Subject: Re: [PATCH -mm 1/3] cgroup: use a hash table for css_set finding
Posted by KAMEZAWA Hiroyuki on Thu, 03 Apr 2008 07:24:59 GMT
View Forum Message <> Reply to Message

On Thu, 03 Apr 2008 13:52:43 +0800
Li Zefan <lizf@cn.fujitsu.com> wrote:

```
> +/* hash table for cgroup groups. This improves the performance to
> + * find an existing css_set */
> +#define CSS_SET_HASH_BITS 7
> +#define CSS_SET_TABLE_SIZE (1 << CSS_SET_HASH_BITS)
> +static struct hlist_head css_set_table[CSS_SET_TABLE_SIZE];
```

How above number is selected ?

```
> +static struct hlist_head *css_set_hash(struct cgroup_subsys_state *css[])
> +{
> + int i;
> + int index;
> + unsigned long tmp = 0UL;
> +
> + for (i = 0; i < CGROUP_SUBSYS_COUNT; i++)
> +  tmp += (unsigned long)css[i];
> +
```

maybe css[i]'s lower 2-3 bits will be ignored. because thery are always 0.

And I don't like "+" for hash. how about

```
==
 for (i = 0; i < CGROUP_SUBSYS_COUNT; i++)
 unsigned long x;
 x = (unsigned long)css[i];
 tmp = (x >> 16) ^ (x & 0xffff)
==
```

or some func, which uses full bits.

```
> + index = hash_long(tmp, CSS_SET_HASH_BITS);
> +
> + return &css_set_table[index];
> +}
> +
>  /* We don't maintain the lists running through each css_set to its
>   * task until after the first call to cgroup_iter_start(). This
>   * reduces the fork()/exit() overhead for people who have cgroups
> @@ -219,6 +240,7 @@ static int use_task_css_set_links;
>  static void unlink_css_set(struct css_set *cg)
>  {
>   write_lock(&css_set_lock);
> + hlist_del(&cg->hlist);
>   list_del(&cg->list);
>   css_set_count--;
```

This css_set_lock is worth to be rwlock ?
how about per hashline spinlock ? (but per-hashline seems overkill..)

Thanks,
-Kame