Subject: [PATCH 15/23] slab: create duplicate cache
Posted by Glauber Costa on Sun, 22 Apr 2012 23:53:32 GMT
View Forum Message <> Reply to Message

This patch provides kmem_cache_dup(), that duplicates
a cache for a memcg, preserving its creation properties.
Object size, alignment and flags are all respected.
An exception is the SLAB_PANIC flag, since cache creation
inside a memcg should not be fatal.

This code is mostly written by Suleiman Souhlal,
with some adaptations and simplifications by me.

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>
---
 mm/slab.c | 36 ++++++++++++++++++++++++++++++++++++
 1 files changed, 36 insertions(+), 0 deletions(-)

diff --git a/mm/slab.c b/mm/slab.c
index 362bb6e..c4ef684 100644
--- a/mm/slab.c
+++ b/mm/slab.c
@@ -301,6 +301,8 @@ static void free_block(struct kmem_cache *cachep, void **objpp, int len,
    int node);
 static int enable_cpucache(struct kmem_cache *cachep, gfp_t gfp);
 static void cache_reap(struct work_struct *unused);
+static int do_tune_cpucache(struct kmem_cache *cachep, int limit,
+   int batchcount, int shared, gfp_t gfp);

 /*
  * This function must be completely optimized away if a constant is passed to
@@ -2593,6 +2595,40 @@ kmem_cache_create(const char *name, size_t size, size_t align,
 }
 EXPORT_SYMBOL(kmem_cache_create);

+#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+struct kmem_cache *
+kmem_cache_dup(struct mem_cgroup *memcg, struct kmem_cache *cachep)
+{
+ struct kmem_cache *new;
+ int flags;
+ char *name;

```
+
+ name = mem_cgroup_cache_name(memcg, cachep);
+ if (!name)
+  return NULL;
+
+ flags = cachep->flags & ~SLAB_PANIC;
+ mutex_lock(&cache_chain_mutex);
+ new = __kmem_cache_create(memcg, name, obj_size(cachep),
+      cachep->memcg_params.orig_align, flags, cachep->ctor);
+
+ if (new == NULL) {
+  mutex_unlock(&cache_chain_mutex);
+  kfree(name);
+  return NULL;
+ }
+
+ if ((cachep->limit != new->limit) ||
+     (cachep->batchcount != new->batchcount) ||
+     (cachep->shared != new->shared))
+  do_tune_cpucache(new, cachep->limit, cachep->batchcount,
+       cachep->shared, GFP_KERNEL);
+ mutex_unlock(&cache_chain_mutex);
+
+ return new;
+}
+#endif /* CONFIG_CGROUP_MEM_RES_CTLR_KMEM */
+
 #if DEBUG
 static void check_irq_off(void)
 {
--
1.7.7.6
```