
Subject: [PATCH 22/33] memory controller resource counters v7 fix
Posted by [Paul Menage](#) on Mon, 17 Sep 2007 21:03:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: David Rientjes <rientjes@google.com>
(container->cgroup renaming by Paul Menage <menage@google.com>)

There's a gotcha in `res_counter_charge_locked()` because of C99 6.3.1.8(1) since both `counter->limit` and `'val'` are of unsigned long type, the result of the subtraction will be the same; no promotion is required. So if `'val'` is greater than `counter->limit`, it will always be larger than `counter->usage` and the conditional will fail. Simply casting this to signed doesn't work since `counter->usage` is also unsigned and thus the result of the subtraction will be promoted to unsigned since the ranks are the same.

Even though the only (current) use of `res_counter_charge()` is with a `'val'` actual of 1, this still fails if you set `counter->limit` to 0. No chance of overflow unless you're running on a machine with 4KB pages and 16TB of memory.

Signed-off-by: David Rientjes <rientjes@google.com>
Signed-off-by: Paul Menage <menage@google.com>

kernel/res_counter.c | 2 +-
1 files changed, 1 insertion(+), 1 deletion(-)

diff -puN kernel/res_counter.c~memory-controller-resource-counters-v7-fix kernel/res_counter.c
--- a/kernel/res_counter.c~memory-controller-resource-counters-v7-fix
+++ a/kernel/res_counter.c
@@ -21,7 +21,7 @@ void res_counter_init(struct res_counter

```
int res_counter_charge_locked(struct res_counter *counter, unsigned long val)
{
- if (counter->usage > (counter->limit - val)) {
+ if (counter->usage + val > counter->limit) {
    counter->failcnt++;
    return -ENOMEM;
}
```

—

--

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
