Subject: [PATCH v4 12/25] sl[au]b: always get the cache from its page in kfree
Posted by Glauber Costa on Mon, 18 Jun 2012 10:28:05 GMT
View Forum Message <> Reply to Message

struct page already have this information. If we start chaining
caches, this information will always be more trustworthy than
whatever is passed into the function

A parent pointer is added to the slub structure, so we can make sure
the freeing comes from either the right slab, or from its rightful
parent.

[v3: added parent testing with VM_BUG_ON]

Signed-off-by: Glauber Costa <glommer@parallels.com>
CC: Christoph Lameter <cl@linux.com>
CC: Pekka Enberg <penberg@cs.helsinki.fi>
---
 mm/slab.c | 5 ++++-
 mm/slab.h | 10 ++++++++++
 mm/slub.c | 3 ++-
 3 files changed, 16 insertions(+), 2 deletions(-)

diff --git a/mm/slab.c b/mm/slab.c
index c30a61c..3783a6a 100644
--- a/mm/slab.c
+++ b/mm/slab.c
@@ -3712,9 +3712,12 @@ EXPORT_SYMBOL(__kmalloc);
  * Free an object which was previously allocated from this
  * cache.
  */
-void kmem_cache_free(struct kmem_cache *cachep, void *objp)
+void kmem_cache_free(struct kmem_cache *s, void *objp)
 {
  unsigned long flags;
+ struct kmem_cache *cachep = virt_to_cache(objp);
+
+ VM_BUG_ON(!((s == cachep) | slab_is_parent(s, cachep)));

  local_irq_save(flags);
  debug_check_no_locks_freed(objp, cachep->size);
diff --git a/mm/slab.h b/mm/slab.h
index 1781580..0a3e712 100644
--- a/mm/slab.h
+++ b/mm/slab.h
@@ -63,4 +63,14 @@ static inline bool cache_match_memcg(struct kmem_cache *cachep,
 }

```
 void __init memcg_slab_register_all(void);
+
+static inline bool slab_is_parent(struct kmem_cache *s,
+       struct kmem_cache *candidate)
+{
+#if defined(CONFIG_CGROUP_MEM_RES_CTLR_KMEM) && defined(CONFIG_DEBUG_VM)
+ return candidate == s->memcg_params.parent;
+#else
+ return false;
+#endif
+}
 #endif
diff --git a/mm/slub.c b/mm/slub.c
index ca4b8e0..e685cfa 100644
--- a/mm/slub.c
+++ b/mm/slub.c
@@ -2597,7 +2597,8 @@ void kmem_cache_free(struct kmem_cache *s, void *x)

  page = virt_to_head_page(x);

- slab_free(s, page, x, _RET_IP_);
+ VM_BUG_ON(!((page->slab == s) | slab_is_parent(page->slab, s)));
+ slab_free(page->slab, page, x, _RET_IP_);

  trace_kmem_cache_free(_RET_IP_, x);
 }
--
1.7.10.2
```