txt
> index c07f7b4..9b08548 100644
> --- a/Documentation/cgroups/memory.txt
> +++ b/Documentation/cgroups/memory.txt
> @@ -71,6 +71,11 @@ Brief summary of control files.
>   memory.oom_control   # set/show oom controls.
>   memory.numa_stat   # show the number of memory usage per numa node
>
> + memory.kmem.limit_in_bytes   # set/show hard limit for kernel memory
> + memory.kmem.usage_in_bytes   # show current kernel memory allocation
> + memory.kmem.failcnt   # show the number of kernel memory usage hits limits
> + memory.kmem.max_usage_in_bytes # show max kernel memory usage recorded
> +
>   memory.kmem.tcp.limit_in_bytes # set/show hard limit for tcp buf memory
>   memory.kmem.tcp.usage_in_bytes # show current tcp buf memory allocation
>   memory.kmem.tcp.failcnt   # show the number of tcp buf memory usage hits limits
> @@ -268,20 +273,62 @@ the amount of kernel memory used by the system. Kernel memory is
fundamentally
> different than user memory, since it can't be swapped out, which makes it
> possible to DoS the system by consuming too much of this precious resource.
>
> +Kernel memory won't be accounted at all until it is limited. This allows for

until limit on a group is set.

> +existing setups to continue working without disruption. Note that it is
> +possible to account it without an effective limit by setting the limits
> +to a very high number (like RESOURCE_MAX -1page).

I have brought that up in an earlier patch already. Why not just do echo
-1 (which translates to RESOURCE_MAX internally) and be done with that.
RESOURCE_MAX-1 sounds quite inconvenient.

> The limit cannot be set
> +if the cgroup have children, or if there are already tasks in the cgroup.

I would start by stating that if children are accounted automatically if
their parent is accounted already and there is no need to set a limit to

enforce that. In fact the limit cannot be set if ....

> +
> +After a controller is first limited, it will be kept being accounted until it

group is limited not the controller.

> +is removed. The memory limitation itself, can of course be removed by writing
> +-1 to memory.kmem.limit_in_bytes

This might be confusing and one could think that also accounting would
be removed. I wouldn't mention it at all.
> +
> Kernel memory limits are not imposed for the root cgroup. Usage for the root
> -cgroup may or may not be accounted.
> +cgroup may or may not be accounted. The memory used is accumulated into
> +memory.kmem.usage_in_bytes, or in a separate counter when it makes sense.

Which separate counter? Is this about tcp kmem?

> +The main "kmem" counter is fed into the main counter, so kmem charges will
> +also be visible from the user counter.
>
> Currently no soft limit is implemented for kernel memory. It is future work
> to trigger slab reclaim when those limits are reached.
>
> 2.7.1 Current Kernel Memory resources accounted
>
> +* stack pages: every process consumes some stack pages. By accounting into
> +kernel memory, we prevent new processes from being created when the kernel
> +memory usage is too high.
> +
> * sockets memory pressure: some sockets protocols have memory pressure
> thresholds. The Memory Controller allows them to be controlled individually
> per cgroup, instead of globally.
>
> * tcp memory pressure: sockets memory pressure for the tcp protocol.
>
> +2.7.3 Common use cases
> +
> +Because the "kmem" counter is fed to the main user counter, kernel memory can
> +never be limited completely independently of user memory. Say "U" is the user
> +limit, and "K" the kernel limit. There are three possible ways limits can be
> +set:
> +
> +   U != 0, K = 0:

K is not 0 it is unaccounted (disabled)

> +   This is the standard memcg limitation mechanism already present before kmem
> +   accounting. Kernel memory is completely ignored.
> +
> +   U,K != 0, K < U:

I would keep K < U
> +   Kernel memory is effectively set as a percentage of the user memory. This

not a percentage it is subset of the user memory

> +   setup is useful in deployments where the total amount of memory per-cgroup
> +   is overcommited. Overcommiting kernel memory limits is definitely not
> +   recommended, since the box can still run out of non-reclaimable memory.
> +   In this case, the admin could set up K so that the sum of all groups is
> +   never greater than the total memory, and freely set U at the cost of his
> +   QoS.
> +
> +   U,K != 0, K >= U:
> +   Since kmem charges will also be fed to the user counter, this setup gives
> +   the admin a unified view of memory. Reclaim will be triggered for the cgroup
> +   for both kinds of memory.

This is also useful for tracking kernel memory allocation.

> +
> 3. User Interface
>
> 0. Configuration
> @@ -290,6 +337,7 @@ a. Enable CONFIG_CGROUPS
> b. Enable CONFIG_RESOURCE_COUNTERS
> c. Enable CONFIG_MEMCG
> d. Enable CONFIG_MEMCG_SWAP (to use swap extension)
> +d. Enable CONFIG_MEMCG_KMEM (to use kmem extension)
>
> 1. Prepare the cgroups (see cgroups.txt, Why are cgroups needed?)
> # mount -t tmpfs none /sys/fs/cgroup
> @@ -406,6 +454,11 @@ About use_hierarchy, see Section 6.
>   Because rmdir() moves all pages to parent, some out-of-use page caches can be
>   moved to the parent. If you want to avoid that, force_empty will be useful.
>
> + Also, note that when memory.kmem.limit_in_bytes is set the charges due to
> + kernel pages will still be seen. This is not considered a failure and the
> + write will still return success. In this case, it is expected that
> + memory.kmem.usage_in_bytes == memory.usage_in_bytes.
> +
>   About use_hierarchy, see Section 6.
>

> 5.2 stat file
> --
> 1.7.11.4
>
> --
> To unsubscribe from this list: send the line "unsubscribe cgroups" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

--
Michal Hocko
SUSE Labs
