Subject: Re: [PATCH 06/11] memcg: kmem controller infrastructure
Posted by Tejun Heo on Mon, 25 Jun 2012 18:06:19 GMT
View Forum Message <> Reply to Message

Again, nits.

On Mon, Jun 25, 2012 at 06:15:23PM +0400, Glauber Costa wrote:
> +#define mem_cgroup_kmem_on 1
> +bool __mem_cgroup_new_kmem_page(gfp_t gfp, void *handle, int order);
> +void __mem_cgroup_commit_kmem_page(struct page *page, void *handle, int order);
> +void __mem_cgroup_free_kmem_page(struct page *page, int order);
> +#define is_kmem_tracked_alloc (gfp & __GFP_KMEMCG)

Ugh... please do the following instead.

static inline bool is_kmem_tracked_alloc(gfp_t gfp)
{
 return gfp & __GFP_KMEMCG;
}

>  #else
>  static inline void sock_update_memcg(struct sock *sk)
>  {
> @@ -416,6 +423,43 @@ static inline void sock_update_memcg(struct sock *sk)
>  static inline void sock_release_memcg(struct sock *sk)
>  {
>  }
> +
> +#define mem_cgroup_kmem_on 0
> +#define __mem_cgroup_new_kmem_page(a, b, c) false
> +#define __mem_cgroup_free_kmem_page(a,b )
> +#define __mem_cgroup_commit_kmem_page(a, b, c)
> +#define is_kmem_tracked_alloc (false)

I would prefer static inlines here too.  It's a bit more code in the
header but leads to less surprises (e.g. arg evals w/ side effects or
compiler warning about unused vars) and makes it easier to avoid
cosmetic errors.

Thanks.

--
tejun