
Subject: [PATCH v3 01/28] slab: move FULL state transition to an initcall
Posted by [Glauber Costa](#) on Fri, 25 May 2012 13:03:21 GMT
[View Forum Message](#) <> [Reply to Message](#)

During `kmem_cache_init_late()`, we transition to the LATE state, and after some more work, to the FULL state, its last state

This is quite different from `slub`, that will only transition to its last state (`SYSFS`), in a (late)initcall, after a lot more of the kernel is ready.

This means that in `slab`, we have no way to taking actions dependent on the initialization of other pieces of the kernel that are supposed to start way after `kmem_init_late()`, such as `cgroups` initialization.

To achieve more consistency in this behavior, that patch only transitions to the LATE state in `kmem_init_late`. In my analysis, `setup_cpu_cache()` should be happy to test for `>= LATE`, instead of `== FULL`. It also has passed some tests I've made.

We then only mark FULL state after the reap timers are in place, meaning that no further setup is expected.

Signed-off-by: Glauber Costa <glommer@parallels.com>

Acked-by: Christoph Lameter <cl@linux.com>

CC: Pekka Enberg <penberg@cs.helsinki.fi>

CC: David Rientjes <rientjes@google.com>

mm/slab.c | 8 ++++----

1 files changed, 4 insertions(+), 4 deletions(-)

diff --git a/mm/slab.c b/mm/slab.c

index e901a36..8658d72 100644

--- a/mm/slab.c

+++ b/mm/slab.c

@@ -1695,9 +1695,6 @@ void __init kmem_cache_init_late(void)

BUG();

mutex_unlock(&cache_chain_mutex);

- /* Done! */

- g_cpucache_up = FULL;

-

/*

* Register a cpu startup notifier callback that initializes

* `cpu_cache_get` for all new cpus

@@ -1727,6 +1724,9 @@ static int __init cpucache_init(void)

*/

for_each_online_cpu(cpu)

```

    start_cpu_timer(cpu);
+
+ /* Done! */
+ g_cpucache_up = FULL;
    return 0;
}
__initcall(cpucache_init);
@@ -2194,7 +2194,7 @@ static size_t calculate_slab_order(struct kmem_cache *cachep,

static int __init_refok setup_cpu_cache(struct kmem_cache *cachep, gfp_t gfp)
{
- if (g_cpucache_up == FULL)
+ if (g_cpucache_up >= LATE)
    return enable_cpucache(cachep, gfp);

    if (g_cpucache_up == NONE) {
--
1.7.7.6

```
