Subject: Re: [PATCH 2/4] Add a __GFP_SLABMEMCG flag
Posted by Glauber Costa on Sat, 09 Jun 2012 08:24:03 GMT
View Forum Message <> Reply to Message

On 06/09/2012 04:56 AM, James Bottomley wrote:
> On Fri, 2012-06-08 at 14:31 -0500, Christoph Lameter wrote:
>> On Fri, 8 Jun 2012, Glauber Costa wrote:
>>
>>> */
>>> #define __GFP_NOTRACK_FALSE_POSITIVE (__GFP_NOTRACK)
>>>
>>> -#define __GFP_BITS_SHIFT 25 /* Room for N __GFP_FOO bits */
>>> +#define __GFP_BITS_SHIFT 26 /* Room for N __GFP_FOO bits */
>>> #define __GFP_BITS_MASK ((__force gfp_t)((1<< __GFP_BITS_SHIFT) - 1))
>>
>> Please make this conditional on CONFIG_MEMCG or so. The bit can be useful
>> in particular on 32 bit architectures.
>
> I really don't think that's at all a good idea. It's asking for trouble
> when we don't spot we have a flag overlap. It also means that we're
> trusting the reuser to know that their use case can never clash with
> CONFIG_MEMGC and I can't think of any configuration where this is
> possible currently.
>
> I think making the flag define of __GFP_SLABMEMCG conditional might be a
> reasonable idea so we get a compile failure if anyone tries to use it
> when !CONFIG_MEMCG.
>

Which is also difficult since that's not code that we can BUG or WARN,
but just a number people or and and into their own flags. And it is too
fragile to rely on any given sequence we put here (like -1UL, etc) to
provide predictable enough results to tell someone he is doing it wrong.

A much better approach if we want to protect against that, is to add
code in the page or slab allocator (or both) to ignore and WARN upon
seeing this flag when !memcg.

I'd leave the flag itself alone.