Subject: [PATCH v3 09/16] sl[au]b: always get the cache from its page in kfree
Posted by Glauber Costa on Tue, 18 Sep 2012 14:12:03 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 | 11 ++++++++++
 mm/slub.c | 4 +++-
 3 files changed, 18 insertions(+), 2 deletions(-)

```
diff --git a/mm/slab.c b/mm/slab.c
index f2d760c..18de3f6 100644
--- a/mm/slab.c
+++ b/mm/slab.c
@@ -3938,9 +3938,12 @@ 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(!slab_equal_or_parent(cachep, s));

  local_irq_save(flags);
  debug_check_no_locks_freed(objp, cachep->object_size);
diff --git a/mm/slab.h b/mm/slab.h
index 6f2a34d..f2501ab 100644
--- a/mm/slab.h
+++ b/mm/slab.h
@@ -60,11 +60,22 @@ static inline bool cache_match_memcg(struct kmem_cache *cachep,
  return cachep->memcg_params.memcg == memcg;
 }
```

```
+static inline bool slab_equal_or_parent(struct kmem_cache *s,
+      struct kmem_cache *p)
+{
+ return (p == s) || (p == s->memcg_params.parent);
+}
 #else
 static inline bool cache_match_memcg(struct kmem_cache *cachep,
          struct mem_cgroup *memcg)
 {
  return true;
 }
+
+static inline bool slab_equal_or_parent(struct kmem_cache *s,
+      struct kmem_cache *p)
+{
+ return true;
+}
 #endif
 #endif
diff --git a/mm/slub.c b/mm/slub.c
index 4778548..a045dfc 100644
--- a/mm/slub.c
+++ b/mm/slub.c
@@ -2604,7 +2604,9 @@ 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(!slab_equal_or_parent(page->slab, s));
+
+ slab_free(page->slab, page, x, _RET_IP_);

  trace_kmem_cache_free(_RET_IP_, x);
 }
--
1.7.11.4
```