Subject: [RFC] slub: show dead memcg caches in a separate file
Posted by Glauber Costa on Thu, 03 May 2012 18:47:21 GMT
View Forum Message <> Reply to Message

One of the very few things that still unsettles me in the kmem
controller for memcg, is how badly we mess up with the
/proc/slabinfo file.

It is alright to have the cgroup caches listed in slabinfo, but once
they die, I think they should be removed right away. A box full
of containers that come and go will rapidly turn that file into
a supreme mess. However, we currently leave them there so we can
determine where our used memory currently is.

This patch attempts to clean this up by creating a separate proc file
only to handle the dead slabs. Among other advantages, we need a lot
less information in a dead cache: only its current size in memory
matters to us.

So besides avoiding polution of the slabinfo files, we can access
dead cache information itself in a cleaner way.

I implemented this as a proof of concept while finishing up
my last round for submission. But I am sending this separately
to collect opinions from all of you. I can either implement
a version of this for the slab, or follow any other route.

Thanks

Signed-off-by: Glauber Costa <glommer@parallels.com>
CC: Christoph Lameter <cl@linux.com>
CC: Pekka Enberg <penberg@cs.helsinki.fi>
CC: Michal Hocko <mhocko@suse.cz>
CC: Kamezawa Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
CC: Johannes Weiner <hannes@cmpxchg.org>
CC: Suleiman Souhlal <suleiman@google.com>
CC: Frederic Weisbecker <fweisbec@gmail.com>
---
 include/linux/slab.h | 3 ++
 mm/slub.c | 82 ++
 2 files changed, 85 insertions(+), 0 deletions(-)

diff --git a/include/linux/slab.h b/include/linux/slab.h
index 5df00c1..d6a0cf4 100644
--- a/include/linux/slab.h
+++ b/include/linux/slab.h
@@ -171,6 +171,9 @@ struct mem_cgroup_cache_params {
 #endif

```
  struct list_head destroyed_list; /* Used when deleting cpuset cache */
 };
+
+extern void memcg_finish_slab_destruction(void);
+extern void memcg_start_slab_destruction(void);
 #endif

 /*
diff --git a/mm/slub.c b/mm/slub.c
index 0652e99..de066e3 100644
--- a/mm/slub.c
+++ b/mm/slub.c
@@ -5657,6 +5657,11 @@ static int s_show(struct seq_file *m, void *p)

  s = list_entry(p, struct kmem_cache, list);

+#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+ if (s->memcg_params.dead)
+  return 0;
+#endif
+
  for_each_online_node(node) {
   struct kmem_cache_node *n = get_node(s, node);

@@ -5688,6 +5693,83 @@ static const struct seq_operations slabinfo_op = {
  .show = s_show,
 };

+#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+static void *s_start_dead(struct seq_file *m, loff_t *pos)
+{
+ loff_t n = *pos;
+
+ down_read(&slub_lock);
+ if (!n) {
+  seq_puts(m, "slab_name                                ");
+  seq_puts(m, "pagesperslab num_slabs");
+  seq_putc(m, '\n');
+ }
+
+ return seq_list_start(&slab_caches, *pos);
+}
+
+static int s_show_dead(struct seq_file *m, void *p)
+{
+ unsigned long nr_slabs = 0;
+ struct kmem_cache *s;
+ int node;
```

```
+
+ s = list_entry(p, struct kmem_cache, list);
+
+ if (!s->memcg_params.dead)
+  return 0;
+
+ for_each_online_node(node) {
+  struct kmem_cache_node *n = get_node(s, node);
+
+  if (!n)
+   continue;
+
+  nr_slabs += atomic_long_read(&n->nr_slabs);
+ }
+
+ seq_printf(m, "%-40s %11d", s->name, (1 << oo_order(s->oo)));
+ seq_printf(m, " %9lu", nr_slabs);
+ seq_putc(m, '\n');
+ return 0;
+}
+
+
+static const struct seq_operations dead_slabinfo_op = {
+ .start = s_start_dead,
+ .next = s_next,
+ .stop = s_stop,
+ .show = s_show_dead,
+};
+
+static int dead_slabinfo_open(struct inode *inode, struct file *file)
+{
+ return seq_open(file, &dead_slabinfo_op);
+}
+
+static const struct file_operations proc_dead_slabinfo_operations = {
+ .open  = dead_slabinfo_open,
+ .read  = seq_read,
+ .llseek  = seq_lseek,
+ .release = seq_release,
+};
+
+static atomic_t dead_memcg_counter;
+
+void memcg_start_slab_destruction(void)
+{
+ if (atomic_add_return(1, &dead_memcg_counter) == 1)
+  proc_create("dead_slabinfo", S_IRUSR, NULL,
+       &proc_dead_slabinfo_operations);
```

```
+}
+
+void memcg_finish_slab_destruction(void)
+{
+ if (atomic_dec_and_test(&dead_memcg_counter))
+  remove_proc_entry("dead_slabinfo", NULL);
+}
+#endif
+
 static int slabinfo_open(struct inode *inode, struct file *file)
 {
  return seq_open(file, &slabinfo_op);
--
1.7.7.6
```