Subject: [PATCH 10/10] memcg: Document kernel memory accounting.
Posted by Suleiman Souhlal on Mon, 27 Feb 2012 22:58:53 GMT
View Forum Message <> Reply to Message

Signed-off-by: Suleiman Souhlal <suleiman@google.com>

 Documentation/cgroups/memory.txt | 44 ++++++++++++++++++++++++++++++++++++++--
 1 files changed, 41 insertions(+), 3 deletions(-)

diff --git a/Documentation/cgroups/memory.txt b/Documentation/cgroups/memory.txt
index 4c95c00..64c6cc8 100644
--- a/Documentation/cgroups/memory.txt
+++ b/Documentation/cgroups/memory.txt
@@ -74,6 +74,11 @@ Brief summary of control files.

  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.usage_in_bytes # show current kernel memory usage
+ memory.kmem.limit_in_bytes # show/set limit of kernel memory usage
+ memory.kmem.independent_kmem_limit # show/set control of kernel memory limit
+   (See 2.7 for details)
+ memory.kmem.slabinfo   # show cgroup's slabinfo

 1. History

@@ -266,10 +271,20 @@ 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 limits are not imposed for the root cgroup. Usage for the root
-cgroup may or may not be accounted.
+cgroup is accounted if CONFIG_CGROUP_MEM_RES_CTLR_KMEM_ACCT_ROOT is set.

-Currently no soft limit is implemented for kernel memory. It is future work
-to trigger slab reclaim when those limits are reached.
+A cgroup's kernel memory is counted into its memory.kmem.usage_in_bytes.
+
+memory.kmem.independent_kmem_limit controls whether or not kernel memory
+should also be counted into the cgroup's memory.usage_in_bytes.
+If it is set, it is possible to specify a limit for kernel memory with
+memory.kmem.limit_in_bytes.
+
+Upon cgroup deletion, all the remaining kernel memory gets moved to the
+root cgroup (if CONFIG_CGROUP_MEM_RES_CTLR_KMEM_ACCT_ROOT is set).
+
+An accounted kernel memory allocation may trigger reclaim in that cgroup,
+and may also OOM.

2.7.1 Current Kernel Memory resources accounted

@@ -279,6 +294,26 @@ per cgroup, instead of globally.

* tcp memory pressure: sockets memory pressure for the tcp protocol.

+* slab memory.
+
+2.7.1.1 Slab memory accounting
+
+Slab gets accounted on a per-page basis, which is done by using per-cgroup
+kmem_caches. These per-cgroup kmem_caches get created on-demand, the first
+time a specific kmem_cache gets used by a cgroup.
+
+Slab memory that cannot be attributed to a cgroup gets charged to the root
+cgroup.
+
+A per-cgroup kmem_cache is named like the original, with the cgroup's name
+in parethesis.
+
+When a cgroup is destroyed, all its kmem_caches get migrated to the root
+cgroup, and "dead" is appended to their name, to indicate that they are not
+going to be used for new allocations.
+These dead caches automatically get removed once there are no more active
+slab objects in them.
+
3. User Interface

0. Configuration
@@ -423,6 +458,8 @@ active_anon - # of bytes of anonymous and swap cache memory on
active
inactive_file - # of bytes of file-backed memory on inactive LRU list.
active_file - # of bytes of file-backed memory on active LRU list.
unevictable - # of bytes of memory that cannot be reclaimed (mlocked etc).
+kernel_bypassed_memory - # of bytes of kernel memory that should have been
+ accounted, but got bypassed to the root cgroup.

status considering hierarchy (see memory.use_hierarchy settings)

@@ -442,6 +479,7 @@ total_active_anon - sum of all children's "active_anon"
total_inactive_file - sum of all children's "inactive_file"
total_active_file - sum of all children's "active_file"
total_unevictable - sum of all children's "unevictable"
+total_kernel_bypassed_memory - sum of all children's "kernel_bypassed_memory"

The following additional stats are dependent on CONFIG_DEBUG_VM.

--

1.7.7.3