Subject: Re: [PATCH 05/11] Add a __GFP_KMEMCG flag
Posted by David Rientjes on Tue, 26 Jun 2012 04:25:54 GMT
View Forum Message <> Reply to Message

On Mon, 25 Jun 2012, Glauber Costa wrote:

> This flag is used to indicate to the callees that this allocation will be
> serviced to the kernel. It is not supposed to be passed by the callers
> of kmem_cache_alloc, but rather by the cache core itself.
>

Not sure what "serviced to the kernel" means, does this mean that the
memory will not be accounted for to the root memcg?

> CC: Christoph Lameter <cl@linux.com>
> CC: Pekka Enberg <penberg@cs.helsinki.fi>
> CC: Michal Hocko <mhocko@suse.cz>
> CC: Kamezawa Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
> CC: Johannes Weiner <hannes@cmpxchg.org>
> CC: Suleiman Souhlal <suleiman@google.com>
> ---
> include/linux/gfp.h | 8 +++++++-
> 1 file changed, 7 insertions(+), 1 deletion(-)
>
> diff --git a/include/linux/gfp.h b/include/linux/gfp.h
> index 1e49be4..8f4079f 100644
> --- a/include/linux/gfp.h
> +++ b/include/linux/gfp.h
> @@ -37,6 +37,9 @@ struct vm_area_struct;
> #define ___GFP_NO_KSWAPD 0x400000u
> #define ___GFP_OTHER_NODE 0x800000u
> #define ___GFP_WRITE 0x1000000u
> +#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
> +#define ___GFP_KMEMCG 0x2000000u
> +#endif
>
> /*
>   * GFP bitmasks..
> @@ -88,13 +91,16 @@ struct vm_area_struct;
> #define __GFP_OTHER_NODE ((__force gfp_t)___GFP_OTHER_NODE) /* On behalf of other
node */
> #define __GFP_WRITE ((__force gfp_t)___GFP_WRITE) /* Allocator intends to dirty page */
>
> +#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
> +#define __GFP_KMEMCG ((__force gfp_t)___GFP_KMEMCG)/* Allocation comes from a
memcg-accounted resource */
> +#endif

Needs a space.

> /*
> * This may seem redundant, but it's a way of annotating false positives vs.
> * allocations that simply cannot be supported (e.g. page tables).
> */
> #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))
>
> /* This equals 0, but use constants in case they ever change */
